IMS Resource List Interoperability Information Model

Version 1.0 Final Specification

Date Issued: 08 July 2004

IPR and Distribution Notices

Document Name: IMS Resource List Interoperability Information Model
Date: 08 July 2004

Table of Contents

1. Introduction

This section is not normative.

1.1 Specification Overview

The Resource List Interoperability (RLI) specification details how structured meta-data can be exchanged between systems that store and expose resources for the purpose of creating resource lists and those that gather and organize those resource lists for educational or training purposes. A typical example of such a resource list is a reading list.

The specification is based on an abstract service behaviors and data model that describes in generalized terms a resource at the item level, a collection of these resources (i.e., a list) and the behaviors associated with a resource list management service. The data model is then bound or expressed in XML, combining elements that map to subsets of key standards, including the IEEE-LOM (Learning Object Metadata), ISO 690-2 for bibliographic citations, and NISO's OpenURL to describe the resource items and aggregated resource list. The abstract service interface is bound to web services expressed as WSDL. The IMS Content Packaging specification wraps the resource list to enable transfer between systems. Because the data model is generalized, other bindings may be (and it is expected, will be) added to future releases of the specification.

The IMS Learning Resource Meta-data specification (now the IEEE-1484.12.1 LOM standard) was developed to provide item-level descriptions for the range of instructional materials that might be used in a virtual learning environment or course management system, from simple static HTML content to technically sophisticated simulations. While the LOM anticipates a broad economy of complex, self-contained e-learning objects, in the higher education realm much of the digital course "content" often categorized loosely as "course resources" parallels fairly traditional collections of supplementary course reading materials. These course resource lists frequently include links to freely available content on the Web and other materials that may not have been formally published, and may also include bibliographic records from OPACs (Online Public Access Catalog) and content licensed by academic libraries from publishers and aggregators.

Naturally, course reserves or course-specific collections of resources have long been and continue to be maintained and organized within academic libraries. The intersection of the library reserves function and the Web as a content distribution mechanism has given rise to the notion of "e-reserves." Integrated Library System (ILS) vendors have developed web-based software tool suites that integrate OPAC, e-reserves, federated searching and personal bibliographic management capabilities. At the same time, e-learning systems provide another logical context and dissemination point for course-specific collections of resources.

Course Resource Lists, then, may be maintained in both library and course management systems. Yet, while instructors are able to add citations and links to readings or post other supplementary resources directly into the course environment, they usually do so in an ad hoc fashion not much different from the manner one would post readings to any free form website. Furthermore, multiple segregated systems create a variety of unnecessarily disparate institutional work flows, ranging from the creation and publication of lists culled from library and other sources by the instructor (as just described), to the complete processing of a course reserves list by the library based on skeletal information the instructor has provided. The variety of ways in which Resource Lists can be created make it difficult for the various systems or tools that harvest, associate and store the Resource Lists to exchange and consolidate these heterogeneous aggregations. The RLI specification begins to address these integration issues by enabling the ready exchange of structured meta-data between e-learning tools and various related applications, including OPACs, reserves modules, or other third-party tools.

The Information Model defines the exchange and maintenance of resource lists among systems through an abstract service interface. The interface is divided between core (Create, Read, Replace, and Delete) and complex behaviors (Assign and De-assign). The latter enable the association of resource lists created in one system or application with groups or courses in another. Multiple implementations of the service interface are possible, although the RLI Binding document defines a WSDL binding.

The initial release of the RLI specification addresses the issue of how to organize, describe and exchange traditional lists of course resources (such as bibliographies). The data model comprises a minimal set of elements for citing print publications. Nevertheless, given the simplicity, generality and design of the RLI model, materials in other media can be adequately described and included within resource lists as currently defined. Just as importantly, the model is designed so that it may be easily extended to specify different resource types in subsequent iterations.

In the future, it is highly unlikely that any one standard will be adopted to describe bibliographic resources, much less digital resources in general. Appendix D includes a mapping of the relevant LOM elements to major bibliographic citation schemas used in the library and publishing worlds. While the RLI specification represents a first step towards a larger resource collections framework, the relatively simple abstract data model on which it is based should ensure interoperability with other emerging frameworks such as METS.

1.1.1 Rationale

This Resource List Interoperability (RLI) specification:

- Enables the creation of tools for the generation and exchange of resource lists according to a standard format.
- Allows for incorporation of resource list data into other e-learning tools, so that future efforts to integrate enterprise learning and library course reserves systems will not need to rely on proprietary solutions, reducing costs to vendors and customers.
- Leverages federated searching tools to capture records for resource lists. This raises the profile of costly and often underutilized library-licensed resources.
- Establishes the potential for a unified resource list creation workflow.

1.1.2 Background and History

The IMS Digital Libraries special interest group identified the organization of course resources as an existing intersection between the library and e-learning domains that had not, as of yet, been addressed in any standardized fashion. It was also recognized as an area that represented a potential quick win—through the development of an interoperability specification—for the broader effort to bring two related domains into closer harmony.

1.2 Components of the Specification

This specification consists of the documents listed below.

1.2.1 RLI Information Model

The RLI Information Model (this document) [RLI, 04a] – Describes the core aspects of the specification and contains parts that are normative for any binding claiming to use this Information Model. It contains details of: semantics, structure, data types, value spaces, multiplicity, and obligation (i.e., whether mandatory or optional).

1.2.2 RLI Binding

The XML/WSDL Binding [RLI, 04b] – Describes a binding of the Information Model using a Web Services infrastructure based upon XML and a SOAP/HTTP transport mechanism. There are many possible bindings of the Information Model but at the current time the only specified binding is based upon XML and WSDL.

1.2.3 RLI Best Practice and Implementation Guide

The Best Practice and Implementation Guide [RLI, 04c] – Provides non-normative guidance on application of the Information Model and WSDL Binding. This includes reference to existing practice in handling information that this specification seeks to support and guidance on practices that will promote interoperability and durability. It also includes examples to illustrate how the conceptual framework maps to practical uses and to identify the relationship between this specification and related IMS specifications. Implementers are encouraged, but not required, to follow guidance in this part of the specification.

1.2.4 RLI Conformance Requirements

The Conformance Requirements [RLI, 04d] – Defines the conformance criteria that must be followed by systems that wish to claim compliance to the Resource List Interoperability Information Model.

1.3 Scope

The following are out of scope for this specification:

- In this specification, the definition of persistent locators is out of scope and not addressed. The RLI specification does state, however, when locators are needed and what meta-data is required for the construction of known, standard resolver schemes such as OpenURL and the Digital Object Identifier.
- Authorization/Access Management. Ensuring authorized access to the resource list creation functionality and the items contained within a resource list is outside the scope of this specification. The RLI specification does provide a meta-data element in which access rights or other intellectual property declarations associated with the resource list can be stated in human readable form for display or other, non-machine actionable purposes.
- Association of Course Identifiers. Dynamically associating system course identifiers to resource lists is critical to the practical working of the RLI spec. The definition of course identifiers is out of scope.
- Update transactions. Lists already created that have been modified must in their entirety replace any previous versions (Destructive Update). Update functionality may be formalized in a later phase of specification development.

1.4 Structure of this Document

The structure of this document is:

2. Resource List Manager Service Description	The description of the overall structure and operation of the RLI Manager Service;
3. Behavioral Model	The definition of the operations of RLI Manager Service and the behaviors supported by the service;
4. RLI Data Model	The definition of the data models for RLI and related data structures;
5. Extending the Service	Identification of the ways in which RLI can be extended both in terms of the addition of new constituent services and proprietary extensions to a service;
Appendix A – Common Components	Identification of the common data structures and services that are used by RLI;
Appendix B – Service Status Codes	A summary list of the status codes, and their causes, that can be returned by each of the operations forming the RLI Service;
Appendix C – Sources of Input	A list of resources used to develop this model;
Appendix D – Comparison of Descriptive Meta-data Schemes for Citation and Resolution	A table displaying the relationships between the meta-data schemes for citation and resolution.

1.5 Nomenclature

ADL	Advanced Distributed Learning
API	Application Programming Interface
IAF	IMS Abstract Framework
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
ILS	Integrated Library System
ISO	International Organization for Standardization

LOM	Learning Object Metadata (usually called "IEEE LOM")
LTSC	Learning Technology Standards Committee
METS	Metadata Encoding and Transmission Standard
MODS	Metadata Object Description Schema
OPAC	Online Public Access Catalog
RFC	Request for Comment (usually used in "IETF RFC ####")
RLI	Resource List Interoperability
SCORM	Sharable Content Object Reference Model
W3C	World Wide Web Consortium
WSDL	Web Services Description Language
XML	Extensible Mark-up Language
XSD	XML Schema Definition

1.6 References

[RLI, 04b] *IMS Resource List Interoperability XML/WSDL Binding v1.0*, A.Jackl, IMS Global Learning Consortium, Inc., July 2004.

[RLI, 04c] *IMS Resources List Interoperability Best Practice and Implementation Guide v1.0*, A.Jackl, IMS Global Learning Consortium, Inc., July 2004.

[RLI, 04d] *IMS Resources List Interoperability Conformance v1.0*, M.Maljkovic, IMS Global Learning Consortium, Inc., July 2004.

[DOI] ANSI/NISO Z39.84 - Syntax for the Digital Object Identifier

[IEEE LOM] IEEE 1484-12:2002, Standard for Learning Object Metadata, http://ltsc.ieee.org

[IMSBUND] *Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications, v1.0*, B.Olivier, M.McKell, IMS Global Learning Consortium, Inc., August 2001.

[IMSCP] *IMS Content Packaging, v1.1.3*, IMS Global Learning Consortium, Inc., June 2003.

[IMSES, 04] *IMS Enterprise Services v1.0*, IMS Global Learning Consortium, Inc., June 2004.

[IMSPLID] *IMS Persistent, Location-Independent, Resource Identifier Implementation Handbook v1.0*, IMS Global Learning Consortium, Inc., April 2001.

[ISO 690-2] ISO 690-2 Standard Information and documentation -- Bibliographic references -- Part 2: Electronic documents or parts thereof

[METS] Metadata Encoding and Transmission Standard, Version 1.3

[OpenURL] ANSI/NISO Z39.88-200x - The OpenURL Framework for Context-Sensitive Services

[PRISM] PRISM v 1.2 Specifications (Publishing Requirements for Industry Standard Metadata

[RFC 1766] Tags for the Identification of Languages, http://www.ietf.org/rfc/rfc1766.txt

[RFC 2119] Key words for use in RFCs to Indicate Requirement Levels, http://www.ietf.org/rfc/rfc2119.txt

[RFC 2396] Uniform Resource Identifiers (URI): Generic Syntax, http://www.ietf.org/rfc/rfc2396.txt

[ISO 639-2, part 2] ISO 639 part 2, Codes for the representation of names of languages -- Part 2: Alpha-3 code, http://www.loc.gov/standards/iso639-2/langhome.html

[ISO 11404] ISO 11404, Language-independent Datatypes, http://www.iso.ch/cate/d19346.html

2. Resource List Manager Service Description

2.1 An Abstract Representation

It is important to remember that this document describes the underlying information model in terms of the abstract API. The manner in which this abstract representation is visualized is not intended to dictate the implementation form of a Resource List Management System. The breakdown of the service into its interface classes is a convenient way to document the set of behaviors. The internal organization of an implementation of the full abstract API is beyond the scope of this specification. The only constraint is that the external behavior of the abstract API complies with this specification. This means that a .NET, Java, etc. physical implementation of this abstract API does not have to represent the functionality using the same breakdown of operations/methods. This physical implementation is not subject to the conformance specification.

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

2.2 Resource List Manager Service Architecture Model

Online Public Access Catalog, Digital Library, Course Management and other third-party systems should facilitate the exchange of Resource Lists. This exchange will be achieved through Export/Import or integrated Send/Receive Actions as defined in the services of the Resource List Manager. Furthermore, updates to the Resource List made in the originating system should be updated in the other systems that have imported/received that particular Resource List.

The basic architectural model for the Resource List Manager Services specification is shown in Figure 2.1.

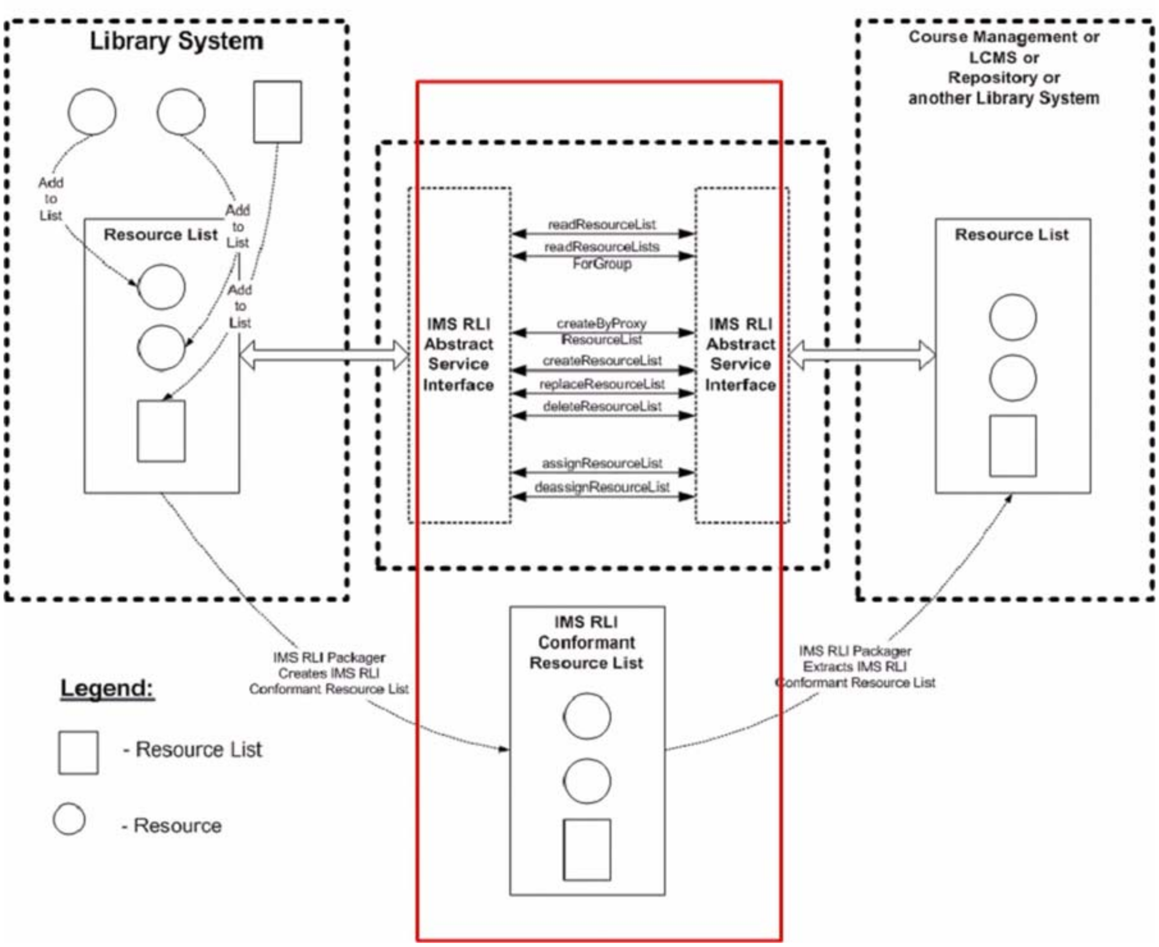

Library System
Add to List
Add to List
Add to List
Resource List
Course Management or LCMS or Repository or another Library System
Resource List
IMS RLI Abstract Service Interface
IMS RLI Abstract Service Interface
readResourceList
readResourceLists ForGroup
createByProxy ResourceList
createResourceList
replaceResourceList
deleteResourceList
assignResourceList
deassignResourceList
IMS RLI Conformant Resource List
IMS RLI Packager Creates IMS RLI Conformant Resource List
IMS RLI Packager Extracts IMS RLI Conformant Resource List
Legend:
- Resource List
- Resource

Figure 2.1 Resource List Manager Service Architecture Model.

In this architecture the scope of the IMS Resource List Manager Services specification is shown as within the red box. The scope of the interoperability is the data and behavioral models of the objects being exchanged.

The IMS Resource List Manager Services specification contains a very simple abstract messaging model that is assumed to underlie the exchange of the data between the communicating Resource List Manager systems. The basic data exchange mechanism is shown in Figure 2.2 in which the ‘source’ and ‘target’ Resource List Manager systems exchange ‘messages’ using a ‘Request/Response’ transaction. This abstract messaging representation is adopted because the Resource List Manager System architecture is based upon a loosely coupled system and the primary IMS binding of this information model is based upon the exchange of XML documents/messages.

It is important to remember that the structure of the exchanged information has NO bearing on how the same information is contained within the ‘source’ and ‘target’ Enterprise systems. It is simply a representation of the data used to facilitate exchange with other parties (the information that crosses the dotted line in Figure 2.2).

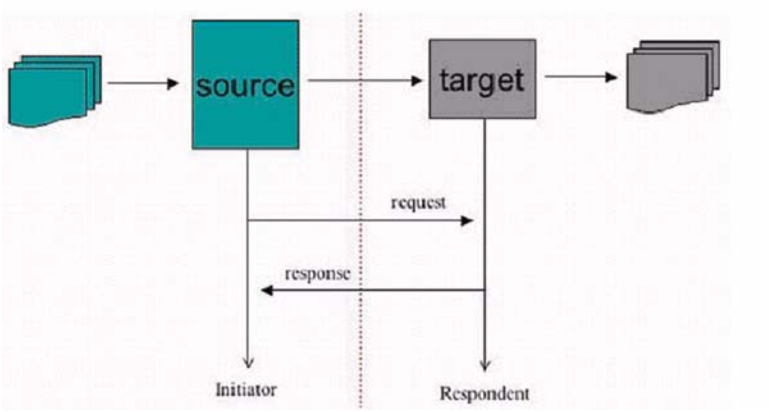


Figure 2.2 Resource List Manager Service abstract information exchange model.

The behavioral descriptions will describe:

- The data that is to be exchanged between the communicating Resource List Manager Systems;
- The operations that can be invoked to support the exchange of the data between the communicating Resource List Manager Systems;
- The permitted sequence of operations that can be used to support the exchange of data between the communicating Resource List Manager Systems.

2.3 Resource List Objects

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

2.4 Synchronous Services

Within the context of the Resource List Manager Services, this specification only deals with synchronous services. Asynchronous services are out of scope for the specification.

Synchronous services are defined as services in which the source service is blocked until the final response from the target service is received.

The abstract-API does not differentiate between synchronous and asynchronous services.

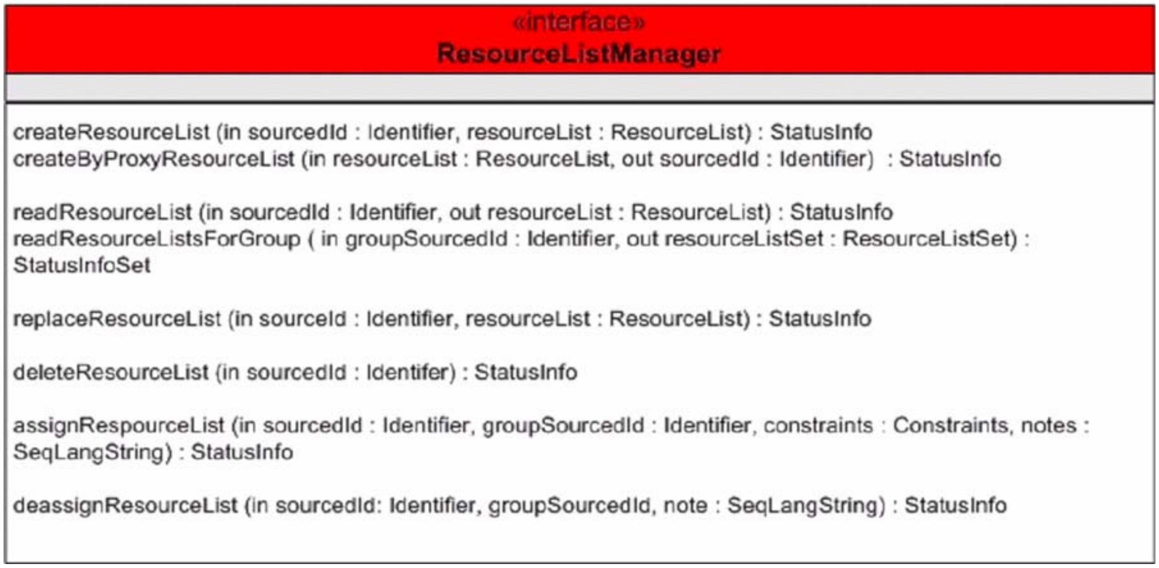
3. Behavioral Model

3.1 Service Definition - Resource List Manager Interface Class Description

3.1.1 Structure

The ResourceListManager interface class is used to model the service responsible for manipulating information about resource lists and describes the operations that are permitted on a single ‘resourceList’ object as shown in Figure 3.1.

These operations are based upon the classic Create/Read/Update/Delete (CRUD) model with variations defined to differentiate subtleties of functionality. The interface stereotype indicates that there are no attributes for this class.



«interface»
ResourceListManager

createResourceList (in sourcedId : Identifier, resourceList : ResourceList) : StatusInfo
createByProxyResourceList (in resourceList : ResourceList, out sourcedId : Identifier) : StatusInfo

readResourceList (in sourcedId : Identifier, out resourceList : ResourceList) : StatusInfo
readResourceListsForGroup (in groupSourcedId : Identifier, out resourceListSet : ResourceListSet) : StatusInfoSet

replaceResourceList (in sourceId : Identifier, resourceList : ResourceList) : StatusInfo

deleteResourceList (in sourcedId : Identifer) : StatusInfo

assignRespourceList (in sourcedId : Identifier, groupSourcedId : Identifier, constraints : Constraints, notes : SeqLangString) : StatusInfo

deassignResourceList (in sourcedId: Identifier, groupSourcedId, note : SeqLangString) : StatusInfo

Figure 3.1 ResourceListManager interface class diagram.

Resource List Manager operations use the following data types and common classes:

- resourceList – the data structure for the resourceList.
- identifier – the container for the unique identifier of the ‘resourceList’ object.
- constraints – the container for data that defines the association between a course and a resources list.
- resourceListSet – the container for the set of Resource Lists and constraints for a uniquely identified group.
- SeqLanString – the container for a set of language alternative versions of a string.
- StatusInfo – the container for the status information that describes the completion state of the operation.
- StatusInfoSet – the container for a set of status information that describes the completion state of the operation.

3.1.2 Operations

The core CRUD operations are summarized in Table 3.1.

Table 3.1 Summary of operations for Resource List Manager.

Operation	Description
createResourceList	Request the creation of a populated ‘resourceList’ on the target system, where the source system is responsible for the allocation of the identifier for the resourceList.
createByProxyResourceList	Request the creation of a populated ‘resourceList’ on the target system, where the target system is responsible for the allocation of the identifier for the resourceList.
readResourceList	Read the full contents of the identified ‘resourceList. The target must return all of the data it has for the identified ‘resourceList’.
readResourceListsforGroup	Read the full contents of the set of ‘resourceList’ associated with the identified “Group”.
replaceResourceList	Write new content into the identified ‘resourceList’ record. The target must write the new data into the ‘resourceList’ record. This is a destructive update of the original information.
deleteResourceList	Request the deletion of a ‘resourceList’. The Resource List and any associations between the Resource List and Groups are deleted.
assignResourceList	Request the target system associate the identified ‘resourceList’ with the identified “Group” and any constraints that apply to the association.
deassignResourceList	Request the target system remove the association between the identified ‘resourceList’ and the identified “Group”.

3.1.2.1 createResourceList

Name:	createResourceList
Return Function Parameter:	*StatusInfo* – the status of the creation request. The permitted status codes are defined in Appendix B.
Supplied (in) Parameters:	*sourcedId:Identifier* – the Identifier allocated by the source system. This is the identifier that must also be assigned within the target system. *resourceList: ResourceList* – the Resource List data that is to be stored in the new record.
Returned (out) Parameters:	None.
Behavior:	When the source issues the ‘createResourceList’ request the target is instructed to create the populated resourceList data structure and to allocate that structure the ‘sourcedId’ passed by the source. If the supplied ‘sourcedId’ has already been allocated to another object then the request is rejected and the appropriate failure code is returned.
Notes:	This request contains the initial content for the resourceList record. More content can be added/replaced using the ‘replaceResourceList’ requests.

The associated interaction sequence diagram is shown in Figure 3.2. The ‘TriggerAction’ results in the ‘Source System’ issuing the ‘createResourceList ()’ request. At some time later the ‘Target System’ responds.

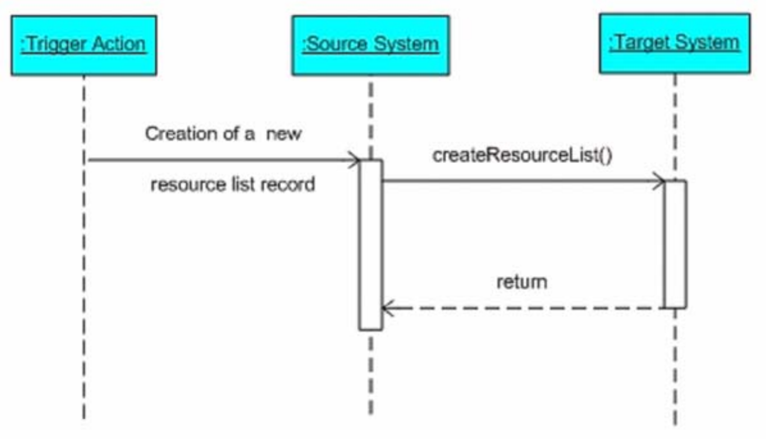


Figure 3.2 The 'createResourceList' operation sequence diagram.

3.1.2.2 createByProxyResourceList ()

Name:	createByProxyResourceList
Return Function Parameter:	*StatusInfo* – the status of the creation request. The permitted status codes are defined in Appendix B.
Supplied (in) Parameters:	*resourceList:ResourceList* – the resourceList data that is to be stored in the new record.
Returned (out) Parameters:	*sourcedId:Identifier* – the identifier allocated by the target to the newly created 'resourceList' record.
Behavior:	When the source issues the 'createByProxyResourceList' request the target is instructed to create the populated resourceList record and to allocate that record a unique 'sourcedId'.
Notes:	This request contains the initial content for the resourceList record. More content can be added/replaced using the 'replaceResourceList' requests.

The associated interaction sequence diagram is shown in Figure 3.3. The 'TriggerAction' results in the 'Source System' issuing the 'createByProxyResourceList ()' request. At some time later the 'Target System' responds.

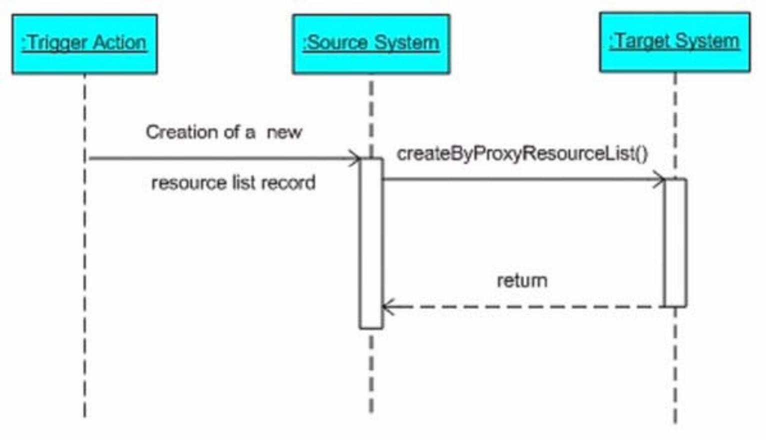


Figure 3.3 The 'createByProxyResourceList' operation sequence diagram.

3.1.2.3 readResourceList ()

Name:	readResourceList
Return Function Parameter:	*StatusInfo* – the status of the read request. The permitted status codes are given in Appendix B.
Supplied (in) Parameters:	*sourcedId:Identifier* – the identifier of the 'resourceList' object to be read.
Returned (out) Parameters:	*resourceList:ResourceList* – the resourceList data that is read from the record.
Behavior:	When the source issues the 'readResourceList' request the target is charged with retrieving the identified record from its database and returning this data to the source. The target is responsible for ensuring that the record contains valid data. If the object identified by the supplied 'id' cannot be located then the request is rejected and the appropriate failure code is returned.
Notes:	The returned resourceList record can only be trusted if the corresponding status code is 'success'.

The associated interaction sequence diagram is shown in Figure 3.4. The 'TriggerAction' results in the 'Source System' issuing the 'readResourceList ()' request. At some time later the 'Target System' responds.

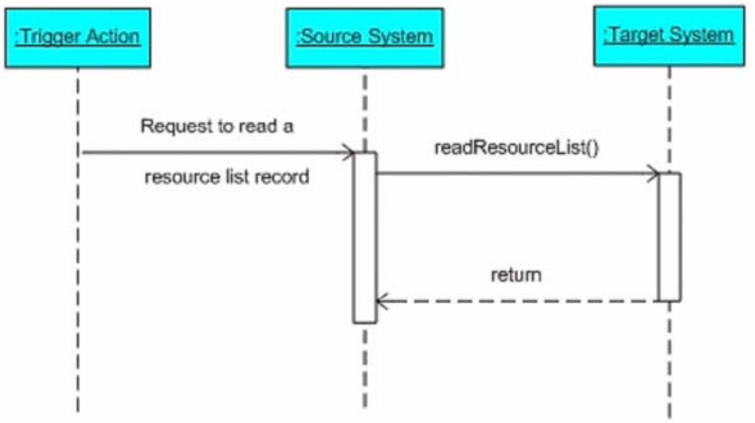


Figure 3.4 The 'readResourceList' operation sequence diagram.

3.1.2.4 readResourceListsForGroup ()

Name:	readResourceListsForGroup
Return Function Parameter:	*StatusInfoSet* – the status for each of the read requests. The permitted status codes (one of these must be returned for each 'group' object identified) are given in Appendix B.
Supplied (in) Parameters:	*groupSourcedId:Identifier* – the identifier of the Group to which the resourceList is being assigned.
Returned (out) Parameters:	*resourceListSet:ResourceLisSet* – the set of resourceList data that is to be read.

Behavior:	When the source issues the ‘readResourceListsForGroup’ request the target is charged with retrieving all the ‘resourceList’ records that are associated with the identified ‘group’ record, i.e., there assign relationships between the ‘group’ record identified and the ‘resourceList’ records returned. If the object identified by the supplied ‘groupSourcedId’ cannot be located then the request is rejected and the appropriate failure code is returned for this part of the operation. The target is responsible for ensuring that the records contain valid data. The target should attempt to successfully complete as much of the request as possible.
Notes:	resourceListSet records are only present if the corresponding status code is ‘success’.

The associated interaction sequence diagram is shown in Figure 3.5. A set of ‘TriggerAction’ events results in the ‘Source System’ issuing the ‘readResourceListsForGroup()’ request. At some time later the ‘Target System’ responds.

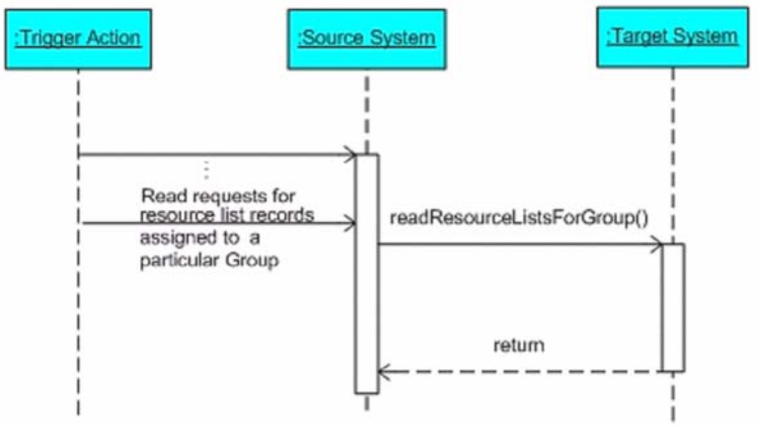

:Trigger Action
:Source System
:Target System
Read requests for resource list records assigned to a particular Group
readResourceListsForGroup()
return

Figure 3.5 The ‘readResourceListsForGroup’ operation sequence diagram.

3.1.2.5 replaceResourceList ()

Name:	replaceResourceList
Return Function Parameter:	*statusInfo:StatusInfo* – the status of the update request. The permitted status codes are given in Appendix B.
Supplied (in) Parameters:	*sourcedId:Identifier* – the identifier of the ‘resourceList’ object to be updated. *resourceList:ResourceList* – the resourceList data that is to be stored in the record.
Returned (out) Parameters:	None.
Behavior:	When the source issues the ‘replaceResourceList’ request the target is charged with writing the supplied information into the identified record. If any part of the write fails e.g. due to partial invalid data then the whole request is rejected and the record is left in its original state. This is a destructive write-over update operation of the entire ‘resourceList’ object. This is equivalent to a ‘createResourceList but for an object that already exists. If the object identified by the supplied ‘sourcedId’ cannot be located then the request is rejected and the appropriate failure code is returned.
Notes:	The source is responsible for determining the reason of the failure.

The associated interaction sequence diagram is shown in Figure 3.6. The ‘TriggerAction’ results in the ‘Source System’ issuing the ‘replaceResourceList ()’ request. At some time later the ‘Target System’ responds.

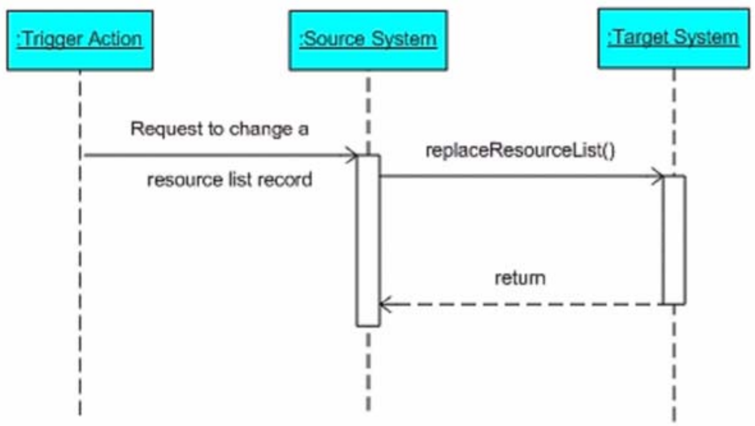


Figure 3.6 The ‘replaceResourceList’ operation sequence diagram.

3.1.2.6 deleteResourceList ()

Name:	deleteResourceList
Return Function Parameter:	*StatusInfo* – the status of the creation request. The permitted status codes are defined in Appendix B.
Supplied (in) Parameters:	*sourcedId:Identifier* – the identifier to be used by the target to identify the ‘resourceList’ object.
Returned (out) Parameters:	None.
Behavior:	When the source issues the ‘deleteResourceList’ request the target is instructed to delete the identified resourceList record and to any associations between the resourceList and any Groups. This is a hard cascaded delete from which there is no recovery. If the object identified by the supplied ‘sourcedId’ cannot be located then the request is rejected and the appropriate failure code is returned.
Notes:	Deletion of the ‘resourceList’ record does not necessarily result in the destruction of the data within the server. The true state of the data in the target is unknown.

The associated interaction sequence diagram is shown in Figure 3.7. The ‘TriggerAction’ results in the ‘Source System’ issuing the ‘deleteResourceList ()’ request. At some time later the ‘Target System’ responds.

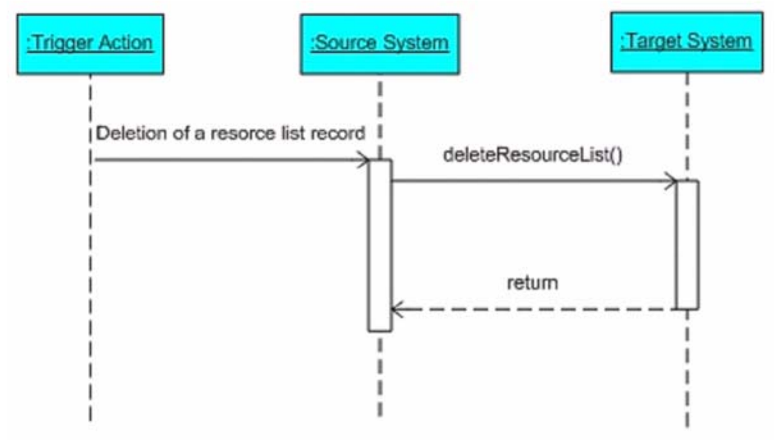


Figure 3.7 The ‘deleteResourceList’ operation sequence diagram.

3.1.2.7 assignResourceList ()

Name:	assignResourceList
Return Function Parameter:	*statusInfo:StatusInfo* – the status of the update request. The permitted status codes are given in Appendix B.
Supplied (in) Parameters:	*sourcedId:Identifier* – the identifier of the ‘resourceList’ object to be updated. *groupSourcedId:Identifier* – the identifier of the Group to which the resourceList is being assigned. *constraints:Constraints* – any constraints temporal, conditional, etc. being placed on the association. *note:SeqLangString* – a field for any notes on the association or assignment.
Returned (out) Parameters:	None.
Behavior:	When the Source issues the assignResourceList Request the target system is charged to associate the identified “resourceList” with the identified “Group” and any temporal, conditional etc. constraints being placed on the association If the object identified by the ‘sourcedId’ cannot be located then the request is rejected and the appropriate failure code is returned. If the group identified by the groupSourcedId can not be located the target may choose to treat the groupSourcedId as new data, or may return the appropriate failure code. The behavior implemented depends on out of band agreements between the interoperating systems. The source supplies the ‘groupSourcedId’. Management of the ‘groupSourcedId’s existence and validation is out of scope for this specification.
Notes:	The source is responsible for determining the reason for any failure.

The associated interaction sequence diagram is shown in Figure 3.8. The ‘TriggerAction’ results in the ‘Source System’ issuing the ‘assignResourceList ()’ request. At some time later the ‘Target System’ responds.

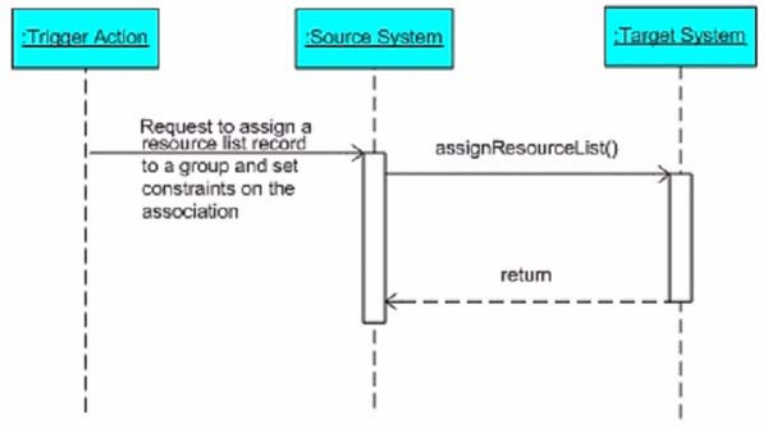

:Trigger Action
:Source System
:Target System
Request to assign a resource list record to a group and set constraints on the association
assignResourceList()
return

Figure 3.8 The ‘deleteResourceList’ operation sequence diagram.

3.1.2.8 deassignResourceList ()

Name:	deassignResourceList
Return Function Parameter:	*statusInfo:StatusInfo* – the status of the update request. The permitted status codes are given in Appendix B.
Supplied (in) Parameters:	*sourcedId:Identifier* – the identifier of the ‘resourceList’ object to be updated. *groupSourcedId:Identifier* – the identifier of the Group to which the resourceList is being assigned. *note:SeqLangString* – a field for any notes on the association or assignment.
Returned (out) Parameters:	None.
Behavior:	When the source issues the deassignResourceRequest the target system is charged with remove the association between the identified “resourceList” and the identified “Group”. If the association identified by the ‘sourcedId’ ‘groupSourcedId’ cannot be located then the request is rejected and the appropriate failure code is returned.
Notes:	The source is responsible for determining the reason for any failure.

The associated interaction sequence diagram is shown in Figure 3.9. The ‘TriggerAction’ results in the ‘Source System’ issuing the ‘assignResourceList ()’ request. At some time later the ‘Target System’ responds.

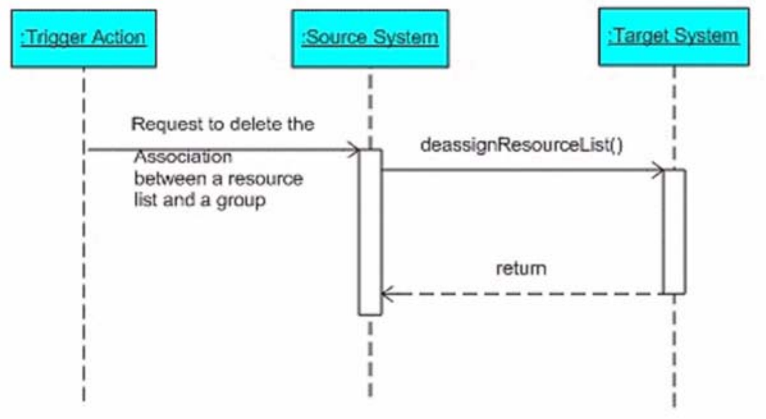


Figure 3.9 The ‘deassignResourceList’ operation sequence diagram.

3.2 Permitted State Sequence

The permitted state activity on an object is shown in Figure 3.10. This state diagram has four states:

- ‘No Object’ state – no resourceList object exits with a particular id;
- ‘Object with Target-created id’ – a resourceList object exists with the id allocated by the Target system;
- ‘Object with Source-created id’ – a resourceList object exists with the id allocated by the Source system.
- ‘Object with associations’ – a resourceList object exists that has been assigned to a group.

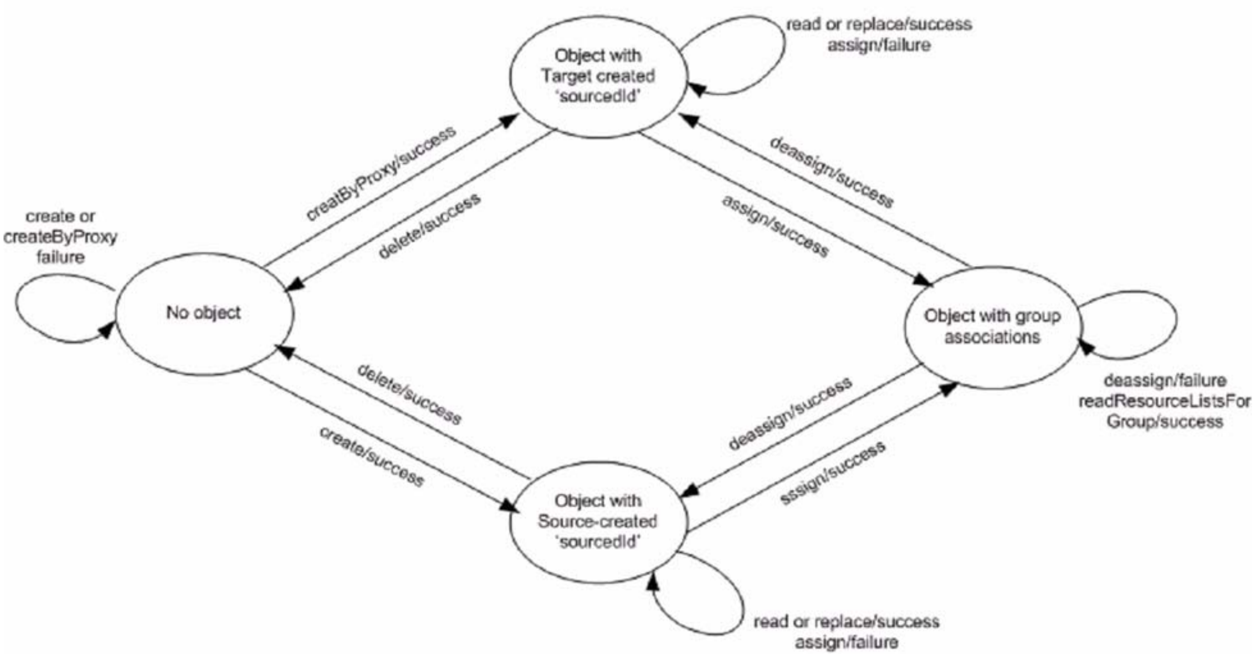


Figure 3.10 The state diagram for the behaviors on a resourceList object.

The start state is ‘No Object’ i.e. the resourceList record has not yet been created. Only the ‘createResourceList ()’ and ‘createByProxy()’ operations are possible. Once the resourceList object has been created then it persists until a successful ‘deleteResourceList ()’ operation is completed. The ‘createResourceList()’ operation takes the system into the ‘Object with Source-created id’ state whereas the ‘createByProxyResourceList ()’ takes the system into the ‘Object with Target-created id’ state.

Once the system is in the ‘Object with Source-created id’ or the ‘Object with Target-created id’ states then the ‘readResourceList()’ and ‘replaceResourceList ()’ operations are now possible.

The ‘assignResourceList()’ operation takes the system into the ‘Object with group associations’ state. Once the system is in the ‘Object with group associations’ state, then the ‘readResourceListsForGroup’ operation is possible.

4. RLI Data Model

The Data Model in this specification is non-normative. It is an informative abstraction. Conformance and normalization is described in the Binding and Conformance components of this specification.

4.1 resourceList

The resource list and its associations are is shown in Figure 4.1. This representation describes the data model that must be supported by the end systems.

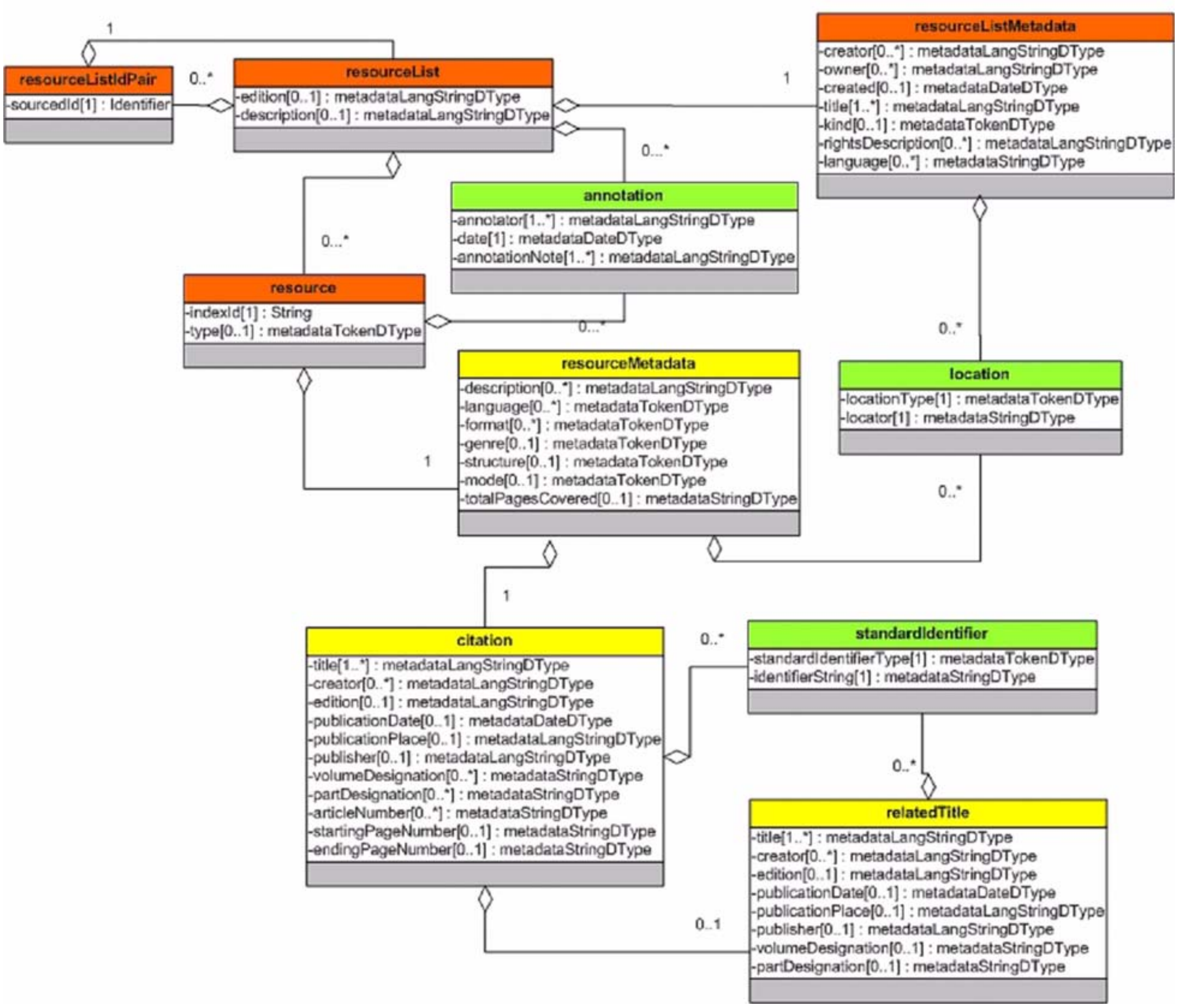

resourceListIdPair
-sourcedId[1] : Identifier
resourceList
-edition[0..1] : metadataLangStringDType
-description[0..1] : metadataLangStringDType
resourceListMetadata
-creator[0..*] : metadataLangStringDType
-owner[0..*] : metadataLangStringDType
-created[0..1] : metadataDateDType
-title[1..*] : metadataLangStringDType
-kind[0..1] : metadataTokenDType
-rightsDescription[0..*] : metadataLangStringDType
-language[0..*] : metadataStringDType
annotation
-annotator[1..*] : metadataLangStringDType
-date[1] : metadataDateDType
-annotationNote[1..*] : metadataLangStringDType
resource
-indexId[1] : String
-type[0..1] : metadataTokenDType
resourceMetadata
-description[0..*] : metadataLangStringDType
-language[0..*] : metadataTokenDType
-format[0..*] : metadataTokenDType
-genre[0..1] : metadataTokenDType
-structure[0..1] : metadataTokenDType
-mode[0..1] : metadataTokenDType
-totalPagesCovered[0..1] : metadataStringDType
location
-locationType[1] : metadataTokenDType
-locator[1] : metadataStringDType
citation
-title[1..*] : metadataLangStringDType
-creator[0..*] : metadataLangStringDType
-edition[0..1] : metadataLangStringDType
-publicationDate[0..1] : metadataDateDType
-publicationPlace[0..1] : metadataLangStringDType
-publisher[0..1] : metadataLangStringDType
-volumeDesignation[0..*] : metadataStringDType
-partDesignation[0..*] : metadataStringDType
-articleNumber[0..*] : metadataStringDType
-startingPageNumber[0..1] : metadataStringDType
-endingPageNumber[0..1] : metadataStringDType
standardIdentifier
-standardIdentifierType[1] : metadataTokenDType
-identifierString[1] : metadataStringDType
relatedTitle
-title[1..*] : metadataLangStringDType
-creator[0..*] : metadataLangStringDType
-edition[0..1] : metadataLangStringDType
-publicationDate[0..1] : metadataDateDType
-publicationPlace[0..1] : metadataLangStringDType
-publisher[0..1] : metadataLangStringDType
-volumeDesignation[0..1] : metadataStringDType
-partDesignation[0..1] : metadataStringDType
1
0..*
1
0...*
0...*
0...*
0..*
1
0..*
1
0..*
0..*
0..1

Figure 4.1 resourceList Class Diagram.

A resource list is an aggregation of content resources associated with one or more courses or learning objects.

Resource lists may be subsumed under other Resource Lists.

A resource list always exists in association with an identifier ‘sourcedId’, either in the input parameters to a service or, in the resource list ID Pair object. The identifier is unique among the systems being integrated.

4.1.1 Attributes

The set of attributes for the resourceList class is summarized in Table 4.1.

Table 4.1 Summary of attributes for the resourceList Class.

Name	Type	Mult	Description
description	MetadataLangStringDType	0..1	A textual explanation of the Resource List.
edition	MetadataLangStringDType	0..1	This is the version information associated with a particular instance of a Resource List.

4.1.2 Associations

The set of associations for the Resource List Class are summarized in Table 4.2.

Table 4.2 Summary of associations for the resourceList Class.

Association Class Name	Mult	Description
resourceListIDPair	0..*	A Resource List may subsume 0 to many Resource Lists. Each Resource List subsumed must be associated with an identifier that is unique among the systems being integrated
resourceListMetadata	1	The categories of elements and sub-elements that must be defined within the schema to describe a Resource List itself.
resource	0..*	The details of a resource associated with the Resource List.
annotation	0..*	Authorized end users may annotate any Resource List. This is considered to be a very basic statement and not a complex structured description

4.1.2.1 resourceListMetadata

The resourceListMetadata Class attributes are described in Table 4.3.

Table 4.3 Attribute Definitions for the resourceListMetadata Class.

Name	Type	Mult	Description
creator	MetadataLangStringDType	0..*	Person or organization primarily responsible for creating the intellectual content of the Resource List.
owner	MetadataLangStringDType	0..*	Person or organization primarily responsible for establishing distribution and use permissions associated with Resource List.
created	MetadataDateDType	0..1	DateTime at which the Resource List was completed and made available.
title	MetadataLangStringDType	1..*	Name given to Resource List by Creator or Owner.
kind	MetadataTokenDType		Genre for the collected set of resources (i.e., a Resource List) created for educational purposes. The value of the attribute is “resource list”
language	MetadataTokenDType	0..*	Primary language in which the Resource List was created.
rightsDescription	MetadataLangStringDType	1..*	Simple, human readable statement about rights related to Resource List as desired by RL owner.

4.1.2.2 Associations

The set of associations for the resourceListMetadata Class are summarized in Table 4.4.

Table 4.4 Summary of associations for the resourceListMetadata Class.

Association Class Name	Mult	Description
location	0..*	Describes storage locations of a Resource List.
standardIdentifier	0..*	Standard number associated with the Resource List as a published item, e.g., DOI.

4.1.3 resourceListIDPair

The ResourceListIDPair Class attributes are described in Table 4.5.

Table 4.5 Attribute Definitions for the ResourceListIDPair Class.

Name	Description
sourcedID	An identifier for a Resource List that is unique among the systems being integrated.

4.1.3.1 Associations

The set of associations for the Resource List ID Pair Class are summarized in Table 4.6.

Table 4.6 Summary of associations for the ResourceListIDPair Class.

Association Class Name	Mult	Description
resourceList	1	The details of the resource list

4.2 resource

The resource Class attributes are described in Table 4.7.

Table 4.7 Attribute definitions for the resource Class.

Name	Type	Mult	Description
indexId	string	1	This assigns a unique identifier to a resource within a specific Resource List. A resource may appear in more than one Resource List.
type	MetadataTokenDType	0..1	Genre of a specific resource on a Resource List, such as bibliographic citation (to a book, online article, etc.)

4.2.1 Associations

The set of associations for the resource Class are summarized in Table 4.8.

Table 4.8 Summary of associations for the resource Class.

Association Class Name	Mult	Description
resourceMetadata	1	The categories of elements and sub-elements that must be defined within the schema to describe a resource itself

Association Class Name	Mult	Description
annotation	0..*	Authorized end users may annotate any resource. This is considered to be a very basic statement and not a complex structured description

4.2.1.1 resourceMetadata

The resourceMetadata Class attributes are described in Table 4.9.

Table 4.9 Attribute Definitions for the resourceMetadata Class.

Name	Type	Mult	Description
description	MetadataLangStringDType	0..1	String describing the use, context or other pertinent descriptive information not included elsewhere.
language	MetadataTokenDType	0..*	Primary language in which resource has been created.
format	MetadataTokenDType	0..*	Datatype or MIME type of resource using IANA media type vocabulary.
genre	MetadataTokenDType	0..1	Type of resource as described by controlled vocabulary for locator strategy used by information source, or learning management system, e.g., OpenURL or DOI
structure	MetadataTokenDType	0..1	Structural type of resource such as physical, digital, abstract, etc. Necessary if creating Digital Object Identifier as means for locating resource. See http://www.doi.org/handbook_2000/metadata.html
mode	MetadataTokenDType	0..1	Primary sensory mode in which the resource is intended to be perceived. Necessary if creating Digital Object Identifier as means for locating resource. See http://www.doi.org/handbook_2000/metadata.html.
totalPagesCovered	MetadataStringDType	0..1	Range of pages included in the resource, as described on the display representation of the resource. May be simple concatenation of beginning page and ending page, or other means for noting when page sequence is not consecutive.

4.2.1.1.1 Associations

The set of associations for the resourceMetadata Class are summarized in Table 4.10.

Table 4.10 Summary of associations for the ResourceMetadata Class.

Association Class Name	Mult	Description
citation	1	The minimum data required to cite a resource as a formal scholarly reference. This list of MD elements primarily focuses upon what is necessary to create a traditional bibliographic citation according to the ISO 690-2, standard Information and documentation -- Bibliographic references -- Part 2: Electronic documents or parts thereof. These meta-data elements should also meet the PRISM v 1.2 Specifications (Publishing Requirements for Industry Standard Metadata).

Association Class Name	Mult	Description
Location	0..*	Describes storage locations of a resource. Solutions to Persistent locators have been declared out of scope. The Location Class contains a type attribute which allows a specific resolution type to be declared, if the structure of the locator is insufficient to determine the required resolution service. Where Meta-data necessary for known standard resolver services is not explicitly contained within a locator, but dynamically constructed, the meta-data has been included within the RLI data model. It is recommended that the meta-data necessary to construct a persistent locator for any resource be captured in a formal structure according to emerging specifications and standards and as desired by implementers.

Citation

The citation Class attributes are described in Table 4.11.

Table 4.11 Attribute Definitions for the citation Class.

Name	Type	Mult	Description
title	MetadataLangStringDType	1..*	Name given to a resource by a Creator or Publisher
creator	MetadataLangStringDType	0..*	Person or organization primarily responsible for creating the intellectual content of the resource, e.g., author of a document.
edition	MetadataLangStringDType	0..1	Version of resource that is to be included in the Resource List.
publicationDate	MetadataDateDType	0..1	Date resource was made available in its present form.
publicationPlace	MetadataLangStringDType	0..1	Location that resource was made available in its present form.
publisher	MetadataLangStringDType	0..1	Person or organization primarily responsible for making the resource available in its present form, e.g., publishing house, or university department.
volumeDesignation	MetadataStringDType	0..*	Designation for subset being cited that is part of larger whole, such as volume of a multi-volume book, or volume of a journal.
partDesignation	MetadataStringDType	0..*	Designation for next layer down in which cited resource is found, such as chapter in a book or issue number of a journal.
articleNumber	MetadataStringDType	0..*	Identifying number of article within journal issue being cited.
startingPageNumber	MetadataStringDType	0..1	Page at which citation begins, as noted on the display image of the resource.
endingPageNumber	MetadataStringDType	0..1	Page at which the citation ends, as noted on the display image of the resource.

Associations

The set of associations for the citation Class are summarized in Table 4.12.

Table 4.12 Summary of associations for the citation Class.

Association Class Name	Mult	Description
standard Identifier	0..*	Standard number associated with resource, or of Source if resource is part of a whole such as a Chapter in a book or article in a journal, e.g.,SICI, ISBN, ISSN, DOI if already constructed.
relatedTitle	0..1	When resource being cited is part of a larger whole, such as volume title of a journal, or chapter of a book, the relatedTitle Identifies the larger whole.

standardIdentifier

The standardIdentifier Class attributes are described in Table 4.13.

Table 4.13 Attribute Definitions for the standardIdentifier Class.

Name	Type	Mult	Description
standardIdentifierType	MetadataTokenDType	1	An optional type of the identifier. Used when the syntax of the identifier string is insufficient to definitively declare the type of the identifier.
identifierString	MetadataStringDType	1	The identifier string

relatedTitle

The relatedTitle Class attributes are described in Table 4.14.

Table 4.14 Attribute Definitions for the relatedTitle Class.

Name	Type	Mult	Description
title	MetadataLangStringDType	1..*	Name given to the related title by a Creator or Publisher
creator	MetadataLangStringDType	0..*	Person or organization primarily responsible for creating the intellectual content of the related title, e.g., author of a document.
edition	MetadataLangStringDType	0..1	Version of the related title that contains the resource
publicationDate	MetadataDateDType	0..1	Date related title was made available in its present form.
publicationPlace	MetadataLangStringDType	0..1	Location that related title was made available in its present form.
publisher	MetadataLangStringDType	0..1	Person or organization primarily responsible for making the related title available in its present form, e.g., publishing house, or university department.
volumeDesignation	MetadataStringDType	0..1	Designation for subset being cited that is part of larger whole, such as volume of a multi-volume book.
partDesignation	MetadataStringDType	0..1	Designation for next layer down such as a multi volume part of a series.

Associations

The set of associations for the relatedTitle Class are summarized in Table 4.15.

Table 4.15 Summary of associations for the relatedTitle Class.

Association Class Name	Mult	Description
standardIdentifier	0..*	Standard number associated with resource, or of Source if resource is part of a whole such as a Chapter in a book or article in a journal, e.g.,SICI, ISBN, ISSN, DOI if already constructed.

4.3 annotation

The annotation Class attributes are described in Table 4.16.

Table 4.16 Attribute Definitions for the annotation Class.

Name	Type	Mult	Description
annotator	MetadataLangStringDType	1	Name given to person or organization who creates the string which comprises the annotation.
date	MetadataDateDType	1	DateTime at which the annotation is created.
annotationNote	MetadataLangStringDType	1	String describing the educational use, context or other pertinent descriptive information not included elsewhere about the Resource List per se or about a resource within a given Resource List.

4.4 location

The location Class attributes are described in Table 4.17.

Table 4.17 Attribute Definitions for the location Class.

Name	Type	Mult	Description
locationType	MetadataTokenDType	1	Type of a locator. Required where the locator string does not contain the locator type within the string, e.g., a call number.
locator	MetadataStringDType		A string used to access the resource. May be a location, e.g., URL, OpenURL or a method that resolves to a location, e.g., URI

4.5 resourceListSet

4.5.1 Description

The resourceListSet and its associations are is shown in Figure 4.2.

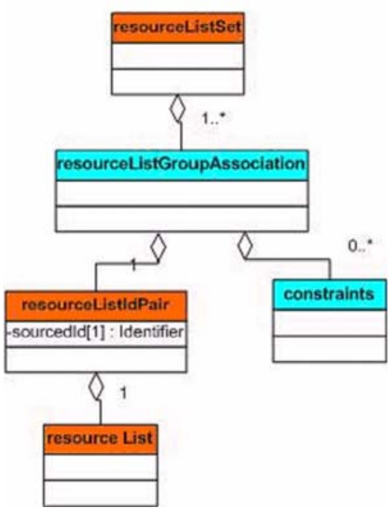


Figure 4.2 resourceListSet Class diagram.

4.5.2 Attributes

None.

4.5.3 Associations

The set of associations for the resourceListSet are summarized in Table 4.18.

Table 4.18 Summary of associations for the resourceListSet Class.

Association Class Name	Mult	Description
resourceListGroupAssociation	1..*	A resource list and the constraints apply for the association of this resource list with this group.

4.5.3.1 resourceListGroupAssociation

4.5.3.1.1 Attributes

None.

4.5.3.1.2 Associations

Table 4.19 Summary of associations for the resourceListGroupAssociation.

Association Class Name	Mult	Description
resourceListIDPair	1..*	A unique identifier for 'Resource List' together with the details of the Resource List associated with the identifier.
constraints	0..1	The set of constraints specified for this resourceListID.

4.6 constraints

4.6.1 Description

Constraints and its associations are is shown in Figure 4.3.

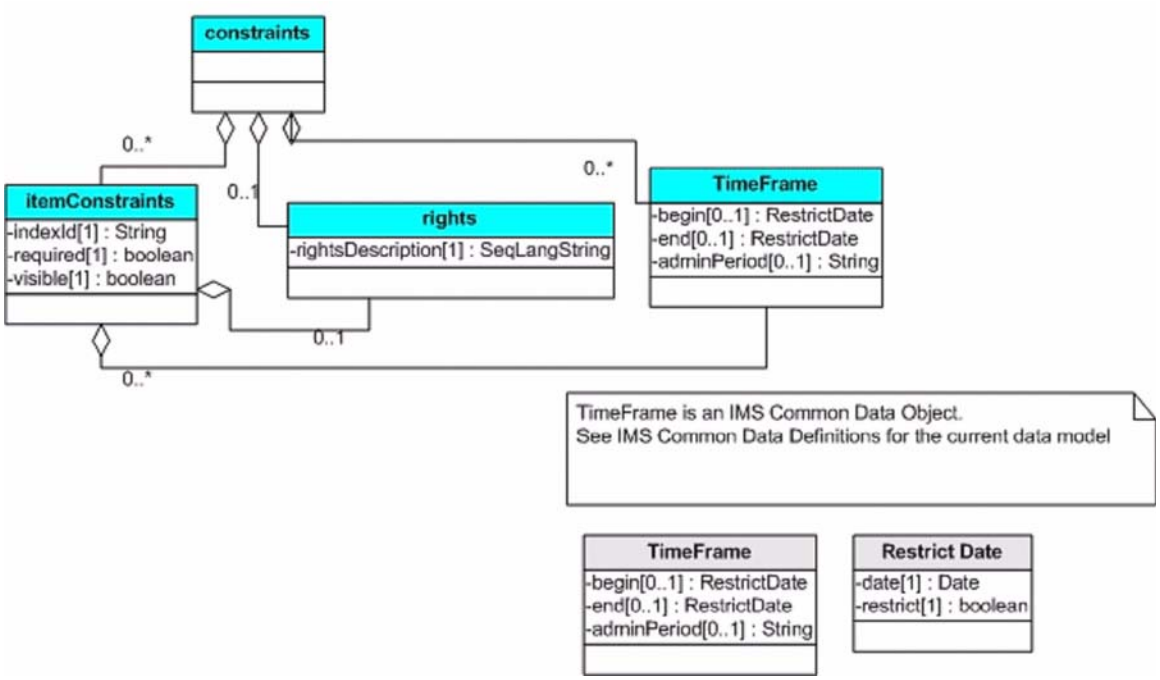

constraints
0..*
0..1
0..*
itemConstraints
-indexId[1] : String
-required[1] : boolean
-visible[1] : boolean
rights
-rightsDescription[1] : SeqLangString
TimeFrame
-begin[0..1] : RestrictDate
-end[0..1] : RestrictDate
-adminPeriod[0..1] : String
0..1
0..*
TimeFrame is an IMS Common Data Object.
See IMS Common Data Definitions for the current data model
TimeFrame
-begin[0..1] : RestrictDate
-end[0..1] : RestrictDate
-adminPeriod[0..1] : String
Restrict Date
-date[1] : Date
-restrict[1] : boolean

Figure 4.3 constraints Class diagram.

Constraints apply to the association between a course and a resources list and define any constraints temporal, conditional, etc. being placed on the association. IMS RLI Implementers are encouraged to leverage constraints available within the behavioral model, in situations where access to resource lists may be of a temporary or restricted nature. However, these constraints are not intended to replace access right management and digital rights management systems, and rules.

4.6.2 Attributes

None.

4.6.3 Associations

The set of associations for the constraints are summarized in Table 4.20.

Table 4.20 Summary of associations for the constraints Class.

Association Class Name	Mult	Description
rights	0..1	Rights restrictions that apply to the association between the resource list and the group.
TimeFrame	0..*	Describes a period for which the association between the resource list and a group is active, i.e., the date range(s) for which the list owner deems a particular resource list relevant within an instructional plan. There is no required change in the behavior of the course. TimeFrame is an IMS Common Data Object. See the IMS Common Data Definitions specification.
itemConstraint	0..*	Constraints that apply to individual resources within the resource list.

4.6.3.1 rights

The rights Class attributes are described in Table 4.21.

Table 4.21 Attribute Definitions for the rights Class.

Name	Type	Mult	Description
rightsDescription	MetadataLangStringDType	1..*	Simple, human readable statement about rights related to Resource List

4.6.4 itemConstraints

The itemConstraints Class attributes are described in Table 4.22.

Table 4.22 Attribute Definitions for the itemConstraints Class.

Name	Type	Mult	Description
indexId	string	1	The unique identifier assigned to a resource within a specific Resource List.
required	Boolean	1	Indicates whether a reading is required for the course, or supplementary material.
visible	Boolean	1	Indicates whether an individual resource is visible in the resource list as instantiated for this association. If there is no item constraint for an individual resource then the default behavior is to treat visible as True for that item.

4.6.4.1 Associations

The set of associations for the itemConstraints Class are summarized in Table 4.23.

Table 4.23 Summary of associations for the itemConstraints Class.

Association Class Name	Mult	Description
rights	0..*	Simple, human readable statement about rights related to an individual item.
TimeFrame	0..*	Describes a period for which an individual resource active i.e., the date range(s) for which the list owner deems a particular resource relevant within an instructional plan. There is no required change in the behavior of the course. TimeFrame is an IMS Common Data Object. See the IMS Common Data Definitions specification.

4.7 Data Types

4.7.1 Description

The RLI Data Types are shown in Figure 4.4.

In the future, it is highly unlikely that any one standard will be adopted to describe bibliographic resources, much less digital resources in general. The Information Model Appendix D includes a mapping of the relevant LOM elements to major bibliographic citation schemas used in the library and publishing worlds. This specification has defined an origin neutral meta-data description of Resource Lists and the resources they describe. However it may be extremely useful for the system processing the Resource List to understand the original meta-data source for the meta-data. The RLI Data Types allow an optional description of the source schema to be associated with the attribute content.

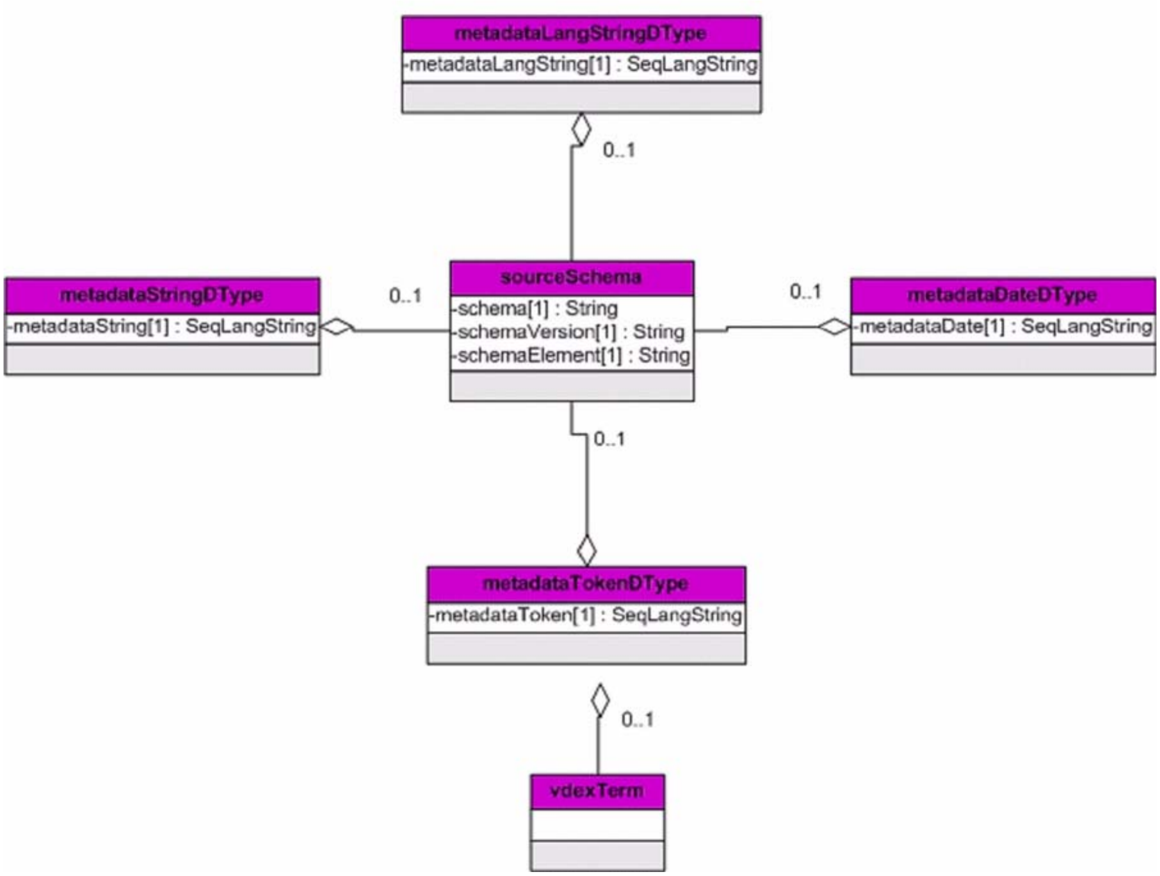

metadataLangStringDType
-metadataLangString[1] : SeqLangString
0..1
metadataStringDType
-metadataString[1] : SeqLangString
0..1
sourceSchema
-schema[1] : String
-schemaVersion[1] : String
-schemaElement[1] : String
0..1
metadataDateDType
-metadataDate[1] : SeqLangString
0..1
metadataTokenDType
-metadataToken[1] : SeqLangString
0..1
vdexTerm

Figure 4.4 RLI Data Types Class diagram.

4.7.2 sourceSchema

4.7.2.1 Attributes

The sourceSchema attributes are described in Table 4.24.

Table 4.24 Attribute definitions for the sourceSchema Class.

Name	Type	Mult	Description
schema	String	1	The schema from which the data was sourced.
schemaVersion	String	1	The version of the schema.
schemaElement	String	1	The schema element (s) from which the data was sourced.

4.7.3 MetadataStringDType

4.7.3.1 Attributes

The MetadataStringDType Class attributes are described in Table 4.25.

Table 4.25 Attribute Definitions for the MetadataStringDType Class.

Name	Type	Mult	Description
metadataString	String	1	Contains the element data for type string.

4.7.3.2 Associations

The set of associations for the MetadataStringDType are summarized in Table 4.26.

Table 4.26 Summary of associations for the MetadataStringDType.

Association Class Name	Mult	Description
sourceSchema	0..1	The schema from which the data was sourced.

4.7.4 MetadataLangStringDType

4.7.4.1 Attributes

The MetadataLangStringDType Class attributes are described in Table 4.27.

Table 4.27 Attribute Definitions for the MetadataLangStringDType Class.

Name	Type	Mult	Description
metadataString	SeqLangString	1	Contains a sequence of one or more language versions of the element data.

4.7.4.2 Associations

The set of associations for the MetadataLangStringDType are summarized in Table 4.28.

Table 4.28 Summary of associations for the MetadataLangStringDType.

Association Class Name	Mult	Description
sourceSchema	0..1	The schema from which the data was sourced.

4.7.5 MetadataDateDType

4.7.5.1 Attributes

The MetadataDateDType Class attributes are described in Table 4.29.

Table 4.29 Attribute Definitions for the MetadataDateDType Class.

Name	Type	Mult	Description
metadataString	date		Contains the element data for type date.

4.7.5.2 Associations

The set of associations for the MetadataDateType are summarized in Table 4.30.

Table 4.30 Summary of associations for the MetadataDateDType.

Association Class Name	Mult	Description
sourceSchema	0..1	The schema from which the data was sourced.

4.7.6 MetadataTokenDType

4.7.6.1 Attributes

The MetadataTokenDType Class attributes are described in Table 4.31.

Table 4.31 Attribute Definitions for the MetadataTokenDType Class.

Name	Type	Mult	Description
metadataString	String	1	Contains the element data where a string is intended to represent a token such as a controlled vocabulary, enumeration etc.

4.7.6.2 Associations

The set of associations for the MetadataTokenDType are summarized in Table 4.32.

Table 4.32 Summary of associations for the MetadataTokenDType.

Association Class Name	Mult	Description
sourceSchema	0..1	The schema from which the data was sourced.
vdexTerm	0..1	This is a place holder to allow vocabulary term to be expressed as a Vdex entry – will liaise with Vdex about the correct way of doing this.

4.8 OCL Definitions

package ResourceListManagementService

context ResourceList

inv: edition.size <= 1024

inv: description.size <= 8192

context ResourceListMetadata

inv: creator.size <= 4096

inv: owner.size <= 4096

inv: created.size <= 128

inv: title.size <= 4096

inv: kind.size <= 128

inv: rightsDescription.size <= 4096

inv: language.size <= 128

context Annotation

inv: annotator.size <= 4096

inv: date.size <= 128

inv: annotationNote.size <=4096

context Location

inv: locationType.size <= 256

inv: locatior.size <= 1024

context ResourceListIdPair

inv: sourcedId <= 2048

context Resource

inv: indexid.size <= 256

inv: type.size <= 128

context ResourceMetadata

inv: description.size <= 4096

inv: language.size <= 4096

inv: format.size <= 512

inv: genre.size <= 256

inv: structure.size <= 256

inv: mode.size <= 128

inv: totalPagesCovered <=128

context Citation

inv: title.size <= 4096

inv: creator.size <= 4096

inv: relatedTitle.size <= 256

inv: edition.size <= 1024

inv: publicationDate.size <= 128

inv: publicationPlace.size <= 512

inv: publisher.size <= 4096

inv: volumeDesignation.size <= 128

inv: partDesignation.size <= 128

inv: articleNumber.size <= 128

inv: startingPageNumber.size <= 64

inv: endingPageNumber.size<= 64

context RelatedTitle

inv: title.size <= 4096

inv: creator.size <= 4096

inv: edition.size <= 1024

inv: publicationDate.size <= 128

inv: publicationPlace.size <= 512

inv: publisher.size <= 256

inv: volumeDesignation.size <= 128

inv: partDesignation.size <= 128

context StandardIdentifier

inv: standardIdentifierType <= 128

inv: identifierString <= 2048

context ItemConstraints

inv: indexId.size <= 256

inv: required.size <= 128

inv: isvisible.size <= 128

context Rights

inv: rightsDescription.size <=4096

5. Extending the Service

5.1 Proprietary Extensions

The proprietary extensions of the specification are based upon two approaches:

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

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

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

5.1.1 Proprietary Operations

The definition of new operations should follow the same format as adopted herein. The new operations should be defined using a new interface type. Every operation must have a returned status code that is the returned value of the operation.

An example of creating such an extension is given in the Resource List Interoperability Best Practice and Implementation Guide [RLI 04c].

5.1.2 Proprietary Data Elements

The addition of proprietary data elements is only permitted directly under the resourceList class. This extension must use the IMSextension class. The format of the extension is limited to a repeated name/value tuple.

An example of creating such an extension is given in the Resource List Interoperability Best Practice and Implementation Guide [RLI, 04c].

5.2 Further Work

The resourceList Management Services Specification and is based on the composition of other clearly defined and functionally discrete operations. This means that the addition of new operations can be achieved without compatibility problems. The areas of work that will be addressed in new versions of the resourceList Management Services are:

a) Inclusion of operations that provide more definitive control over the full set of ‘resourceList’ records e.g., ‘readAllResources()’, ‘deleteAllResources()’, etc.

b) Inclusion of the ‘queryResource()’ operation. This will enable the service to be queried to supply details of the objects that conform to the set of search criteria;

c) Inclusion of discovery operations that allow a system to discover what resourceList records exist e.g. ‘discoverResourceLists()’. This would return the list of the sourceIds of all of the existing resourceList objects;

d) Inclusion of operations that support direct manipulation of some of the sub-objects within the main services e.g., the ability to add/delete/read/change an individual resource or add an annotation to a resource without recourse to the manipulation of the full resourceList object. The corresponding operations for all aggregated parts of the resourceList data model will be considered.

Appendix A – Common Components

The set of common classes that are used throughout this specification are listed in Table A.1. The definitions of these classes can be found in the IMS Common Classes specification.

Table A.1 The set of common classes used by the ResourceListManager Service.

Class Name	Description
Identifier	The unique identification key for a single object.
IMSextension	The standardized extension mechanism for all IMS data model extensions. This is a repeated name/value tuple structure.
SeqLangString	The container for multiple language specified versions of a string
StatusInfo	The container for the detailed status information that is returned by the target to the source in response to a request for action on a single record.
StatusInfoSet	The container for the set of detailed status information that is returned by the target to the source in response to a request for action on a set of records.
TimeFrame	Information that constrains the begin end ad administrative times for access to particular activities, systems, services, etc.

Appendix B – Service Status Codes

B1 – Summary List of Codes

The summary list of status codes that can be returned by the different operations through the StatusInfo object is given in Table B.1 (a detailed description of these codes is given in Section B2). The key to the entries is: ‘Y’ denotes the code may be returned by that operation. A blank entry means that the code cannot be returned by that operation.

Table B.1 Status codes for the ResourceListManager class operations.

Status Code	create	CreateByProxy	read	Read resourceList ForGroup	replace	delete	assign	deassign
‘fullsuccess’	Y	Y	Y	Y	Y	Y	Y	Y
‘idallocfail’		Y						
‘overflowfail’	Y	Y						
‘idallocinusefail’	Y							
‘invaliddata’	Y	Y		Y	Y		Y	
‘incompletedata’	Y	Y	Y	Y	Y		Y	
‘unknownobject’			Y	Y	Y	Y	Y	Y
‘deletefailure’						Y		
‘targetreadfailure’			Y	Y				
‘linkfailure’	Y	Y	Y	Y	Y	Y	Y	Y
‘unsupported’	Y	Y	Y	Y	Y	Y	Y	Y

B2 – Explanation of Operation Specific Codes

B2.1 – ‘create’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The creation request has been fully and successfully implemented by the target system and the ‘resourceList’ record has been created with a unique identifier.
‘idallocinusefail’	The target could not allocate the unique ‘identifier’ to the ‘resourceList’ record as the identifier is already in use.
‘overflowfail’	The target could not create the ‘resourceList’ record due to lack of target allocation memory.
‘invaliddata’	Part or all of the supplied data was detected as invalid by the target system.
‘incompletedata’	Some mandatory part of the data has been detected as missing by the target system.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

B2.2 – ‘createByProxy’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The creation request has been fully and successfully implemented by the target system and the ‘resourceList’ record has been created with the identifier supplied by the source.
‘idallocfail’	The target could not allocate a unique ‘identifier’ to the ‘resourceList’ record as there are no more spare identifiers available.

Status Code	Explanation of the Cause of the Code
‘overflowfail’	The target could not create the ‘resourceList’ record due to lack of target allocation memory.
‘invaliddata’	Part or all of the supplied data was detected as invalid by the source system.
‘incompletedata’	Some mandatory part of the data has been detected as missing by the target system.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

B2.3 – ‘read’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The read request has been fully and successfully implemented by the target system and the identified ‘resourceList’ record has been returned to the source.
‘unknownobject’	The ‘sourcedId’ identifier is unknown in the target system and so the object data could not be read.
‘targetreadfailure’	The target system has detected an error in the stored resourceList record and so cannot return the data.
‘invaliddata’	Part or all of the returned data was detected as invalid by the source system.
‘incompletedata’	Some mandatory part of the data has been detected as missing by the source system.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

B2.4 – ‘readResourceListsForGroup’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The read request has been fully and successfully implemented by the target system and the ‘resourceList’ objects for the identified ‘group’ have been returned to the source system.
‘unknownobject’	The ‘groupSourcedId’ identifier for the Group is unknown in the target system and so the object information could not be returned.
‘targetreadfailure’	The target system has detected an error in the stored resourceList record and so cannot return the data.
‘invaliddata’	Part or all of the returned data was detected as invalid by the source system.
‘incompletedata’	Some mandatory part of the data has been detected as missing by the source system.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

B2.5 – ‘replace’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The replace request has been fully and successfully implemented by the target system and the identified ‘resourceList’ record has been changed on the target system.
‘unknownobject’	The ‘sourcedId’ identifier is unknown in the target system and so the object could not be changed.
‘invaliddata’	Part or all of the returned data was detected as invalid by the target system.

Status Code	Explanation of the Cause of the Code
‘incompletedata’	Some mandatory part of the data has been detected as missing by the target system.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

B2.6 – ‘delete’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The deletion request has been fully and successfully implemented by the target system and the ‘resourceList’ record has been deleted. The corresponding group associations have also been deleted.
‘unknownobject’	The ‘sourcedId’ identifier is unknown in the target system and so the object could not be deleted.
‘deletefailure’	The target system has not been able to delete the identified resourceList object.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

B2.7 – ‘assign’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The assign request has been fully and successfully implemented by the target system and an association has been created between the identified ‘resourceList’ and the identified ‘Group’
‘unknownobject’	The ‘sourcedId’, or ‘groupSourcedId’ identifier is unknown in the target system and so the association could not be created.
‘invaliddata’	Part or all of the returned data was detected as invalid by the target system.
‘incompletedata’	Some mandatory part of the data has been detected as missing by the target system.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

B2.8 – ‘deassign’ Operation

Status Code	Explanation of the Cause of the Code
‘fullsuccess’	The deletion request has been fully and successfully implemented by the target system and the association between the ‘Resource List’ and the ‘group’ has been deleted.
‘unknownobject’	The ‘sourcedId’ or ‘groupSourcedId’ identifier is unknown in the target system and so the association object could not be deleted.
‘deletefailure’	The target system has not been able to delete the identified association object.
‘linkfailure’	There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered.
‘unsupported’	This operation is not supported by the target system.

Appendix C – Sources of Input

In addition to the Use Cases (see the RLI Best Practices Document), the Project Team studied existing applications that currently address or fulfill user needs in the Resource List domain. Below is a sample list of products that enable the creation of Resource Lists at either the course or personal level. More importantly, the Project Team still needs to develop a methodology for reviewing, evaluating and incorporating existing specifications and standards in the schema definition work.

Course Resource List Management System

e.g., Sentient Learning's DISCOVER product, http://www.sentientlearning.com/home

Personal Bibliographic Search and Management Tools (Desktop Clients)

ISI ResearchSoft's EndNOTE, http://endnote.com/

ISI ResearchSoft's ProCite, http://www.procite.com/

Personal Bibliographic Search and Management Tools (ASP)

ExLibris Group's MetaLib, http://www.exlibrisgroup.com/metalib.htm

MuseGlobal Inc.'s Muse products, http://www.museglobal.com/index.html

Fretwell Downing's ZPortal, http://www.fdusa.com/products/zportal.html

Endeavor Information System's ENCompass, http://encompass.endinfosys.com/

WebFeat Inc.'s WebFeat products, http://www.webfeat.org/webfeat_products.html

RefWork's RefWork product, http://www.refworks.com/refworks.shtml

Item Level Meta-data Description

IMS-LOM, IEEE 1484.12.1-2002, Standard for Learning Object Metadata, http://www.ieee.org/

USMARC XML, http://www.loc.gov/standards/marcxml/

Dublin Core, ANSI/NISO Z39.85 - Dublin Core Metadata Element Set - 2001 http://dublincore.org/groups/citation/

MODS, Metadata Object Description Standard, http://www.loc.gov/standards/mods/

ONIX for Books, http://www.editeur.org/onixfiles2.1/prodinf%202.1.html and http://www.loc.gov/marc/onix2marc.html

Packaging and Transfer Protocols

IMS Content Packaging, Version 1.1.3, Final Specification, 2003-June-12, http://www.imsglobal.org/content/packaging/index.cfm

METS, Metadata Encoding and Transmission Standard, Version 1.3, http://www.loc.gov/standards/mets/

Locator schemas

OpenURL, ANSI/NISO Z39.88-200x - The OpenURL Framework for Context-Sensitive Services http://xml.coverpages.org/ni2001-03-13-b.html

URl, Uniform Resource Locators, http://www.faqs.org/rfcs/rfc1738.html

PURL, Persistent Uniform Resource Locators, http://purl.oclc.org/

DOI, ANSI/NISO Z39.84 - Syntax for the Digital Object Identifier - 2000, http://www.niso.org/standards/standard_detail.cfm?std_id=480

Standard ID Schemes/Categories

Common standard numbering schemes of interest in digital content management include those standardized by ISO:

ISBN: ISO 2108:1992 International Standard Book Numbering (ISBN)

ISSN: ISO 3297:1998 International Standard Serial Number (ISSN)

ISRC: ISO 3901:2001 International Standard Recording Code (ISRC)

ISRN: ISO 10444:1997 International Standard Technical Report Number (ISRN)

ISMN: ISO 10957:1993 International Standard Music Number (ISMN)

ISWC: ISO 15707:2001 International Standard Musical Work Code (ISWC)

ISAN: Draft ISO 15706: International Standard Audiovisual Number (ISAN)

V-ISAN: Draft ISO 20925: Version Identifier for audiovisual works (V-ISAN)

ISTC: Draft ISO 21047: International Standard Text Code (ISTC)

Note: While these ISO TC46 identifiers were originally simple numbering schemes, of late they have also begun to adopt the notion of associating some minimal structured descriptive meta-data with the identifier. Also relevant are the ISO- affiliated NISO standards including:

DOI, ANSI/NISO Z39.84 - Syntax for the Digital Object Identifier - 2000, http://www.niso.org/standards/standard_detail.cfm?std_id=480

SICI, ANSI/NISO Z39.56 - Serial Item and Contribution Identifier (SICI) -1996 (R2002), http://www.niso.org/standards/standard_detail.cfm?std_id=530

URI, Uniform Resource Identifier, http://www.w3.org/TR/uri-clarification

Appendix D – Comparison of Descriptive Meta-data Schemes for Citation and Resolution

RLI Element Category	Req'd for Citation	Req'd for Locator	Applies to RL	See Best Practice	LOM Element	LOM Notes	ISO Citation Element Name (690-2)	ISO: Whole (doc or jrnl)	ISO Part (of doc)	ISO Part (article)	OpenURL (varies depending upon genre)	DOI Kernel MD req'd	Comments
NOTE: Blue cells indicate RLI recommended metadata elements that do not appear in the IEEE LOM schema.													
Creator	x	x	x		2.3.2:LifeCycle. Contribute.Entity	Used when 2.3.1:LifeCycle.Contribute.Role has a value of “Author”.	Primary Resp	x	x	x	aulast, aufirst, auinit, auinit1, auinitm [separate elements]	Primary agent (responsible for resource), Primary agent role [separate elements]	Can be used at the Resource & Resources List level.
Title	x	x	x		1.2:General.Title	The name given to the resource being used or cited by the CREATOR or PUBLISHER	Title of contribution	x	x	x	title, stitle, atitle [separate elements]	Title	Can be used at the Resource & Resources List level.
Related Title	x	x			7.2:Relation.Resource	Title of whole in which cited Resource is a part. 7.1:Relation.Kind. Should have value “IsPartOf”.	Title of serial		x	x	Issue		The name of the whole in which the resource is found, e.g., the journal in which the cited article is located or the book in which a cited chapter is found.
Edition	x		x		2.1:Lifecycle.Version		Edition	x	x	x			Edition either of Resource itself, e.g., of book; or edition of Source, if Resource is part of a whole such as a Chapter of a book. Can be used at either the Resource and Resources List level.
Place of publication	x			x		Unused; Considered unnecessary?	Place of Publication	x	x				
Publisher	x				2.3.2: Lifecycle.Contribute.Entity.	Used when 2.3.3: Lifecycle.Contribute.Role has a value of “Publisher”	Publisher	x	x				Can be used at the Resource & Resources List level.
Publication Date	x	x	x		2.3.3: Lifecycle.Contribute.Date .	Used when 2.3.1: Lifecycle.Contribute.Role has a value of “Publisher”	Date of publ	x	x		date (publ date), ssn (season of publication), quarter are separate elements		Can be used at the Resource & Resources List level. For Resources List, this is the date at which the RL is made available.

RLI Element Category	Req'd for Citation	Req'd for Locator	Applies to RL	See Best Practice	LOM Element	LOM Notes	ISO Citation Element Name (690-2)	ISO: Whole (doc or jrnl)	ISO Part (of doc)	ISO Part (article)	OpenURL (varies depending upon genre)	DOI Kernel MD req'd	Comments
Standard identifier	x	x	x		1.1.1:General. Identifier.Entry	LOM Element 1.1.2: General.Catalog.Entry would name the identifier schema, such as ISSN, ISBN. Recommend use of URI when possible.	Standard Num	x	x	x	issn, eissn, coden, isbn, sici, bici [separate elements]	Identifier (standard # other than DOI, e.g., ISSN)	Standard number associated with Resource, or of Source if Resource is part of a whole such as a Chapter in a book or article in a journal, e.g.,SICI, ISBN, ISSN, DOI if already constructed. May also be used for RL identifier.
Volume Designation	x	x		x	7.2.2:Resource. Description	Should include the necessary parts to the bibliographic citation (volume or part number, article number, pages, etc.) within the description	Chapter or part		x		Volume		Designation of subset being cited when Resource is part of a whole, such as volume of a multi-volume book, chapter of a book or volume of a journal.
Part Designation	x	x		x	7.2.2:Resource. Description	Should include the necessary parts to the bibliographic citation (volume or part number, article number, pages, etc.) within the description	Issue designation		x	x	Part		Designation of next layer of subset being cited, such as issue number of a journal in which Resource is found.
Article Number		x		x	7.2.2:Resource. Description	Should include the necessary parts to the bibliographic citation (volume or part number, article number, pages, etc.) within the description	Issue designation			x	Artnum (article #)		Necessary to construct OpenURL. Included by OpenURL as designator for specific numbering identifier for Resource as article, if the number exists.
Starting page no.		x		x	7.2.2:Resource. Description	Should include the necessary parts to the bibliographic citation (volume or part number, article number, pages, etc.) within the description	Issue designation		x	x	spage (start pg)		Necessary to construct OpenURL.
Ending page no.		x		x	7.2.2:Resource. Description	Should include the necessary parts to the bibliographic citation (volume or part number, article number, pages, etc.) within the description	Issue designation		x	x	epage (end pg)		Necessary to construct OpenURL.
Total pages covered		x		x	7.2.2:Resource. Description	Should include the necessary parts to the bibliographic citation (volume or part number, article number, pages, etc.) within the description					pages (total pgs covered)		Necessary to construct OpenURL.

About This Document

Title	IMS Resource List Interoperability Information Model
Editor	Alex Jackl (IMS)
Team Co-Leads	Oliver Heyer (UC Berkeley), Mladen Maljkovic (WebCT)
Version	1.0
Version Date	08 July 2004
Status	**Final Specification**
Summary	This document describes the Resource List Interoperability Information Model
Revision Information	July 2004
Purpose	This document has been approved by the IMS Technical Board and is made available for adoption.
Document Location	http://www.imsglobal.org/rli/rliv1p0/imsrli_infov1p0.html

To register any comments or questions about this specification please visit:
http://www.imsglobal.org/developers/ims/imsforum/categories.cfm?catid=22

List of Contributors

The following individuals contributed to the development of this document:

Name	**Organization**
Phil Barker	CETIS
Kerry Blinco	DEST
Lorna Campbell	CETIS
Norm Friesen	Industry Canada
Oliver Heyer	University of California, Berkeley
Nancy Hoebelheinrich	Stanford University
Alex Jackl	IMS Global Learning Consortium, Inc.
Mladen Maljkovic	WebCT
Jon Mason	DEST
Howard Noble	Sentient
Claude Ostyn	Click2learn, Inc.
Dan Rehak	Carnegie Mellon
Scott Wilson	CETIS

Revision History

Version No.	Release Date	Comments
Base Document 1.0	11 November 2003	Initial version of the Resources List Interoperability Information Model.
Public Draft 1.0	31 May 2004	Public Draft version of the Resources List Interoperability Information Model.
Final Specification 1.0	08 July 2004	This is the formal Final version of the IMS Resource List Interoperability Information Model.

Index

IMS Global Learning Consortium, Inc. ("IMS") is publishing the information contained in this IMS Resource List Interoperability Information Model *("Specification") for purposes of scientific, experimental, and scholarly collaboration only.*

IMS 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 would appreciate receiving your comments and suggestions.

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

Please refer to Document Name: IMS Resource List Interoperability Information Model

Date: 08 July 2004