



***Society of Cable
Telecommunications
Engineers***

ENGINEERING COMMITTEE
Digital Video Subcommittee

SCTE 130-8 2013

Digital Program Insertion–Advertising Systems Interfaces
Part 8–General Information Service (GIS)

NOTICE

The Society of Cable Telecommunications Engineers (SCTE) Standards are intended to serve the public interest by providing specifications, test methods and procedures that promote uniformity of product, interchangeability and ultimately the long term reliability of broadband communications facilities. These documents shall not in any way preclude any member or non-member of SCTE from manufacturing or selling products not conforming to such documents, nor shall the existence of such standards preclude their voluntary use by those other than SCTE members, whether used domestically or internationally.

SCTE assumes no obligations or liability whatsoever to any party who may adopt the Standards. Such adopting party assumes all risks associated with adoption of these Standards, and accepts full responsibility for any damage and/or claims arising from the adoption of such Standards.

Attention is called to the possibility that implementation of this standard may require the use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. SCTE shall not be responsible for identifying patents for which a license may be required or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Patent holders who believe that they hold patents which are essential to the implementation of this standard have been requested to provide information about those patents and any related licensing terms and conditions. Any such declarations made before or after publication of this document are available on the SCTE web site at <http://www.scte.org>.

All Rights Reserved

© Society of Cable Telecommunications Engineers, Inc.
140 Philips Road
Exton, PA 19341

TABLE OF CONTENTS

1.0	SCOPE	1
2.0	REFERENCES	1
2.1	Normative References	1
2.1.1	Normative Reference Acquisition.....	1
2.2	Informative References	2
2.2.1	Informative Reference Acquisition.....	2
3.0	COMPLIANCE NOTATION.....	2
4.0	DEFINITIONS AND ACRONYMS.....	2
5.0	ABBREVIATIONS	4
6.0	GENERAL INFORMATION SERVICE OVERVIEW.....	4
7.0	NOTATIONAL CONVENTIONS	7
7.1	Normative XML Schema	7
7.2	Document Conventions	7
8.0	PROCESSING CONVENTIONS.....	8
8.1	Unknown/Unrecognized/Unsupported XML Elements and Attributes	8
9.0	XML NAMESPACES	8
10.0	GIS MESSAGE TYPES.....	9
10.1	@version Attribute.....	11
10.2	Request Base Message Type	11
10.3	Response Base Message Type.....	11
10.4	Notification Base Message Type.....	11
10.5	Acknowledgement Base Message Type	11
10.6	Messages Requiring Notification Registration.....	11
10.7	Default Logical Service Channel EndPoint.....	12
11.0	GIS MESSAGE EXCHANGE.....	13
11.1	ListSupportedFeaturesRequest and ListSupportedFeaturesResponse Message Types.....	14
11.1.1	ListSupportedFeaturesRequest Message Type.....	14
11.1.2	ListSupportedFeaturesResponse Message Type	15
11.2	ListQualifiersRequest and ListQualifiersResponse Message Types.....	18
11.2.1	ListQualifiersRequest Message Type	18
11.2.2	ListQualifiersResponse Message Type.....	20
11.3	ListNotificationRegistrationRequest and ListNotificationRegistrationResponse Message Types.....	22
11.3.1	ListNotificationRegistrationRequest Message Types	22
11.3.2	ListNotificationRegistrationResponse Message Type	24
11.4	NotificationRegistrationRequest and NotificationRegistrationResponse Message Types.....	26
11.4.1	NotificationRegistrationRequest Message Type.....	26
11.4.2	NotificationRegistrationResponse Message Type	28
11.5	Notification and NotificationAcknowledgement Message Types	30
11.5.1	Notification Message Type.....	30
11.5.2	NotificationAcknowledgement Message Type.....	32
11.6	CreateCursorRequest and CreateCursorResponse Message Types	34
11.6.1	CreateCursorRequest Message Type	34
11.6.2	CreateCursorResponse Message Type.....	36
11.7	CancelCursorRequest and CancelCursorResponse Message Types	38
11.7.1	CancelCursorRequest Message Type.....	38

11.7.2	<i>CancelCursorResponse Message Type</i>	40
11.8	QueryRequest and QueryResponse Message Types	42
11.8.1	<i>QueryRequest Message Type</i>	42
11.8.2	<i>QueryResponse Message Type</i>	44
11.9	NotificationDeregisterRequest and NotificationDeregisterResponse Message Types	46
11.9.1	<i>NotificationDeregisterRequest Message Type</i>	46
11.9.2	<i>NotificationDeregisterResponse Message Type</i>	48
11.10	DeregistrationNotification and DeregistrationAcknowledgement Message Types	50
11.10.1	<i>DeregistrationNotification Message Type</i>	50
11.10.2	<i>DeregistrationAcknowledgement Message Type</i>	52
11.11	Service Check Messages	54
11.12	Service Status Messages	54
12.0	GIS ELEMENT DETAILS	54
12.1	AdvancedFilterElement	56
12.2	AdvancedQueryFilter	57
12.3	AdvancedQueryLanguage	58
12.4	AdvancedQueryResult	59
12.5	AdvancedQueryResultData	60
12.6	BasicFilterElement	60
12.7	BasicQueryDataModelDescription	61
12.8	BasicQueryFilter	63
12.9	BasicQueryResult	64
12.10	BasicQueryResultAbstract	65
12.11	Cursor	66
12.12	EnumerationValue	68
12.13	MaxFloat	68
12.14	MaxInteger	69
12.15	MaxLength	69
12.16	MinFloat	70
12.17	MinInteger	70
12.18	Qualifier	71
12.19	QualifierDeclaration	72
12.20	QualifierDescription	72
12.21	QualifierSet	75
12.22	Query	75
12.23	QueryResult	78
12.24	ServiceDataModel	83
12.25	ServiceDataModelProfile	83
12.26	UniqueQualifier	84
12.27	UniqueQualifierDeclaration	85
13.0	SCTE 130 PART 8 GIS ATTRIBUTE TYPES AND COMMON TYPES	86
13.1	cursorIdAttrType Attribute Type	86
13.2	cursorIdRefAttrType Attribute Type	86
13.3	expandOutputAttrType Attribute Type	87
13.4	filterElementNameType Attribute Type	87
13.5	filterElementValueAttrType Attribute Type	87
13.6	MinMaxFloatType Complex Type	87
13.7	MinMaxIntegerType Complex Type	87
13.8	MaxLengthType Complex Type	87
13.9	NotificationTypeEnumeration Attribute Type	88
13.10	qualifierNameAttrType Attribute Type	88
13.11	QualifierValueTypeEnumerationType Attribute Type	88
13.12	QueryFilterOpTypeEnumeration Attribute Type	88
13.13	queryIdAttrType Attribute Type	89

13.14	queryIdRefAttrType Attribute Type.....	89
13.15	queryLanguageAttrType Attribute Type.....	89
13.16	resultSetSizeOnlyAttrType Attribute Type.....	89
13.17	totalResultSetSizeAttrType Attribute Type.....	89
13.18	uniqueQualifierNameAttrType Attribute Type.....	89
14.0	BASIC QUERIES AND REGULAR EXPRESSIONS	90
14.1	Regular Expressions and Wildcards.....	90
APPENDIX A. STATUSCODE ELEMENT @DETAIL ATTRIBUTE VALUES (NORMATIVE) ...		93
APPENDIX B. COMPLEX QUERIES & EXPANDED OUTPUT EXAMPLES (INFORMATIVE) .		95
B.1	Multiple Filter Elements.....	95
B.2	Expanded Output.....	96
APPENDIX C. ADVANCED QUERIES (INFORMATIVE).....		100
C.1	Advanced Query Examples	100
APPENDIX D. CURSOR EXAMPLES (INFORMATIVE)		101
D.1	Creating Cursors	101
D.2	Traversing a Cursor.....	102
D.3	Canceling Existing Cursors.....	103
APPENDIX E. MESSAGE EXAMPLES (INFORMATIVE).....		105
E.1	Listing Supported Features.....	105
E.2	Listing Qualifiers.....	107
E.3	Query Examples	109
E.4	Notification Registration Examples.....	113
E.5	Notification.....	114
APPENDIX F. MUTABLE GIS INTERFACE (NORMATIVE)		116
F.1	SCOPE	116
F.2	INTRODUCTION.....	116
F.3	DATA INTEGRITY	119
F.4	ERROR HANDLING.....	119
F.5	XML NAMESPACES.....	121
F.6	ENHANCED GIS MESSAGE TYPES	121
F.6.1	LISTSUPPORTEDFEATURESREQUEST MESSAGE TYPE.....	121
F.6.2	LISTQUALIFIERS MESSAGES.....	122
F.7	GIS MUTABLE MESSAGE TYPES.....	122
F.7.1	MUTABLEOPERATION MESSAGE TYPES	123
F.7.1.1	MUTABLEOPERATIONREQUEST MESSAGE TYPE.....	123
F.7.1.2	MUTABLEOPERATIONRESPONSE MESSAGE TYPE.....	126
F.7.2	BATCHCREATE MESSAGES	128
F.7.2.1	BATCHCREATEREQUEST MESSAGE TYPE	128
F.7.2.2	BATCHCREATERESPONSEMESSAGE TYPE	130
F.7.3	BATCHITEM MESSAGES	132
F.7.3.1	BATCHITEMREQUEST MESSAGE TYPE	132
F.7.3.2	BATCHITEMRESPONSE MESSAGE TYPE	134

F.7.4 BATCHOPERATION MESSAGES.....	136
F.7.4.1 BATCHOPERATIONREQUEST MESSAGE TYPE.....	136
F.7.4.2 BATCHOPERATIONRESPONSE MESSAGE TYPE.....	138
F.7.5 BATCH NOTIFICATION MESSAGES	140
F.7.5.1 BATCHNOTIFICATION MESSAGE TYPE.....	140
F.7.5.2 BATCHNOTIFICATIONACKNOWLEDGEMENT MESSAGE TYPE	142
F.7.6 MUTABLE OPERATION NOTIFICATION MESSAGES	144
F.7.6.1 MUTABLEOPERATIONNOTIFICATION MESSAGE TYPE.....	144
F.7.6.2 MUTABLEOPERATIONACKNOWLEDGEMENT MESSAGE TYPE	146
F.7.7 LIST BATCH MESSAGES.....	147
F.7.7.1 LISTBATCHREQUEST MESSAGE TYPE.....	147
F.7.7.2 LISTBATCHRESPONSE MESSAGE TYPE.....	149
F.8 MUTABLE GIS ATTRIBUTE TYPES AND COMMON TYPES	151
F.8.1 BATCHIDATTRTYPE ATTRIBUTE TYPE.....	151
F.8.2 BATCHIDREFATTRTYPE ATTRIBUTE TYPE.....	151
F.8.3 TRANSACTIONLEVELTYPEENUMERATION ATTRIBUTE TYPE	151
F.8.4 UPDATEIDATTRTYPE ATTRIBUTE TYPE.....	152
F.8.5 UPDATEIDREFATTRTYPE ATTRIBUTE TYPE.....	152
F.9 MUTABLE GIS ELEMENT DETAILS.....	152
F.9.1 BATCH	153
F.9.2 BATCHCREATEREQUESTABSTRACT.....	153
F.9.3 BATCHINFORMATION	154
F.9.4 BATCHOPERATIONREQUESTABSTRACT	155
F.9.5 CANCELBATCH.....	156
F.9.6 DATAMODELDESCRIPTIONQUERY.....	157
F.9.7 DATAMODELDESCRIPTIONQUERYRESULT	158
F.9.8 EXECUTEBATCH	158
F.9.9 EXPRESSION	159
F.9.10 EVALUATIONARGUMENT	160
F.9.11 MUTABLEQUALIFIERDESCRIPTION.....	161
F.9.12 MUTABLESERVICEDATAMODELPROFILE	163
F.9.13 PUTDATAMODELQUALIFIER.....	164
F.9.14 PUTQUALIFIER	164
F.9.15 QUALIFIERVERVALUE	166
F.9.16 QUALIFIERVERVALUEDECREMENT.....	167
F.9.17 QUALIFIERVERVALUEEXPRESSION	167
F.9.18 QUALIFIERVERVALUEINCREMENT.....	168

F.9.19 REMOVEDATAMODELQUALIFIER	169
F.9.20 REMOVEQUALIFIER	170
F.9.21 UPDATEERROR	171
F.9.22 UPDATEEXPRESSIONLANGUAGE	172
APPENDIX G. MUTABLE GIS EXAMPLES (INFORMATIVE)	173
G.1 LISTING SUPPORTED FEATURES.....	173
G.2 LISTING QUALIFIERS.....	173
G.3 MUTABLE OPERATION.....	174
G.4 BATCH MUTABLE OPERATION.....	177
G.5 DATA MODEL DESCRIPTION NOTIFICATION	180
15.0 SCHEMA CHANGES.....	181

LIST OF FIGURES

Figure 1. General Information Service Interface	5
Figure 2. Substitution Group Schema Convention.....	8
Figure 3. GIS Message Exchange	13
Figure 4. ListSupportedFeaturesRequestType XML Schema	14
Figure 5. ListSupportedFeaturesResponseType XML Schema	16
Figure 6. ListQualifiersRequestType XML Schema.....	19
Figure 7. ListQualifiersResponseType XML Schema	21
Figure 8. ListNotificationRegistrationRequestType XML Schema	23
Figure 9. ListNotificationRegistrationResponseType XML Schema.....	25
Figure 10. NotificationRegistrationRequestType XML Schema	27
Figure 11. NotificationRegistrationResponseType XML Schema	29
Figure 12. NotificationType XML Schema	31
Figure 13. NotificationAcknowledgementType XML Schema.....	33
Figure 14. CreateCursorRequestType XML Schema.....	35
Figure 15. CreateCursorResponseType XML Schema	37
Figure 16. CancelCursorRequestType XML Schema	39
Figure 17. CancelCursorResponseType XML Schema	41
Figure 18. QueryRequestType XML Schema	43
Figure 19. QueryResponseType XML Schema	45
Figure 20. NotificationDeregisterRequestType XML Schema	47
Figure 21. NotificationDeregisterResponseType XML Schema.....	49
Figure 22. DeregistrationNotificationType XML Schema.....	51
Figure 23. DeregistrationAcknowledgementType XML Schema	53
Figure 24. AdvancedFilterElement Element XML Schema.....	56
Figure 25. AdvancedQueryFilter Element XML Schema	57
Figure 26. AdvancedQueryLanguage Element XML Schema	58
Figure 27. AdvancedQueryResult Element XML Schema.....	59
Figure 28. AdvancedQueryResultData Element XML Schema	60
Figure 29. BasicFilterElement Element XML Schema	61
Figure 30. BasicQueryDataModelDescription Element XML Schema	62
Figure 31. BasicQueryFilter Element XML Schema	63
Figure 32. BasicQueryResult Element XML Schema.....	64
Figure 33. BasicQueryResultAbstract Element XML Schema	66
Figure 34. Cursor Element XML Schema.....	67
Figure 35. EnumerationValue Element XML Schema.....	68
Figure 36. MaxFloat Element XML Schema.....	69
Figure 37. MaxInteger Element Xml Schema.....	69
Figure 38. MaxLength Element XML Schema	70
Figure 39. MinFloat Element XML Schema.....	70
Figure 40. MinInteger Element XML Schema.....	71
Figure 41. Qualifier Element XML Schema	71
Figure 42. QualifierDeclaration Element XML Schema.....	72
Figure 43. QualifierDescription Element XML Schema.....	73
Figure 44. QualiferSet Element XML Schema	75
Figure 45. Query Element XML Schema.....	76
Figure 46. QueryResult Element XML Schema.....	78
Figure 47. ServiceDataModel Element XML Schema.....	83
Figure 48. ServiceDataModelProfile Element XML Schema.....	83
Figure 49. UniqueQualifer Element XML Schema.....	84
Figure 50. UniqueQualifer Element XML Schema.....	86
Figure 51. Mutable Operation Sequence Diagram.....	117

Figure 52. Batch Operation Sequence Diagram.....	118
Figure 53. MutableOperationRequestType XML Schema.....	124
Figure 54. MutableOperationResponseType XML Schema	127
Figure 55. BatchCreateRequestType XML Schema	129
Figure 56. BatchCreateResponseType XML Schema.....	131
Figure 57. BatchItemRequestType XML Schema	133
Figure 58. BatchItemResponseType XML Schema.....	135
Figure 59. BatchOperationRequestType XML Schema.....	137
Figure 60. BatchOperationResponseType XML Schema	139
Figure 61. BatchNotificationType XML Schema	141
Figure 62. BatchNotificationAcknowledgementType XML Schema	143
Figure 63. MutableOperationNotificationType XML Schema	145
Figure 64. MutableOperationAcknowledgementType XML Schema.....	146
Figure 65. ListBatchRequestType XML Schema	148
Figure 66. ListBatchResponseType XML Schema.....	150
Figure 67. Batch XML Schema	153
Figure 68. BatchCreateRequestAbstract XML Schema.....	154
Figure 69. BatchInformation XML Schema	155
Figure 70. BatchOperationRequestAbstract XML Schema	156
Figure 71. CancelBatch XML Schema	157
Figure 72. DataModelDescriptionQuery XML Schema	157
Figure 73. DataModelDescriptionQueryResult XML Schema	158
Figure 74. ExecuteBatch XML Schema.....	159
Figure 75. Expression XML Schema.....	160
Figure 76. EvaluationArgument XML Schema.....	160
Figure 77. MutableQualifierDescription XML Schema.....	162
Figure 78. MutableServiceDataModelProfile XML Schema.....	163
Figure 79. PutDataModelQualifier XML Schema	164
Figure 80. PutQualifier XML Schema	165
Figure 81. QualifierValue XML Schema.....	166
Figure 82. QualifierValueDecrement XML Schema.....	167
Figure 83. QualifierValueExpression XML Schema	168
Figure 84. QualifierValueIncrement XML Schema.....	169
Figure 85. RemoveDataModelQualifier XML Schema	169
Figure 86. RemoveQualifier XML Schema	170
Figure 87. UpdateError Element XML Schema	171
Figure 88. UpdateExpressionLanguage Element XML Schema.....	172

LIST OF TABLES

Table 1. Normative References	1
Table 2. Informative References.....	2
Table 3. Compliance Notation	2
Table 4. XML Namespace Declarations.....	8
Table 5. GIS Specific Message Types	10
Table 6. ListSupportedFeaturesResponseType core:Callout @message values	17
Table 7. List ADS Registration Inquiry Granularity Control.....	24
Table 8. NotificationRegistrationRequest core:Callout @message Values	28
Table 9. Notification @noticeType Values	30
Table 10. GIS Elementary Element Details	55
Table 11. Advanced Query Languages	56
Table 12. Allowed QualifierDescription Element @valueType Values	74
Table 13. QueryResult Element Truth Table for an Advertising Service Using the BasicQueryResult Element	81
Table 14. QueryFilterOpTypeEnumeration Values	88
Table 15. GIS Regular Expressions.....	91
Table 16. Regular Expression Examples	92
Table 17. StatusCode Details.....	93
Table 18. Part 2 and Part 8 StatusCode Detail Usage.....	94
Table 19. Message Element Types By Mutable Operation Type.....	120
Table 20. core:StatusCode Element @class Attribute Values	121
Table 21. Mutable GIS XML Namespace Declarations	121
Table 22. GIS Message Types Altered by the Mutable GIS	121
Table 23. Mutable GIS Message Types.....	123
Table 24. TransactionLevelTypeEnumeration Values.....	152
Table 25. Mutable GIS Elementary Element Details.....	153

LIST OF EXAMPLES

Example 1. SISListSupportedFeaturesRequest Message Definition.....	11
Example 2. BasicQueryFilter Element.....	90
Example 3. BasicQueryFilter Element.....	90
Example 4. BasicQueryFilter Element.....	95
Example 5. BasicQueryFilter Element.....	96
Example 6. BasicQueryFilter Element.....	96
Example 7. Query Element	97
Example 8. QueryResult Element	97
Example 9. Query Element using UniqueQualifer	98
Example 10. Query Element	98
Example 11. QueryResult Element	98
Example 12. AdvancedQueryFilter Element	100
Example 13. QueryResult Element	100
Example 14. CreateCursorRequest Message	101
Example 15. CreateCursorResponse Message	102
Example 16. QueryRequest Message.....	102
Example 17. QueryRequest Message.....	103
Example 18. CancelCursorRequest Message.....	103
Example 19. CancelCursorResponse Message	104
Example 20. ListSupportedFeaturesRequest Message.....	105
Example 21. ListSupportedFeaturesResponse Message	105
Example 22. ListSupportedFeaturesResponse Message	106
Example 23. ListSupportedFeaturesResponse Message	107
Example 24. ListQualifiersRequest Message.....	107
Example 25. ListQualifiersResponse Message	108
Example 26. QueryRequest Message.....	109
Example 27. QueryResponse Message	110
Example 28. QueryRequest Message.....	110
Example 29. QueryResponse Message	111
Example 30. QueryRequest Message.....	111
Example 31. QueryResponse Message	112
Example 32. NotificationRegistrationRequest Message	113
Example 33. NotificationRegistrationRequest Message	114
Example 34. Notification Message	115
Example 35. ListSupportedFeaturesRequest Message.....	173
Example 36. ListSupportedFeaturesResponse Message	173
Example 37. ListQualifiersRequest Message.....	173
Example 38. ListQualifiersRequest Message.....	174
Example 39. MutableOperationRequest Message (synchronous).....	175
Example 40. MutableOperationResponse Message.....	176
Example 41. MutableOperationRequest Message (asynchronous)	176
Example 42. MutableOperationNotification Message	177
Example 43. MutableOperationAcknowledgement Message	177
Example 44. BatchCreateRequest Message.....	177
Example 45. BatchCreateResponse Message	178
Example 46. BatchItemRequest Message.....	178
Example 47. BatchItemResponse Message.....	178
Example 48. BatchOperationRequest Message	179
Example 49. BatchOperationResponse Message.....	179
Example 50. BatchNotification Message.....	179
Example 51. BatchNotificationAcknowledgement Message	180

Example 52. NotificationRegistrationRequest Message	180
Example 53. Notification Message	180

Digital Program Insertion—Advertising Systems Interfaces

Part 8—General Information Service (GIS)

1.0 SCOPE

This document, SCTE 130 Part 8, describes the Digital Program Insertion Advertising Systems Interfaces' General Information Service (GIS) messaging and data type specification using XML, XML Namespaces, and XML Schema.

2.0 REFERENCES

2.1 Normative References

The following standards contain provisions that, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

[SCTE130-2]	ANSI/SCTE 130-2 2008a: Digital Program Insertion—Advertising Systems Interfaces Part 2—Core Data Elements
[SCTE130-7]	ANSI/SCTE 130-7 2009: Digital Program Insertion—Advertising Systems Interfaces Part 7—Message Transport
[W3C–XPath]	Xpath — W3C REC: XML Path Language (XPath) Version 1.0. November 16, 1999
[W3C–XQuery]	XQuery — W3C REF: An XML Query Language (XQuery) Version 1.0. January 23, 2007

Table 1. Normative References

All normative references found in [SCTE130-2] are included and apply to this document. See [SCTE130-2] for additional information.

All normative references found in [SCTE130-7] are included and apply to this document. See [SCTE130-7] for additional information.

2.1.1 Normative Reference Acquisition

2.1.1.1 SCTE Standards: United States of America

Society of Cable Telecommunications Engineers Inc., 140 Philips Road, Exton, PA 19341; Telephone 800-542-5040; Facsimile: 610-363-5898; E-mail: standards@scte.org; URL: <http://www.scte.org>.

2.1.1.2 W3C Standards

MIT, 32 Vassar Street, Room 32-G515, Cambridge, MA 02139, USA;
Telephone: +1.617.258.5999; <http://www.w3.org>.

2.2 Informative References

The following documents may provide valuable information to the reader but are not required when complying with this standard.

[SCTE130-1]	SCTE 130-1 2008: Digital Program Insertion— Advertising Systems Interfaces Part 1—Overview
-------------	---

Table 2. Informative References

2.2.1 Informative Reference Acquisition

2.2.1.1 SCTE Standards: United States of America

See Section 2.1.1.1

3.0 COMPLIANCE NOTATION

“SHALL”	This word or the adjective “REQUIRED” means that the item is an absolute requirement of this specification.
“SHALL NOT”	This phrase means that the item is an absolute prohibition of this specification.
“SHOULD”	This word or the adjective “RECOMMENDED” means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighted before choosing a different course.
“SHOULD NOT”	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
“MAY”	This word or the adjective “OPTIONAL” means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

Table 3. Compliance Notation

4.0 DEFINITIONS AND ACRONYMS

Throughout this standard the terms below have specific meanings. Because some of the terms are defined in other SCTE documents having very specific technical meanings, the

reader is referred to the original source for their definition. For terms defined by this standard, brief definitions are given below.

All [SCTE130-2] definitions are included herein. See [SCTE130-2] for additional information.

Advanced Query: The "Advanced Query" interface defined by SCTE 130 Part 8 permits the consumer of a logical service implementation derived from SCTE 130 Part 8 to use an advanced query language to formulate queries against a logical service's data model.

AdvancedQueryLanguage: As used in this document, the term "Advanced Query Language" refers to any language used to formulate queries against a logical service's data model using the advanced query interface. XPath and XQuery are each examples of an advanced query language.

AdvancedQueryFilter: An "Advanced Query Filter" is a collection of free form data items that individually represent complete query terms for a given query language. The individual terms are additively applied (ANDed) together during a query operation against a specific service data model, which results in the identification of a collection of objects contained within the data store.

Basic Query: To obtain information from a logical service implementation derived from SCTE 130 Part 8, a logical service consumer issues a "query" against the data. The "Basic Query" interface is based on an exchange of name/value pairs, referred to as qualifiers, and requires no specialized knowledge of advanced query languages such as XQuery.

BasicQueryFilter: A "Basic Query Filter" is a collection of name and value pairs additively applied (ANDed) together during a basic query operation against a specific service data model, which results in the identification of a collection of objects contained within the data store.

Cursor: A temporary construct containing static data. Consumers of logical services implementing the GIS interface may create and access cursor information using the standard query mechanisms described in this document.

Data Model: A Data Model is a formal view of the data items contained in an information store to which an information service implementing this standard will provide access and is specified for purposes of formulating and executing queries against the information store's data. This standard specifies one data model that may be used for querying a logical service's information store with this standard's "Basic Query" interface. More complex data models may be specified independently of this standard and may be queried with this standard's "Advanced Query" interface. In this latter case, the mechanisms by which a consumer incorporates a data model specification so that meaningful queries may be issued against it are outside the scope of this standard.

Default Data Model: The data model that is used when no specific data model is provided. The default data model typically occupies the first location in a data model sequence (i.e., a list of data models).

Qualifier: A “Qualifier” is a name/value pair used to describe one characteristic of an object in a logical service's basic query data model. For instance, <Qualifier name=”Age” value=”30to40”/> is an example of a qualifier where "Age" is the characteristic's name and "30to40" is the characteristic's value.

QualifierSet: A “QualifierSet” is a complete set of Qualifier elements that describe an object in a logical service's basic query data model.

UniqueQualifier: A “Unique Qualifier” is a set of one or more Qualifier elements that – taken together – uniquely identify an object in a logical service's basic query data model. To "uniquely identify" an object means that no other object in the data store has the same UniqueQualifier. However, an object may have more than one UniqueQualifier.

UniqueQualifierDeclaration: The “Unique Qualifier Declaration” defines the set of Qualifier element characteristic name identifiers comprising a unique qualifier. When each specified Qualifier element's named characteristic identifier is paired with a value (i.e., a name/value pair which is a Qualifier element), the result is a service data model UniqueQualifier.

5.0 ABBREVIATIONS

All [SCTE130-2] abbreviations are included herein. See [SCTE130-2] for additional information.

CDATA: (Character **DATA**) XML data that is not parsed. CDATA carries markup examples that would otherwise be interpreted as XML because of the tags.

6.0 GENERAL INFORMATION SERVICE OVERVIEW

Several logical services defined by SCTE 130 provide registration, query and notification services to service consumers.

Since the semantic elements of a logical service interface providing these functions are common to more than one logical service, the specification of these common semantic elements has been factored out into this separate General Information Service (GIS) specification.

For example, the Placement Opportunity Information Service (POIS) and the Subscriber Information Service (SIS) specifications may incorporate the registration, query and notification components specified herein.

Figure 1 provides an abstract representation of the GIS interface that may be implemented by an SCTE 130 logical service.

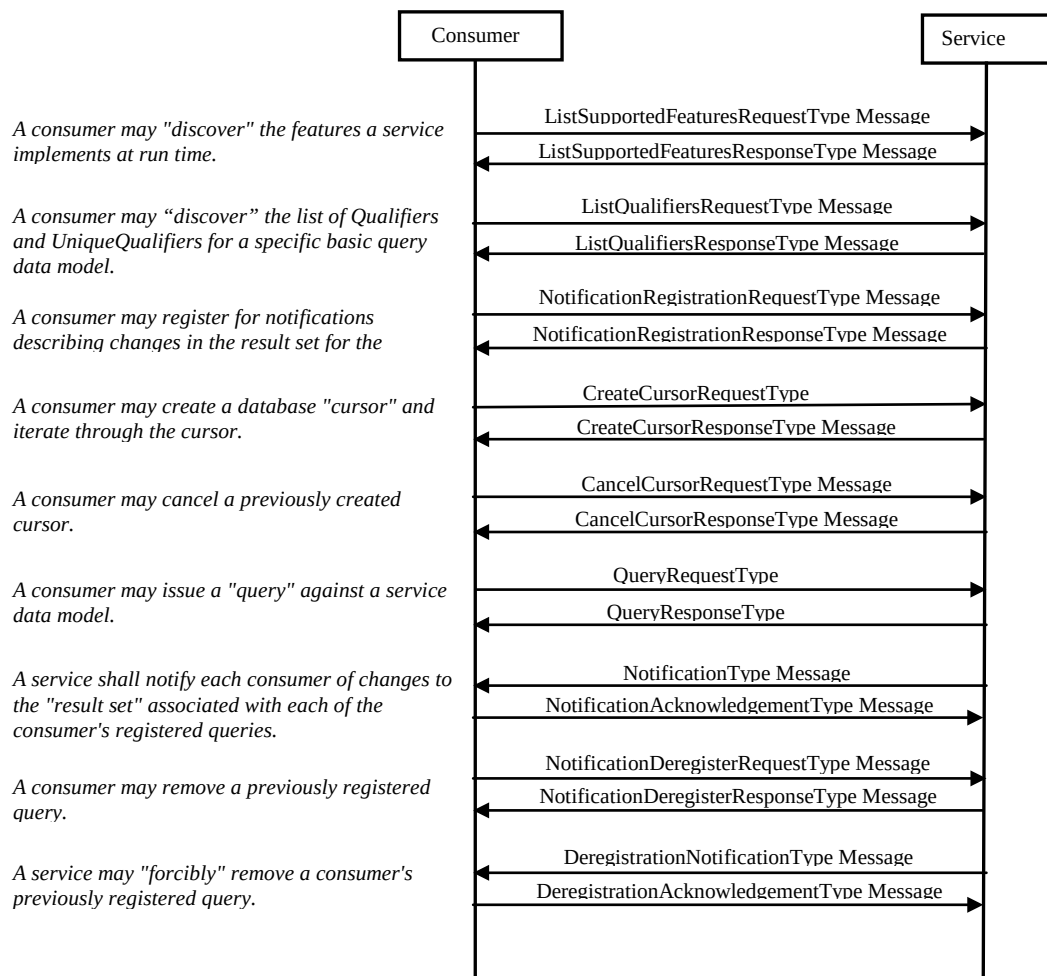


Figure 1. General Information Service Interface

The GIS interface may be implemented by a logical service that provides data to other logical services. This specification describes the messaging that may be used by a consumer to retrieve data by querying one or more data models supported by a logical service.

Since a logical service's data model may be an abstract representation of a large data store, a consumer may optionally need to locally cache data for performance optimization. The GIS specification provides support for consumer local data caching by allowing the consumer to "register" notification queries with a logical service. In this case, the result set for each query shall be returned to the consumer via "notification" messages reflecting differences (add, delete, or modification). Changes in the logical service's underlying data store while the registered query is "active" shall result in notification messages whose content reflects the changes.

Two kinds of query interfaces may be used to discover information about a logical services' data – a “basic query” and an “advanced query”.

Basic queries leverage a limited name/value regular expression grammar. No specialized knowledge of advanced query languages is needed when formulating a basic query. An advanced query is formulated using XQuery or XPath or any other query language.

A logical service may choose to implement either the basic query interface or the advanced query interface or both. See Section 12.3 for additional information on advanced query language support and Section 12.0 for additional information on basic and advanced query mechanisms.

Logical services implementing the interface described herein may support more than one service data model and each service data model may be queried with either the basic query mechanism or the advanced query mechanism or both. When more than one service data model is supported, the first service data model listed in the ListSupportedFeaturesResponse message type is referred to as the default data model (i.e., the first ServiceDataModelProfile element is distinguished as the default service data model). The default data model shall be used when no service data model is specified.

A service data model which may be queried using the basic query mechanism shall be constrained to describing a set of objects decorated with one or more name/value pairs – called “Qualifiers”. An additional constraint on the basic query data model is each object in the data model shall be uniquely identifiable by at least one subset of its name/value pairs – called the “UniqueQualifier”. The unique qualifier composition is described using the UniqueQualifierDeclaration element which provides the characteristic name identifier of the name/value pair using a sequence of QualifierDeclaration elements. See Section 12.27 and Section 12.19 for additional information on the UniqueQualifierDeclaration and QualifierDeclaration elements respectively.

As a concrete example of a service data model that could be easily queried using the basic query interface, consider a Subscriber Information Service (SIS) providing information to other logical services about subscribers. In this case, data representing each subscriber might be comprised of name/value pairs representing demographic information (such as age, income and preferences), location information (such as zip code, service group and street address), and equipment information (such as set-top type and MACAddress). The UniqueQualifier element for a subscriber might be comprised of one Qualifier element – the subscriber’s MACAddress.

A service data model may have more than unique qualifier (i.e., more than one UniqueQualifierDeclaration element). When a service data model has more than one UniqueQualifierDeclaration element, each UniqueQualifierDeclaration element shall have a @uniqueQualifierName attribute and no two UniqueQualifiersDeclarations shall use the same string for the @uniqueQualifierName attribute. Furthermore, the first UniqueQualifierDeclaration element listed in the BasicQueryDataModelDescription element shall be referred to as the default unique qualifier declaration (or default unique

qualifier). The default unique qualifier declaration shall be used when no specific unique qualifier declaration is referenced. See Section 12.27 and Section 12.19 for additional information on the UniqueQualifierDeclaration and QualifierDeclaration elements respectively.

This specification also describes cursor based operations for both basic and advanced queries. Consumers may request the creation of temporary static, cursor based information with specified lifetimes, and then iterate over the information until the cursor life cycle completes (i.e., expires). See section 12.11 for additional information on cursors.

The GIS specification includes a notification model with associated notification management functions such as registration, deregistration and listing of active registrations.

The GIS specification does not restrict the number of data stores that a logical service may use.

7.0 NOTATIONAL CONVENTIONS

7.1 Normative XML Schema

SCTE 130 Part 8 employs the same notational conventions as [SCTE130-2]. Refer to [SCTE130-2] for an explanation of notational conventions.

7.2 Document Conventions

SCTE 130 Part 8 employs the same document conventions as [SCTE130-2]. Refer to [SCTE130-2] for an explanation of document conventions. For example, the XML schema illustration is explained there.

This specification utilizes XML substitution groups for additional extensibility. XML substitution groups designate elements as substitutes for other element declarations without changing the original schema documents. Within this document, substitutable elements are graphically identified using the following illustrative technique.

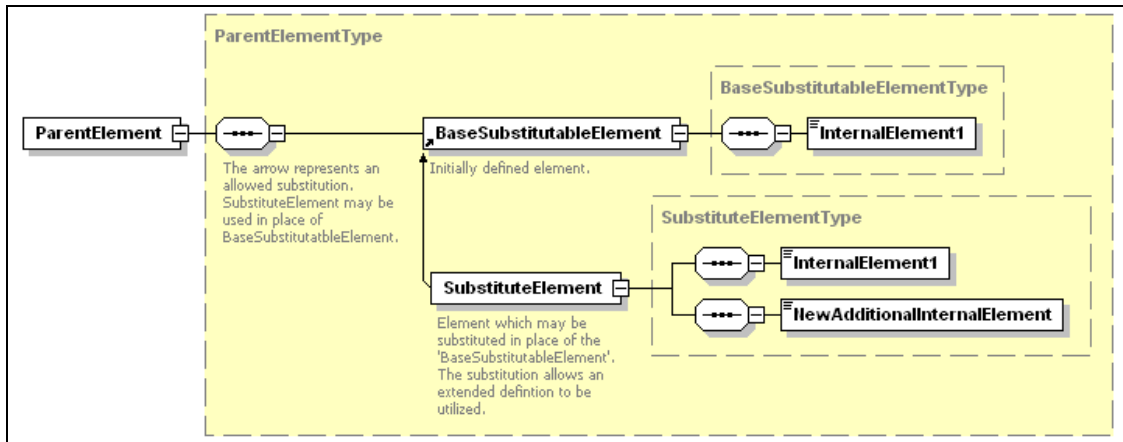


Figure 2. Substitution Group Schema Convention

In Figure 2, the element referred to as “SubstituteElement” may be used in place of the element named “BaseSubstitutableElement” provided the XML namespace declarations are included in the document as per [W3C-XSD]. The diagram’s illustrative arrow signals the reader of the possible element substitution.

8.0 PROCESSING CONVENTIONS

8.1 Unknown/Unrecognized/Unsupported XML Elements and Attributes

See [SCTE130-2] for information.

9.0 XML NAMESPACES

This specification uses the ‘gis’ prefix, as described in Table 4, for the interface associated with the specific XML namespace URI that shall be used by all implementations. Table 4 lists the prefix, the corresponding namespace, and a description of the defining specification used herein.

Prefix	Namespace	Description
core	http://www.scte.org/schemas/130-2/2008a/core	See [SCTE130-2].
gis	http://www.scte.org/schemas/130-8/2010a/gis	SCTE 130 Part 8.
xsd	http://www.w3.org/2001/XMLSchema	XML foundation. See W3C.

Table 4. XML Namespace Declarations

Unless otherwise stated, all references to XML elements illustrated in this document are from the ‘gis’ namespace. Elements from other namespaces shall be prefixed with the name of the external namespace, e.g. <core:XXX>.

10.0 GIS MESSAGE TYPES

The following topics are covered by [SCTE130-2] and this specification considers all aspects defined therein to be normative and applicable herein. See [SCTE130-2] for additional information.

- Message format
- XML message carriage
- Transport mechanisms
- Message error handling

The GIS message interface shall include the messages defined in [SCTE130-2] and logical services that incorporate the GIS message interface shall conform to the messaging transport mechanism described in [SCTE130-7].

Table 5 identifies additional SCTE 130 Part 8 (GIS) specific message types.

Message	Description
ListSupportedFeaturesRequestType	Request to retrieve a list of a logical service's supported features
ListSupportedFeaturesResponseType	Response to ListSupportedFeaturesRequest
ListQualifiersRequestType	Request to retrieve a list of names that may be used to construct basic queries using name/value pairs
ListQualifiersResponseType	Response to ListQualifiersRequest
ListNotificationRegistrationRequestType	Request to list existing registrations
ListNotificationRegistrationResponseType	Response to ListNotificationRegistrationRequest
NotificationRegistrationRequestType	Registration request for notification
NotificationRegistrationResponseType	Response to NotificationRegistrationRequest
NotificationType	Notification message indicating a change to the result set of a registered query
NotificationAcknowledgementType	Response to Notification
CreateCursorRequestType	Request to create a cursor
CreateCursorResponseType	Response to CreateCursorRequest
CancelCursorRequestType	Request to cancel an existing cursor
CancelCursorResponseType	Response to CancelCursorRequest
QueryRequestType	Request to acquire records from the GIS
QueryResponseType	Response to QueryRequest
NotificationDeregisterRequestType	Request to de-register a previously accepted registration
NotificationDeregisterResponseType	Response to NotificationDeregisterRequest
DeregistrationNotificationType	Deregistration notification
DeregistrationAcknowledgementType	Deregistration notification acknowledgement

Table 5. GIS Specific Message Types

Because this specification defines an abstract messaging interface used by other logical service implementations, this specification defines the message types rather than message elements. Every logical service implementing the interface specified by SCTE 130 Part 8 shall define a set of message elements semantically equivalent to the message types defined in SCTE 130 Part 8 by sub-classing the message types listed in Table 5 and by assigning a new message element name to each message formed by prefixing a string representing the logical service type to the message type name.

For example, the Subscriber Information Service (SIS) schema definition might contain this definition for the ListSupportedFeaturesRequest message.

```
<xsd:element name="SISListSupportedFeaturesRequest" type="gis:ListSupportedFeaturesRequestType"/>
```

Example 1. SISListSupportedFeaturesRequest Message Definition

The following subsections provide specific information on SCTE 130 Part 8's use of attributes and message types defined in [SCTE130-2]. For detailed information on these attributes and message types see [SCTE130-2].

10.1 @version Attribute

SCTE 130 Part 8 specifies message types with the intent that other logical services may create service specific concrete message definitions using the SCTE 130 Part 8 message types. Thus, SCTE 130 Part 8 shall not specify a value for the @version attribute. Logical services that derive messages from the SCTE 130 Part 8 message types shall set the @version attribute to a value that reflects the logical service's revision.

10.2 Request Base Message Type

All GIS top level **request** message types are derived from the core:Msg_RequestBaseType abstract base message type. See [SCTE130-2] for details on the attributes and elements contained in this base message.

10.3 Response Base Message Type

All GIS top level **response** message types are derived from the core:Msg_ResponseBaseType abstract base message type. See [SCTE130-2] for details on the attributes and elements contained in this base message.

10.4 Notification Base Message Type

All GIS top level **notification** message types are derived from the core:Msg_NotificationBaseType abstract base message type. See [SCTE130-2] for details on the attributes and elements contained in this base message.

10.5 Acknowledgement Base Message Type

All GIS top level **acknowledgement** message types are derived from the core:Msg_AcknowledgementBaseType abstract base message type. See [SCTE130 2] for details on the attributes and elements contained in this base message.

10.6 Messages Requiring Notification Registration

Consumers are required to register with any logical service implementing the GIS interface in order to receive Notification messages. The logical service shall not send a NotificationDeregisterRequest message type to any consumer when the consumer has no notification requests registered with the logical service. All other GIS message types do not require registration and may be sent at any time.

10.7 Default Logical Service Channel EndPoint

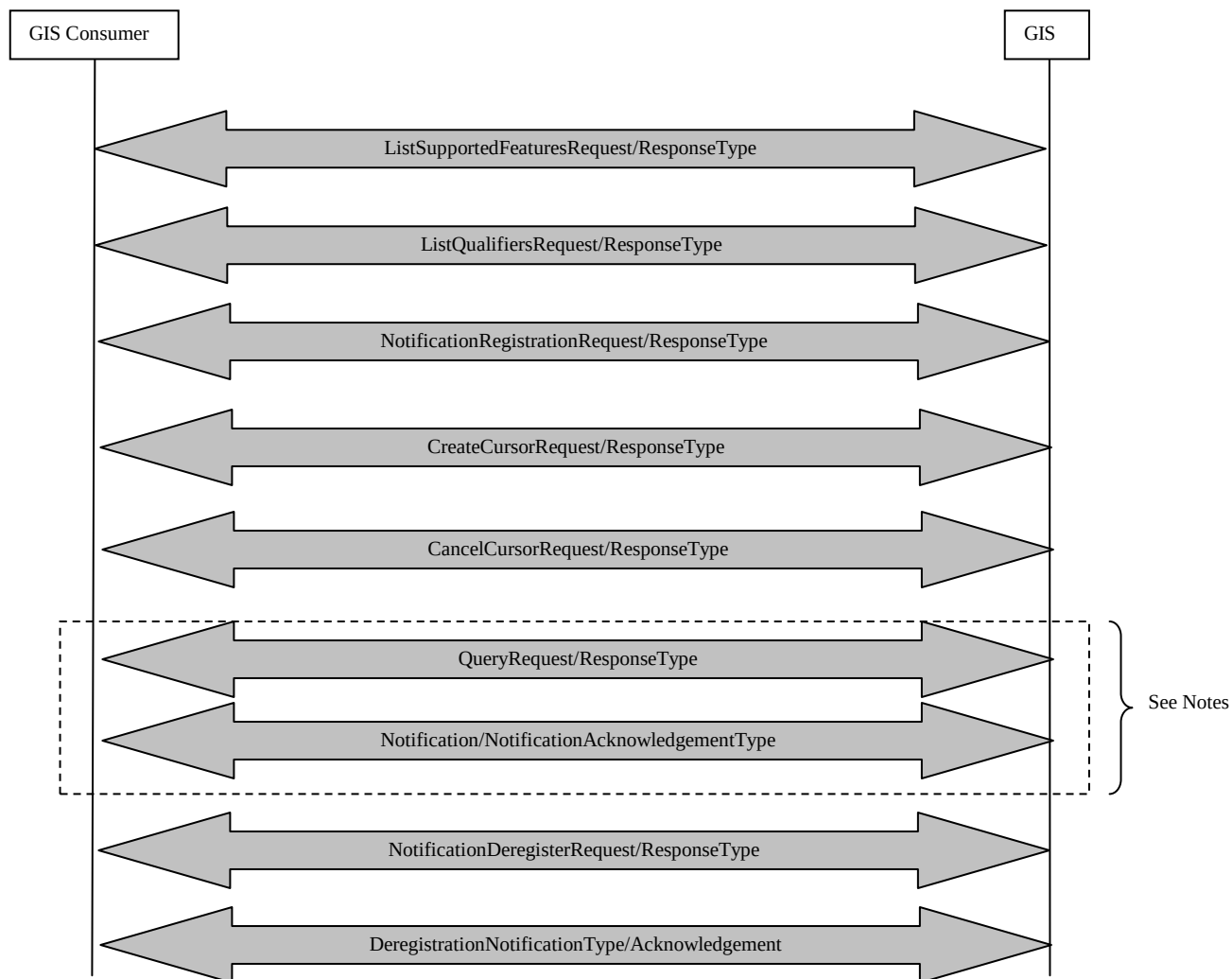
A logical service incorporating the GIS interface shall support the receipt of the ListSupportedFeaturesRequestType message type. Typically, this message type is received on a default (or well known) logical service channel endpoint address. Other GIS messages types may also be offered on the same logical service channel endpoint or they may be optionally offered on additional logical service channel endpoints. In order to discover all GIS provided logical service channel endpoints, consumers should complete a ListSupportedFeaturesRequestType/ListSupportedFeatureResponseType message type exchange.

See Section 11.1.2 for additional information on the ListSupportedFeaturesResponseType message type.

See [SCTE130-2] for a complete description of the terms (message, endpoint and logical service channel).

11.0 GIS MESSAGE EXCHANGE

The following diagram illustrates a typical message exchange between a GIS consumer and the GIS.



Notes: The Query and Notification exchange may be called or may occur repeatedly between logical service channel setup and tear down.

Figure 3. GIS Message Exchange

Figure 3 illustrates the message exchanges specific to the GIS. The service check and service status message exchanges defined in [SCTE130-2] are not depicted in this illustration.

11.1 ListSupportedFeaturesRequest and ListSupportedFeaturesResponse Message Types

The ListSupportedFeaturesRequest and ListSupportedFeaturesResponse message types allow consumers to inquire about the service data models and advanced query languages supported by a logical service.

11.1.1 ListSupportedFeaturesRequest Message Type

The ListSupportedFeaturesRequest message type allows the consumer of a logical service to inquire about the service data models and advanced query languages supported by the service.

The XML schema definition for this message is illustrated in Figure 4.

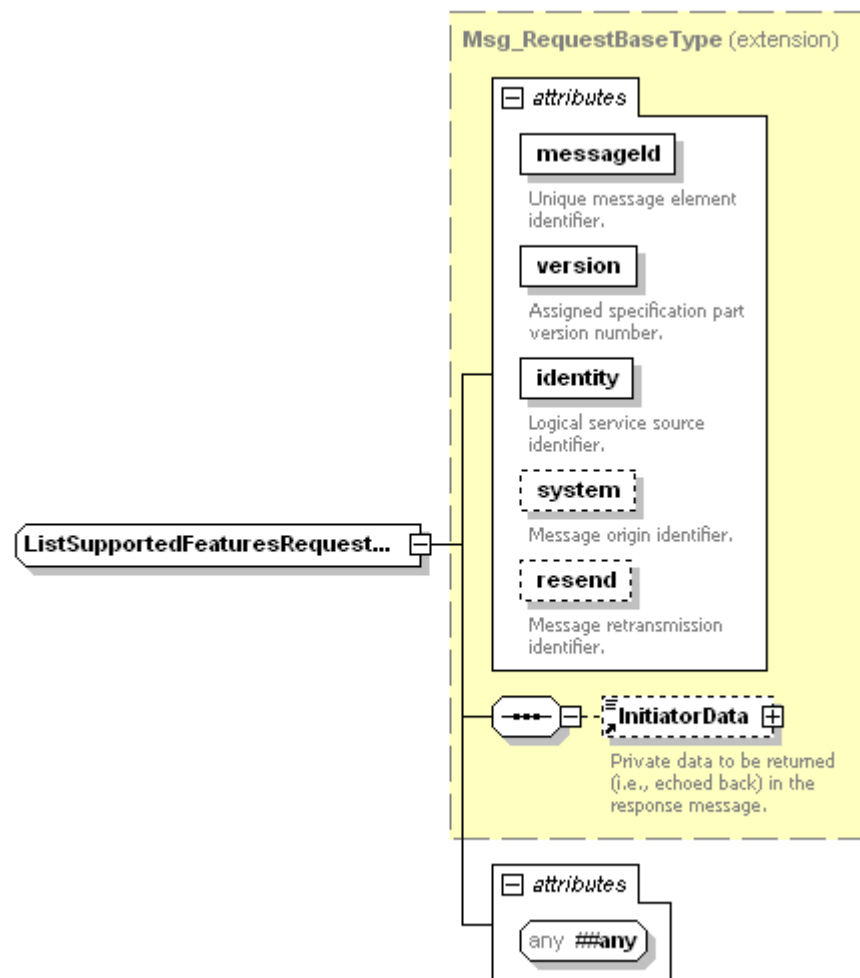


Figure 4. ListSupportedFeaturesRequestType XML Schema

The ListSupportedFeaturesRequest message type defines no elements in addition to those that are already defined by core:Msg_RequestBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

11.1.2 ListSupportedFeaturesResponse Message Type

If a ListSupportedFeaturesRequest type message is successful, the matching ListSupportedFeaturesResponse message type shall contain, at a minimum, a single core:Callout element containing one or more core:Address element(s).

If a ListSupportedFeaturesRequest type message is not successful, the matching ListSupportedFeaturesResponse message type shall not contain a core:Callout element.

See [SCTE130-2] for additional information on the core:Callout element. The XML schema definition for this message is shown in Figure 5.

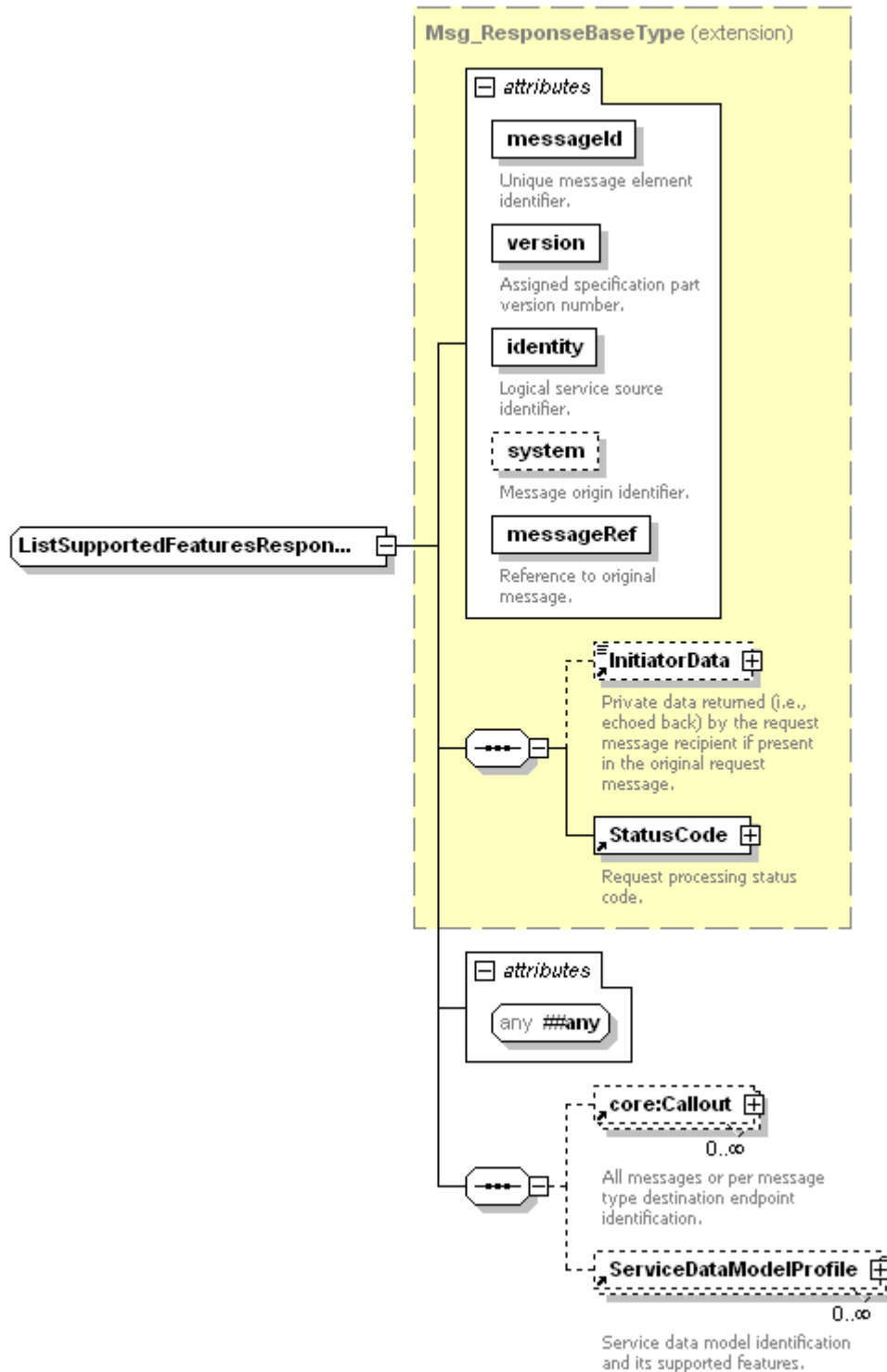


Figure 5. ListSupportedFeaturesResponseType XML Schema

The ListSupportedFeaturesResponse message type is derived from the core namespace base type core:Msg_ResponseBaseType and defines the following

attributes and elements in addition to those already defined by core:Msg_ResponseBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

core:Callout [Optional] — Zero or more core:Callout elements specifying the service channel message endpoint(s). See [SCTE130-2] for additional information on the core:Callout element.

Table 6 contains the values for the @message attribute of the core:Callout element. Values for the @message attribute should be used exactly as defined in this table where "xxx" is a string representing the logical service as shown in Example 1.

@message Attribute Value	Description
xxxNotificationRegistrationRequest	Destination endpoint for message of type NotificationRegistrationRequestType
xxxNotificationDeregisterRequest	Destination endpoint for message of type NotificationDeregisterRequestType
xxxListNotificationRegistrationRequest	Destination endpoint for message of type ListNotificationRegistrationRequestType
xxxListQualifiersRequest	Destination endpoint for message of type ListQualifiersRequestType
xxxCreateCursorRequest	Destination endpoint for message of type CreateCursorRequestType
xxxCancelCursorRequest	Destination endpoint for message of type CancelCursorRequestType
xxxQueryRequest	Destination endpoint for message of type QueryRequestType
ServiceStatusNotification	Destination endpoint for message of type core:ServiceStatusNotificationType
...	User defined address endpoint outside of the scope of this specification. The string shall be prefixed with the text "private:"

Table 6. ListSupportedFeaturesResponseType core:Callout @message values

All message values listed in Table 6 and not present in the ListSupportedFeaturesResponseType message's core:Callout XML element sequence shall be available through the default endpoint if present. (The default endpoint is identified by a core:Callout element not having the @message attribute.) See [SCTE130-2] for additional information.

ServiceDataModelProfile[Optional] — A list of zero or more ServiceDataModelProfile elements – one for each service data model supported by the logical service. The first ServiceDataModelProfile element is referred to as the default service data model and shall be used when no service data model is specified. For more information on the ServiceDataModelProfile element see Section 12.25.

11.2 ListQualifiersRequest and ListQualifiersResponse Message Types

The ListQualifiersRequest and ListQualifiersResponse message types allow consumers to discover the qualifiers associated with service data models that may be queried using the basic query interface. If a specific service data model does not support the basic query interface, the ListQualifiersResponse message type shall return an error (specified later in this section).

11.2.1 ListQualifiersRequest Message Type

The ListQualifiersRequest message type allows the consumer of a logical service that implements the GIS interface to inquire about the unique qualifiers, the qualifier names and the qualifier descriptions used by a logical service's data model.

The XML schema definition for this message is shown in Figure 6.

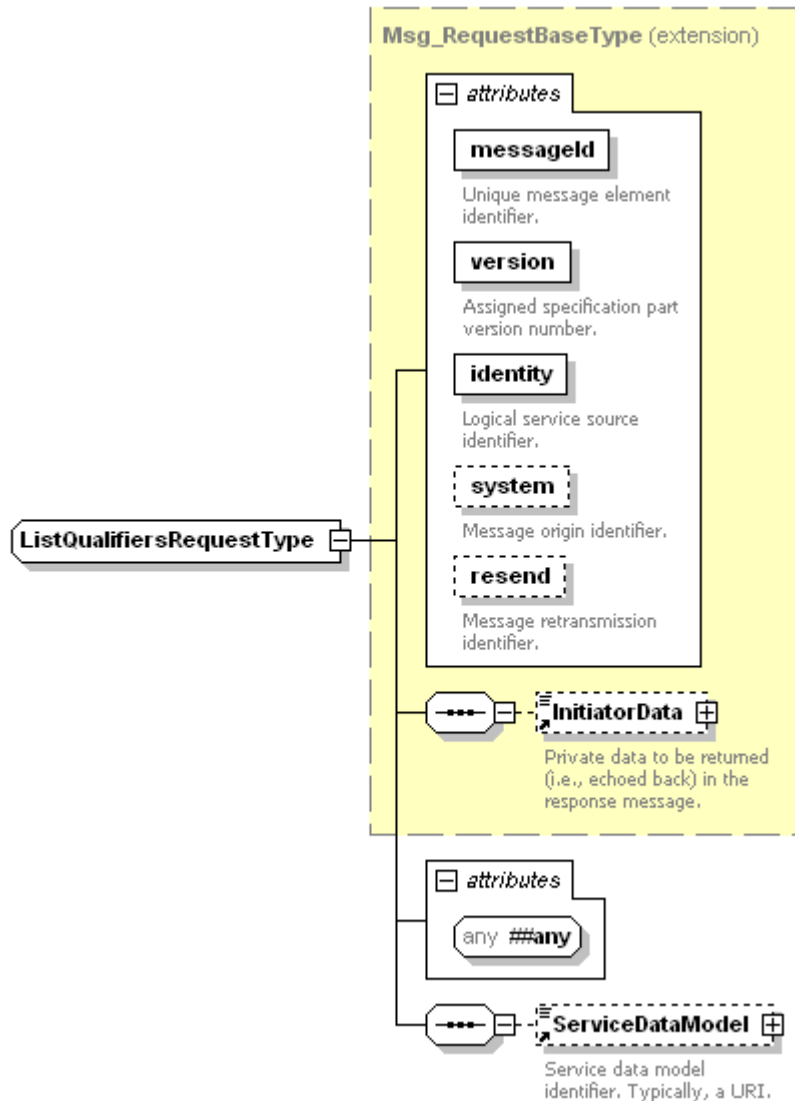


Figure 6. ListQualifiersRequestType XML Schema

The ListQualifiersRequest message type defines the following attributes and elements in addition to those that are already defined on the abstract base message core:Msg_RequestBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

ServiceDataModel [Optional] — The ServiceDataModel element contains a unique identifier (usually a URI) for a service data model that may be accessed using the basic query interface. If the ServiceDataModel element is not present,

the default service data model shall be selected. For additional information on the ServiceDataModel element, see Section 12.24

11.2.2 ListQualifiersResponse Message Type

If the service data model specified by the ServiceDataModel element in the ListQualifiersRequest type message is known to the logical service and is a basic query data model, the matching ListQualifiersResponse message type shall contain a single BasicQueryDataModelDescription element describing the service data model. If the service data model is known but does not support a basic query interface (i.e., it only supports advanced queries), the ListQualifiersResponse message type shall return an error where the core:StatusCode element shall contain a @detailCode attribute set to the value core:NotSupported.

If the service data model specified by the ServiceDataModel element in the ListQualifiersRequest message is not known to the logical service, the matching ListQualifiersResponse message type shall not contain a BasicQueryDataModelDescription element and the core:StatusCode element shall contain a @detailCode attribute set to the value core:ResourceNotFound.

The XML schema definition for this message is illustrated in Figure 7.

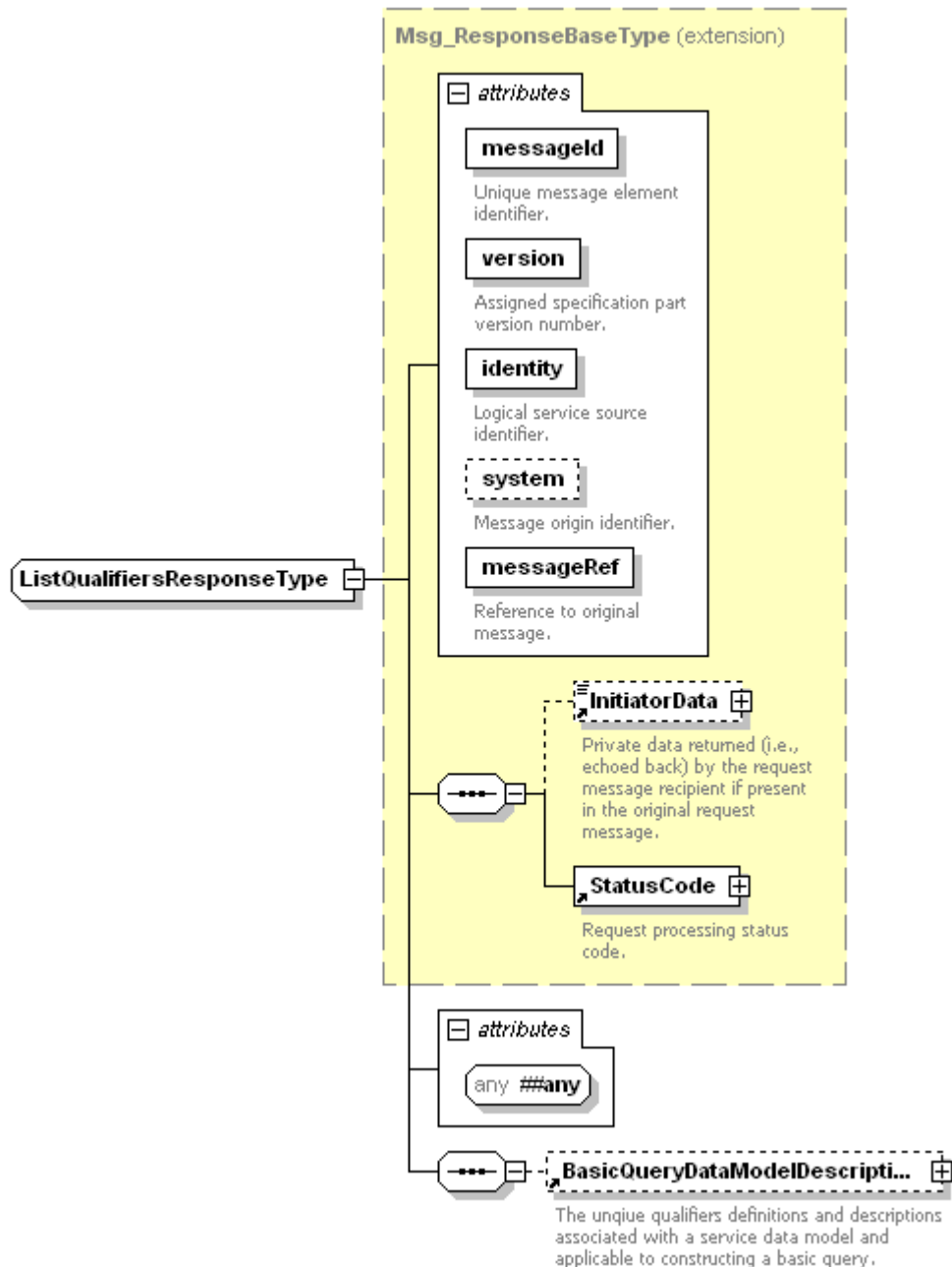


Figure 7. ListQualifiersResponseType XML Schema

The ListQualifiersResponse message type is derived from the core namespace base type core:Msg_ResponseBaseType and defines the following attributes and elements in addition to those that are already defined on the abstract base message core:Msg_ResponseBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete

messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

BasicQueryDataModelDescription [Optional] — The BasicQueryDataModelDescription element defines the service data model's unique qualifiers and provides descriptions for a group of qualifiers. For more information on the BasicQueryDataModelDescription element see Section 12.7.

11.3 ListNotificationRegistrationRequest and ListNotificationRegistrationResponse Message Types

A GIS consumer may inquire about current notification registrations by using the ListNotificationRegistrationRequest message type. The logical service shall respond to the ListNotificationRegistrationRequest type message with a ListNotificationRegistrationResponse type message. This message exchange allows a GIS consumer to discover the active queries previously installed by one or more NotificationRegistrationRequest type messages.

There are two list registration granularity levels:

- Per logical service (i.e., via the @identity attribute)
- Per registration message

A logical service may restrict access to registration information by returning a status code equating to “not authorized” for the list registration request on a per logical service basis. The per registration message granularity level shall be supported by all GIS compliant logical services. The per logical service message granularity level may be supported by any GIS compliant logical service.

11.3.1 ListNotificationRegistrationRequest Message Types

The ListNotificationRegistrationRequest message may be issued to a logical service to retrieve information about active notification registrations.

The XML schema definition for this message is illustrated in Figure 8.

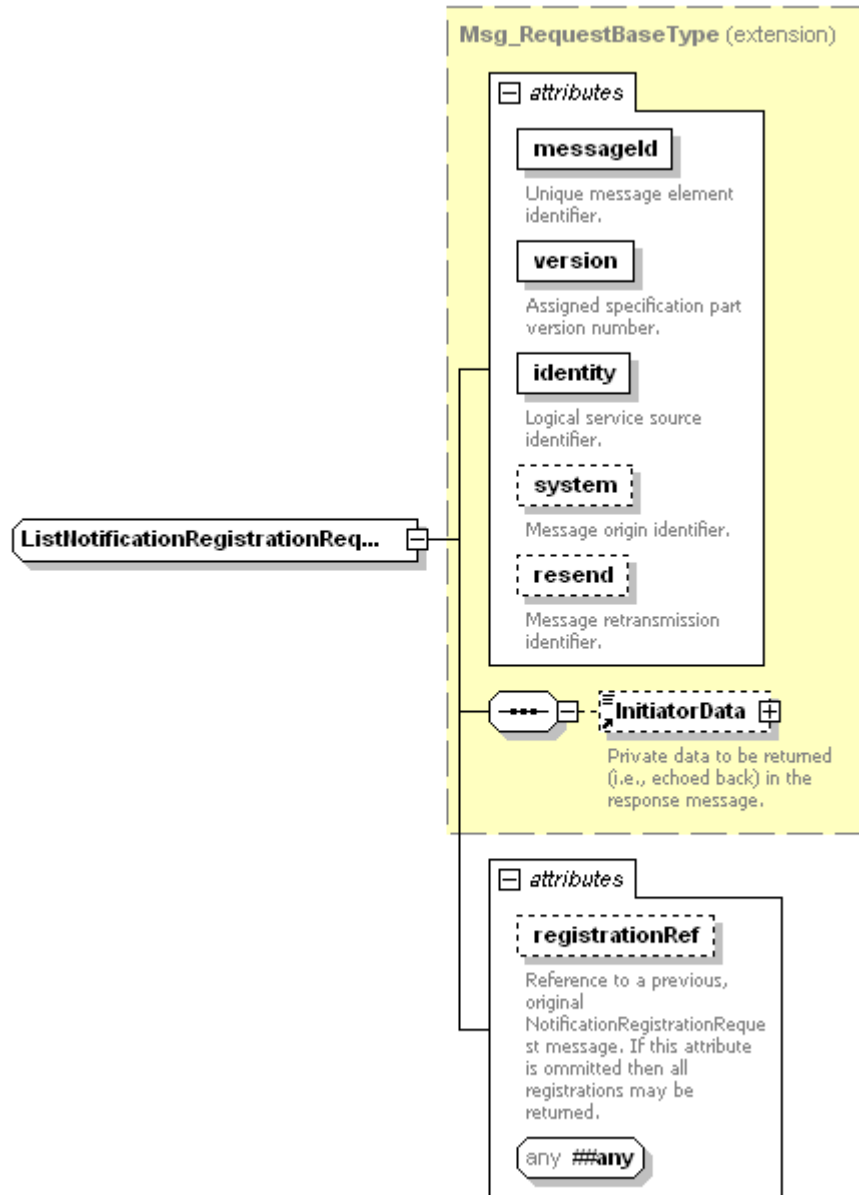


Figure 8. ListNotificationRegistrationRequestType XML Schema

The ListNotificationRegistrationRequest message returns all active registrations associated with a specific logical service source as identified by the @identity attribute when the @registrationRef attribute is omitted. If only a specific registration is to be returned, both the @identity and the @registrationRef attributes shall be included. The following table defines the inclusion relationships.

@identity	@registrationRef	Description	Access Restrictions Permitted
Included	Omitted	All registrations matching the supplied @identity value.	Yes
Included	Included	A specific registration matching the @identity and @registrationRef values.	No

Table 7. List ADS Registration Inquiry Granularity Control

The access restrictions permitted column indicates whether the status code equivalent to core:NotAuthorized may optionally be returned.

The ListNotificationRegistrationRequest message type is derived from the core namespace base type core:Msg_RequestBaseType and defines the following attributes and elements in addition to those that are already defined on the abstract base message core:Msg_RequestBaseType.

@registrationRef [Optional, core:registrationRefAttrType] — A reference to an original NotificationRegistrationRequest type message. This attribute instructs the logical service to list only the notification registration request identified by this reference. The value shall be the NotificationRegistrationRequest message's original @messageId attribute value. If this attribute is not present, then the matching ListNotificationRegistrationResponse type message shall contain information about all notifications registered for the consumer as identified by the @identity attribute.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

11.3.2 ListNotificationRegistrationResponse Message Type

The ListNotificationRegistrationResponse message type is the response pair to the previously defined ListNotificationRegistrationRequest message type.

The XML schema definition for this message is shown in Figure 9.

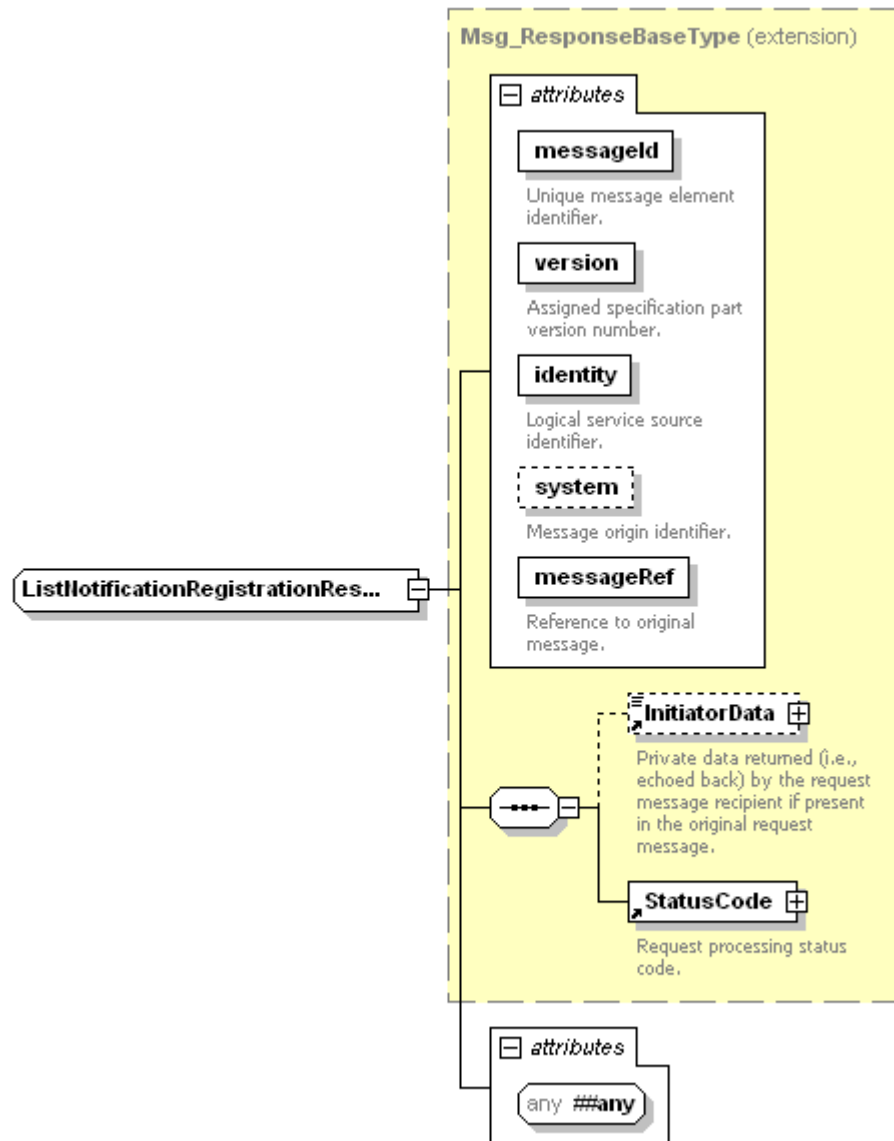


Figure 9. ListNotificationRegistrationResponseType XML Schema

The ListNotificationRegistrationResponse message type is derived from the core namespace base type core:Msg_ResponseBaseType and defines the following attributes and elements in addition to those that are already defined on the abstract base message core:Msg_ResponseBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

A logical service incorporating the GIS NotificationRegistrationRequest message type shall also include the GIS ListNotificationRegistrationRequest and GIS ListNotificationRegistrationResponse message types (typically defined as the xxxListNotificationRegistrationRequest and xxxListNotificationRegistrationResponse messages by the logical service). The xxxListNotificationRegistrationResponse message, which shall extend the GIS ListNotificationRegistrationResponse message type, shall include the logical service's xxxNotificationRegistrationRequest message as part of the element type definition and the inclusion should be specified as a sequence of zero or more instances of the xxxNotificationRegistrationRequest message.

The xxxListNotificationRegistrationResponse message type's xxxNotificationRegistrationRequest shall be a recoded copy of the accepted registration message. The message element order does not convey any information (e.g., element order does not reflect registration order). See [SCTE130-2] for compliance requirements.

If the sub-classed ListNotificationRegistrationResponse message type is unable to locate the specific registration as specified by the ListNotificationRegistrationRequest message type's @registrationRef attribute or by the @identity attribute, the response message type's core:StatusCode element shall contain the value core:ResourceNotFound (i.e., a response of zero located registrations shall return an error).

11.4 NotificationRegistrationRequest and NotificationRegistrationResponse Message Types

A logical service that implements the GIS interface shall support registration for notification message delivery. The NotificationRegistrationRequest message type allows a consumer to specify notification interests relative to a basic or an advanced query.

On receipt of an update, addition or deletion event from its underlying data store affecting the result set of a registered query, the logical service shall send a Notification type message to each matching registered consumer.

11.4.1 NotificationRegistrationRequest Message Type

The NotificationRegistrationRequest message type allows a consumer to specify a set of notification interests by registering a query against a logical service's data model. These registered queries shall be examined by the logical service relative to changes in any data relevant to the query. If any change to the data causes a change to the query result for a registered query, a notification containing the new result shall be sent to the consumer in the form of a Notification message type.

The XML schema representation of the NotificationRegistrationRequest message type is illustrated in Figure 10.

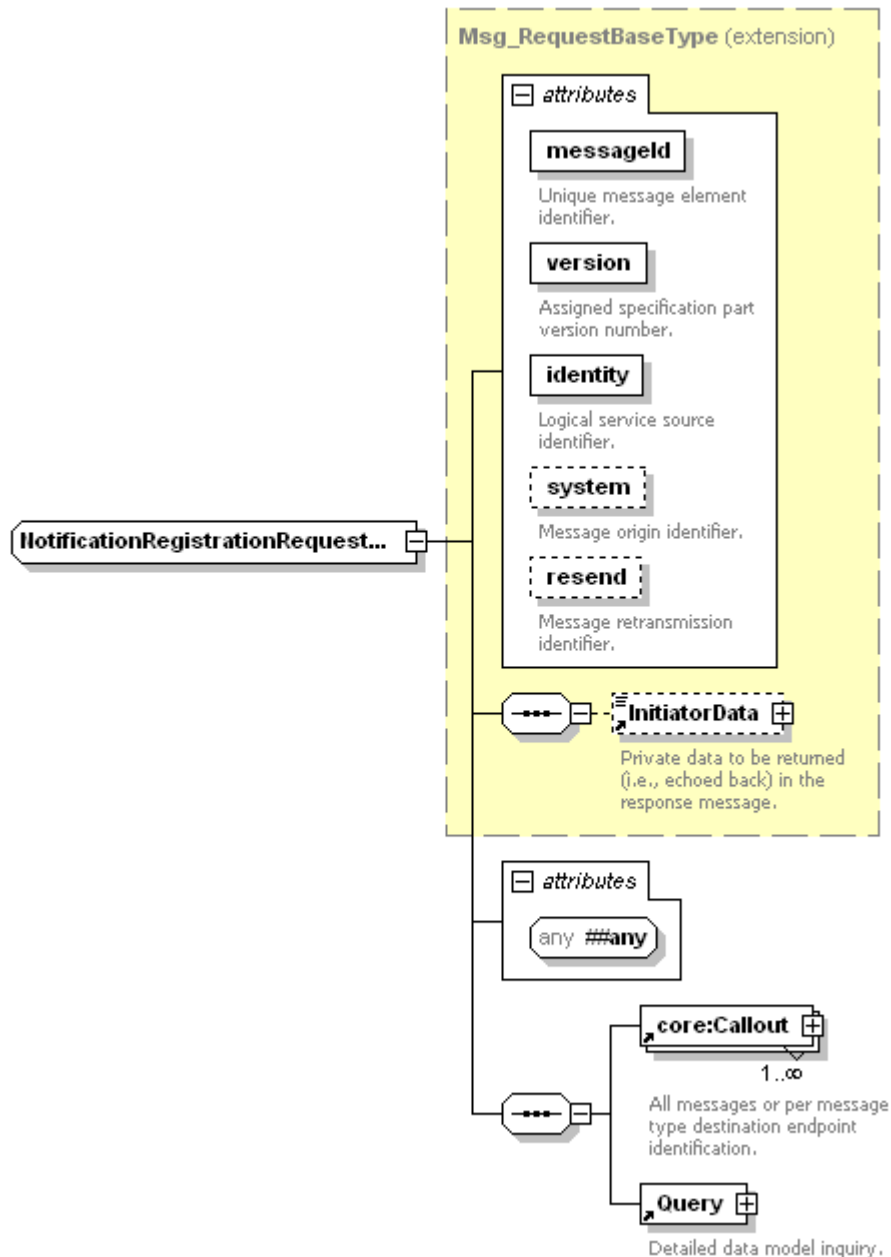


Figure 10. NotificationRegistrationRequestType XML Schema

The NotificationRegistrationRequest message type is derived from the core namespace base type core:Msg_RequestBaseType and defines the following attributes and elements in addition to those already defined on the abstract base message core:Msg_RequestBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

core:Callout [Required] — The core:Callout element provides callback message and address information to the GIS. See [SCTE130-2] for a complete description of the core:Callout element.

Before generating a NotificationRegistrationResponse type message, the logical service may send a core:ServiceCheckRequest message to a core:Address, in order to verify the validity of the callback endpoint.

An logical service implementing the GIS interface shall recognize the values listed in Table 8 as values for the core:Callout @message attribute. Values for the @message attribute should be used exactly as defined in this table where "xxx" is a string representing the logical service as shown in Example 1.

@message Attribute Value	Description
xxxNotification	Value associated with the address endpoint where Notification type messages shall be sent.
ServiceStatusNotification	Value associated with the address endpoint where core:ServiceStatusNotification messages shall be sent.
xxxDeregistrationNotification	Value associated with the address endpoint where DeregistrationNotification type messages shall be sent.
...	User defined address endpoint outside of the scope of this specification. The string shall be prefixed with the text "private:".

Table 8. NotificationRegistrationRequest core:Callout @message Values

All message values listed in Table 8 and not present in the NotificationRegistrationRequestType message's core:Callout XML element sequence shall be available through the default endpoint if present. (The default endpoint is identified by a core:Callout element not having the @message attribute.) See [SCTE130-2] for additional information. Either the xxxNotification or the default endpoint shall be present in the NotificationRegistrationRequest core:Callout element sequence.

Query [Required] — The Query element contains elements defining an inquiry for notification purposes. See Section 12.22 for additional details on the Query element.

11.4.2 NotificationRegistrationResponse Message Type

Upon completion of processing a NotificationRegistrationRequest type message, the logical service shall respond with a NotificationRegistrationResponse type message.

The XML schema diagram for the NotificationRegistrationResponse type message is shown in Figure 11.

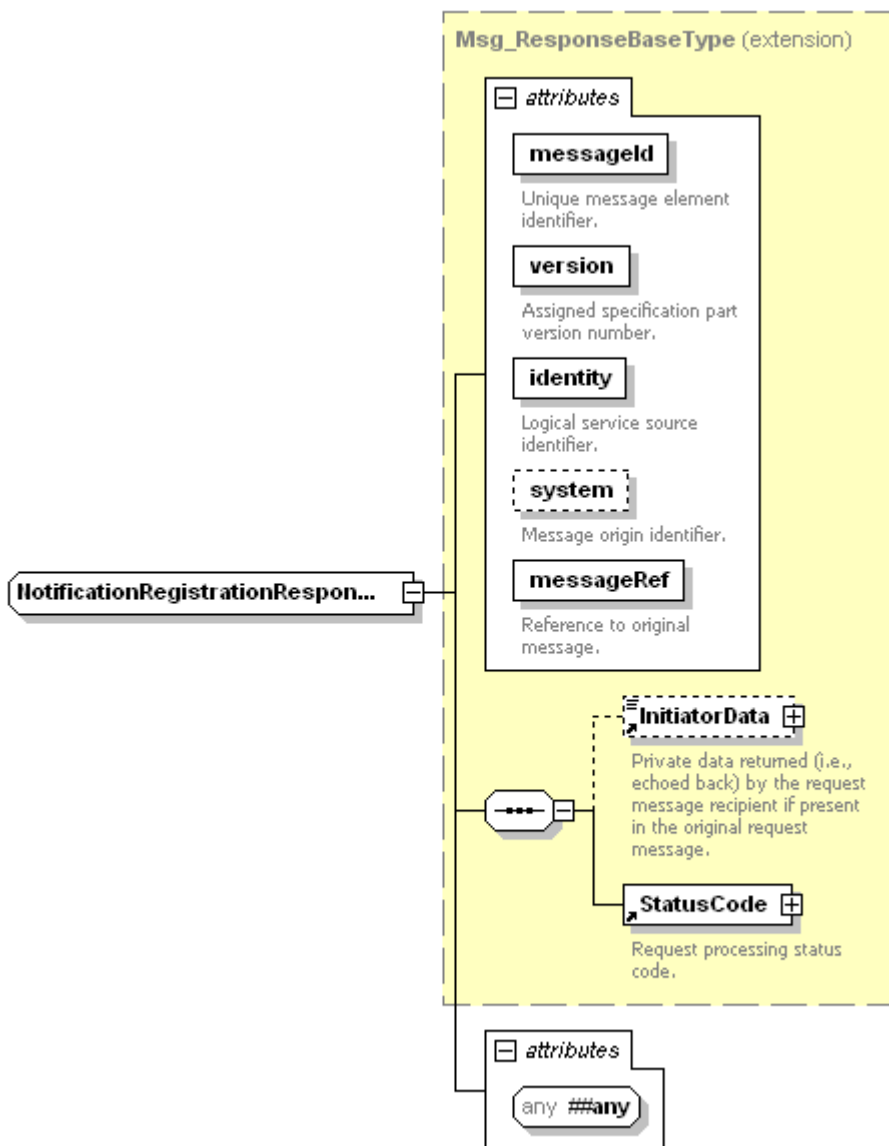


Figure 11. NotificationRegistrationResponseType XML Schema

The NotificationRegistrationResponse message type is derived from the core namespace base type core:Msg_ResponseBaseType and defines no elements in addition to those already defined on the abstract base message core:Msg_ResponseBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

11.5 Notification and NotificationAcknowledgement Message Types

A logical service that implements the GIS interface shall support the exchange of Notification and NotificationAcknowledgement type messages with registered consumers for the purpose of notifying the consumer of changes in data relevant to the consumer's registered queries.

11.5.1 Notification Message Type

Upon detection of a change in the result set returned for one or more queries registered with the service, the logical service shall send a Notification type message to any consumers with registered queries that are affected. A change to the result set of a registered query includes the set of definitions described in Table 9. The values shall appear in the @noticeType attribute exactly as they appear in Table 9 (i.e. all in lower case).

@noticeType Attribute Value	Description
update	A service's data store has been changed and the update has changed the result set of a registered query.
new	Data has been added to a service's data store and the additions have changed the result set of a registered query.
delete	Data has been deleted from a service's data store and the deletion(s) have changed the result set of a registered query.
...	User defined Notification types outside of the scope of this specification. The string shall be prefixed with the text "private:".

Table 9. Notification @noticeType Values

The XML schema for the Notification type message is illustrated in Figure 12.

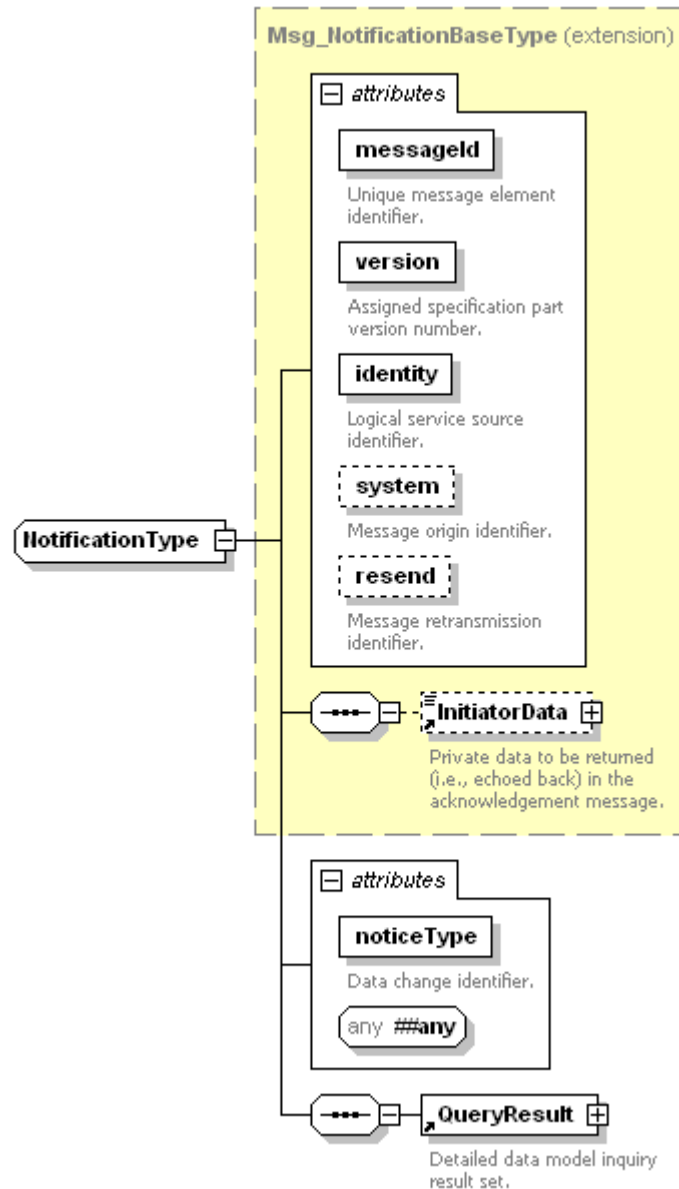


Figure 12. NotificationType XML Schema

The Notification message type is derived from the core namespace base type `core:Msg_NotificationBaseType` and defines the following attributes and elements in addition to those defined by the `core:Msg_NotificationBaseType`.

@noticeType [Required, NotificationTypeEnumeration] — The `@noticeType` attribute is an enumeration that shall contain one of the values listed above in Table 9. See Section 13.9 for additional information.

@###any [Optional] — The `@###any` attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete

messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

QueryResult [Required] — This element contains the list of service data model components subject to the change in the data and is returned in response to the consumer's registered queries. See Section 12.23 for additional information on the QueryResult element.

11.5.2 NotificationAcknowledgement Message Type

Upon the receipt of a Notification message type, a GIS consumer shall respond with a NotificationAcknowledgement message type.

The XML schema for the NotificationAcknowledgement message type is shown in Figure 13.

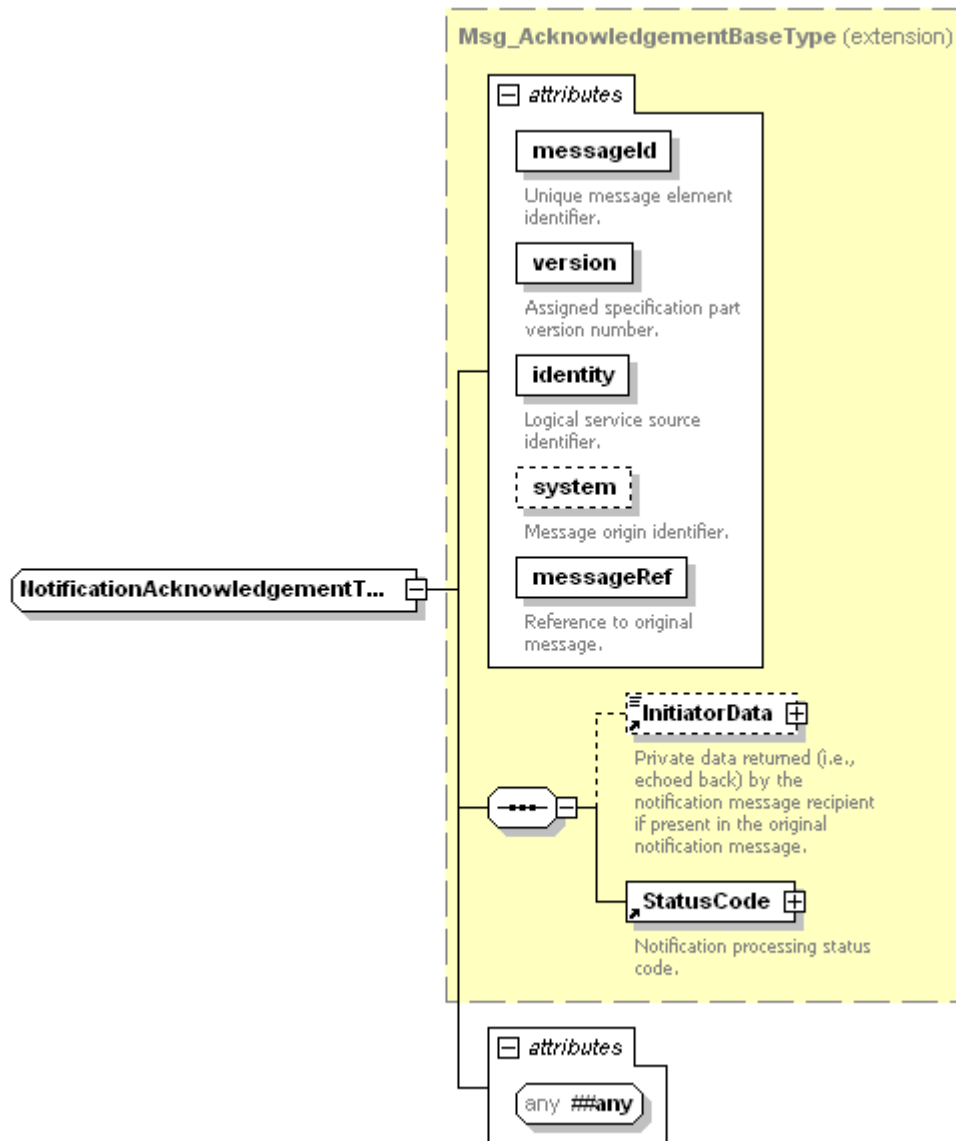


Figure 13. NotificationAcknowledgementType XML Schema

The NotificationAcknowledgement message type is derived from the core namespace base type core:Msg_AcknowledgementBaseType and defines no elements in addition to those defined by the core:Msg_AcknowledgementBaseType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

11.6 CreateCursorRequest and CreateCursorResponse Message Types

A logical service that implements the GIS interface shall support cursors of static asset information for both basic and advanced queries which shall exist for a specified duration. Upon creation of a cursor using the GIS interface, the data information in the cursor shall remain static relative to the referenced data store.

Cursors have a limited life span, which is first requested by the consumer, but may be overridden by a logical service. As part of the cursor request message, the consumer shall specify a `@cursorExpires` attribute. This date and time is a request to a logical service for a specific end date and time for the cursor identified by the `@cursorId` attribute. A logical service, in order to maintain overall system health, may choose to override a requested cursor expires end date and time and substitute a different, implementation specific, cursor expires end date and time. See Section 12.11 for additional information on cursors.

11.6.1 CreateCursorRequest Message Type

The CreateCursorRequest message type is used to create an instance of a static cursor on a logical service that implements the GIS interface.

The XML schema for the CreateCursorRequest message type is listed in Figure 14.

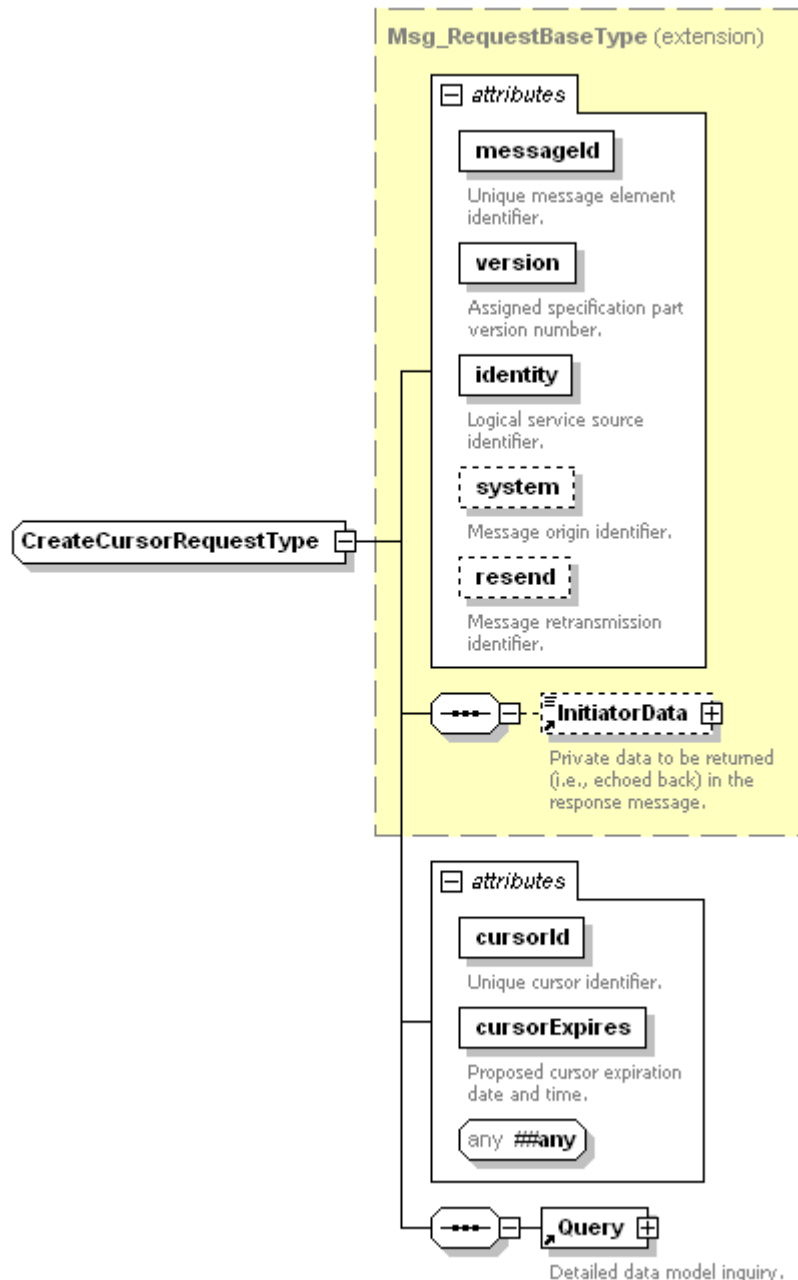


Figure 14. CreateCursorRequestType XML Schema

The CreateCursorRequest message type is derived from the core namespace base type core:Msg_RequestBaseType and defines the following attributes and elements in addition to those defined in the core:Msg_RequestBaseType.

@cursorId [Required, cursorIdAttrType] — The @cursorId attribute is a consumer generated identifier which shall be a service channel unique value. See Section 13.1 for additional information on the cursorIdAttrType type.

@cursorExpires [Required, core:dateTimeTimezoneType] — The @cursorExpires attribute is a consumer request for a cursor expiration date and time. A logical service shall not be required to create the cursor with the requested end date and time. A logical service may override the requested end date and time by returning an implementation specific end date and time that better fits within the implementation's specific design constraints. See [SCTE130-2] for information on the core:dateTimeTimezoneType.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

Query [Required] — The query element contains the necessary attributes and elements for a logical service to execute one or more inquiries against its data store. Inquiry selected items shall be added to a static cursor construct identified by the supplied @cursorId attribute. See Section 12.22 for additional information on the Query element.

11.6.2 CreateCursorResponse Message Type

Upon receipt of a CreateCursorRequest type message, a logical service shall attempt to create the required cursor and shall respond to the consumer with a CreateCursorResponse type message. If the query is not successful (i.e., the core:StatusCode value does not equate to success), the cursor shall not be established.

The XML schema for the CreateCursorResponse message type is listed in Figure 15

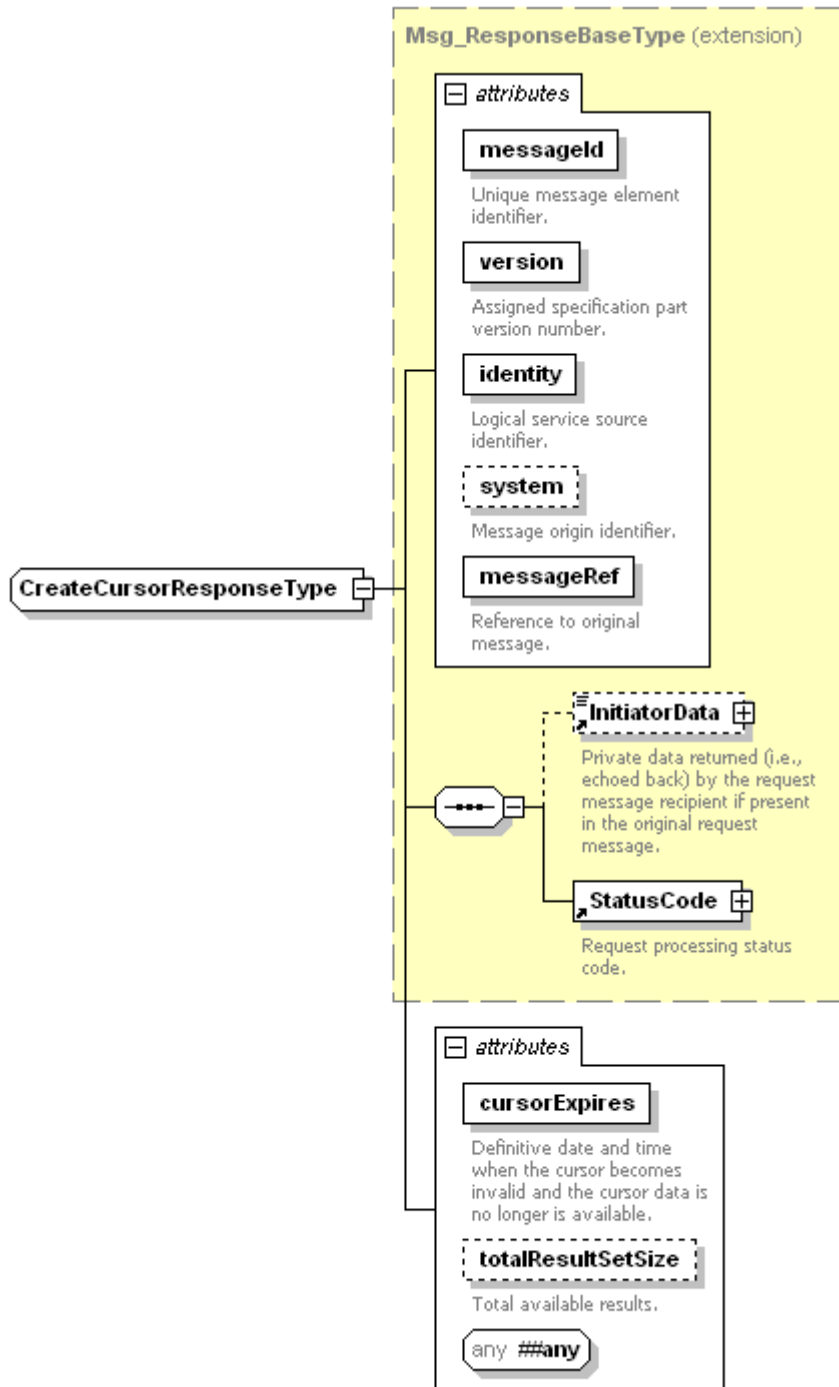


Figure 15. CreateCursorResponseType XML Schema

The CreateCursorResponse message type is derived from the core namespace base type core:Msg_ResponseBaseType and defines the following attributes and elements in addition to those already defined in the core:Msg_ResponseBaseType.

@cursorExpires [Required, core:dateTimeTimezoneType] — The @cursorExpires attribute contains the logical service determined cursor expiration date and time. The value may be either the user requested end date and time from the CreateCursorRequest message, or a logical service specified expiration date and time. See [SCTE130-2] for additional information on the core:dateTimeTimezoneType.

@totalResultSetSize [Optional, totalResultSetSizeAttrType] — The @totalResultSetSize attribute specifies the result count contained in the cursor and the value is dependent upon the Query element's inquiry composition. This attribute shall be present if the Query element contained a basic query (i.e., the Query element contained a UniqueQualifier or BasicQueryFilter element). This attribute may be present if the Query element contained an advanced query (i.e., the Query element included an AdvancedQueryFilter element). If the advanced query set the Query element's @expandOutput attribute to "false", this attribute shall be present. Otherwise, this attribute's presence is optional and its value is implementation dependent. See Section 13.17 for additional information on the totalResultSetSizeAttrType type and its contents relative to the original inquiry composition.

@##any[Optional] — The @##any attribute enables a logical service incorporating the SCTE 130 Part 8 interface to optionally extend concrete messages derived from SCTE 130 Part 8 message types with attributes not specified by the schema.

11.7 CancelCursorRequest and CancelCursorResponse Message Types

A logical service shall allow a consumer to cancel an existing cursor before the expiration time has been passed.

A logical service consumer may complete interacting with a cursor before the cursor actually expires, and may choose to terminate the cursor. Once a cursor has been terminated or has expired, a logical service may release resources associated with the cursor.

Any additional communications from the consumer that reference the canceled or expired cursor shall result in an error with the core:StatusCode @detail attribute set to the value 8001 (Cursor Undefined), as described in Table 17.

11.7.1 CancelCursorRequest Message Type

This message type allows a logical service consumer to terminate a cursor before the cursor's expiration time.

The XML schema for the cancel cursor request message type is illustrated in Figure 16.