

# **Comprehensive Large Array-data Stewardship System (CLASS)**

## **Common Submission Interface Control Document**

**September**



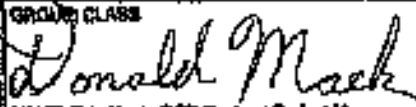

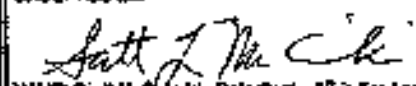
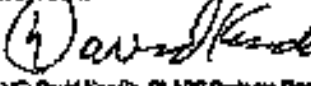

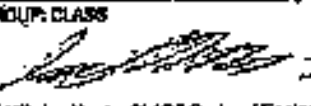




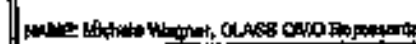
**2013**

**Prepared by:**  
**Diversified Global Partners JV LLC (DGP)**

**Prepared for:**  
**U.S. Department of Commerce**  
**National Oceanic and Atmospheric Administration (NOAA)**  
**National Environmental Satellite, Data, and Information Service (NESDIS)**

## Approval Page

Comprehensive Large Array-data  
Stewardship System (CLASS)Common Submission  
Interface Control Document

PROGRAM: CLASS		DOCUMENT RELEASE DATE: May 2013	
APPROVAL			
GROUP: CLASS  NAME: Don Mack, OSP Contract Project Manager	GROUP: CLASS  NAME: Cora Lino, OSP Deputy Contract Project Manager		
GROUP: CLASS  NAME: Scott McConachy, Data Center Migration Lead	GROUP: CLASS  NAME: David Kandy, CLASS Systems Engineering Lead		
GROUP: CLASS  NAME: Ken Tucker, CLASS Program Manager	GROUP: CLASS  NAME: Jay Woods, CLASS Systems Engineer		
GROUP: CLASS  NAME: Scott Kogut, CLASS ISSO	GROUP: CLASS  NAME: Nancy Ritzky, Aurora Branch Chief, NOCC		
GROUP: CLASS  NAME: Kelly Prandrup, ICD Chief, NOCC	GROUP: CLASS  NAME: John Ralph, Technical Director, NOCC		
CDM REVIEW BY:  NAME: Michele Wagner, CLASS CDM Representative			





## Table of Contents

<b>Section 1.0</b>	<b>Introduction .....</b>	<b>1</b>
1.1	Background and Purpose .....	1
1.2	Scope .....	1
1.3	Constraints .....	2
1.4	Acronym Usage .....	2
1.5	Document Organization .....	2
1.6	References and Related Documents .....	3
1.7	Document Maintenance .....	3
 <b>Section 2.0</b>	 <b>Interface Overview .....</b>	 <b>4</b>
2.1	Data Transfer Overview .....	4
2.2	Validation Overview .....	5
 <b>Section 3.0</b>	 <b>Submissions to CLASS.....</b>	 <b>6</b>
3.1	Collection Registration .....	6
3.1.1	Collection Information .....	7
3.1.2	Collection Processing .....	7
3.1.3	File Submission Processing Based on Collections .....	7
3.2	Ingest Transfers and Notifications .....	8
3.3	Submission Manifest .....	8
3.3.1	Submission Manifest File Naming Convention .....	9
3.3.2	Required and Optional Elements in the Submission Manifest.....	9
3.3.3	Submission Manifest Element Definitions .....	12
3.4	Data Files.....	15
3.4.1	Data File Validation .....	15
3.4.2	Data File Duplicates .....	16
 <b>Section 4.0</b>	 <b>Notifications to Data Providers.....</b>	 <b>17</b>
4.1	Submission Manifest Notifications .....	17
4.2	Data File Notifications .....	17
4.2.1	Delivery of Ingest Report .....	17
4.2.2	Ingest Report File Naming Convention.....	18
4.2.3	Ingest Report States Defined.....	18
4.2.4	Information Conveyed in an Ingest Report .....	18

## Appendices

Appendix A. Submission Manifest XML Schema .....	A-1
Appendix B. Example Submission Manifest .....	B-1
Appendix C. Ingest Report Schema.....	C-1
Appendix D. Example Ingest Report .....	D-1

Appendix E. Acronyms.....E-1

## List of Figures

Figure 1 – Data Transfers ..... 4

## List of Tables

Table 1 – Related and Reference Documents ..... 3  
Table 2 – Data Provider - CLASS Transfer Descriptions..... 5  
Table 3 – Collection Information Elements..... 7  
Table 4 – Submission Manifest File Naming Convention Elements ..... 9  
Table 5 – Required Submission Manifest Elements Included Once..... 9  
Table 6 – Required and Optional Submission Manifest Elements for each <ingestfile>..... 9  
Table 7 – Element Definitions ..... 12  
Table 8 – Options Available for Handling Duplicate Data Files..... 16  
Table 9 – UUID Element Definition..... 19

## Section 1.0 Introduction

This document, the CLASS Common Submission Interface Control Document, describes the interface between data providers and the CLASS System. This interface will use a common process to submit disparate data sets to CLASS for archival storage.

### 1.1 Background and Purpose

NOAA provides timely access to global environmental data from satellites and other sources through its three data centers, known collectively as the NOAA National Data Centers. The data archives amassed by these centers provide a record of Earth's changing environment, and support numerous research and operational applications worldwide.

CLASS is NOAA's enterprise-wide information technology system designed to support long-term, secure preservation and standards-based access to environmental data collections and information. CLASS supports the ingest, quality control, archival storage of, and public access to data and science information. The system is modeled to support the NOAA-adopted OAIS-RM, which identifies high-level roles and responsibilities of various archival components and illustrates the connections between functional entities in order to fulfill archive requirements.

This document describes a new CLASS submission interface that provides a standard mechanism to submit disparate data sets for archival storage in CLASS.

### 1.2 Scope

This document describes the interface to be used by multiple data providers to submit information for ingest into the CLASS Archival Storage system and to receive information from CLASS on the outcome of their submissions. The initial set of data providers who will make use of this interface includes the three NOAA National Data Centers, NCDC, NGDC, and NODC. This document defines how these data providers can submit data to CLASS Ingest and Archival Storage.

Submission of data through this interface is unrelated to how the data can be subsequently accessed. It is the data centers' responsibility to provide access to data archived using the CS process. This access by customers will be provided by data center client software tools that make use of the CLASS M2M interface.

Definitions are established for data information exchange in the following areas:

- Collection Registration
- CLASS Ingest Interface
- Data provider submission manifest Schema Definitions
- CLASS Ingest Report Schema Definition

Certain implementation details such as personnel contact information, Internet host addresses, and values for operator-tunable parameters, are not included in this document. These details will be recorded in the applicable data provider-specific ICDs and Operations Agreements between each data provider and CLASS.



This document does not include changes that may be needed in the M2M interface or the Archive Manager Metrics Reports, or the implementation of UUID generation. The requirements for these functions have been negotiated independently, and implementation of the functions will be allocated to the appropriate systems.

### 1.3 Constraints

The following constraints may impact the processes described in this document.

- Based on the current capabilities of replication servers at the CLASS nodes, the CS capacity will initially be limited to 3 TB per day for each CLASS node.
- The CS interface will only support single-file SIPs. Multi-file submissions will not be supported (e.g., two files cannot be submitted together as one entity). For example, a submission of a header file, data file, and trailer file that should be considered as a unit (AIP) will not be supported by CS.
- Multiple single-file submissions can be included in a single submission manifest.
- In the initial implementation, CS will not catalog any files inside of a container or aggregate file (e.g., a tar file). The contents of a container cannot be searched or ordered independently. The file or “object” that is staged for ingest into the CLASS Archival Storage system will be exactly what is delivered when that object is ordered.
- CS will not provide a mechanism to support cross-referencing between files.
- Support for the SHA-384 algorithm will not be implemented in the initial implementation but is planned for the CLASS release following the CS implementation.

### 1.4 Acronym Usage

The definitions of acronyms in this document are excluded from the body and are provided in the acronym list located in Appendix E.

### 1.5 Document Organization

This CS ICD is organized as follows:

- Section 1.0, Introduction, outlines the purpose and scope of the Common Submission ICD, as well as the organization and relationship of this document to other pertinent documents.
- Section 2.0, Interface Overview, explains the high-level interfaces and delivery methods.
- Section 3.0, Submissions to CLASS, explains the submissions the data providers make to CLASS.
- Section 4.0, Notifications to data providers, describes notifications that CLASS sends to the data providers.
- Appendix A, Submission Manifest XML Schema, presents the XML schema for the submission manifest.
- Appendix B, Example Submission Manifest, presents an example submission manifest.
- Appendix C, Ingest Report Schema, presents the XML schema for the Ingest Report.
- Appendix D, Example Ingest Report, presents an example Ingest Report
- Appendix E, Acronyms, lists acronyms used in this document.

## 1.6 References and Related Documents

Table 1 contains a list of reference and related documents.

**Table 1 – Related and Reference Documents**

Number	Document Number	Document Title
1	CLASS-1252-CLS-REQ-L2SR	CLASS Level 2 Requirements
2	CLASS-1257-CLS-REQ-L3SR	CLASS Level 3 Requirements
3	N/A	CLASS Common Submission Campaign Specific Requirements
4	CLASS-1474-CLS-DOC-CS	CLASS Common Submission Concept of Operations
5	CLASS-1087-CLS-DSN	CLASS Software Description Document
6	CLASS-1295-CLS-DSN	CLASS Hardware Description Document
7	CLASS-1333-CLS-ICD-NCDC	CLASS-NCDC ICD
8	CLASS-1405-CLS-ICD-NGDC	CLASS-NGDC ICD
9	CLASS-1468-CLS-ICD-NODC	CLASS-NODC ICD (Draft)
10	CLASS-1454-CLS-ICD-M2M	CLASS M2M ICD

Current versions of the ICDs between CLASS and each data center (shown in Table 1) provide more detailed information pertaining to network connectivity, implementation details, and the Ingest Report.

The baseline documentation for the CLASS Project is available online in the CLASS Document Repository at <https://kt.nsof.class.noaa.gov>.

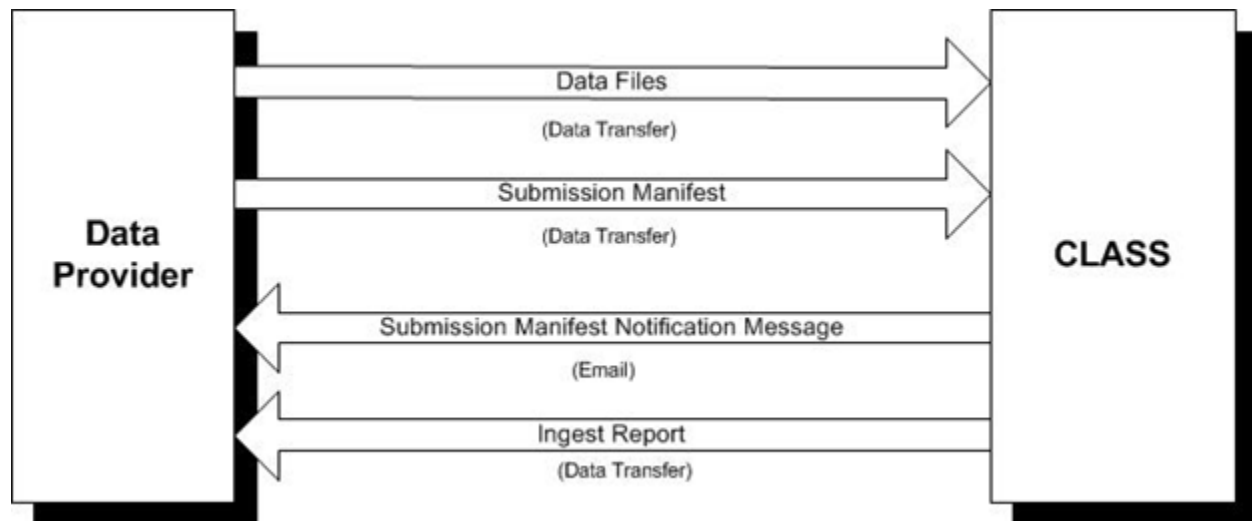
## 1.7 Document Maintenance

The CS ICD has been reviewed and is under baseline control. Any changes to this document will be submitted to the CLASS CCB for authorization and approval.

## Section 2.0 Interface Overview

This section provides a high-level description of the actions required for the transfer, validation, and ingest reporting of information to be preserved in the Archive.

All information submissions from a data provider to CLASS utilize a submission manifest to initiate the CLASS ingest process. The submission manifest contains a listing of data provider data files that are to be preserved within the archive. Submission manifests and data files are transferred from a data provider to CLASS. Figure 1 depicts these data flows and directions.



**Figure 1 – Data Transfers**

### 2.1 Data Transfer Overview

For each submission, a data provider first transfers data files followed by an associated submission manifest to a designated CLASS landing zone.

When CLASS discovers a submission manifest in the landing zone during periodic checks of that directory, CLASS processes the discovered submission manifests. If a submission manifest is considered valid, CLASS then processes each data file listed within it. If a submission manifest is considered invalid, then CLASS sends an email notification to the data provider.

CLASS generates and stores events indicating the ingest and archive state for each data file listed in validated submission manifests. These events appear in Ingest Reports that CLASS writes to a known subdirectory within the CLASS landing zone. Data provider software can retrieve these files via FTP, and analyze them to identify any files that were not successfully transmitted or ingested by CLASS.

**Table 2 – Data Provider - CLASS Transfer Descriptions**

<b>Activity Title</b>	<b>Description</b>	<b>Source of Primary Data Flow</b>
Data Files	The data files/products to be archived by CLASS.	data provider
Submission Manifest	Identifies files to be transferred to CLASS for the defined period (submission).	data provider
Submission Manifest Notification Message	Email notifications sent to data provider identifying submission manifest errors.	CLASS
Ingest Report	A file generated by CLASS that lists the ingest/archive state of data files associated with a particular submission manifest.	CLASS

## 2.2 Validation Overview

CLASS CS will validate the submission manifest including checking for proper formatting of the XML, detecting the presence of selected required fields, and confirming the number of data files referenced in the manifest. If this validation fails, CLASS will send an email to the data provider indicating the reason(s) for failure.

CLASS CS will also validate each data file listed in the manifest. CLASS will compare the file size and checksum against the values provided in the submission manifest. Additional data file element validation is also performed; refer to Section 3.4.1 for more information. If a data file is considered valid, then CLASS ingests it, otherwise, CLASS will flag it as invalid. In either case, information about each file is reported in subsequent Ingest Reports.

## Section 3.0 Submissions to CLASS

This section describes the input interface to the CLASS CS function. There are two primary inputs to this function: submission manifests and data files. The submission manifest contents and format are the focus of this section.

Section 3.3.2 describes the elements (Descriptive Information) that can be submitted. Appendix A contains the XML schema for the submission manifest. Appendix B provides an example of a Submission Manifest.

For CS, ingest transfers begin with the transfer of data files from the data provider to CLASS, followed by a submission manifest. The submission manifest contains the necessary information for CLASS to ingest the data files that have been transferred to CLASS as a part of the submission.

The file size and checksum listed in the submission manifest for each data file is used to validate the integrity of the corresponding data file in the CLASS landing zone.

Additionally, descriptive information about each data file is extracted from the submission manifest and written to the CLASS Data Management system during ingest, ensuring that the descriptive information remains with the data file. Both components, the data file and the descriptive information, are comprised of the Archive Information Package in the CLASS archival storage.

### 3.1 Collection Registration

Collection registration allows a designated information steward to identify a collection of files and then register an ID for that collection with the CLASS System. The information about that collection will be stored in CLASS and validated during the CS process. For restriction levels and duplicate file handling, values contained in the Collection Registry can be inherited by the files submitted for that collection. Therefore, this registration must be completed before any files within that collection are submitted to CLASS and stored.

The initial registration process will follow these steps:

1. Information steward submits a set of collection information, including a unique Collection ID, to a designated CLASS POC via email as a request for the registration of a collection.
2. CLASS validates and stores the collection information in a Collection Registry.
3. CLASS notifies the requestor via email of successful or unsuccessful collection registration.

Once the collection is registered with its unique provider-supplied Collection ID, the data center can then submit files to CLASS by including Collection ID as file metadata. Only files containing a registered Collection ID will be accepted by the CS process. Files that do not contain a registered Collection ID will be placed on hold during the ingest process. Should that occur, a CLASS Operator will contact a data center representative to resolve the issue with the non-registered Collection ID.

The information steward will have the capability to update and maintain the collection information as needed. However, the persistent Collection ID cannot be changed once data has been archived in CLASS for that Collection ID.

### 3.1.1 Collection Information

The following information will be submitted by the data center and stored in CLASS for each collection.

**Table 3 – Collection Information Elements**

Element	Required?	Definition
Collection ID	Y	Persistent and unique key for collection.
Provider's Collection Descriptive Title	N	Collection description.
Provider Name	Y	Organization that is responsible for providing the data.
Information Steward	N	Organization that is knowledgeable about the data.
Provider POC Email	Y	Email address used for resolving operational issues.
Restriction Level	Y	Field indicating what access level is used for this collection (see Section 3.1.3.1).
Duplicate Processing	Y	Instruction for how to handle duplicates: replace, hold, reject (see Section 3.1.3.2).
CLASS Data Configuration	Y	CLASS data configuration to which this collection belongs - must be one of the following: CS_NCDC, CS_NGDC, CS_NODC.
DOI	N	Digital Object Identifier.

### 3.1.2 Collection Processing

When collection information is submitted, the CLASS operator will perform the following validation:

- Verify that all required fields are present.
- Ensure that the Collection ID is unique within CLASS.
- Verify that the CLASS Data Type exists in CLASS.
- Ensure that the Restriction Level is valid.
- Ensure that the Duplicate Processing field is valid.
- Verify that the Provider POC Email is formatted as an email address.

In the initial implementation of collection registration, a data center representative will send an email to a CLASS POC with the collection information. A CLASS operator will then enter that information into the Collection Registry. If any of the validation above fails, the CLASS operator will notify the information steward of the errors via email.

### 3.1.3 File Submission Processing Based on Collections

When files are submitted to CLASS to be preserved for the long-term, CLASS will validate that the collection ID submitted with the file, in the submission manifest, already exists in the Collection Registry. Based on that Collection ID, there are two instances where information from the collection will be used for individual files as described in Sections 3.1.3.1 and 3.1.3.2.

### 3.1.3.1 Restriction Level

The CS process will use the following hierarchy or inheritance sequence for determining the restriction level of each file.

1. If the restriction level is provided in the file metadata, CLASS will assign that level to the file.
2. If the restriction level is not provided in the file metadata, CLASS will look in the collection record for the file and use the restriction level from that record.

### 3.1.3.2 Duplicate File Handling

The CS process will use the following method for determining the processing of duplicate files. If CS detects a duplicate file based on the file name, CLASS will look in the collection record for that file and retrieve the Duplicate Processing value from that record.

The three options for the handling of duplicate files are:

- **Reject** – the file is rejected.
- **Hold** – the file is placed on hold and a CLASS operator is notified.
- **Replace** – the new file replaces the existing file.

## 3.2 Ingest Transfers and Notifications

CLASS ingest of data files is performed as a periodic sequence of submissions triggered by the submission of a submission manifest. Each submission is handled as follows:

1. The data provider pushes a submission's data files into the appropriate CLASS landing zone, transferring the submission manifest file last.
2. CLASS detects the presence of submission manifests by looking for appropriate filenames and validates discovered submission manifests against the XML schema defined in Appendix A.
3. CLASS processes all data files listed within each validated submission manifest.
4. CLASS validates data files by verifying that they meet required constraints. See Section 3.4.1 and 4.1 for information on what CLASS validates.
5. Data files passing this validation are ingested.

## 3.3 Submission Manifest

The submission manifest will be used to identify the files that are being submitted, to specify the Descriptive Information for each file, and as a trigger for CLASS to begin its processing.

If data files are sent without a submission manifest, these files will expire and be removed from the landing zone. This is configurable per landing zone.

Refer to Appendix A. for the XML schema for a submission manifest. If the submission manifest fails validation against this schema defined in Appendix A, the entire submission manifest will be rejected and an email will be sent to the data provider as described in Section 4.1.

### 3.3.1 Submission Manifest File Naming Convention

**NOTE:** In specifying the file naming conventions described herein, normal fonts are used for literals and italicized fonts for variables.

The name of each submission manifest file is of the following form:

CS\_CLASS\_MANIFEST\_HostName\_Dyyyyddd\_pppppppp\_nnnnnnnnn

Where the elements are defined by:

**Table 4 – Submission Manifest File Naming Convention Elements**

<b>CS_CLASS_MANIFEST</b>	Identifies this as a submission manifest file for a CS transaction and remains static.
<i>HostName</i>	Is the name of the server that pushes the submission manifest to CLASS.
<i>Dyyyyddd</i>	Represents the date in UTC when the submission manifest file was created.
<i>pppppppp</i>	Represents the ProcessID of the process that generated the submission manifest file in the format pppppppp.
<i>nnnnnnnnn</i>	Represents the Process Time of the process that generated the submission manifest file in nanoseconds in the format nnnnnnnnn.

An example Common Submission manifest file name follows:

CS\_CLASS\_MANIFEST\_borg5\_D2013085\_00031933\_264417000

### 3.3.2 Required and Optional Elements in the Submission Manifest

Tables 5 and 6 list the elements that can be included in the submission manifest. For each element, the table identifies whether it is required in the manifest, the element type (e.g., string, integer, date/time), and its maximum length.

**Table 5 – Required Submission Manifest Elements Included Once**

Element	Required	Type	Max Size	Definition
begin_time	Y	Date/time		The date/time the Submission Manifest was created.
end_time	Y	Date/time		The date/time the Submission Manifest was created. Each end_time tag value must be unique among all submission manifests from each data provider.
number_of_files	Y	Integer	9,999	Number of files included in the submission manifest.

**Table 6 – Required and Optional Submission Manifest Elements for each <ingestfile>**

Element	Required	Type	Max Size	Definition
Collection ID	Y	String	20	A unique identifier for the collection with which the file is



Element	Required	Type	Max Size	Definition
				associated. Each file must have a Collection ID previously registered in CLASS.
Provider	Y	String	25	The role played by those persons or organizations that provide the information assets for long term preservation in an archive. (As applied to common ingest/archive, the Provider is the responsible data center.)
Submitted File Name	Y	String	255	The name of the file that is in the CLASS landing zone, ready for ingest. This value should be unique in CLASS.
File size	Y	Int8	2 <sup>63</sup> - 1	File size (number in bytes) of the file that is in the CLASS loading zone, ready for ingest.
Checksum	Y	String	512	The checksum (hash) value.
Checksum algorithm name and version	Y	String	15	Name and version of the checksum algorithm.
Restriction Level	N	Int	0 – 9	Restriction levels range from 0 - publicly available to 9 - not available to anyone.
Producer	N	String	25	The persons or organization that produces the information assets to be preserved.
Steward	N	String	25	The persons or organization that provides support to users by having a keen understanding of the data, including overall data quality and intended uses of the data.
Data Provider File Name	N	String	255	The data provider File Name is the original name of the file, prior to pre-ingest renaming.
File Format	N	String	10	The format of the file referenced in the submission manifest that is being sent to CLASS for archival storage.
File Compression Method	N	String	10	Method of compression used during pre-processing.
Original Archive date from Provider	N	Date/ time		When this file was first archived by the provider.

Element	Required	Type	Max Size	Definition
Creation Date/Time of the File	N	Date/time		When this file was created by the originator/producer.
Edition	N	String	15	Stored by CLASS but implies no special processing.
Version	N	String	15	Stored by CLASS but implies no special processing.
Browse image reference information	N	String	255	This could be a UUID referring to another object in the CLASS archive, or some other reference.
Platform name	N	String	60	Buoy, Station, Site, Satellite – for some collections with thousands of granules, they could come from different platforms (NOAA 15, etc.).
<b>User Defined Fields</b>				
Text1	N	String	255	Defined by the data provider.
Text2	N	String	255	Defined by the data provider.
Date1	N	Date/time		Defined by the data provider.
Date2	N	Date/time		Defined by the data provider.
Integer1	N	Int	2 <sup>31</sup> – 1	Defined by the data provider.
Integer2	N	Int	2 <sup>31</sup> – 1	Defined by the data provider.
Memo1	N	Memo		Defined by the data provider.
<b>Sub-element Temporal</b>				
Beginning Date/time of data coverage	N	Date/time		Beginning date and time applicable to the contents of the file.
Ending Date/time of data coverage	N	Date/time		Ending date and time applicable to the contents of the file.
Beginning Paleo Year	N	String	30	Beginning date and time applicable to the contents of the file – for paleontology data files.
Ending Paleo Year	N	String	30	Ending date and time applicable to the contents of the file – for paleontology data files.
<b>Sub-element Spatial</b>				
Point Latitude	N	Dec		The latitude of the point.
Point Longitude	N	Dec		The longitude of the point.

Element	Required	Type	Max Size	Definition
Bounding box - Latitude North	N	Dec		The north latitude of the area of coverage.
Bounding box - Latitude South	N	Dec		The south latitude of the area of coverage.
Bounding box - Longitude East	N	Dec		The east longitude of the area of coverage.
Bounding box - Longitude West	N	Dec		The west longitude of the area of coverage.

If any of the fields exceed the maximum length, CLASS will truncate the value, ingest the file, and no notification will be sent.

### 3.3.2.1 Restriction Levels

The restriction level field is optional in the manifest. If it is provided, CLASS Ingest will use that value for the ingested data file. If it is not provided, CLASS will look up the restriction level value that was set for the collection for that data file. For the reason that the restriction level is required for the collection, CLASS will use that value for the ingested data file.

### 3.3.2.2 Duplicate File Processing

The CS process will use the following method for determining the processing of duplicate files. If CS detects a duplicate file based on the file name, CLASS will look in the collection record for that file and retrieve the Duplicate Processing value from that record.

The three options for the handling of duplicate files are:

- **Reject** – the file is rejected.
- **Hold** – the file is placed on hold and a CLASS operator is notified.
- **Replace** – the new file replaces the existing file.

### 3.3.3 Submission Manifest Element Definitions

Table 7 provides definitions and, if applicable, the domain of values for the elements.

**Table 7 – Element Definitions**

XML Element	Definition	Domain (* Means and Value)
collection_ID	Collection ID: A unique identifier for the collection with which the file is associated. Each file must have a Collection ID. The Collection ID is generated by CLASS through the Collection Registration process and sent to the data provider for inclusion in this field.	Assigned by the data centers.
provider	Provider: The persons or organizations that provide the information assets for long term preservation in an archive. (As	NCDC, NGDC, NODC, *

XML Element	Definition	Domain (* Means and Value)
	applied to CS, the Provider is the responsible data center.)	
file_name	Submitted file name: File name of the file that is in the CLASS loading zone, ready for ingest.	The file name may be made up of up to 255 characters. It must be unique for all information sent to CLASS from a single data provider.
file_size	File size: Number in bytes of the size of the file.	>0
<i>checksum</i> : value	Checksum: A checksum or hash sum is a fixed-size datum computed from an arbitrary block of digital data for the purpose of detecting accidental errors that may have been introduced during its transmission or storage.	
<i>checksum</i> : algorithm	Checksum algorithm and version: Name and version of the checksum algorithm.	MD5, SHA-384
restriction_level	Restriction Level: Restriction Levels range from 0 - publicly available to 9 - not available to anyone. The numbers 1,2,3,4,5,6, 7 & 8 have been used to associate specific user ids with visibility to specific data types.	0-9
producer	Producer: The persons or organization that produces the information assets to be preserved. Generally, there should only be one producer of a collection.	*
steward	Information Steward: The persons or organization that provides support to users by having a keen understanding of the data, including overall data quality and intended uses of the data.	NCDC, NGDC, NODC
provider_file_name	Data Provider File Name: All files sent to CLASS from a single data provider must have unique file names. To meet this requirement, data providers may need to re-name legacy files. The data provider File Name is the original name of the file, prior to pre-ingest renaming.	*
file_format	File Format: The format of the file referenced in the Submission Manifest that is being sent to CLASS for archival storage.	e.g., tar, gzip, netCDF
file_compression	File Compression Method: Method of compression used during pre-processing.	*

XML Element	Definition	Domain (* Means and Value)
provider_archive_date	Original Archive Date from Provider: When this file was first archived by the provider.	e.g. 2009-01-06
file_creation_date	Creation Date/Time: When this file was created by the originator/producer.	e.g. 2013-01-16T23:12:57Z
file_edition	Edition: The CLASS System will not have any understanding of the values stored for edition. To the CLASS System, this value is a string value and the understanding of differences between string values stored for each file is not implied.	*
file_version	Version: The CLASS System will not have any understanding of the values stored for version. To the CLASS System, this value is a string value and the understanding of differences between string values stored for each file is not implied.	*
browse_image	Browse Image Reference Information: This could be a UUID referring to another object in the CLASS archive, or some other reference.	*
platform_name	Platform Name: Buoy, Station, Site, Satellite – for some collections with thousands of granules, they could come from different platforms (NOAA 15, etc.).	*
text_1	Text1: completely up to the data provider.	*
text_2	Text2: completely up to the data provider.	*
date_1	Date1: completely up to the data provider.	*
date_2	Date2: completely up to the data provider.	*
integer_1	Integer1: completely up to the data provider.	*
integer_2	Integer2: completely up to the data provider.	*
memo_1	Memo1: completely up to the data provider.	* Any XML or HTML content should be properly escaped to avoid validation errors.
begin_date_time	Beginning date/time of data coverage: Beginning date and time applicable to the contents of the file, using ISO 8601 standard.	Valid date/time value starting with 1 AD (TBR).
end_date_time	Ending Date/time of data coverage: Ending date and time applicable to the contents of the file, using ISO 8601	Valid date/time value starting with 1 AD (TBR).

XML Element	Definition	Domain (* Means and Value)
	standard.	
begin_paleo	Beginning Paleo Year: Beginning date and time applicable to the contents of the file solving for paleo date issue in UNIX, using ISO 8601 standard.	Date (year only - TBR) for data prior to 1 AD.
end_paleo	Ending Paleo Year: Ending date and time applicable to the contents of the file solving for the paleo date issue in UNIX, using ISO 8601 standard.	Date (year only - TBR) for data prior to 1 AD.
latitude_point	Point – Latitude: The latitude of the point.	
longitude_point	Point – Longitude: The longitude of the point.	
<i>Bounding_box:north</i>	Bounding box – Latitude North Point: Contains the latitude north point representing the box (area of coverage).	
<i>Bounding_box:south</i>	Bounding box – Latitude South Point: Contains the latitude south point representing the box (area of coverage).	
<i>Bounding_box:east</i>	Bounding box – Longitude East Point: Contains the longitude east point representing the box (area of coverage).	
<i>Bounding_box:west</i>	Bounding box – Longitude West Point: Contains the longitude west point representing the box (area of coverage).	

**Note:** In Table 7 the spatial sub-elements point and bounding box are mutually exclusive. If a bounding box is defined using the four bounding box elements then a point should not also be defined using the two point elements and vice versa.

### 3.4 Data Files

Data files must be pushed to the CLASS landing zone before the submission manifest they are referenced in is pushed to the CLASS landing zone.

The status of ingest and archive for each data file is included in an Ingest Report, which is described in more detail in Section 4.2 of this document.

#### 3.4.1 Data File Validation

The integrity of each data file is validated using the file size and checksum specified in the submission manifest. Data files that do not pass these integrity checks will not be ingested by CLASS and the failure will be reported in an Ingest Report.

The following lists identify the error conditions that will be recorded in the Ingest Report. In each case, the data file will not be ingested by CLASS. The error conditions are grouped by Acquisition Failure and Ingest Failure as defined in Section 4.2.3.

**Acquisition Failure:**

- Data file is not present in the landing zone.
- Checksum algorithm is not valid.
- Provided checksum does not match calculated checksum.
- Provided file size does not match calculated file size.

**Ingest Failure:**

- Any field is outside of the specified bounds/domain.
- A duplicate file was received but the Collection was configured to reject duplicate files.

**3.4.2 Data File Duplicates**

Data files with the same name as a previously ingested data file are considered duplicates. The options available for handling duplicate data files are shown in Table 8.

**Table 8 – Options Available for Handling Duplicate Data Files**

Replace	File will be replaced in archival storage.
Reject	Ingest Report will indicate file rejected.
Place on hold	File will not continue processing without the intervention of a CLASS operator.

Refer to the CLASS Common Submission Concept of Operations for information on how the setting of this processing can be defined for each data collection.

## Section 4.0 Notifications to Data Providers

### 4.1 Submission Manifest Notifications

The contents of submission manifests that are determined to be invalid by CLASS will not be processed for ingest. Each submission manifest will be validated against the XML schema defined in Appendix A.

The following errors will cause CLASS to reject a submission manifest and will result in an email sent to the data provider:

- The submission manifest fails validation against the schema defined in Appendix A.
- The number of files specified in the <number\_of\_files> element does not match the number of <ingestfile> elements present in the submission manifest.

### 4.2 Data File Notifications

CLASS provides confirmation of the ingest and archive of each file submitted in the CLASS Ingest Report. This report will include a CLASS-generated UUID for each CS data file. The UUID will be unique and persistent across CLASS. Ingest Reports are generated for other data ingested by CLASS and are not exclusive to data ingested by the CS process. The initial release of CS will include UUIDs for only CS data. Other data types will include a <file\_uuid> element in the Ingest Report, but the value will be empty. Similarly, a <collection\_id> element will be present in Ingest Reports for all data, however only CS data will contain a non-empty value.

Ingest Reports are used to inform the data provider of the status of files submitted to CLASS and referenced in a valid submission manifest. They are intended to ultimately assure the data provider that files have successfully completed the Ingest and Archive processes or inform the data provider that an error has occurred during the Ingest process. Ingest Reports are created and placed in a 'status' directory in the CLASS landing zone at a configurable frequency. Each Ingest Report includes data files that have been newly identified in a submission manifest or have changed state since the last Ingest Report.

Each report contains information on files that have reached a definite state of Success or Failure during the reporting period along with any new files that are In Process since the previous Ingest Report. Section 3.4.1 defines the validation performed on files that could result in Failure states reported in an Ingest Report. It is possible for files to reach a state of Success or Failure without ever being reported as In Process if the period of the report is of sufficient duration for a file to reach a definite state before the next Ingest Report is generated. This is the nature of the reporting cycle and is not to be viewed as an error. Files that are placed on hold due to a non-registered Collection ID will remain in the In Process state until manual intervention is taken by a CLASS Operator, or that Collection ID is registered with CLASS. Once the Collection ID is registered with CLASS, files that were placed on hold due to the non-registered Collection ID will proceed through the ingest process.

#### 4.2.1 Delivery of Ingest Report



CLASS will deliver Ingest Reports to a 'status' subdirectory within the CLASS landing zone, where the data provider can access and pull them. The naming convention of the Ingest Report will ensure a unique name for each report generated. CLASS will retain Ingest Report files for an agreed upon, configurable amount of time (e.g., four weeks), in case of the need for the data provider to retrieve them again.

#### 4.2.2 Ingest Report File Naming Convention

The file name for the Ingest Report is in the following form:

CLASS\_INGEST\_REPORT\_Dyyyymmdd.Thhmmss

Dyyyymmdd.Thhmmss represents the date and time in UTC at which the file was created.

#### 4.2.3 Ingest Report States Defined

The following Ingest states are defined for Ingest Reports:

- **In Process** – The file has been identified by CLASS in the submission manifest and has not reached a state of Success or Failure. [Known internally to CLASS as “In-Process of Ingest”]
- **Successful Ingest** – The file has completed the Ingest process and is ready to be written to tape. [Known internally to CLASS as “Successful Ingest”]
- **Acquisition Failure** – The file was not received by CLASS, the received file’s checksum or size are different from those listed in the manifest, the file’s checksum or size cannot be verified using the manifest, or the checksum algorithm specified in the manifest is not supported by CLASS. [Known internally to CLASS as “Failure to acquire a SIP (data file)"]
- **Ingest Failure** – Acquisition of the file succeeded, an attempt was made to ingest the file, but the attempt did not complete successfully (this can be caused by invalid DI such as restriction level, or by internal CLASS errors) [Known internally to CLASS as “Ingest Failure”]

The following archive state is defined for Ingest Reports:

- **Archive Status** – The date/time the AIP/file was successfully transferred to the online archive file system at each CLASS node.

#### 4.2.4 Information Conveyed in an Ingest Report

Ingest Reports convey the following summary information in each report:

1. Time of report generation.
2. Start coverage time of the report.
3. End coverage time of the report.
4. Number of files in the report.

Ingest Reports convey the following information about each data file. This information comes from the submission manifest containing this data file and is used to allow the Provider to uniquely identify the data file.

1. File Name provided in the submission manifest.

2. Provider supplied File Size.
3. Provider supplied Checksum(s).
4. Collection ID if available
5. File name of the submission manifest containing the data file.
6. Date of the submission manifest containing the data file.

Ingest Reports convey the following information pertaining to each data file. This information conveys the status of each data file.

1. CLASS Datatype.
2. Date and Time the data file entered this state.
3. CLASS-generated UUID if available (see element definition in Table 9).
4. Indication of state the data file has reached.
  - a. If reported as In-Process (only list the state).
  - b. If reported as Successful Ingest.
    - i. CLASS-created unique and persistent identifier.
    - ii. Actual file size.
    - iii. Actual checksum and algorithm used.
  - c. If reported as Acquisition Failure.
    - i. Error message written to the logs.
  - d. If reported as Ingest Failure.
    - i. Error message written to the logs.
5. Indication of the archive state the data file has reached for each CLASS node:
  - a. The name of the CLASS node.
    - i. If the data file has not yet been archived at the CLASS node a value of 0 will be present.
    - ii. If the data file has been archived at the CLASS node the date/time of archival will be present.

Table 9 provides more detail about the CLASS supplied UUID that will appear for each CS data file in an Ingest Report.

**Table 9 – UUID Element Definition**

Element	Definition	Domain
file_uuid	Universally Unique Identifier. CLASS will generate UUIDs for each file ingested.	The UUID is a sequence of 32 hexadecimal digits with the following layout, referred to as variant 2: wwwwww-mmmm-1hhh-vsss-xxxxxxxxxxxx where: -wwwwww is low order field of timestamp -mmmm is middle field of timestamp -1hhh is version number (1) followed by high field of timestamp -vsss is variant (first 2 bits) multiplexed with high field of clock sequence (14 bits) -xxxxxxxxxxxx is MAC address Version 1 UUID in accordance with RFC

Element	Definition	Domain
		4122, see <a href="http://www.ietf.org/rfc/rfc4122.txt">http://www.ietf.org/rfc/rfc4122.txt</a>

#### 4.2.4.1 Common for Every Report

In general the Ingest Report is structured as follows:

```
<ingest_report>
<report_gen_time>yyyy-mm-dd hh:mm:ssZ</report_gen_time>
<start_coverage_time>yyyy-mm-dd hh:mm:ssZ</start_coverage_time>
<end_coverage_time>yyyy-mm-dd hh:mm:ssZ</end_coverage_time>
<num_files_reported>k</num_files_reported>
<sentfile>
{see Sections 4.2.4.2 and 4.2.4.3 for details}
</sentfile>

{Repeat <sentfile></sentfile> lines k-1 times}

</ingest_report>
```

The <sentfile> tag sub elements will vary depending on the status of each file in the report.

Each tag is described as follows:

- **report\_gen\_time** – The creation date and time of the Ingest Report.
- **ingest\_report** – The root element indication that this xml file is an Ingest Report.
- **start\_coverage\_time and end\_coverage\_time** – Defines the coverage time of the report.
- **num\_files\_reported** – Number of files included in the report.
- **sentfile** – Contains additional sub elements that identify and report on each SIP in the report.

#### 4.2.4.2 Data File Identification Information

The following information will be included for each file inside of the <sentfile> tag:

```
<provider_supplied_filename>filename</provider_supplied_filename>
<provider_supplied_file_size>fileSize</provider_supplied_file_size>
<provider_supplied_checksum>checksum</provider_supplied_checksum>
<file_uuid>file_uuid</file_uuid>
<collection_ID>collection_id</collection_ID>
<manifest>manifestFilename</manifest>
<manifest_date>manifestDate</manifest_date>
```

Each tag is described as follows:

- **provider\_supplied\_filename** – The name of the file as named in the submission manifest.
- **provider\_supplied\_file\_size** – The size of the file as reported in the submission manifest.
- **provider\_supplied\_checksum** – The checksum of the file as reported in the submission manifest.
- **file\_uuid** – Universally Unique Identifier. CLASS will generate a UUID for each file ingested.
- **manifest** – The name of the submission manifest that identified the file.
- **manifest\_date** – The date of the submission manifest that identified the file.

### 4.2.4.3 Data File Status

The following information will be included for each file inside of the <sentfile> tag:

```

<ingest_status>status</ingest_status>
<datatype>datatype</datatype>
<ingest_status_datetime>datetimeOfStatus</ingest_status_datetime>
<archive>
  <node>NCDC</node>
  <datetime>datetimeOfArchival_NCDC</datetime>
</archive>
<archive>
  <node>NGDC</node>
  <datetime>datetimeOfArchival_NGDC</datetime>
</archive>

```

Information specific to Successful Ingest, Failure to acquire file, Ingest failure, and Date/Time of Archival will be included as indicated in Sections 4.2.4.3.1 and 4.2.4.3.2. No additional information is required for In-Process of Ingest.

Each tag is described as follows:

- **Ingest Status** – Indicates the state the file has reached as defined in Section 4.2.3. Note that Failure to acquire file is reported as Acquisition Failure.
- **datatype** – The datatype CLASS associated the file with.
- **ingest\_status\_datetime** – The date and time the file reached the specified status.
- **archive** – Information pertaining to the archive status for a CLASS node.
- **node** – The CLASS node at which the data file is archived.
  - **NCDC** – The CLASS node/location that the file is archived on the CLASS Shared File System
  - **NGDC** – The CLASS node/location that the file is archived on the CLASS Shared File System
- **datetime** – The Date/Time of archival on the CLASS Shared File System at the CLASS node specified by the node element.

#### 4.2.4.3.1 Successful Ingest and Archival Case

For the Successful Ingest and Archive Status case the following information is added inside the <sent\_file> tag:

```

<filename>filename</filename>
<filesize>filesize</filesize>
<file_uuid>8743428c-ef91-41d4-1405-44665544000</file_uuid>
<collection_ID>NXL2DP</collection_ID>
<checksum>checksum</checksum>
<checksum_algorithm>checksumAlgorithm</checksum_algorithm>
<archive>
  <node>NCDC</node>
  <datetime>datetimeOfArchival</datetime>
</archive>
<archive>
  <node>NGDC</node>
  <datetime>datetimeOfArchival</datetime>

```

</archive>

Each tag is described as follows:

- **filename** – The name of the file that is being reported on.
- **filesize** – The size of the file as determined by CLASS.
- **file\_uuid** – Universally Unique Identifier. CLASS will generate UUIDs for each CS file ingested.
- **collection\_ID** – A unique identifier for the collection with which the file is associated.
- **checksum** – The checksum for the file as calculated by CLASS. It is a sequence of 32 hexadecimal digits (lower case) representing the 128-bit message digest of the data file computed by the MD5 Message-Digest Algorithm, or the SHA-384 algorithm.
- **checksum\_algorithm** – The algorithm used to calculate the checksum by CLASS.
- **archive** – Information pertaining to the archive status for a CLASS node.
- **node** – The CLASS node at which the data file is archived.
  - **NCDC** – The CLASS node/location that the file is archived on the CLASS Shared File System.
  - **NGDC** – The CLASS node/location that the file is archived on the CLASS Shared File System.
- **datetime** – The Date/Time of archival on the CLASS Shared File System at the CLASS node specified by the node element.

#### 4.2.4.3.2 Failure to Acquire File and Ingest Failure Cases

For the Failure to acquire file and Ingest failure cases, the following information is added inside the <sent\_file> tag:

<error\_message>Checksum did not match.</error\_message>

The value listed above is only an example.

The tag is described as follows:

- **error\_message** – The error message written to the CLASS logs.

## Appendix A. Submission Manifest XML Schema

In the schema that follows, default values are used for minOccurs and maxOccurs. If either minOccurs or maxOccurs is not specified, each is assumed to have a value of "1." Specifying an offset value from UTC time is not supported for the xs:dateTime type. Doing so will cause the data file to be placed on hold during the ingest process and the data file will not be ingested.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
  xmlns="http://www.class.noaa.gov/cs"
  targetNamespace="http://www.class.noaa.gov/cs"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <xs:element name="manifest" type="manifestType"/>

  <xs:complexType name="manifestType">
    <xs:sequence>
      <xs:element name="begin_time" type="xs:dateTime"/>
      <xs:element name="end_time" type="xs:dateTime"/>
      <xs:element name="number_of_files" type="xs:nonNegativeInteger"/>
      <xs:element name="ingestfiles" type="ingestfilesType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ingestfilesType">
    <xs:sequence>
      <xs:element name="ingestfile" type="ingestfileType" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ingestfileType">
    <xs:sequence>
      <xs:element name="collection_ID" type="xs:string"/>
      <xs:element name="file_name" type="file_nameType"/>
      <xs:element name="file_size" type="xs:nonNegativeInteger"/>
      <xs:element name="checksum" type="checksumType"/>
      <xs:element name="ingestfile_di" type="ingestfile_diType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:simpleType name="file_nameType">
    <xs:restriction base="xs:string">
      <xs:maxLength value="255"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="checksumType">
    <xs:sequence>
      <xs:element name="algorithm" type="xs:string"/>
      <xs:element name="value" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ingestfile_diType">
    <xs:sequence>
      <xs:element name="provider" type="xs:string"/>
      <xs:element name="restriction_level" type="xs:integer" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

```

<xs:element name="steward" type="xs:string" minOccurs="0"/>
<xs:element name="producer" type="xs:string" minOccurs="0"/>
<xs:element name="provider_file_name" type="xs:string" minOccurs="0"/>
<xs:element name="file_format" type="xs:string" minOccurs="0"/>
<xs:element name="file_compression" type="xs:string" minOccurs="0"/>
<xs:element name="provider_archive_date" type="xs:dateTime" minOccurs="0"/>
<xs:element name="file_creation_date" type="xs:dateTime" minOccurs="0"/>
<xs:element name="file_edition" type="xs:string" minOccurs="0"/>
<xs:element name="file_version" type="xs:string" minOccurs="0"/>
<xs:element name="browse_image" type="xs:string" minOccurs="0"/>
<xs:element name="platform_name" type="xs:string" minOccurs="0"/>
<xs:element name="user_defined" type="user_definedType" minOccurs="0"/>
<xs:element name="temporal" type="temporalType" minOccurs="0"/>
<xs:element name="spatial" type="spatialType" minOccurs="0"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="user_definedType">
  <xs:sequence>
    <xs:element name="text_1" type="xs:string" minOccurs="0"/>
    <xs:element name="text_2" type="xs:string" minOccurs="0"/>
    <xs:element name="date_1" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="date_2" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="integer_1" type="xs:integer" minOccurs="0"/>
    <xs:element name="integer_2" type="xs:integer" minOccurs="0"/>
    <xs:element name="memo_1" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="temporalType">
  <xs:choice>
    <xs:group ref="contemporaryDateTime"/>
    <xs:group ref="paleoDateTime"/>
  </xs:choice>
</xs:complexType>

<xs:group name="contemporaryDateTime">
  <xs:sequence>
    <xs:element name="begin_date_time" type="xs:dateTime" minOccurs="0"/>
    <xs:element name="end_date_time" type="xs:dateTime" minOccurs="0"/>
  </xs:sequence>
</xs:group>

<xs:group name="paleoDateTime">
  <xs:sequence>
    <xs:element name="begin_paleo" type="xs:string" minOccurs="0"/>
    <xs:element name="end_paleo" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:group>

<xs:complexType name="spatialType">
  <xs:choice>
    <xs:element name="lat_lon_point" type="lat_lon_pointType"/>
    <xs:element name="bounding_box" type="bounding_boxType"/>
  </xs:choice>
</xs:complexType>

<xs:complexType name="lat_lon_pointType">
  <xs:sequence>
    <xs:element name="latitude" type="xs:decimal"/>
    <xs:element name="longitude" type="xs:decimal"/>
  </xs:sequence>
</xs:complexType>

```

```
<xs:complexType name="bounding_boxType">
  <xs:sequence>
    <xs:element name="north" type="xs:decimal"/>
    <xs:element name="south" type="xs:decimal"/>
    <xs:element name="east" type="xs:decimal"/>
    <xs:element name="west" type="xs:decimal"/>
  </xs:sequence>
</xs:complexType>

</xs:schema>
```



## Appendix B. Example Submission Manifest

The following CS manifest is an example using NCDC's NEXRAD data.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://www.class.noaa.gov/cs"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.class.noaa.gov/cs_manifest.xsd">
  <begin_time>2013-01-16T23:06:21Z</begin_time>
  <end_time>2013-01-16T23:06:21Z</end_time>
  <number_of_files>2</number_of_files>
  <ingestfiles>
    <ingestfile>
      <collection_ID>NXL2DP</collection_ID>
      <file_name>NWS_NEXRAD_NXL2DP_KSOX_20130116200000_20130116205959.tar</file_name>
      <file_size>813056012345</file_size>
      <checksum>
        <algorithm>MD5</algorithm>
        <value>563613c54dc8aece56c3afedcf4aec1</value>
      </checksum>
      <ingestfile_di>
        <provider>NCDC</provider>
        <restriction_level>3</restriction_level>
        <steward>NCDC</steward>
        <producer>NWS</producer>
        <provider_file_name>nexrad_data_file_1</provider_file_name>
        <file_format>proprietary radar format</file_format>
        <file_compression/>
        <provider_archive_date>2013-01-16T23:06:21Z</provider_archive_date>
        <file_creation_date>2013-01-16T23:05:07Z</file_creation_date>
        <file_edition/>
        <file_version/>
        <browse_image/>
        <platform_name/>
        <user_defined>
          <text_1>information in the first text field</text_1>
          <text_2>information in the second text field</text_2>
          <date_1>2012-12-21T11:05:04Z</date_1>
          <date_2>2012-12-22T11:05:04Z</date_2>
          <integer_1>1</integer_1>
          <integer_2>2</integer_2>
          <memo_1/>
        </user_defined>
      </ingestfile_di>
    </ingestfile>
  </ingestfiles>
  <temporal>
    <!--begin_date_time>2013-01-16T19:00:00Z</begin_date_time>
    <end_date_time>2013-01-16T20:00:00Z</end_date_time-->
    <!--begin_paleo>really_old_date</begin_paleo-->
    <end_paleo>another old date</end_paleo>
  </temporal>
  <spatial>
    <!--lat_lon_point>
      <latitude>35.569656</latitude>
      <longitude>-82.551956</longitude>
    </lat_lon_point-->
    <bounding_box>
      <north>46.1</north>
      <south>-1232.1</south>
      <east>111.123</east>
      <west>13214314.</west>
    </bounding_box>
  </spatial>
</manifest>
```

```
</spatial>
</ingestfile_di>
</ingestfile>
<ingestfile>
  <collection_ID>NXL2DP</collection_ID>
  <file_name>NWS_NEXRAD_NXL2DP_KSOX_20130116200000_20110116205950.tar</file_name>
  <file_size>813052345</file_size>
  <checksum>
    <algorithm>JUNK</algorithm>
    <value>563613c54dc8aece56c3afedcf4aecl</value>
  </checksum>
  <ingestfile_di>
    <provider>NCDC</provider>
    <restriction_level>3</restriction_level>
    <steward>NCDC</steward>
    <producer>NWS</producer>
    <provider_file_name>nexrad_data_file_1</provider_file_name>
    <file_format>proprietary radar format</file_format>
    <file_compression/>
    <provider_archive_date>2013-01-16T23:06:21Z</provider_archive_date>
    <file_creation_date>2013-01-16T23:05:07Z</file_creation_date>
    <file_edition/>
    <file_version/>
    <browse_image/>
    <platform_name/>
    <user_defined>
      <text_2>information in the second text field</text_2>
      <date_2>2012-12-22T11:05:04Z</date_2>
      <integer_1>1</integer_1>
      <integer_2>2</integer_2>
      <memo_1/>
    </user_defined>
    <temporal>
      <begin_date_time>2013-01-16T19:00:00Z</begin_date_time>
      <end_date_time>2013-01-16T20:00:00Z</end_date_time>
      <!--begin_paleo>really_old_date</begin_paleo>
      <end_paleo>another old date</end_paleo-->
    </temporal>
    <spatial>
      <lat_lon_point>
        <latitude>35.569656</latitude>
        <longitude>-82.551956</longitude>
      </lat_lon_point>
      <!--bounding_box>
        <north>46.1</north>
        <south>-1232.1</south>
        <east>111.123</east>
        <west>13214314.</west>
      </bounding_box-->
    </spatial>
  </ingestfile_di>
</ingestfile>
</ingestfiles>
</manifest>
```

## Appendix C. Ingest Report Schema

```

<xs:schema      attributeFormDefault="unqualified"      elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="ingest_report">
  <xs:complexType>
    <xs:sequence>
      <xs:element type="xs:string" name="start_coverage_time"/>
      <xs:element type="xs:string" name="end_coverage_time"/>
      <xs:element type="xs:int" name="num_files_reported"/>
      <xs:element type="xs:string" name="report_gen_time"/>
      <xs:element name="sentfile" maxOccurs="unbounded" minOccurs="0">
        <xs:complexType>
          <xs:choice maxOccurs="unbounded" minOccurs="0">
            <xs:element type="xs:string" name="file_uuid"/>
            <xs:element type="xs:string" name="collection_ID "/>
            <xs:element type="xs:string" name="checksum"/>
            <xs:element type="xs:string" name="checksum_algorithm"/>
            <xs:element type="xs:string" name="datatype"/>
            <xs:element type="xs:string" name="ingest_status_datetime"/>
            <xs:element type="xs:string" name="filename"/>
            <xs:element type="xs:int" name="filesize"/>
            <xs:element type="xs:string" name="manifest"/>
            <xs:element type="xs:string" name="manifest_date"/>
            <xs:element type="xs:string" name="provider_supplied_checksum"/>
            <xs:element type="xs:int" name="provider_supplied_file_size"/>
            <xs:element type="xs:string" name="provider_supplied_filename"/>
            <xs:element type="xs:string" name="ingest_status"/>
            <xs:element type="xs:string" name="error_message"/>
            <xs:element name="archive">
              <xs:complexType>
                <xs:sequence>
                  <xs:element type="xs:string" name="node"/>
                  <xs:element type="xs:string" name="datetime"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:choice>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>

```

## Appendix D. Example Ingest Report

An example Ingest Report follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<ingest_report>
  <start_coverage_time>2013-01-16T06:00:00Z</start_coverage_time>
  <end_coverage_time>2013-01-17T06:00:00Z</end_coverage_time>
  <num_files_reported>6</num_files_reported>
  <report_gen_time>2013-01-17T06:00:21Z</report_gen_time>
  <sentfile>
    <file_uuid>8743428c-ef91-41d4-1405-44665544000</file_uuid>
    <collection_ID>NXL2DP</collection_ID>
    <checksum>563613c54dc8aece56c3afedcf4aec1</checksum>
    <checksum_algorithm>MD5</checksum_algorithm>
    <datatype>NXL2DP</datatype>
    <ingest_status_datetime>2013-01-16T23:12:57Z</ingest_status_datetime>
    <filename>NWS_NEXRAD_NXL2DP_KSOX_20130116200000_20130116205959.tar</filename>
    <filesize>8130560</filesize>
    <manifest>NCDC_CLASS_MANIFEST_borg5_wxradar_D2013016_00001767_536397000</manife
st>
    <manifest_date>2013-01-16T23:06:21</manifest_date>
    <provider_supplied_checksum>563613c54dc8aece56c3afedcf4aec1</provider_supplied
_checksum>
    <provider_supplied_file_size>8130560</provider_supplied_file_size>
    <provider_supplied_filename>NWS_NEXRAD_NXL2DP_KSOX_20130116200000_201301162059
9.tar</provider_supplied_filename>
    <ingest_status>Successful Ingest</ingest_status>
    <archive>
      <node>NCDC</node>
      <datetime>0</datetime>
    </archive>
    <archive>
      <node>NGDC</node>
      <datetime>0</datetime>
    </archive>
  </sentfile>
  <sentfile>
    <file_uuid>48863b30-ef91-41d4-1405-44665544000</file_uuid>
    <collection_ID>NXL2DP</collection_ID>
    <checksum>8073e45043ad6b684ee979df577f92d5</checksum>
    <checksum_algorithm>MD5</checksum_algorithm>
    <datatype>NXL2DP</datatype>
    <ingest_status_datetime>2013-01-16T09:13:22Z</ingest_status_datetime>
    <filename>NWS_NEXRAD_NXL2DP_KMKX_20130116060000_20130116065959.tar</filename>
    <filesize>3072000</filesize>
    <manifest>NCDC_CLASS_MANIFEST_borg5_wxradar_D2013016_00013438_824562000</manife
st>
    <manifest_date>2013-01-16T09:05:43Z</manifest_date>
    <provider_supplied_checksum>8073e45043ad6b684ee979df577f92d5</provider_supplied
_checksum>
    <provider_supplied_file_size>3072000</provider_supplied_file_size>
    <provider_supplied_filename>NWS_NEXRAD_NXL2DP_KMKX_20130116060000_201301160659
9.tar</provider_supplied_filename>
    <ingest_status>Successful Ingest</ingest_status>
    <archive>
      <node>NCDC</node>
      <datetime>2013-01-16T09:18:03Z</datetime>
    </archive>
  </sentfile>
</ingest_report>
```

```

        <node>NGDC</node>
        <datetime>0</datetime>
        </archive>
</sentfile>
<sentfile>
  <file_uuid>5c50c270-ef91-41d4-1405-44665544000</file_uuid>
  <collection_ID>NXL2DP</collection_ID>
  <checksum>eee77fd2a6ecb9ce7733028dbclaf85b</checksum>
  <checksum_algorithm>MD5</checksum_algorithm>
  <datatype>NXL2DP</datatype>
  <ingest_status_datetime>2013-01-16T09:12:25Z</ingest_status_datetime>
  <filename>NWS_NEXRAD_NXL2DP_KLRX_20130116060000_20130116065959.tar</filename>
  <filesize>2508800</filesize>
  <manifest>NCDC_CLASS_MANIFEST_borg5_wxradar_D2013016_00027637_598203000</manife
st>
  <manifest_date>2013-01-16T09:05:36Z</manifest_date>
  <provider_supplied_checksum>eee77fd2a6ecb9ce7733028dbclaf85b</provider_supplied
_checksum>
  <provider_supplied_file_size>2508800</provider_supplied_file_size>
  <provider_supplied_filename>NWS_NEXRAD_NXL2DP_KLRX_20130116060000_2013011606595
9.tar</provider_supplied_filename>
  <ingest_status>Successful Ingest</ingest_status>
  <archive>
  <node>NCDC</node>
  <datetime>2013-01-16T09:13:03Z</datetime>
  </archive>
  <archive>
  <node>NGDC</node>
  <datetime>2013-01-16T09:18:04Z</datetime>
  </archive>
</sentfile>
<sentfile>
  <collection_ID>NXL2DP</collection_ID>
  <datatype>NXL2DP</datatype>
  <ingest_status_datetime>2013-01-17T05:12:19Z</ingest_status_datetime>
  <manifest>NCDC_CLASS_MANIFEST_borg5_wxradar_D2013017_00014750_598062000</manife
st>
  <manifest_date>2013-01-17T05:05:47Z</manifest_date>
  <provider_supplied_checksum>d01fe3e39d1e03f10525fe58ff1cf51d</provider_supplied
_checksum>
  <provider_supplied_file_size>4843520</provider_supplied_file_size>
  <provider_supplied_filename>NWS_NEXRAD_NXL2DP_KGRK_20130117020000_2013011702595
9.tar</provider_supplied_filename>
  <ingest_status>In-Process of Ingest</ingest_status>
  <archive>
  <node>NCDC</node>
  <datetime>0</datetime>
  </archive>
  <archive>
  <node>NGDC</node>
  <datetime>0</datetime>
  </archive>
</sentfile>
<sentfile>
  <ingest_status_datetime>2013-01-17T03:13:20Z</ingest_status_datetime>
  <manifest>NCDC_CLASS_MANIFEST_borg5_wxradar_D2013017_00028842_285736000</manife
st>
  <manifest_date>2013-01-17T03:05:46Z</manifest_date>
  <provider_supplied_checksum>c816293e2b33d1b99c579ef9a33096a4</provider_supplied
_checksum>
  <provider_supplied_file_size>6758400</provider_supplied_file_size>
  <provider_supplied_filename>NWS_NEXRAD_NXL2DP_KBBX_20130117000000_2013011700595
9.tar</provider_supplied_filename>

```

```
<ingest_status>Acquisition Failure</ingest_status>
<error_message>SIP not found.</error_message>
<archive>
  <node>NCDC</node>
  <datetime>0</datetime>
</archive>
<archive>
  <node>NGDC</node>
  <datetime>0</datetime>
</archive>
</sentfile>
<sentfile>
  <ingest_status_datetime>2013-01-16T09:12:50Z</ingest_status_datetime>
  <manifest>NCDC_CLASS_MANIFEST_borg5_wxradar_D2013016_00002376_973265000</manife
st>
  <manifest_date>2013-01-16T09:05:30Z</manifest_date>
  <provider_supplied_checksum>0ee03e7b4d71616440a5989051a19331</provider_supplied
_checksum>
  <provider_supplied_file_size>6993920</provider_supplied_file_size>
  <provider_supplied_filename>NCEP_WxCharts_TIFF_2000063.tar</provider_supplied_f
ilename>
  <ingest_status>Ingest Failure</ingest_status>
  <datatype>NXL2DP</datatype>
  <error_message>checksum from manifest does not match calculated</error_message>
  <archive>
    <node>NCDC</node>
    <datetime>0</datetime>
  </archive>
  <archive>
    <node>NGDC</node>
    <datetime>0</datetime>
  </archive>
</sentfile>
</ingest_report>
```

## Appendix E. Acronyms

<b><u>Acronym</u></b>	<b><u>Definition</u></b>
AD	Active Directory
AIP	Archive Information Package
CCB	Configuration Control Board
CLASS	Comprehensive Large Array-data Stewardship System
CS	Common Submission
DGP	Diversified Global Partners Joint Venture Limited Liability Company
DI	Description Information
DOI	Digital Object Identifier
FTP	File Transfer Protocol
HTML	Hyper Text Markup Language
ICD	Interface Control Document
ID	Identifier
ISD	Information Systems Division
ISO	International Organization for Standardization
ISSO	Information System Security Officer
MAC	Media Access Control
MD	Message Digest
M2M	Machine-to-Machine
NCDC	National Climatic Data Center
NESDIS	National Environmental Satellite Data and Information Service
netCDF	Network Common Data Format
NEXRAD	NEXt generation weather RADAR Program
NGDC	National Geophysical Data Center
NOAA	National Oceanic and Atmospheric Administration
NODC	National Oceanographic Data Center
NSOF	NOAA Satellite Operations Facility
OAIS-RM	Open Archival Information System- Reference Model
POC	Point of Contact
QMO	Quality Management Organization
QMP	Quality Management Plan
RFC	Remote Function Call
SHA	Secure Hash Algorithm
SIP	Submission Information Package
TB	Terabyte
TBR	To Be Reviewed
TIM	Technical Interchange Meeting
UTC	Universal Time Coordinated
UUID	Universal Unique Identifier
XML	Extensible Markup Language