The Consultative Committee for Space Data Systems

## Draft Recommendation for Space Data System Standards

# OPEN ARCHIVAL INFORMATION SYSTEM INTEROPERABILITY FRAMEWORK (OAIS-IF) ARCHITECTURE DESCRIPTION

## PROPOSED DRAFT RECOMMENDED STANDARD

## CCSDS 000.0-W-0

# BLUE BOOK
## May 2021

# AUTHORITY

| | |
|---|---|
| Issue: | Blue Book, Issue 0 |
| Date: | May 2019 |
| Location: | Not Applicable |

**(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF AUTHORITY:)**

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4), and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the e-mail address below.

This document is published and maintained by:

CCSDS Secretariat
National Aeronautics and Space Administration
Washington, DC, USA
E-mail: secretariat@mailman.ccsds.org

# STATEMENT OF INTENT

**(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF INTENT:)**

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommendations** and are not in themselves considered binding on any Agency.

CCSDS Recommendations take two forms: **Recommended Standards** that are prescriptive and are the formal vehicles by which CCSDS Agencies create the standards that specify how elements of their space mission support infrastructure shall operate and interoperate with others; and **Recommended Standards** that are more descriptive in nature and are intended to provide general guidance about how to approach a particular problem associated with space mission support. This **Recommended Standard** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommended Standard** is entirely voluntary and does not imply a commitment by any Agency or organization to implement its recommendations in a prescriptive sense.

No later than five years from its date of issuance, this **Recommended Standard** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Standard** is issued, existing CCSDS-related member Standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such Standards or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new Standards and implementations towards the later version of the Recommended Standard.

# FOREWORD

This document is a draft technical Recommended Standard for use in developing and maintaining broader consensus on what is required for an archive to provide permanent, or indefinite long term, preservation of digital information.

This draft Recommended Standard establishes a framework of specifications that forms the basis for the Open Archival Information System (OAIS) Interoperability Framework (IF). OAIS is a long-established Process Framework (PF) to enable digital preservation in trustworthy archives. The OAIS-IF supplements OAIS with interoperable technical specifications that will allow interoperability between users and multiple archives, and between multiple archives. The OAIS-IF is not required for an archive to cite compliance with OAIS.

OAIS provides a basis for further standardization within an archival context. OAIS-IF is an example of that further standardization.

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Standard is therefore subject to CCSDS document management and change control procedures, which are defined in the *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4). Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be sent to the CCSDS Secretariat at the e-mail address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- China National Space Administration (CNSA)/People's Republic of China.
- Deutsches Zentrum für Luft- und Raumfahrt (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Federal Space Agency (FSA)/Russian Federation.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- Japan Aerospace Exploration Agency (JAXA)/Japan.
- National Aeronautics and Space Administration (NASA)/USA.
- UK Space Agency/United Kingdom.

Observer Agencies

- Austrian Space Agency (ASA)/Austria.
- Belgian Federal Science Policy Office (BFSPO)/Belgium.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
- Chinese Academy of Sciences (CAS)/China.
- Chinese Academy of Space Technology (CAST)/China.
- Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
- Danish National Space Center (DNSC)/Denmark.
- Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
- Electronics and Telecommunications Research Institute (ETRI)/Korea.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
- Hellenic National Space Committee (HNSC)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Institute of Space Research (IKI)/Russian Federation.
- KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- Korea Aerospace Research Institute (KARI)/Korea.
- Ministry of Communications (MOC)/Israel.
- National Institute of Information and Communications Technology (NICT)/Japan.
- National Oceanic and Atmospheric Administration (NOAA)/USA.
- National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
- National Space Organization (NSPO)/Chinese Taipei.
- Naval Center for Space Technology (NCST)/USA.
- Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
- South African National Space Agency (SANSA)/Republic of South Africa.
- Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
- Swedish Space Corporation (SSC)/Sweden.
- Swiss Space Office (SSO)/Switzerland.
- United States Geological Survey (USGS)/USA.

# PREFACE

This document is a draft CCSDS Recommended Standard. Its 'Blue Book' status indicates that the CCSDS believes the document to be technically mature and has released it for formal review by appropriate technical organizations. As such, its technical contents are not stable, and several iterations of it may occur in response to comments received during the review process.

Implementers are cautioned **not** to fabricate any final equipment in accordance with this document's technical content.

# DOCUMENT CONTROL

| Document | Title and Issue | Date | Status |
|---|---|---|---|
| CCSDS 000.0-W-0 | [Document Title], Proposed Draft Recommended Standard, Issue 0 | [Month Year] | Current proposed draft |

# TABLE OF CONTENTS

# TABLE OF FIGURES

# 1 INTRODUCTION

## 1.1 PURPOSE AND SCOPE

The purpose of this document is to define the CCSDS and International Organization for Standardization (ISO) **Open Archival Information System** (OAIS) Interoperability Framework (IF). An OAIS is an Archive, consisting of an organization, which may be part of a larger organization, of people and systems, that has accepted the responsibility to preserve information and make it available for a **Designated Community**. The OAIS-IF is a supplement to that overarching standard that adds capabilities for system interoperability between users and archives, and between coordinating archives. This document outlines a data system architectural approach and a set of specifications for interfaces required for interoperability and that are visible to Producers and Consumers. This standard is the Architecture Description document that sets the overall architectural framework for the OAIS-IF suite of standards.

The OAIS-IF is an implementable framework for digital repositories that enables international and collaborative research. Its aim is to provide a set of interoperable protocols and interface specifications that will enable the access and re-use of the data, both within and across the operational boundaries of trusted digital repositories. The OAIS-IF is designed to be effectively applied broadly across a spectrum of small, medium, and large use cases and involving a wide variety of stakeholders.

Implementers and system developers that plan to develop systems compliant with the OAIS-IF suite of standards should have a solid grasp of the precepts, concepts and terminology of the Reference Model for an OAIS as described in CCSDS 650.0-M-2.

The information being maintained in these Archives has been deemed to need **Long Term Preservation**, even if the OAIS itself is not permanent. **Long Term** is long enough to be concerned with the impacts of changing technologies, as well as support for new media and data formats, or with a changing Knowledge Base of the Designated Community or changes within the Designated Community or its definition. Long Term may extend indefinitely. Further treatment of the scope of Long Term preservation is in the RM for OAIS, CCSDS 650.0-M-2.-

In terms of scope, this Architecture Description Document is intended to specify normative requirements only for the OAIS-IF components of an OAIS. To describe the overall architecture it also describes components in the client (producer or consumer) systems and in the OAIS Archive "below" the OAIS-IF components. However, these are intended to illuminate the core assumptions behind the architecture design, and not specify any components in the client systems, nor archive components external to the OAIS-IF. The interfaces between the OAIS-IF and external functions are the key assets specified to achieve interoperability across those interfaces. They are fully specified and normative in this document. However, underlying functions below the interfaces within the client or archive systems may be developed differently than this description as long as they support the specified normative functions of the interoperable interfaces.

## 1.2 APPLICABILITY

Like the OAIS Reference Model in CCSDS 650.0-M-2, this document may be applicable to any Archive that complies with that OAIS standard as well as any archive that wishes to interoperate using the standard. It is specifically applicable to organizations with the responsibility of making information available for the Long Term. This includes organizations with other responsibilities, such as processing and distribution in response to programmatic needs.

This architecture is specifically designed to supplement OAIS Archives. However, this architecture or components of it may be used by archives that are partially or fully non-compliant to the Reference Model for OAIS. The authors of this standard cannot guarantee that these technical approaches will work to fulfill objectives of archives that are not fully OAIS compliant. It is hoped that in these cases partial implementation of the OAIS-IF will encourage greater adoption of the RM for OAIS as archives learn the value of the OAIS practices that enable truly trustworthy Archives for preserving valuable information.

It is intended that the functionality and components in OAIS-IF will exactly mirror the content of the RM for OAIS. However, since these are two separate documents with updates released at different times and different approval cycles, it may be that new functions can be added to OAIS-IF that are not yet in the RM for OAIS. Likewise, there may be new functions in OAIS that are not yet in the OAIS-IF. The intention is to keep the OAIS RM practice and the OAIS-IF specification as closely aligned as possible. However, perfect alignment may not be possible at every given point in time.

These specifications, including the functional and information modeling concepts, are relevant to the comparison and design of facilities which hold information, on a temporary basis, for three reasons:

- When taking into consideration the rapid pace of technology changes or possible changes in a Designated Community, there is the likelihood that facilities, thought to be holding information on a temporary basis, will in fact find that some or much of their information holdings will need Long Term Preservation attention. Stable OAIS-IF standards will help abate the disruption of technology changes.
- Although some facilities holding information may themselves be temporary, some or all of their information may need to be preserved indefinitely. Such facilities need to become active participants in the Long Term Preservation effort and adoption of OAIS-IF will facilitate that transition.
- Regardless of preservation objectives, this architecture enables interoperability for efficiency benefits, preservation benefits, and cross-discipline research benefits.

## 1.3 OAIS-IF STAKEHOLDERS

In a broad sense, OAIS-IF has applicability to the following stakeholders. This is not an exclusive list, but is intended to illustrate how the document should be of interest to key organization participants.

- **Any organization who has implemented or plans to implement an OAIS-compliant system.** Not all OAIS-compliant systems will have OAIS-IF capabilities. Indeed, as this first version of OAIS-IF is released, none of the OAIS systems in the world will be OAIS-IF compliant. But OAIS

implementers should evaluate the benefits to themselves and their customers from implementing an OAIS-IF compliant interoperable archive. Therefore, they have a stake in OAIS-IF.

- **Managers**, who we assume are key decision makers and determine technology adoption and use. We use the Manager stakeholder broadly for anyone who sets overall OAIS policy.
- **Application Software Developers**, who are those responsible for providing software at an application level (i.e. software implementing any of the six functional entities[1] of an OAIS). Application software is likely to be repository-specific.
- **Infrastructure Software Developers**, who are those responsible for providing the underlying software framework or environment which may be used by application software developers. This software is much less likely to be repository specific. The distinction between application and infrastructure is not necessarily exact but the separation from application software is useful in identifying the parts of OAIS-IF that form part of the underling infrastructure and are more likely to be reused from repository to repository.

## 1.4 RATIONALE

The rationale for OAIS and the Reference Model for OAIS is captured in CCSDS 650.0-M-2.

The rationale for the OAIS Interoperability Framework includes the rationale for OAIS (not repeated here) because it supports OAIS by augmenting it with capabilities for interoperability.

The rationale and vision for OAIS-IF is that in the long-range future it will provide:

- A common user interface experience for users (providers and consumers) of OAIS Archives when accessing many diverse kinds of archives through the OAIS-IF.

- An efficient standardized way for archives to exchange data between archives using the same standardized OAIS-IF interfaces.

- Given broad acceptance of OAIS-IF in the OAIS community, a better chance that long-term preservation will work because future generations can easily find the interfacing resources (plug-ins, etc.) that can be used to access legacy archives.

- Enhanced capabilities for cross-discipline research when many different disciplines use the same interface, and access to a new archive outside of their Designated Community can be accomplished via OAIS-IF.

## 1.5 CONFORMANCE

An Archive may conform to the Reference Model for OAIS without conforming to the OAIS-IF.

An OAIS Archive that also conforms to OAIS-IF must implement the normative sections of this document, namely sections 3 and 4.

While the OAIS Reference Model does not define or require any particular method of implementation, the OAIS-IF must necessarily bound some implementation options in order to insure interoperability. However, the goal of OAIS-IF is to only limit implementation options necessary for interoperability. This is intended to restrict implementation at the interface of systems, but those interfaces are usually characterized to support underlying functionality as required by the OAIS Reference Model. Therefore the definitions at the interfaces and protocols may necessarily imply some underlying functionality as part of the OAIS-IF suite of standards. As described in section 1.1, Purpose and Scope, the description of that functionality outside the OAIS-IF is not normative, and may be implemented in different ways, as long as it supports the specified normative functionality for that interface.

**A conformant OAIS-IF Archive may provide additional services that are beyond those required of the OAIS-IF standards**.

This document does not assume or endorse any specific computing platform, system environment, system design paradigm, system development methodology, database management system, database design paradigm, data definition language, technology, or media required for implementation.

The OAIS-IF is designed as an interoperability framework to support the development of interoperability between archives, both OAIS Archives and non-OAIS archives, using the OAIS-IF standard. As such, it attempts to address all the major activities of an ***interoperable*** information-preserving Archive in order to define a consistent and useful set of interoperability terms and concepts. A standard or other document that claims to be conformant to the OAIS-IF shall use the terms and concepts defined in the OAIS-IF in the same manner.

## 1.6 DOCUMENT STRUCTURE

### 1.6.1 ORGANIZATION BY SECTION

A general description of this document's sections are:

- Section 1 *Purpose and Scope* describes the problem space and rationale for OAIS-IF, and advice on what to expect from the document organization and conventions.
- Section 2 *Overview* provides an informative (non-normative) explanation of the relationships between OAIS-IF components and between them and the environment..
- Section 3 *Interoperability Framework* is a normative description of the requirements on the components of an OAIS-IF architecture. It presents the technical concepts that OAIS-IF uses in order to perform the functions of an OAIS in an interoperable way.
- (Add explanation of annexes once they are solidified)

This Blue Book begins with a description of the context for the creation of OAIS-IF in the form of the motivation and rationale for the framework. Further sections in this Blue Book then offer greater levels of detail about OAIS-IF generated directly from a formal model of the OAIS-IF. This detailed information is presented using the object-oriented paradigm. Each class, attribute, and relationship is formally defined using text and UML diagrams. It is anticipated that these sections will be primarily applicable to system developers but will be of interest to other stakeholders.

It is expected that after this document is approved and published by CCSDS, the model will be made available online by CCSDS. This should be a valuable aid to system developers of OAIS-IF systems.

### 1.6.2 TYPOGRAPHICAL CONVENTIONS

There are many terms which are used in this framework and which need to have well-defined meanings. These terms are defined in subsection 1.6.2. When first used in the text, they are shown in bold and are capitalized. Subsequent use employs capitalization only. Because of their extensive use in this document, the defined terms 'data' and 'information' will not always be capitalized unless they are part of another defined term. The defined term 'archive' will not be capitalized unless it is used as the equivalent of an 'OAIS Archive'.

Many diagrams are included throughout this reference model, primarily in Sections 4 and 6. In text discussing the diagrams, block names are capitalized and flows are italicized.

## 1.7 DEFINITIONS

### 1.7.1 ACRONYMS AND ABBREVIATIONS

**AIC** Archival Information Collection

**AIP** Archival Information Package

**AIU** Archival Information Unit

**API** Application Programming Interface

**ASCII** American Standard Code for Information Interchange

**CCSDS** Consultative Committee for Space Data Systems

**CD-ROM** Compact Disk - Read Only Memory

**CDO** Content Data Object

**CRC** Cyclic Redundancy Check

**CSV** Comma Separated Value

**DBMS** Data Base Management System

**DIP** Dissemination Information Package

**DRM** Digital Rights Management

| | |
|---|---|
| **FITS** | Flexible Image Transport System |
| **FTP** | File Transfer Protocol |
| **HFMS** | Hierarchical File Management System |
| **IEEE** | Institute of Electrical and Electronics Engineers |
| **IETF** | Internet Engineering Task Force |
| **ISBN** | International Standard Book Number |
| **ISO** | International Organization for Standardization |
| **MPEG** | Moving Picture Experts Group |
| **OAIS** | Open Archival Information System |
| **PDF** | Portable Document Format |
| **PDI** | Preservation Description Information |
| **QA** | Quality Assurance |
| RFC | Request For Comment |
| **SIP** | Submission Information Package |
| **UML** | Unified Modeling Language |
| **VHS** | Video Home System |
| **WWW** | World Wide Web |
| **XFDU** | XML Formatted Data unit |
| **XML** | eXtensible Markup Language |

### 1.7.2 TERMINOLOGY

There are many terms which are used in this standard and which need to have well-defined meanings. These terms are defined in this subsection. When first used in the text, they are shown in bold and are capitalized. Subsequent use employs capitalization only.

This standard is applicable to all disciplines and organizations that do, or expect to, preserve and provide information in digital form, these terms cannot match all of those familiar to any particular discipline (e.g., traditional archives, digital libraries, science data centers). Rather, the approach taken is to use terms that are not already overloaded with meaning so as to reduce conveying unintended meanings. Therefore, it is expected that all disciplines and organizations will find that they need to map some of their more familiar terms to those of the OAIS Reference Model and OAIS-IF. This should not be difficult and is viewed as a contribution, rather than a deterrent, to the success of these

standards. For example, archival science focuses on preservation of the 'record'. This term is not used in these standards, but one mapping might approximately equate it with 'Content Data Object within an Archival Information Package'.

TERMS TO BE SUPPLIED (Probably after current OAIS Red Book is published)

## 1.8 REFERENCES

The following publications contain provisions which, through reference in this text, constitute provisions of this document. At the time of publication, the editions indicated were valid. All publications are subject to revision, and users of this document are encouraged to investigate the possibility of applying the most recent editions of the publications indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS publications.

[Only references required as part of the specification are listed in the References subsection. See CCSDS A20.0-Y-4, *CCSDS Publications Manual* (Yellow Book, Issue 4, April 2014) for additional information on this subsection.]

Reference Model for an Open Archival Information System (OAIS). Magenta Book. CCSDS 650.0-M-2 Issue 2. June 2012. (to be changed to Issue 3 when issue 3 is released)

Audit and Certification of Trustworthy Digital Repositories. Magenta Book. Recommended Practice CCSDS 652.0-M-1. September 2011. (to be changed when issue 3 is released)

OTHER REFERENCES TO BE SUPPLIED

# 2 OVERVIEW

The following concepts set the context for normative definitions starting in section 3.

## 2.1 OAIS INTEROPERABILITY FRAMEWORK (OAIS-IF)

An OAIS Archive is an organization that intends to preserve information for access and use by a Designated Community.

An Open Archival Information System (OAIS) is an Archive, an organization that intends to preserve information for access and use by a Designated Community.. It meets a set of responsibilities that allows an OAIS Archive to be distinguished from other uses of the term 'Archive'.

The OAIS Interoperability Framework (OAIS-IF) is a framework based on the concepts presented in the OAIS Reference Model (RM) and augmented with features designed during several decades of digital archive development. The OAIS-IF is designed to be implementable and is an interoperable framework that fosters the acquisition, stewardship, and continuing access to data products, related information resources, and services for a Designated Community.

The environment surrounding an OAIS includes Management, Consumers, and Producers. The resulting environment of the OAIS-IF is illustrated in figure 1.

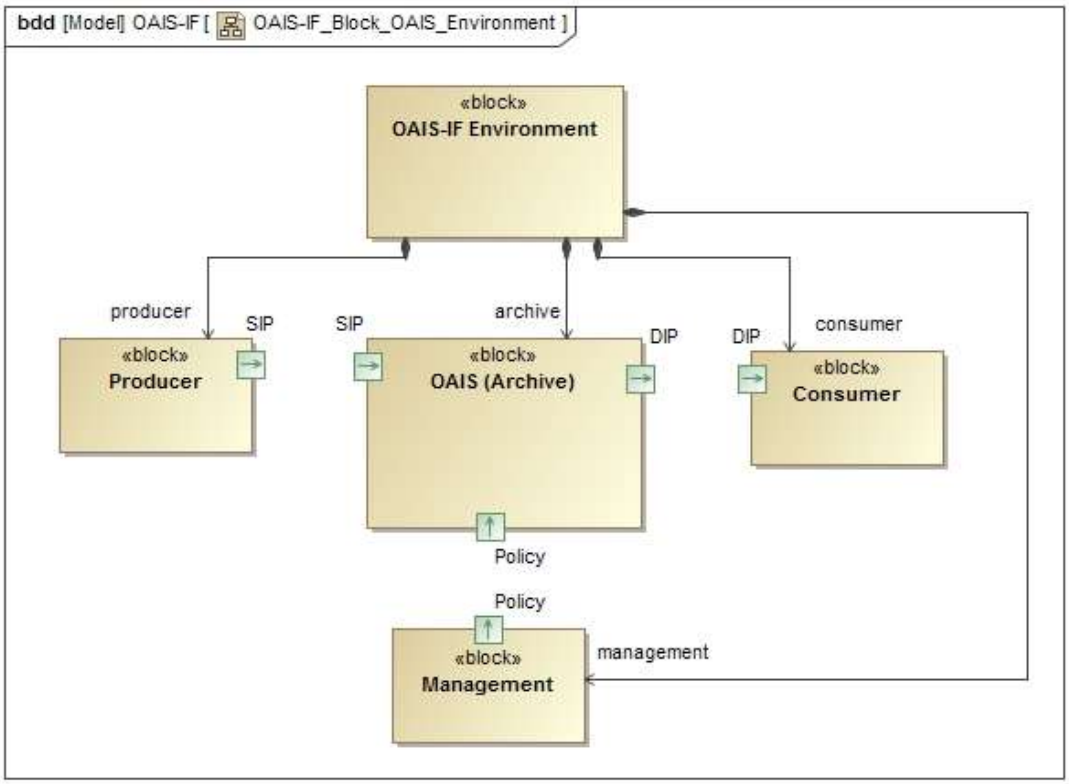


**Figure 1 - OAIS Environment**

Management is the role played by those who set overall OAIS policy as one component in a broader policy domain, for example as part of a larger organization.

Producer is the role played by those persons or client systems that provide the information to be preserved. This can include other OAISes or internal OAIS persons or systems. A Producer creates a Submission Information Package(s) (SIPs) and submits it to the Archive where it is processed into one or more Archival Information Packages (AIPs).

A Consumer is the role played by those persons, or client systems, who interact with OAIS services to find preserved information of interest and to access that information. A Consumer receives a Dissemination Information Package(s) (DIP) from the Archive.

## 2.2 OAIS FUNCTIONAL ENTITIES

Within an OAIS (Archive), an OAIS Functional Entity is an entity responsible for an operational function in a specific part of an Open Archive Information System (OAIS). The OAIS functional entities include Access, Administration, Archival Storage, Data Management, Ingest, and Preservation Planning. The OAIS Interoperability Framework, being based on the OAIS Reference Model, has two additional functional entities the Archive Abstraction Layer and the Analytics Platform.

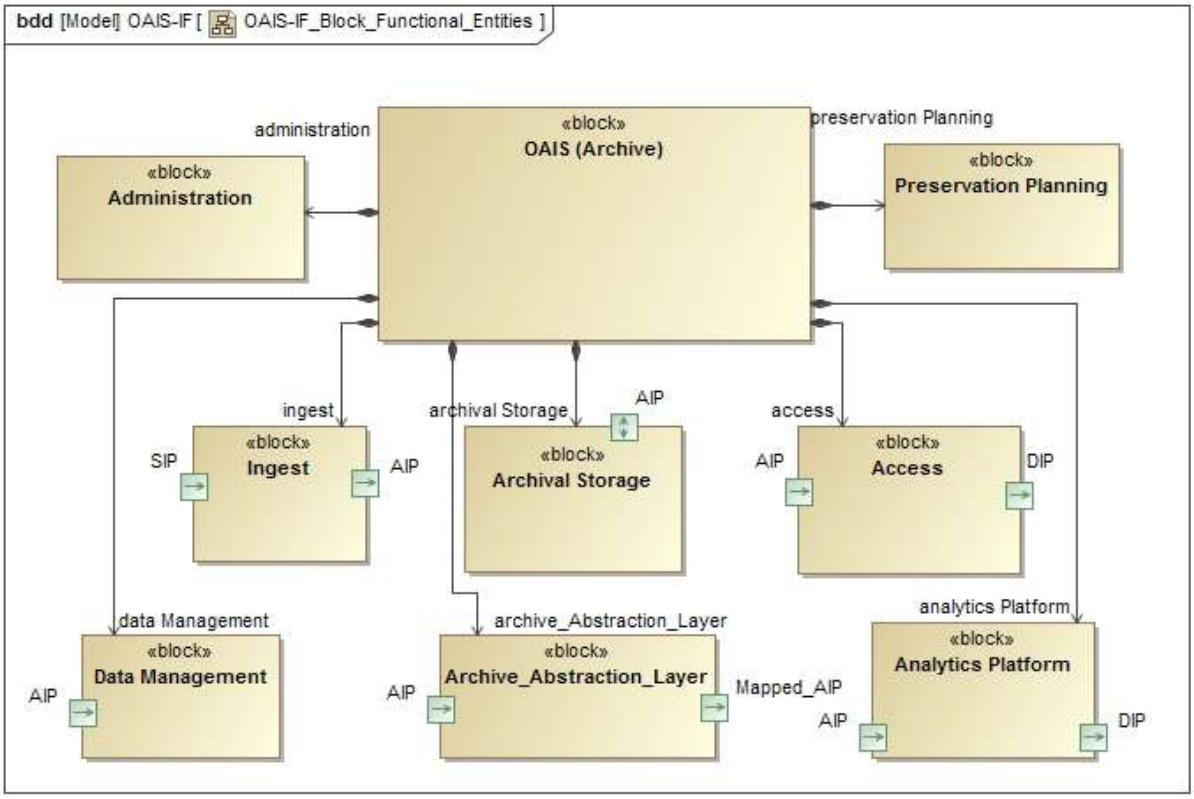


**Figure 2 - OAIS Functional Entities**

The Access Functional Entity (aka Access) contains the services and functions which make the archival information holdings and related services visible to Consumers. Access generates and provides a DIP to a Consumer, produces a Query Response for a Consumer, and provides Report Assistance to a Consumer.

The Administration Functional Entity (aka Administration) contains the services and functions needed to control the operation of the other OAIS functional entities on a day-to-day basis. For Consumers and Producers Administration sends Information Requests, Bills and Special Request Responses to Consumers. Final Ingest Report and possible liens are sent to a Producer.

The Archival Storage Functional Entity (aka Archival Storage) contains the services and functions used for the storage and retrieval of Archival Information Packages.

The Data Management Functional Entity (aka Data Management) contains the services and functions for populating, maintaining, and accessing a wide variety of information. Some examples of this information are catalogs and inventories on what may be retrieved from Archival Storage, processing algorithms that may be run on retrieved data, Consumer access statistics, Consumer billing, Event Based Orders, security controls, and OAIS schedules, policies, and procedures.

The Ingest Functional Entity (aka Ingest) contains the services and functions that accept Submission Information Packages from Producers, prepares Archival Information Packages for storage, and ensures that Archival Information Packages and their supporting Descriptive Information become established within the OAIS. Ingest sends Receipt Confirmation to a Producer.

The Preservation Planning Functional Entity (aka Preservation Planning) provides the services and functions for monitoring the environment of the OAIS and provides recommendations and preservation plans to ensure that the information stored in the OAIS remains accessible to, and understandable by, and sufficiently usable by, the Designated Community over the Long Term, even if the original computing environment becomes obsolete. Preservation Planning surveys a Consumer and surveys a Producer.

The Archive Abstraction Layer Functional Entity provides a mapping and possible translation between an object class in the OAIS Information Model and an object class in a non-conforming information model. package. For example a Consumer asking for Provenance Information as defined in in the OAIS Information Model could receive information about a derived product, the source products, and processing software that was grouped and classified as processing history in a non-OAIS information package. This is of course if the Archive Abstraction Layer had definitions of the two information models, how their components were related, and how to translate from one to the other if needed.
The Analytical Platform is a unified data analysis solution designed to address the demands of users beyond the data management infrastructure necessary for maintaining a long-term trusted digital repository. In general it provides contextual analyzed data from across the repository and joins different tools for creating analytics systems.

## 2.3 OAIS INTEROPERABILITY FRAMEWORK DEFINITION

The OAIS Interoperability Framework (OAIS-IF) involves the Producer and Consumer as they perform their roles and interact with the Ingest and Access functional entities of an OAIS. Three viewpoints of the framework are presented, an abstract component architecture, an information model, and an abstract functional interface. The abstract component architecture consists of a hierarchy of three major components: Client, OAIS Interoperability Framework, and OAIS IF Archive. The OAIS Interoperability Framework in turn consists of the Consumer and Producer Interfaces, the OAIS Information Model, the Abstraction Layer, and the Adapter layer.
The OAIS Information Model provides the framework's domain of discourse. The entities defined in the domain of discourse are the objects passed between functional entities of the framework. For example, the OAIS Digital Object is passed between a Consumer and an archive.

The abstract functional interface is defined within the Abstraction Layer and consists of formally interfaces and their operations. These interfaces must be implemented by the functional entities. For example, an Adapter for an archive must implement the Adapter Interface, Message Interface, and the Identifier Interface. Due to inheritance, the Adapter also implements the Information Object Interface.

The Negotiate Interface is used by a Client and the archive to identify one or more implemented Adapters to be used by the Consumer Interface and Producer Interface and the archive to interoperate. Once an Adapter has been set, the Consumer and Producer interoperate via Messages

using any implemented protocol. Any entity defined in the Information Model may be passed between any two connected functional entities that implement the Information Object Interface.

.

# 3 INTEROPERABILITY FRAMEWORK

In this section the environment in which the OAIS Interoperability Framework (OAIS-IF) exists is diagramed to provide a context for the normative OAIS-IF Interface and OAIS Information Model subsections of this document. [CN]. The OAIS-IF and its associated components, Client and OAIS_IF_Archive, are decomposed into their constituent sub-components. The entities comprising the OAIS Information Model and the Access and Ingest functional entities come directly from the OAIS Reference Model. A new component, the Negotiate, has been added to represent the need to obtain agreement on common communication methods and common extensions to the OAIS Information Model, between clients and archives.

A component in represents a modular part of a system that encapsulates the state and behavior of a set of elements such as attributes or methods. Its behavior is defined in terms of provided and required interfaces, is self-contained, and substitutable.

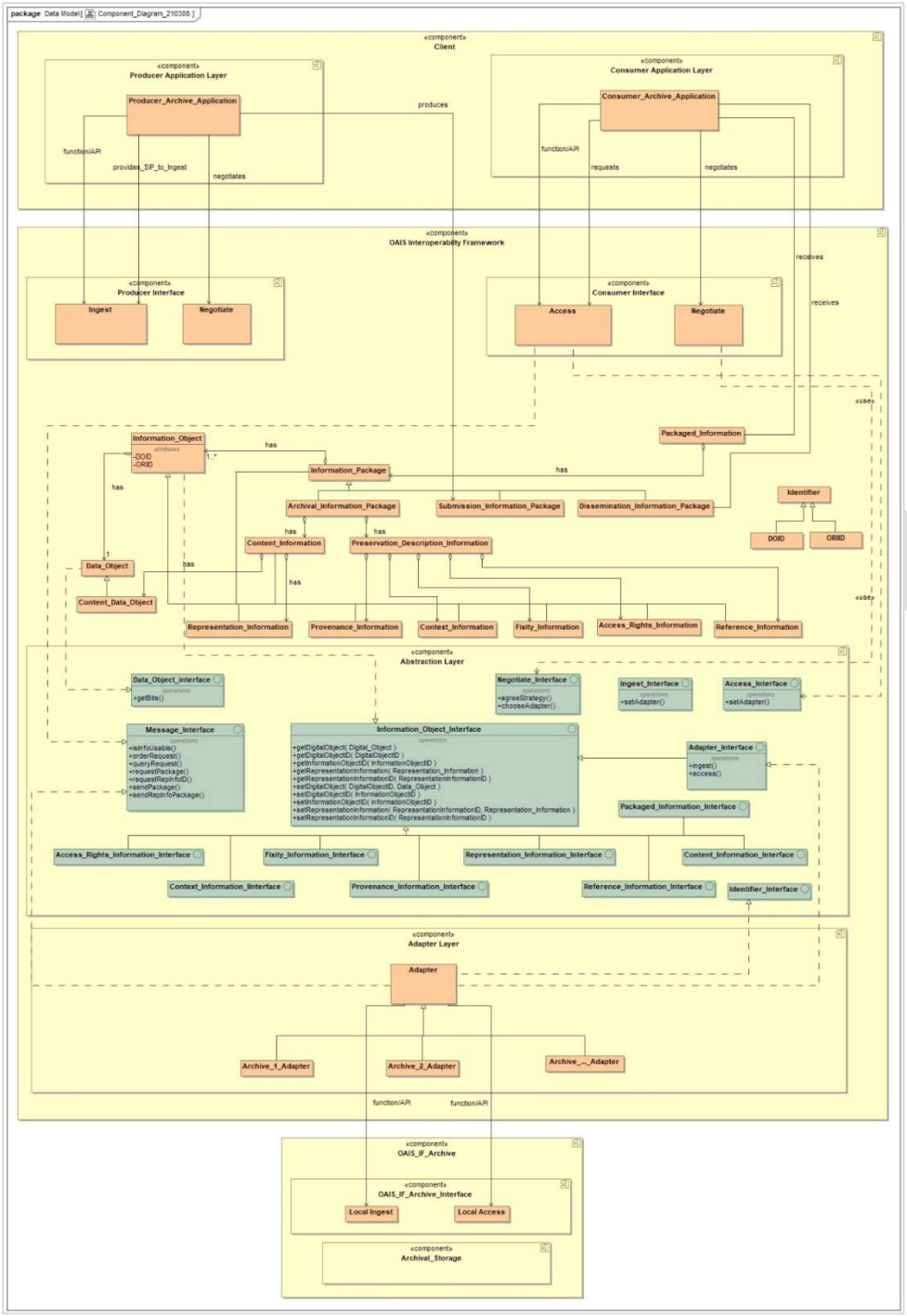


**Figure 3 - Component Diagram**

## 3.1 CLIENT

A Client is a computer system or process that requests a service of another computer system or process (a server) using some kind of protocol and accepts the server's responses.

The Client object class represents a computer system or process that requests a service of another computer system or process (a server) using a specific protocol and accepts the server's responses. The Client class is a Component. This section is informative.

### 3.1.1 CONSUMER_APPLICATION_LAYER

The Consumer Application Layer contains a program or group of programs designed for a Consumer. The Consumer Application Layer component is a Component and is an element of Client. This section is informative.

#### 3.1.1.1 Consumer_Archive_Application

A Consumer Archive Application is an application for use by a user performing the role of a consumer for a data archive.

The Consumer Archive Application object class represents an application for use by a user performing the role of a consumer in a data archive. The Consumer Archive Application negotiates for Adapters via the Negotiate service and makes requests for Information Packages or Dissemination Information Packages through the Access service. As an application it may invoke OAIS Interoperability Framework services either through functions calls by consuming the Framework or through an Application Programming Interface (API) provided for the Framework. The Consumer Archive Application is an element of the Consumer Application Layer.

### 3.1.2 PRODUCER_APPLICATION_LAYER

The Producer Application Layer contains a program or group of programs designed for Producers. The Producer Application Layer component is a subclass of Component and is an element of Client. This section is informative.

#### 3.1.2.1 Producer_Archive_Application

A Producer_Archive_Application an application for use by a user performing the role of a producer for a data archive.

The Producer Archive Application object class represents an application for use by a user performing the role of a producer in a data archive. The Producer Archive Application negotiates for Adapters via the Negotiate service and ingests Submission Information Packages through the Ingest service. As an application it may invoke OAIS Interoperability Framework services either through functions calls by consuming the Framework or through an Application Programming Interface (API) provided for the Framework. The Producer Archive Application is an element of the Producer Application Layer.

## 3.2 OAIS_INTEROPERABILITY_FRAMEWORK

The OAIS Interoperability Framework is an abstraction based on the OAIS Functional and Information Models in which software providing generic functionality can be selectively changed by additional user-written code to provide application-specific software that interoperates across digital repositories.

The OAIS Interoperability Framework is an abstraction based on the OAIS Functional and Information Models in which software providing generic functionality can be selectively changed by additional user-written code to provide application-specific software that interoperates across digital repositories. This section is normative.

### 3.2.1 PRODUCER_INTERFACE

The Producer Interface is an abstraction of producer services.

The Producer Interface object class is a well-defined entry point for producer services. The Producer Interface class is a subclass of Component and is an element of the OAIS Interoperability Framework. This section is informative.

#### 3.2.1.1 Ingest

Ingest Functional Entity (aka Ingest): The OAIS functional entity that contains the services and functions that accept Submission Information Packages from Producers, prepares Archival Information Packages for storage, and ensures that Archival Information Packages and their supporting Descriptive Information become established within the OAIS.

The Ingest object class provides the functions necessary to accept Submission Information Packages and register them through the Archive Interface. It implements the Message Service Interface that provides a protocol or interaction pattern for communication. It also implements the Access Interface that uses an Adapter to interoperate with the archive. The Adapter to be used is negotiated with the archive. The Ingest class is an element of the Consumer and Producer Interface components. This section is normative.

#### 3.2.1.2 Negotiate

The Negotiate function allows a user and the OAIS to negotiate adapters that enable them to interoperate.

The Negotiate object class provides the functions necessary to identify and choose specialized Adapters that are mutually acceptable to interoperate between the client and the archive. The Negotiate class implements the Negotiate Interface which provides agreeStrategy and chooseAdapter methods. It also implements the Message Service Interface that provides a protocol or interaction pattern for communication. The Negotiate service is an element of the Consumer Interface component. This section is normative.

### 3.2.2 CONSUMER_INTERFACE

The Consumer Interface provides abstractions of consumer services.

The Consumer Interface object class is a well-defined entry point for consumer services. The Consumer Interface class is a subclass of Component and is an element of the OAIS Interoperability Framework. This section is informative.

#### 3.2.2.1 Access

Access Functional Entity (aka Access): The OAIS functional entity that contains the services and functions which make the archival information holdings and related services visible to Consumers.

The Access object class provides the functions necessary to locate and retrieve Information Packages and Dissemination Information Packages through the Archive Interface. It implements the Message Service Interface that provides a protocol or interaction pattern for communication. It also implements the Access Interface that uses an Adapter to interoperate with the archive. The Adapter to be used is negotiated with the archive. The Ingest class is an element of the Consumer and Producer Interface components. This section is normative.

#### 3.2.2.2 Negotiate

See 3.2.1.2.

### 3.2.3 INFORMATION MODEL

The Information Model describe the types of information that are exchanged and managed within an OAIS. This subsection also defines the specific Information Objects that are used within the OAIS to preserve and access the information entrusted to the Archive. This detailed model of OAIS-related Information Objects is intended to aid the architect or designer of future OAIS systems. The objects discussed in this subsection are conceptual and should not be taken to imply any specific implementations. [C2] However the Interfaces described within this document defined normative methods for getting and putting each Information Object.

An information model is a representation of concepts and the relationships, constraints, rules, and operations to specify data semantics for a chosen domain of discourse. This section is normative.

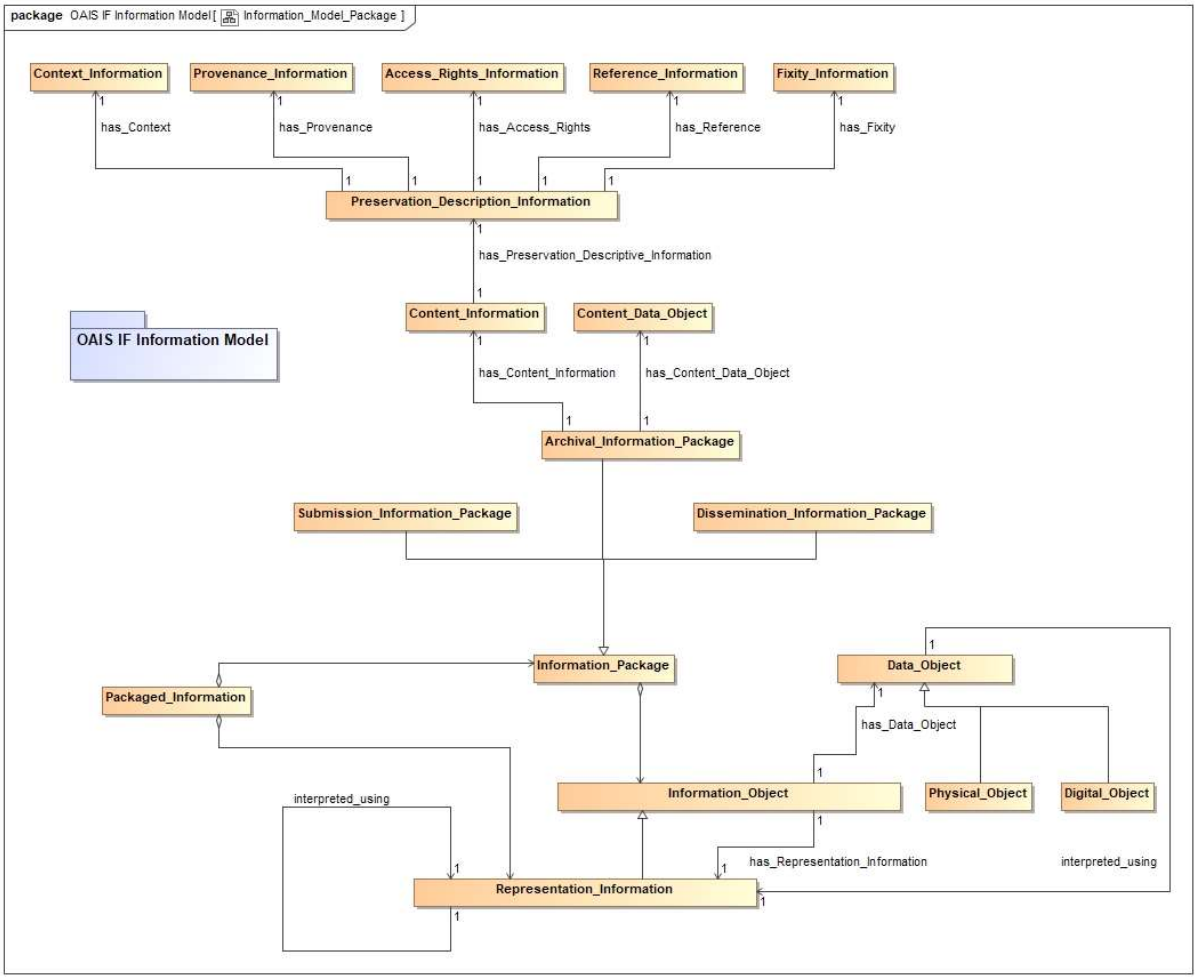


**Figure 4 - OAIS Information Model**

### 3.2.3.1 Access_Rights_Information

Access Rights Information: The information that identifies the access restrictions pertaining to the Content Information, including the legal framework, licensing terms, and access control. It contains the access and distribution conditions stated within the Submission Agreement, related to both preservation (by the OAIS) and final usage (by the Consumer). It also includes the specifications for the application of rights enforcement measures. [C1]

The Access Rights Information object class implements the Access Rights Information Interface. The Access Rights Information class is also a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.2 Archival_Information_Package

Archival Information Package (AIP): An Information Package, consisting of the Content Information and the associated Preservation Description Information (PDI), which is preserved within an OAIS. [C1]

The Archival Information Package object class is a subclass of the Information Package class and is composed of a Content Information class and a Preservation Description Information (PDI) class. It is also indirectly a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.3 Content_Data_Object

Content Data Object: The Data Object, that together with associated Representation Information, comprises the Content Information. [C1]

The Content Data Object is the Data object of Content Information. It has associated Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.4 Content_Information

Content Information: A set of information that is the original target of preservation or that includes part or all of that information. It is an Information Object composed of its Content Data Object and its Representation Information. [C1]

The Content Information object class is composed of a Content Data Object and a Representation Information class. It implements the Information Object Interface, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. Content Information is further described by Preservation Descriptive Information. This section is normative.

#### 3.2.3.5 Context_Information

Context Information: The information that documents the relationships of the Content Information to its environment. This includes why the Content Information was created and how it relates to other Content Information objects. [CC]

The Context Information object class implements the Context Information Interface. The Context Information class is also a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.6 Dissemination_Information_Package

Dissemination Information Package (DIP): An Information Package, derived from one or more AIPs, and sent by Archives to the Consumer in response to a request to the OAIS. [C1]

The Dissemination Information Package object class is a subclass of the Information Package class and is composed of a Content Information class and a Preservation Description Information (PDI) class. It is also indirectly a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.7 Fixity_Information

Fixity Information: The information which documents the mechanisms that ensure that the Content Information object has not been altered in an undocumented manner. An example is a Cyclical Redundancy Check (CRC) code for a file. [C1]

The Fixity Information object class implements the Fixity Information Interface. The Fixity Information class is also a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.8 Information_Object

Information Object: A Data Object together with its Representation Information. [C1]

The Information Object class is composed of a Data Object and a Representation Information class. It implements the Information Object Interface is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.9 Information_Package

Information Package: A logical container composed of optional Content Information and optional associated Preservation Description Information. Associated with this Information Package is Packaging Information used to delimit and identify the Content Information and Package Description information used to facilitate searches for the Content Information. [C1]

The Information Package object class is composed of a Content Information class and a Preservation Description Information (PDI) class. It is a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.10 Preservation_Description_Information

Preservation Description Information (PDI): The information which is necessary for adequate preservation of the Content Information and which can be categorized as Provenance, Reference, Fixity, Context, and Access Rights Information. [C1]

The Preservation Description Information object class is composed of Access Rights Information, Context Information, Fixity Information, Provenance Information, and Reference Information. It provides preservation description for Content Information. It is also a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.11 Provenance_Information

Provenance Information: The information that documents the history of the Content Information. This information tells the origin or source of the Content Information, any changes that may have taken place since it was originated, and who has had custody of it since it was originated. The Archive is responsible for creating and preserving Provenance Information from the point of Ingest; however, earlier Provenance Information should be provided by the Producer. Provenance Information adds to the evidence to support Authenticity. [C1]

The Provenance Information object class implements the Provenance Information Interface. The Provenance Information class is also a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.12 Reference_Information

Reference Information: The information that is used as an identifier for the Content Information. It also includes identifiers that allow outside systems to refer unambiguously to a particular Content Information. An example of Reference Information is an ISBN. [C1]

The Reference Information object class implements the Reference Information Interface. The Reference Information class is also a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.13 Representation_Information

Representation Information: The information that maps a Data Object into more meaningful concepts. An example of Representation Information for a bit sequence which is a FITS file might consist of the FITS standard which defines the format plus a dictionary which defines the meaning in the file of keywords which are not part of the standard. Another example is JPEG software which is used to render a JPEG file; rendering the JPEG file as bits is not very meaningful to humans but the software,

which embodies an understanding of the JPEG standard, maps the bits into pixels which can then be rendered as an image for human viewing. [C1]

The Representation Information object class implements the Representation Information Interface. The Representation Information class is also a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

#### 3.2.3.14 Submission_Information_Package

Submission Information Package (SIP): An Information Package that is delivered by the Producer to the OAIS for use in the construction or update of one or more AIPs and/or the associated Descriptive Information. [C1]

The Submission Information Package object class is a subclass of the Information Package class and is composed of a Content Information class and a Preservation Description Information (PDI) class. It is also indirectly a subclass of the Information Object class and inherits its properties. It implements the Information Object Interface, is composed of a Data Object and Representation Information, is managed by an Archival Storage functional entity, and is an element of the OAIS Interoperability Framework component. This section is normative.

### 3.2.4 ABSTRACTION_LAYER

The Abstraction_Layer contains the interfaces, the well-defined entry points that define the contracts for the interoperability framework. The Abstraction Layer class is a Component. This section is normative.

#### 3.2.4.1 Interfaces

An Interface is the abstraction of a service that only defines the operations supported by that service, but not their implementations. This section is normative.

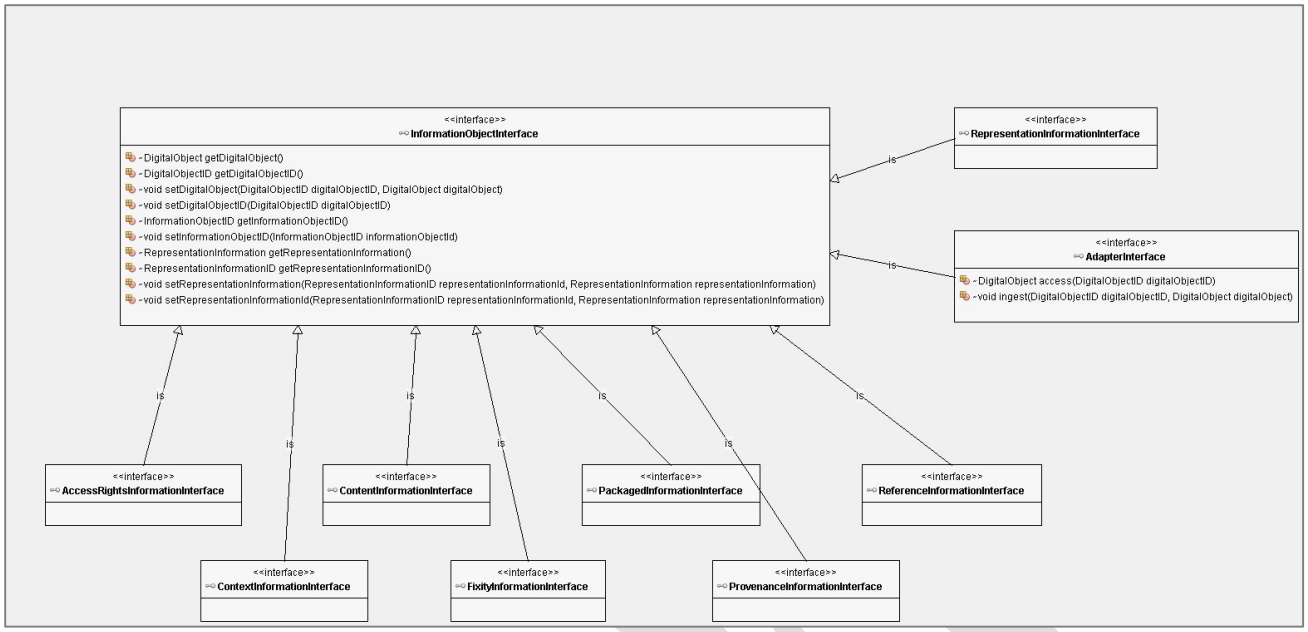


**Figure 5 – Abstraction Layer Interfaces**

#### 3.2.4.1.1 Access_Interface

The Access Interface is a well-defined entry point for the Access functional entity class. The interface requires a setAdapter method and is an element of the Abstraction Layer component. This section is normative.

#### 3.2.4.1.2 Access_Rights_Information_Interface

The Access Rights Information Interface is a well-defined entry point for accessing Access Rights Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface AccessRightsInformationInterface

  extends InfoObjectInterface
  ```

  The Access Rights Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Access Rights Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  Access Rights Information: The information that identifies the access restrictions pertaining to the Content Information, including the legal framework, licensing terms, and access control. It contains the access and distribution conditions stated within the Submission Agreement, related to both preservation (by the OAIS) and final usage (by the Consumer). It also includes the specifications for the application of rights enforcement measures.

  - **Method Summary**

    - **Methods inherited from interface InfoObjectInterface**

      getDataObject, getDataObjectID, getInfoObjectID, getRepInfo, getRepInfoDataObject, getRepInfoDataObjectID, setDataObject, setInfoObjectID, setRepInfoDataObject

**Figure 6 - Access Rights Information Interface**

#### 3.2.4.1.3 Adapter_Interface

The Adapter Interface is a well-defined entry point for the Adapter object class. The interface requires an access and an ingest method. The interface is a subclass of the Information Object Interface and inherits its methods. It is an element of the Abstraction Layer component. This section is normative.

info.oais.interfaces.oaisif

**Interface AdapterInterface**

- All Superinterfaces: InformationObjectInterface

  public interface **AdapterInterface** extends InformationObjectInterface

  The Adapter Interface is a well-defined entry point and contract for to enable the ingest and access of Digital Objects. The Adapter Interface extends the Information Object Interface and inherits methods for accessing and Information Object, its subclasses, and its components.

  - **Method Summary**

    All Methods Instance Methods Abstract Methods

| Modifier and Type | Method and Description |
|---|---|
| DigitalObject | access(DigitalObjectID digitalObjectID)<br>The access method returns a Digital Object given a digital object identifier. |
| void | ingest(DigitalObjectID digitalObjectID, DigitalObject digitalObject)<br>The ingest method registers a Digital Object using the digital object identifier and puts the Digital Object in an Object Store. |

    - **Methods inherited from interface InformationObjectInterface**

      getDigitalObject, getDigitalObjectID, getInformationObjectID, getRepresentationInformation, getRepresentationInformationID, setDigitalObject, setDigitalObjectID, setInformationObjectID, setRepresentationInformation, setRepresentationInformationId

  - **Method Detail**
    - *access*

      DigitalObject access(DigitalObjectID digitalObjectID)

      The access method returns a Digital Object given a digital object identifier.

      Parameters: digitalObjectID - an identifier for a digital object
      Returns: DigitalObject a digital object

    - *ingest*
    - void ingest(DigitalObjectID digitalObjectID,
      DigitalObject digitalObject)

      The ingest method registers a Digital Object using the digital object identifier and puts the Digital Object in an Object Store.

      Parameters: digitalObjectID - an identifier for a digital object

**Figure 7 - Adapter Interface**

#### 3.2.4.1.4 Archival_Information_Package_Interface

- All Superinterfaces:
  InfoObjectInterface, InformationPackageInterface

  ```
  public interface ArchivalInformationPackageInterface
  extends InformationPackageInterface
  ```

  The Archival Information Package Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Archival Information Package is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Data Object of this Information Object. The Data Object is a container for the Content Information and Preservation Description Information.

  Archival Information Package (AIP): An Information Package, consisting of the Content Information and the associated Preservation Description Information (PDI), which is preserved within an OAIS.

- **METHOD SUMMARY**

| All Methods | |
|---|---|
| **Modifier and Type** | **Method and Description** |
| `Object` | `getContentInformation()`<br>The getContentInformation method returns the Content Information component of this Archival Information Package. |
| `Object` | `getPreservationDescriptionInformation()`<br>The getPreservationDescriptionInformation method returns the Preservation Description Information component of this Archival Information Package. |
| `void` | `setContentInformation(Identifier identifier, InfoObject infoObject)`<br>The setContentInformation method sets the Content Information component of this Archival Information Package. |
| `void` | `setPreservationDescriptionInformation(Identifier identifier, InfoObject infoObject)`<br>The setPreservationDescriptionInformation method sets the Preservation Description Information component of this Archival Information Package. |

- **METHODS INHERITED FROM INTERFACE INFOOBJECTINTERFACE**

  `getDataObject`, `getDataObjectID`, `getInfoObjectID`, `getRepInfo`, `getRepInfoDataObject`, `getRepInfoDataObjectID`, `setDataObject`, `setInfoObjectID`, `setRepInfoDataObject`

- **METHOD DETAIL**

  - **getContentInformation**

    `Object getContentInformation()`

    The getContentInformation method returns the Content Information component of this Archival Information Package.

    Returns:

    Object the Content Information for this InfoObject

  - **getPreservationDescriptionInformation**

    `Object getPreservationDescriptionInformation()`

    The getPreservationDescriptionInformation method returns the Preservation Description Information component of this Archival Information Package.

    Returns:

    Object the Preservation Description Information for this InfoObject

  - **setContentInformation**

  - `void setContentInformation(Identifier identifier, InfoObject infoObject)`

    The setContentInformation method sets the Content Information component of this Archival Information Package.

    Parameters:

    `identifier` - the identifier for this infoObject

    `infoObject` - the Content Information component for this InfoObject

  - **setPreservationDescriptionInformation**

  - `void setPreservationDescriptionInformation(Identifier identifier, InfoObject infoObject)`

    The setPreservationDescriptionInformation method sets the Preservation Description Information component of this Archival Information Package.

    Parameters:

    `identifier` - the identifier for this infoObject

    `infoObject` - the Preservation Description Information component for this InfoObject

#### 3.2.4.1.5 Content_Data_Object_Interface

- All Superinterfaces:
  DataObjectInterface

  ```
  public interface ContentDataObjectInterface
  extends DataObjectInterface
  ```

  The Content Data Object Interface is a well-defined entry point for getting and putting the Identifier and Object of a Content Data Object.

  Content Data Object is a subclass of Data Object and inherits methods for getting and putting the Object and Identifier of this Content Data Object.

  Content Data Object: The Data Object, that together with associated Representation Information, comprises the Content Information.

  - **METHOD SUMMARY**
    - **METHODS INHERITED FROM INTERFACE DATAOBJECTINTERFACE**

      `getIdentifier`, `getObject`, `setObject`

#### 3.2.4.1.6 Content_Information_Interface

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface ContentInformationInterface

  extends InfoObjectInterface
  ```

  The Content Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Content Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  Content Information: A set of information that is the original target of preservation or that includes part or all of that information. It is an Information Object composed of its Content Data Object and its Representation Information.

  - **Method Summary**
    - **Methods inherited from interface InfoObjectInterface**

      `getDataObject`, `getDataObjectID`, `getInfoObjectID`, `getRepInfo`, `getRepInfoDataObject`, `getRepInfoDataObjectID`, `setDataObject`, `setInfoObjectID`, `setRepInfoDataObject`

The Content Information Interface is a well-defined entry point for accessing Content Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

**Figure 8 - Content Information Interface**

#### 3.2.4.1.7 Context_Information_Interface

The Context Information Interface is a well-defined entry point for accessing Context Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface ContextInformationInterface

  extends InfoObjectInterface
  ```

  The Context Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Context Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  Context Information is a subclass of Information Object and inherits Context Information: The information that documents the relationships of the Content Information to its environment. This includes why the Content Information was created and how it relates to other Content Information objects.

  - **Method Summary**
    - **Methods inherited from interface InfoObjectInterface**

      getDataObject, getDataObjectID, getInfoObjectID, getRepInfo, getRepInfoDataObject, getRepInfoDataObjectID, setDataObject, setInfoObjectID, setRepInfoDataObject

**Figure 9 - Context Information Interface**

#### 3.2.4.1.8 Data_Object_Interface.

The Data Object Interface is a well-defined entry point for accessing a Data Object. The interface requires a getObject method and is an element of the Abstraction Layer component. This section is normative.

- All Known Implementing Classes:

  DataObject

  ```
  public interface DataObjectInterface
  ```

  The Data Object Interface is a well-defined entry point for getting and putting the Identifier and Object of a Data Object.

  - **Method Summary**

| All Methods | |
|---|---|
| **Modifier and Type** | **Method and Description** |
| `Identifier` | `getIdentifier()`<br>The getIdentifier method returns the Identifier of this DataObject. |
| `Object` | `getObject()`<br>The getObject method returns the Object for this Data Object |
| `void` | `setObject(Identifier identifier, java.lang.Object object)`<br>The setObject method sets the object for this Data Object. |

  - **Method Detail**

    - **getIdentifier**

      ```
      info.oais.interfaces.oaisif.Identifier getIdentifier()
      ```

      The getIdentifier method returns the Identifier of this DataObject.

      Returns:

      Identifier the identifier for this DataObject

    - **getObject**

      ```
      getObject()
      ```

The getObject method returns the Object for this Data Object

Returns:

Object the object for this dataObject

- **setObject**
- `void setObject(Identifier identifier,`

  `java.lang.Object object)`

The setObject method sets the object for this Data Object.

Parameters:

`identifier` - the identifier for this dataObject

`object` - the object for this dataObject

**Figure 10 – Digital Object Interface**

#### 3.2.4.1.9 Fixity_Information_Interface

The Fixity Information Interface is a well-defined entry point for accessing Fixity Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface FixityInformationInterface

  extends InfoObjectInterface
  ```

  The Fixity Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Fixity Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  Fixity Information: The information which documents the mechanisms that ensure that the Content Information object has not been altered in an undocumented manner. An example is a Cyclical Redundancy Check (CRC) code for a file.

  - **Method Summary**
    - **Methods inherited from interface InfoObjectInterface**

      getDataObject, getDataObjectID, getInfoObjectID, getRepInfo, getRepInfoDataObject, getRepInfoDataObjectID, setDataObject, setInfoObjectID, setRepInfoDataObject

**Figure 11 - Fixity Information Interface**

#### 3.2.4.1.10 Identifier_Interface

The Identifier Interface is a well-defined entry point for accessing identification information. The interface requires a setAdapter method and is an element of the Abstraction Layer component. This section is normative.

#### 3.2.4.1.11 Information_Object_Interface

The Information Object Interface is a well-defined entry point for accessing an Information Object. The interface is an element of the Abstraction Layer component. This section is normative.

- All Known Implementing Classes:

  AccessRightsInformation, ContentInformation, ContextInformation, FixityInformation, InfoObject, PackagedInformation, ProvenanceInformation, ReferenceInformation, RepresentationInformation

```
public interface InfoObjectInterface
```

The InfoObject (Information Object) Interface is a well-defined entry point and contract for getting and putting Information Objects and its components.

- **Method Summary**

| All Methods | |
|---|---|
| **Modifier and Type** | **Method and Description** |
| Object | getDataObject()<br>The getDataObject method returns the Data Object component of this Information Object. |
| Identifier | getDataObjectID()<br>The getDataObjectID method returns the identifier of the Data Object component of this Information Object. |
| Identifier | getInfoObjectID()<br>The getInfoObjectID method returns the identifier of this Information Object. |
| InfoObject | getRepInfo()<br>The getRepInfo method returns the Representation Information component of this Information Object. |
| Object | getRepInfoDataObject() |

| | |
|---|---|
| | The getRepInfoDataObject method returns a Data Object component of this Information Object. |
| `Identifier` | `getRepInfoDataObjectID()`<br><br>The getRepInfoDataObjectID method returns the Identifier of a Data Object component of this Information Object. |
| `void` | `setDataObject(Identifier identifier, DataObject dataObject)`<br><br>The setDataObject method sets the Data Object component of this Information Object. |
| `void` | `setInfoObjectID(Identifier identifier)`<br><br>The setInfoObjectID method sets the identifier of this Information Object. |
| `void` | `setRepInfoDataObject(Identifier identifier, RepInfoDataObject repInfoDataObject)`<br><br>The setRepInfoDataObject method sets a Data Object component of this Information Object. |

- **Method Detail**
  - **getInfoObjectID**

    ```
    getInfoObjectID()
    ```

    The getInfoObjectID method returns the identifier of this Information Object.

    Returns:

    Identifier the identifier of this InfoObject

  - **getDataObjectID**

    ```
    getDataObjectID()
    ```

    The getDataObjectID method returns the identifier of the Data Object component of this Information Object.

    Returns:

Identifier the identifier of the DataObject

- **getRepInfoDataObjectID**

```
getRepInfoDataObjectID()
```

The getRepInfoDataObjectID method returns the Identifier of a Data Object component of this Information Object. More specifically this method returns the Identifier of the Data Object of the Representation Information for this Information Object.

Returns:

Identifier the identifier of the RepInfoDataObject

- **getDataObject**

```
getDataObject()
```

The getDataObject method returns the Data Object component of this Information Object.

Returns:

Object the DataObject for this InfoObject

- **getRepInfo**

```
InfoObject getRepInfo()
```

The getRepInfo method returns the Representation Information component of this Information Object.

Returns:

InfoObject the RepInfo for this InfoObject

- **getRepInfoDataObject**

```
getRepInfoDataObject()
```

The getRepInfoDataObject method returns a Data Object component of this Information Object. More specifically this method returns the Data Object of the Representation Information for this Information Object.

Returns:

Object the DataObject for this InfoObject's RepInfo

- **setInfoObjectID**

```
void setInfoObjectID(Identifier identifier)
```

The setInfoObjectID method sets the identifier of this Information Object.

Parameters:

`identifier` - the identifier for this InfoObject

- **setDataObject**
- `void setDataObject(Identifier identifier,`

  `DataObject dataObject)`

The setDataObject method sets the Data Object component of this Information Object.

Parameters:

`identifier` - the identifier for this DataObject

`dataObject` - the DataObject for this InfoObject

- **setRepInfoDataObject**
- `void setRepInfoDataObject(Identifier identifier,`

  `RepInfoDataObject repInfoDataObject)`

The setRepInfoDataObject method sets a Data Object component of this Information Object. More specifically this method sets the Data Object of the Representation Information for this Information Object.

Parameters:

`identifier` - the identifier for this dataObject

`repInfoDataObject` - the RepInfoDataObject for this InfoObject's RepInfo

**Figure 12 - Information Object Interface**

#### 3.2.4.1.12 Ingest_Interface

The Ingest Interface is a well-defined entry point for the Ingest functional entity class. The interface requires a setAdapter method and is an element of the Abstraction Layer component. This section is normative.

### 3.2.4.1.13 Message_Interface

The Message Interface is a well-defined entry point for the Message object class. The interface requires the following methods: agreeStrategy, getReceiver, getSender, isInfoUsable, orderRequest, queryRequest, requestPackage, requestRepInfoIDs, and sendPackagethod. The interface is an element of the Abstraction Layer component. This section is normative.

info.oais.interfaces.oaisif

**Interface MessageInterface**

```
public interface MessageInterface
```

- **Method Summary**

All Methods Instance Methods Abstract Methods

| Modifier and Type | Method and Description |
|---|---|
| RepresentationInformationID | agreeStrategy(RepresentationInformationID or iid1)<br>The agreeStrategy method allows a Reference Implementation specification (Representation Information of reference implementations) to be sent to a requestor for consideration as a candidate for enabling inter-operation. |
| boolean | isInfoUsable()<br>The isInfoUsable method returns a value of true if this Information Object is useful, otherwise it returns false. |
| Identifier[] | orderRequest(Object request)<br>The orderRequest method, given a request object, returns a list of Packaged Information Object Identifiers. |
| Identifier[] | queryRequest(Object query)<br>The queryRequest method, given a query object, returns a list of Information Object Identifiers. |
| Object | requestPackage(Identifier id1)<br>The requestPackage method returns the identified Information Package |
| RepresentationInformationID | requestRepresentationInformationID()<br>The requestRepresentationInformationIDs method returns the Representation Information Identifier of this Information Object. |
| void | sendPackage(PackagedInformation pdi, Identifier i1)<br>The sendPackage method sends Packaged Information to a requestor |
| void | sendRepresentationInformationPackage(RepresentationInformationID oriid, PackagingInformation pi, PackagedInformation pdi)<br>The sendRepresentationInformationPackage method sends an Representation Information Package to a requestor |

- **Method Detail**
  - **agreeStrategy**

    `RepresentationInformationID agreeStrategy(RepresentationInformationID oriid1)`

    The agreeStrategy method allows a Reference Implementation specification (Representation Information of reference implementations) to be sent to a requestor for consideration as a candidate for enabling inter-operation.

    The returned Identifier, if equal to the one sent, becomes the active Reference Implementation. A new candidate is being suggested if the returned Identifier is not equal to the one sent.

    Parameters: `oriid1` - the Representation Information Identifier of the candidate Reference Implementation.
    Returns: RepresentationInformationID the Representation Information of the from the original requestor.

  - **isInfoUsable**

    `boolean isInfoUsable()`

    The isInfoUsable method returns a value of true if this Information Object is useful, otherwise it returns false.

    Returns: true if this Information Object is useful, otherwise return false.

  - **orderRequest**

    `Identifier[] orderRequest(Object request)`

    The orderRequest method, given a request object, returns a list of Packaged Information Object Identifiers.

    Parameters: `request` - a request object
    Returns: Identifier a list of Information Object identifiers.

  - **queryRequest**

    `Identifier[] queryRequest(Object query)`

    The queryRequest method, given a query object, returns a list of Information Object Identifiers.

Parameters: `query` - a query object
Returns: Identifier a list of Information Object identifiers.

- **requestPackage**

  `Object requestPackage(Identifier id1)`

  The requestPackage method returns the identified Information Package

  Parameters: `id1` - an identifier of an Information Package
  Returns: Object

- **requestRepresentationInformationID**

  `RepresentationInformationID requestRepresentation InformationID()`

  The requestRepresentationInformationIDs method returns the Representation Information Identifier of this Information Object.

  Returns: RepresentationInformationID the Representation Information Identifier

- **sendPackage**
- `void sendPackage(PackagedInformation pdi, Identifier i1)`

  The sendPackage method sends Packaged Information to a requestor

  Parameters: `pdi` - the Packaged Information
  `i1` - an identifier

- **sendRepresentationInformationPackage**
- `void sendRepresentationInformationPackage(Representationinformationid oriid,`
- `PackagingInformation pi,`

  `PackagedInformation pdi)`

  The sendRepresentationInformationPackage method sends an Representation Information Package to a requestor

  Parameters: `oriid` - the Representation Information Identifier
  `pi` - the Packaging Information
  `pdi` - the Packaged Information

**Figure 13 – Message Interface**

#### 3.2.4.1.14 Message_Service_Interface

The Message Service Interface is a well-defined entry point and contract for the expected behaviour of both service providers and users of a service. The operations of the interface are triggered by the exchange of messages in predefined interaction patterns.The class has Send, Request, Submit, Invoke, Progress, and PublishSubscribe methods. The interface is an element of the Abstraction Layer component. This section is normative.

#### 3.2.4.1.15 Negotiate_Interface

The Negotiate Interface is a well-defined entry point for the Negotiate object class. The interface requires an agreeStrategy and a chooseAdapter method. It is an element of the Abstraction Layer component. This section is normative.

#### 3.2.4.1.16 Packaged_Information_Interface

The Packaged Information Interface is a well-defined entry point for accessing Packaged Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface PackagedInformationInterface

  extends InfoObjectInterface
  ```

  The Packaged Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Packaged Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  - **Method Summary**
    - **Methods inherited from interface InfoObjectInterface**

      getDataObject, getDataObjectID, getInfoObjectID, getRepInfo, getRepInfoDataObject, getRepInfoDataObjectID, setDataObject, setInfoObjectID, setRepInfoDataObject

**Figure 14 - Packaged Information Interface**

#### 3.2.4.1.17 Provenance_Information_Interface

The Provenance Information Interface is a well-defined entry point for accessing Provenance Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface ProvenanceInformationInterface

  extends InfoObjectInterface
  ```

  The Provenance Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Provenance Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  Provenance Information: The information that documents the history of the Content Information. This information tells the origin or source of the Content Information, any changes that may have taken place since it was originated, and who has had custody of it since it was originated. The Archive is responsible for creating and preserving Provenance Information from the point of Ingest; however, earlier Provenance Information should be provided by the Producer. Provenance Information adds to the evidence to support Authenticity.

  - **Method Summary**
    - **Methods inherited from interface InfoObjectInterface**

      getDataObject, getDataObjectID, getInfoObjectID, getRepInfo, getRepInfoDataObject, getRepInfoDataObjectID, setDataObject, setInfoObjectID, setRepInfoDataObject

**Figure 15 – Provenance Information Interface**

#### 3.2.4.1.18 Reference_Information_Interface

The Reference Information Interface is a well-defined entry point for accessing Reference Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface ReferenceInformationInterface

  extends InfoObjectInterface
  ```

  The Reference Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Reference Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  Reference Information: The information that is used as an identifier for the Content Information. It also includes identifiers that allow outside systems to refer unambiguously to a particular Content Information. An example of Reference Information is an ISBN.

  - **Method Summary**
    - **Methods inherited from interface InfoObjectInterface**

      getDataObject, getDataObjectID, getInfoObjectID, getRepInfo, getRepInfoDataObject, getRepInfoDataObjectID, setDataObject, setInfoObjectID, setRepInfoDataObject

**Figure 16 – Reference Information Interface**

#### 3.2.4.1.19 Representation_Information_Interface

The Representation Information Interface is a well-defined entry point for accessing Representation Information. The interface is a subclass of the Information Object Interface and inherits its methods. The interface is an element of the Abstraction Layer component. This section is normative.

- All Superinterfaces:

  InfoObjectInterface

  ```
  public interface RepresentationInformationInterface

  extends InfoObjectInterface
  ```

  The Representation Information Interface is a well-defined entry point and contract for getting and putting this Information Object and its components.

  Representation Information is a subclass of Information Object and inherits methods for getting and putting the component Representation Information and Digital Object of this Information Object.

  Representation Information: The information that maps a Data Object into more meaningful concepts. An example of Representation Information for a bit sequence which is a FITS file might consist of the FITS standard which defines the format plus a dictionary which defines the meaning in the file of keywords which are not part of the standard. Another example is JPEG software which is used to render a JPEG file; rendering the JPEG file as bits is not very meaningful to humans but the software, which embodies an understanding of the JPEG standard, maps the bits into pixels which can then be rendered as an image for human viewing.

  - **Method Summary**
    - **Methods inherited from interface InfoObjectInterface**

      getDataObject, getDataObjectID, getInfoObjectID, getRepInfo, getRepInfoDataObject, getRepInfoDataObjectID, setDataObject, setInfoObjectID, setRepInfoDataObject

**Figure 17 - Representation Information Interface**

#### 3.2.4.1.20 Representation Information Data Object Interface

The Representation Information Data Object Interface is a well-defined entry point for accessing the Data Object of an Information Object's Representation. The interface is an element of the Abstraction Layer component. This section is normative.

- All Known Implementing Classes:

  RepInfoDataObject

  `public interface RepInfoDataObjectInterface`

  The Representation Information Data Object Interface is a well-defined entry point for getting and putting the Identifier and Object of a Data Object, the Data Object of an Information Object's Representation Information.

  - **Method Summary**

| All Methods | |
|---|---|
| **Modifier and Type** | **Method and Description** |
| `Identifier` | `getIdentifier()`<br>The getIdentifier method returns the Identifier of this RepInfoDataObject. |
| `Object` | `getObject()`<br>The getObject method returns the Object for this RepInfoDataObject |
| `void` | `setObject(Identifier identifier, Object object)`<br>The setObject method sets the Object for this RepInfoDataObject. |

  - **Method Detail**

    - **getIdentifier**

      `info.oais.interfaces.oaisif.Identifier getIdentifier()`

      The getIdentifier method returns the Identifier of this RepInfoDataObject.

      Returns:

      Identifier the identifier for this repInfoDataObject

    - **getObject**

`getObject()`

The getObject method returns the Object for this RepInfoDataObject

Returns:

Object the object for this repInfoDataObject

- **setObject**
- `void setObject(Identifier identifier, Object object)`

The setObject method sets the Object for this RepInfoDataObject.

Parameters:

`identifier` - the identifier for this repInfoDataObject

`object` - the object for this repInfoDataObject

**Figure 18 - Representation Information Data Object Interface**

### 3.2.5 ADAPTER LAYER

An Adapter (Binding) is a wrapper library that bridges two programming languages, so that a library written for one language can be used in another language. The adapter interface is normative. The listed adapters are informative.

#### 3.2.5.1 Adapter

An Adapter (Binding) is a wrapper library that bridges two programming languages, so that a library written for one language can be used in another language. Within the OAIS-IF an adaptor provides a bridge between the OAIS Information Model and an information model used by an archive. More specifically the necessary and sufficient function for an Adapter to be able to return the appropriate Representation Information for an Information Object.

The Adapter object class is an abstract class from which specialized adapters are extended. It implements the Adapter Interface which provides access and ingest methods and inherited get and put methods on Data Objects and Representation Information. It also implements the Message Interface which provides a set of standard messages and the Message Service Interface which provides a standard protocol or interaction pattern for communication. Finally, it implements the Identifier Interface which has been extended to handle local Identifiers. The Adapter's access and ingest methods

can use either programmatic function calls or Application Programming Interfaces (APIs) to invokes Local Access and Ingest functional entities in the Archive interface. It is an element of the Adapter Layer within the OAIS Interoperability Framework component. This section is normative.

## 3.3 OAIS_IF_ARCHIVE

An OAIS IF Archive is an organization that intends to preserve information for access and use by a Designated Community and acknowledges the OAIS IF can be used to interoperate with other OAIS IF Archives. This section is informative.

### 3.3.1 OAIS_IF_ARCHIVE_INTERFACE

The OAIS IF Archive Interface is a well-defined entry point for the OAIS_IF_Archive and provides a contract for the exchange of information.

#### 3.3.1.1 Local_Access

The Local Access class provides an access capability for an Archive.

The Local Access object class provides the functions to locate and retrieve information packages in the Archive. The Local Access class is an element of the Archive Interface component. A special Adapter must address any OAIS-IF requirement that is not met by a corresponding Local Access instance. This section is informative.

#### 3.3.1.2 Local_Ingest

The Local Ingest class provides an ingest capability for an Archive.

The Local Ingest object class provides the functions to accept information packages and register them in the Archive. A special Adapter must address any OAIS-IF requirement that is not met by a corresponding Local Ingest instance. The Local Ingest class is an element of the Archive Interface component. This section is informative.

### 3.3.2 ARCHIVAL_STORAGE

Archival Storage Functional Entity (aka Archival Storage): The OAIS functional entity that contains the services and functions used for the storage and retrieval of Archival Information Packages.

The Archival Storage class is a subclass of Component. This section is informative.

# ANNEX A

# IMPLEMENTATION CONFORMANCE STATEMENT (ICS) PROFORMA

# (NORMATIVE)

## A1 INTRODUCTION

### A1.1 OVERVIEW

This annex provides the Implementation Conformance Statement (ICS) Requirements List (RL) for an implementation of [Specification]. The ICS for an implementation is generated by completing the RL in accordance with the instructions below. An implementation claiming conformance must satisfy the mandatory requirements referenced in the RL.

### A1.2 ABBREVIATIONS AND CONVENTIONS

The RL consists of information in tabular form. The status of features is indicated using the abbreviations and conventions described below.

Item Column

The item column contains sequential numbers for items in the table.

Feature Column

The feature column contains a brief descriptive name for a feature. It implicitly means "Is this feature supported by the implementation?"

Status Column

The status column uses the following notations:

- M mandatory;
- O optional;
- C conditional;
- X prohibited;
- I out of scope;
- N/A not applicable.

Support Column Symbols

The support column is to be used by the implementer to state whether a feature is supported by entering Y, N, or N/A, indicating:

Y Yes, supported by the implementation.

N No, not supported by the implementation.

N/A Not applicable.

The support column should also be used, when appropriate, to enter values supported for a given capability.

## A1.3 INSTRUCTIONS FOR COMPLETING THE RL

An implementer shows the extent of compliance to the Recommended Standard by completing the RL; that is, the state of compliance with all mandatory requirements and the options supported are shown. The resulting completed RL is called an ICS. The implementer shall complete the RL by entering appropriate responses in the support or values supported column, using the notation described in A1.2. If a conditional requirement is inapplicable, N/A should be used. If a mandatory requirement is not satisfied, exception information must be supplied by entering a reference X*i*, where *i* is a unique identifier, to an accompanying rationale for the noncompliance.

# A2 ICS PROFORMA FOR [SPECIFICATION]

## A2.1 GENERAL INFORMATION

### A2.1.1 Identification of ICS

#### A2.1.1.1 Test

| | |
|---|---|
| Date of Statement (DD/MM/YYYY) | |
| ICS serial number | |
| System Conformance statement cross-reference | |

### A2.1.2 Identification of Implementation Under Test

| | |
|---|---|
| Implementation Name | |
| Implementation Version | |
| Special Configuration | |

| | |
|---|---|
| Other Information | |

### A2.1.3 Identification of Supplier

| | |
|---|---|
| Supplier | |
| Contact Point for Queries | |
| Implementation Name(s) and Versions | |
| Other information necessary for full identification, e.g., name(s) and version(s) for machines and/or operating systems;<br><br>System Name(s) | |

### A2.1.4 Identification of Specification

| [CCSDS Document Number] | |
|---|---|
| Have any exceptions been required?<br><br>NOTE – A YES answer means that the implementation does not conform to the Recommended Standard. Non-supported mandatory capabilities are to be identified in the ICS, with an explanation of why the implementation is non-conforming. | Yes [ ] No [ ] |

## A2.2 REQUIREMENTS LIST

[See CCSDS A20.1-Y-1, *CCSDS Implementation Conformance Statements* (Yellow Book, Issue 1, April 2014).]

# ANNEX B

# SECURITY, SANA, AND PATENT CONSIDERATIONS

# (INFORMATIVE)

## B1 SECURITY CONSIDERATIONS

### B1.1 SECURITY CONCERNS WITH RESPECT TO THE CCSDS DOCUMENT

#### B1.1.1 Data Privacy

#### B1.1.2 Data Integrity

#### B1.1.3 Authentication of Communicating Entities

#### B1.1.4 Control of Access to Resources

#### B1.1.5 Availability of Resources

#### B1.1.6 Auditing of Resource Usage

### B1.2 POTENTIAL THREATS AND ATTACK SCENARIOS

### B1.3 CONSEQUENCES OF NOT APPLYING SECURITY TO THE TECHNOLOGY

## B2 SANA CONSIDERATIONS

[See CCSDS 313.0-Y-1, *Space Assigned Numbers Authority (SANA)—Role, Responsibilities, Policies, and Procedures* (Yellow Book, Issue 1, July 2011).]

## B3 PATENT CONSIDERATIONS

[See CCSDS A20.0-Y-4, *CCSDS Publications Manual* (Yellow Book, Issue 4, April 2014).]

# ANNEX C

# INFORMATIVE REFERENCES

# (INFORMATIVE)

[C1] *Reference Model For An Open Archival Information System (OAIS)*. Issue 2.1. CCSDS Record (Pink Book), CCSDS 650.0-P-2.1. Washington, D.C.: CCSDS, October 2020.