Records Management Services (RMS), v1.0

OMG Document Number: dtc/2010-12-19

Associated Documents

Title	Document Number
RMS Version 1.0 WSDL Files	**dtc/2010-11-33**
RMS Version 1.0 XSD Files	**dtc/2010-11-34**
XMI File of the RMS Version 1.0 UML Model	**dtc/2010-12-20**

applied), gov/09-03-08 (Non-normative Enterprise Architect Files), gov/09-03-10 (machine-readable artifacts)

This OMG document replaces the submission document (gov/2009-03-03, Alpha). It is an OMG Adopted Beta Specification and is currently in the finalization phase. Comments on the content of this document are welcome, and should be directed to *issues@omg.org* by December 4, 2009.

You may view the pending issues for this specification from the OMG revision issues web page *http://www.omg.org/issues/.*

The FTF Recommendation and Report for this specification will be published on February 2, 2010. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

LICENSES

The companies listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that: (1) both the copyright notice identified above and this permission notice appear on any copies of this specification; (2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or

transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

GENERAL USE RESTRICTIONS

DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227- 7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 140 Kendrick Street, Needham, MA 02494, U.S.A.

TRADEMARKS

MDA®, Model Driven Architecture®, UML®, UML Cube logo®, OMG Logo®, CORBA® and XMI® are registered trademarks of the Object Management Group, Inc., and Object Management Group™, OMG™ , Unified Modeling Language™, Model Driven Architecture Logo™, Model Driven Architecture Diagram™, CORBA logos™, XMI Logo™, CWM™, CWM Logo™, IIOP™ , IMM™, MOF™ , OMG Interface Definition Language (IDL)™ , and OMG Systems Modeling Language (OMG SysML)™ are trademarks of the Object Management Group. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

OMG's Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page *http://www.omg.org*, under Documents, Report a Bug/Issue (http://www.omg.org/technology/agreement.htm).

Table of Contents

Preface

About the Object Management Group

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at *http://www.omg.org/.*

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. A catalog of all OMG Specifications is available from the OMG website at:

http://www.omg.org/technology/documents/spec_catalog.htm

Specifications within the Catalog are organized by the following categories:

OMG Modeling Specifications

- UML
- MOF
- XMI
- CWM
- Profile specifications

OMG Middleware Specifications

- CORBA/IIOP
- IDL/Language Mappings
- Specialized CORBA specifications
- CORBA Component Model (CCM)

Platform Specific Model and Interface Specifications

- CORBAservices
- CORBAfacilities
- OMG Domain specifications
- OMG Embedded Intelligence specifications
- OMG Security specifications.

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
140 Kendrick Street
Building A, Suite 300
Needham, MA 02494
USA
Tel: +1-781-444-0404 Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult *http://www.iso.org*

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and

syntax elements. **`Courier - 10 pt. Bold:`** Programming language elements.

Helvetica/Arial - 10 pt: Exceptions

Note – Terms that appear in *italics* are defined in the glossary. Italic text also represents the name of a document, specification, or other publication.

Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to *http://www.omg.org/ technology/agreement.htm.*

1 Scope of the Specification

This specification provides models for software services to support management activities for electronic records. Following the OMG MDA framework, models are provided that describe the platform independent model (PIM) which defines the business domain of Records Management and the RM services to be provided.Three technology-specific implementations are specified:

1. PSM-1 – Web Services definition for Records Management Services in Web Service Description Language (WSDL). This is actually supplied as ten WSDL files; one for each Records Management Service.

2. PSM-2 – A Records Management Service XSD. The XSD is for use in creating XML files for import/export of Managed Records from compliant environments and to use as a basis for forming XQuery/XPath statements for the query service.

3. PSM-3 – An Attribute Profile XSD. The XSD is for capturing and communicating attribute profiles to permit flexible attribution of certain types of Records Management Objects.

The scope of the services to be modeled extends from record receipt, identification and capture to record disposition. It is the business owner who decides when and what to set aside as a record. Once that decision is made, the required services can be applied to the record, assuring its proper management and disposition. The record as set aside by the business owner remains unchanged even as the record's management attributes are populated and updated during its life-cycle. The sum of a record and its records management attributes (current and historical) is a managed record.

Human factors, knowledge, and business rules are brought to bear on the activities of deciding what makes up a record, how it should be categorized or aggregated, how long it should be retained and its ultimate disposition. These activities and the decisions behind them for the most part require human intervention and therefore cannot be directly supported by software services. Once decided, however, many of the activities may be implemented or carried out by software services. Hence, a records management service cannot create a disposition schedule; it can implement disposition based on criteria the business owner provides that is executable within a computing environment.

Given these considerations, the following items are specifically excluded from the scope of the services in compliance with the RFP.

- Storage methods and media
- Storage location (where location is mentioned in the specification, it is a logical concept and has nothing to do with a physical location or server).
- Requirements for privacy and information security
- Security classification and declassification
- Disposition schedule creation and maintenance (including Category, and Category Schema creation and maintenance)

- Pre-established Annotation creation and maintenance.
- Creation or Maintenance of entities in the Party model (i.e., these are presumed to pre-exist for use by the services.)

In records management practice, there are relationships and dependencies among the business activities to be supported by the required services. Those relationships and dependencies are not to be made explicit in the logical specification of the services (i.e., the PIM). Instead, each service is specified independently, allowing each to be implemented by itself or in conjunction with other services to address requirements across the record life cycle. However, it is important that all such business relationships and dependencies are captured in an overall model of the context of use of the services (i.e., the CIM), in order to make it clear how the proposed services may be composed in order to meet the business needs of records management.

Further, there are a number of general services that are necessary to the implementation of records management, but are normally provided as part of the operating environment, not specifically as part of the records management implementation. Such supporting services include

- Search and retrieval
- Transfer and destruction
- Identification and authentication
- Authorization and access control
- Audit logging of user actions
- Systems management
- Maintenance, backup, recovery (both system and disaster)

Such services are also excluded from this specification of services as requested by the RFP. However, the submission team felt it was imperative to include query services in order to make the RMS commercially viable, so a query service was included.

2 Conformance Criteria

Conformant software shall:

1. Support Records Management functions as specified in Sections 2.8 forward in a Web Services client/server environment using the provided Web Service Definition Language files. (The appendices are to be considered informative).

2. Be able to produce and consume XML documents based on the RMS XSD for the purpose of exchanging ManagedRecord information.

3. Be able to form and respond to XQuery statements through the query service using the RMS XSD to form the query.

4. Be able to form and consume XML documents based on the RMS Attribute Profile XSD for the purpose of exchanging information on Data Profiles, and of instances of data described in those profiles.

3 Normative References

http://www.w3.org/TR/xmlschema-1/ - XML Schema Part 1: Structures Second Edition, W3C Recommendation, 28 October 2004

http://www.w3.org/TR/xmlschema-2/ - XML Schema Part 2: Datatypes Second Edition W3C Recommendation, 28 October 2004

http://www.w3.org/TR/wsdl - Web Services Description Language (WSDL) 1.1, W3C Note, 15 March 2001

4 Terms and Definitions

Term	Definition
Archival Bond	[RMSC] The interrelationships between a record and other records resulting from the same business act, transaction, or process, to one or more previous and subsequent records resulting from the same type business act, transaction, or process within a specific time period. Usually accomplished by associating the records to each other through a record category.
Authenticated Record	[RMSC] A record with a populated authenticity indicator attribute that provides the benchmark for subsequent validation of authenticity during the entire record life cycle.
Benchmark	[RMSC] A standard by which something can be judged or measured.
Captured Case File Record	[RMSC] A uniquely identified record carrying the date it was initially controlled as a record within an electronic environment.
Captured Record	[RMSC] A uniquely identified declared record carrying the date it was initially controlled as a record within an electronic environment along with the record creator unique identifier. A captured record is to be considered synonymous with other names used within an electronic environment such as object, electronic object, coherent information, and file, etc.
Captured Records	[RMSC] Plural form of captured record. A uniquely identified declared record carrying the date it was initially controlled as a record within an electronic environment along with the record creator unique identifier. Captured record is to be considered synonymous with other names used within an electronic environment such as object, electronic object, coherent information, and file, etc.
Case File	[RMSC] A collection of documents (a file) relating to a specific action, transaction, event, person, place, project, investigation or other subject.[1]
Case File Part	[RMSC] An individual item (e.g., document, file, record) that with others makes up the case file.
Categorization	[RMSC] Any scheme developed or used by an agency to organize

[1] Society of American Archivists, A Glossary of Archival and Records Terminology, s.v., "case file" – "Syn: subject file; transactional file DF: dossier. Case files are sometimes referred to as a project file or, in Canada, a transactional file. Also called dossiers, although that term has a more general sense of file. They are often found in the context of social services agencies (public and private), and Congressional papers."

Term	Definition
Schema	records. This may include a diagrammatic representation or outline of the descriptive classification assigned to records or records disposition codes.
Declared Case File Record	[RMSC] An electronic document or object that is considered by the business owner to evidence one or more organization, function, policy, decision, procedure, transaction, or activity completely enough to be maintained and managed as a record, either for the conduct of current business or for future reference.[2]
Declared Record	[RMSC] An electronic document or object that is considered by the business owner to evidence one or more organization, function, policy, decision, procedure, transaction, or activity completely enough to be maintained and managed as a record, either for the conduct of current business or for future reference.
Disposition Authority	[RMSC] The legally binding instrument that authorizes the disposition of records, regardless of business environment e.g. for Federal records usually the SF 115 approved by the Archivist, for Presidential records the Presidential Record Act, for the financial records of publicly-held companies the Sarbanes-Oxley Act, etc.
Disposition Instruction	[RMSC] Mandatory and specific directions, derived from a disposition authority, that guide the retention and disposal of a record, including retention periods, dates for action, etc.
Document	[JRMS Issue 8] Any bit string. When "set-aside" the "Document" becomes a "ManagedRecord", or part of a ManagedRecord.
Identifier	[RMSC] The name, position, application or system designation (or concatenation of that data and other data about the actor and/or the environment) differentiating one actor from another.
Identified Record	[RMSC] A record that has a populated attribute differentiating it from all other records within the electronic environment.
Managed Record	[RMSC] A record as set aside by a business owner that has been subject to records management activities.
Provenance	[RMSC] Ties the record to the circumstances of its creation at the time of its creation and maintains this information throughout the record's active use for business purposes. Provenance establishes the person or system and the agency in which the record was created or received, the

[2] Meant to be synonymous with Declare Record in Capture Record Service.

Term	Definition
	record keeper responsible for the record custody, the date upon which that record keeper assumed that responsibility, and the identity and chronology of subsequent custodians(s), if applicable. [SAA] n. (provenancial, adj.); 1. The origin or source of something. – 2. Information regarding the origins, custody, and ownership of an item or collection. Notes: Provenance is a fundamental principle of archives, referring to the individual, family, or organization that created or received the items in a collection. The principle of provenance or the respect des fonds dictates that records of different origins (provenance) be kept separate to preserve their context.
Provenancial Record	[RMSC] A record for which information about its time and place of its creation has been collected and preserved. This information supports the reliability of the record as evidence of its creator and the activity from which it results.
Record Category	[RMSC] A descriptive term that identifies the relationships between a record and other records resulting from the same business activity; one way of implementing archival bond.
Record Creator	[RMSC] An individual, application, or system procedure in an electronic environment specifically designed in accordance with the business rules to carry out the legal authorities of the organization to which the individual, application, or system procedure belongs. [JRMS Issue 9] It's important not to confuse the creator of the document as an electronic artifact, and the creation of a "record" The person (or actor) who identifies it as a record and sets it aside is the RecordCreator… the attribute identifying the document creator may be "unknown"
Record Keeper	[RMSC] The administrative entity, unit, office, or person responsible for the custody and ongoing management of the records during their active business use.
Revocation Order	[RMSC] A legally binding, or a legitimate order or notice to release a suspend disposition authority.
Scheduled Record	[RMSC] A record with a disposition instruction (transfer, retention, or destruction) from an established disposition authority.
Suspend	[RMSC] A legally binding order, notice, or freeze on the execution of

Term	Definition
Disposition Authority	the established disposition instruction of an established disposition authority.
Suspended Record	[RMSC] A scheduled record which is subject to at least one suspend disposition authority.
System Date	[RMSC] The calendar date made available within the electronic environment, usually provided as a service by the operating system for use by programs, applications and other executable operations.
Transfer	[JRMS] In a Record Transfer, the legal custody of Record moves from the current organization of provenance to another Authority.
Move	[JRMS] A move is not a record transfer (see Transfer). It is the change of physical control to another location without transfer of legal custody. All the metadata stays with RME of the legal custodian

5 Symbols and Typographical Conventions

Model class color conventions:

- ManagedRecord is the central concept of the Records Management Service specification. It is shown as a "Yellow" Class in class diagrams.
- In each package's diagrams, those classes that are defined in that package appear as "Salmon" in color. Those classes that are defined in other packages appear "Grey" in color.

6 Additional Information

6.1 Specification Overview

6.1.1 Platform Independent Models

Section 2.8, "Platform Independent Model" contains the normative Platform Independent Model of the RMS Specification. It consists of:

- 2.8.2, Package: RmsDomainModel"
  This can also be interpreted as the Computation Independent Model (CIM). It is based on the domain concepts found in the CIM specified in the Requirements Document: "Functional Requirements, Attributes, and Unified Modeling Language Class Diagrams for Records, Management Services," 7-September-2006, produced by the 19 U.S. Interagency Project Team and the Records Management Service Components Program Office of the National Archives and Records Administration.
- 2.8.3, " Package: AttributeProfile"
  This facility provides for flexible attribution of Records Management Objects consistent with extant standards.
- 2.8.4, "Package: RmsServices"
  Provides the model of the services defined for Records Management, consisting of:
  - 2.8.4.1, "RmsSolution"
    The RmsSolution package contains elements that represent clients of the RMS services. These are generally referred to as RMS Clients and RMS Applications.
  - 2.8.4.2 "RmsProcessServices"
    This package is an architectural placeholder for process-related services. The current version of the specification is meant to be independent of business process that generate records. Future versions of the specification may include process services that manage records management functions.
  - 2.8.4.3 "RmsCoreServices"
    This package contains the specifications of the specifications of the RM Services in the Core Business Layer of the RMS architecture.
  - 2.8.4.4 "RmsUtilityServices"
    Includes utility packages. Specifically the definition of the AttributeProfile Service and the Parties Service.

The platform independent model is fully defined in the normative XMI file, RMS.xmi.xml

6.1.2 Platform Specific Models

Section 2.9, "Platform Specific Model" describes the machine-readable files constituting the Platform Specific Model of the RMS Specification.

Two platform specific models are defined: a set of XSD models for RMS information exchange and a set of WSDL models defining the web services of the RMS and their operations. Intermediary models for the XSD's and WSDL were automatically generated from the descriptions in the PIM through an MDA tool. In the case of the XSD files those models were intermediary machine-readable artifacts which required hand editing to complete. The heuristic for the files is documented in the section "Platform Specific Models"

6.2 Design Rationale

6.2.1 Computer Independent Model (CIM)

The specification's Platform Independent Model (PIM) was developed by directly evolving the CIM (Business Domain Model) of Records Management Component Services requirements document produced by an Interagency Project Team of 19 US Federal Agencies (see reference [RMSC]) and essentially remains so. It is used as a PIM in conjunction with the AttributeProfile package.

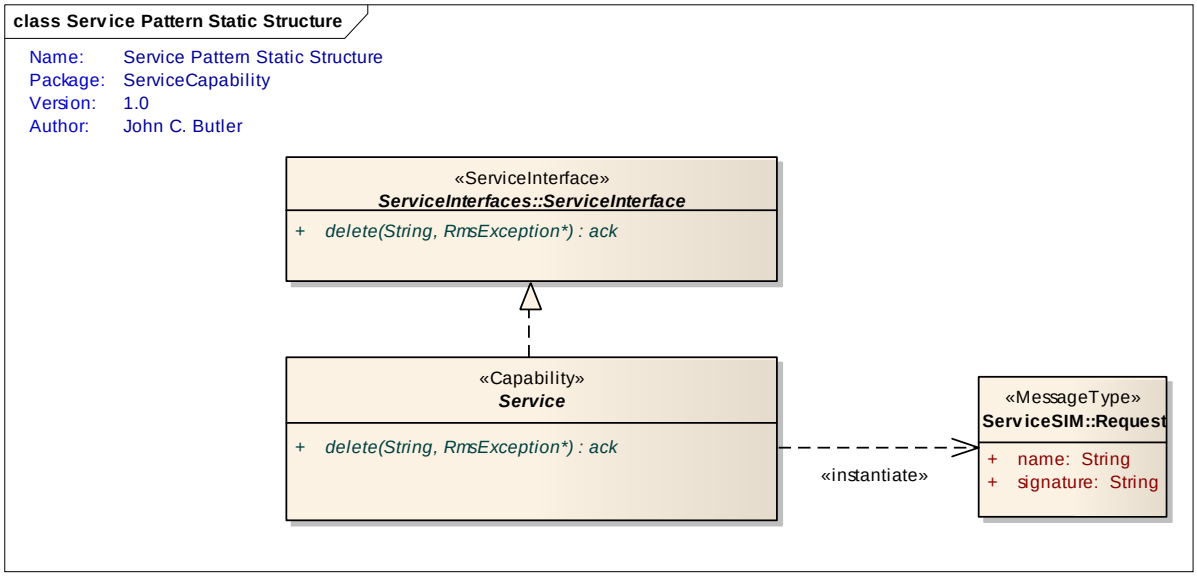
6.2.2 Attribute Profile Facility

The basic model has been intentionally kept minimal. Through the attribute profile facility further standard profiles can be defined to enable attribution for multiple communities. It enables subtypes of AttributableObject to be attributed according to a specific profile of attribute definitions.

6.2.3 Service Specification Patterns

The Service Pattern package on which the Records Management Services were built consisted of the documentation of a desired capability realized by a service interface. The capability instantiates a service information model that is derived from the Business Domain, Attribute Profile, and related models

Service Pattern Static Structure



6.2.4 Package Fine Structure

When the specifications get down to specific details of Records Management (in information models or service models), the definitions are focused around functional packages as follows:

- Annotation
- Authenticity
- Category
- Document
- Dispositions
- ManagedRecord
- Party

6.2.5 RMS Query Service

In service-oriented architectures, services must efficiently process requests that are represented in XML. These services use a variety of data sources and supporting services to produce an appropriate response to a request. One implementation of SOA, Web services depends heavily on XML processing and data integration. XQuery is specifically designed to abridge the tasks of defining queries and filtering data across a service-oriented architecture using XML.

What SQL has been for the relational databases and client-servers, XQuery is for world of Web services and service-oriented architecture (SOA). It is a general purpose query language designed by the W3C to address the enormous amounts of information stored in XML within an enterprise and across the internet. It provides the ability to :

- Select information on a specified criteria
- Join data across multiple documents

- Perform arithmetic calculations
- Provide filters on returns data
- Search for information within a document

XQuery can be used to access highly structured data, semi-structured data as well as relatively unstructured data. The specification of XQuery is related to XPath. XPath is a XML based language that provides a means to describe a path or location within an XML documents to retrieve data. XPath navigates the specific contents of a document; XQuery provides the means to query across documents with the use of complex string manipulations, time comparisons, mathematical calculations and predicate logic. XQuery borrows many of the ideas from Structured Query Language (SQL). XQuery is often seen as the service frontend to relational database on the backend; many third party relational databases support XQuery.

XQuery follows the FLWOR structure for defining queries. Pronounced “flower” it is an acronym standing for “For”, “Let”, “Where”, “Order by” & “Return”:

- “for” defines a sequence of tuples
- “let” binds the sequence to variables
- “where” provides Boolean filters on the tuples
- “order by” sorts the tuples
- “return” is evaluated once on every tuple filtering what should be returned

Example:

```
doc("ManagedRecord.xml")//[ capturedate >= '2007-01-01' and capturedate <= '2008-12-31']
Will return all ManagedRecords and their elements that have a capturedate between
1/1/2007 and 12/31/2008.
doc("ManagedRecord.xml")//[ capturedate >= '2007-01-01' and capturedate <= '2008-12-
31']/assignment/category/[name = ‘4240 Contracting Officer Appointment’]
Will return all ManagedRecords and their elements that have a capturedate between
1/1/2007 and 12/31/2008 with a category name = ‘4240 Contracting Officer Appointment’
```

Providing data access services within SOA that returns data as XML isolates the enterprise databases and integration layers. It protects the source of the data and ensures that changes to underlying information are shared throughout the enterprise. Data services can be defined at the logical level and implemented with XQuery. Using XQuery for data services simplifies programs, improves productivity and performance.

6.3 Changes to OMG Specifications

6.4 Acknowledgements

7 Introduction to Records Management

The specification describes a set of services that support the basic activities to be applied to a record over its life-cycle from “set aside” to its disposition where disposition is either its destruction or transfer to another legitimate authority. It makes no attempt to define what should or should not be a record. Record determination is based on the rules of the business creating the records. The services are used to establish evidence of the management of records as well as the record itself.

The activity of “set aside” is a term used to describe simply the action of an individual who deems something to be a record. By this decision, the individual confers onto the record requirements for its maintenance and management, and the capture of evidence that these requirements are met. Evidence includes information such as the individual who made this decision, initiation of a way to ensure its authenticity, and its provenance - the organization and the business reasoning for why it is a record. Authentication is the term used to ensure that the record has not been changed in any way and has not been tampered or altered at any point in its life-cycle.

The decision to keep something as a record is typically driven by the business rules associated with the business activity that created it. Further, the record is not generally managed in isolation from other records. The record is usually kept with other records like it, that is, you keep records of travel with other records of travel. This “keeping together” establishes the concept of categorization; a travel record is kept with other travel records in the Travel Records Category. Categories are usually segregated into chronological sets such as a calendar or fiscal year. A business sets this interval at whatever is best for them, usually based on a decision of cost and ease of use; there are exceptions. An external authority such as a court or regulatory agency can direct the chronological interval to manage records and the period of time to hold them before they can be destroyed or transferred. Keeping records in sets allows easy retrieval of a record. You can find a record created in 2006 easier in the set of only the 2006 records rather than finding it in a set that covers ten years.

During a records life-cycle, many things can happen and these activities are captured as evidence of the management of the record. A recordkeeper can change, the method of ensuring authenticity can be updated to a more secure method. The record category can change or the disposition can be changed. Disposition of a record can be suspended by a legitimate authority, that is, a record identified for destruction cannot be destroyed because an order has been received suspending the destruction pending the outcome of some other event, e.g. a court case, a hearing on an issue, administrative hold pending a review of the disposition itself.

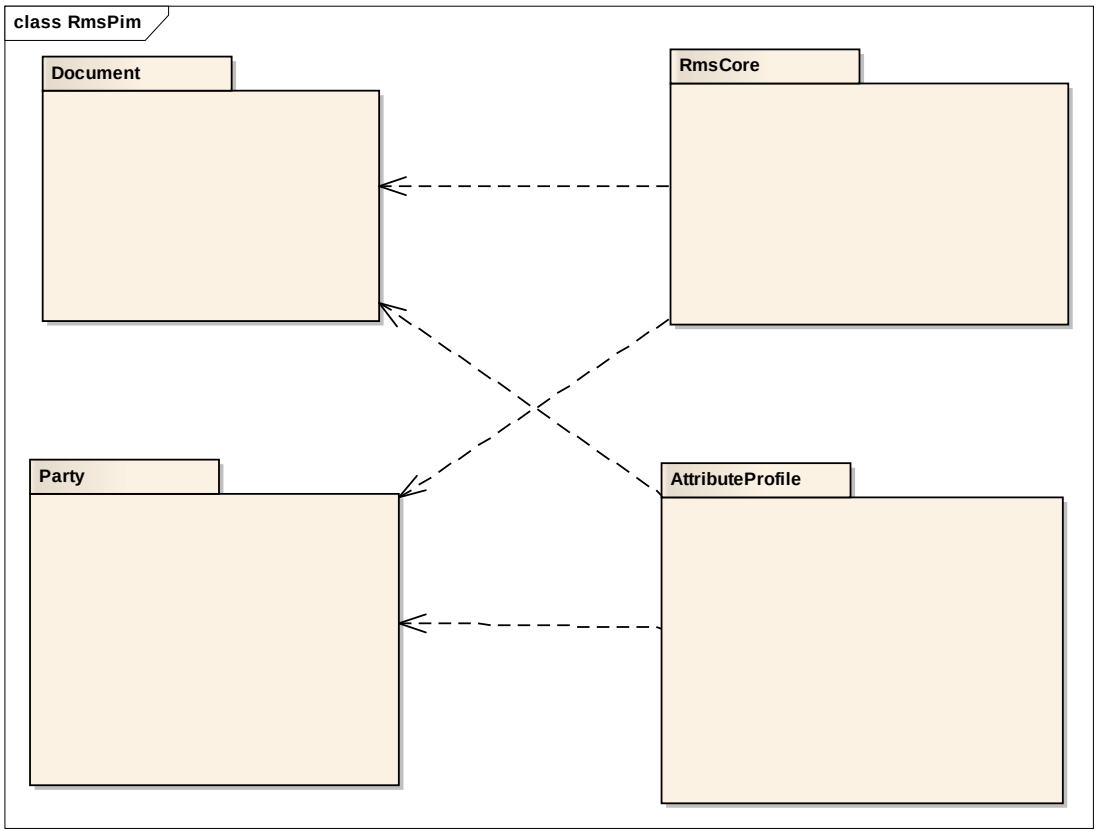
The disposition of a record is the final outcome in a records life-cycle. There are only two possible outcomes for a record; destruction or transfer. Transfer is sending them to an archive or donating them to a library. In either event, the transfer is the passing of legal custody of the record to another entity.

8 Platform Independent Model

The RmsPim package contains the general Platform Independent Model of the RMS Specification. This model is used to capture a structural and behavioral specification in a manner that can be implemented in a variety of technologies.

8.1 Package: Overview

Rms Pim Overview



The Records Management Services PIM Domain Models consists of four packages.

The diagram shows the dependencies among the packages.

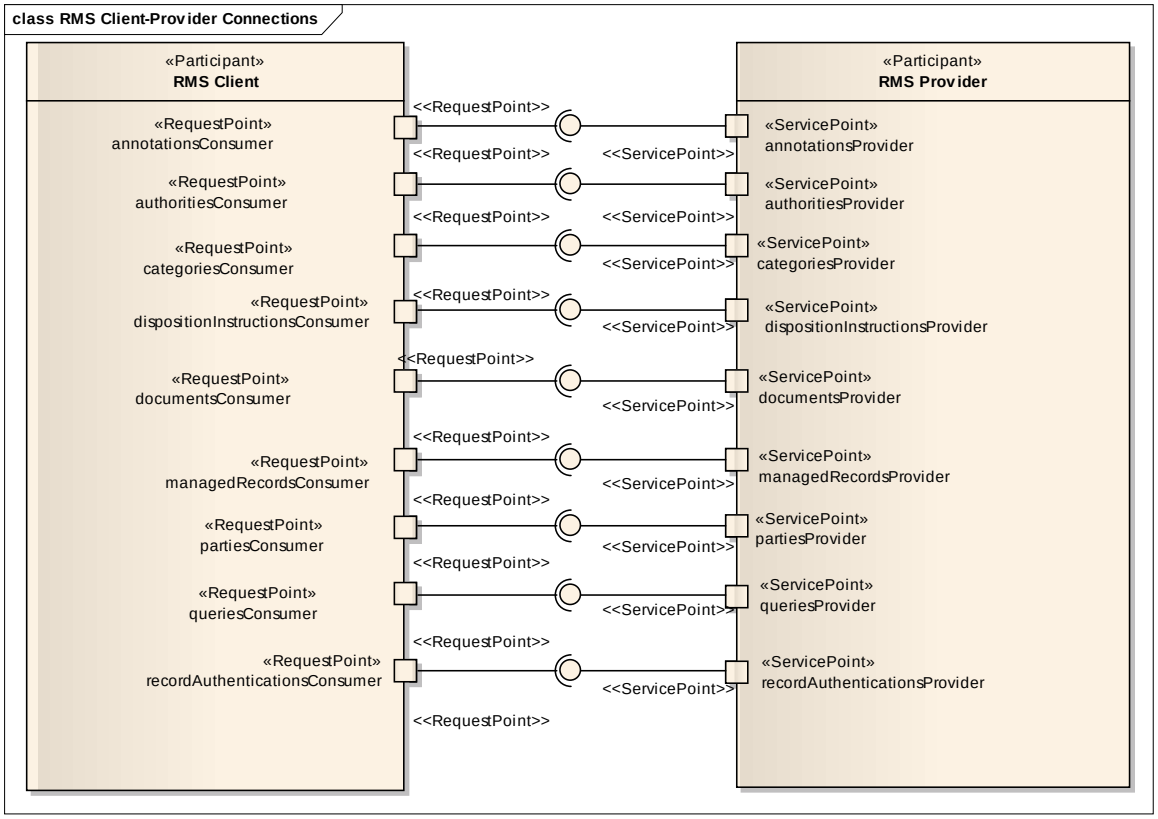
The RmsCore class structure of Records Management based on the work of the Records Management Services Component Interagency Project Team.

The AttributeProfile package provides the capability of specifying attribution by class type for the major RECORDS MANAGEMENT classes. This enables attribution based on the business context of the RECORDS MANAGEMENT environment allowing attribution according to such standards as DoD 5015.2, Dublin Core, etc.

The Document package collects the elements needed to support records that are one or more electronic "bit streams". Each bit stream is represented by a Document.

The Party package collects the elements necessary for modeling organizational structure. In the case of records management, however, the purpose is to capture assignment of responsibility of actions and custodianship, not represent the overall organizational structure.

RMS Client-Provider Connections



8.1.1 Participant: Overview::RMS Client

The RMS Client represents any hardware/software system that uses the service from an RMS Provider.

Attributes

Connections

8.1.2 Participant: Overview::RMS Provider

An RMS Provider is any hardware/software system that supports the records management services as specified in this model.

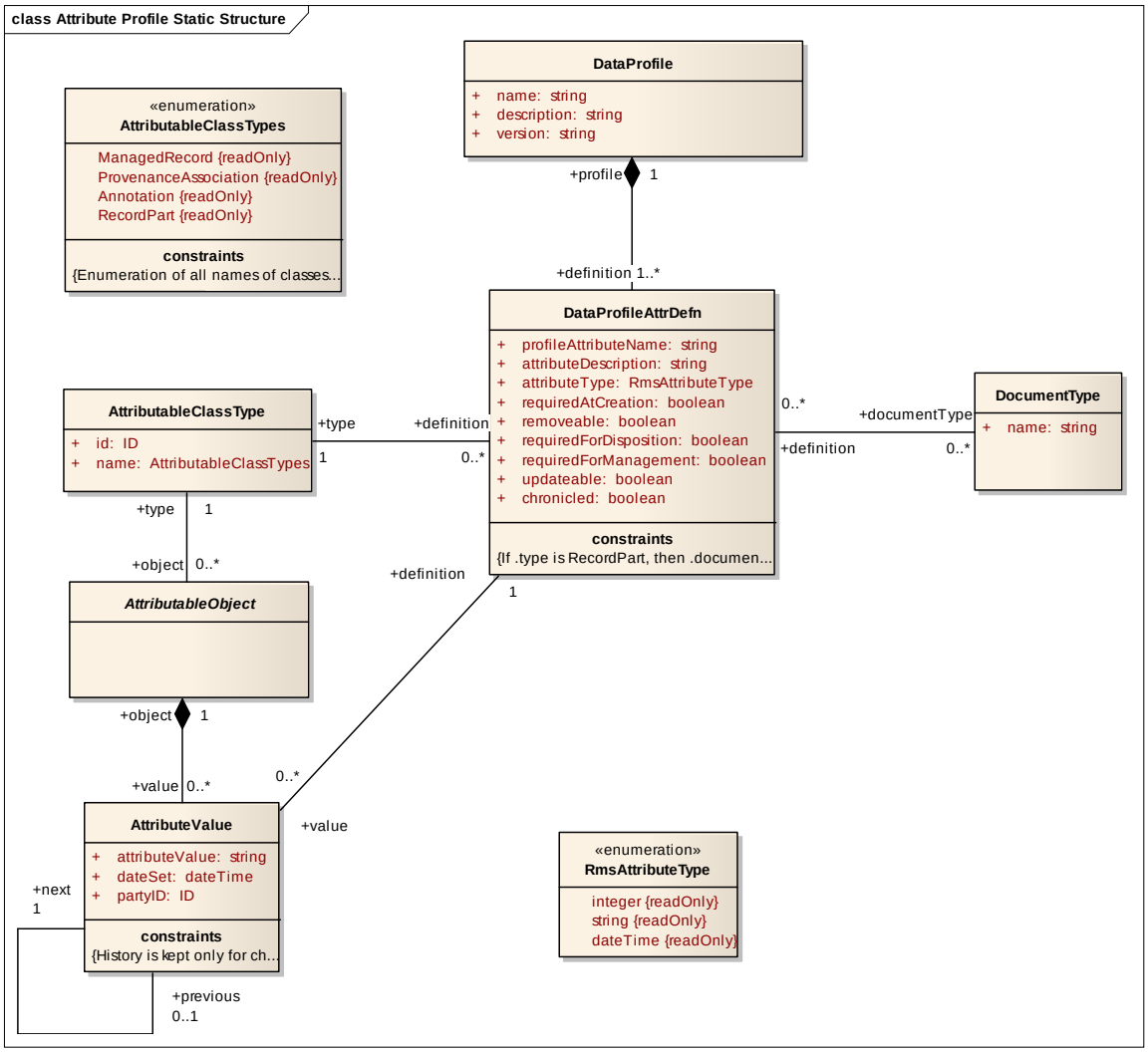
Attributes

Connections

8.2 Package: AttributeProfile

The AttributeProfile package provides the capability of specifying attribution by class type for the major RECORDS MANAGEMENT classes. This enables attribution based on the business context of the RECORDS MANAGEMENT environment allowing attribution according to such standards as DoD 5015.2, Dublin Core, etc.

Attribute Profile Static Structure



The AttributeProfile provides the capability of specifying attribution by class type for the major RECORDS MANAGEMENT classes. This enables attribution based on the business context of the RECORDS MANAGEMENT environment allowing attribution according to such standards as DoD 5015.2, Dublin Core, etc.

A DataProfile is a named collection of attribute definitions. Specific constraints on the behavior of the attributes in a records management environment is specified through the

DataProfileAttrDefn. The type of data represented by the AttributeValue.attributeValue is specified here as well.

The types of objects that can be specified for attribution in the RMS are enumerated in AttributableClassTypes

If the object specified by a DataProfileAttrDefn is a RecordPart, it must specify oneor more DocumentType's to which the attribute applies

8.2.1 Class: AttributeProfile::AttributableClassType

The specialization class type of the Attributable object which aggregates the set of attributes as defined by the DataProfileAttrDefn's that can be assigned as AttributeValue's to an AttributableObject.

Attributes

Attribute: AttributableClassType.id

Type: ID
Description: Unique Identifier

Attribute: AttributableClassType.name

Type: AttributableClassTypes
Description: The name of the records management domain classes whose objects can be assigned AttributeValue's if defined in a DataProfile.

Connections

Association

Indicates the object type to which the DataProfileAttrDefn applies.

From Class: AttributeProfile::DataProfileAttrDefn
In the Role of: definition
Multiplicity: 0..*
Description: The definition of an attribute that applies to the AttibutableClassType

To Class: AttributeProfile::AttributableClassType
In the Role of: type
Multiplicity: 1
Description: The class of object to which the DataProfileAttrDefn applies

Association

The classtype of the AttributableObject. The type determined by reflection must match that in RMSAttributableClassTypes. (Supporting both reflective and non-reflective languages).

From Class:	AttributeProfile::AttributableObject
In the Role of:	object
Multiplicity:	0..*
Description:	The object that is attributed.

To Class:	AttributeProfile::AttributableClassType
In the Role of:	type
Multiplicity:	1
Description:	The type of the object that is attributed.

8.2.2 Enumeration: AttributeProfile::AttributableClassTypes

The classes whose instances can be attributed through an RMSDataProfile, i.e., be assigned AttributeValue's.

Attributes

Attribute: AttributableClassTypes.ManagedRecord

Type:	string
Description:	ManagedRecord is attibutable

Attribute: AttributableClassTypes.ProvenanceAssociation

Type:	string
Description:	ProvenanceAssociation is attibutable.

Attribute: AttributableClassTypes.Annotation

Type:	string
Description:	Annotation is attributable.

Attribute: AttributableClassTypes.RecordPart

Type:	string
Description:	RecordPart is attributable.

Connections

Constraint Name: Enumeration of all names of classes that are subtypes of AttributableObject

8.2.3 Class: AttributeProfile::AttributableObject

An object that can be attributed through the AttributeProfile services.

Attributes

Connections

Aggregation

The collection of an objects attributes.

From Class:	AttributeProfile::AttributeValue
In the Role of:	value
Multiplicity:	0..*
Description:	The value of an object attribute.

To Class:	AttributeProfile::AttributableObject
In the Role of:	object
Multiplicity:	1
Description:	The attributed object.

Association

The classtype of the AttributableObject. The type determined by reflection must match that in RMSAttributableClassTypes. (Supporting both reflective and non-reflective languages).

From Class:	AttributeProfile::AttributableObject
In the Role of:	object
Multiplicity:	0..*
Description:	The object that is attributed.

To Class:	AttributeProfile::AttributableClassType
In the Role of:	type
Multiplicity:	1
Description:	The type of the object that is attributed.

Generalization

From Class:	ManagedRecord::ManagedRecord

To Class:	AttributeProfile::AttributableObject

Generalization

From AssociationClass: ManagedRecord::ProvenanceAssociation

To Class:	AttributeProfile::AttributableObject

Generalization

From Class:	Annotation::Annotation

To Class:	AttributeProfile::AttributableObject

Generalization

From Class: ManagedRecord::RecordPart

To Class: AttributeProfile::AttributableObject

8.2.4 Class: AttributeProfile::AttributeValue

A value of an attribute associated with an AttributableObject.

Attributes

Attribute: AttributeValue.attributeValue
Type: string
Description: The string representing the value of the attribute of the AttributableObject

Attribute: AttributeValue.dateSet
Type: dateTime
Description: The date/time that the value of the AttributableObject was set.

Attribute: AttributeValue.partyID
Type: ID
Description: The ID of the party that set the attribute value.

Connections

Constraint Name: History is kept only for chronicled attributes. Those whose RMSDataProfileAttribute.chronicled = True

Aggregation

The collection of an objects attributes.

From Class: AttributeProfile::AttributeValue
In the Role of: value
Multiplicity: 0..*
Description: The value of an object attribute.

To Class: AttributeProfile::AttributableObject
In the Role of: object
Multiplicity: 1
Description: The attributed object.

Association

The RMS attribute definition on which the AttributeValue is based.

From Class:	AttributeProfile::AttributeValue
In the Role of:	value
Multiplicity:	0..*
Description:	The AttributeValue which is based on the RMSAttributeDefn

To Class:	AttributeProfile::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	1
Description:	The RMSAttributeDefn on which the AttributeValue is based.

Association

The history of the attribute's value.

From Class:	AttributeProfile::AttributeValue
In the Role of:	next
Multiplicity:	1
Description:	The superceding AttributeValue for a chronicled attribute.

To Class:	AttributeProfile::AttributeValue
In the Role of:	previous
Multiplicity:	0..1
Description:	The superceded AttributeValue for a chronicled attribute.

8.2.5 Class: AttributeProfile::DataProfile

A profile of attribute definitions that may apply to AttributableObject's under organizational, ad hoc, or de jure standards or conventions.

Attributes

Attribute: DataProfile.name

Type:	string
Description:	The unique name of the data profile.

Attribute: DataProfile.description

Type:	string
Description:	Textual description of the DataProfile

Attribute: DataProfile.version

Type:	string
Description:	The version of the DataProfile

Connections

Aggregation

Collects the DataProfileAttrDefn's that apply to this DataProfile.

From Class:	AttributeProfile::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	1..*
Description:	The definition of an attribute that is a member of the DataProfile

To Class:	AttributeProfile::DataProfile
In the Role of:	profile
Multiplicity:	1
Description:	The DataProfile of which the DataProfileAttrDefn is a member.

8.2.6 Class: AttributeProfile::DataProfileAttrDefn

A member of a DataProfile that describes an attribute that is applicable to a specific AttributableClassType.

If the object specified by a DataProfileAttrDefn is a RecordPart, it must specify oneor more DocumentType's to which the attribute applies.

Attributes

Attribute: DataProfileAttrDefn.profileAttributeName

Type:	string
Description:	The name of the profile, unique in the context of its DataProfile. There may be multiple DataProfileAttrDefn's with the same name if they are in different DataProfile's

Attribute: DataProfileAttrDefn.attributeDescription

Type:	string
Description:	Textual description of the attribute.

Attribute: DataProfileAttrDefn.attributeType

Type:	RmsAttributeType
Description:	The AttributableClassType to which the DataProfileAttrDefn applies.

Attribute: DataProfileAttrDefn.requiredAtCreation

Type:	boolean
Description:	If "True", the object must be provided an AttributeValue conformant to this definition at time of creation.

Attribute: DataProfileAttrDefn.removeable

Type:	boolean

Description: If "True", the object's AttributeValue conformant to this definition may be removed (deleted).

Attribute: DataProfileAttrDefn.requiredForDisposition

Type: boolean

Description: If "True", the object's AttributeValue conformant to this definition must be present before final disposition (Transfer or Destroy) can be performed on the ManagedRecord associated with an object with this AttributeValue.

Attribute: DataProfileAttrDefn.requiredForManagement

Type: boolean

Description: If "True", the object's AttributeValue conformant to this definition must be present for management of the ManagedRecord associated with an object with this AttributeValue.

Attribute: DataProfileAttrDefn.updateable

Type: boolean

Description: If "True", the object's AttributeValue conformant to this definition may be updated. In the case that the value is not chronicled, the .attributeValue may be changed with new .dateSet, and .party. In the case that the value is chronicled, a new AttributeValue of the same RMSAttributeDefn is created and linked to the previous one through the next/previous association.

Attribute: DataProfileAttrDefn.chronicled

Type: boolean

Description: When .chronicled and .updateable = "True" , a new AttributeValue of the same RMSAttributeDefn may be created and linked to the previous one through the next/previous association.

Connections

Constraint Name: If .type is RecordPart, then .documentType must point to one or more DocumentType's

Association

The RMS attribute definition on which the AttributeValue is based.

From Class: AttributeProfile::AttributeValue

In the Role of: value

Multiplicity: 0..*

Description:	The AttributeValue which is based on the RMSAttributeDefn
To Class:	AttributeProfile::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	1
Description:	The RMSAttributeDefn on which the AttributeValue is based.

Association

Indicates the object type to which the DataProfileAttrDefn applies.

From Class:	AttributeProfile::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	0..*
Description:	The definition of an attribute that applies to the AttibutableClassType
To Class:	AttributeProfile::AttributableClassType
In the Role of:	type
Multiplicity:	1
Description:	The class of object to which the DataProfileAttrDefn applies

Aggregation

Collects the DataProfileAttrDefn's that apply to this DataProfile.

From Class:	AttributeProfile::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	1..*
Description:	The definition of an attribute that is a member of the DataProfile
To Class:	AttributeProfile::DataProfile
In the Role of:	profile
Multiplicity:	1
Description:	The DataProfile of which the DataProfileAttrDefn is a member.

Association

If the DataProfileAttrDefn pertains to a RecordPart, then one or more DocumentType's are specified. If the RecordPart has a Document of one of those DocumentType's then the DataProfileAttrDefn applies to that RecordPart.

From Class:	AttributeProfile::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	0..*
Description:	The attribute definition that applies to RecordParts of this DocumentType.

To Class:	AttributeProfile::DocumentType
In the Role of:	documentType
Multiplicity:	0..*
Description:	The DocumentType's that makes the DataProfileAttrDefn eligible for use on a RecordPart.

8.2.7 Class: AttributeProfile::DocumentType

Placeholder for the Document Type to which the profile applies.

Attributes

Attribute: DocumentType.name

Type:	string
Description:	The name of the DocumentType

Connections

Association

If the DataProfileAttrDefn pertains to a RecordPart, then one or more DocumentType's are specified. If the RecordPart has a Document of one of those DocumentType's then the DataProfileAttrDefn applies to that RecordPart.

From Class:	AttributeProfile::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	0..*
Description:	The attribute definition that applies to RecordParts of this DocumentType.

To Class:	AttributeProfile::DocumentType
In the Role of:	documentType
Multiplicity:	0..*
Description:	The DocumentType's that makes the DataProfileAttrDefn eligible for use on a RecordPart.

8.2.8 Enumeration: AttributeProfile::RmsAttributeType

The type of the attribute.

Attributes

Attribute: RmsAttributeType.integer
Type: string

Attribute: RmsAttributeType.string
Type: string

Attribute: RmsAttributeType.dateTime
Type: string

Connections

8.2.9 Class: AttributeProfile::RmsProfiledItems

Attributes

Connections

8.2.10 Class: AttributeProfile::TimeDelta

See the BaseTypes package of the RmsDomainModel for a definition of this element.

Attributes

Connections

8.2.11 Class: AttributeProfile::Id

See the BaseTypes package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Id.id
Type: string
Description: See the BaseTypes package of the RmsDomainModel for a definition of this element.

Connections

8.2.12 Class: AttributeProfile::TimeStamp

See the BaseTypes package of the RmsDomainModel for a definition of this element.

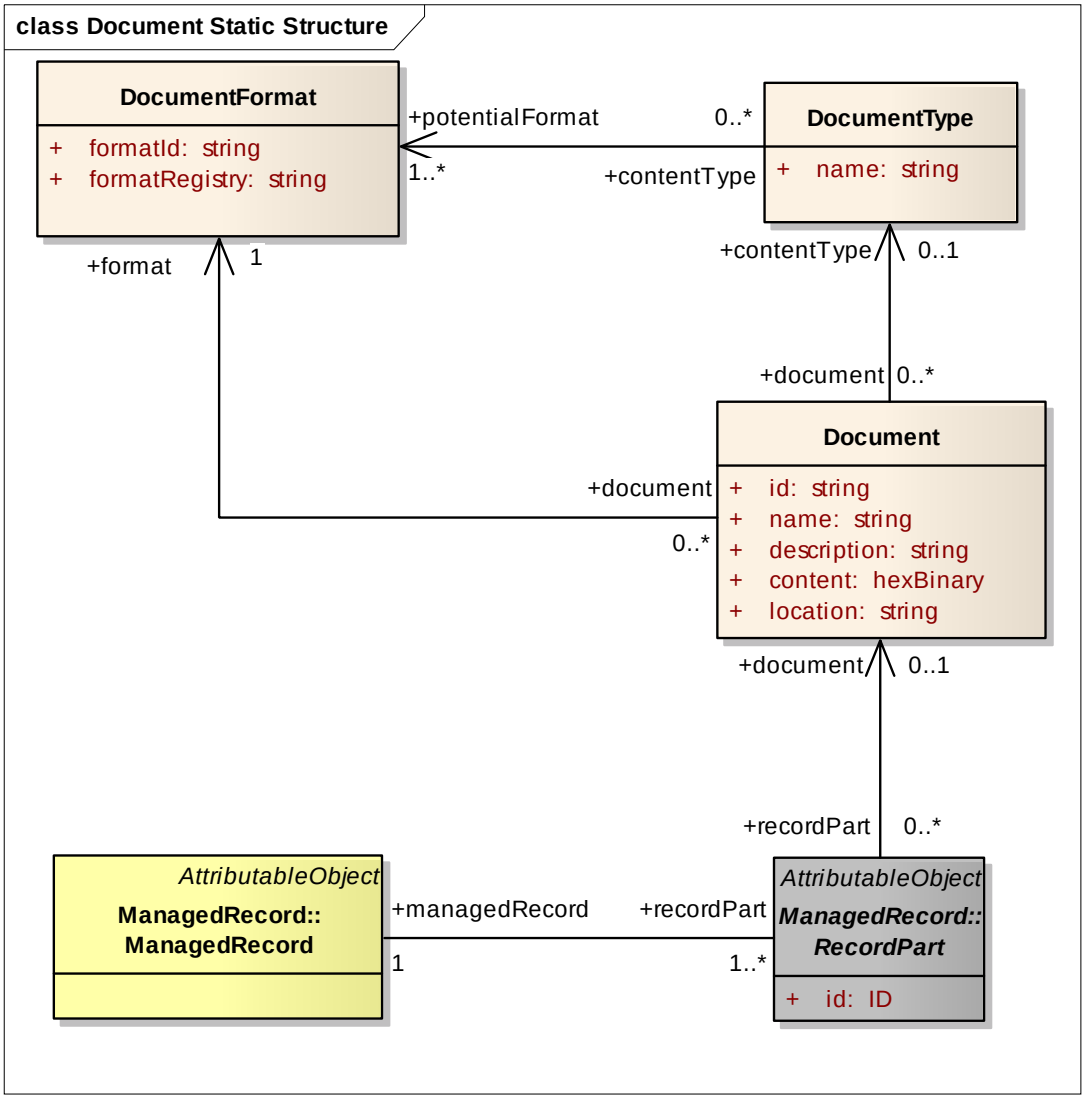
Attributes

Connections

8.3 Package: Document

The Document package collects the elements needed to support records that are one or more electronic "bit streams". Each bit stream is represented by a Document.

Document Static Structure



Models the concepts of "Documents" that are managed as a ManagedRecord. A Document in the Records Management Services is interpreted simply as "bit strings" without presumption of form or purpose.

Documents can be used in many ManagedRecords because a single Document can represent evidence of multiple business activities/purposes. When final disposition of one ManagedRecord in which it participates occurs (transfer or destroy), the ManagedRecord is destroyed. The Document itself is destroyed only when the ManagedRecord in final disposition is the only one that still refers to it.

8.3.1 Class: Document::DocumentFormat

Represents the format of a Document. It can be something as simple as "mime types" or the specification of a format documented in a formal format registry.

Attributes

Attribute: DocumentFormat.formatId

Type: string
Description: The identifier of the format within the specified format registry. For example ".doc" if the registry is that of W3C mime types. This is not the usual ".id" found commonly in this specification. This is the "stringified" (if necessary) unique id in the context of the .formatRegistry.

Attribute: DocumentFormat.formatRegistry

Type: string
Description: The specification of the data format registry. For example, this can be a URI or URL unambiguously specifying the registry being referenced.

Connections

Association

Associates a Document with its DocumentFormat

From Class: Document::Document
In the Role of: document
Multiplicity: 0..*
Description: The document

To Class: Document::DocumentFormat
In the Role of: format
Multiplicity: 1
Description: The documents format.

Association

Associates the DocumentFormat's that can be used in the representation of a DocumentType

From Class: Document::DocumentType
In the Role of: contentType
Multiplicity: 0..*
Description: A contentType to be mapped to possible formats.

To Class: Document::DocumentFormat

In the Role of: potentialFormat
Multiplicity: 1..*
Description: DocumentFormats that can be used in the representation of the DocumentType

8.3.2 Class: Document::DocumentType

The document type is a designation defined for the convenience of the organization. The string can be used to define any concept concerning the document that serves the organization. It has a 1..* cardinality with DocumentFormat, e.g., the DocumentType might be "Building Layout" & the possible formats may be .gif, .jpeg, etc.)

Attributes

Attribute: DocumentType.name

Type: string
Description: The name of the DocumentType.

Connections

Association

Associates the DocumentFormat's that can be used in the representation of a DocumentType

From Class: Document::DocumentType
In the Role of: contentType
Multiplicity: 0..*
Description: A contentType to be mapped to possible formats.

To Class: Document::DocumentFormat
In the Role of: potentialFormat
Multiplicity: 1..*
Description: DocumentFormats that can be used in the representation of the DocumentType

Association

Associates a document with its content type.

From Class: Document::Document
In the Role of: document
Multiplicity: 0..*
Description: The document

To Class: Document::DocumentType
In the Role of: contentType
Multiplicity: 0..1

Description: The content type of the Document

8.3.3 Class: Document::Document

A bit stream being managed as a ManagedRecord, or part of a ManagedRecord.

Attributes

Attribute: Document.id
Type: string
Description: Unique Identifier

Attribute: Document.name
Type: string
Description: The name of the document

Attribute: Document.description
Type: string
Description: Text description of the document

Attribute: Document.content
Type: hexBinary
Description: The binary element that is the document.

Attribute: Document.location
Type: string
Description: The logical location of the document. The RMS services do not address physical location.

Connections

Association

Associates a Document with its DocumentFormat

From Class: Document::Document
In the Role of: document
Multiplicity: 0..*
Description: The document

To Class: Document::DocumentFormat
In the Role of: format
Multiplicity: 1
Description: The documents format.

Association

Associates a document with its content type.

From Class:	Document::Document
In the Role of:	document
Multiplicity:	0..*
Description:	The document

To Class:	Document::DocumentType
In the Role of:	contentType
Multiplicity:	0..1
Description:	The content type of the Document

Association

Associates the ManagedRecord with the Documents that comprise it.

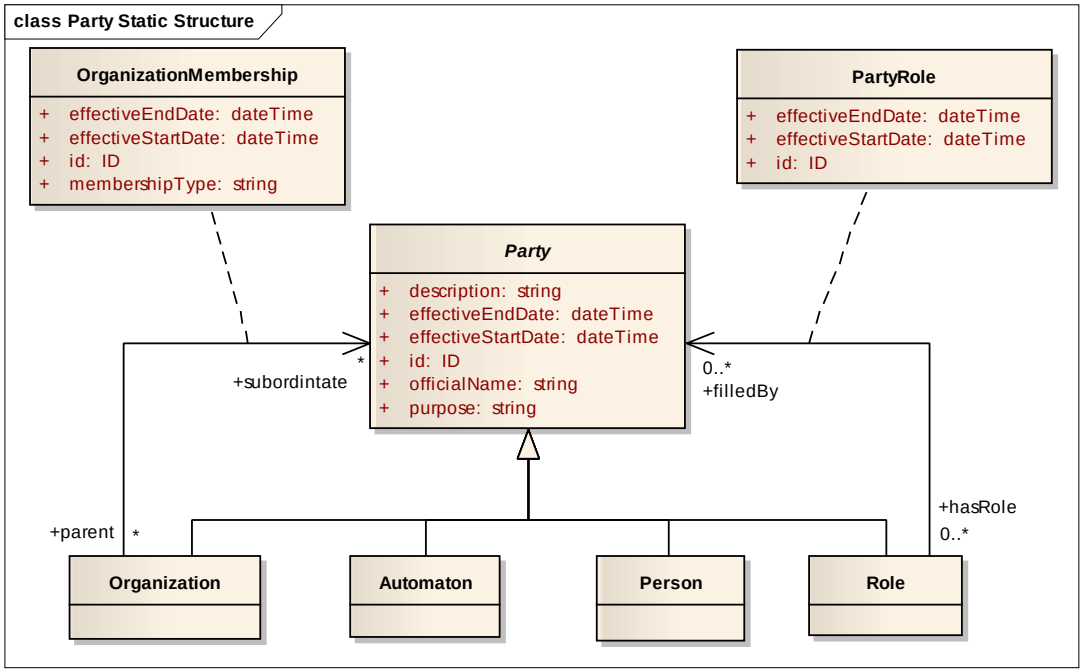
From Class:	ManagedRecord::RecordPart
In the Role of:	recordPart
Multiplicity:	0..*
Description:	The part membership of a document in a ManagedRecord.

To Class:	Document::Document
In the Role of:	document
Multiplicity:	0..1
Description:	The document that is part of the ManagedRecord

8.4 Package: Party

The Party package collects the elements necessary for modeling organizational structure. In the case of records management, however, the purpose is to capture assignment of responsibility of actions and custodianship, not represent the overall organizational structure.

Party Static Structure



The Party Model is related to the organizational structure of the organization in which records are being managed, but is not identical to it. The purpose of the model is to be able to express Provenance and to identify the Roles in the organization that attribute aspects of the Managed Record.

Provenance is minimally specified by the top-level organization authorized as a valid organization for Provenance. If there is an organization that contains it, it is not instantiated in this model.

Provenance can be expressed in greater detail through a chain of suborganizations. However, at the discretion of the organization, some suborganizations are not expressed in the Provenance chain, in which case the Party's belonging to it are "collapsed" into its containing suborganization. These "suppressed" organizations are not instantiated in this model.

Though related to the overall organizational structure, there is no requirement for it to be wholly consistent. The intent is to provide the designation of Provenance consistent with the business practices of the organization.

The organization structure can be determined for any point in time using the .effectiveStartDate and .effectiveEndDate on the Party, OrganizationMembership, and PartyRole classes.

8.4.1 Class: Party::Automaton

Some automated system that may serve in the generation or set-aside of records.

Attributes

Connections

Generalization

From Class: Party::Automaton

To Class: Party::Party

8.4.2 AssociationClass: Party::OrganizationMembership

Used to indicate the membership of a Party in an organization and the type of that membership. .

Attributes

Attribute: OrganizationMembership.id
Type: ID
Description: Unique identifier

Attribute: OrganizationMembership.membershipType
Type: string
Description: The type of membership in an Organization. This can be anything that serves the business case of the organization owning the Records Management Environment.

Attribute: OrganizationMembership.effectiveStartDate
Type: dateTime
Description: The effective start date of the membership.

Attribute: OrganizationMembership.effectiveEndDate
Type: dateTime
Description: The effectiveEndDate of the membership.

Connections

8.4.3 AssociationClass: Party::PartyRole

PartyRole indicates the Party that fills a particular Role, if known.

Attributes

Attribute: PartyRole.id
Type: ID
Description: Unique identifier.

Attribute: PartyRole.effectiveStartDate

Type: dateTime
Description: The effective date/time that the Role was filled by the Party.

Attribute: PartyRole.effectiveEndDate

Type: dateTime
Description: The effective date/time that the Party was removed from the Role.

Connections

Association

The Authority for the CaseFileRecordDefinition

From Class: ManagedRecord::CaseFileRecordDefinition
In the Role of: definition
Multiplicity: 0..*
Description: The authorized CaseFileRecordDefinition.

To Class: Party::Authority
In the Role of: authority
Multiplicity: 1
Description: The Authority authorizing the CaseFileRecordDefinition.

8.4.4 In the Role of:In the Role of:In the Role of:Class: Party::Organization

Organization is used to represent the hierarchy of departments and the participants in each organization (Automaton's, Person's, Role's). It is also used to indicate external organizations which are used as Authority's etc. to document aspects of the ManagedRecord.

Attributes

Connections

Association

Associates a Transfer action with the destination Organization.

From Class: Dispositions::Transfer
In the Role of: action
Multiplicity: 0..1
Description: The transfer action.

To Class: Party::Organization
In the Role of: destination
Multiplicity: 0..1

Description: The destination Organization of the Transfer action.

Association

Associates the Move action with its logical destination Organization.

From Class: Dispositions::Move
In the Role of: move
Multiplicity: 0..1
Description: The Move action requiring the move to the Organization

To Class: Party::Organization
In the Role of: destination
Multiplicity: 1
Description: The logical destination (Organization) of the Move action.

AssociationClass

A subtype of Party representing an organizational unit for the purposes of assigning Provenance. It does not necessarily correspond with the "customary" concept of organization structure.

From Class: Party::Organization
In the Role of: parent
Multiplicity: *
Description: A containing organization.

To Class: Party::Party
In the Role of: subordintate
Multiplicity: *
Description: A Party contained in the organization.

Generalization

From Class: Party::Organization

To Class: Party::Party

8.4.5 Class: Party::Party

Party is the abstract supertype of all participants in the organization.

Attributes

Attribute: Party.id
Type: ID
Description: Unique identifier.

Attribute: Party.officialName

Type: string
Description: The official name of the Party.

Attribute: Party.purpose

Type: string
Description: The purpose of the Party in the Records Management Environment.

Attribute: Party.description

Type: string
Description: A textual description of the Party

Attribute: Party.effectiveEndDate

Type: dateTime
Description: The date/time of the end of the Party's participation in the Records Management Environment.

Attribute: Party.effectiveStartDate

Type: dateTime
Description: The date/time of the start of the Party's participation in the Records Management Environment.

Connections

AssociationClass

Associates a Role with the Party that fills it.

From Class: Party::Role
In the Role of: hasRole
Multiplicity: 0..*
Description: The roles held by a Party.

To Class: Party::Party
In the Role of: filledBy
Multiplicity: 0..*
Description: The Partys fillling a Role.

AssociationClass

A subtype of Party representing an organizational unit for the purposes of assigning Provenance. It does not necessarily correspond with the "customary" concept of organization structure.

From Class: Party::Organization
In the Role of: parent
Multiplicity: *

Description: A containing organization.

To Class: Party::Party
In the Role of: subordintate
Multiplicity: *
Description: A Party contained in the organization.

Generalization

From Class: Party::Automaton

To Class: Party::Party

Association

Associates the Annotation with the Party that created it.

From Class: Annotation::Annotation
In the Role of: annotation
Multiplicity: 0..*
Description: The created annotation.

To Class: Party::Party
In the Role of: creator
Multiplicity: 1
Description: The creator of the annotation

Generalization

From Class: Party::Organization

To Class: Party::Party

Association

Documents the Party that annotated the ManagedRecord

From Class: Annotation::ManagedRecordAnnotation
In the Role of: managedRecordAnnotation
Multiplicity: 0..*
Description: The association of an Annotation with its ManagedRecord as created by the Party

To Class: Party::Party
In the Role of: annotator
Multiplicity: 1
Description: The Party which created tha association between the ManagedRecord and its Annotation.

Generalization

From Class: Party::Role

To Class: Party::Party

Generalization

From Class: Party::Person

To Class: Party::Party

8.4.6 Class: Party::Person

Your typical or non-typical homo-sapiens.

Attributes

Connections

Generalization

From Class: Party::Person

To Class: Party::Party

8.4.7 Class: Party::Role

A role in the organization which can be filled by some Party.

Attributes

Connections

Association

Creator of the CategorizationSchema

From Class: Category::CategorizationSchema
In the Role of: schema
Multiplicity: 0..*
Description: The created CategorizationSchema.

To Class: Party::Role
In the Role of: creator
Multiplicity: 1
Description: The creator of the CategorizationSchema

Association

Documents the Role that performs an Activity

From Class:	Category::Activity
In the Role of:	performs
Multiplicity:	0..*
Description:	Activity performed by a Role

To Class:	Party::Role
In the Role of:	performedBy
Multiplicity:	1
Description:	Role that performs an Activity

Association

Creator of the CaseFileRecordDefinition

From Class:	ManagedRecord::CaseFileRecordDefinition
In the Role of:	definition
Multiplicity:	1
Description:	The definition of a ManagedRecord that is a case file.

To Class:	Party::Role
In the Role of:	creator
Multiplicity:	0..*
Description:	The creator of the CaseFileRecordDefinition

Association

The actor that performed the recorded action

From Class:	ManagedRecord::CaseFileAction
In the Role of:	action
Multiplicity:	0..*
Description:	The action taken by the actor on the case file.

To Class:	Party::Role
In the Role of:	actor
Multiplicity:	1
Description:	The actor performing the action on the case file.

AssociationClass

Records the current and historical provenance of the ManagedRecord.

From Class:	Party::Role
In the Role of:	assignedProvenance

Multiplicity: 1..*
Description: The Role to whom or which the Provenance of the ManagedRecord is assigned in this ProvenanceAssociation.

To Class: ManagedRecord::ManagedRecord
In the Role of: recordWithProvenance
Multiplicity: 0..*
Description: The provenance of the ManagedRecord as recorded in this ProvenanceAssociation.

Generalization

From Class: RmsRoles::Authority

To Class: Party::Role

AssociationClass

Associates a Role with the Party that fills it.

From Class: Party::Role
In the Role of: hasRole
Multiplicity: 0..*
Description: The roles held by a Party.

To Class: Party::Party
In the Role of: filledBy
Multiplicity: 0..*
Description: The Partys fillling a Role.

Generalization

From Class: Party::Role

To Class: Party::Party

Generalization

From Class: RmsRoles::RecordKeeper

To Class: Party::Role

Generalization

From Class: RmsRoles::RecordCreator

To Class: Party::Role

Generalization

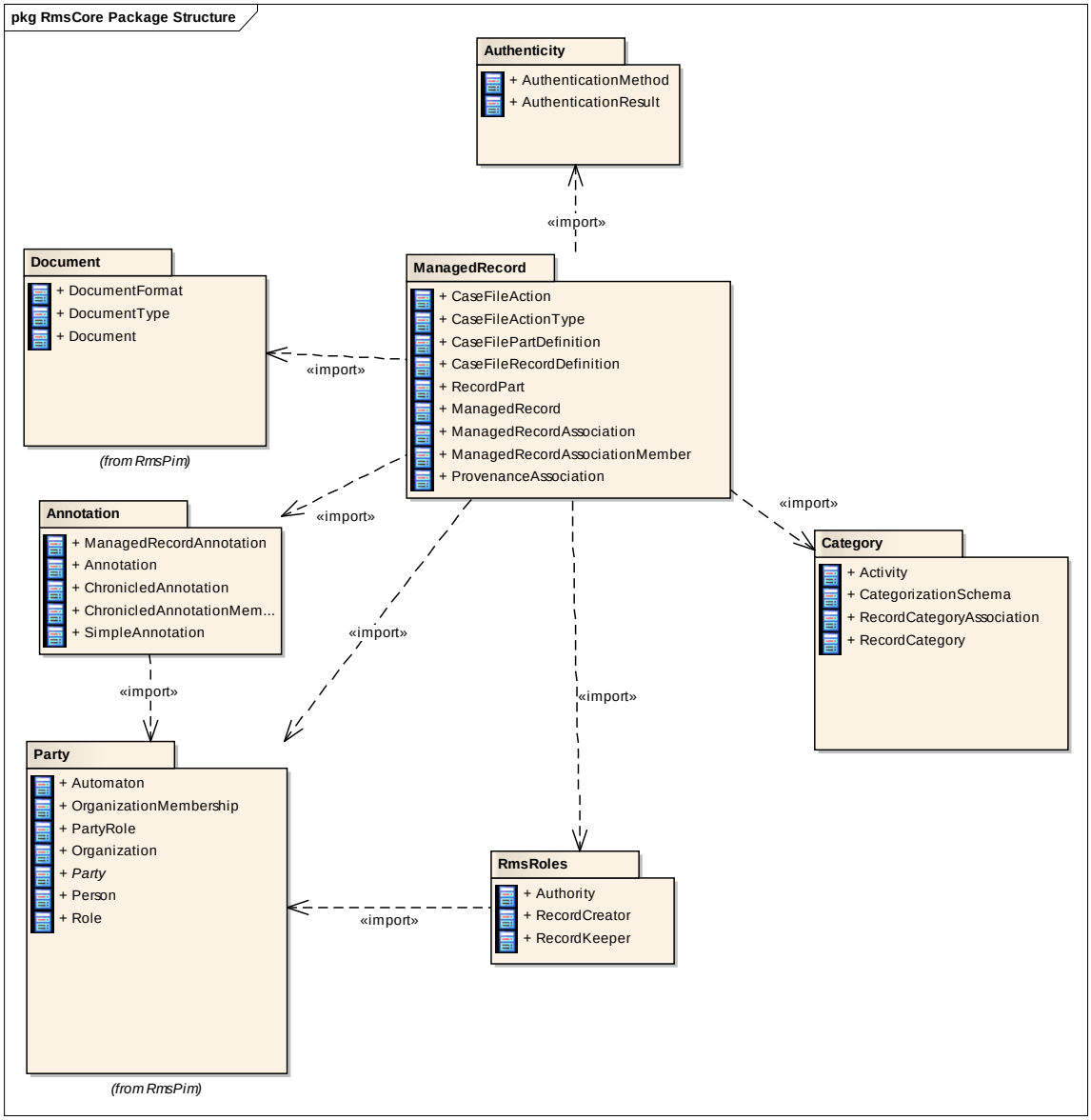
From Class: RmsRoles::DispositionInstructionCreator

To Class: Party::Role

8.5 Package: RmsCore

The core class structure of Records Management based on the work of the Records Management Services Component Interagency Project Team.

RmsCore Package Structure

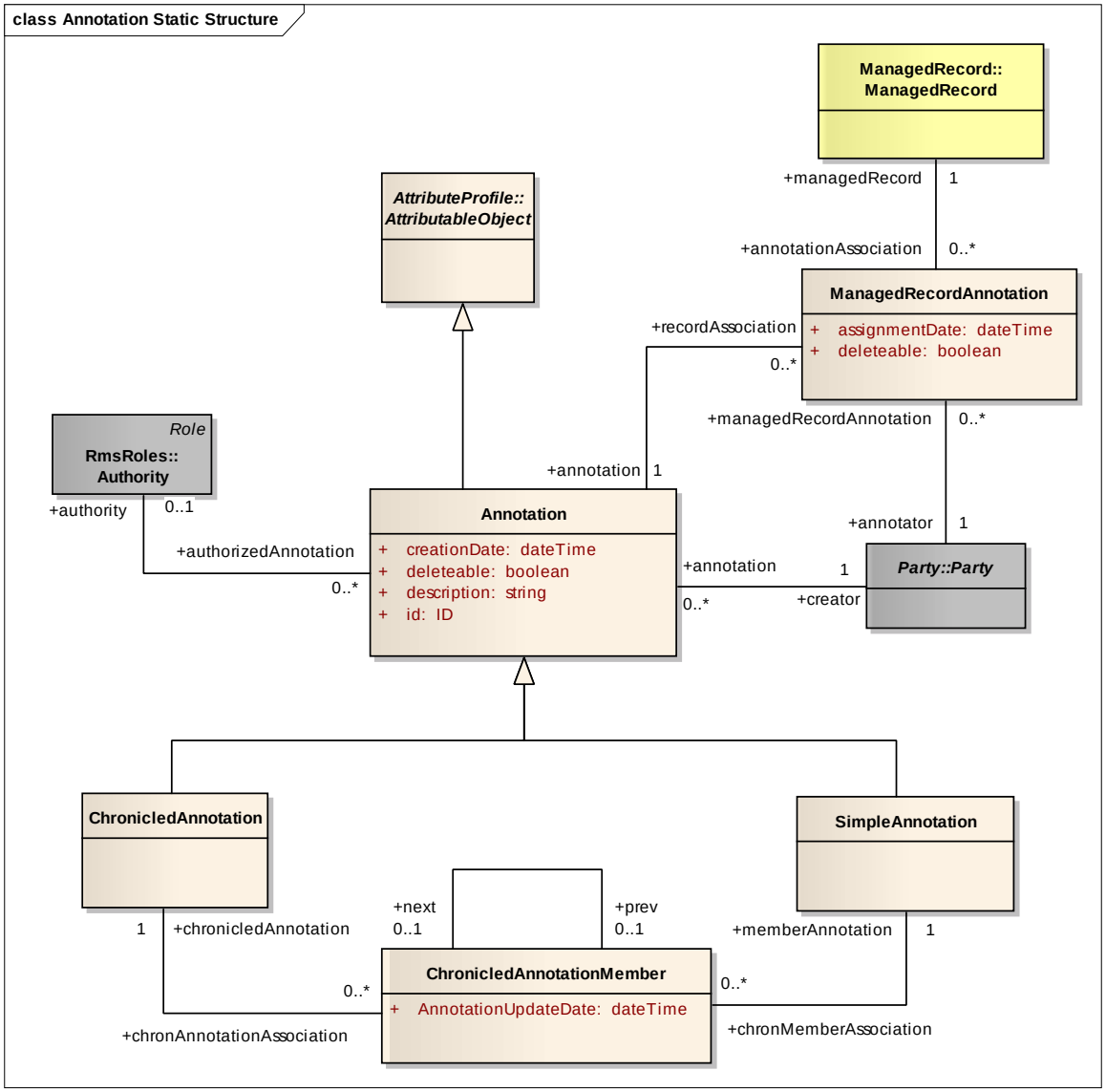


Documents the dependencies among the Domain Model packages.

8.5.1 Package: Annotation

The Annotation package collects the elements needed to support the records management concept of annotated records.

Annotation Static Structure



Annotations are applied to ManagedRecords for any purpose that suits the business needs of an organization.

Annotations can exist independently of ManagedRecords providing a "catalog" of Annotations which can be attached to a ManagedRecord through ManagedRecordAnnotation. The Party making the ManagedRecordAnnotation is mandatory.

Organizations that mark official security designations do so through Annotations. However, the model does not address restrictions and access management requirements for the handling of classified records. A security designation is an example of an Annotation which would be pre-defined for use, and would have an Authority designated as its definer.

When an annotation is pre-defined, it has its "deleteable" flag set and an Authority defined.

Annotations can be informal notes attached for any reason to the ManagedRecord. The annotator may choose to make the annotation "deleteable" or not.

When marked as "deleteable", the Annotation cannot be deleted from the Records Management Environment.

ChronicledAnnotation's may be updated (but its history must be kept). This is done by adding a "new" SimpleAnnotation through a ChronicledAnnotationMember. For example a "Security Classification" could be a ChronicledAnnotation, and its members would reflect the security levels of the ManagedRecord over its life.

An Annotation may be required to be permanent, in which case the "deleteable" flag is "False" on the ManagedRecordAnnotation.

8.5.1.1 Class: Annotation::ManagedRecordAnnotation

An association class that links a ManagedRecord with its Annotations and points to the Party which annotated the ManagedRecord. Annotations can be "pre-defined" and not yet attached to any ManagedRecord. There is no requirement for a ManagedRecord to have an Annotation. Annotations may have an Authority associated with them.

Attributes

Attribute: ManagedRecordAnnotation.assignmentDate

Type: dateTime
Description: The date/time that the Annotation was assigned to the ManagedRecord

Attribute: ManagedRecordAnnotation.deleteable

Type: boolean
Description: When set the flag connotes that the association can be deleted.

Connections

Association

Links a ManagedRecord to its Annotations through ManagedRecordAnnotation.

From Class: ManagedRecord::ManagedRecord
In the Role of: managedRecord
Multiplicity: 1
Description: The annotated ManagedRecord

To Class:	Annotation::ManagedRecordAnnotation
In the Role of:	annotationAssociation
Multiplicity:	0..*
Description:	The associations of the ManagedRecord to its associations with Annotations.

Association

Documents the Party that annotated the ManagedRecord

From Class:	Annotation::ManagedRecordAnnotation
In the Role of:	managedRecordAnnotation
Multiplicity:	0..*
Description:	The association of an Annotation with its ManagedRecord as created by the Party

To Class:	Party::Party
In the Role of:	annotator
Multiplicity:	1
Description:	The Party which created tha association between the ManagedRecord and its Annotation.

Association

Links an Annotation to the ManagedRecords it annotates through ManagedRecordAnnotation

From Class:	Annotation::Annotation
In the Role of:	annotation
Multiplicity:	1
Description:	The Annotation to the ManagedRecord

To Class:	Annotation::ManagedRecordAnnotation
In the Role of:	recordAssociation
Multiplicity:	0..*
Description:	The association of the Annotation with its ManagedRecords

8.5.1.2 Class: Annotation::Annotation

Annotation carries its meaning in its "description" attribute.

Deleteable Annotations have their "deleteable" flag set. SimpleAnnotation which take part in a ChronicledAnnotation must not be deleteable, though the ChronicledAssociation may be, in which case deletion requires the deletion of all its ChronicledAnnotation Members. Whether the particular SimpleAnnotation are deleted depends on the business rules of the organization for that particular Annotation.

Attributes

Attribute: Annotation.id

Type: ID
Description: A unique identifier

Attribute: Annotation.creationDate

Type: dateTime
Description: The date/time that the Annotation was created.

Attribute: Annotation.description

Type: string
Description: A textual description of the meaning of the annotation.

Attribute: Annotation.deleteable

Type: boolean
Description: If set, the Annotation is deleteable.

Connections

Generalization

From Class: Annotation::Annotation

To Class: AttributeProfile::AttributableObject

Association

Associates an Annotation with the Party responsible for authorizing it.

From Class: RmsRoles::Authority
In the Role of: authority
Multiplicity: 0..1
Description: The Authority for the Annotation

To Class: Annotation::Annotation
In the Role of: authorizedAnnotation
Multiplicity: 0..*
Description: The authorized Annotation.

Association

Associates the Annotation with the Party that created it.

From Class: Annotation::Annotation
In the Role of: annotation
Multiplicity: 0..*
Description: The created annotation.

To Class: Party::Party
In the Role of: creator
Multiplicity: 1
Description: The creator of the annotation

Generalization

From Class: Annotation::SimpleAnnotation

To Class: Annotation::Annotation

Generalization

From Class: Annotation::ChronicledAnnotation

To Class: Annotation::Annotation

Association

Links an Annotation to the ManagedRecords it annotates through ManagedRecordAnnotation

From Class: Annotation::Annotation
In the Role of: annotation
Multiplicity: 1
Description: The Annotation to the ManagedRecord

To Class: Annotation::ManagedRecordAnnotation
In the Role of: recordAssociation
Multiplicity: 0..*
Description: The association of the Annotation with its ManagedRecords

8.5.1.3 Class: Annotation::ChronicledAnnotation

An Annotation which may change, but the history of which must be maintained. It is an aggregation of SimpleAnnotation that are time ordered.

Attributes

Connections

Association

Associates a ChronicledAnnotation with each of its SimpleAnnotation (updates) through ChronicledAnnotationMember.

From Class: Annotation::ChronicledAnnotation

In the Role of: chronicledAnnotation
Multiplicity: 1
Description: The ChronicledAnnotation

To Class: Annotation::ChronicledAnnotationMember
In the Role of: chronAnnotationAssociation
Multiplicity: 0..*
Description: The association of the ChronicledNotation with its member NonChronicledAnnotation.

Generalization

From Class: Annotation::ChronicledAnnotation

To Class: Annotation::Annotation

8.5.1.4 Class: Annotation::ChronicledAnnotationMember

Collects the members of a ChronicledAnnotation.

Attributes

Attribute: ChronicledAnnotationMember.AnnotationUpdateDate

Type: dateTime
Description: Date/Time that the NonChroncledAnnotation was added to the ChronicledAnnotation

Connections

Association

Associates a SimpleAnnotation with any ChronicledAnnotation of which it is a member.

From Class: Annotation::ChronicledAnnotationMember
In the Role of: chronMemberAssociation
Multiplicity: 0..*
Description: The association to the ChronicledAssociation

To Class: Annotation::SimpleAnnotation
In the Role of: memberAnnotation
Multiplicity: 1
Description: The NonChronicledAnnotation that participates in one or more ChronicledAnnotations

Association

Implements a time-ordered chain of SimpleAnnotation belonging to the ChronicledAnnotation.

From Class:	Annotation::ChronicledAnnotationMember
In the Role of:	prev
Multiplicity:	0..1
Description:	The previous member

To Class:	Annotation::ChronicledAnnotationMember
In the Role of:	next
Multiplicity:	0..1
Description:	The next member.

Association

Associates a ChronicledAnnotation with each of its SimpleAnnotation (updates) through ChronicledAnnotationMember.

From Class:	Annotation::ChronicledAnnotation
In the Role of:	chronicledAnnotation
Multiplicity:	1
Description:	The ChronicledAnnotation

To Class:	Annotation::ChronicledAnnotationMember
In the Role of:	chronAnnotationAssociation
Multiplicity:	0..*
Description:	The association of the ChronicledNotation with its member NonChronicledAnnotation.

8.5.1.5 Class: Annotation::SimpleAnnotation

An Annotation which is not updateable.

Attributes

Connections

Generalization

From Class:	Annotation::SimpleAnnotation

To Class:	Annotation::Annotation

Association

Associates a SimpleAnnotation with any ChronicledAnnotation of which it is a member.

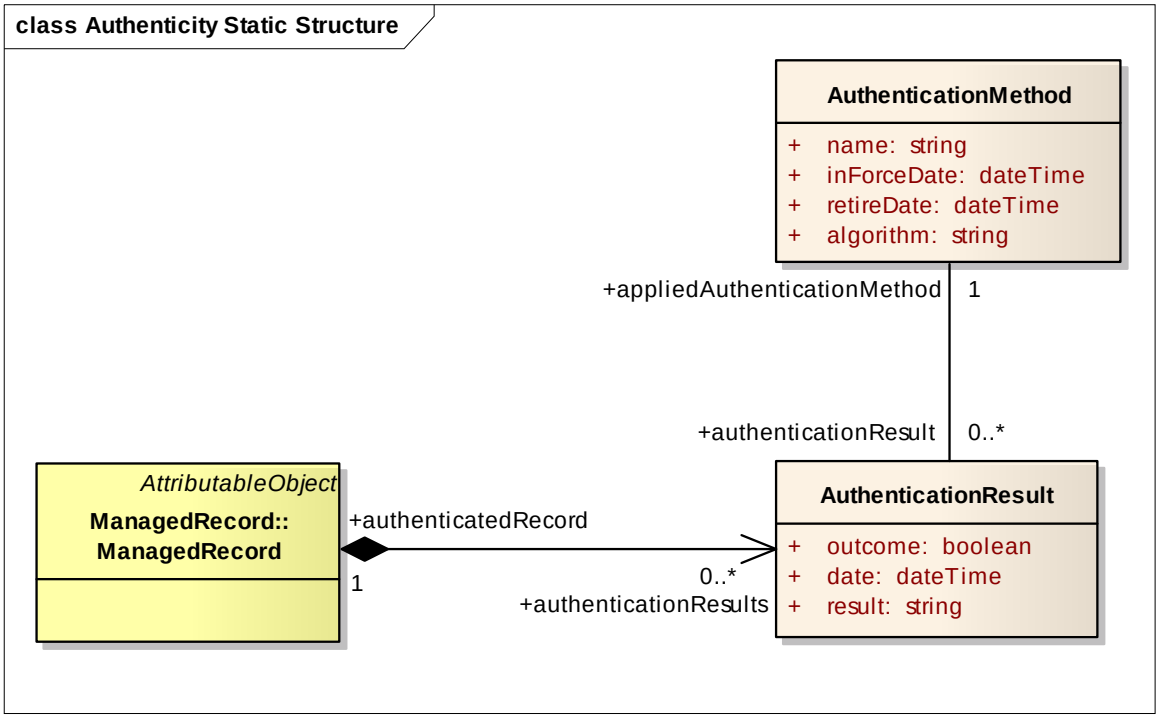
From Class:	Annotation::ChronicledAnnotationMember
In the Role of:	chronMemberAssociation
Multiplicity:	0..*
Description:	The association to the ChronicledAssociation

To Class:	Annotation::SimpleAnnotation
In the Role of:	memberAnnotation
Multiplicity:	1
Description:	The NonChronicledAnnotation that participates in one or more ChronicledAnnotations

8.5.2 Package: Authenticity

The Authenticity package collects the elements needed to support the records management concept of authentic records, i.e., providing assurance that what is retrieved from a record management environment is identical to that which was put there.

Authenticity Static Structure



Authenticity is supported to provide validation that a ManagedRecord is the same thing that was originally entered in the system. This is defined at the service level. No specific algorithm for maintaining authenticity is specified. Algorithms may be simple, or highly complex at the discretion of the organization, based on its business needs.

The AuthenticationMethod represents the algorithm used to calculate an AuthenticationResult. There can only be one AuthenticationMethod "inforce" at one

time. When a new method is put in place, the old one must be "retired" as indicated by its "retireDate", indicating that it is not to be used any longer.

What aspects of a ManagedRecord are included in the calculation is up to the organization. It may be based on content (documents), the metadata as defined by these services, or a subset of the metadata.

To validate authenticity of a ManagedRecord, a new AuthenticationResult is calculated using the algorithm of the AuthenticationMethod that was used to calculate the latest AuthenticationResult. The AuthenticationResult for the first time the method was used is looked up and compared to the new AuthenticationResult. If identical the "outcome" is set to "True", otherwise, it is set to "False"

An AuthenticationMethod cannot be deleted as long as there is any AuthenticationResult associated with it. In the normal course of events, ManagedRecords that are associated with a retired AuthenticationMethod (through its most recent AuthenticationResult" will have their authenticity recalculated through the inForce method. This is done by first validating the ManagedRecord's authenticity under the old method. If successful, the AuthenticationResult calculated with the new inForce AuthenticationMethod is recorded.

8.5.2.1 Class: Authenticity::AuthenticationMethod

The AuthenticationMethod represents the algorithm used to calculate an AuthenticationResult. There can only be one AuthenticationMethod "inforce" at one time.

Attributes

Attribute: AuthenticationMethod.name
Type: string
Description: The name of the Authentication Method.

Attribute: AuthenticationMethod.inForceDate
Type: dateTime
Description: The time that the new AuthenticationMethod is to be used.

Attribute: AuthenticationMethod.retireDate
Type: dateTime
Description: The time that the AuthenticationMethod was removed from use through supercession by a new AuthenticationMethod.

Attribute: AuthenticationMethod.algorithm
Type: string
Description: A description of the algorithm used by the AuthenticationMethod.

Connections

Association

Documents the Authentication used to compute an AuthenticationResult

From Class:	Authenticity::AuthenticationMethod
In the Role of:	appliedAuthenticationMethod
Multiplicity:	1
Description:	The AuthenticationMethod used to calculate the AuthenticationResult

To Class:	Authenticity::AuthenticationResult
In the Role of:	authenticationResult
Multiplicity:	0..*
Description:	The AuthenticationResult calculated using the AuthenticationMethod.

8.5.2.2 Class: Authenticity::AuthenticationResult

The result of applying the AuthenticationMethod to a ManagedRecord

Attributes

Attribute: AuthenticationResult.outcome

Type:	boolean
Description:	False if this AuthenticationResult is not identical to that obtained the first time that the AuthenticationMethod used to calculate this AuthenticationResult was applied.

Attribute: AuthenticationResult.date

Type:	dateTime
Description:	The date/time when the AuthenticationResult was calculated.

Attribute: AuthenticationResult.result

Type:	string
Description:	The actual binary results of applying the AuthenticationMethod's algorithm to the ManagedRecord.

Connections

Aggregation

Associates ManagedRecords and the AuthenticationResults that apply to them.

From Class:	Authenticity::AuthenticationResult
In the Role of:	authenticationResults
Multiplicity:	0..*

Description: The result of applying a particular AuthenticationMethod to a ManagedRecord

To Class: ManagedRecord::ManagedRecord
In the Role of: authenticatedRecord
Multiplicity: 1
Description: The ManagedRecord to which an AuthenticationResult applies.

Association

Documents the Authentication used to compute an AuthenticationResult

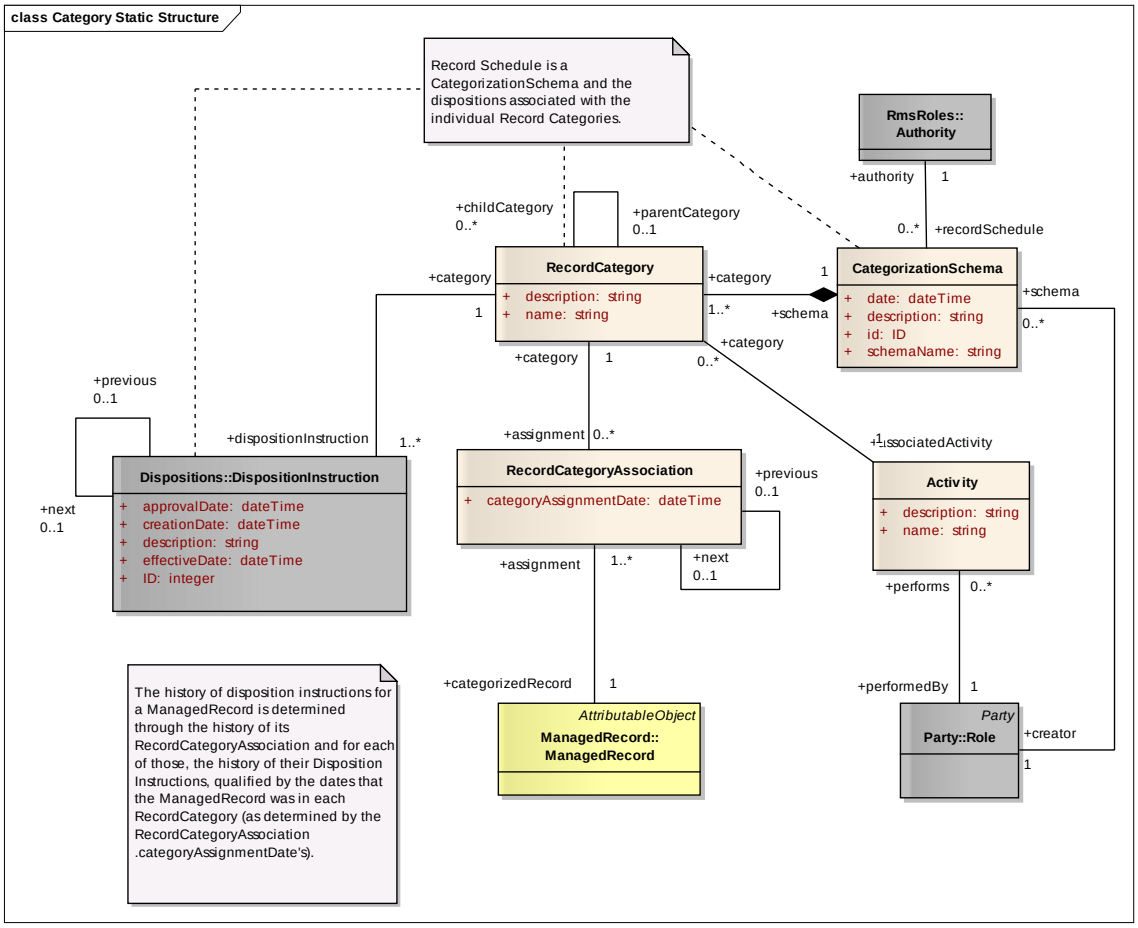
From Class: Authenticity::AuthenticationMethod
In the Role of: appliedAuthenticationMethod
Multiplicity: 1
Description: The AuthenticationMethod used to calculate the AuthenticationResult

To Class: Authenticity::AuthenticationResult
In the Role of: authenticationResult
Multiplicity: 0..*
Description: The AuthenticationResult calculated using the AuthenticationMethod.

8.5.3 Package: Category

The ManagedRecord package collects the elements needed to support the records management concept of record categories.

Category Static Structure



A key concept of managing a record is its RecordCategory. The RecordCategory associates the ManagedRecord with some business Activity that requires that records of it be kept.

The RecordCategory always has a DispositionInstruction associated with it. If it is not know at the time the RecordCategory is created, then it is assigned a default such as "Unknown", or "To Be Determined". The DispositionInstruction governs the ManagedRecords life cycle of retention, transfer, destruction, etc.

The RecordCategory's are accumulated in a Categorization Schema which is flat or made hierarchical through the parent/child relationship. Even as a hierarchy, each RecordCategory will have a DispositionInstruction and may have ManagedRecords assigned to it through a RecordCategoryAssociation.

Note: Record Schedule is a CategorizationSchema and the dispositions associated with the individual Record Categories.

Note: The history of disposition instructions for a ManagedRecord is determined through the history of its RecordCategoryAssociation and for each of those, the history of their

Disposition Instructions, qualified by the dates that the ManagedRecord was in each RecordCategory (as determined by the RecordCategoryAssociation .categoryAssignmentDate's).

8.5.3.1 Class: Category::Activity

A business activity which requires the management of records associated with it.

Attributes

Attribute: Activity.name

Type: string
Description: The name of the business activity

Attribute: Activity.description

Type: string
Description: A textual description of the business activity.

Connections

Association

Associates the RecordCategory with the business Activity documented by records in that RecordCategory

From Class: Category::RecordCategory
In the Role of: category
Multiplicity: 0..*
Description: The RecordCategorys documenting the Activity.

To Class: Category::Activity
In the Role of: associatedActivity
Multiplicity: 1
Description: The Activity associated with the RecordCategory

Association

Documents the Role that performs an Activity

From Class: Category::Activity
In the Role of: performs
Multiplicity: 0..*
Description: Activity performed by a Role

To Class: Party::Role
In the Role of: performedBy
Multiplicity: 1
Description: Role that performs an Activity

8.5.3.2 Class: Category::CategorizationSchema

A Record Schedule is implementable through CategorizationSchema, as can Retention Schedules, etc. CategorizationSchema can be used to organize categories by subject, function, etc.

Attributes

Attribute: CategorizationSchema.id

Type: ID
Description: Unique Identifier

Attribute: CategorizationSchema.schemaName

Type: string
Description: The name of the CategorizationSchema

Attribute: CategorizationSchema.description

Type: string
Description: Description of the CategorySchema.

Attribute: CategorizationSchema.date

Type: dateTime
Description: The date/time that the Categorization was created.

Connections

Aggregation

Aggregates the RecordCategory's that are part of the CategorizationSchema.

From Class: Category::RecordCategory
In the Role of: category
Multiplicity: 1..*
Description: A RecordCategory in the CategorizationSchema

To Class: Category::CategorizationSchema
In the Role of: schema
Multiplicity: 1
Description: The CategorizationSchema containing the RecordCategory

Association

Relates the CategorizationSchema to the Authority that has approved it.

From Class: Category::CategorizationSchema
In the Role of: recordSchedule
Multiplicity: 0..*

Description: The approved CategorizationSchema

To Class: RmsRoles::Authority
In the Role of: authority
Multiplicity: 1
Description: The Authority that approved the CategorizationSchema

Association

Creator of the CategorizationSchema

From Class: Category::CategorizationSchema
In the Role of: schema
Multiplicity: 0..*
Description: The created CategorizationSchema.

To Class: Party::Role
In the Role of: creator
Multiplicity: 1
Description: The creator of the CategorizationSchema

8.5.3.3 Class: Category::RecordCategoryAssociation

Associates ManagedRecords with their RecordCategory

Attributes

Attribute: RecordCategoryAssociation.categoryAssignmentDate
Type: dateTime
Description: The date/time that the ManagedRecord was associated with the RecordCategory

Connections

Association

Links a RecordCategory to its ManagedRecords through RecordCategoryAssociation

From Class: Category::RecordCategoryAssociation
In the Role of: assignment
Multiplicity: 0..*
Description: Assignment of ManagedRecord to RecordCategory.

To Class: Category::RecordCategory
In the Role of: category
Multiplicity: 1
Description: RecordCategory with ManagedRecords

Association

Track the history of a ManagedRecord's RecordCategory assignments.

From Class:	Category::RecordCategoryAssociation
In the Role of:	next
Multiplicity:	0..1
Description:	The next RecordCategory to which the ManagedRecord was assigned. If there is no next association, this assignment is the current one.

To Class:	Category::RecordCategoryAssociation
In the Role of:	previous
Multiplicity:	0..1
Description:	The previous RecordCategory to which the ManagedRecord was assigned.

Association

A ManagedRecord is associated with a RecordCategory through RecordCategoryAssociation. A ManagedRecord is associated with a single RecordCategory at a time. If the Category is not known at time of ManagedRecord set-aside, then it is assigned a default such as "Unknown" or "To Be Determined". The history of RecordCategoryAssociation's are kept through prev/next relationship on RecordCategoryAssociation. The current RecordCategoryAssociation is the latest assigned.

From Class:	ManagedRecord::ManagedRecord
In the Role of:	categorizedRecord
Multiplicity:	1
Description:	The ManagedRecord being categorized.

To Class:	Category::RecordCategoryAssociation
In the Role of:	assignment
Multiplicity:	1..*
Description:	The Category to which the ManagedRecord has been assigned through RecordCategoryAssociation.

8.5.3.4 Class: Category::RecordCategory

The RecordCategory associates the ManagedRecord with some business Activity that requires that records of it be kept.

Attributes

Attribute: RecordCategory.name

Type:	string
Description:	Name of the RecordCategory

Attribute: RecordCategory.description

Type: string
Description: Text description of the RecordCategory

Connections

Association

Links a RecordCategory to its ManagedRecords through RecordCategoryAssociation

From Class: Category::RecordCategoryAssociation
In the Role of: assignment
Multiplicity: 0..*
Description: Assignment of ManagedRecord to RecordCategory.

To Class: Category::RecordCategory
In the Role of: category
Multiplicity: 1
Description: RecordCategory with ManagedRecords

Aggregation

Aggregates the RecordCategory's that are part of the CategorizationSchema.

From Class: Category::RecordCategory
In the Role of: category
Multiplicity: 1..*
Description: A RecordCategory in the CategorizationSchema

To Class: Category::CategorizationSchema
In the Role of: schema
Multiplicity: 1
Description: The CategorizationSchema containing the RecordCategory

Association

Associates the RecordCategory with the business Activity documented by records in that RecordCategory

From Class: Category::RecordCategory
In the Role of: category
Multiplicity: 0..*
Description: The RecordCategorys documenting the Activity.

To Class: Category::Activity
In the Role of: associatedActivity

Multiplicity: 1
Description: The Activity associated with the RecordCategory

Association

Relates a RecordCategory to its history of DispositionInstruction's

From Class: Dispositions::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 1..*
Description: The DispositionInsturction associated with the RecordCategory

To Class: Category::RecordCategory
In the Role of: category
Multiplicity: 1
Description: The RecordCategory associated with the DispositionInstruction.

Association

Hierarchical links among RecordCategory's in a CategorizationSchema

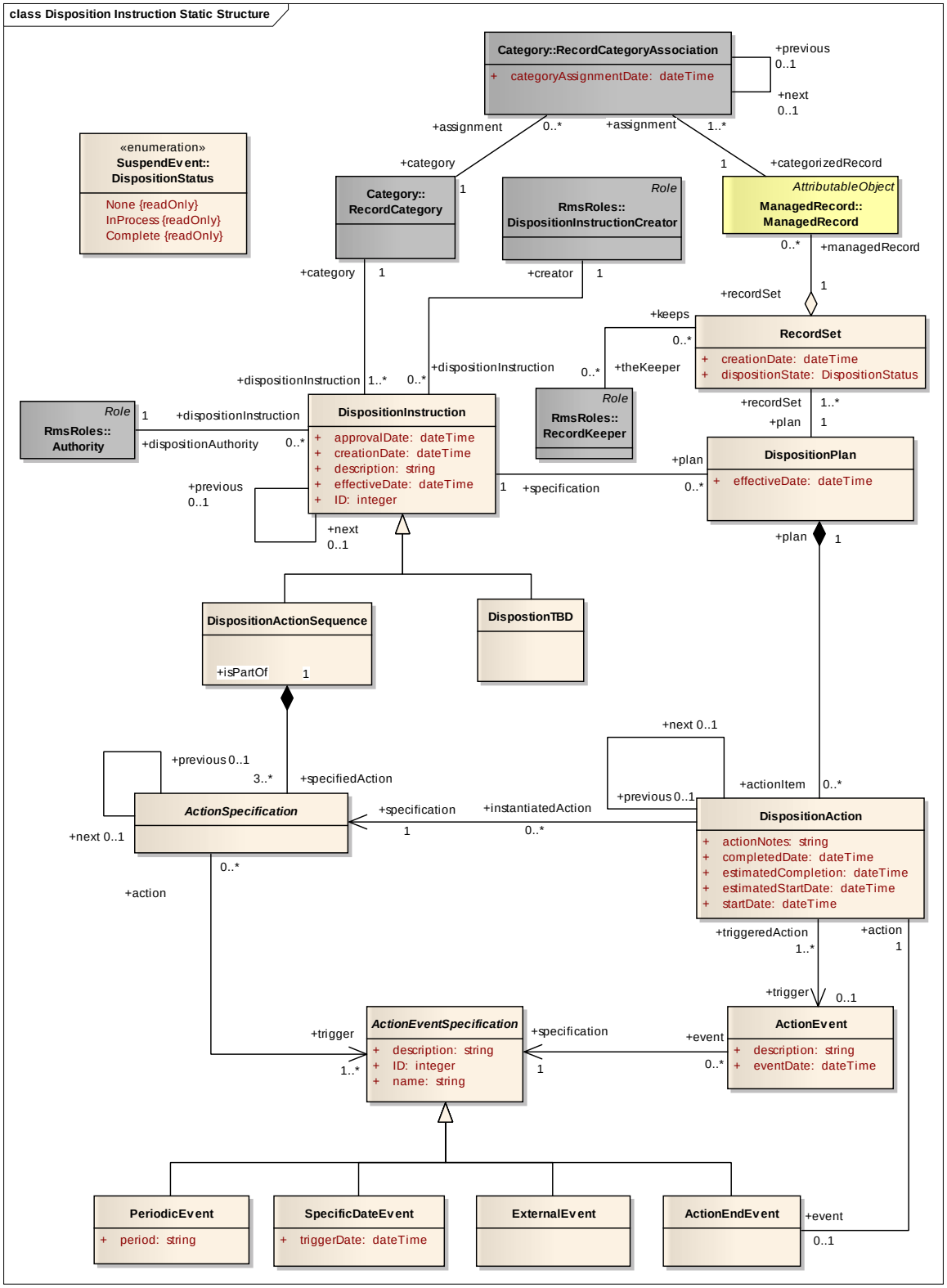
From Class: Category::RecordCategory
In the Role of: parentCategory
Multiplicity: 0..1
Description: The parent RecordCategory

To Class: Category::RecordCategory
In the Role of: childCategory
Multiplicity: 0..*
Description: The child RecordCategory

8.5.4 Package: Dispositions

The Document package collects the elements needed to support records that are one or more electronic "bit streams". Each bit stream is represented by a Document.

Disposition Instruction Static Structure

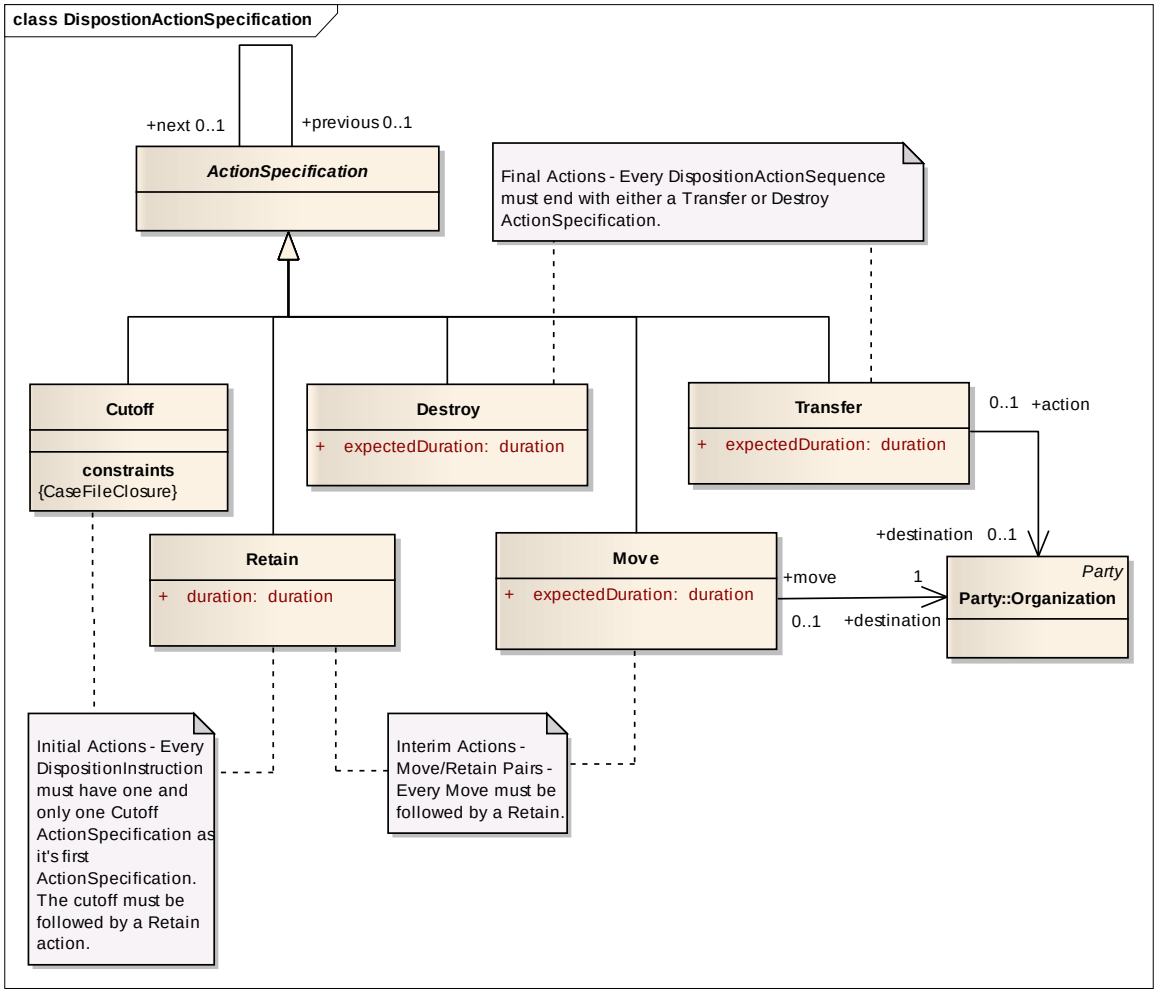


Disposition is one of the key concepts of Records Management. Without Records Management one could (expensively) keep everything in perpetuity, or keep things haphazardly. By assigning a ManagedRecord to a RecordCategory, every ManagedRecord has associated with it a DispositionInstruction which governs its life-cycle. Records are not kept longer than necessary; they are kept as long as needed; and their timely destruction or transfer is assured.

The DispositionInstruction is a "plan" by which the life-cycle of RecordSet's of ManagedRecords is managed. The DispositionInstruction consists of ActionSpecification's which will be performed against RecordSet's created for a DispositionPlan based on the DispositionInstruction.

When a ManagedRecord is added to the system and assigned to a RecordCategory. It is assigned to an open RecordSet created with the DispositionPlan in accordance with the DispositionInstruction. The DispositionInstruction is a "template" for the DispositionPlan, specifying the actions and their trigger events.

DispostionActionSpecification



The first ActionSpecification is always a Cutoff action which "closes" the RecordSet's associated with a DispositionPlan. Cutoff cannot be executed on a RecordSet that contains a ManagedRecord with isCaseFile = True (i.e., the ManagedRecord is a "Case File Record") that has not been "closed" (i.e., ManagedRecord.closedDate has been filled with a valid dateTime.)

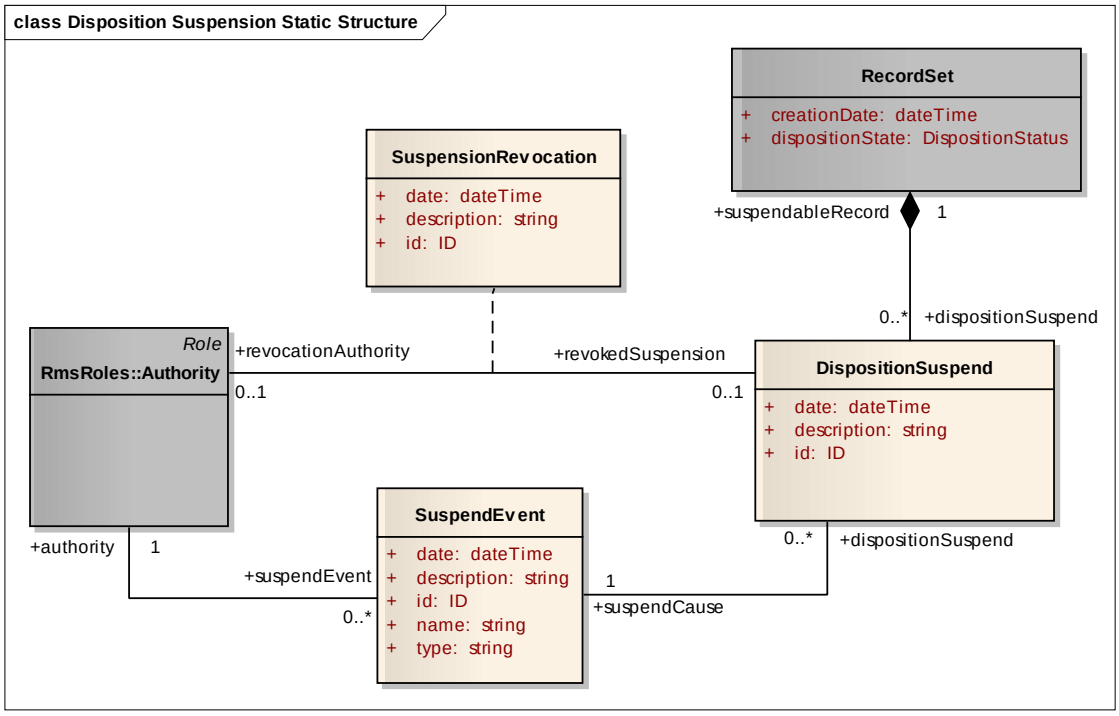
Interim Actions follow. The first interim action is always a Retain (if no retention is desired, then the Retain.duration is set to zero.) If there is a Move action, it must be followed by another Retain action. There may be many Move/Retain pairs of interim actions. The Move action allows the transition of RecordSet's to cheaper remote storage. However, the destination of a Move is a logical destination, an Organization. This specification does not address matters of physical storage.

Each DispositionInstruction ends with a final action, either a Transfer or Destroy. Each of these actions results in the ManagedRecord being deleted from the system.

In a Transfer action, legal custody of Record moves from the current organization of provenance to another. At the end of the operation the ManagedRecord would normally be removed from the Records Management Environment. (A transfer certificate might be kept, but that would be another ManagedRecord)

In a Destroy action, destruction of the ManagedRecords in their RecordSet's immediately begins. (A destruction certificate might be kept, but that would be another ManagedRecord)

Disposition Suspension Static Structure



SuspendEvent's are customarily things like court orders in which the DispositionPlan for a some set of ManagedRecords must be suspended, i.e., nothing more is to be done with the ManagedRecord in terms of Move, Transfer, or Destroy until the suspension is revoked.

DispositionSuspend's are created as a result of a SuspendEvent and are associated with a single RecordSet. If multiple RecordSet's fall under a SuspendEvent then additional DispositionSuspend's are created.

RecordSet's are the "unit of management" of ManagedRecords under a DispositionPlan. However, many RecordSet's can be managed under a single disposition plan. This is useful under multiple suspensions. Take for example a RecordSet-1 (RS1) consisting of three ManagedRecords (MR1, MR2, MR3) that is under some SuspensionEvent (SE1) as documented through the DispositionSuspend (DS1). If there is another suspend event SE2 against MR1, then RS1 will have to be re-factored. RS2 would be created consisting of (MR1) which would be removed from RS1.

Two new DispositionSuspend's would be created to link the SuspensionEvent's to their affected RecordSet's.

SE1 would now point to two DispositionSuspend's (the original DS1 pointing to RS1, and a new DS2 pointing to RS2) while SE2 would point to a new DS3 which would also point to RS2.

The RecordSet aggregates all of the DispositionSuspend's that affect it. The RecordSet's DispositionPlan remains suspended until there is a SuspensionRevocation attached to every DispositionSuspend in the aggregation issued by an appropriate (and recorded) Authority.

8.5.4.1 Class: Dispositions::DispositionInstruction

The DispositionInstruction is used as an "action template" to form DispositionPlan's with one or more RecordSet's in which to accumulate and manage the ManagedRecords.

DispositionInstruction's can change for a RecordCategory. The history is kept by the previous/next association. The current one is the latest entered that has an expired effectiveDate. (It is possible to change a DispositionInstruction effective on so-and-so date, in which case it would be entered here with a future effectiveDate.)

Changing the effectiveDate requires a new DispositionInstruction.

Attributes

Attribute: DispositionInstruction.ID
Type: integer
Description: Unique Identifier

Attribute: DispositionInstruction.description
Type: string
Description: Text description of the DispositionInstruction

Attribute: DispositionInstruction.creationDate
Type: dateTime
Description: The date/time that the DispositionInstruction was created.

Attribute: DispositionInstruction.approvalDate
Type: dateTime
Description: The date/time that the DispositionInstruction was approved by the dispositionAuthority.

Attribute: DispositionInstruction.effectiveDate
Type: dateTime
Description: The effective date/time of the DispositionInstruction.

Connections

Association

Relates a RecordCategory to its history of DispositionInstruction's

From Class:	Dispositions::DispositionInstruction
In the Role of:	dispositionInstruction
Multiplicity:	1..*
Description:	The DispositionInsturction associated with the RecordCategory

To Class:	Category::RecordCategory
In the Role of:	category
Multiplicity:	1
Description:	The RecordCategory associated with the DispositionInstruction.

Generalization

From Class:	Dispositions::DispositionActionSequence

To Class:	Dispositions::DispositionInstruction

Generalization

From Class:	Dispositions::DispostionTBD

To Class:	Dispositions::DispositionInstruction

Association

Associates a DispositionPlan with the template DispositionInstruction from which it is derived.

From Class:	Dispositions::DispositionPlan
In the Role of:	plan
Multiplicity:	0..*
Description:	The DispositionPlan for the RecordSet(s) based on the DispositionInstruction.

To Class:	Dispositions::DispositionInstruction
In the Role of:	specification
Multiplicity:	1
Description:	The DispositionInstruction that served as the specification for the DispositionPlan

Association

Associates a DispositionInstruction with the Authority that approved it.

From Class: RmsRoles::Authority
In the Role of: dispositionAuthority
Multiplicity: 1
Description: The Authority that approved the DispositionInstruction.

To Class: Dispositions::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 0..*
Description: The DispositionInstruction approved by the Authority.

Association

Creator of the DispositionInstruction

From Class: Dispositions::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 0..*
Description: The creator of the DispositionInstruction

To Class: RmsRoles::DispositionInsructionCreator
In the Role of: creator
Multiplicity: 1
Description: The created DispositionInstruction

Association

Records the history of DispositionInstruction's for a RecordCategory. The current, operative DispositionInstruction is the latest one with an expired effectiveDate

From Class: Dispositions::DispositionInstruction
In the Role of: previous
Multiplicity: 0..1
Description: A previous DispositionInstruction.

To Class: Dispositions::DispositionInstruction
In the Role of: next
Multiplicity: 0..1
Description: A next DispositionInstructioin

8.5.4.2 Class: Dispositions::DispostionTBD

A RecordCategory must have a valid DispositionInstruction however in some situations ManagedRecords are being set-aside and there is no approved record schedule. In this circumstance the DispositionInstruction is DispositionTBD which is an "actionless" instruction. This allows a DispositionPlan to be created to collect the ManagedRecords into RecordSet's to "wait" for an approved DispositionInstruction with a

DispositionActionSequence. Under these circumstances there will not be any DispositionAction's accumulated under a DispositionPlan.

Attributes

Connections

Generalization

From Class:	Dispositions::DispostionTBD
To Class:	Dispositions::DispositionInstruction

8.5.4.3 Class: Dispositions::DispositionActionSequence

The specification of a sequence of actions to be used as a template for DispositionPlan's. An DispositionActionSequence has at least three ActionSpecification's beginning with a Cutoff followed by a Retain. Optionally there can be any number of pairs of Move/Retain actions. The final action in a sequence is either a Transfer or Destroy action.

Attributes

Connections

Aggregation

Aggregates ActionSpecification's that constitute a DispositionInstruction.

From Class:	Dispositions::ActionSpecification
In the Role of:	specifiedAction
Multiplicity:	3..*
Description:	An ActionSpecification that is part of the DispositionInstruction

To Class:	Dispositions::DispositionActionSequence
In the Role of:	isPartOf
Multiplicity:	1
Description:	The DispositionInstruction that aggregates the ActionSpecifications that constitute its definition.

Generalization

From Class:	Dispositions::DispositionActionSequence
To Class:	Dispositions::DispositionInstruction

8.5.4.4 Class: Dispositions::ActionEvent

An event conformant with an ActionEventSpecification that triggers a DispositionAction in a DispositionPlan.

Attributes

Attribute: ActionEvent.eventDate

Type: dateTime
Description: The date/time that an ActionEvent occurred.

Attribute: ActionEvent.description

Type: string
Description: Text description of the ActionEvent.

Connections

Association

Associates an ActionEvent with its defining ActionEventSpecification.

From Class: Dispositions::ActionEvent
In the Role of: event
Multiplicity: 0..*
Description: The specified event.

To Class: Dispositions::ActionEventSpecification
In the Role of: specification
Multiplicity: 1
Description: The event specification.

Association

Association of a DispositionAction with the ActionEvent that triggered it.

From Class: Dispositions::DispositionAction
In the Role of: triggeredAction
Multiplicity: 1..*
Description: The DispositionActions triggered by the ActionEvent.

To Class: Dispositions::ActionEvent
In the Role of: trigger
Multiplicity: 0..1
Description: The ActionEvent that triggered the DispositionAction

8.5.4.5 Class: Dispositions::ActionSpecification

ActionSpecification's are the only possible actions that can be performed on a RecordSet. Each is triggered by an event specified by an ActionEventSpecification.

Attributes

Connections

Generalization

From Class: Dispositions::Move

To Class: Dispositions::ActionSpecification

Generalization

From Class: Dispositions::Retain

To Class: Dispositions::ActionSpecification

Generalization

From Class: Dispositions::Cutoff

To Class: Dispositions::ActionSpecification

Association

DispositionAction is instantiated when a DispositionPlan is created for a new RecordSet. The ActionSpecification's in the DispositionInstruction associated with the DispositionPlan serve as the template.

From Class:	Dispositions::DispositionAction
In the Role of:	instantiatedAction
Multiplicity:	0..*
Description:	A DispositionAction based on the ActionSpecification

To Class:	Dispositions::ActionSpecification
In the Role of:	specification
Multiplicity:	1
Description:	The ActionSpecification that served as the template for the DispositionAction.

Association

Associates the specification of the trigger(s) that can initiate this ActionSpecification (as instantiated in a DispositionAction and occurring in an ActionEvent)

From Class:	Dispositions::ActionSpecification
In the Role of:	action
Multiplicity:	0..*
Description:	The ActionSpecification for which an ActionEventSpecifications (trigger specifications) is associated.

To Class:	Dispositions::ActionEventSpecification
In the Role of:	trigger
Multiplicity:	1..*
Description:	The ActionEventSpecifications (triggers) that can actuate this DispositionActions based on this ActionSpecification

Aggregation

Aggregates ActionSpecification's that constitute a DispositionInstruction.

From Class:	Dispositions::ActionSpecification
In the Role of:	specifiedAction
Multiplicity:	3..*
Description:	An ActionSpecification that is part of the DispositionInstruction

To Class:	Dispositions::DispositionActionSequence
In the Role of:	isPartOf
Multiplicity:	1
Description:	The DispositionInstruction that aggregates the ActionSpecifications that constitute its definition.

Association

Specifies the action sequence of a DispositionInstruction

From Class:	Dispositions::ActionSpecification
In the Role of:	previous
Multiplicity:	0..1
Description:	The action which must occur previous to this action. If there is no "previous" then this must be a Cutoff.

To Class:	Dispositions::ActionSpecification
In the Role of:	next
Multiplicity:	0..1

Description: The action that will occur after this action. If there is no "next" then this ActionSpecification must be a Transfer or Destroy.

Generalization

From Class: Dispositions::Destroy

To Class: Dispositions::ActionSpecification

Generalization

From Class: Dispositions::Transfer

To Class: Dispositions::ActionSpecification

8.5.4.6 Class: Dispositions::Cutoff

Cutoff is like a start node in a process specification for actions on a RecordSet. It is most often triggered by a PeriodicEvent, but no matter what triggers it, the action goes immediately to whatever the next step in the process is, i.e., there is no "duration" to the action. (It is in a real sense an "event" in it's own right).

Attributes

Connections

Generalization

From Class: Dispositions::Cutoff

To Class: Dispositions::ActionSpecification

Constraints

CaseFileClosure

Description: Cutoff cannot be executed on a RecordSet that contains a ManagedRecord with isCaseFile = True that has not been closed.

8.5.4.7 Class: Dispositions::Retain

An action that the RecordSet is to be retained as is for the specified Retain.duration.

The work Hold is often used as synonymous with Retain.

Attributes

Attribute: Retain.duration

Type: duration
Description: The duration for which a RecordSet is to be retained before further action.

Connections

Generalization

From Class: Dispositions::Retain

To Class: Dispositions::ActionSpecification

8.5.4.8 Class: Dispositions::Move

A move is not a record transfer. It is the change of control to another location without transfer of legal custody. All the metadata stays with RECORDS MANAGEMENT ENVIRONMENT of the legal custodian.

Location of the record has been changed but the legal custody of the record emains the same. If a record is in a Move action it's still part of the disposition.

Location concepts in RMS are not to be thought of as managing a record's physical location in a server farm, for example. Location is a logical concept, not a physical one.

Attributes

Attribute: Move.expectedDuration

Type: duration
Description: The expected duration of the Move action for the RecordSet. This is used for projection of workload & may be updated from time to time to improve the estimate.

Connections

Association

Associates the Move action with its logical destination Organization.

From Class: Dispositions::Move
In the Role of: move
Multiplicity: 0..1
Description: The Move action requiring the move to the Organization

To Class: Party::Organization
In the Role of: destination

Multiplicity: 1
Description: The logical destination (Organization) of the Move action.

Generalization

From Class: Dispositions::Move

To Class: Dispositions::ActionSpecification

8.5.4.9 Class: Dispositions::Transfer

Legal custody of Record moves from the current organization of provenance to another. At the end of the operation the ManagedRecord would normally be removed from the Records Management Environment. (A transfer certificate might be kept, but that would be another ManagedRecord)

Location concepts in RMS are not to be thought of as specifying a record's physical location in a server farm, for example. Location is a logical concept, not a physical one and refers to an Organization.

Attributes

Attribute: Transfer.expectedDuration

Type: duration
Description: The expected duration of the Transfer action for the RecordSet. This is used for projection of workload & may be updated from time to time to improve the estimate.

Connections

Association

Associates a Transfer action with the destination Organization.

From Class: Dispositions::Transfer
In the Role of: action
Multiplicity: 0..1
Description: The transfer action.

To Class: Party::Organization
In the Role of: destination
Multiplicity: 0..1
Description: The destination Organization of the Transfer action.

Generalization

From Class: Dispositions::Transfer

To Class: Dispositions::ActionSpecification

8.5.4.10 Class: Dispositions::Destroy

This ActionSpecification is one of the "final actions" to be taken against the ManagedRecords in a RecordSet. The ManagedRecords (and their Documents and associated metadata not shared with other ManagedRecords) are destroyed.

Attributes

Attribute: Destroy.expectedDuration

Type: duration

Description: The expected duration of the Destroy action. This is used for projection of workload & may be updated from time to time to improve the estimate.

Connections

Generalization

From Class: Dispositions::Destroy

To Class: Dispositions::ActionSpecification

8.5.4.11 Class: Dispositions::ActionEventSpecification

Used to build DispositionInstruction's which consist of ActionSpecification's whose triggering events must be specified. This is the specification of the events that can be or are used in an ActionSpecification.

Attributes

Attribute: ActionEventSpecification.ID

Type: integer

Description: Unique identifier.

Attribute: ActionEventSpecification.name

Type: string

Description: The name of the ActionEventSpecification

Attribute: ActionEventSpecification.description

Type: string

Description: A text description of the ActionEventSpecification

Connections

Association

Associates an ActionEvent with its defining ActionEventSpecification.

From Class:	Dispositions::ActionEvent
In the Role of:	event
Multiplicity:	0..*
Description:	The specified event.

To Class:	Dispositions::ActionEventSpecification
In the Role of:	specification
Multiplicity:	1
Description:	The event specification.

Association

Associates the specification of the trigger(s) that can initiate this ActionSpecification (as instantiated in a DispositionAction and occurring in an ActionEvent)

From Class:	Dispositions::ActionSpecification
In the Role of:	action
Multiplicity:	0..*
Description:	The ActionSpecification for which an ActionEventSpecifications (trigger specifications) is associated.

To Class:	Dispositions::ActionEventSpecification
In the Role of:	trigger
Multiplicity:	1..*
Description:	The ActionEventSpecifications (triggers) that can actuate this DispositionActions based on this ActionSpecification

Generalization

From Class:	Dispositions::ActionEndEvent

To Class:	Dispositions::ActionEventSpecification

Generalization

From Class:	Dispositions::ExternalEvent

To Class:	Dispositions::ActionEventSpecification

Generalization

From Class:	Dispositions::SpecificDateEvent

To Class:	Dispositions::ActionEventSpecification

Generalization

From Class: Dispositions::PeriodicEvent

To Class: Dispositions::ActionEventSpecification

8.5.4.12 Class: Dispositions::PeriodicEvent

A periodic event, e.g., year-end, school-year-end, month-end. The instances of this can be numerous and often dependent on the organization (e.g., school-year-end, fiscal-year-end), therefore no attempt to enumerate the possibilities is made. The PeriodicEvent.period must be a string that is interpretable to a system in order to make the DispositionAction.estimatedStartDate's for Cutoff, etc.

Attributes

Attribute: PeriodicEvent.period

Type: string

Description: A string representing the event period (e.g., calendar-year-end, fiscal-year-end, month-end, etc.)

Connections

Generalization

From Class: Dispositions::PeriodicEvent

To Class: Dispositions::ActionEventSpecification

8.5.4.13 Class: Dispositions::SpecificDateEvent

An event that is defined in terms of a specific date/time.

Attributes

Attribute: SpecificDateEvent.triggerDate

Type: dateTime

Description: The date/time at which the event is occurs and triggers whatever appropriate DispositionAction's.

Connections

Generalization

From Class: Dispositions::SpecificDateEvent

To Class: Dispositions::ActionEventSpecification

8.5.4.14 Class: Dispositions::ExternalEvent

An event that is specific to the Records Management Environment deployment, but is not generated (or even necessarily anticipated) by the Environment itself.

Attributes

Connections

Generalization

From Class: Dispositions::ExternalEvent

To Class: Dispositions::ActionEventSpecification

8.5.4.15 Class: Dispositions::ActionEndEvent

The event of completion of a specific DispositionAction.

Attributes

Connections

Association

Associates an ActionEndEvent with the DispositionAction that ended.

From Class: Dispositions::ActionEndEvent
In the Role of: event
Multiplicity: 0..1
Description: The event of the DispositionAction ending.

To Class: Dispositions::DispositionAction
In the Role of: action
Multiplicity: 1
Description: The DispositionAction that has ended.

Generalization

From Class: Dispositions::ActionEndEvent

To Class: Dispositions::ActionEventSpecification

8.5.4.16 Class: Dispositions::DispositionPlan

The DispositionPlan is an "instantiation" of the DispositionInstruction applied to a particular RecordSet as defined by a Cutoff action. It aggregates DispositionAction's that are similarly derived from ActionSpecification's.

Attributes

Attribute: DispositionPlan.creationDate

Type:	dateTime
Description:	The creation date/time of the DispositionPlan. In many cases this is the same as the cutoffDate of the "previous" RecordSet, however DispositionPlan's may be established ahead of time in anticipation of the Cutoff event.

Connections

Association

Associates a DispositionPlan with the template DispositionInstruction from which it is derived.

From Class:	Dispositions::DispositionPlan
In the Role of:	plan
Multiplicity:	0..*
Description:	The DispositionPlan for the RecordSet(s) based on the DispositionInstruction.
To Class:	Dispositions::DispositionInstruction
In the Role of:	specification
Multiplicity:	1
Description:	The DispositionInstruction that served as the specification for the DispositionPlan

Association

Associates a DispositionPlan with the RecordSet's subject to it.

From Class:	Dispositions::RecordSet
In the Role of:	recordSet
Multiplicity:	1..*
Description:	A RecordSet subject to the DispositionPlan
To Class:	Dispositions::DispositionPlan
In the Role of:	plan
Multiplicity:	1
Description:	The DispositionPlan governing the disposition of the RecordSet.

Aggregation

DispositionPlan is an "instantiation" of a DispositionInstruction which serves as a template of action for any given RecordSet. DispositionPlan's

aggregate DispositionAction's which are based on the ActionSpecification's of the DispositionInstruction.

From Class:	Dispositions::DispositionAction
In the Role of:	actionItem
Multiplicity:	0..*
Description:	The action planned or completed for the DispositionPlan

To Class:	Dispositions::DispositionPlan
In the Role of:	plan
Multiplicity:	1
Description:	The DispositionPlan of which the DispositionAction is a part of.

8.5.4.17 Class: Dispositions::RecordSet

RecordSet's are the fundamental unit of Disposition. DispositionPlan's are applied against RecordSet's. As ManagedRecords are set-aside and assigned RecordCategory's, they are placed in RecordSet's until a Cutoff event (which may be immediate, but more usually associated with a periodicity such as monthly or quarterly). The Cutoff event is the first event in RecordSet's DispositionPlan which dictates its life-cycle.

The ManagedRecords governed by a DispositionPlan are often in one RecordSet; however, they may be partitioned into multiple RecordSet's for any business purpose.

A mandatory partitioning into RecordSet's occurs for suspended records (see the DispositionSuspend Package). Records governed by a particular DispositionPlan but suspended by some Authority are segregated into one or more RecordSet's assigned to the suspension.

Attributes

Attribute: RecordSet.creationDate

Type:	dateTime
Description:	The creation date/time of the RecordSet. Often corresponds to the cutoffDate of the previous RecordSet subject to the DispositionInstruction. However, the RecordSet (along with its DispositionPlan) may be created in anticipation of the Cutoff.

Attribute: RecordSet.dispositionState

Type:	DispositionStatus
Description:	The status of final disposition for the RecordSet.

Connections

Association

Collects the ManagedRecords under a RecordSet which is used for its Disposition. RecordSet's are determined by the ManagedRecord's RecordCategoryAssociation as documented in the Category Package and the Dispositions Package.

From Class:	ManagedRecord::ManagedRecord
In the Role of:	managedRecord
Multiplicity:	0..*
Description:	The ManagedRecord assigned to a RecordSet

To Class:	Dispositions::RecordSet
In the Role of:	recordSet
Multiplicity:	1
Description:	The RecordSet containing the ManagedRecord

Association

RecordSet's that are subject to suspension of their disposition form an aggregation of those ManagedRecords.

From Class:	Dispositions::DispositionSuspend
In the Role of:	dispositionSuspend
Multiplicity:	0..*
Description:	The DispositionSuspend that has been levied against the RecordSet

To Class:	Dispositions::RecordSet
In the Role of:	suspendableRecord
Multiplicity:	1
Description:	The RecordSet against which a DispositionSuspend has been levied.

Association

Associates a DispositionPlan with the RecordSet's subject to it.

From Class:	Dispositions::RecordSet
In the Role of:	recordSet
Multiplicity:	1..*
Description:	A RecordSet subject to the DispositionPlan

To Class:	Dispositions::DispositionPlan
In the Role of:	plan
Multiplicity:	1
Description:	The DispositionPlan governing the disposition of the RecordSet.

Association

Documents a RecordSet currently assigned to a RecordKeeper. The current RecordKeeper is identified by the latest assigned RecordKeeper.

From Class:	RmsRoles::RecordKeeper
In the Role of:	theKeeper
Multiplicity:	0..*
Description:	The ManagedRecord's assigned RecordKeeper

To Class:	Dispositions::RecordSet
In the Role of:	keeps
Multiplicity:	0..*
Description:	The Party that serves as RecordKeeper for the ManagedRecord.

8.5.4.18 Enumeration: Dispositions::DispositionStatus

The possible values of dispositionStatus for a ManagedRecord

Attributes

Attribute: DispositionStatus.None

Type:	unspecified
Description:	No final disposition has been triggered for the ManagedRecord

Attribute: DispositionStatus.InProcess

Type:	unspecified
Description:	A final disposition has been triggered for this RecordSet. It is in process of Transfer, or Destruction. It should be treated as no longer part of the Records Management Environment. However, it is possible that a suspension can be placed on a RecordSet that has dispositionStatus of "inProcess". Even if the RecordSet has been partially destroyed or transferred, the action is suspended.

Attribute: DispositionStatus.Complete

Type:	unspecified
Description:	Final disposition has been completed. This is used for transfers and marks the completion of transferring the RecordSet to its destination organization. The RecordSet is now eligible for removal (deletion) from the system. As always, until the RecordSet has actually been deleted, it is subject to DispositionSuspend.

Connections

8.5.4.19 Class: Dispositions::DispositionAction

DispositionAction is an "instantiation" based on the template ActionSpecification, used to populate a DispositionPlan based on its template, DispositionInstruction.

Attributes

Attribute: DispositionAction.estimatedStartDate

Type: dateTime
Description: The estimated start date/time for the action. This is based on calculations of estimated dates and durations of preceding DispositionAction's and will require update from time to time as more information is know. This is used to help project work loads based on particular actions scheduled for RecordSet's.

Attribute: DispositionAction.estimatedCompletion

Type: dateTime
Description: Based on the estimatedStartDate together with the expectedDuration of the ActionSpecification. If there is no duration attribute, it is assumed to be immediate, i.e., zero duration, e.g., Cutoff.

Attribute: DispositionAction.startDate

Type: dateTime
Description: The actual start date/time of the action.

Attribute: DispositionAction.completedDate

Type: dateTime
Description: The actual completion date/time of the action.

Attribute: DispositionAction.actionNotes

Type: string
Description: Any notation concerning the execution of the DispositionAction.

Connections

Association

DispositionAction is instantiated when a DispositionPlan is created for a new RecordSet. The ActionSpecification's in the DispositionInstruction associated with the DispositionPlan serve as the template.

From Class: Dispositions::DispositionAction
In the Role of: instantiatedAction
Multiplicity: 0..*

Description: A DispositionAction based on the ActionSpecification

To Class: Dispositions::ActionSpecification
In the Role of: specification
Multiplicity: 1
Description: The ActionSpecification that served as the template for the DispositionAction.

Association

Association of a DispositionAction with the ActionEvent that triggered it.

From Class: Dispositions::DispositionAction
In the Role of: triggeredAction
Multiplicity: 1..*
Description: The DispositionActions triggered by the ActionEvent.

To Class: Dispositions::ActionEvent
In the Role of: trigger
Multiplicity: 0..1
Description: The ActionEvent that triggered the DispositionAction

Association

Mirroring the next/previous of the ActionSpecification order, this is an instantiation of those actions as a DispositionPlan for a given RecordSet as determined by Cutoff.

From Class: Dispositions::DispositionAction
In the Role of: next
Multiplicity: 0..1
Description: The next DispositionAction to be performed or to be performed.

To Class: Dispositions::DispositionAction
In the Role of: previous
Multiplicity: 0..1
Description: The previous DispositionAction to be performed or to be performed.

Aggregation

DispositionPlan is an "instantiation" of a DispositionInstruction which serves as a template of action for any given RecordSet. DispositionPlan's aggregate DispositionAction's which are based on the ActionSpecification's of the DispositionInstruction.

From Class: Dispositions::DispositionAction

In the Role of: actionItem
Multiplicity: 0..*
Description: The action planned or completed for the DispositionPlan

To Class: Dispositions::DispositionPlan
In the Role of: plan
Multiplicity: 1
Description: The DispositionPlan of which the DispositionAction is a part of.

Association

Associates an ActionEndEvent with the DispositionAction that ended.

From Class: Dispositions::ActionEndEvent
In the Role of: event
Multiplicity: 0..1
Description: The event of the DispositionAction ending.

To Class: Dispositions::DispositionAction
In the Role of: action
Multiplicity: 1
Description: The DispositionAction that has ended.

8.5.4.20 Class: Dispositions::SuspendEvent

A SuspendEvent can be the action of a court, in which case it is a suspend order or it may be the result of an action by a record manager or some other authorized party. The SuspendEvent is associated with the DispositionSuspend's that it is responsible for. The DispositionSuspend's apply to RecordSet's. Under discovery the SuspendEvent can expand the ManagedRecords affected by adding them to a RecordSet (when subject to the same DispositionPlan), or by creating new RecordSet's under previously unaffected DispositionPlan's.

Attributes

Attribute: SuspendEvent.id
Type: ID
Description: Unique Identifier

Attribute: SuspendEvent.description
Type: string
Description: Text Description of the SuspendEvent (e.g., the court order or some other action that causes

Attribute: SuspendEvent.name
Type: string
Description: Name of the SuspendEvent

Attribute: SuspendEvent.date

Type: dateTime
Description: The date/time that the SuspendEvent occurred.

Attribute: SuspendEvent.type

Type: string
Description: The type of the SuspendEvent. These values are locally defined.

Connections

Association

Associates a SuspendEvent with the Authority which declared it.

From Class: Dispositions::SuspendEvent
In the Role of: suspendEvent
Multiplicity: 0..*
Description: The authorized SuspendEvent.

To Class: RmsRoles::Authority
In the Role of: authority
Multiplicity: 1
Description: The Authority declaring the SuspendEvent.

Association

Associates the SuspendEvent with the DispositionSuspend's that have been established in compliance with the SuspendEvent.

From Class: Dispositions::SuspendEvent
In the Role of: suspendCause
Multiplicity: 1
Description: The SuspendEvent that resulted in the DispositionSuspend.

To Class: Dispositions::DispositionSuspend
In the Role of: dispositionSuspend
Multiplicity: 0..*
Description: One of the DispositionSuspends that resulted from a SuspendEvent.

8.5.4.21 Class: Dispositions::DispositionSuspend

One or more DispositionSuspend can be placed agains a RecordSet. If the RecordSet has records that are not to be suspended then the RecordSet is repartitioned so that the DispositionSuspend applies to a RecordSet in its entirety.

No interim or final disposition can be executed on a ManagedRecord in a RecordSet.that has a DispositionSuspend unless that DispositionSuspend has a SuspensionRevocation associated with it.

Attributes

Attribute: DispositionSuspend.id

Type: ID
Description: Unique identifier

Attribute: DispositionSuspend.description

Type: string
Description: A textual description of the DispositionSuspend.

Attribute: DispositionSuspend.date

Type: dateTime
Description: The date/time of the creation of the DispositionSuspend.

Connections

Association

RecordSet's that are subject to suspension of their disposition form an aggregation of those ManagedRecords.

From Class:	Dispositions::DispositionSuspend
In the Role of:	dispositionSuspend
Multiplicity:	0..*
Description:	The DispositionSuspend that has been levied against the RecordSet

To Class:	Dispositions::RecordSet
In the Role of:	suspendableRecord
Multiplicity:	1
Description:	The RecordSet against which a DispositionSuspend has been levied.

AssociationClass

Records the authorized revocation of a particular DispositionSuspend.

From Class:	RmsRoles::Authority
In the Role of:	revocationAuthority
Multiplicity:	0..1
Description:	The Authority authorizing the revocation of a DispositionSuspend.

To Class:	Dispositions::DispositionSuspend

In the Role of: revokedSuspension
Multiplicity: 0..1
Description: A DispositionSuspend revoked by the Authority.

Association

Associates the SuspendEvent with the DispositionSuspend's that have been established in compliance with the SuspendEvent.

From Class: Dispositions::SuspendEvent
In the Role of: suspendCause
Multiplicity: 1
Description: The SuspendEvent that resulted in the DispositionSuspend.

To Class: Dispositions::DispositionSuspend
In the Role of: dispositionSuspend
Multiplicity: 0..*
Description: One of the DispositionSuspends that resulted from a SuspendEvent.

8.5.4.22 AssociationClass: Dispositions::SuspensionRevocation

A SuspensionRevocation releases a RecordSet from a particular DispositionSuspend. Note that a RecordSet can be under multiple DispositionSuspend's. Only when all DispositionSuspend's have been revoked through the creation of a SuspensionRevocation authorized by a suitable Party can final disposition proceed.

Attributes

Attribute: SuspensionRevocation.id
Type: ID
Description: Unique identifier

Attribute: SuspensionRevocation.date
Type: dateTime
Description: The date/time that the SuspensionRevocation is authorized.

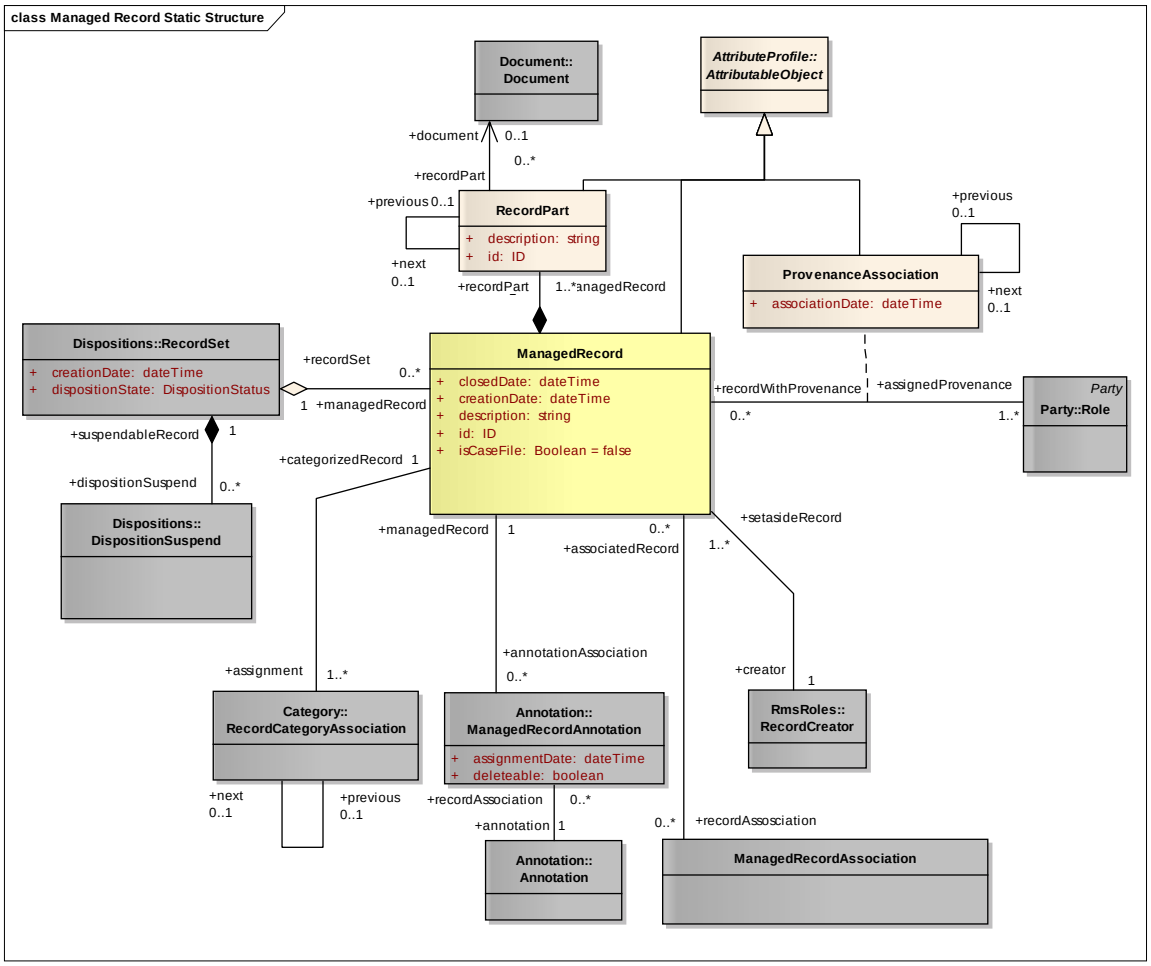
Attribute: SuspensionRevocation.description
Type: string
Description: A textual description of the SuspensionRevocation

Connections

8.5.5 Package: ManagedRecord

The ManagedRecord package collects the elements needed to support the basic concepts of a ManagedRecord. It is shown here in its full context of many of the key concepts associated with managing a record.

Managed Record Static Structure

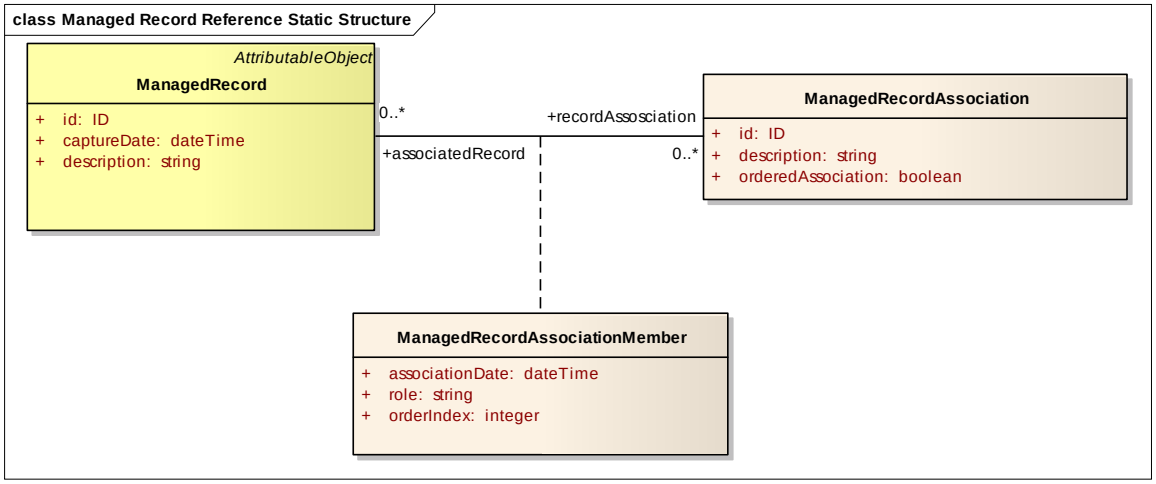


ManagedRecord is shown here in its full context of the key concepts of managing a record. The ManagedRecord always has a ProvenanceAssociation that documents the custodial organization responsible for the ManagedRecord. A ManagedRecord can consist of many RecordPart's which are Document's (i.e., bit streams). The "usual" ManagedRecord (i.e., a record that is not a Case File) are comprised of immutable RecordPart's which is to say that once a ManagedRecord is "set aside" there can be no changes to the RecordPart's (either by deletion of addition), or to their corresponding Document.

This specification is not prescriptive as to what is and what is not a record to be managed under these services. Each organization should comply with what is legally ascribed to their definition of record. In order to eliminate conflict for implementation, the

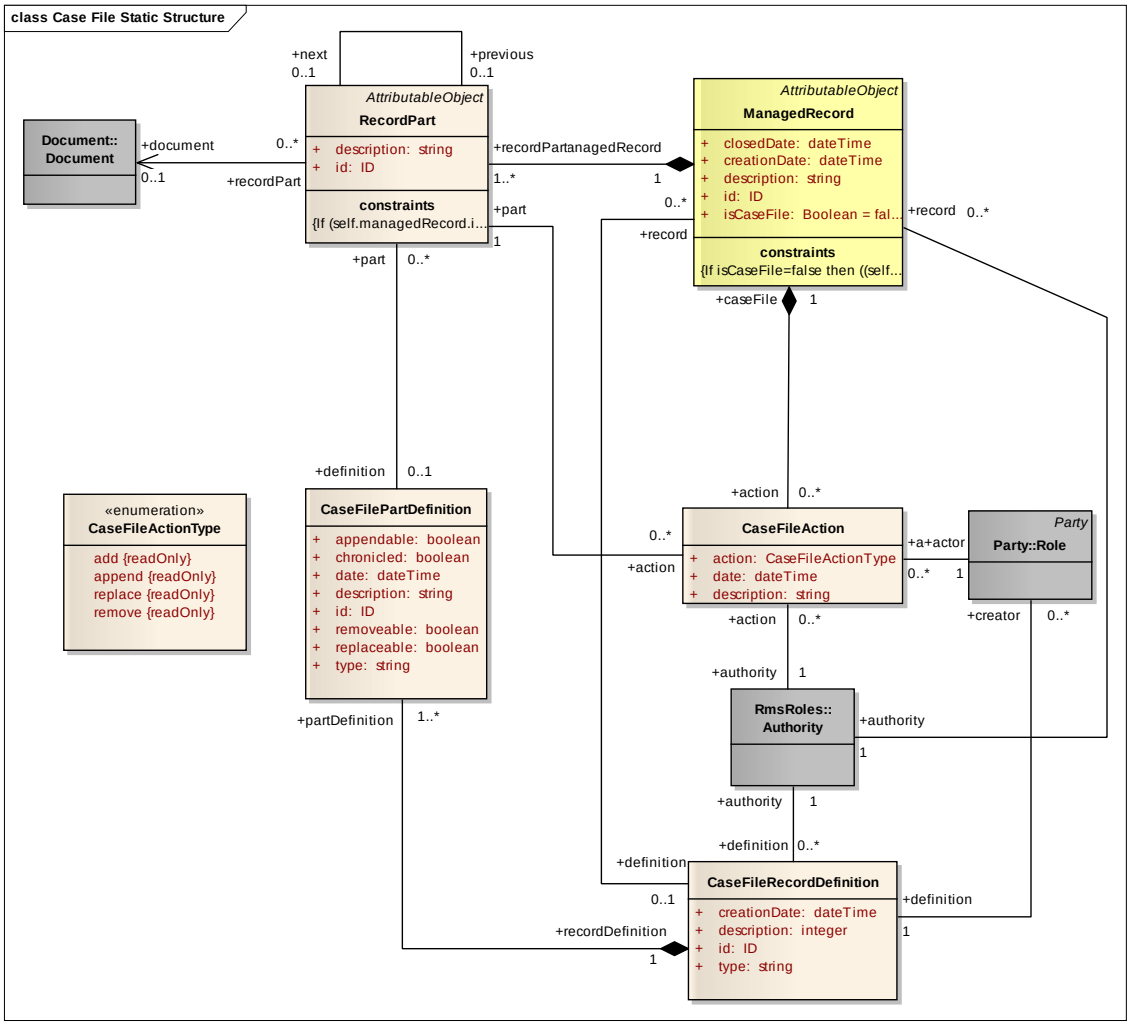
description used to create the services is provided without prejudice to any formal, legal, or socially accepted definitions that may exist. "The reader is reminded the business owner decides when and what to set aside as a record. When that decision is made the functions described in the services can be applied to the record, assuring its proper management and disposition. The record as set aside by the business owner remains unchanged even as the records management attributes are populated and updated during its life-cycle. The sum of a record and its records management attributes (current and historical) is a managed record." Interagency Project Team and the Records Management Service Components Program Office of the National Archives and Records Administration, Functional Requirements, Attributes, and Unified Modeling Language Class Diagrams for Records Management Services, September 7, 2006

Managed Record Reference Static Structure



ManagedRecordAssociation supports the grouping of ManagedRecords into groups (ordered or unordered, differentiated or undifferentiated) for any business purpose.

Case File Static Structure



ManagedRecord's are generally immutable. Once set-aside, it's authentication base is calculated and no change to the managed record is allowed. At any time the authentication can be re-calculated and any deviation from the base authentication indicates that the ManagedRecord is not faithfully that which had been set-aside originally.

However, immutability is not a constraint for CaseFileRecords. A ManagedRecord is designated a CaseFileRecord by setting ManagedRecord.isCaseFile to True.

In the situation of Case Files, certain RecordPart's may be allowed to be added, appended, removed (deleted), or replaced (those actions enumerated in CaseFileActionType.) These actions are constrained by the RecordPart's CaseFilePartDefinition which is contained in an overall description of the CaseFileRecord and its rules in CaseFileRecordDefinition.

All actions taken on the CaseFileRecord are recorded in CaseFileAction along with the Party::Role which took the action and the RmsRoles::Authority which authorized the taking of such action.

8.5.5.1 Class: ManagedRecord::RecordPart

A ManagedRecord can be comprised of numerous Documents (bit strings) as collected by RecordPart.

Attributes

Attribute: RecordPart.id

Type: ID
Description: Unique Identifier

Connections

Constraint Name: Only a RecordPart that is part of a case file can exist without pointing to a Document.

Generalization

From Class: ManagedRecord::RecordPart

To Class: AttributeProfile::AttributableObject

Association

Associates the ManagedRecord with the Documents that comprise it.

From Class: ManagedRecord::RecordPart
In the Role of: recordPart
Multiplicity: 0..*
Description: The part membership of a document in a ManagedRecord.

To Class: Document::Document
In the Role of: document
Multiplicity: 0..1
Description: The document that is part of the ManagedRecord

Association

Collects the Documents (bit streams) that constitute the ManagedRecord. A Document can be included in more than one ManagedRecord. If a ManagedRecord containing the Document is "destroyed", the Document will only be "destroyed" if it does not participate in any other ManagedRecord. This happens when a Document serves more than one

business purpose or task the artifacts of which are being managed in the Records Management Environment.

From Class:	ManagedRecord::ManagedRecord
In the Role of:	managedRecord
Multiplicity:	1
Description:	The ManagedRecord

To Class:	ManagedRecord::RecordPart
In the Role of:	recordPart
Multiplicity:	1..*
Description:	One of the Documents (perhaps the only) that comprise the ManagedRecord.

8.5.5.2 Class: ManagedRecord::ManagedRecord

An instance of this class represents one distinct managed record. All information associated with this particular managed record is held in instances of related classes linked to this instance of ManagedRecord.

All relationships between ManagedRecord and the associated classes constituting the complete information set for one managed record are designed in a way, so that deletion of the ManagedRecord instance causes the automatic deletion of all other instances constituting this information set.

This specification is not prescriptive as to what is and what is not a record to be managed under these services. Each organization should comply with what is legally ascribed to their definition of record. In order to eliminate conflict for implementation, the description used to create the services is provided without prejudice to any formal, legal, or socially accepted definitions that may exist. "The reader is reminded the business owner decides when and what to set aside as a record. When that decision is made the functions described in the services can be applied to the record, assuring its proper management and disposition. The record as set aside by the business owner remains unchanged even as the records management attributes are populated and updated during its life-cycle. The sum of a record and its records management attributes (current and historical) is a managed record." Interagency Project Team and the Records Management Service Components Program Office of the National Archives and Records Administration, Functional Requirements, Attributes, and Unified Modeling Language Class Diagrams for Records Management Services, September 7, 2006

Attributes

Attribute: ManagedRecord.id

Type:	ID
Description:	Unique Identifier

Attribute: ManagedRecord.captureDate

Type: dateTime
Description: The date/time that the Record was set-aside for treatment as a ManagedRecord.

Attribute: ManagedRecord.description

Type: string
Description: Text describing the ManagedRecord

Connections

Constraint Name: If isCaseFile=false then ((self.action is empty) and (self.definition is null))

Generalization

From Class: ManagedRecord::ManagedRecord

To Class: AttributeProfile::AttributableObject

Association

Documents the Party which set-aside the ManagedRecord

From Class: ManagedRecord::ManagedRecord
In the Role of: setasideRecord
Multiplicity: 1..*
Description: The ManagedRecord.

To Class: RmsRoles::RecordCreator
In the Role of: creator
Multiplicity: 1
Description: The Party that set-aside (created) the ManagedRecord. This has nothing to do with who created the Documents that comprise the ManagedRecord.

AssociationClass

Records the current and historical provenance of the ManagedRecord.

From Class: Party::Role
In the Role of: assignedProvenance
Multiplicity: 1..*
Description: The Role to whom or which the Provenance of the ManagedRecord is assigned in this ProvenanceAssociation.

To Class: ManagedRecord::ManagedRecord
In the Role of: recordWithProvenance

Multiplicity: 0..*
Description: The provenance of the ManagedRecord as recorded in this ProvenanceAssociation.

AssociationClass

Links ManagedRecord's into 0, 1 or more groupings (ManagedRecordAssociation's). ManagedRecordAssociation's may be empty, i.e., they are pre-defined and waiting for members to be assigned.

From Class: ManagedRecord::ManagedRecord
In the Role of: associatedRecord
Multiplicity: 0..*
Description: The ManagedRecord in the ManagedRecordAssociation.

To Class: ManagedRecord::ManagedRecordAssociation
In the Role of: recordAssosciation
Multiplicity: 0..*
Description: The ManagedRecordAssociation to which the ManagedRecord belongs.

Aggregation

Associates ManagedRecords and the AuthenticationResults that apply to them.

From Class: Authenticity::AuthenticationResult
In the Role of: authenticationResults
Multiplicity: 0..*
Description: The result of applying a particular AuthenticationMethod to a ManagedRecord

To Class: ManagedRecord::ManagedRecord
In the Role of: authenticatedRecord
Multiplicity: 1
Description: The ManagedRecord to which an AuthenticationResult applies.

Association

Links a ManagedRecord to its Annotations through ManagedRecordAnnotation.

From Class: ManagedRecord::ManagedRecord
In the Role of: managedRecord
Multiplicity: 1
Description: The annotated ManagedRecord

To Class: Annotation::ManagedRecordAnnotation
In the Role of: annotationAssociation
Multiplicity: 0..*
Description: The associations of the ManagedRecord to its associations with Annotations.

Association

Collects the ManagedRecords under a RecordSet which is used for its Disposition. RecordSet's are determined by the ManagedRecord's RecordCategoryAssociation as documented in the Category Package and the Dispositions Package.

From Class: ManagedRecord::ManagedRecord
In the Role of: managedRecord
Multiplicity: 0..*
Description: The ManagedRecord assigned to a RecordSet

To Class: Dispositions::RecordSet
In the Role of: recordSet
Multiplicity: 1
Description: The RecordSet containing the ManagedRecord

Association

A ManagedRecord is associated with a RecordCategory through RecordCategoryAssociation. A ManagedRecord is associated with a single RecordCategory at a time. If the Category is not known at time of ManagedRecord set-aside, then it is assigned a default such as "Unknown" or "To Be Determined". The history of RecordCategoryAssociation's are kept through prev/next relationship on RecordCategoryAssociation. The current RecordCategoryAssociation is the latest assigned.

From Class: ManagedRecord::ManagedRecord
In the Role of: categorizedRecord
Multiplicity: 1
Description: The ManagedRecord being categorized.

To Class: Category::RecordCategoryAssociation
In the Role of: assignment
Multiplicity: 1..*
Description: The Category to which the ManagedRecord has been assigned through RecordCategoryAssociation.

Association

Collects the Documents (bit streams) that constitute the ManagedRecord. A Document can be included in more than one ManagedRecord. If a

ManagedRecord containing the Document is "destroyed", the Document will only be "destroyed" if it does not participate in any other ManagedRecord. This happens when a Document serves more than one business purpose or task the artifacts of which are being managed in the Records Management Environment.

From Class:	ManagedRecord::ManagedRecord
In the Role of:	managedRecord
Multiplicity:	1
Description:	The ManagedRecord

To Class:	ManagedRecord::RecordPart
In the Role of:	recordPart
Multiplicity:	1..*
Description:	One of the Documents (perhaps the only) that comprise the ManagedRecord.

8.5.5.3 Class: ManagedRecord: :CaseFileAction

Provides a record of the CaseFileAction's performed on a ManagedRecord that is a case file. Documenting who (via Role).

Attributes

Attribute: CaseFileAction.action

Type: CaseFileActionType
Description: The action taken on the RecordPart. Can be one of those enumerated by CaseFileActionType

Attribute: CaseFileAction.description

Type: string
Description: A description, if necessary describing the circumstances or details surrounding the action.

Attribute: CaseFileAction.date

Type: dateTime
Description: The date/time that the action was taken.

Connections

Aggregation

The history of actions taken on a case file.

From Class:	ManagedRecord: :CaseFileAction
In the Role of:	action
Multiplicity:	0..*

Description: An CaseFileAction performed on the ManagedRecord.

To Class: ManagedRecord: :ManagedRecord
In the Role of: caseFile
Multiplicity: 1
Description: The ManagedRecord whose history the CaseFileAction belongs.

Association

The actor that performed the recorded action

From Class: ManagedRecord: :CaseFileAction
In the Role of: action
Multiplicity: 0..*
Description: The action taken by the actor on the ManagedRecord.

To Class: Party::Role
In the Role of: actor
Multiplicity: 1
Description: The actor performing the action on the ManagedRecord.

Association

The RecordPart affected by a CaseFileAction.

From Class: ManagedRecord: :CaseFileAction
In the Role of: action
Multiplicity: 0..*
Description: The action taken on the case file record (i.e. ManagedRecord with isCaseFile=true).

To Class: ManagedRecord::CaseFilePart
In the Role of: part
Multiplicity: 1
Description: The RecordPart involved in the CaseFileAction

Association

The Authority under which the CaseFileAction was taken.

From Class: ManagedRecord: :CaseFileAction
In the Role of: action
Multiplicity: 0..*
Description: The authorized CaseFileAction.

To Class: RmsRole::Authority
In the Role of: authority

Multiplicity: 1
Description: The Authority under which the CaseFileAction was performed.

8.5.5.4 Enumeration: ManagedRecord: :CaseFileActionType

The possible actions that can be taken on a case file. These are represented as an enumeration of constant integers to identify the action.

Attributes

Attribute: CaseFileActionType.add
Type: string
Description: Add a RecordPart, associating a Document with the ManagedRecord.

Attribute: CaseFileActionType.append
Type: string
Description: Append a Document that is associated with a ManagedRecord through a RecordPart.

Attribute: CaseFileActionType.replace
Type: string
Description: The RecordPart is made to point to a different Document, effectively replacing it in the ManagedRecord.

Attribute: CaseFileActionType.remove
Type: string
Description: The Document is removed from the ManagedRecord.

Connections

8.5.5.5 Class: ManagedRecord: :CaseFileRecordDefinition

The definition of a ManagedRecord that is a case file, serving together with CaseFilePartDefinition as a template for creating a ManagedRecord as a case file.

Attributes

Attribute: CaseFileRecordDefinition.id
Type: ID
Description: Unique Identifier

Attribute: CaseFileRecordDefinition.type
Type: string
Description: The type of ManagedRecord

Attribute: CaseFileRecordDefinition.description

Type: integer
Description: A description of the CaseFileRecordDefinition

Attribute: CaseFileRecordDefinition.creationDate

Type: dateTime
Description: The date/time that the CaseFileRecordDefinition was created.

Connections

Association

CaseFileRecord Definition

From Class:	ManagedRecord: : ManagedRecord
In the Role of:	record
Multiplicity:	0..*
Description:	The defined ManagedRecord.
To Class:	ManagedRecord::CaseFileRecordDefinition
In the Role of:	definition
Multiplicity:	1
Description:	The definition of the structure of the case file.

Association

Creator of the CaseFileRecordDefinition

From Class:	ManagedRecord: :CaseFileRecordDefinition
In the Role of:	definition
Multiplicity:	1
Description:	The definition of the structure of the case file
To Class:	Party::Role
In the Role of:	creator
Multiplicity:	0..*
Description:	The creator of the CaseFileRecordDefinition

Association

The Authority for the CaseFileRecordDefinition

From Class:	ManagedRecord: :CaseFileRecordDefinition
In the Role of:	definition
Multiplicity:	0..*
Description:	The authorized CaseFileRecordDefinition.
To Class:	Party::Authority

In the Role of: authority
Multiplicity: 1
Description: The Authority authorizing the CaseFileRecordDefinition

Aggregation

Defined parts of a case file.

From Class: ManagedRecord: :CaseFilePartDefinition
In the Role of: partDefinition
Multiplicity: 1..*
Description: The definition of a RecordPart

To Class: CaseFile: :CaseFileRecordDefinition
In the Role of: recordDefinition
Multiplicity: 1
Description: Definition of a ManagedRecord

8.5.5.6 Class: ManagedRecord::ManagedRecordAssociation

An Association (group) in which a ManagedRecord can participate. The ManagedRecordAssociation may be ordered.

Attributes

Attribute: ManagedRecordAssociation.id
Type: ID
Description: Unique Identifier

Attribute: ManagedRecordAssociation.description
Type: string
Description: Textual description of the association (grouping) of ManagedRecords.

Attribute: ManagedRecordAssociation.orderedAssociation
Type: boolean
Description: When "True" it means that the ManagedRecordAssociation is ordered. The order of a particular ManagedRecord is designated by ManagedRecordAssociationMember.orderIndex

Connections

AssociationClass

Links ManagedRecord's into 0, 1 or more groupings (ManagedRecordAssociation's). ManagedRecordAssociation's may be empty, i.e., they are pre-defined and waiting for members to be assigned.

From Class:	ManagedRecord::ManagedRecord
In the Role of:	associatedRecord
Multiplicity:	0..*
Description:	The ManagedRecord in the ManagedRecordAssociation.

To Class:	ManagedRecord::ManagedRecordAssociation
In the Role of:	recordAssosciation
Multiplicity:	0..*
Description:	The ManagedRecordAssociation to which the ManagedRecord belongs.

8.5.5.7 AssociationClass: ManagedRecord::ManagedRecordAssociationMember

Records membership of a ManagedRecord in a ManagedRecordAssociation. It documents the role (if any) of the ManagedRecord membership in the ManagedRecordAssociation, and the order (if any) of the membership in the ManagedRecordAssociation.

Attributes

Attribute: ManagedRecordAssociationMember.associationDate

Type:	dateTime
Description:	The date/time that the ManagedRecord was added to the ManagedRecordAssociation.

Attribute: ManagedRecordAssociationMember.role

Type:	string
Description:	The role of the ManagedRecord in the ManagedRecordAssociation.

Attribute: ManagedRecordAssociationMember.orderIndex

Type:	integer
Description:	The order (if any) of the ManagedRecord in the ManagedRecordAssociation.

Connections

8.5.5.8 AssociationClass: ManagedRecord::ProvenanceAssociation

Documents the custodial organization responsible for the ManagedRecord. The initial organization is derived from the RecordCreator.

Attributes

Attribute: ProvenanceAssociation.associationDate

Type:	dateTime

Description: The date/time that the Provenance was assigned.

Connections

Generalization

From AssociationClass: ManagedRecord::ProvenanceAssociation

To Class: AttributeProfile::AttributableObject

Association

Provides a link to historical (if any) assignments of provenance.

From AssociationClass: ManagedRecord::ProvenanceAssociation
In the Role of: next
Multiplicity: 0..1
Description: The ProvenanceAssociation superceding this one. (If there is none, this is the current ProvenanceAssociation).

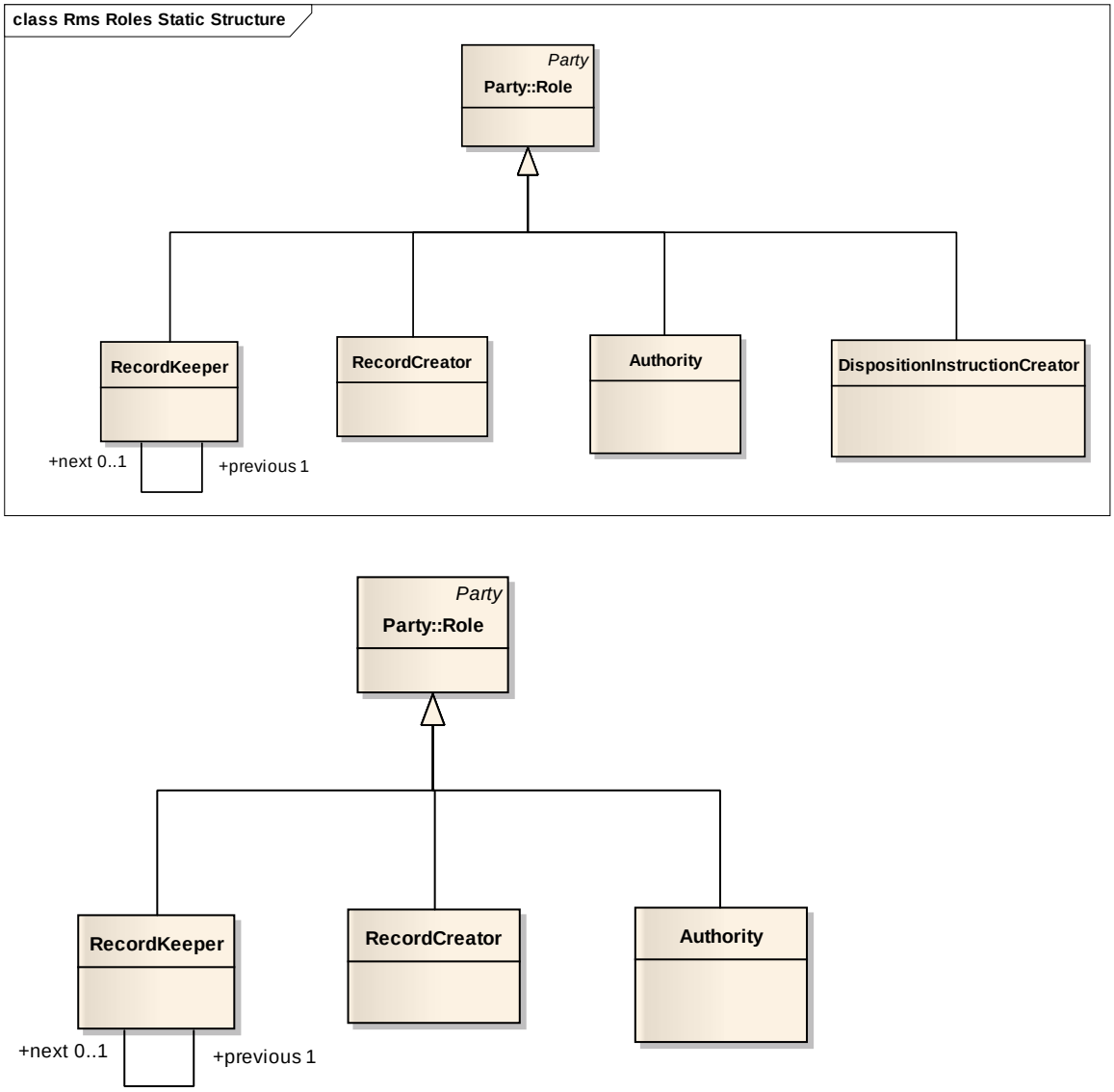
To AssociationClass: ManagedRecord::ProvenanceAssociation
In the Role of: previous
Multiplicity: 0..1
Description: The ProvenanceAssociation preceding this one.

8.5.6 Package: RmsRoles

The RmsRoles package extends the Party package to include the roles necessary for assigning responsibility of actions and custodianship in a records management environment. It is not intended to represent the overall organization structure.

A party model in a records management context is related to the organizational structure of the organization in which records are being managed, but is not identical to it. The purpose of the model is to be able to express Provenance and to identify the Roles in the organization that attribute aspects of the Managed Record.

RmsRoles Static Structure



The Party Model in this context is related to the organizational structure of the organization in which records are being managed, but is not identical to it. The purpose of the model is to be able to express Provenance and to identify the Roles in the organization that attribute aspects of the Managed Record.

Provenance is minimally specified by the top-level organization authorized as a valid organization for Provenance. If there is an organization that contains it, it is not instantiated in this model.

Provenance can be expressed in greater detail through a chain of suborganizations. However, at the discretion of the organization, some suborganizations are not expressed in the Provenance chain, in which case the Party's belonging to it are "collapsed" into its

containing suborganization. These "suppressed" organizations are not instantiated in this model.

Though related to the overall organizational structure, there is no requirement for it to be wholly consistent. The intent is to provide the designation of Provenance consistent with the business practices of the organization.

8.5.6.1 Class: RmsRoles::Authority

Authority's are used to cite the Authority through which an action or assignment was authorized. Used for such things as an Annotation denoting security classification, the Suspension or SuspensionRevocation of a record, etc.

Attributes

Connections

Association

Relates the CategorizationSchema to the Authority that has approved it.

From Class:	Category::CategorizationSchema
In the Role of:	recordSchedule
Multiplicity:	0..*
Description:	The approved CategorizationSchema

To Class:	RmsRoles::Authority
In the Role of:	authority
Multiplicity:	1
Description:	The Authority that approved the CategorizationSchema

Association

The Authority for the CaseFileRecordDefinition

From Class:	ManagedRecord::CaseFileRecordDefinition
In the Role of:	definition
Multiplicity:	0..*
Description:	The authorized CaseFileRecordDefinition.

To Class:	RmsRoles::Authority
In the Role of:	authority
Multiplicity:	1
Description:	The Authority authorizing the CaseFileRecordDefinition.

Association

Associates a DispositionInstruction with the Authority that approved it.

From Class: RmsRoles::Authority
In the Role of: dispositionAuthority
Multiplicity: 1
Description: The Authority that approved the DispositionInstruction.

To Class: Dispositions::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 0..*
Description: The DispositionInstruction approved by the Authority.

Association

The Authority under which the CaseFileAction was taken.

From Class: ManagedRecord::CaseFileAction
In the Role of: action
Multiplicity: 0..*
Description: The authorized CaseFileAction.

To Class: RmsRoles::Authority
In the Role of: authority
Multiplicity: 1
Description: The Authority under which the CaseFileAction was performed.

Association

Associates an Annotation with the Party responsible for authorizing it.

From Class: RmsRoles::Authority
In the Role of: authority
Multiplicity: 0..1
Description: The Authority for the Annotation

To Class: Annotation::Annotation
In the Role of: authorizedAnnotation
Multiplicity: 0..*
Description: The authorized Annotation.

Generalization

From Class: RmsRoles::Authority

To Class: Party::Role

AssociationClass

Records the authorized revocation of a particular DispositionSuspend.

From Class: RmsRoles::Authority
In the Role of: revocationAuthority
Multiplicity: 0..1
Description: The Authority authorizing the revocation of a DispositionSuspend.

To Class: Dispositions::DispositionSuspend
In the Role of: revokedSuspension
Multiplicity: 0..1
Description: A DispositionSuspend revoked by the Authority.

Association

Associates an SuspendEvent with the Authority which declared it.

From Class: Dispositions::SuspendEvent
In the Role of: suspendEvent
Multiplicity: 0..*
Description: The authorized SuspendEvent.

To Class: Party::Authority
In the Role of: authority
Multiplicity: 1
Description: The Authority declaring the SuspendEvent.

8.5.6.2 Class: RmsRoles::RecordCreator

Record Creator is a person, juridical person, organization or system (e.g., consolidation of real-time data into reports that are officially distributed.) However, the role must trace up to an organization (agency in the case of government)

Ultimately, only an agency (organization) has provenance. Provenance can be established "lower" in the organization as long as it ultimately resolves to the agency (organization) at the top-level.

Attributes

Connections

Association

Documents the Party which set-aside the ManagedRecord

From Class: ManagedRecord::ManagedRecord
In the Role of: setasideRecord
Multiplicity: 1..*
Description: The ManagedRecord.

To Class: RmsRoles::RecordCreator
In the Role of: creator
Multiplicity: 1
Description: The Party that set-aside (created) the ManagedRecord. This has nothing to do with who created the Documents that comprise the ManagedRecord.

Generalization

From Class: RmsRoles::RecordCreator

To Class: Party::Role

8.5.6.3 Class: RmsRoles::RecordKeeper

The administrative entity, unit, office, or person responsible for the custody and ongoing management of the records during their active business use.

Attributes

Attribute: RecordKeeper.assignmentDate

Type: dateTime
Description: The date/time that a RecordKeeper was assigned to one or more ManagedRecords.

Connections

Association

Documents the RecordSet's currently assigned RecordKeeper. The current RecordKeeper is identified by the latest assigned RecordKeeper

From Class: RmsRoles::RecordKeeper
In the Role of: theKeeper
Multiplicity: 0..*
Description: The ManagedRecord's assigned RecordKeeper

To Class: Dispositions::RecordSet
In the Role of: keeps
Multiplicity: 0..*
Description: The Party that serves as RecordKeeper for the ManagedRecord.

Association

Keeps the history of RecordKeeper's for specific RecordSets.

From Class: RmsRoles::RecordKeeper

In the Role of:	next
Multiplicity:	0..1
Description:	The next RecordKeeper (if there is no next, this is the current RecordKeeper.

To Class:	RmsRoles::RecordKeeper
In the Role of:	previous
Multiplicity:	1
Description:	The previous RecordKeeper.

Generalization

From Class:	RmsRoles::RecordKeeper

To Class:	Party::Role

8.5.6.4 Class: RmsRoles::DispositionInstructionCreator

DispositionInstructionCreator's are used to document the Party that created a Disposition Instruction.

Attributes

Connections

Association

Creator of the DispositionInstruction

From Class:	Dispositions::DispositionInstruction
In the Role of:	dispositionInstruction
Multiplicity:	0..*
Description:	The creator of the DispositionInstruction

To Class:	RmsRoles::DispositionInsructionCreator
In the Role of:	creator
Multiplicity:	1
Description:	The created DispositionInstruction

Generalization

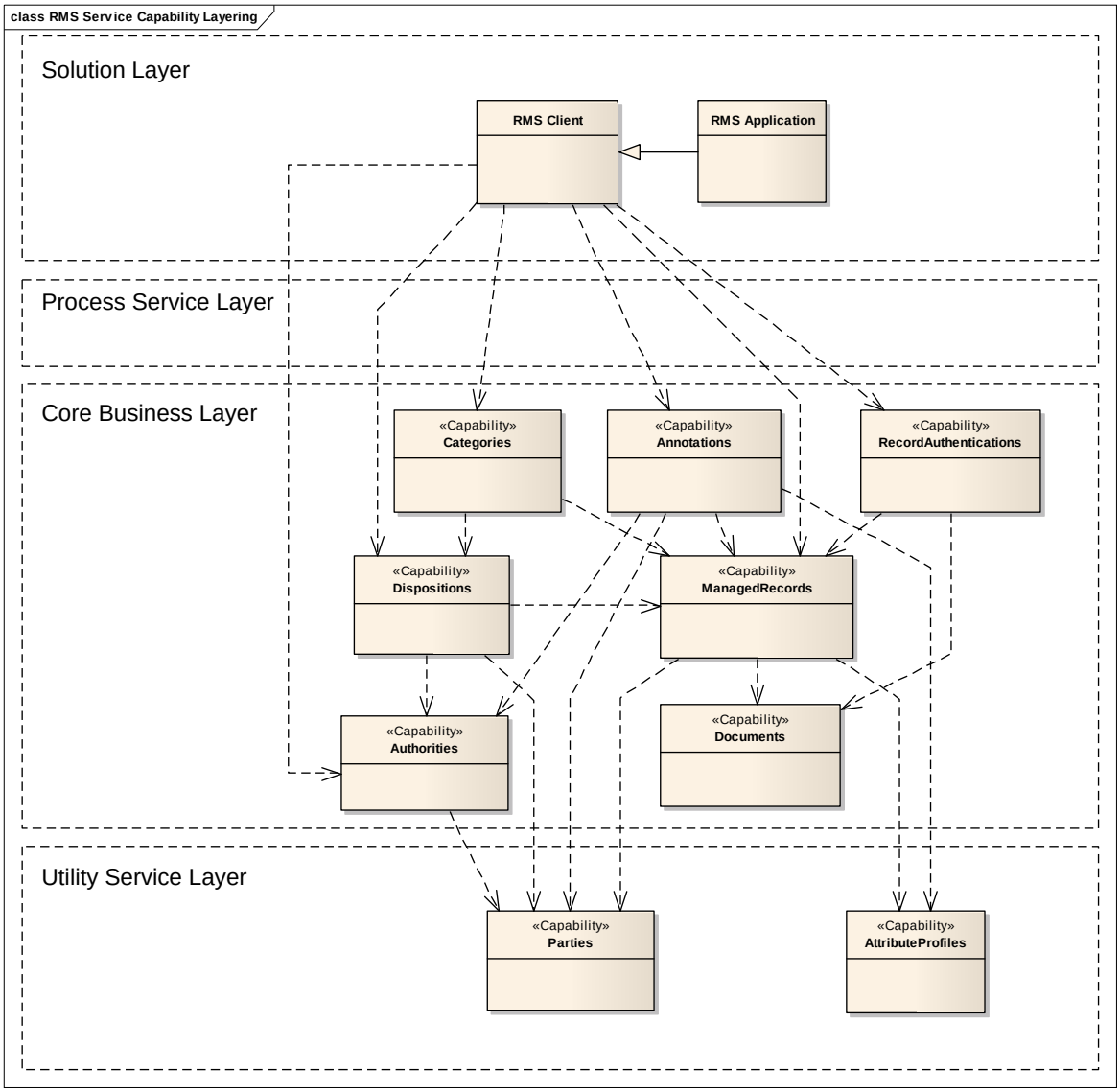
From Class:	RmsRoles::DispositionInstructionCreator

To Class:	Party::Role

8.6 Package: RmsServices

The RmsServices package contains subpackages for each service provided by RMS. RmsSolution maps the capabilities available to RMS clients. RmsProcessServices is a placeholder for future process oriented services (corresponds to the empty "process services layer" of the RMS Service Capability Layering diagram that opens the RmsServices section. The RmsCoreServices

RMS Service Capability Layering

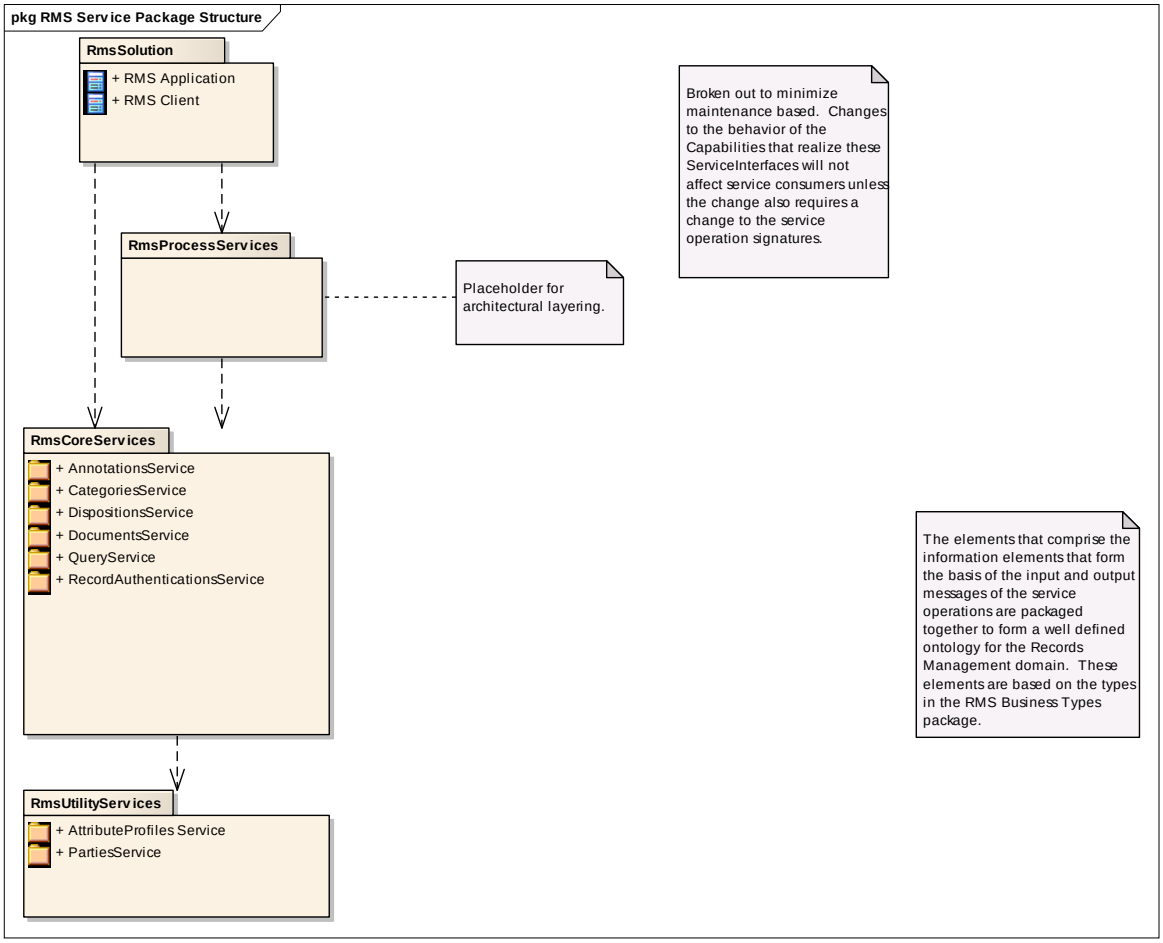


Services within the RmsServices package are layered into two primary layers: Core Business Layer and Utility Service Layer. The Core Business services represent the primary records management services while the Utility Layer holds services that support the Core Business Services.

The Process Service Layer is a placeholder for services that support records management processes that may be required in future versions.

The Solution Layer holds the end user solutions that will use the records management services.

RMS Service Package Structure

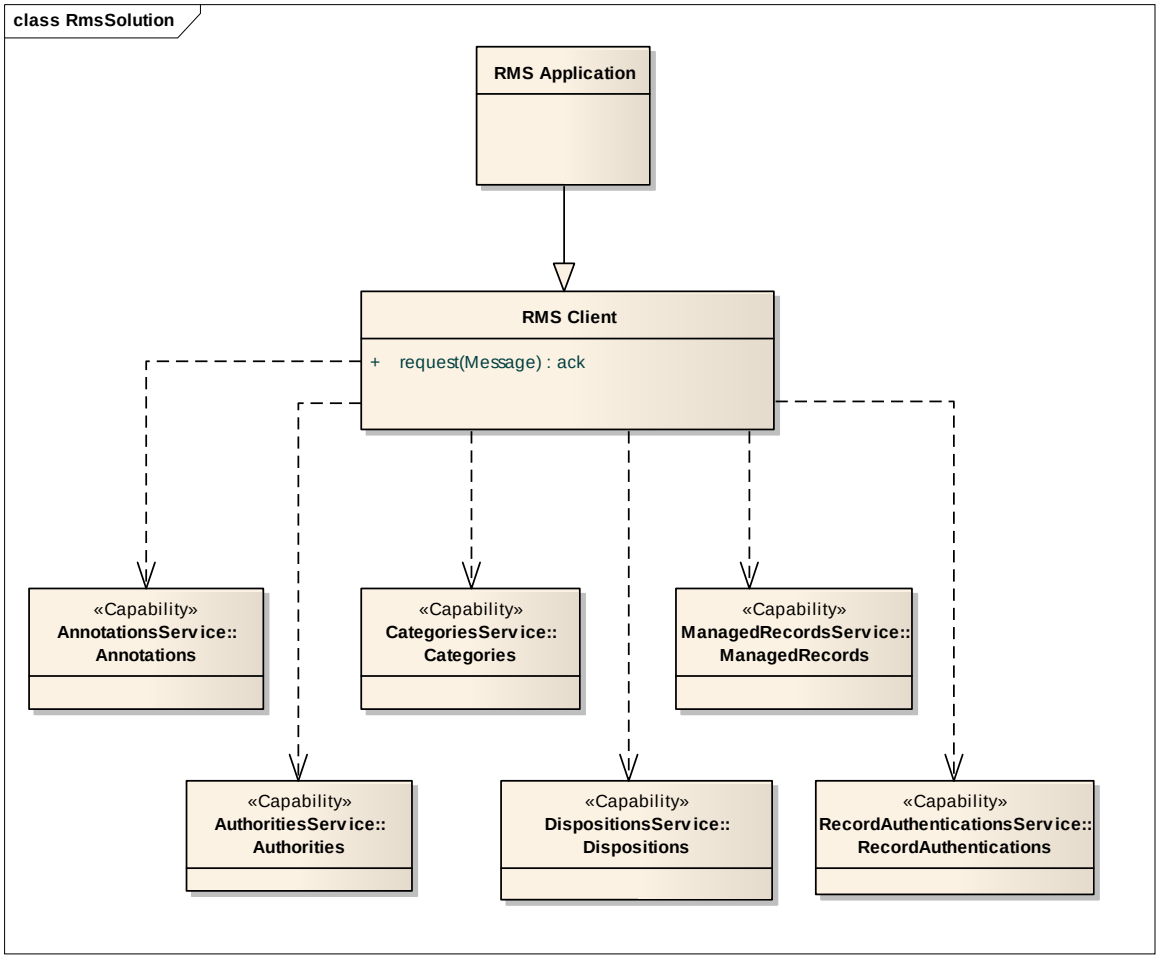


The Records Management Services have been layered based on the kind of functionality they provide. The layers include Solution, Process, Core, and Utility. Services in each layer have been placed into a corresponding package with dependencies between the packages as shown.

8.6.1 Package: RmsSolution

The RmsSolution package contains elements that represent clients of the RMS services. These are generally referred to as RMS Clients and RMS Applications.

RmsSolution



As shown, RMS Clients will use the capabilities provided by the Annotations, Authorities, Categories, Dispositions, ManagedRecords, and RecordAuthentications services. RMS Applications are simply a type of RMS Client that provide additional capabilities typically used by Records Managers that are outside the scope of this specification.

8.6.1.1 Class: RmsSolution::RMS Application

RMS Applications are simply a type of RMS Client that provide additional capabilities typically used by Records Managers that are outside the scope of this specification.

Attributes

Connections

Generalization

From Class: RmsSolution::RMS Application

To Class: RmsSolution::RMS Client

8.6.1.2 Class: RmsSolution::RMS Client

An RMS Client is any application that uses the RECORDS MANAGEMENT Services defined in this specification. These might be office automation tools like word processors, e-mail clients, or other business applications.

Attributes

Connections

Dependency

RMS Clients will use the RecordAuthentications service to manage authentication methods and results and to assess the authenticity of records within RMS.

From Class: RmsSolution::RMS Client

To Class: RecordAuthenticationsService::RecordAuthentications

Dependency

RMS Clients will use the Authorities service to manage information about the organizations that have authority for records within RMS.

From Class: RmsSolution::RMS Client

To Class: AuthoritiesService::Authorities

Generalization

From Class: RmsSolution::RMS Application

To Class: RmsSolution::RMS Client

Dependency

RMS Clients will use the ManagedRecords service to capture and maintain managed records within RMS.

From Class: RmsSolution::RMS Client

To Class: ManagedRecordsService::ManagedRecords

Dependency

RMS Clients will use the Annotations service to manage record annotations within RMS.

From Class: RmsSolution::RMS Client

To Class: AnnotationsService::Annotations

Dependency

RMS Clients will use the Dispositions service to manage information about disposition instructions, suspensions, and RecordSet's within RMS.

From Class: RmsSolution::RMS Client

To Class: DispositionsService::Dispositions

Dependency

RMS Clients will use the Categories service to manage information about the record schemas within RMS.

From Class: RmsSolution::RMS Client

To Class: CategoriesService::Categories

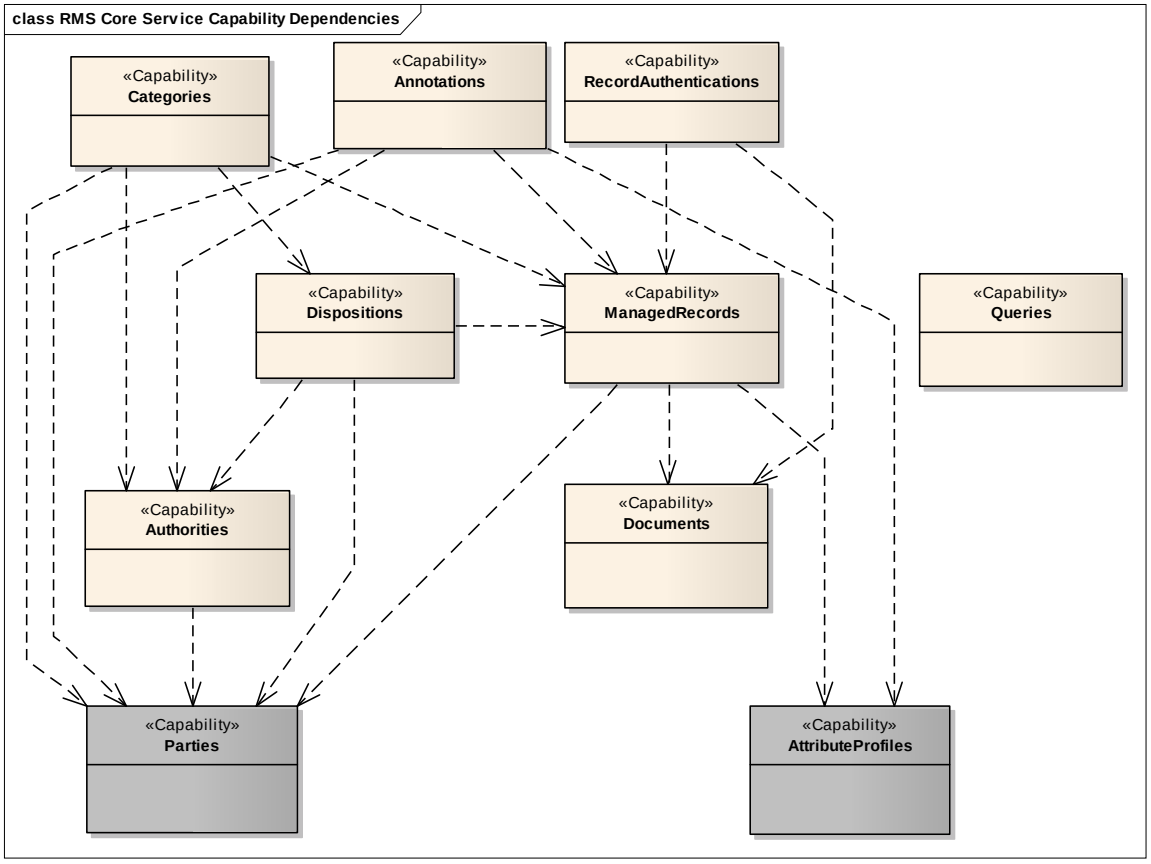
8.6.2 Package: RmsProcessServices

The RMS Process Services package is an architectural placeholder for process-related services. The current version of the specification is meant to be independent of business processes that generate records. Future versions of the specification may include process services that manage records management functions.

8.6.3 Package: RmsCoreServices

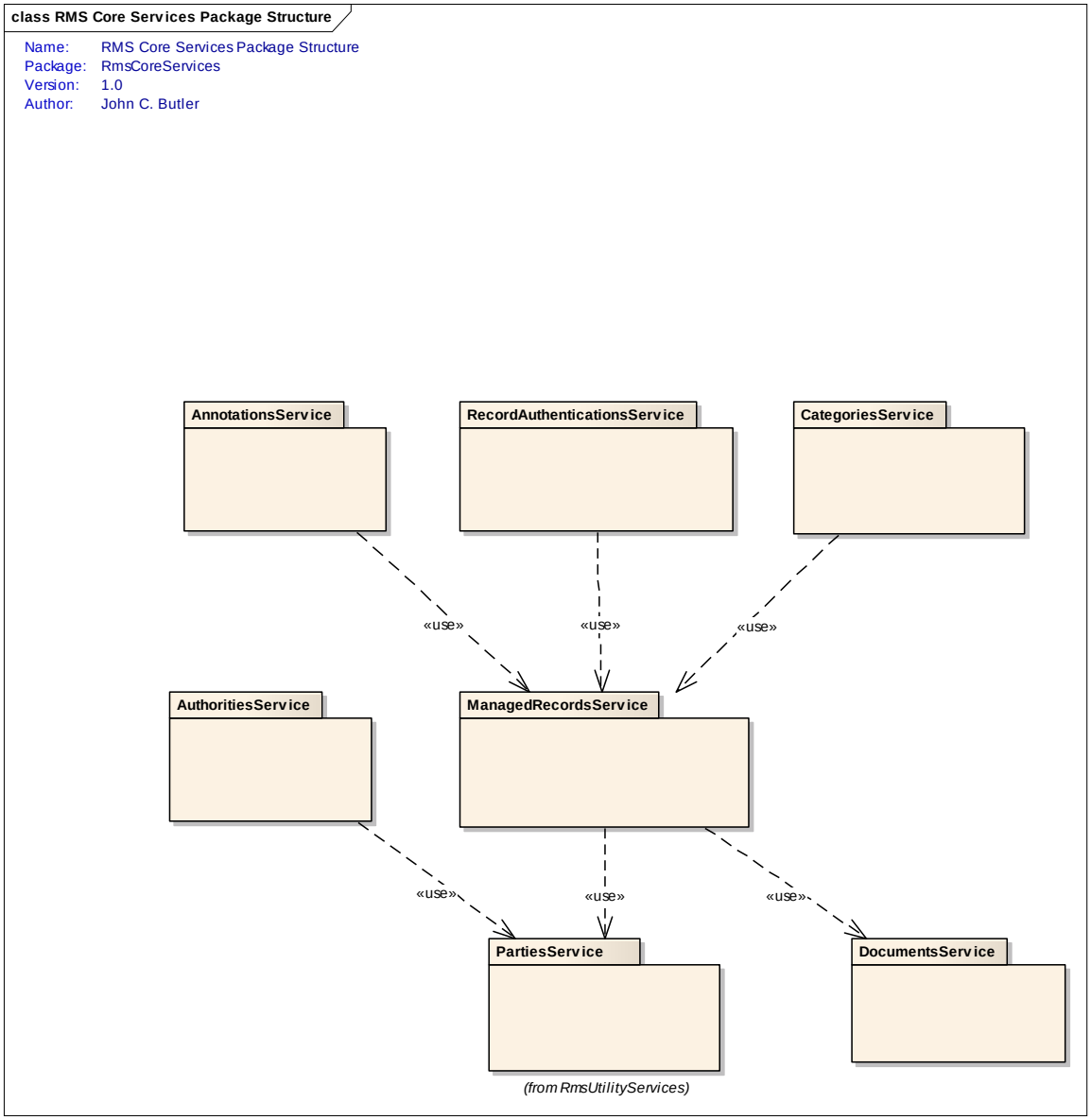
The RMS Core Services package contains the specifications of the RECORDS MANAGEMENT Services in the Core Business Layer of the RMS architecture.

RMS Core Service Capability Dependencies



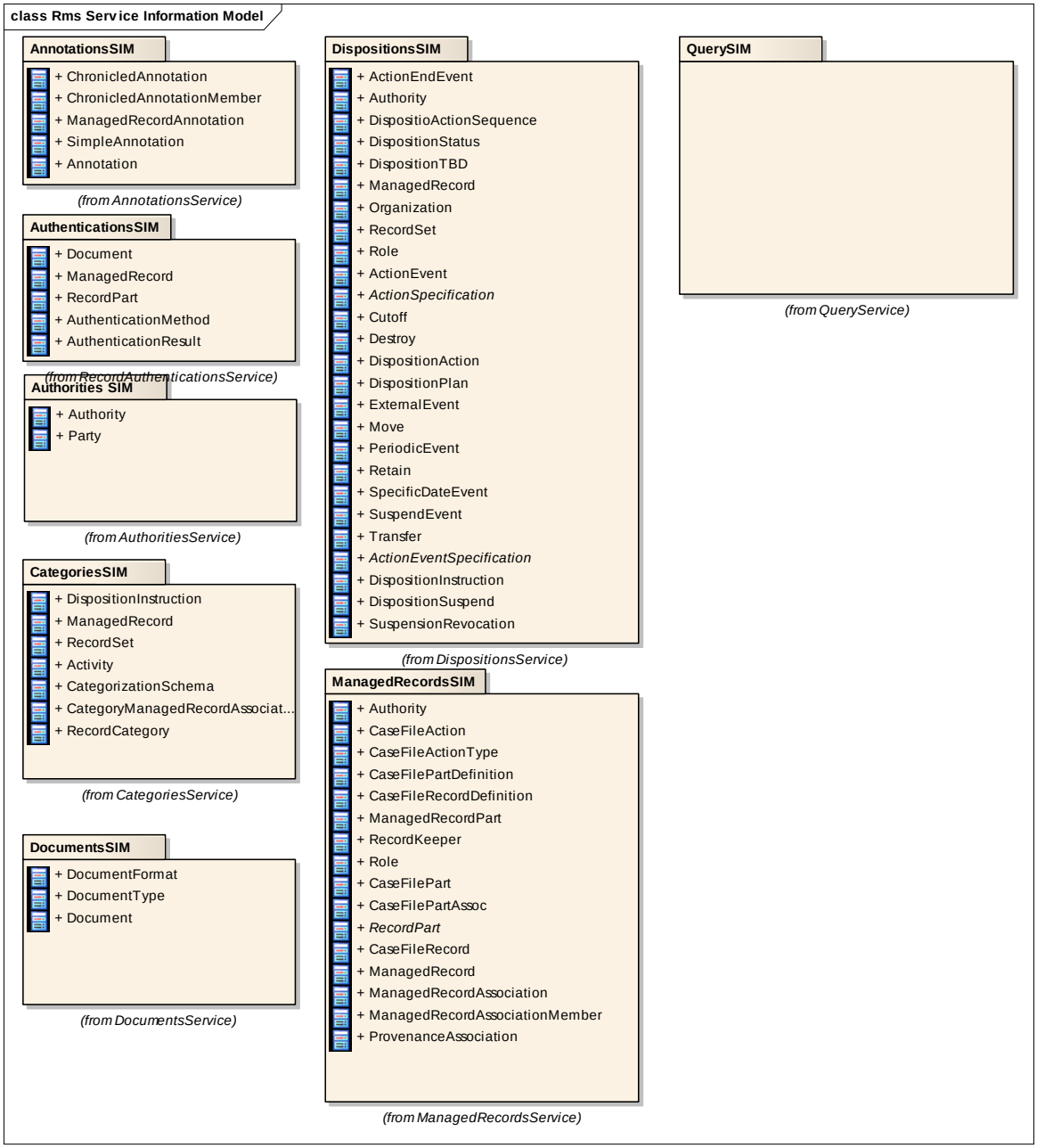
Though service specifications to not generally include dependencies of that service on others services, it is often important for a set of related services to use each other in well understood ways. The dependencies between the Service Capabilities in this diagram indicates the required constraints.

RMS Core Services Package Structure



This diagram illustrates the dependencies between the packages of the RMS Core Services.

Rms Service Information Model

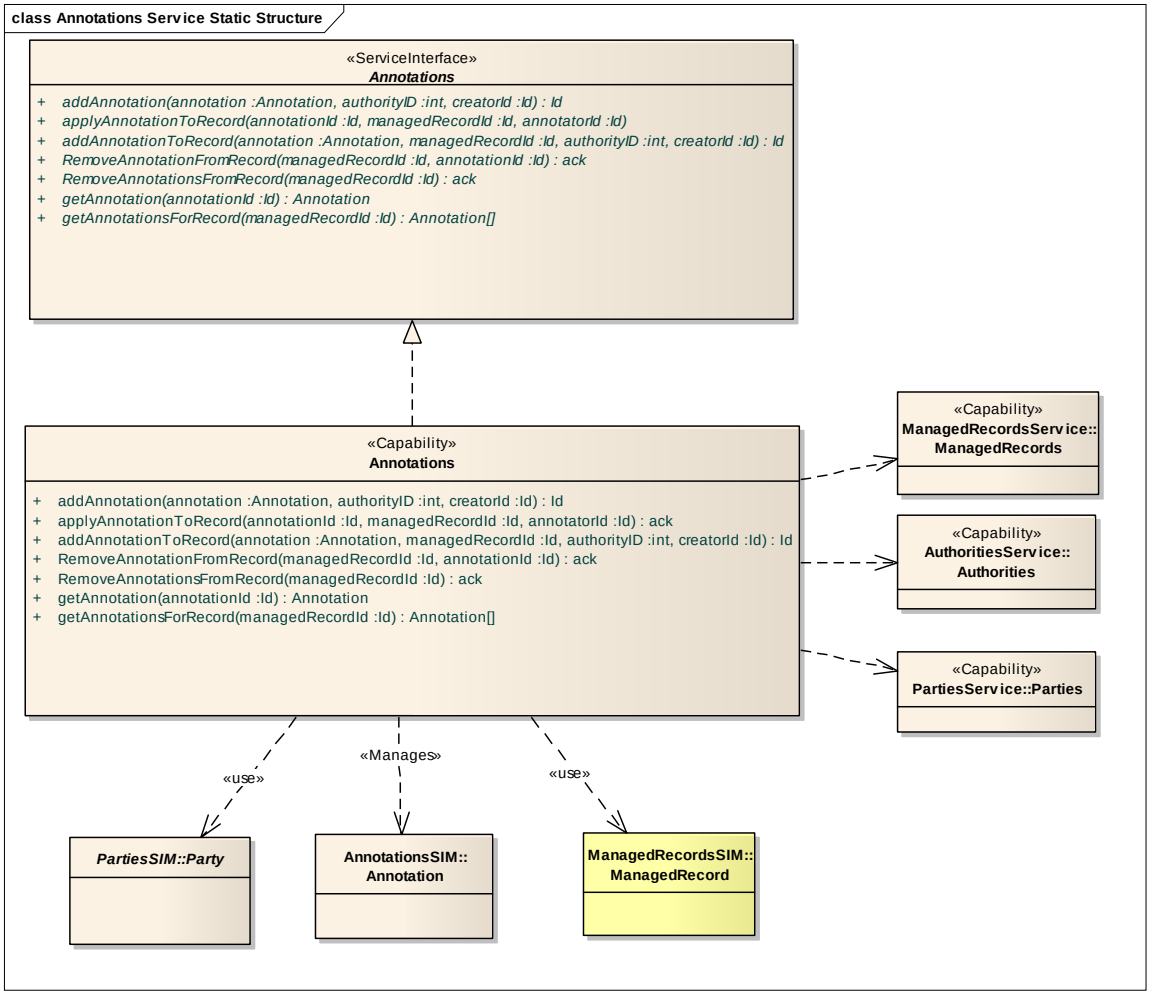


This diagram shows the various Service Information Model (SIM) packages that support the RMS Core Business Services. Each package is contained within the package that defines the associated service. They are included here to facilitate understanding of the information elements contained throughout these services.

8.6.3.1 Package: AnnotationsService

The AnnotationsService package contains the model elements that together define the Annotations service.

Annotations Service Static Structure



Implementations of the Annotations service will need to realize the Annotations ServiceInterface and all the behaviors associated with the Annotations Capability. This includes management of the Annotation instances on particular ManagedRecords. The Annotations service will depend on the Party service for the authority responsible for the Annotation and the ManagedRecords service for references to the ManagedRecord with which the Annotation is associated.

8.6.3.1.1 Class: AnnotationsService::Annotations

The Annotations Capably specifies the required behavior and constraints of any implementations of the Annotations ServiceInterface in a platform independent manner. Annotations are markings on records that help to differentiate them from other records in the same category or across categories. These are typically used to support business needs for special handling or management of the record.

Attributes

Connections

Dependency

The Annotations service maintains references to the parties that create annotations and apply them to records.

From Class: AnnotationsService::Annotations

To Class: PartiesService::Parties

Dependency

The Annotations service manages information related to annotations.

From Class: AnnotationsService::Annotations

To Class: AnnotationsSIM::Annotation

Realisation

From Class: AnnotationsService::Annotations

To Interface: AnnotationsService::Annotations

Dependency

RMS Clients will use the Annotations service to manage record annotations within RMS.

From Class: RmsSolution::RMS Client

To Class: AnnotationsService::Annotations

Dependency

The Annotations service maintains references to information related to managed records.

From Class: AnnotationsService::Annotations

To Class: ManagedRecordsSIM::ManagedRecord

Dependency

From Class: AnnotationsService::Annotations

To Class: AttributeProfiles Service::AttributeProfiles

Dependency

The Annotations service maintains reference information related to parties associated with annotations.

From Class: AnnotationsService::Annotations

To Class: PartiesSIM::Party

Dependency

The Annotations service maintains references to the records being annotated.

From Class: AnnotationsService::Annotations

To Class: ManagedRecordsService::ManagedRecords

Dependency

The Annotations service maintains references to the Authorities for the annotations.

From Class: AnnotationsService::Annotations

To Class: AuthoritiesService::Authorities

8.6.3.1.2 Interface: AnnotationsService::Annotations

The Annotations ServiceInterface is a platform independent specification of the operation signatures of the Annotations service. Refer to the Annotations Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: AnnotationsService::Annotations

To Interface: AnnotationsService::Annotations

8.6.3.1.3 Capabilities: AnnotationsService::Annotations

addAnnotation: Id

Scope Public

Description This operation provides the ability to add an annotation to a managed record. It requires the ID of the managed record as well as the Authority from which the annotation is coming. Parameters include the annotation to be added, the Id of the authority of the annotation and the Id of the creator of the annotation. The operation returns the Id of the annotation that was added.

Parameters annotation: Annotation {in}
authorityID: int {in}
creatorId: Id {in}

addAnnotationToRecord: Id

Scope Public

Description This operation allows a new Annotation to be added to a particular ManagedRecord. Parameters include the annotation to be added, the Id of the managed record, the Id of the authority for the annotation, and the Id of the creator of the annotation. The operation returns an acknowledgement indicating success or failure.

Parameters annotation: Annotation {in}
managedRecordId: Id {in}
authorityID: int {in}
creatorId: Id {in}

applyAnnotationToRecord: ack

Scope Public

Description This operation allows a particular Annotation to be applied to a particular ManagedRecord. Parameters include the Id of the annotation being added to the ManagedRecord, the Id of the managed record, and the Id of the party annotating the managed record. The operation returns an acknowledgement indicating success or failure.

Parameters annotationId: Id {in}
managedRecordId: Id {in}
annotatorId: Id {in}

getAnnotation: Annotation

Scope	Public
Description	This operation returns the annotation with a given Id. The only parameter is the Id of the annotation to be returned.
Parameters	annotationId: Id {in}

getAnnotationsForRecord: Annotation

Scope	Public
Description	The getAnnotationsforRecord operation returns a list of the annotations for a record with a given ID. The only parameter is the Id of the managed record.
Parameters	managedRecordId: Id {in}

RemoveAnnotationFromRecord: ack

Scope	Public
Description	This operation provides the ability to remove an annotation from a managed record. Parameters include the Id of the ManagedRecord and the Id of the annotation to be removed. The operation returns an acknowledgement indicating success or failure.
Parameters	managedRecordId: Id {in} annotationId: Id {in}

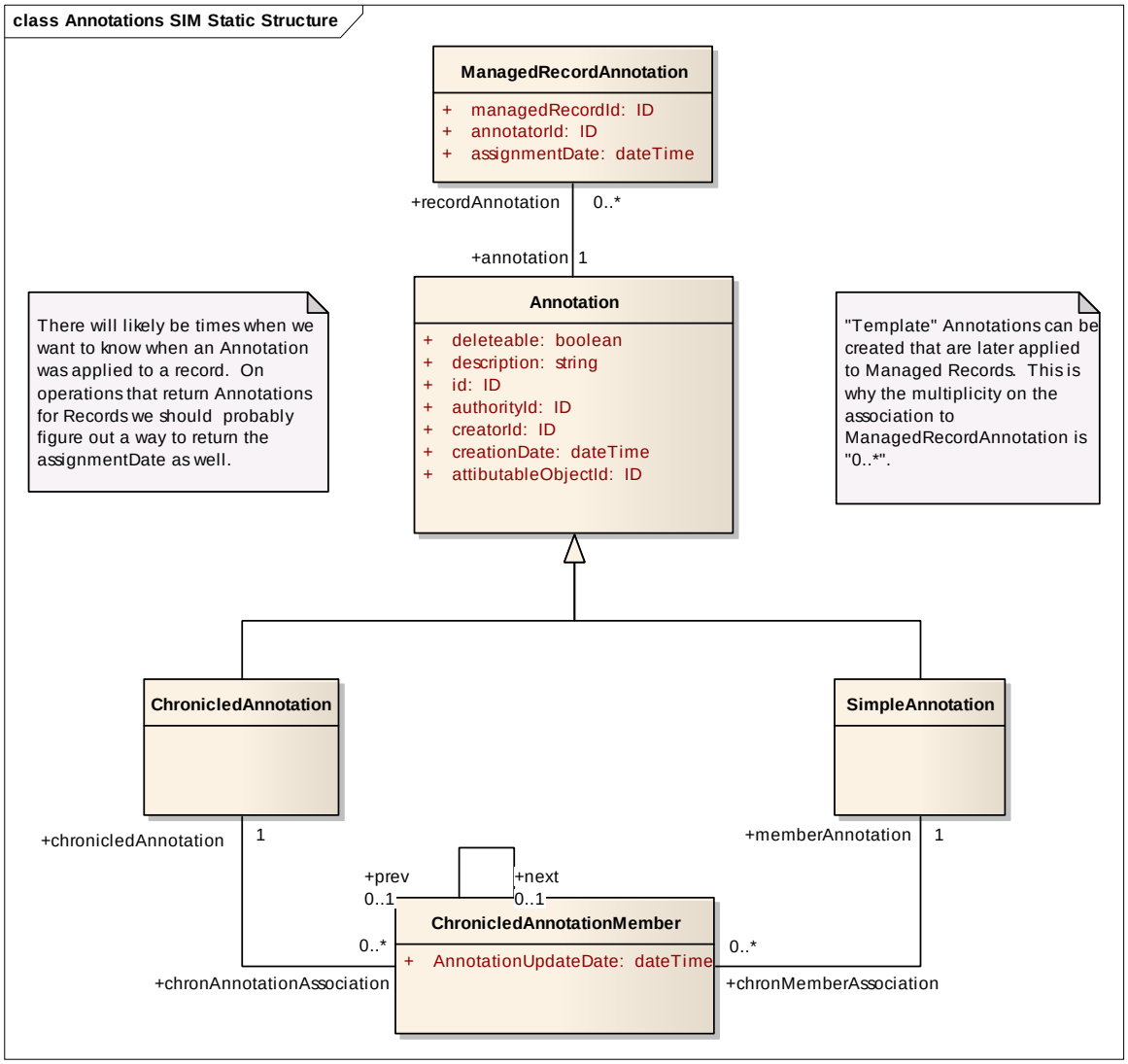
RemoveAnnotationsFromRecord: ack

Scope	Public
Description	This operation provides the ability to remove all annotations from a particular managed record. The only parameter is the Id of the managed record. The operation returns an acknowledgement indicating success or failure.
Parameters	managedRecordId: Id {in}

8.6.3.1.4 Package: AnnotationsSIM

The AnnotationsSIM package contains the elements used to define the parameters and information management responsibilities of the Annotations service.

Annotations SIM Static Structure



The Annotations SIM Static Structure diagram shows the information elements that comprise parameters or information that the Annotations service must manage.

There will likely be times when we want to know when an Annotation was applied to a record. On operations that return Annotations for Records we should probably figure out a way to return the assignmentDate as well.

"Template" Annotations can be created that are later applied to Managed Records. This is why the multiplicity on the association to ManagedRecordAnnotation is "0..*".

8.6.3.1.4.1 Class: AnnotationsSIM::ChronicledAnnotation

See the Annotation package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Association

See the Annotation package of the RmsDomainModel package for a definition of this association.

From Class: AnnotationsSIM::ChronicledAnnotation
In the Role of: chronicledAnnotation
Multiplicity: 1

To Class: AnnotationsSIM::ChronicledAnnotationMember
In the Role of: chronAnnotationAssociation
Multiplicity: 0..*

Generalization

From Class: AnnotationsSIM::ChronicledAnnotation

To Class: AnnotationsSIM::Annotation

8.6.3.1.4.2 Class: AnnotationsSIM::ChronicledAnnotationMember

See the Annotation package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: ChronicledAnnotationMember.AnnotationUpdateDate

Type: dateTime
Description: See the Annotation package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Annotation package of the RmsDomainModel package for a definition of this association.

From Class: AnnotationsSIM::SimpleAnnotation
In the Role of: memberAnnotation
Multiplicity: 1

To Class: AnnotationsSIM::ChronicledAnnotationMember
In the Role of: chronMemberAssociation
Multiplicity: 0..*

Association

See the Annotation package of the RmsDomainModel package for a definition of this association.

From Class: AnnotationsSIM::ChronicledAnnotation
In the Role of: chronicledAnnotation
Multiplicity: 1

To Class: AnnotationsSIM::ChronicledAnnotationMember
In the Role of: chronAnnotationAssociation
Multiplicity: 0..*

Association

See the Annotation package of the RmsDomainModel package for a definition of this association.

From Class: AnnotationsSIM::ChronicledAnnotationMember
In the Role of: next
Multiplicity: 0..1

To Class: AnnotationsSIM::ChronicledAnnotationMember
In the Role of: prev
Multiplicity: 0..1

8.6.3.1.4.3 Class: AnnotationsSIM::ManagedRecordAnnotation

See the Annotation package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: ManagedRecordAnnotation.managedRecordId
Type: ID
Description: The Id of the ManagedRecord to which this Annotation is attached.

Attribute: ManagedRecordAnnotation.annotatorId
Type: ID
Description: A unique identifier for the Party that applied the Annotation to the ManagedRecord.

Attribute: ManagedRecordAnnotation.assignmentDate
Type: dateTime

Description: See the Annotation package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Annotation package of the RmsDomainModel package for a definition of this association.

From Class: AnnotationsSIM::Annotation
In the Role of: annotation
Multiplicity: 1

To Class: AnnotationsSIM::ManagedRecordAnnotation
In the Role of: recordAnnotation
Multiplicity: 0..*

8.6.3.1.4.4 Class: AnnotationsSIM::SimpleAnnotation

See the Annotation package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Association

See the Annotation package of the RmsDomainModel package for a definition of this association.

From Class: AnnotationsSIM::SimpleAnnotation
In the Role of: memberAnnotation
Multiplicity: 1

To Class: AnnotationsSIM::ChronicledAnnotationMember
In the Role of: chronMemberAssociation
Multiplicity: 0..*

Generalization

From Class: AnnotationsSIM::SimpleAnnotation

To Class: AnnotationsSIM::Annotation

8.6.3.1.4.5 Class: AnnotationsSIM::Annotation

See the Annotation package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Annotation.deleteable

Type: boolean
Description: See the Annotation package of the RmsDomainModel for a definition of this element.

Attribute: Annotation.description

Type: string
Description: See the Annotation package of the RmsDomainModel for a definition of this element.

Attribute: Annotation.id

Type: ID
Description: See the Annotation package of the RmsDomainModel for a definition of this element.

Attribute: Annotation.authorityId

Type: ID
Description: A reference to the Id of the Party that is responsible for this Annotation. This field is not required if the Annotation was create not by a Party but by legislation or some other regulation.

Attribute: Annotation.creatorId

Type: ID
Description: A unique identifier for the Party that created the Annotation.

Attribute: Annotation.creationDate

Type: dateTime
Description: See the Annotation package of the RmsDomainModel for a definition of this element.

Attribute: Annotation.attibutableObjectId

Type: ID
Description: A reference to the Id of the AttributableObject that stores AtributeProfile information for the Annotation.

Connections

Association

See the Annotation package of the RmsDomainModel package for a definition of this association.

From Class: AnnotationsSIM::Annotation
In the Role of: annotation
Multiplicity: 1

To Class: AnnotationsSIM::ManagedRecordAnnotation
In the Role of: recordAnnotation
Multiplicity: 0..*

Dependency

The Annotations service manages information related to annotations.

From Class: AnnotationsService::Annotations

To Class: AnnotationsSIM::Annotation

Generalization

From Class: AnnotationsSIM::SimpleAnnotation

To Class: AnnotationsSIM::Annotation

Generalization

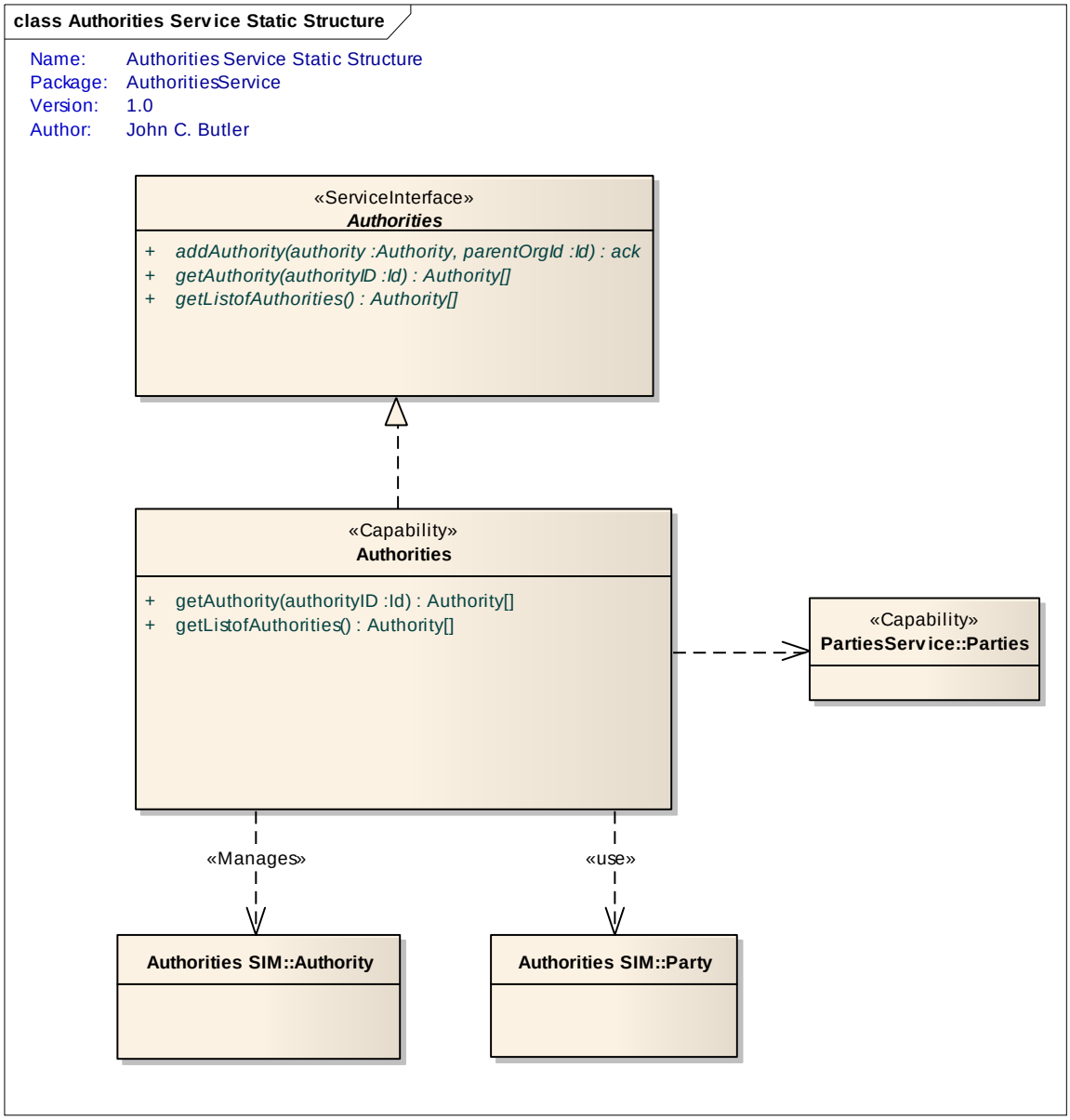
From Class: AnnotationsSIM::ChronicledAnnotation

To Class: AnnotationsSIM::Annotation

8.6.3.2 Package: AuthoritiesService

The AuthoritiesService package contains the model elements that together define the Authorities service.

Authorities Service Static Structure



Implementations of the Authorizations service will need to realize the Authorizations ServiceInterface and all the behaviors associated with the Authorizations Capability. This includes management of the Authorizations instances on particular ManagedRecords. The Annotations service will depend on the Party service for the authority responsible for the Annotation and the ManagedRecords service for references to the ManagedRecord with which the Annotation is associated.

8.6.3.2.1 Class: AuthoritiesService::Authorities

The Authorities Capability represents a specification of the required functionality of the Authorities service.

Attributes

Connections

Dependency

The Categories service maintains a reference to information in the Authorities service related to the authority for a categorization schema.

From Class: CategoriesService::Categories

To Class: AuthoritiesService::Authorities

Realisation

From Class: AuthoritiesService::Authorities

To Interface: AuthoritiesService::Authorities

Dependency

RMS Clients will use the Authorities service to manage information about the organizations that have authority for records within RMS.

From Class: RmsSolution::RMS Client

To Class: AuthoritiesService::Authorities

Dependency

The Authorities service uses the Parties service to managed the data about the Authority.

From Class: AuthoritiesService::Authorities

To Class: PartiesService::Parties

Dependency

From Class: AuthoritiesService::Authorities

To Class: Authorities SIM::Party

Dependency

The Authorities service manages information related to Authority for RMS elements.

From Class: AuthoritiesService::Authorities

To Class: Authorities SIM::Authority

Dependency

The Dispositions service maintains references to the Authorities for the disposition instructions.

From Class: DispositionsService::Dispositions

To Class: AuthoritiesService::Authorities

Dependency

The Annotations service maintains references to the Authorities for the annotations.

From Class: AnnotationsService::Annotations

To Class: AuthoritiesService::Authorities

8.6.3.2.2 Interface: AuthoritiesService::Authorities

The Authorities ServiceInterface is a platform independent specification of the operation signatures of the Authorities service. Refer to the Authorities Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: AuthoritiesService::Authorities

To Interface: AuthoritiesService::Authorities

8.6.3.2.3 Capabilities: Authorities Service::Authorities

getAuthority: Authority

Scope Public

Description The getAuthory operation returns the Authority with the given Id.

Parameters authorityID: Id {in}

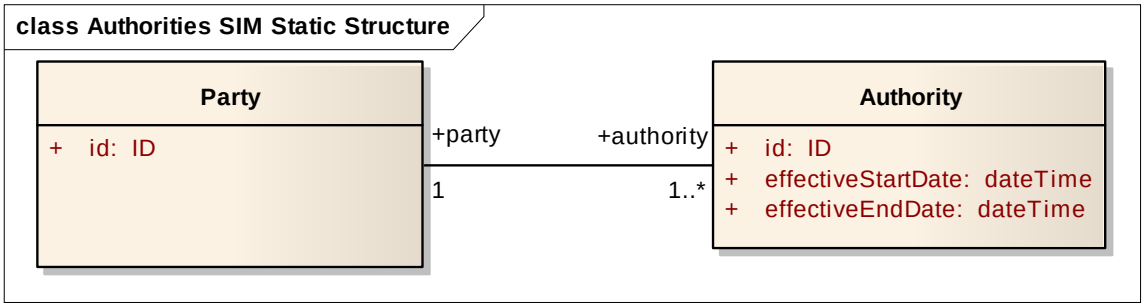
getListofAuthorities: Authority

Scope	Public
Description	The getListOfAhothorities operation returns a complete list of the Authorities within the system.

8.6.3.2.4 Parameters Package: Authorities SIM

The Authoress package contains the elements used to define the parameters and information management responsibilities of the Authorities service.

Authorities SIM Static Structure



The Authorities SIM Static Structure diagram shows the information elements that comprise parameters or information that the Authorities service must manage.

8.6.3.2.4.1 Class: Authorities SIM::Authority

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Authority.id
Type: ID
Description: A unique identifier for the Authority.

Attribute: Authority.effectiveStartDate
Type: dateTime
Description: The date upon which the Authority takes effect.

Attribute: Authority.effectiveEndDate
Type: dateTime
Description: The date upon which the Authority ends.

Connections

Association

An association between an Authority for a particular records management element and the party that has that responsibility.

From Class: Authorities SIM::Authority
In the Role of: authority
Multiplicity: 1..*

To Class: Authorities SIM::Party
In the Role of: party
Multiplicity: 1

Dependency

The Authorities service manages information related to Authority for RMS elements.

From Class: AuthoritiesService::Authorities

To Class: Authorities SIM::Authority

8.6.3.2.4.2 Class: Authorities SIM::Party

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Party.id

Type: ID
Description: See the Party package of the RmsDomainModel for a definition of this element.

Connections

Association

An association between an Authority for a particular records management element and the party that has that responsibility.

From Class: Authorities SIM::Authority
In the Role of: authority
Multiplicity: 1..*

To Class: Authorities SIM::Party
In the Role of: party
Multiplicity: 1

Dependency

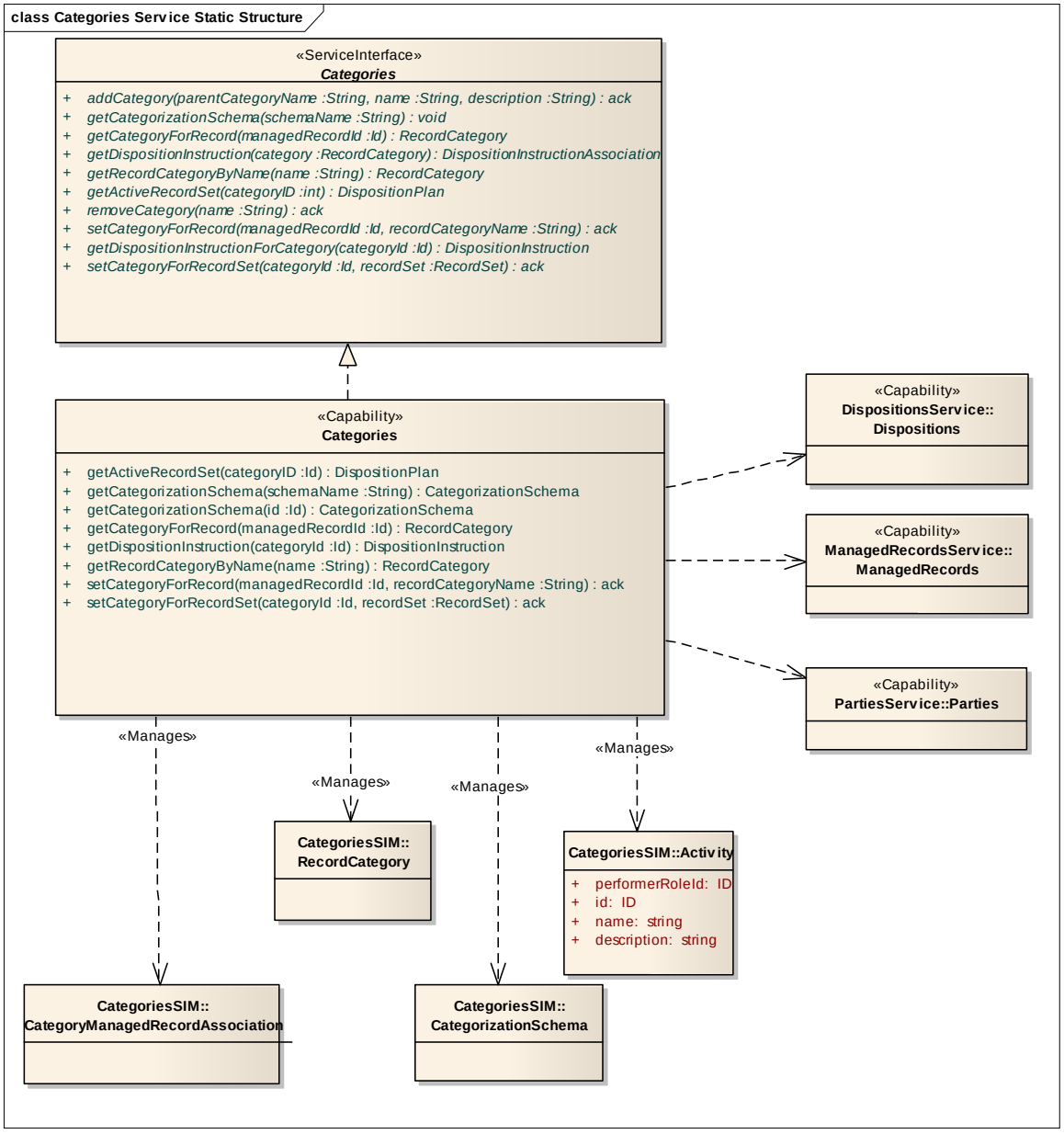
From Class: AuthoritiesService::Authorities

To Class: Authorities SIM::Party

8.6.3.3 Package: CategoriesService

The CategoriesService package contains the model elements that together define the Categories service.

Categories Service Static Structure



Implementations of the Categories service will need to realize the Categories ServiceInterface and all the behaviors associated with the Categories Capability. This

includes providing access to record schedules captured in the model as CategorizationSchema's. It is assumed that the actual setup of the schemas is outside the scope of this specification. The functionality provided herein simply allows for the use of the schema and application of its categories to ManagedRecords.

8.6.3.3.1 Class: CategoriesService::Categories

The Categories Capably specifies the required behavior and constraints of any implementations of the Categories ServiceInterface in a platform independent manner. It provides the ability to assign ManagedRecords to RecordCategory's either individually or as a set. It also tracks the DispositionInstruction assigned to a RecordCategory. It does not provide the ability to create CategorizationSchema's or the RecordCategory's therein.

Attributes

Connections

Dependency

The Categories service maintains a reference to information in the Authorities service related to the authority for a categorization schema.

From Class: CategoriesService::Categories

To Class: AuthoritiesService::Authorities

Dependency

The Categories service maintains references to managed records and the categories they fall into.

From Class: CategoriesService::Categories

To Class: ManagedRecordsService::ManagedRecords

Dependency

The Categories service maintains references to information managed in by the Parties for the role that performs a particular business activity that generated a record.

From Class: CategoriesService::Categories

To Class: PartiesService::Parties

Realisation

From Class: CategoriesService::Categories

To Interface: CategoriesService::Categories

Dependency

The Categories service manages information about the business activities and the categorization of the records produced by those activities.

From Class: CategoriesService::Categories

To Class: CategoriesSIM::Activity

Dependency

The Categories service manages information about records associated with a particular category in a schema.

From Class: CategoriesService::Categories

To Class: CategoriesSIM::CategoryManagedRecordAssociation

Dependency

The Categories service maintains references to the disposition instructions that apply to record categories.

From Class: CategoriesService::Categories

To Class: DispositionsService::Dispositions

Dependency

RMS Clients will use the Categories service to manage information about the record schemas within RMS.

From Class: RmsSolution::RMS Client

To Class: CategoriesService::Categories

Dependency

The Categories service manages information about record category schemas.

From Class: CategoriesService::Categories

To Class: CategoriesSIM::CategorizationSchema

Dependency

The Categories service manages information about record categories in a schema.

From Class: CategoriesService::Categories

To Class: CategoriesSIM::RecordCategory

8.6.3.3.2 Interface: CategoriesService::Categories

The Categories ServiceInterface is a platform independent specification of the operation signatures of the Categories service. Refer to the Categories Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: CategoriesService::Categories

To Interface: CategoriesService::Categories

8.6.3.3.3 Capabilities: CategoriesService::Categories

getActiveRecordSet: DispositionPlan

Scope Public

Description This operation returns the active RecordSet for the RecordCategory if one exists. The only parameter is the category Id.

Parameters categoryID: Id {in}

getCategorizationSchema: CategorizationSchema

Scope Public

Description This operation returns the CategorizationSchema with the given name.

Parameters schemaName: String {in}

getCategorizationSchema: CategorizationSchema

Scope	Public
Description	This operation returns the CategorizationSchema with the given Id.
Parameters	id: Id {in}

getCategoryForRecord: RecordCategory

Scope	Public
Description	This operation returns the RecordCategory for a particular ManagedRecord.
Parameters	managedRecordId: Id {in}

getDispositionInstruction: DispositionInstruction

Scope	Public
Description	This operation returns the current DispositionInstruction associated with the RecordCategory specified by the categoryId.
Parameters	categoryId: Id {in}

getRecordCategoryByName: RecordCategory

Scope	Public
Description	This operation returns the RecordCategory with the given name.
Parameters	name: String {in}

setCategoryForRecord: ack

Scope	Public

Description This operation assigns a ManagedRecord to a RecordCategory. Must notify the Dispositions service to add the record to the appropriate RecordSet. A ManagedRecord can be in only one RecordCategory at a time. The operation returns an acknowledgement indicating success or failure.

Parameters managedRecordId: Id {in}
recordCategoryName: String {in}

setCategoryForRecordSet: ack

Scope Public

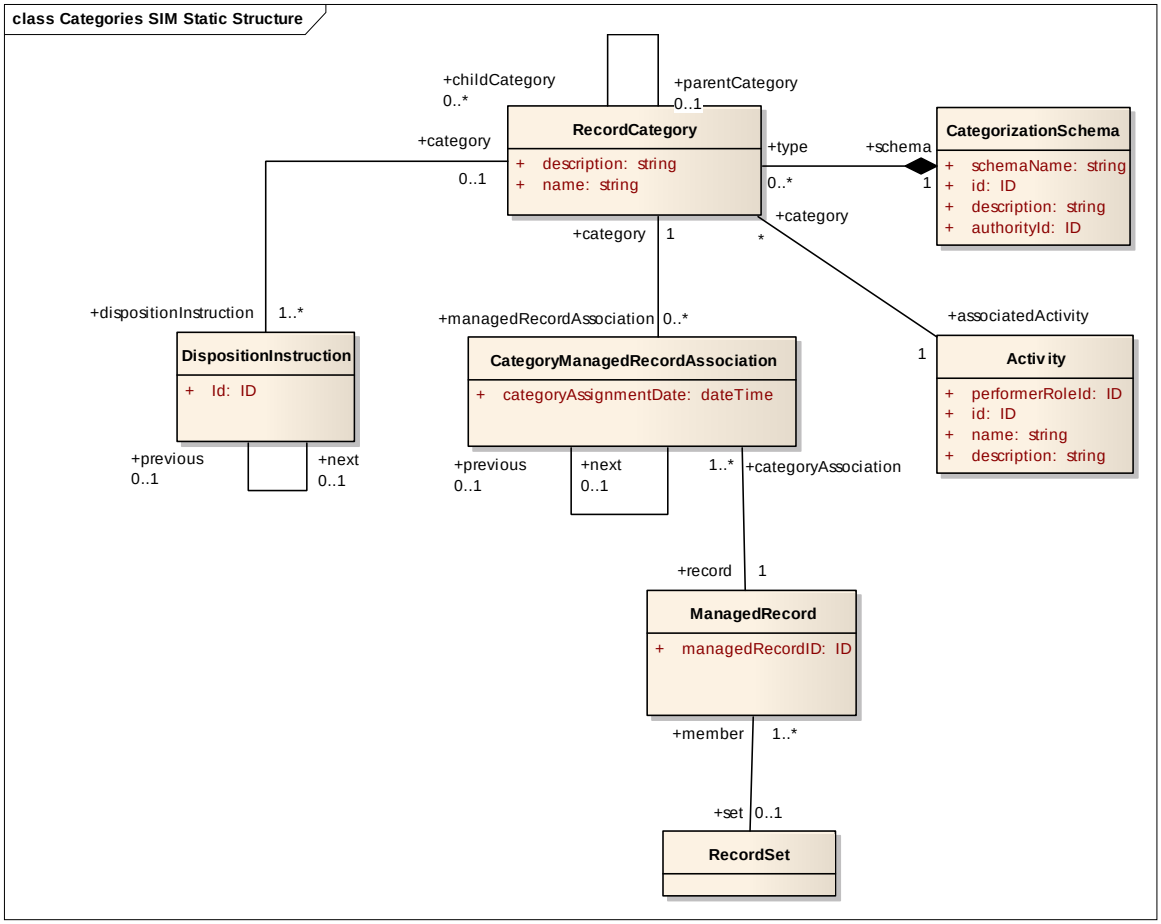
Description This operation sets all ManagedRecords in the RecordSet to the RecordCategory with the given Id. The operation returns an acknowledgement indicating success or failure.

Parameters categoryId: Id {in}

8.6.3.3.4 recordSet: RecordSet {in} Package: CategoriesSIM

The CategoriesSIM package contains the elements used to define the parameters and information management responsibilities of the Categories service.

Categories SIM Static Structure



The Categories SIM Static Structure diagram shows the information elements that comprise parameters or information that the Categories service must manage.

8.6.3.3.4.1 Class: CategoriesSIM::DispositionInstruction

See the Dispositions package of the RmsDomainModel for a definition of this element. It is included here as an element to store reference information about Dispositions.

Attributes

Attribute: DispositionInstruction.Id

Type: ID

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordCategory
In the Role of: category
Multiplicity: 0..1

To Class: CategoriesSIM::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 1..*

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::DispositionInstruction
In the Role of: previous
Multiplicity: 0..1

To Class: CategoriesSIM::DispositionInstruction
In the Role of: next
Multiplicity: 0..1

8.6.3.3.4.2 Class: CategoriesSIM::ManagedRecord

Provides a reference from within the Categories service for a particular ManagedRecord. Please see the ManagedRecords Package within the RmsDomainModel for a more complete definition. It is included here as an element to store reference information about the corresponding managed record.

Attributes

Attribute: ManagedRecord.managedRecordID

Type: ID
Description: A unique identifier for a particular ManagedRecord.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordSet
In the Role of: set
Multiplicity: 0..1

To Class: CategoriesSIM::ManagedRecord
In the Role of: member
Multiplicity: 1..*

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::CategoryManagedRecordAssociation
In the Role of: categoryAssociation
Multiplicity: 1..*

To Class: CategoriesSIM::ManagedRecord
In the Role of: record
Multiplicity: 1

8.6.3.3.4.3 Class: CategoriesSIM::RecordSet

See the Dispositions package of the RmsDomainModel for a definition of this element. It is included here as an element to store reference information for record sets.

Attributes

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordSet
In the Role of: set
Multiplicity: 0..1

To Class: CategoriesSIM::ManagedRecord
In the Role of: member
Multiplicity: 1..*

8.6.3.3.4.4 Class: CategoriesSIM::Activity

See the Category package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Activity.performerRoleId
Type: ID
Description: The Id of the party performing the activity.

Attribute: Activity.id

Type: ID
Description: A unique identifier for the ActivityType.

Attribute: Activity.name

Type: string
Description: See the Category package of the RmsDomainModel for a definition of this element.

Attribute: Activity.description

Type: string
Description: See the Category package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordCategory
In the Role of: category
Multiplicity: *

To Class: CategoriesSIM::Activity
In the Role of: associatedActivity
Multiplicity: 1

Dependency

The Categories service manages information about the business activities and the categorization of the records produced by those activities.

From Class: CategoriesService::Categories

To Class: CategoriesSIM::Activity

8.6.3.3.4.5 Class: CategoriesSIM::CategorizationSchema

See the Category package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: CategorizationSchema.schemaName

Type: string
Description: See the Category package of the RmsDomainModel for a definition of this element.

Attribute: CategorizationSchema.id

Type: ID
Description: See the Category package of the RmsDomainModel for a definition of this element.

Attribute: CategorizationSchema.description

Type: string
Description: See the Category package of the RmsDomainModel for a definition of this element.

Attribute: CategorizationSchema.authorityId

Type: ID
Description: An Id of the authority responsible for schema.

Connections

Aggregation

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordCategory
In the Role of: type
Multiplicity: 0..*

To Class: CategoriesSIM::CategorizationSchema
In the Role of: schema
Multiplicity: 1

Dependency

The Categories service manages information about record category schemas.

From Class: CategoriesService::Categories

To Class: CategoriesSIM::CategorizationSchema

8.6.3.3.4.6 Class: CategoriesSIM::CategoryManagedRecordAssociation

See the Category package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: CategoryManagedRecordAssociation.categoryAssignmentDate

Type: dateTime
Description: See the Category package of the RmsDomainModel for a definition of this element.

Connections

Constraint Name: self.previous.managedRecordID = self.managedRecordID

Constraint Name: self.next.managedRecordID = self.managedRecordID

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::CategoryManagedRecordAssociation
In the Role of: managedRecordAssociation
Multiplicity: 0..*

To Class: CategoriesSIM::RecordCategory
In the Role of: category
Multiplicity: 1

Dependency

The Categories service manages information about records associated with a particular category in a schema.

From Class: CategoriesService::Categories

To Class: CategoriesSIM::CategoryManagedRecordAssociation

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::CategoryManagedRecordAssociation
In the Role of: previous
Multiplicity: 0..1

To Class: CategoriesSIM::CategoryManagedRecordAssociation
In the Role of: next
Multiplicity: 0..1

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::CategoryManagedRecordAssociation
In the Role of: categoryAssociation
Multiplicity: 1..*

To Class: CategoriesSIM::ManagedRecord
In the Role of: record
Multiplicity: 1

8.6.3.3.4.7 Class: CategoriesSIM::RecordCategory

See the Category package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: RecordCategory.description

Type: string
Description: See the Category package of the RmsDomainModel for a definition of this element.

Attribute: RecordCategory.name

Type: string
Description: See the Category package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordCategory
In the Role of: parentCategory
Multiplicity: 0..1

To Class: CategoriesSIM::RecordCategory
In the Role of: childCategory
Multiplicity: 0..*

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::CategoryManagedRecordAssociation
In the Role of: managedRecordAssociation
Multiplicity: 0..*

To Class: CategoriesSIM::RecordCategory
In the Role of: category
Multiplicity: 1

Aggregation

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordCategory
In the Role of: type
Multiplicity: 0..*

To Class: CategoriesSIM::CategorizationSchema
In the Role of: schema
Multiplicity: 1

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordCategory
In the Role of: category
Multiplicity: *

To Class: CategoriesSIM::Activity
In the Role of: associatedActivity
Multiplicity: 1

Association

See the Category package of the RmsDomainModel for a definition of this association.

From Class: CategoriesSIM::RecordCategory
In the Role of: category
Multiplicity: 0..1

To Class: CategoriesSIM::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 1..*

Dependency

The Categories service manages information about record categories in a schema.

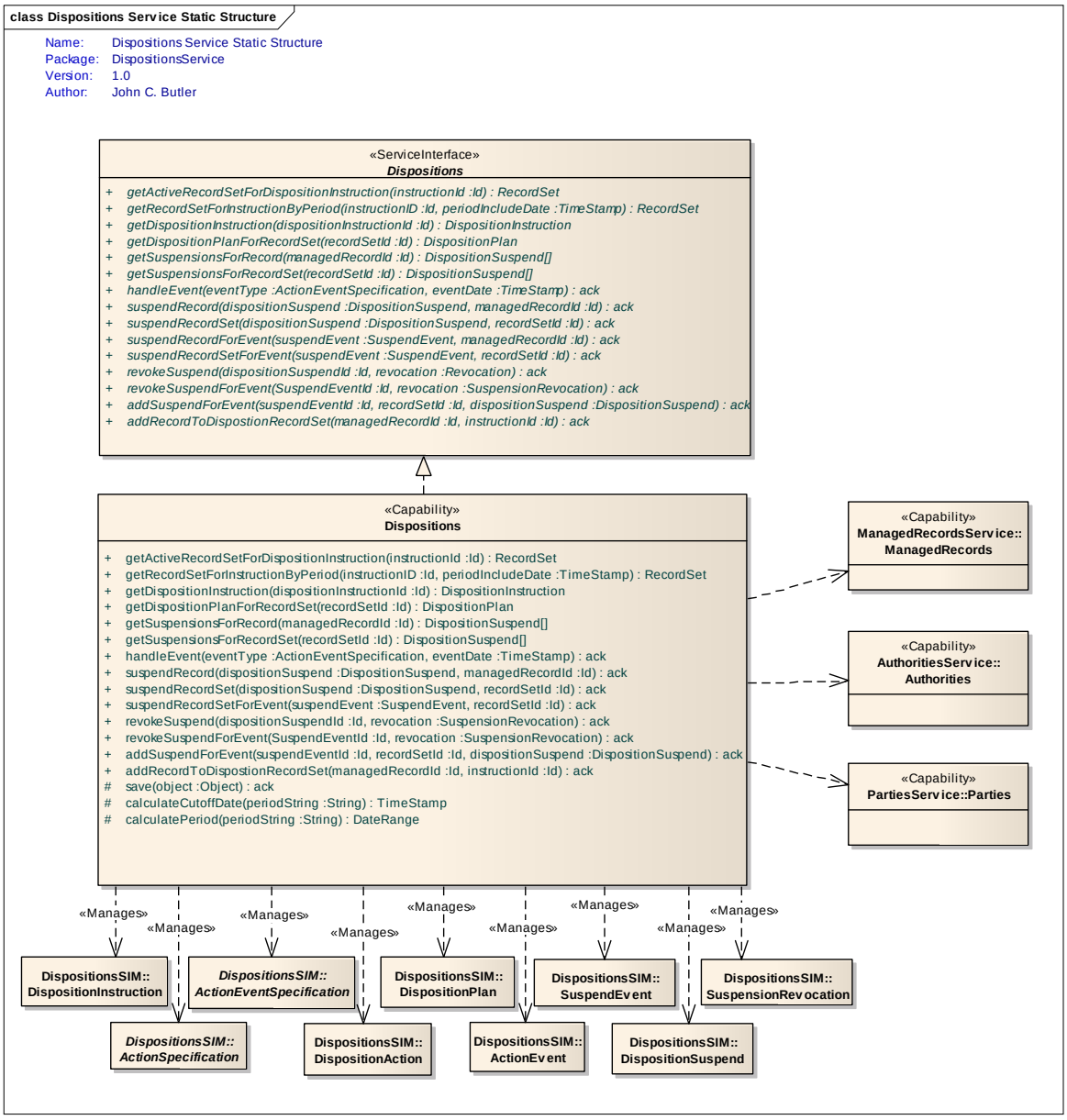
From Class: CategoriesService::Categories

To Class: CategoriesSIM::RecordCategory

8.6.3.4 Package: DispositionsService

The DispositionsService package contains the model elements that together define the Dispositions service.

Dispositions Service Static Structure



Implementations of the Dispositions service will need to realize the Dispositions ServiceInterface and all the behaviors associated with the Dispositions Capability. This includes management of a wide variety of information related to DispositionInstruction's, DispositionPlan's (instances of DispositionInstruction's for a particular RecordSet), Suspensions, Revocations. The Dispositions service will depend on the Party service for

the authority responsible for the Suspensions and SuspendEvent's and on the ManagedRecords service for references to the ManagedRecord.

8.6.3.4.1 Class: DispositionsService::Dispositions

The Dispositions Capably specifies the required behavior and constraints of any implementations of the Dispositions ServiceInterface in a platform independent manner. Dispositions are the actions that will be taken on a set of records once the set has been cut off.

Attributes

Connections

Dependency

The Dispositions service maintains references to information maintained by the Parties service such as the creator of a disposition instruction and the organization that represents the destination of a move.

From Class: DispositionsService::Dispositions

To Class: PartiesService::Parties

Dependency

The Dispositions service manages information related to suspensions since they impact the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::SuspendEvent

Dependency

The Dispositions service manages information related to suspensions since they impact the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionSuspend

Dependency

The Dispositions service manages information related to suspension revocations since they impact the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::SuspensionRevocation

Dependency

The Dispositions service maintains references to the managed records that are in a recordset.

From Class: DispositionsService::Dispositions

To Class: ManagedRecordsService::ManagedRecords

Dependency

RMS Clients will use the Dispositions service to manage information about disposition instructions, suspensions, and RecordSet's within RMS.

From Class: RmsSolution::RMS Client

To Class: DispositionsService::Dispositions

Dependency

The Dispositions service manages information related to disposition ActionSpecification's since they potentially trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::ActionSpecification

Realisation

From Class: DispositionsService::Dispositions

To Interface: DispositionsService::Dispositions

Dependency

The Categories service maintains references to the disposition instructions that apply to record categories.

From Class: CategoriesService::Categories

To Class: DispositionsService::Dispositions

Dependency

The Dispositions service maintains references to the Authorities for the disposition instructions.

From Class: DispositionsService::Dispositions

To Class: AuthoritiesService::Authorities

Dependency

The Dispositions service manages information related to Disposition Instructions.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionInstruction

Dependency

The Dispositions service manages information related to disposition ActionEventSpecification's since they potentially trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::ActionEventSpecification

Dependency

The Dispositions service manages information related to DispositionPlan's which are the instantiations of DispositionInstruction's for one or more RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionPlan

Dependency

The Dispositions service manages information related to DispositionAction's since they potentially trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionAction

Dependency

The Dispositions service manages information related to disposition ActionEvent's since they trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::ActionEvent

8.6.3.4.2 Interface: DispositionsService::Dispositions

The Dispositions ServiceInterface is a platform independent specification of the operation signatures of the Dispositions service. Refer to the Dispositions Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: DispositionsService::Dispositions

To Interface: DispositionsService::Dispositions

8.6.3.4.3 Capabilities: DispositionsService::Dispositions

addRecordToDispostionRecordSet: ack

Scope Public

Description This operation adds a record to a recordset for a disposition instruction with a given id. Parameters include the id of the managed record and the disposition instruction. The operation returns an acknowledgement of whether or not the operation executed successfully.

Parameters managedRecordId: Id {in}
instructionId: Id {in}

addSuspendForEvent: ack

Scope Public

Description This operation adds a suspension for a particular suspend event. This typically be the addition of a disposition suspend against another recordset. The operation returns an acknowledgement of whether or not the operation executed successfully.

Parameters suspendEventId: Id {in}
recordSetId: Id {in}
dispositionSuspend: DispositionSuspend {in}

calculateCutoffDate: TimeStamp

Scope	Protected
Description	This operation is to be implemented by the provider to determine when a cutoff date will occur.
Parameters	periodString: String {in}

calculatePeriod: DateRange

Scope	Protected
Description	This operation is to be implemented by the provider to determine the length of a period.
Parameters	periodString: String {in}

getActiveRecordSetForDispositionInstruction: RecordSet

Scope	Public
Description	This operation returns a recordset for the disposition instruction with the given id. If there is no disposition instruction for a category then the default instruction should be returned. If an instruction has not been set and no default is set, an exception should be raised.
Parameters	instructionId: Id {in}

getDispositionInstruction: DispositionInstruction

Scope	Public
Description	This operation returns the disposition instruction with the given id.
Parameters	dispositionInstructionId: Id {in}

getDispositionPlanForRecordSet: DispositionPlan

Scope	Public
Description	This operation returns the disposition plan for a recordset with a given id.
Parameters	recordSetId: Id {in}

getRecordSetForInstructionByPeriod: RecordSet

Scope	Public
Description	This operation returns the RecordSet for a particular RecordCategory that was active during the time period indicated by the periodIncludeDate. The active period runs from the RecordSet.periodStartDate to the RecordSet.periodEndDate. Parameters include the id of the instruction and a date the active period must include.
Parameters	instructionID: Id {in} periodIncludeDate: TimeStamp {in}

getSuspensionsForRecord: DispositionSuspend

Scope	Public
Description	This operation returns an array of suspensions for a managed record with a given id.
Parameters	managedRecordId: Id {in}

getSuspensionsForRecordSet: DispositionSuspend

Scope	Public
Description	This operation returns an array of suspensions for a recordset with a given id.
Parameters	recordSetId: Id {in}

handleEvent: ack

Scope	Public
Description	This operation processes action events related to dispositions. The operations includes parameters for the event type and the event date. The operation returns an acknowledgement of whether or not the operation executed successfully.
Parameters	eventType: ActionEventSpecification {in} eventDate: TimeStamp {in}

revokeSuspend: ack

Scope	Public
Description	This operation revokes the suspension with the given id. The parameters include the id of the disposition suspension and the suspension revocation. The operation returns an acknowledgement of whether or not the operation executed successfully.
Parameters	dispositionSuspendId: Id {in} revocation: SuspensionRevocation {in}

revokeSuspendForEvent: ack

Scope	Public
Description	This operation revokes all suspensions associated with a particular suspend event. The parameters include the id of the suspend event and the suspension revocation. The operation returns an acknowledgement of whether or not the operation executed successfully.
Parameters	SuspendEventId: Id {in} revocation: SuspensionRevocation {in}

save: ack

Scope	Protected

Description This private operation represents a responsibility of any implementation of the Dispositions service to be able to save a given object.

Parameters object: Object {in}

suspendRecord: ack

Scope Public

Description This operation places a suspension on a particular record. If the record is part of a recordset with additional managed records then a separate recordset will be created for that record and the suspension will be associated with the new recordset. Parameters include the disposition suspension and the id of the managed record to be suspended. The operation returns an acknowledgement of whether or not the operation executed successfully.

Parameters dispositionSuspend: DispositionSuspend {in}
managedRecordId: Id {in}

suspendRecordSet: ack

Scope Public

Description This operation places a suspension on an entire recordset with the given id. Parameters include the disposition suspension and the id of the recordset to be suspended. The operation returns an acknowledgement of whether or not the operation executed successfully.

Parameters dispositionSuspend: DispositionSuspend {in}
recordSetId: Id {in}

suspendRecordSetForEvent: ack

Scope Public

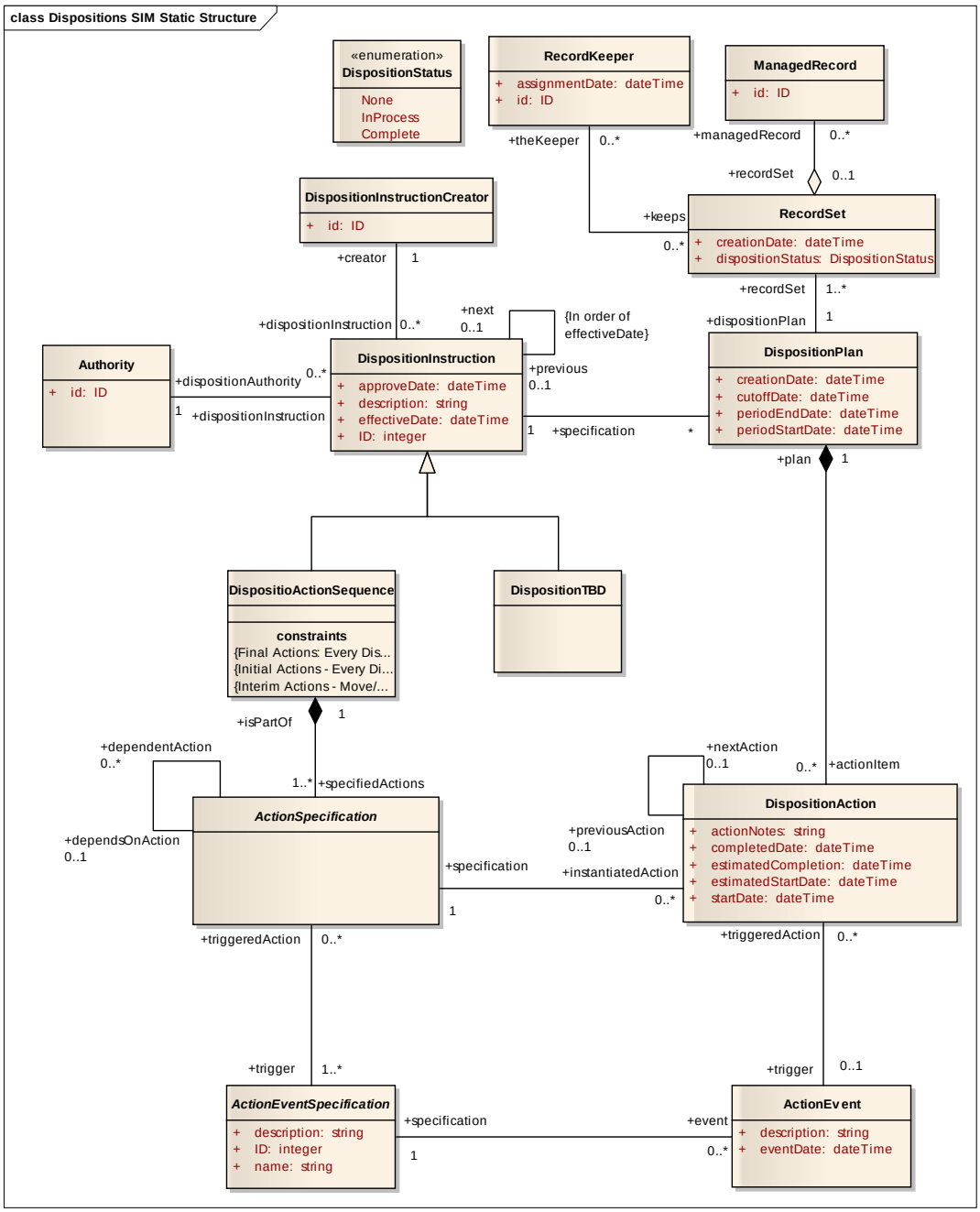
Description This operation a recordset based on a suspend event. The operation returns an acknowledgement of whether or not the operation executed successfully.

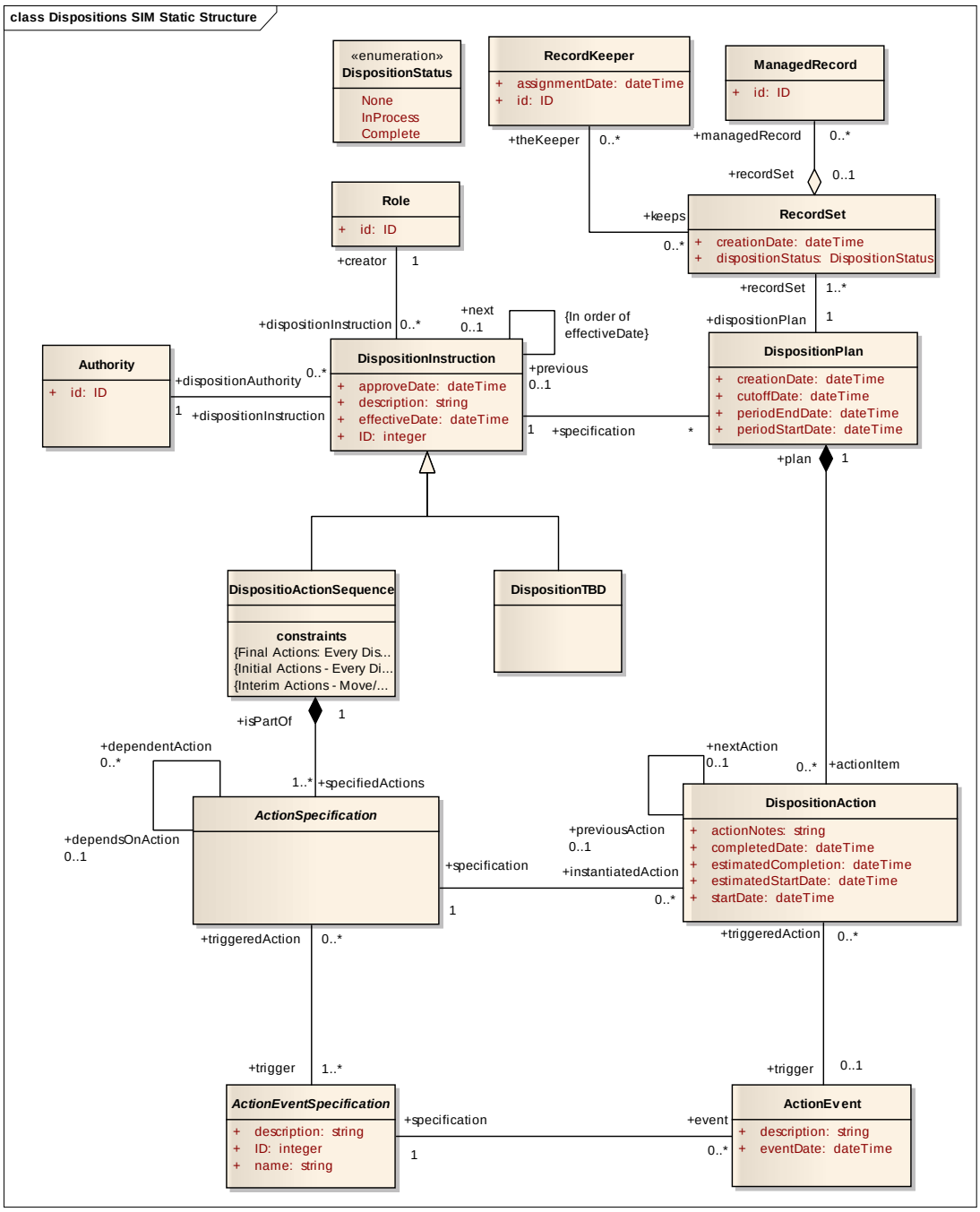
Parameters suspendEvent: SuspendEvent {in}
recordSetId: Id {in}

8.6.3.4.4 Package: DispositionsSIM

The DispositionsSIM package contains the elements used to define the parameters and information management responsibilities of the Dispositions service.

Dispositions SIM Static Structure

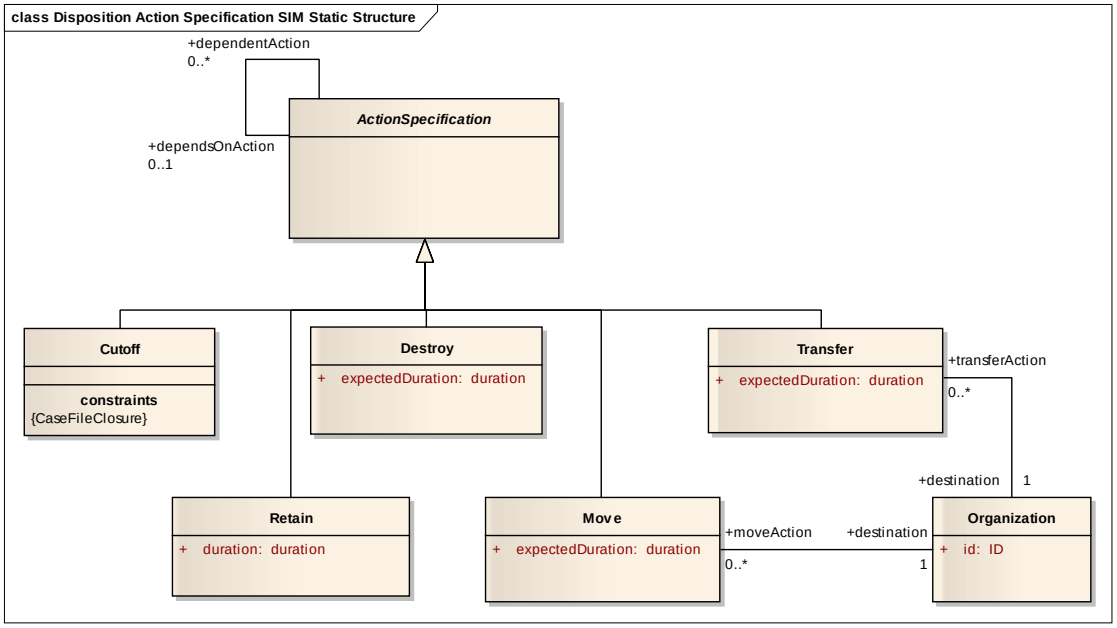




When a record is assigned to a record category, a disposition plan is generated based on the Disposition instruction. The Plan represents an assigned instance of the DispositionInstruction for the particular Record. The Plan also provides a mechanism to estimate the future RMS work load.

If record category is changed then the disposition plan must be deleted unless the record category points to the same DispositionInstruction.

Disposition Action Specification SIM Static Structure



When a record is assigned to a record category, a disposition plan is generated based on the Disposition instruction. The Plan represents an assigned instance of the DispositionInstruction for the particular Record. The Plan also provides a mechanism to estimate the future RMS work load.

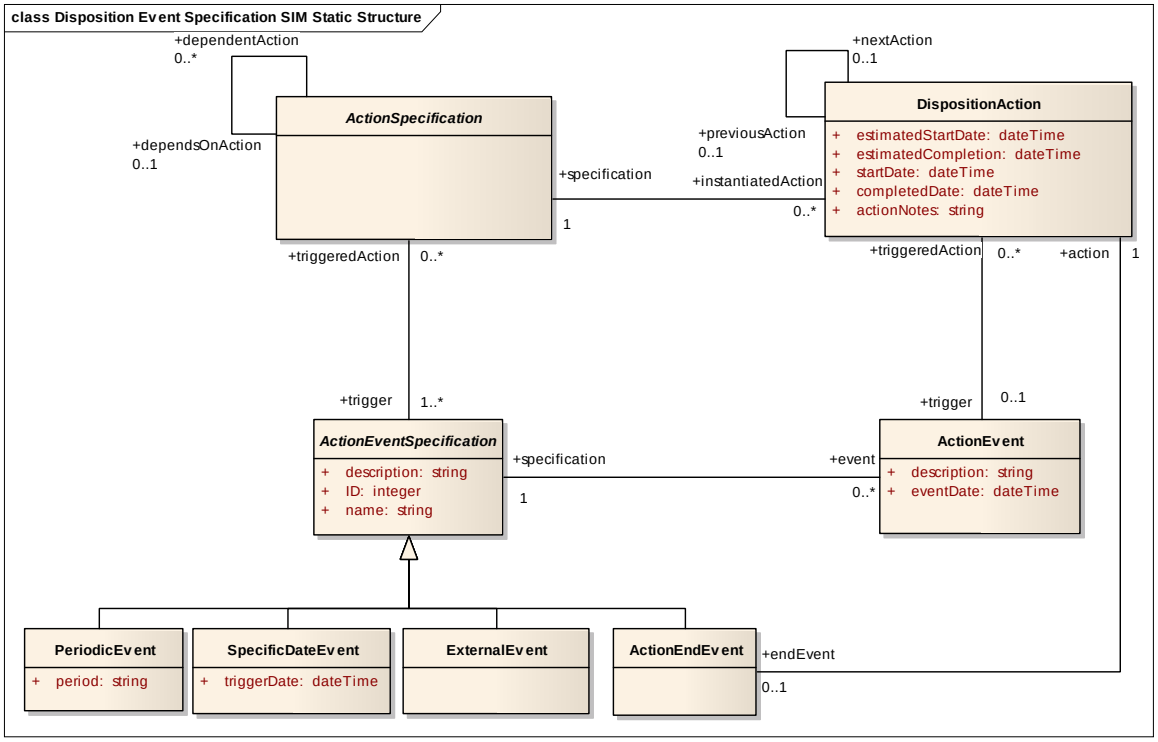
If record category is changed then the disposition plan must be deleted unless the record category points to the same DispositionInstruction.

Initial Actions - Every DispositionInstruction must have one and only one Cutoff ActionSpecification as it's first ActionSpecification. The cutoff must be followed by a Retain action. Cutoff cannot be executed on a RecordSet that contains a ManagedRecord with isCaseFile = True (i.e., the ManagedRecord is a "Case File Record") that has not been "closed" (i.e., ManagedRecord.closedDate has been filled with a valid dateTime.)

Interim Actions - Move/Retain Pairs - Every Move must be followed by a Retain.

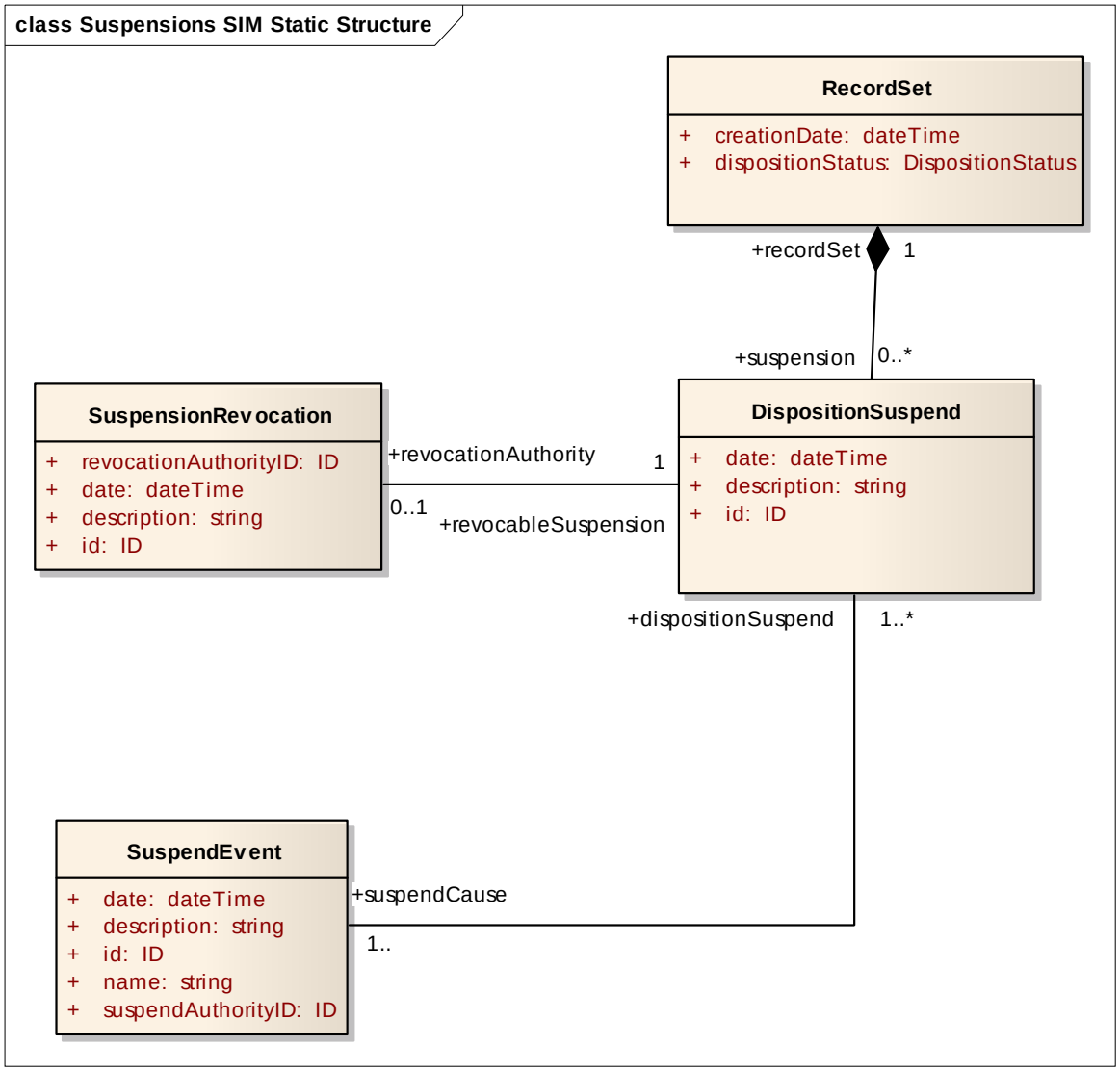
Final Actions - Every DispositionActionSequence must end with either a Transfer or Destroy ActionSpecification.

Disposition Event Specification SIM Static Structure



Actions are triggered by ActionEvent's and sometimes generate events as in the case of ActionEndEvent. Actions and Events provide a mechanism for defining and tracking the execution of DispositionInstruction's for RecordSet's. Please see the Disposition package of the RmsDomainModel for more detail.

Suspensions SIM Static Structure



The Suspensions SIM Static Structure diagram shows the information elements related to the suspension of managed record dispositions and the corresponding suspension revocations that the Dispositions service must manage.

8.6.3.4.4.1 Class: DispositionsSIM::ActionEndEvent

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: action
Multiplicity: 1

To Class: DispositionsSIM::ActionEndEvent
In the Role of: endEvent
Multiplicity: 0..1

Generalization

From Class: DispositionsSIM::ActionEndEvent

To Class: DispositionsSIM::ActionEventSpecification

8.6.3.4.4.2 Class: DispositionsSIM::ActionEvent

Please see the Dispositions package of the RmsDomainModel for a detailed definition of ActionEvent.

Attributes

Attribute: ActionEvent.description
Type: string

Attribute: ActionEvent.eventDate
Type: dateTime

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: triggeredAction
Multiplicity: 0..*

To Class: DispositionsSIM::ActionEvent
In the Role of: trigger
Multiplicity: 0..1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::ActionEvent
In the Role of: event
Multiplicity: 0..*

To Class: DispositionsSIM::ActionEventSpecification
In the Role of: specification
Multiplicity: 1

Dependency

The Dispositions service manages information related to disposition ActionEvent's since they trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::ActionEvent

8.6.3.4.4.3 Class: DispositionsSIM::ActionSpecification

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Generalization

From Class: DispositionsSIM::Destroy

To Class: DispositionsSIM::ActionSpecification

Generalization

From Class: DispositionsSIM::Cutoff

To Class: DispositionsSIM::ActionSpecification

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: instantiatedAction
Multiplicity: 0..*

To Class: DispositionsSIM::ActionSpecification
In the Role of: specification
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::ActionSpecification
In the Role of: triggeredAction
Multiplicity: 0..*

To Class: DispositionsSIM::ActionEventSpecification
In the Role of: trigger
Multiplicity: 1..*

Aggregation

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::ActionSpecification
In the Role of: specifiedActions
Multiplicity: 1..*

To Class: DispositionsSIM::DispositioActionSequence
In the Role of: isPartOf
Multiplicity: 1

Dependency

The Dispositions service manages information related to disposition ActionSpecification's since they potentially trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::ActionSpecification

Generalization

From Class: DispositionsSIM::Transfer

To Class: DispositionsSIM::ActionSpecification

Generalization

From Class: DispositionsSIM::Move

To Class: DispositionsSIM::ActionSpecification

Generalization

From Class: DispositionsSIM::Retain

To Class: DispositionsSIM::ActionSpecification

AssociationClass

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::ActionSpecification
In the Role of: dependentAction
Multiplicity: 0..*

To Class: DispositionsSIM::ActionSpecification
In the Role of: dependsOnAction
Multiplicity: 0..1

8.6.3.4.4.4 Class: DispositionsSIM::Authority

See the Party package of the RmsDomainModel for a definition of this element. Note that this element is only a reference to the authority maintained by the Authorities service.

Attributes

Attribute: Authority.id

Type: ID
Description: A unique identifier of the disposition authority for the instruction.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 0..*

To Class: DispositionsSIM::Authority
In the Role of: dispositionAuthority
Multiplicity: 1

8.6.3.4.4.5 Class: DispositionsSIM::Cutoff

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Generalization

From Class: DispositionsSIM::Cutoff

To Class: DispositionsSIM::ActionSpecification

Constraints

CaseFileClosure

Description: Cutoff action cannot be executed on a RecordSet that contains a ManagedRecord with isCaseFile = True that has not been closed.

8.6.3.4.4.6 Class: DispositionsSIM::Destroy

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Destroy.expectedDuration

Type: duration

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Generalization

From Class: DispositionsSIM::Destroy

To Class: DispositionsSIM::ActionSpecification

8.6.3.4.4.7 Class: DispositionsSIM::DispositioActionSequence

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Constraint Name: Final Actions: Every DispositionActionSequence must end with either a Transfer or Destroy ActionSpecification.

Constraint Name: Initial Actions - Every DispositionInstruction must have one and only one Cutoff ActionSpecification as it's first ActionSpecification. The cutoff must be followed by a Retain action.

Constraint Name: Interim Actions - Move/Retain Pairs - Every Move must be followed by a Retain

Generalization

From Class: DispositionsSIM::DispositioActionSequence

To Class: DispositionsSIM::DispositionInstruction

Aggregation

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::ActionSpecification
In the Role of: specifiedActions
Multiplicity: 1..*

To Class: DispositionsSIM::DispositioActionSequence
In the Role of: isPartOf
Multiplicity: 1

8.6.3.4.4.8 Class: DispositionsSIM::DispositionAction

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: DispositionAction.estimatedStartDate
Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionAction.estimatedCompletion
Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionAction.startDate
Type: dateTime

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionAction.completedDate

Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionAction.actionNotes

Type: string
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: action
Multiplicity: 1

To Class: DispositionsSIM::ActionEndEvent
In the Role of: endEvent
Multiplicity: 0..1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: instantiatedAction
Multiplicity: 0..*

To Class: DispositionsSIM::ActionSpecification
In the Role of: specification
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: triggeredAction

Multiplicity: 0..*

To Class: DispositionsSIM::ActionEvent
In the Role of: trigger
Multiplicity: 0..1

Dependency

The Dispositions service manages information related to DispositionAction's since they potentially trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionAction

Aggregation

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: actionItem
Multiplicity: 0..*

To Class: DispositionsSIM::DispositionPlan
In the Role of: plan
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: nextAction
Multiplicity: 0..1

To Class: DispositionsSIM::DispositionAction
In the Role of: previousAction
Multiplicity: 0..1

8.6.3.4.4.9 Class: DispositionsSIM::DispositionPlan

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: DispositionPlan.cutoffDate

Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionPlan.creationDate

Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionPlan.periodStartDate

Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionPlan.periodEndDate

Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::RecordSet
In the Role of: recordSet
Multiplicity: 1..*

To Class: DispositionsSIM::DispositionPlan
In the Role of: dispositionPlan
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionPlan
Multiplicity: *

To Class: DispositionsSIM::DispositionInstruction
In the Role of: specification
Multiplicity: 1

Dependency

The Dispositions service manages information related to DispositionPlan's which are the instantiations of DispositionInstruction's for one or more RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionPlan

Aggregation

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionAction
In the Role of: actionItem
Multiplicity: 0..*

To Class: DispositionsSIM::DispositionPlan
In the Role of: plan
Multiplicity: 1

8.6.3.4.4.10 Enumeration: DispositionsSIM::DispositionStatus

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: DispositionStatus.None

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionStatus.InProcess

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionStatus.Complete

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

8.6.3.4.4.11 Class: DispositionsSIM::DispositionTBD

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Generalization

From Class: DispositionsSIM::DispositionTBD

To Class: DispositionsSIM::DispositionInstruction

8.6.3.4.4.12 Class: DispositionsSIM::ExternalEvent

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Generalization

From Class: DispositionsSIM::ExternalEvent

To Class: DispositionsSIM::ActionEventSpecification

8.6.3.4.4.13 Class: DispositionsSIM::ManagedRecord

See the ManagedRecord package of the RmsDomainModel for a definition of this element. Note that this element is only a reference to the managed record maintained by the ManagedRecords service.

Attributes

Attribute: ManagedRecord.id

Type: ID
Description: Unique identifier for the managed record.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::RecordSet
In the Role of: recordSet
Multiplicity: 0..1

To Class: DispositionsSIM::ManagedRecord
In the Role of: managedRecord
Multiplicity: 0..*

8.6.3.4.4.14 Class: DispositionsSIM::Move

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Move.expectedDuration

Type: duration
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::Move
In the Role of: moveAction
Multiplicity: 0..*

To Class: DispositionsSIM::Organization
In the Role of: destination
Multiplicity: 1

Generalization

From Class: DispositionsSIM::Move

To Class: DispositionsSIM::ActionSpecification

8.6.3.4.4.15 Class: DispositionsSIM::Organization

See the Parties package of the RmsDomainModel for a definition of this element. It should be noted that this element is a placeholder that acts as a reference element to the Parties service.

Attributes

Attribute: Organization.id

Type: ID
Description: A unique identifier for the organization.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::Move
In the Role of: moveAction
Multiplicity: 0..*

To Class: DispositionsSIM::Organization
In the Role of: destination
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::Transfer
In the Role of: transferAction
Multiplicity: 0..*

To Class: DispositionsSIM::Organization
In the Role of: destination
Multiplicity: 1

8.6.3.4.4.16 Class: DispositionsSIM::PeriodicEvent

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: PeriodicEvent.period
Type: string
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Generalization

From Class: DispositionsSIM::PeriodicEvent

To Class: DispositionsSIM::ActionEventSpecification

8.6.3.4.4.17 Class: DispositionsSIM::RecordSet

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: RecordSet.creationDate
Type: dateTime

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: RecordSet.dispositionStatus

Type: DispositionStatus
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::RecordSet
In the Role of: recordSet
Multiplicity: 1..*

To Class: DispositionsSIM::DispositionPlan
In the Role of: dispositionPlan
Multiplicity: 1

Aggregation

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionSuspend
In the Role of: suspension
Multiplicity: 0..*

To Class: DispositionsSIM::RecordSet
In the Role of: recordSet
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::RecordSet
In the Role of: recordSet
Multiplicity: 0..1

To Class: DispositionsSIM::ManagedRecord
In the Role of: managedRecord
Multiplicity: 0..*

8.6.3.4.4.18 Class: DispositionsSIM::Retain

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Retain.duration

Type: duration
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Generalization

From Class: DispositionsSIM::Retain

To Class: DispositionsSIM::ActionSpecification

8.6.3.4.4.19 Class: DispositionsSIM::DispositionInstructionCreator

The role that created the disposition instruction. This element is a reference to the information managed by the Parties service.

Attributes

Attribute: Role.id

Type: ID

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 0..*

To Class: DispositionsSIM::DispositionInstructionCreator
In the Role of: creator
Multiplicity: 1

8.6.3.4.4.20 Class: DispositionsSIM::SpecificDateEvent

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: SpecificDateEvent.triggerDate

Type: dateTime

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Generalization

From Class: DispositionsSIM::SpecificDateEvent

To Class: DispositionsSIM::ActionEventSpecification

8.6.3.4.4.21 Class: DispositionsSIM::SuspendEvent

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: SuspendEvent.date

Type: dateTime

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: SuspendEvent.description

Type: string

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: SuspendEvent.id

Type: ID

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: SuspendEvent.name

Type: string

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: SuspendEvent.suspendAuthorityID

Type: ID

Description: Identifier for the authority invoking the suspension.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::SuspendEvent
In the Role of: suspendCause
Multiplicity: 1..

To Class: DispositionsSIM::DispositionSuspend
In the Role of: dispositionSuspend
Multiplicity: 1..*

Dependency

The Dispositions service manages information related to suspensions since they impact the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::SuspendEvent

8.6.3.4.4.22 Class: DispositionsSIM::Transfer

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Transfer.expectedDuration
Type: duration

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::Transfer
In the Role of: transferAction
Multiplicity: 0..*

To Class: DispositionsSIM::Organization
In the Role of: destination
Multiplicity: 1

Generalization

From Class: DispositionsSIM::Transfer

To Class: DispositionsSIM::ActionSpecification

8.6.3.4.4.23 Class: DispositionsSIM::ActionEventSpecification

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: ActionEventSpecification.description
Type: string

Attribute: ActionEventSpecification.ID
Type: integer

Attribute: ActionEventSpecification.name
Type: string

Connections

Generalization

From Class: DispositionsSIM::ActionEndEvent

To Class: DispositionsSIM::ActionEventSpecification

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::ActionSpecification
In the Role of: triggeredAction
Multiplicity: 0..*

To Class: DispositionsSIM::ActionEventSpecification
In the Role of: trigger
Multiplicity: 1..*

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::ActionEvent
In the Role of: event
Multiplicity: 0..*

To Class: DispositionsSIM::ActionEventSpecification

In the Role of: specification
Multiplicity: 1

Dependency

The Dispositions service manages information related to disposition ActionEventSpecification's since they potentially trigger changes to the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::ActionEventSpecification

Generalization

From Class: DispositionsSIM::SpecificDateEvent

To Class: DispositionsSIM::ActionEventSpecification

Generalization

From Class: DispositionsSIM::PeriodicEvent

To Class: DispositionsSIM::ActionEventSpecification

Generalization

From Class: DispositionsSIM::ExternalEvent

To Class: DispositionsSIM::ActionEventSpecification

8.6.3.4.4.24 Class: DispositionsSIM::DispositionInstruction

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: DispositionInstruction.approveDate
Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionInstruction.ID
Type: integer
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionInstruction.description
Type: string

Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionInstruction.effectiveDate

Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Generalization

From Class: DispositionsSIM::DispositioActionSequence

To Class: DispositionsSIM::DispositionInstruction

Generalization

From Class: DispositionsSIM::DispositionTBD

To Class: DispositionsSIM::DispositionInstruction

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 0..*

To Class: DispositionsSIM::Authority
In the Role of: dispositionAuthority
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionInstruction
In the Role of: dispositionInstruction
Multiplicity: 0..*

To Class: DispositionsSIM::DispositionInstructionCreator
In the Role of: creator
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionPlan
Multiplicity: *

To Class: DispositionsSIM::DispositionInstruction
In the Role of: specification
Multiplicity: 1

Dependency

The Dispositions service manages information related to Disposition Instructions.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionInstruction

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionInstruction
In the Role of: previous
Multiplicity: 0..1

To Class: DispositionsSIM::DispositionInstruction
In the Role of: next
Multiplicity: 0..1

8.6.3.4.4.25 Class: DispositionsSIM::DispositionSuspend

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: DispositionSuspend.date

Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionSuspend.description

Type: string
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: DispositionSuspend.id

Type: ID
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::SuspensionRevocation
In the Role of: revocationAuthority
Multiplicity: 0..1

To Class: DispositionsSIM::DispositionSuspend
In the Role of: revocableSuspension
Multiplicity: 1

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::SuspendEvent
In the Role of: suspendCause
Multiplicity: 1..

To Class: DispositionsSIM::DispositionSuspend
In the Role of: dispositionSuspend
Multiplicity: 1..*

Dependency

The Dispositions service manages information related to suspensions since they impact the disposition of RecordSet's.

From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::DispositionSuspend

Aggregation

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::DispositionSuspend

In the Role of: suspension
Multiplicity: 0..*

To Class: DispositionsSIM::RecordSet
In the Role of: recordSet
Multiplicity: 1

8.6.3.4.4.26 Class: DispositionsSIM::SuspensionRevocation

See the Dispositions package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: SuspensionRevocation.revocationAuthorityID
Type: ID
Description: Identifier for the authority revoking the suspension.

Attribute: SuspensionRevocation.date
Type: dateTime
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: SuspensionRevocation.description
Type: string
Description: See the Dispositions package of the RmsDomainModel for a definition of this element.

Attribute: SuspensionRevocation.id
Type: ID

Connections

Association

See the Dispositions package of the RmsDomainModel for a definition of this association.

From Class: DispositionsSIM::SuspensionRevocation
In the Role of: revocationAuthority
Multiplicity: 0..1

To Class: DispositionsSIM::DispositionSuspend
In the Role of: revocableSuspension
Multiplicity: 1

Dependency

The Dispositions service manages information related to suspension revocations since they impact the disposition of RecordSet's.

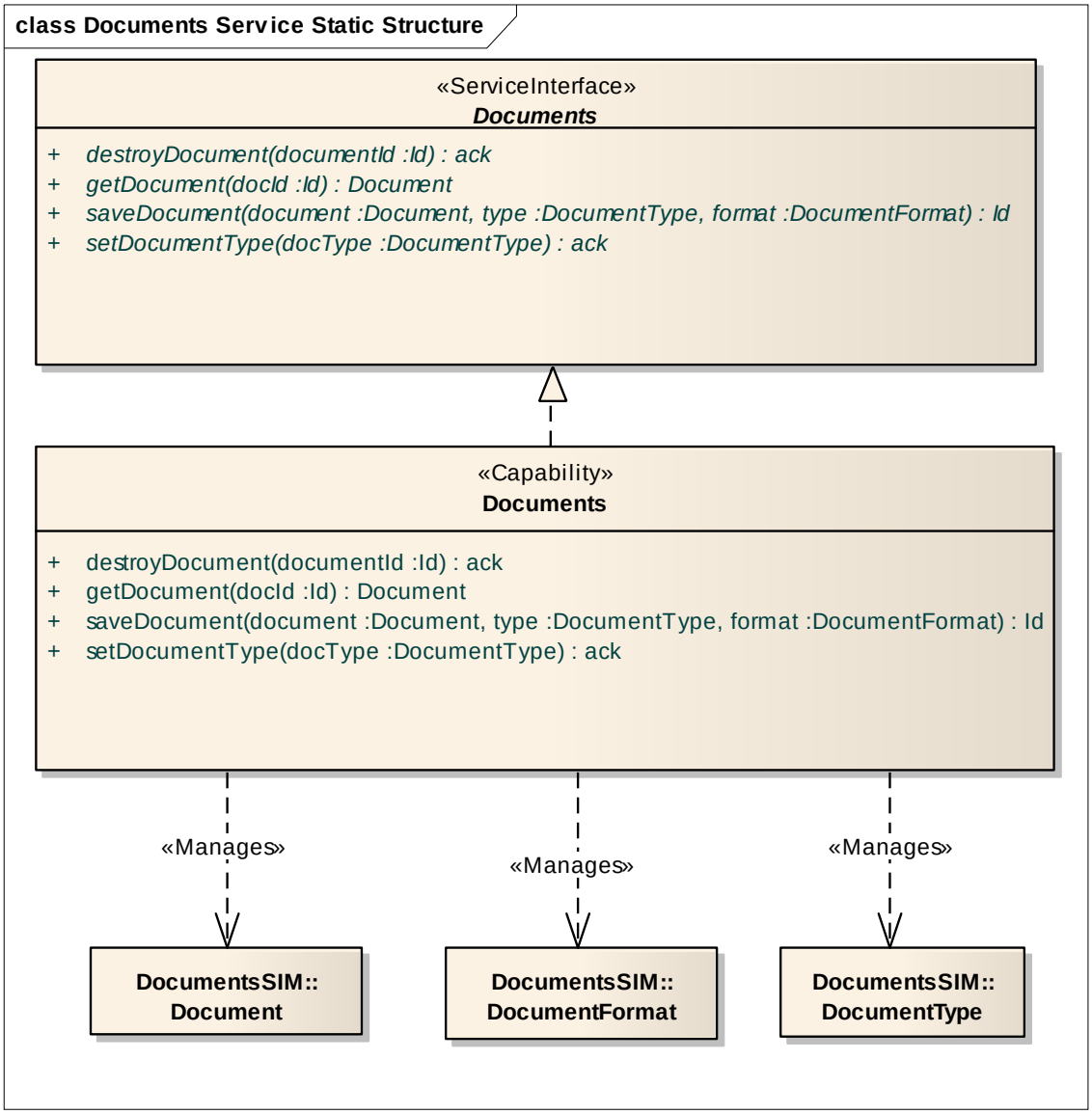
From Class: DispositionsService::Dispositions

To Class: DispositionsSIM::SuspensionRevocation

8.6.3.5 Package: DocumentsService

The DocumentsService package contains the model elements that together define the Documents service.

Documents Service Static Structure



Implementations of the Documents service will need to realize the Documents ServiceInterface and all the behaviors associated with the Documents Capability. This includes management of the documents which comprise the content of ManagedRecords.

8.6.3.5.1 Class: DocumentsService::Documents

Attributes

Connections

Dependency

The RecordAuthentications service maintains references to the documents associated with managed records that have been authenticated.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: DocumentsService::Documents

Realisation

From Class: DocumentsService::Documents

To Interface: DocumentsService::Documents

Dependency

The Documents service manages information related to the documents that represent the content of managed records.

From Class: DocumentsService::Documents

To Class: DocumentsSIM::Document

Dependency

The Documents service manages information related to the documents that represent the content of managed records.

From Class: DocumentsService::Documents

To Class: DocumentsSIM::DocumentFormat

Dependency

The Documents service manages information related to the documents that represent the content of managed records.

From Class: DocumentsService::Documents

To Class: DocumentsSIM::DocumentType

Dependency

The ManagedRecords service depends on the Documents service to manage the actual documents that make up the contents of the record.

From Class: ManagedRecordsService::ManagedRecords

To Class: DocumentsService::Documents

8.6.3.5.2 Interface: DocumentsService::Documents

Attributes

Connections

Realisation

From Class: DocumentsService::Documents

To Interface: DocumentsService::Documents

8.6.3.5.3 Capabilities: DocumentsService::Documents

destroyDocument: ack

Scope Public

Description This operation removes the Document from the Records Management Environment. The operation returns an acknowledgement indicating success or failure.

Parameters documentId: Id {in}

getDocument: Document

Scope Public

Description This operation returns the Document associated with the docid.

Parameters docId: Id {in}

saveDocument: Id

Scope	Public
Description	This operation "uploads" a Document to the Records Management Environment along with its type and format. It returns with the system assigned Id.
Parameters	document: Document {in} type: DocumentType {in} format: DocumentFormat {in}

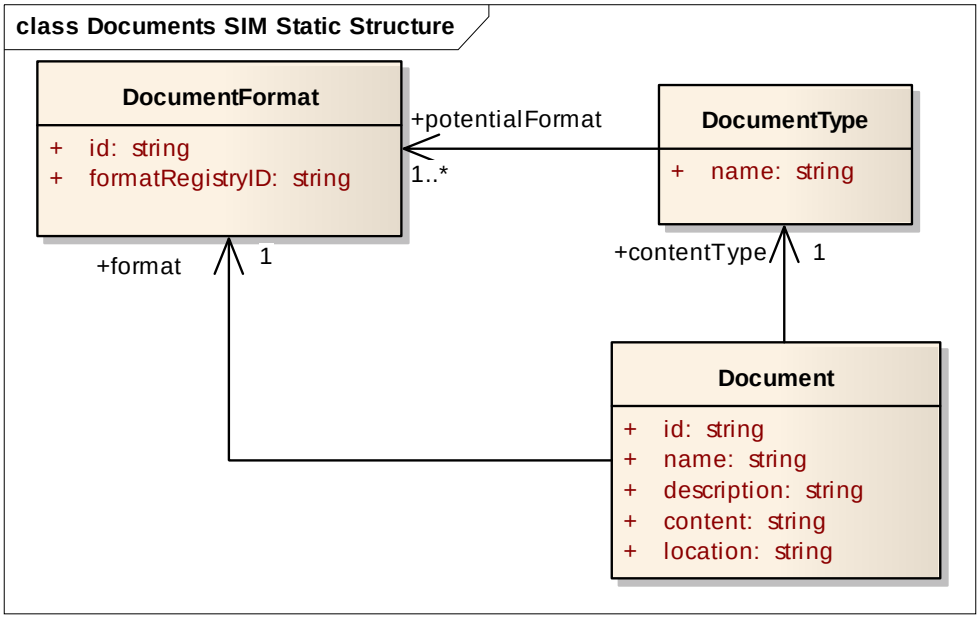
setDocumentType: ack

Scope	Public
Description	This operation creates a DocumentType The operation returns an acknowledgement indicating success or failure.
Parameters	docType: DocumentType {in}

8.6.3.5.4 Package: DocumentsSIM

The DocumentsSIM package contains the elements used to define the parameters and information management responsibilities of the Documents service.

Documents SIM Static Structure



The Documents SIM Static Structure diagram shows the information elements related to the documents that represent the contents of managed record that the Documents service must manage.

8.6.3.5.4.1 Class: DocumentsSIM::DocumentFormat

See the Document package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: DocumentFormat.id

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Attribute: DocumentFormat.formatRegistryID

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Document package of the RmsDomainModel for a definition of this association.

From Class: DocumentsSIM::DocumentType

To Class: DocumentsSIM::DocumentFormat
In the Role of: potentialFormat
Multiplicity: 1..*

Association

See the Document package of the RmsDomainModel for a definition of this association.

From Class: DocumentsSIM::Document

To Class: DocumentsSIM::DocumentFormat
In the Role of: format
Multiplicity: 1

Dependency

The Documents service manages information related to the documents that represent the content of managed records.

From Class: DocumentsService::Documents

To Class: DocumentsSIM::DocumentFormat

8.6.3.5.4.2 Class: DocumentsSIM::DocumentType

See the Document package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: DocumentType.name

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Document package of the RmsDomainModel for a definition of this association.

From Class: DocumentsSIM::Document

To Class: DocumentsSIM::DocumentType
In the Role of: contentType
Multiplicity: 1

Association

See the Document package of the RmsDomainModel for a definition of this association.

From Class: DocumentsSIM::DocumentType

To Class: DocumentsSIM::DocumentFormat
In the Role of: potentialFormat
Multiplicity: 1..*

Dependency

The Documents service manages information related to the documents that represent the content of managed records.

From Class: DocumentsService::Documents

To Class: DocumentsSIM::DocumentType

8.6.3.5.4.3 Class: DocumentsSIM::Document

See the Document package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Document.id

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Attribute: Document.name

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Attribute: Document.description

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Attribute: Document.content

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Attribute: Document.location

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Connections

Association

See the Document package of the RmsDomainModel for a definition of this association.

From Class: DocumentsSIM::Document

To Class: DocumentsSIM::DocumentType
In the Role of: contentType
Multiplicity: 1

Association

See the Document package of the RmsDomainModel for a definition of this association.

From Class: DocumentsSIM::Document

To Class: DocumentsSIM::DocumentFormat
In the Role of: format
Multiplicity: 1

Dependency

The Documents service manages information related to the documents that represent the content of managed records.

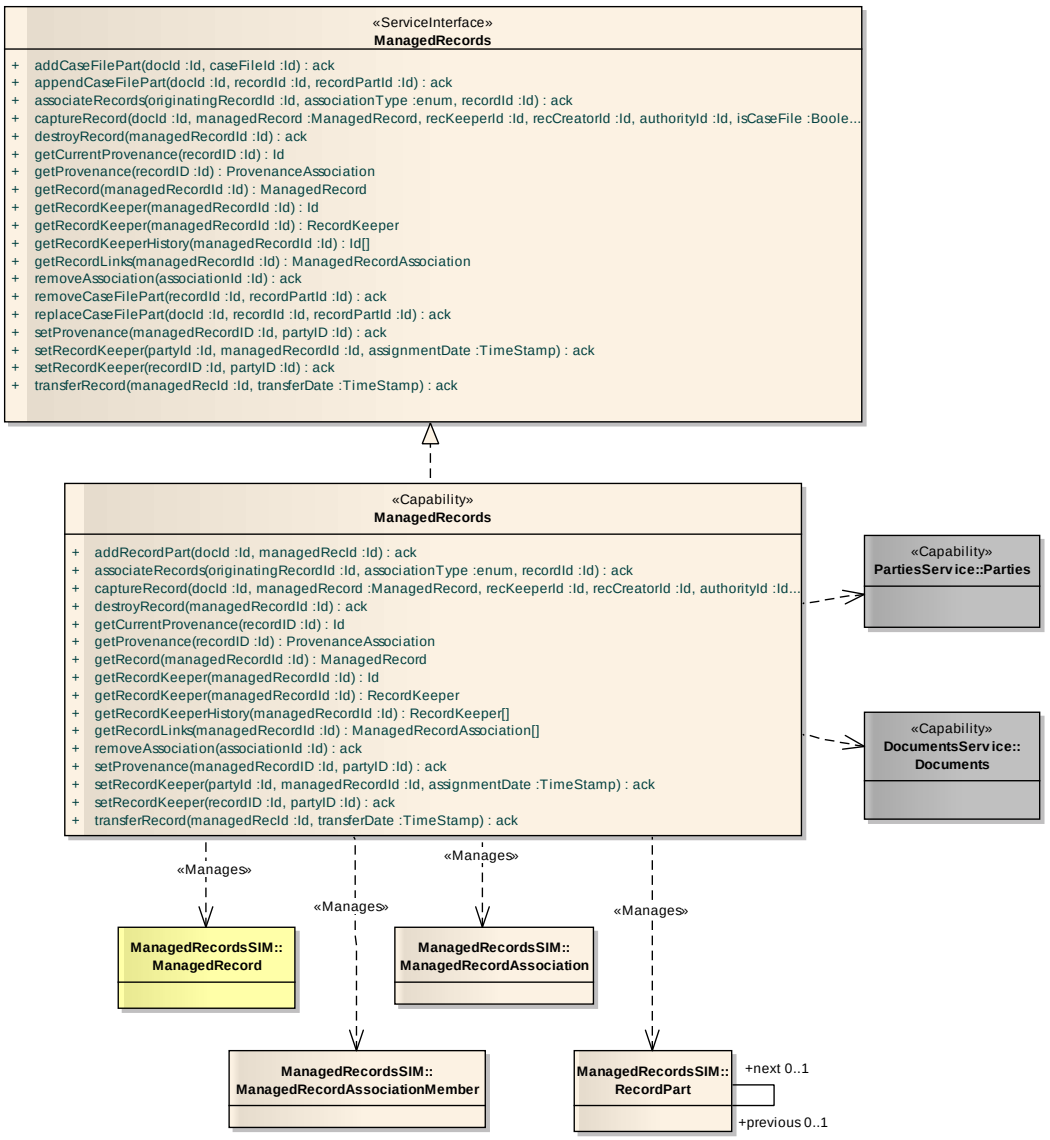
From Class: DocumentsService::Documents

To Class: DocumentsSIM::Document

8.6.3.6 Package: ManagedRecordsService

The ManagedRecordsService package contains the model elements that together define the ManagedRecords service.

Managed Records Service Static Structure



Implementations of the ManagedRecords service will need to realize the ManagedRecords ServiceInterface and all the behaviors associated with the ManagedRecords Capability. This includes management of a wide variety of information related to ManagedRecords including the RecordPart's, associations to other ManagedRecords, and CaseFileDefinitions. The service depends on the Parties service for information regarding RecordKeeper's and RecordCreator's. It depends on the Documents service to manage the documents that make up the contents of the ManagedRecords and, the Authorities service to manage information regarding the parties with legal authority over the managed records.

8.6.3.6.1 Class: ManagedRecordsService::ManagedRecords

The ManagedRecords Capably specifies the required behavior and constraints of any implementation of the ManagedRecords ServiceInterface in a platform independent

manner. Managed records are the records generated during the course of business that an organization is interested in tracking and includes case files. This specification is not prescriptive about what is and is not a record or case file.

The ManagedRecords service depends on the Documents service to manage the actual documents that make up the contents of the record. The ManagedRecords service maintains the additional metadata about the document required for records management.

The Managed Records service also depends on the Parties service to maintain information regarding organizations that play various roles with respect to the managed record such as record keeper or record creator.

Attributes

Connections

Dependency

The ManagedRecords service maintains references to the party that created a record, that has provenance over the record, and that is currently keeping the record.

From Class: ManagedRecordsService::ManagedRecords

To Class: PartiesService::Parties

Dependency

The Categories service maintains references to managed records and the categories they fall into.

From Class: CategoriesService::Categories

To Class: ManagedRecordsService::ManagedRecords

Dependency

The RecordAuthentications service maintains references to the managed records that have been authenticated.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: ManagedRecordsService::ManagedRecords

Dependency

The ManagedRecords service is responsible for managing the associations of related managed records.

From Class: ManagedRecordsService::ManagedRecords

To AssociationClass:
ManagedRecordsSIM::ManagedRecordAssociation Member

Dependency

From Issue: ManagedRecordsService::Can we add parts to a record? If not, how do we save an e-mail with attachment? Do we use Rec Association for that?

To Class: ManagedRecordsService::ManagedRecords

Dependency

RMS Clients will use the ManagedRecords service to capture and maintain managed records within RMS.

From Class: RmsSolution::RMS Client

To Class: ManagedRecordsService::ManagedRecords

Realisation

From Class: ManagedRecordsService::ManagedRecords

To Interface: ManagedRecordsService::ManagedRecords

Dependency

The ManagedRecords service is responsible for managing the information related to managed records and their record parts.

From Class: ManagedRecordsService::ManagedRecords

To Class: ManagedRecordsSIM::RecordPart

Dependency

The ManagedRecords service is responsible for managing the associations of related managed records.

From Class: ManagedRecordsService::ManagedRecords

To Class: ManagedRecordsSIM::ManagedRecordAssociation

Dependency

The Dispositions service maintains references to the managed records that are in a recordset.

From Class: DispositionsService::Dispositions

To Class: ManagedRecordsService::ManagedRecords

Dependency

From Class: ManagedRecordsService::ManagedRecords

To Class: AttributeProfiles Service::AttributeProfiles

Dependency

The ManagedRecords service is responsible for managing the information related to managed records and their record parts.

From Class: ManagedRecordsService::ManagedRecords

To Class: ManagedRecordsSIM::ManagedRecord

Dependency

The ManagedRecords service depends on the Documents service to manage the actual documents that make up the contents of the record.

From Class: ManagedRecordsService::ManagedRecords

To Class: DocumentsService::Documents

Dependency

The Annotations service maintains references to the records being annotated.

From Class: AnnotationsService::Annotations

To Class: ManagedRecordsService::ManagedRecords

8.6.3.6.2 Interface: ManagedRecordsService::ManagedRecords

The ManagedRecords ServiceInterface is a platform independent specification of the operation signatures of the ManagedRecords service. Refer to the ManagedRecords Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: ManagedRecordsService::ManagedRecords

To Interface: ManagedRecordsService::ManagedRecords

8.6.3.6.3 Capabilities: ManagedRecordsService::ManagedRecords

addCaseFilePart: ack

Scope Public

Description This operation adds a document with a given document id as a record part of a case file (i.e. a managed record with isCaseFile set to true) with the given record id. The operation returns an acknowledgement of the success or failure of the operation.

Parameters docId: Id {in}
caseFileId: Id {in}

addRecordPart: ack

Scope Public

Description This operation adds a document with a given document id as a record part of a managed record with the given record id. The operation returns an acknowledgement of the success or failure of the operation.

Parameters docId: Id {in}
managedRecId: Id {in}

appendCaseFilePart: ack

Scope Public

Description This operation adds a RecordPart with a pointer to a document with a given document id immediately following the RecordPart with the given id to the managed record with the given record id. The operation returns an acknowledgement of the success or failure of the operation.

Parameters docId: Id {in}

recordId: Id {in}
recordPartId: Id {in}

associateRecords: ack

Scope Public

Description This operation creates an association between the originating record and another record. The type of association is specified using the associationType enumeration. The operation returns an acknowledgement of the success or failure of the operation.

Parameters originatingRecordId: Id {in}
associationType: enum {in}
recordId: Id {in}

captureRecord: managedRecId

Scope Public

Description This operation captures a new managed record for a document with the given document id. The operation also takes the id of the record keeper, the id of the record creator, and the id of the authority responsible for the record. The operation returns the id of the new managed record.

Parameters docId: Id {in}
managedRecord: ManagedRecord {in}
recKeeperId: Id {in}
recCreatorId: Id {in}
authorityId: Id {in}
isCaseFile: Boolean {in}

destroyRecord: ack

Scope Public

Description This operation removes the managed record with the given id from the system and returns an acknowledgement of the success or failure of the operation.

Parameters managedRecordId: Id {in}

getCurrentProvenance: Id

Scope Public

Description This operation returns the id of the Party with provenance for the managed record with the given id.

Parameters recordID: Id {in}

getProvenance: ProvenanceAssociation

Scope Public

Description This operation returns the provenance association for the record with the given record id.

Parameters recordID: Id {in}

getRecord: ManagedRecord

Scope Public

Description This operation returns the managed record associated with the given record id.

Parameters managedRecordId: Id {in}

getRecordKeeper: Id

Scope Public

Description This operation returns the id of the current record keeper of the record with the given id.

Parameters managedRecordId: Id {in}

getRecordKeeper: RecordKeeper

Scope Public

Description This operation returns the record keeper for the record with the given record id.

Parameters managedRecordId: Id {in}

getRecordKeeperHistory: RecordKeeper

Scope Public

Description This operation returns a list of record keepers of the record keepers of the record with the given id.

Parameters managedRecordId: Id {in}

getRecordLinks: ManagedRecordAssociation

Scope Public

Description This operation returns a list of managed record associations for the record with the given record id.

Parameters managedRecordId: Id {in}

removeAssociation: ack

Scope Public

Description This operation removes the ManagedRecordAssociationMember element from the ManagedRecordAssociation. If there is only one other ManagedRecord connected to the ManagedRecordAssociation then the association should be removed. The operation returns an acknowledgement of the success or failure of the operation.

Parameters associationId: Id {in}

removeCaseFilePart: ack

Scope Public

Description This operation removes the RecordPart with the given recordPartId from the ManagedRecord with the given recordId.

Parameters recordId: Id {in}
recordPartId: Id {in}

replaceCaseFilePart: ack

Scope Public

Description This operation replaces the RecordPart with an id of recordPartId on the managedRecord with the an id of recordId with a new RecordPart that points to a Document with an id of docId.

Parameters docId: Id {in}
recordId: Id {in}
recordPartId: Id {in}

setProvenance: ack

Scope Public

Description This operation sets the provenance for the record to the party with the given party id and returns an acknowledgement of the success or failure of the operation.. This operation needs to ensure that the RecordKeeper is within the organization that now has provenance.

Parameters managedRecordID: Id {in}
partyID: Id {in}

setRecordKeeper: ack

Scope Public

Description This operation sets the Party with the given id as the record keeper for the record with the given record id at the

assignment date. The operation returns an acknowledgement of the success or failure of the operation.

Parameters
partyId: Id {in}
managedRecordId: Id {in}
assignmentDate: TimeStamp {in}

setRecordKeeper: ack

Scope
Public

Description
This operation sets the record keeper id for the record with the given id to the party id passed to the operation. This operation must ensure that the Party associated with the PartyId parameter is in the Provenance Structure identified through the most recent ProvenanceAssociation. The operation returns an acknowledgement of the success or failure of the operation.

Parameters
recordID: Id {in}
partyID: Id {in}

transferRecord: ack

Scope
Public

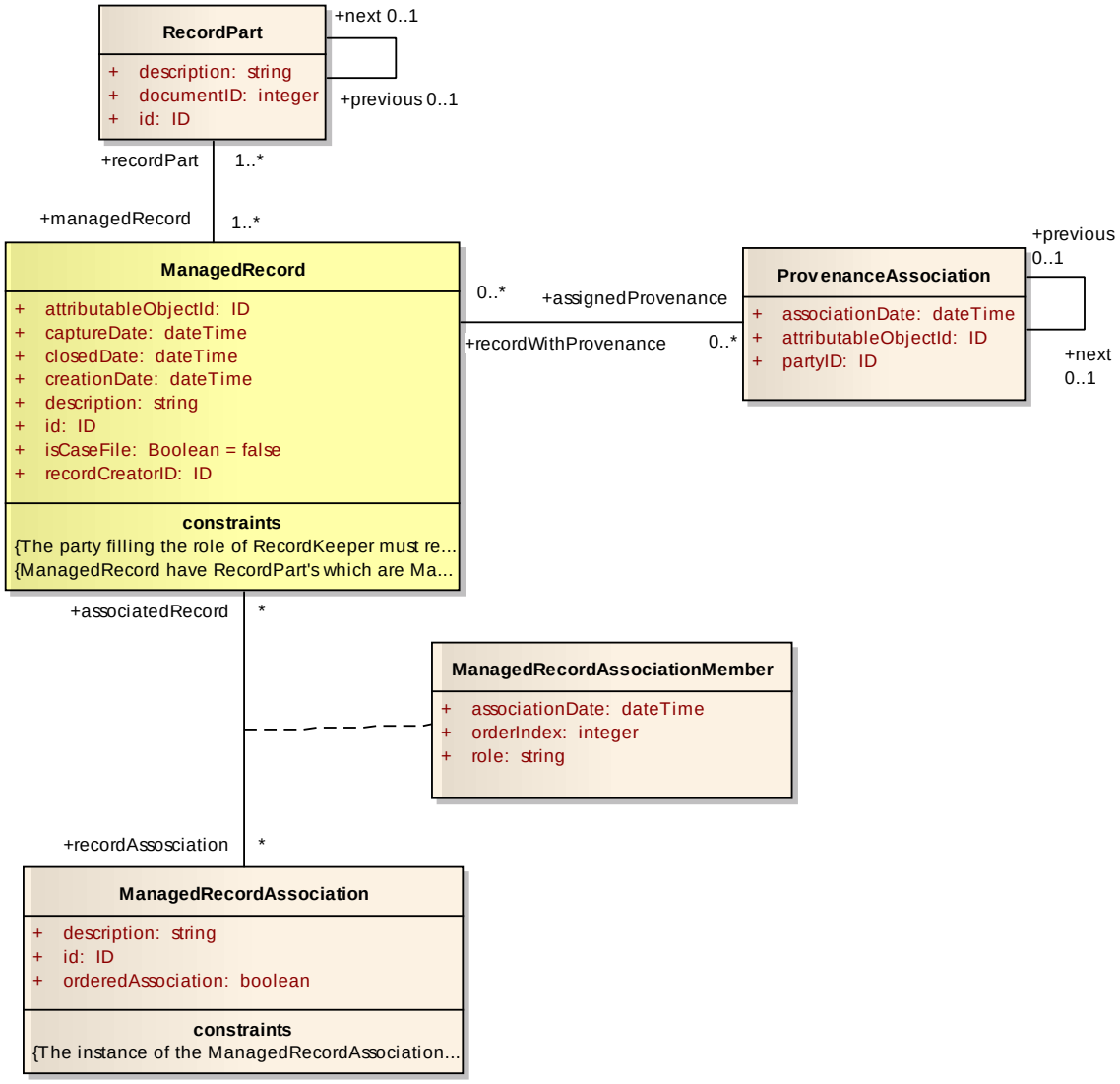
Description
This operation marks a record with the given id as being transferred as of the transfer date. The operation returns an acknowledgement of the success or failure of the operation.

Parameters
managedRecId: Id {in}
transferDate: TimeStamp {in}

8.6.3.6.4 Package: ManagedRecordsSIM

The ManagedRecordsSIM package contains the elements used to define the parameters and information management responsibilities of the ManagedRecords service.

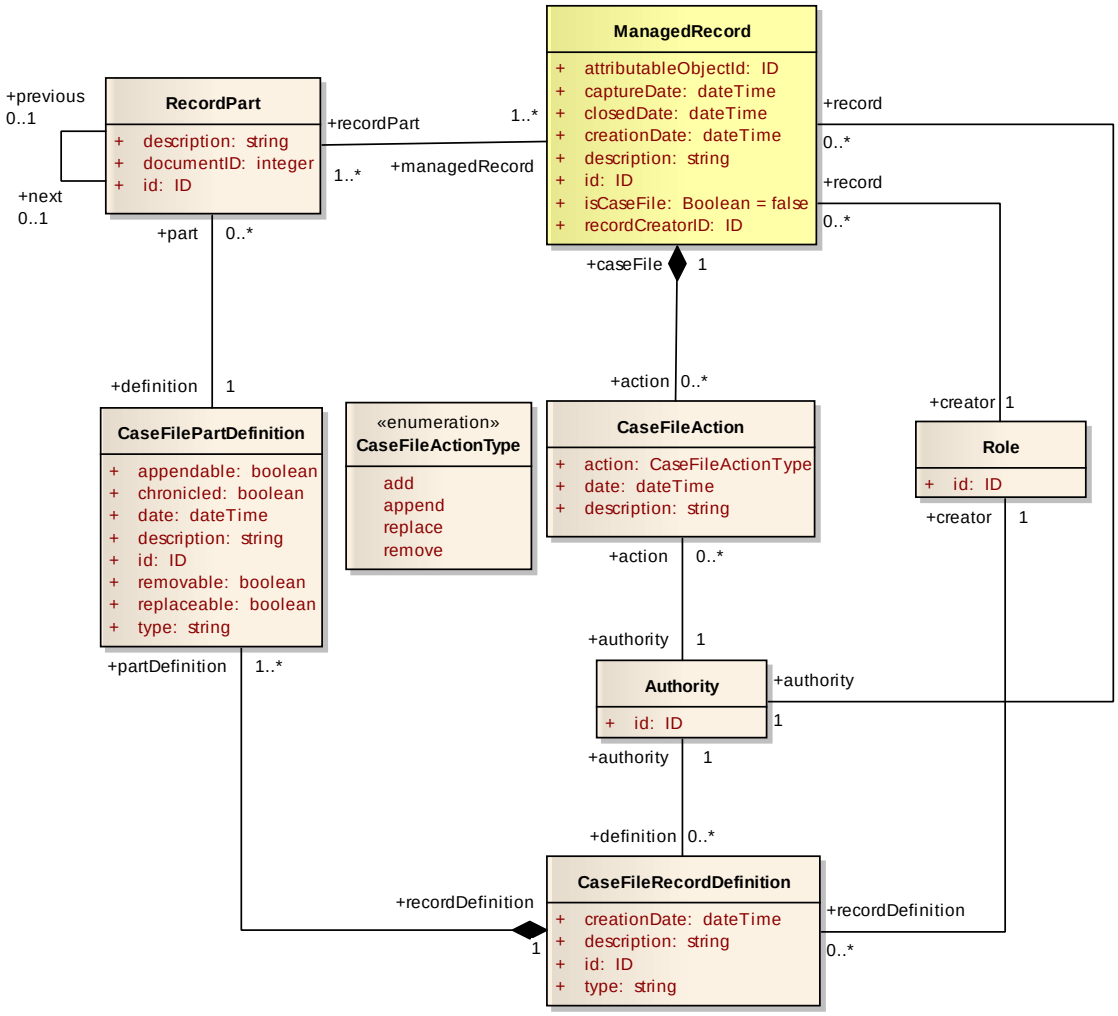
Managed Records SIM Static Structure



The Managed Records SIM Static Structure diagram shows the information elements related to Managed Records that this service must manage or use in referencing information managed by other services.

The instance of the ManagedRecordAssociation shall be destroyed after the last managed record disconnects from the association

Managed Records Case File SIM Static Structure



The Managed Records SIM Static Structure diagram shows information used or managed by the ManagedRecords service associated with ManagedRecords. Case files are a type of ManagedRecord wherein the RecordPart's may be allowed to be changed over time. Contrary to simple ManagedRecords, the RecordPart's of a ManagedRecord with isCaseFile set to "true" may be added, removed or replaced over the course of the active period of the record as stipulated in the CaseFilePartDefinition metadata.

8.6.3.6.4.1 Class: ManagedRecordsSIM::Authority

See the Party package of the RmsDomainModel for a definition of this element. The element is used here to maintain a reference to information maintained by the Authorities service.

Attributes

Attribute: Authority.id

Type: ID
Description: Unique identifier for the Authority.

Connections

Association

From Class: ManagedRecordsSIM::CaseFileRecordDefinition
In the Role of: definition
Multiplicity: 0..*

To Class: ManagedRecordsSIM::Authority
In the Role of: authority
Multiplicity: 1

Association

See the ManagedRecords package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ManagedRecord
In the Role of: record
Multiplicity: 0..*

To Class: ManagedRecordsSIM::Authority
In the Role of: authority
Multiplicity: 1

Association

From Class: ManagedRecordsSIM::CaseFileAction
In the Role of: action
Multiplicity: 0..*

To Class: ManagedRecordsSIM::Authority
In the Role of: authority
Multiplicity: 1

8.6.3.6.4.2 Class: ManagedRecordsSIM::CaseFileAction

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: CaseFileAction.action

Type: CaseFileActionType

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileAction.description

Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileAction.date

Type: dateTime
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Aggregation

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::CaseFileAction
In the Role of: action
Multiplicity: 0..*

To Class: ManagedRecordsSIM::ManagedRecord
In the Role of: caseFile
Multiplicity: 1

Association

From Class: ManagedRecordsSIM::CaseFileAction
In the Role of: action
Multiplicity: 0..*

To Class: ManagedRecordsSIM::Authority
In the Role of: authority
Multiplicity: 1

8.6.3.6.4.3 Enumeration: ManagedRecordsSIM::CaseFileActionType

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: CaseFileActionType.add

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileActionType.append

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileActionType.replace

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileActionType.remove

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

8.6.3.6.4.4 Class: ManagedRecordsSIM::CaseFilePartDefinition

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: CaseFilePartDefinition.id

Type: ID

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFilePartDefinition.type

Type: string

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFilePartDefinition.description

Type: string

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFilePartDefinition.date

Type: dateTime

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFilePartDefinition.appendable

Type: boolean

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFilePartDefinition.chronicled

Type: boolean

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFilePartDefinition.removable

Type: boolean
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFilePartDefinition.replaceable

Type: boolean
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Aggregation

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::CaseFilePartDefinition
In the Role of: partDefinition
Multiplicity: 1..*

To Class: ManagedRecordsSIM::CaseFileRecordDefinition
In the Role of: recordDefinition
Multiplicity: 1

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::CaseFilePart
In the Role of: part
Multiplicity: 0..*

To Class: ManagedRecordsSIM::CaseFilePartDefinition
In the Role of: definition
Multiplicity: 1

8.6.3.6.4.5 Class: ManagedRecordsSIM::CaseFileRecordDefinition

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: CaseFileRecordDefinition.id

Type: ID
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileRecordDefinition.type

Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileRecordDefinition.description

Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: CaseFileRecordDefinition.creationDate

Type: dateTime
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Association

From Class: ManagedRecordsSIM::CaseFileRecordDefinition
In the Role of: definition
Multiplicity: 0..*

To Class: ManagedRecordsSIM::Authority
In the Role of: authority
Multiplicity: 1

Aggregation

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::CaseFilePartDefinition
In the Role of: partDefinition
Multiplicity: 1..*

To Class: ManagedRecordsSIM::CaseFileRecordDefinition
In the Role of: recordDefinition
Multiplicity: 1

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class:	ManagedRecordsSIM::Role
In the Role of:	creator
Multiplicity:	1

To Class:	ManagedRecordsSIM::CaseFileRecordDefinition
In the Role of:	recordDefinition
Multiplicity:	0..*

8.6.3.6.4.6 Class: ManagedRecordsSIM::RecordKeeper

See the Party package of the RmsDomainModel for a definition of this element. This element is a placeholder to provide a reference to information managed by the Parties service.

Attributes

Attribute: RecordKeeper.assignmentDate

Type:	dateTime
Description:	See the Party package of the RmsDomainModel for a definition of this element.

Attribute: RecordKeeper.id

Type:	ID
Description:	Unique identifier of the record keeper.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class:	ManagedRecordsSIM::RecordKeeper
In the Role of:	theKeeper
Multiplicity:	0..*

To Class:	ManagedRecordsSIM::ManagedRecord
In the Role of:	keeps
Multiplicity:	0..*

8.6.3.6.4.7 Class: ManagedRecordsSIM::RecordPart

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: RecordPart.id

Type: ID
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: RecordPart.documentID

Type: integer
Description: Unique identifier of the document within the Documents service that comprises the content of the ManagedRecord.

Attribute: RecordPart.description

Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ManagedRecord
In the Role of: managedRecord
Multiplicity: 1..*

To Class: ManagedRecordsSIM::RecordPart
In the Role of: recordPart
Multiplicity: 1..*

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM: :RecordPart
In the Role of: previous
Multiplicity: 0.. 1

To Class: ManagedRecordsSIM:: RecordPart
In the Role of: next
Multiplicity: 0.. 1

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM: : RecordPart
In the Role of: part
Multiplicity: 0..*

To Class: ManagedRecordsSIM: :CaseFilePartDefinition
In the Role of: definition
Multiplicity: 1

Dependency

The ManagedRecords service is responsible for managing the information related to managed records and their record parts.

From Class: ManagedRecordsService::ManagedRecords

To Class: ManagedRecordsSIM::RecordPart

8.6.3.6.4.8 Class: ManagedRecordsSIM::Role

See the Party package of the RmsDomainModel for a definition of this element. This element is used as a reference to information maintained by the Parties service.

Attributes

Attribute: Role.id

Type: ID
Description: See the Party package of the RmsDomainModel for a definition of this element.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::Role
In the Role of: creator
Multiplicity: 1

To Class: ManagedRecordsSIM::ManagedRecord
In the Role of: record
Multiplicity: 0..*

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class:	ManagedRecordsSIM::Role
In the Role of:	creator
Multiplicity:	1

To Class:	ManagedRecordsSIM::CaseFileRecordDefinition
In the Role of:	recordDefinition
Multiplicity:	0..*

8.6.3.6.4.9 Class: ManagedRecordsSIM::ManagedRecord

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: ManagedRecord.id

Type: ID
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecord.isCaseFile

Type: boolean
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecord.captureDate

Type: dateTime
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecord.description

Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecord.recordCreatorID

Type: ID
Description: Unique identifier of the record creator as managed by the Parties service.

Attribute: ManagedRecord.attributableObjectId

Type: ID
Description: A reference to the Id of the AttributableObject that stores AtributeProfile information for the ManagedRecord.

Attribute: ManagedRecord.creationDate

Type: dateTime
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecord.closedDate

Type: dateTime
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ManagedRecord
In the Role of: managedRecord
Multiplicity: 1..*

To Class: ManagedRecordsSIM::RecordPart
In the Role of: recordPart
Multiplicity: 1..*

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ProvenanceAssociation
In the Role of: assignedProvenance
Multiplicity: 0..*

To Class: ManagedRecordsSIM::ManagedRecord
In the Role of: recordWithProvenance
Multiplicity: 0..*

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM: :ManagedRecord
In the Role of: record
Multiplicity: 0..*

To Class: ManagedRecordsSIM: :Authority

In the Role of: authority
Multiplicity: 1

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM: :Role
In the Role of: creator
Multiplicity: 1

To Class: ManagedRecordsSIM: : ManagedRecord
In the Role of: record
Multiplicity: 0..*

Aggregation

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM: :CaseFileAction
In the Role of: action
Multiplicity: 0..*

To Class: ManagedRecordsSIM: : ManagedRecord
In the Role of: caseFile
Multiplicity: 1

AssociationClass

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ManagedRecord
In the Role of: associatedRecord
Multiplicity: *

To Class: ManagedRecordsSIM::ManagedRecordAssociation
In the Role of: recordAssosciation
Multiplicity: *

Dependency

The Annotations service maintains references to information related to managed records.

From Class: AnnotationsService::Annotations

To Class: ManagedRecordsSIM::ManagedRecord

Dependency

The ManagedRecords service is responsible for managing the information related to managed records and their record parts.

From Class: ManagedRecordsService::ManagedRecords

To Class: ManagedRecordsSIM::ManagedRecord

8.6.3.6.4.10 Class: ManagedRecordsSIM::ManagedRecordAssociation

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: ManagedRecordAssociation.id

Type: ID
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecordAssociation.description

Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecordAssociation.orderedAssociation

Type: boolean
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Constraint Name: The instance of the ManagedRecordAssociation shall be destroyed after the last managed record disconnects from the association

AssociationClass

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ManagedRecord
In the Role of: associatedRecord
Multiplicity: *

To Class: ManagedRecordsSIM::ManagedRecordAssociation

In the Role of: recordAssosciation
Multiplicity: *

Dependency

The ManagedRecords service is responsible for managing the associations of related managed records.

From Class: ManagedRecordsService::ManagedRecords

To Class: ManagedRecordsSIM::ManagedRecordAssociation

8.6.3.6.4.11 AssociationClass: ManagedRecordsSIM::ManagedRecordAssociationMember

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: ManagedRecordAssociationMember.associationDate
Type: dateTime
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecordAssociationMember.role
Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecordAssociationMember.orderIndex
Type: integer
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Dependency

The ManagedRecords service is responsible for managing the associations of related managed records.

From Class: ManagedRecordsService::ManagedRecords

To AssociationClass: ManagedRecordsSIM::ManagedRecordAssociation Member

8.6.3.6.4.12 Class: ManagedRecordsSIM::ProvenanceAssociation

See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: ProvenanceAssociation.associationDate

Type: dateTime
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ProvenanceAssociation.partyID

Type: ID
Description: Unique identifier of the party with provenance of the particular managed record.

Attribute: ProvenanceAssociation.attributableObjectId

Type: ID
Description: A reference to the Id of the AttributableObject that stores AtributeProfile information for the ProvenanceAssociation.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ProvenanceAssociation
In the Role of: next
Multiplicity: 0..1

To Class: ManagedRecordsSIM::ProvenanceAssociation
In the Role of: previous
Multiplicity: 0..1

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: ManagedRecordsSIM::ProvenanceAssociation
In the Role of: assignedProvenance
Multiplicity: 0..*

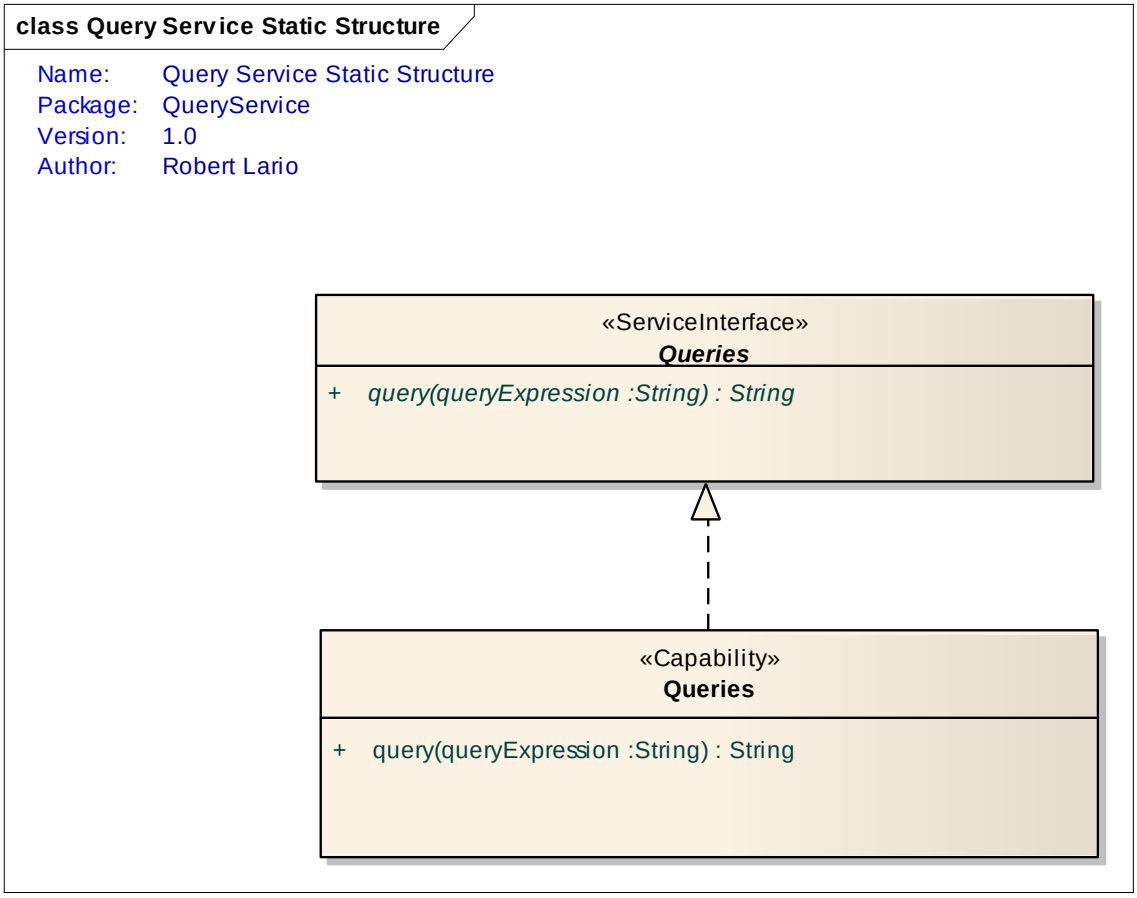
To Class: ManagedRecordsSIM::ManagedRecord
In the Role of: recordWithProvenance

Multiplicity: 0..*

8.6.3.7 Package: QueryService

Provides the capability of returning RECORDS MANAGEMENT entities base on their RmsDomainModel as specified in the query string. The returning parameter is also based upon the RmsDomainModel.

Query Service Static Structure



Implementations of the Queries service will need to realize the Queries ServiceInterface and the behaviors associated with the Queries Capability. A single operation called 'query' is defined. The query operation takes a string as an input parameter and returns the results as a string. The input parameter qualifies the requested elements and the return string contains the elements that match the request. Both the input parameter and return parameter structure is base upon the RmsDomainModel

The functionality provided herein simply allows for the use of the schema and application of its queries to ManagedRecords.

8.6.3.7.1 Class: QueryService::Queries

The Queries Capably specifies the required behavior and constraints of any implementations of the Queries ServiceInterface in a platform independent manner. It provides the ability query ManagedRecords based on the RmsDomainModel elements and relationships.

A single operation called 'query' is defined. The query operation takes a string as an input parameter and returns the results as a string. The input parameter qualifies the requested elements and the return string contains the elements that match the request.

Attributes

Connections

Realisation

From Class: QueryService::Queries

To Interface: QueryService::Queries

8.6.3.7.2 Interface: QueryService::Queries

The Queries ServiceInterface is a platform independent specification of the operation signatures of the Queries service. Refer to the Queries Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: QueryService::Queries

To Interface: QueryService::Queries

8.6.3.7.3 Capabilities: QueryService::Queries

query: String

Scope Public

Description This operation passes an XQuery/Xpath string to the Records Management Environment. The response is an XML string formatted according to the RMS XSD.

Parameters queryExpression: String {in}

\

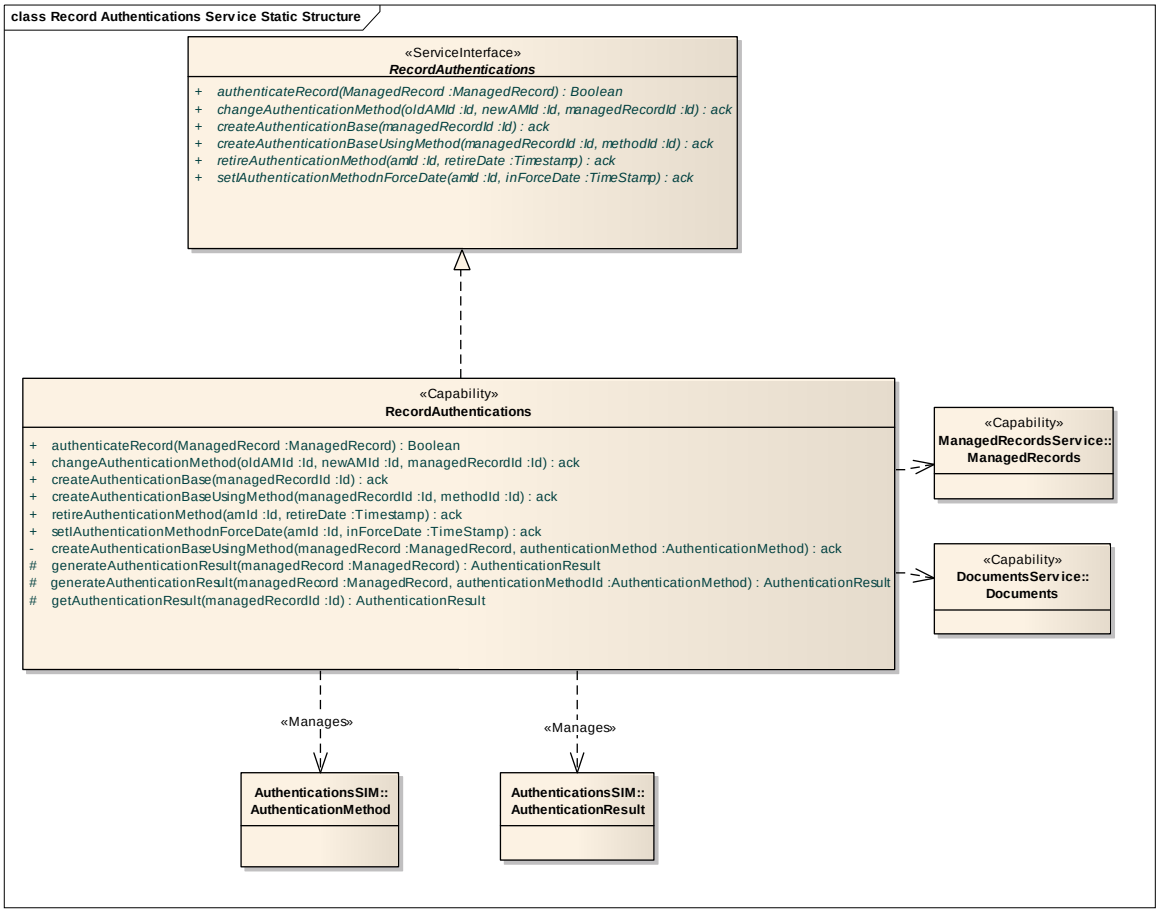
8.6.3.7.4 Package: QuerySIM

Refer to the RmsDomainModel for the elements used to define the parameters and information management responsibilities of the Queries service. See 'RmsDomainModel Package Structure'.

8.6.3.8 Package: RecordAuthenticationsService

The RecordAuthenticationsService package contains the model elements that together define the RecordAuthentications service.

Record Authentications Service Static Structure



Implementations of the RecordAuthentications service will need to realize the RecordAuthentications ServiceInterface and all the behaviors associated with the RecordAuthentications Capability. This includes management of authentication methods and the results of execution of those methods on managed records.

The service depends on the ManagedRecords service to get information relevant to the record to be authenticated. It also depends on the Documents service to get the actual contents of the record.

8.6.3.8.1 Class: RecordAuthenticationsService::RecordAuthentications

The RecordAuthentications Capably specifies the required behavior and constraints of any implementation of the RecordAuthentications ServiceInterface in a platform independent manner. This service provides the ability to manage authentication methods, to execute those authentication methods on managed records and to maintain the results of those authentications to enable the assessment of authenticity of a particular record.

The service depends on the ManagedRecords service to get information relevant to the record to be authenticated. It also depends on the Documents service to get the actual contents of the record.

Attributes

Connections

Dependency

The RecordAuthentications service maintains references to the documents associated with managed records that have been authenticated.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: DocumentsService::Documents

Dependency

The RecordAuthentications service maintains references to the managed records that have been authenticated.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: ManagedRecordsService::ManagedRecords

Dependency

The RecordAuthentications service is responsible for managing the information regarding the authentication methods used to verify the authenticity of managed records.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: AuthenticationsSIM::AuthenticationMethod

Dependency

The RecordAuthentications service is responsible for managing the information regarding the authentication methods used to verify the

authenticity of managed records. This includes the authentication results created while authenticating a record.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: AuthenticationsSIM::AuthenticationResult

Realisation

From Class: RecordAuthenticationsService::RecordAuthentications

To Interface: RecordAuthenticationsService::RecordAuthentications

Dependency

RMS Clients will use the RecordAuthentications service to manage authentication methods and results and to assess the authenticity of records within RMS.

From Class: RmsSolution::RMS Client

To Class: RecordAuthenticationsService::RecordAuthentications

8.6.3.8.2 Interface: RecordAuthenticationsService::RecordAuthentications

The RecordAuthentications ServiceInterface is a platform independent specification of the operation signatures of the RecordAuthentications service. Refer to the RecordAuthentications Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: RecordAuthenticationsService::RecordAuthentications

To Interface: RecordAuthenticationsService::RecordAuthentications

8.6.3.8.3 Capabilities: RecordAuthenticationsService::RecordAuthentications

authenticateRecord: Boolean

Scope Public

Description This operation calculates the AuthenticationResult for the ManagedRecord and compares it to the AuthenticationBase. It returns Boolean true if they are the same, and false if they are not.

Parameters ManagedRecord: ManagedRecord {in}

changeAuthenticationMethod: ack

Scope Public

Description This operation enables the change of AuthenticationMethod of a ManagedRecord from an old AuthenticationMethod to the current AuthenticationMethod. It causes the generation of a new AuthenticationBase using the new AuthenticationMethod. The operation returns an acknowledgement indicating success or failure.

Parameters oldAMId: Id {in}
newAMId: Id {in}
managedRecordId: Id {in}

createAuthenticationBase: ack

Scope Public

Description This operation calculates the AuthenticationBase for the ManagedRecord identified by the managedRecordId using the current AuthenticationMethod. The operation returns an acknowledgement indicating success or failure.

Parameters managedRecordId: Id {in}

createAuthenticationBaseUsingMethod: ack

Scope Public

Description The operation establishes a new AuthenticationBase for a ManagedRecord using a particular AuthenticationMethod passed as its ID. The operation returns an acknowledgement indicating success or failure.

Parameters managedRecordId: Id {in}
methodId: Id {in}

createAuthenticationBaseUsingMethod: ack

Scope	Private
Description	The operation establishes a new AuthenticationBase for a ManagedRecord using a particular AuthenticationMethod which is passed as a parameter. The operation returns an acknowledgement indicating success or failure.
Parameters	managedRecord: ManagedRecord {in} authenticationMethod: AuthenticationMethod {in}

generateAuthenticationResult: AuthenticationResult

Scope	Protected
Description	This operation returns an AuthenticationResult calculated using the current AuthenticationMethod.
Parameters	managedRecord: ManagedRecord {in}

generateAuthenticationResult: AuthenticationResult

Scope	Protected
Description	This operation returns an AuthenticationResult calculated using the specified AuthenticationMethod.
Parameters	managedRecord: ManagedRecord {in} authenticationMethodId: AuthenticationMethod {in}

getAuthenticationResult: AuthenticationResult

Scope	Protected
Description	This operation returns the AutheticationResult for the ManagedRecord.
Parameters	managedRecordId: Id {in}

retireAuthenticationMethod: ack

Scope	Public
Description	This operation is used to mark the retirement date of an AuthenticationMethod. The operation returns an acknowledgement indicating success or failure.
Parameters	amId: Id {in} retireDate: Timestamp {in}

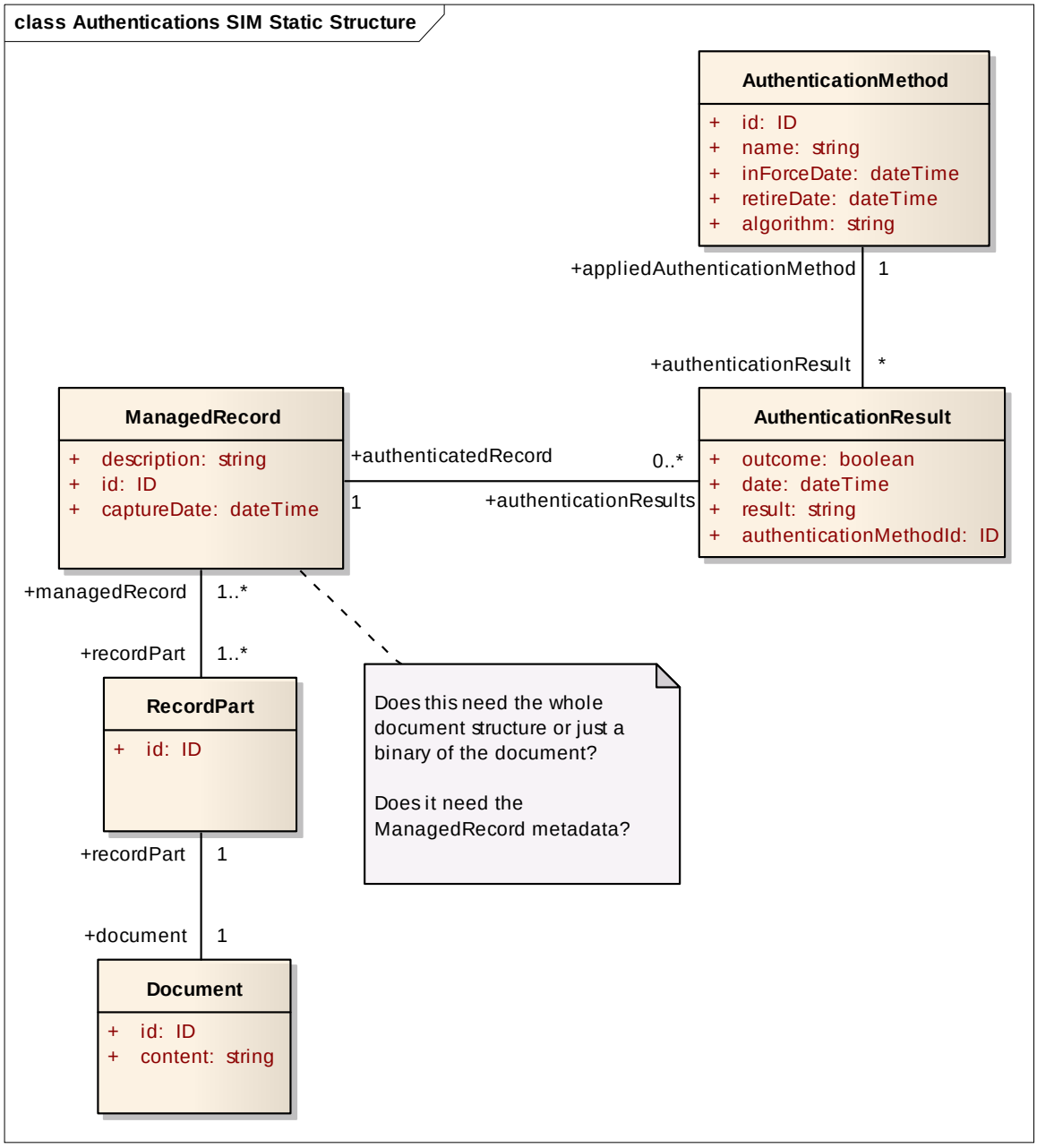
setIAuthenticationMethodnForceDate: ack

Scope	Public
Description	This operation marks when a new AuthenticationMethod will be put inForce. The operation returns an acknowledgement indicating success or failure.
Parameters	amId: Id {in} inForceDate: TimeStamp {in}

8.6.3.8.4 Package: AuthenticationsSIM

The AuthenticationsSIM package contains the elements used to define the parameters and information management responsibilities of the Authentications service.

Authentications SIM Static Structure



The Authentications SIM Static Structure diagram shows the information elements related to Authentications that this service must manage or use in referencing information managed by other services.

8.6.3.8.4.1 Class: AuthenticationsSIM::Document

See the Document package of the RmsDomainModel for a definition of this element. The data associated with documents is maintained by the Documents service.

Attributes

Attribute: Document.id

Type: ID
Description: See the Document package of the RmsDomainModel for a definition of this element.

Attribute: Document.content

Type: string
Description: See the Document package of the RmsDomainModel for a definition of this element.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::RecordPart
In the Role of: recordPart
Multiplicity: 1

To Class: AuthenticationsSIM::Document
In the Role of: document
Multiplicity: 1

8.6.3.8.4.2 Class: AuthenticationsSIM::ManagedRecord

See the ManagedRecord package of the RmsDomainModel for a definition of this element. The data associated with Managed Records is maintained by the ManagedRecords service.

Attributes

Attribute: ManagedRecord.description

Type: string
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecord.id

Type: ID
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Attribute: ManagedRecord.captureDate

Type: dateTime

Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::ManagedRecord
In the Role of: managedRecord
Multiplicity: 1..*

To Class: AuthenticationsSIM::RecordPart
In the Role of: recordPart
Multiplicity: 1..*

Association

See the Authenticity package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::AuthenticationResult
In the Role of: authenticationResults
Multiplicity: 0..*

To Class: AuthenticationsSIM::ManagedRecord
In the Role of: authenticatedRecord
Multiplicity: 1

8.6.3.8.4.3 Class: AuthenticationsSIM::RecordPart

See the ManagedRecord package of the RmsDomainModel for a definition of this element. The data associated with record parts is maintained by the ManagedRecords service.

Attributes

Attribute: RecordPart.id

Type: ID
Description: See the ManagedRecord package of the RmsDomainModel for a definition of this element.

Connections

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::RecordPart
In the Role of: recordPart
Multiplicity: 1

To Class: AuthenticationsSIM::Document
In the Role of: document
Multiplicity: 1

Association

See the ManagedRecord package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::ManagedRecord
In the Role of: managedRecord
Multiplicity: 1..*

To Class: AuthenticationsSIM::RecordPart
In the Role of: recordPart
Multiplicity: 1..*

8.6.3.8.4.4 Class: AuthenticationsSIM::AuthenticationMethod

See the Authenticity package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: AuthenticationMethod.id
Type: ID
Description: Unique identifier for the authentication method.

Attribute: AuthenticationMethod.name
Type: string
Description: See the Authenticity package of the RmsDomainModel for a definition of this element.

Attribute: AuthenticationMethod.inForceDate
Type: dateTime
Description: See the Authenticity package of the RmsDomainModel for a definition of this element.

Attribute: AuthenticationMethod.retireDate
Type: dateTime
Description: See the Authenticity package of the RmsDomainModel for a definition of this element.

Attribute: AuthenticationMethod.algorithm

Type: string
Description: See the Authenticity package of the RmsDomainModel for a definition of this element.

Connections

Dependency

The RecordAuthentications service is responsible for managing the information regarding the authentication methods used to verify the authenticity of managed records.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: AuthenticationsSIM::AuthenticationMethod

Association

See the Authenticity package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::AuthenticationMethod
In the Role of: appliedAuthenticationMethod
Multiplicity: 1

To Class: AuthenticationsSIM::AuthenticationResult
In the Role of: authenticationResult
Multiplicity: *

8.6.3.8.4.5 Class: AuthenticationsSIM::AuthenticationResult

See the Authenticity package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: AuthenticationResult.outcome

Type: boolean
Description: See the Authenticity package of the RmsDomainModel for a definition of this element.

Attribute: AuthenticationResult.date

Type: dateTime
Description: See the Authenticity package of the RmsDomainModel for a definition of this element.

Attribute: AuthenticationResult.result

Type: string

Description: See the Authenticity package of the RmsDomainModel for a definition of this element.

Attribute: AuthenticationResult.authenticationMethodId

Type: ID
Description: Unique identifier of the method used to generate this result.

Connections

Dependency

The RecordAuthentications service is responsible for managing the information regarding the authentication methods used to verify the authenticity of managed records. This includes the authentication results created while authenticating a record.

From Class: RecordAuthenticationsService::RecordAuthentications

To Class: AuthenticationsSIM::AuthenticationResult

Association

See the Authenticity package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::AuthenticationMethod
In the Role of: appliedAuthenticationMethod
Multiplicity: 1

To Class: AuthenticationsSIM::AuthenticationResult
In the Role of: authenticationResult
Multiplicity: *

Association

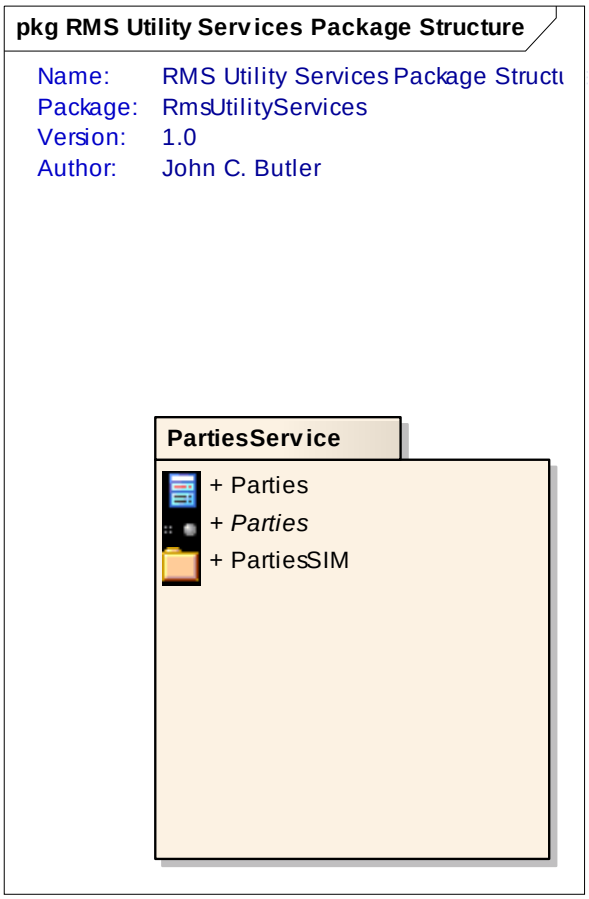
See the Authenticity package of the RmsDomainModel for a definition of this association.

From Class: AuthenticationsSIM::AuthenticationResult
In the Role of: authenticationResults
Multiplicity: 0..*

To Class: AuthenticationsSIM::ManagedRecord
In the Role of: authenticatedRecord
Multiplicity: 1

8.6.4 Package: RmsUtilityServices

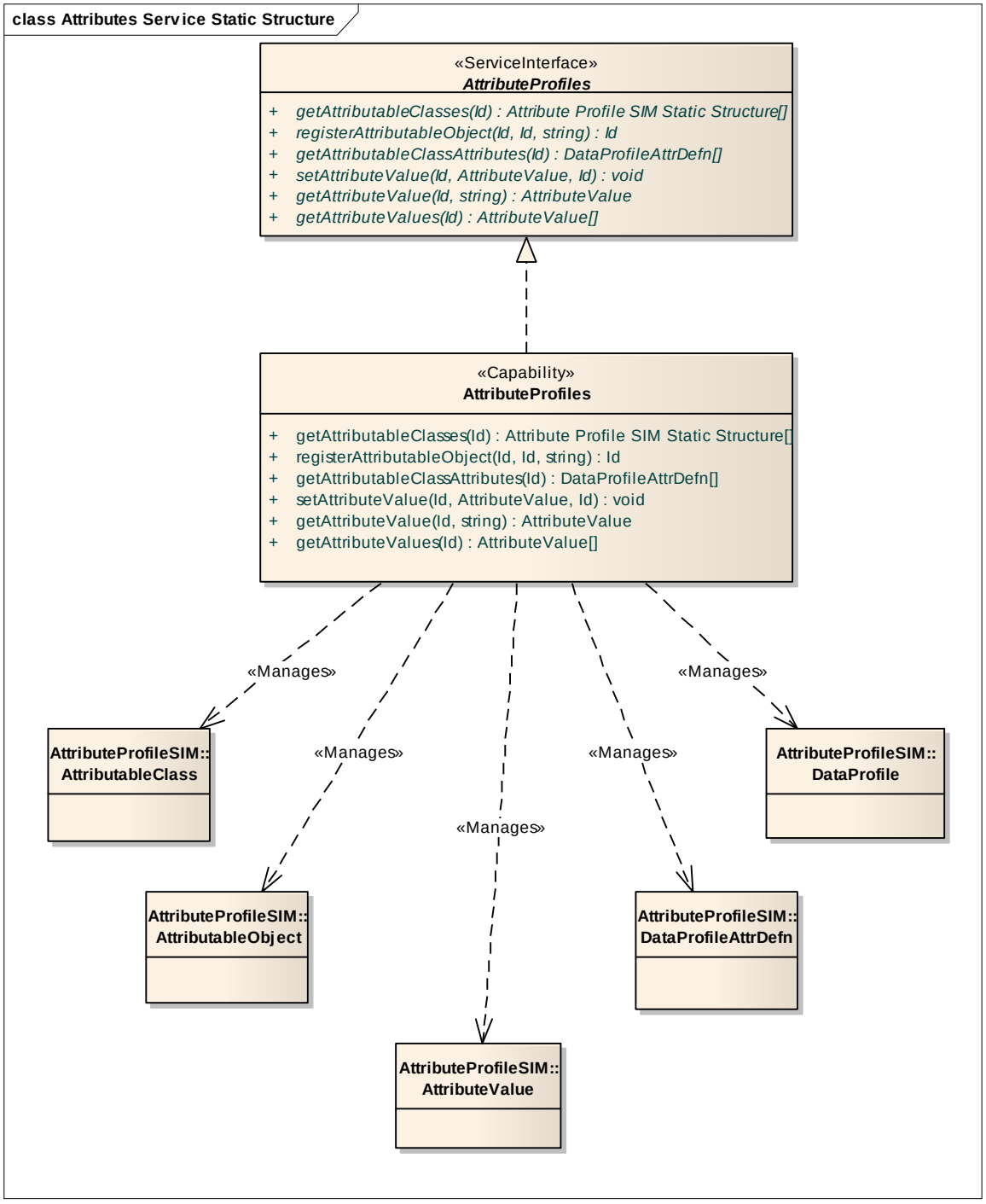
RMS Utility Services Package Structure



8.6.4.1 Package: AttributeProfiles Service

The AttributeProfilesService package contains the model elements that together define the AttributeProfiles service.

Attributes Service Static Structure



Implementations of the AttributeProfiles service will need to realize the AttributeProfiles ServiceInterface and all the behaviors associated with the AttributeProfiles Capability. The AttributeProfiles service maintains the instance data for profiles that have been added to the records management services. This service is not responsible for entering

the profiles themselves. It's responsibilities are limited to providing information contained in existing profiles and for storing instance data for attributable objects.

8.6.4.1.1 Class: AttributeProfiles Service::AttributeProfiles

The AttributeProfiles Capably specifies the required behavior and constraints of any implementation of the AttributeProfiles ServiceInterface in a platform independent manner. Attribute profiles enable the dynamic addition of metadata to managed records based on profiles set up in the system. Objects are registered with the AttributeProfiles service so that attributes can be set for that particular opject based on the profile.

Attributes

Connections

Dependency

From Class: AnnotationsService::Annotations

To Class: AttributeProfiles Service::AttributeProfiles

Dependency

From Class: ManagedRecordsService::ManagedRecords

To Class: AttributeProfiles Service::AttributeProfiles

Dependency

The AttributeProfiles service is responsible for managing the information regarding AttributableClasses - the classes of object that are referenced in profiles..

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::AttributableClass

Dependency

The AttributeProfiles service is responsible for managing the specific values for attributes of an AttributableObject.

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::AttributeValue

Dependency

The AttributeProfiles service is responsible for managing the information regarding Data Profile Attribute Definitions.

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::DataProfileAttrDefn

Dependency

The AttributeProfiles service is responsible for managing the information regarding Data Profile definitions.

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::DataProfile

Dependency

The AttributeProfiles service is responsible for managing the information regarding specific objects, AttributableObjects, that have attribute values set.

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::AttributableObject

Realisation

From Class: AttributeProfiles Service::AttributeProfiles

To Interface: AttributeProfiles Service::AttributeProfiles

8.6.4.1.2 Interface: AttributeProfiles Service::AttributeProfiles

The AttributeProfiles's ServiceInterface is a platform independent specification of the operation signatures of the AttributeProfiles's service. Refer to the AttributeProfiles's Capability for definitions of the service operations.

Attributes

Connections

Realisation

From Class: AttributeProfiles Service::AttributeProfiles

To Interface: AttributeProfiles Service::AttributeProfiles

8.6.4.1.3 Capabilities: AttributeProfiles Service::AttributeProfiles

getAttributableClassAttributes: DataProfileAttrDefn

Scope	Public
Description	This operation returns an array of the attributes definitions for a particular attributable class and profile.
Parameters	attributableClassId: Id {in}

getAttributableClasses: Attribute Profile SIM Static Structure

Scope	Public
Description	This operation returns an array of AttributableClass's in the system for a given data profile.
Parameters	dataProfileId: Id {in}

getAttributeValue: AttributeValue

Scope	Public
Description	This operation returns the attribute value with the given name.
Parameters	attribObjectId: Id {in} attributeName: string {in}

getAttributeValues: AttributeValue

Scope	Public
Description	This operation returns an array of attribute values for the attributable object with the given id.
Parameters	attribObjectId: Id {in}

registerAttributableObject: Id

Scope	Public

Description	This operation provides the ability to register an object so that additional attributes can be maintained for it.
Parameters	attributableClassId: Id {in} objectId: Id {in} objectType: string {in}

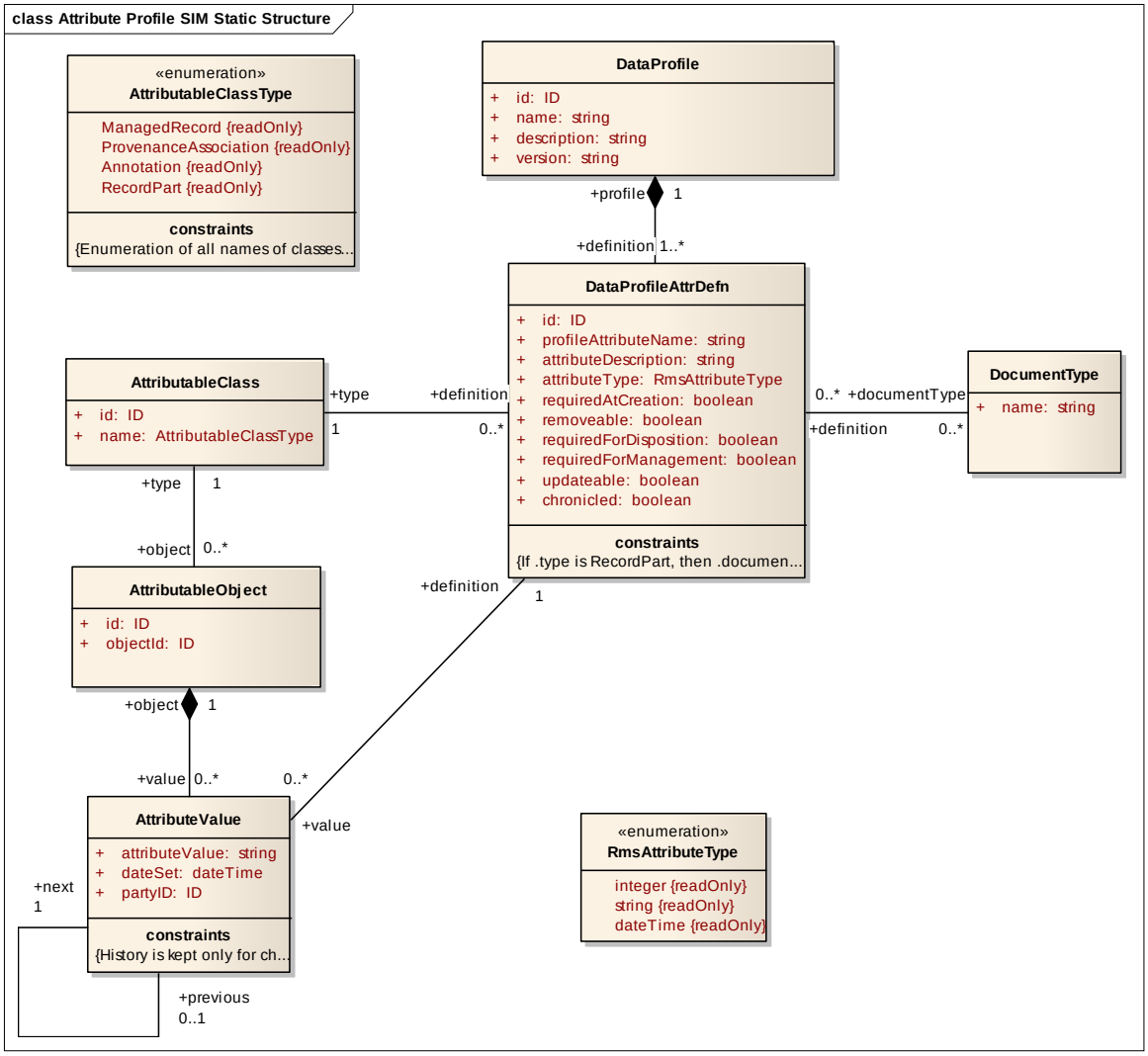
setAttributeValue: void

Scope	Public
Description	This operation provides the ability to set an attribute value for an attributable object.
Parameters	attribObjId: Id {in} attributeValue: AttributeValue {in} profileAttrDefnId: Id {in}

8.6.4.1.4 Package: AttributeProfileSIM

The AttributeProfileSIM package contains the elements used to define the parameters and information management responsibilities of the AttributeProfiles service.

Attribute Profile SIM Static Structure



The AttributeProfile Service SIM Static Structure diagram shows the information elements that comprise parameters or information that the AttributeProfile service must manage.

See the AttributeProfile Package for definitions of these elements.

8.6.4.1.4.1 Class: AttributeProfileSIM::AttributableClass

The specialization class type of the Attributable object which aggregates the set of attributes as defined by the DataProfileAttrDefn's that can be assigned as AttributeValue's to an AttributableObject.

Attributes

Attribute: AttributableClass.id

Type: ID
Description: Unique Identifier

Attribute: AttributableClass.name

Type: AttributableClassType
Description: The name of the records management domain classes whose objects can be assigned AttributeValue's if defined in a DataProfile.

Connections

Dependency

The AttributeProfiles service is responsible for managing the information regarding AttributableClasses - the classes of object that are referenced in profiles..

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::AttributableClass

Association

Indicates the object type to which the DataProfileAttrDefn applies.

From Class: AttributeProfileSIM::DataProfileAttrDefn
In the Role of: definition
Multiplicity: 0..*

To Class: AttributeProfileSIM::AttributableClass
In the Role of: type
Multiplicity: 1

Association

The classtype of the AttributableObject. The type determined by reflection must match that in RMSAttributableClassTypes. (Supporting both reflective and non-reflective languages).

From Class: AttributeProfileSIM::AttributableObject
In the Role of: object
Multiplicity: 0..*

To Class: AttributeProfileSIM::AttributableClass
In the Role of: type
Multiplicity: 1

8.6.4.1.4.2 Enumeration: AttributeProfileSIM::AttributableClassType

The classes whose instances can be attributed through an RMSDataProfile, i.e., be assigned AttributeValue's.

Attributes

Attribute: AttributableClassType.ManagedRecord
Type: string

Attribute: AttributableClassType.ProvenanceAssociation
Type: string

Attribute: AttributableClassType.Annotation
Type: string

Attribute: AttributableClassType.RecordPart
Type: string

Connections

Constraint Name: Enumeration of all names of classes that are subtypes of AttributableObject

8.6.4.1.4.3 Class: AttributeProfileSIM::AttributableObject

An object that can be attributed through the AttributeProfile services.

Attributes

Attribute: AttributableObject.id
Type: ID

Attribute: AttributableObject.objectId
Type: ID

Connections

Aggregation

The collection of an objects attributes.

From Class:	AttributeProfileSIM::AttributeValue
In the Role of:	value
Multiplicity:	0..*
Description:	The value of an object attribute.

To Class:	AttributeProfileSIM::AttributableObject
In the Role of:	object
Multiplicity:	1

Description: The attributed object.

Association

The classtype of the AttributableObject. The type determined by reflection must match that in RMSAttributableClassTypes. (Supporting both reflective and non-reflective languages).

From Class: AttributeProfileSIM::AttributableObject
In the Role of: object
Multiplicity: 0..*

To Class: AttributeProfileSIM::AttributableClass
In the Role of: type
Multiplicity: 1

Dependency

The AttributeProfiles service is responsible for managing the information regarding specific objects, AttributableObjects, that have attribute values set.

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::AttributableObject

8.6.4.1.4.4 Class: AttributeProfileSIM::AttributeValue

A value of an attribute associated with an AttributableObject.

Attributes

Attribute: AttributeValue.attributeValue
Type: string
Description: The string representing the value of the attribute of the AttributableObject

Attribute: AttributeValue.dateSet
Type: dateTime
Description: The date/time that the value of the AttributableObject was set.

Attribute: AttributeValue.partyID
Type: ID

Connections

Constraint Name: History is kept only for chronicled attributes. Those whose RMSDataProfileAttribute.chronicled = True

Association

The RMS attribute definition on which the AttributeValue is based.

From Class:	AttributeProfileSIM::AttributeValue
In the Role of:	value
Multiplicity:	0..*
Description:	The AttributeValue which is based on the RMSAttributeDefn

To Class:	AttributeProfileSIM::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	1
Description:	The RMSAttributeDefn on which the AttributeValue is based.

Association

From Class:	AttributeProfileSIM::AttributeValue
In the Role of:	next
Multiplicity:	1

To Class:	AttributeProfileSIM::AttributeValue
In the Role of:	previous
Multiplicity:	0..1

Aggregation

The collection of an objects attributes.

From Class:	AttributeProfileSIM::AttributeValue
In the Role of:	value
Multiplicity:	0..*
Description:	The value of an object attribute.

To Class:	AttributeProfileSIM::AttributableObject
In the Role of:	object
Multiplicity:	1
Description:	The attributed object.

Dependency

The AttributeProfiles service is responsible for managing the specific values for attributes of an AttributableObject.

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::AttributeValue

8.6.4.1.4.5 Class: AttributeProfileSIM::DataProfile

A profile of attribute definitions that may apply to AttributableObject's under organizational, ad hoc, or de jure standards or conventions.

Attributes

Attribute: DataProfile.id
Type: ID

Attribute: DataProfile.name
Type: string
Description: The unique name of the data profile.

Attribute: DataProfile.description
Type: string
Description: Textual description of the DataProfile

Attribute: DataProfile.version
Type: string
Description: The version of the DataProfile

Connections

Aggregation

Collects the DataProfileAttrDefn's that apply to this DataProfile.

From Class: AttributeProfileSIM::DataProfileAttrDefn
In the Role of: definition
Multiplicity: 1..*
Description: The definition of an attribute that is a member of the DataProfile

To Class: AttributeProfileSIM::DataProfile
In the Role of: profile
Multiplicity: 1
Description: The DataProfile of which the DataProfileAttrDefn is a member.

Dependency

The AttributeProfiles service is responsible for managing the information regarding Data Profile definitions.

From Class: AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::DataProfile

8.6.4.1.4.6 Class: AttributeProfileSIM::DataProfileAttrDefn

A member of a DataProfile that describes an attribute that is applicable to a specific AttributableClassType.

If the object specified by a DataProfileAttrDefn is a RecordPart, it must specify oneor more DocumentType's to which the attribute applies.

Attributes

Attribute: DataProfileAttrDefn.id

Type: ID
Description: Unique Identifier

Attribute: DataProfileAttrDefn.profileAttributeName

Type: string
Description: The name of the profile, unique in the context of its DataProfile. There may be multiple DataProfileAttrDefn's with the same name if they are in different DataProfile's

Attribute: DataProfileAttrDefn.attributeDescription

Type: string
Description: Textual description of the attribute.

Attribute: DataProfileAttrDefn.attributeType

Type: RmsAttributeType
Description: The AttributableClassType to which the DataProfileAttrDefn applies.

Attribute: DataProfileAttrDefn.requiredAtCreation

Type: boolean
Description: If "True", the object must be provided an AttributeValue conformant to this definition at time of creation.

Attribute: DataProfileAttrDefn.removeable

Type: boolean
Description: If "True", the object's AttributeValue conformant to this definition may be removed (deleted).

Attribute: DataProfileAttrDefn.requiredForDisposition

Type: boolean
Description: If "True", the object's AttributeValue conformant to this definition must be present before final disposition (Transfer

or Destroy) can be performed on the ManagedRecord associated with an object with this AttributeValue.

Attribute: DataProfileAttrDefn.requiredForManagement

Type: boolean
Description: If "True", the object's AttributeValue conformant to this definition must be present for management of the ManagedRecord associated with an object with this AttributeValue.

Attribute: DataProfileAttrDefn.updateable

Type: boolean
Description: If "True", the object's AttributeValue conformant to this definition may be updated. In the case that the value is not chronicled, the .attributeValue may be changed with new .dateSet, and .party. In the case that the value is chronicled, a new AttributeValue of the same RMSAttributeDefn is created and linked to the previous one through the next/previous association.

Attribute: DataProfileAttrDefn.chronicled

Type: boolean
Description: When .chronicled and .updateable = "True" , a new AttributeValue of the same RMSAttributeDefn may be created and linked to the previous one through the next/previous association.

Connections

Constraint Name: If .type is RecordPart, then .documentType must point to one or more DocumentType's

Association

If the DataProfileAttrDefn pertains to a RecordPart, then one or more DocumentType's are specified. If the RecordPart has a Document of one of those DocumentType's then the DataProfileAttrDefn applies to that RecordPart.

From Class: AttributeProfileSIM::DataProfileAttrDefn
In the Role of: definition
Multiplicity: 0..*

To Class: AttributeProfileSIM::DocumentType
In the Role of: documentType
Multiplicity: 0..*

Association

The RMS attribute definition on which the AttributeValue is based.

From Class:	AttributeProfileSIM::AttributeValue
In the Role of:	value
Multiplicity:	0..*
Description:	The AttributeValue which is based on the RMSAttributeDefn

To Class:	AttributeProfileSIM::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	1
Description:	The RMSAttributeDefn on which the AttributeValue is based.

Aggregation

Collects the DataProfileAttrDefn's that apply to this DataProfile.

From Class:	AttributeProfileSIM::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	1..*
Description:	The definition of an attribute that is a member of the DataProfile

To Class:	AttributeProfileSIM::DataProfile
In the Role of:	profile
Multiplicity:	1
Description:	The DataProfile of which the DataProfileAttrDefn is a member.

Association

Indicates the object type to which the DataProfileAttrDefn applies.

From Class:	AttributeProfileSIM::DataProfileAttrDefn
In the Role of:	definition
Multiplicity:	0..*

To Class:	AttributeProfileSIM::AttributableClass
In the Role of:	type
Multiplicity:	1

Dependency

The AttributeProfiles service is responsible for managing the information regarding Data Profile Attribute Definitions.

From Class:	AttributeProfiles Service::AttributeProfiles

To Class: AttributeProfileSIM::DataProfileAttrDefn

8.6.4.1.4.7 Class: AttributeProfileSIM::DocumentType

Attributes

Attribute: DocumentType.name
Type: string

Connections

Association

If the DataProfileAttrDefn pertains to a RecordPart, then one or more DocumentType's are specified. If the RecordPart has a Document of one of those DocumentType's then the DataProfileAttrDefn applies to that RecordPart.

From Class: AttributeProfileSIM::DataProfileAttrDefn
In the Role of: definition
Multiplicity: 0..*

To Class: AttributeProfileSIM::DocumentType
In the Role of: documentType
Multiplicity: 0..*

8.6.4.1.4.8 Enumeration: AttributeProfileSIM::RmsAttributeType

The type of the attribute.

Attributes

Attribute: RmsAttributeType.integer
Type: string

Attribute: RmsAttributeType.string
Type: string

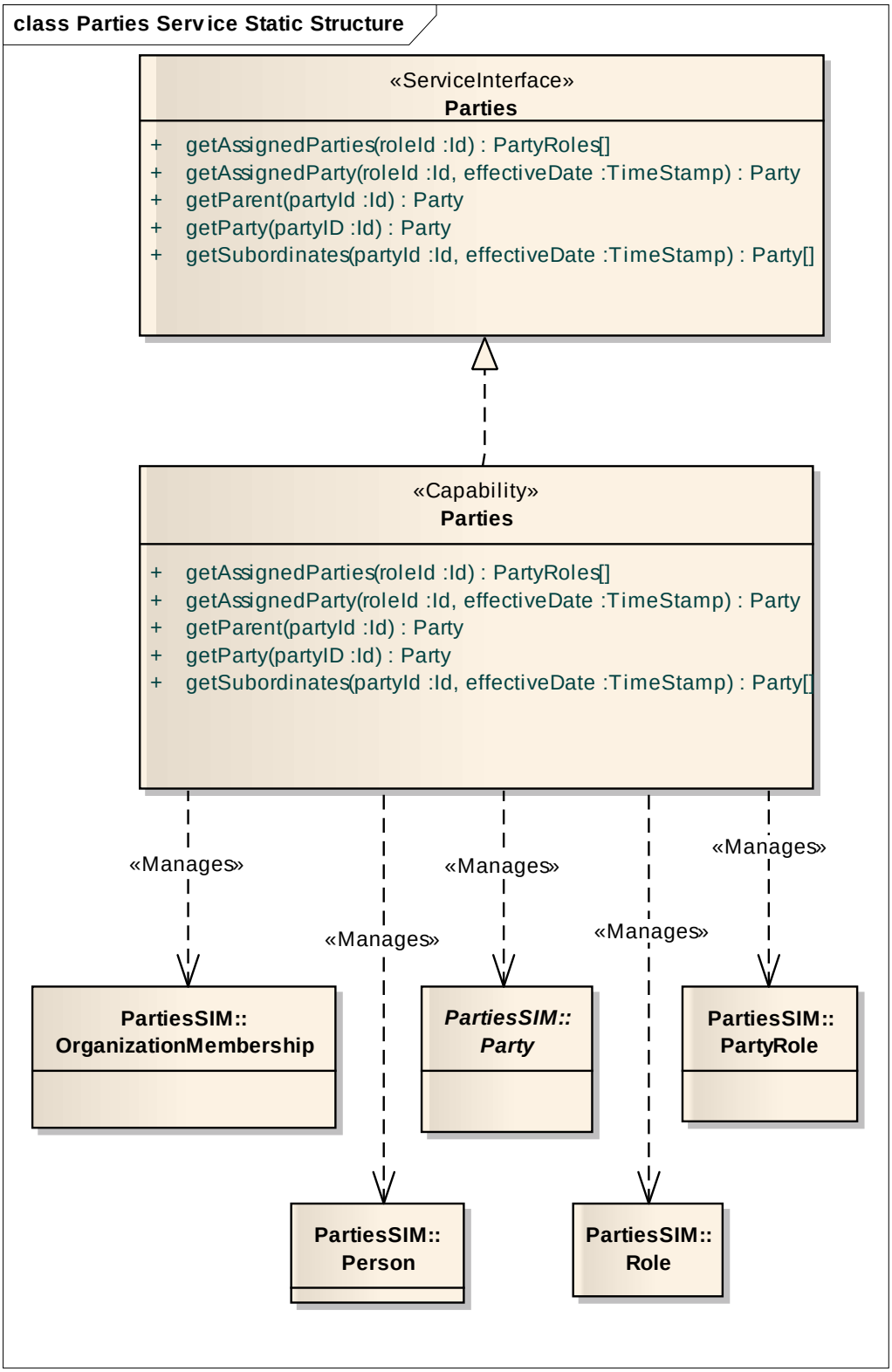
Attribute: RmsAttributeType.dateTime
Type: string

Connections

8.6.4.2 Package: PartiesService

The PartiesService package contains the model elements that together define the Parties service.

Parties Service Static Structure



Implementations of the AttributeProfiles service will need to realize the AttributeProfiles ServiceInterface and all the behaviors associated with the AttributeProfiles Capability. The AttributeProfiles service maintains the instance data for profiles that have been added to the records management services. This service is not responsible for entering the profiles themselves. It's responsibilities are limited to providing information contained in existing profiles and for storing instance data for attributable objects.

8.6.4.2.1 Class: PartiesService::Parties

Attributes

Connections

Dependency

The Annotations service maintains references to the parties that create annotations and apply them to records.

From Class: AnnotationsService::Annotations

To Class: PartiesService::Parties

Dependency

The Dispositions service maintains references to information maintained by the Parties service such as the creator of a disposition instruction and the organization that represents the destination of a move.

From Class: DispositionsService::Dispositions

To Class: PartiesService::Parties

Dependency

The ManagedRecords service maintains references to the party that created a record, that has provenance over the record, and that is currently keeping the record.

From Class: ManagedRecordsService::ManagedRecords

To Class: PartiesService::Parties

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To Class: PartiesSIM::Role

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To Class: PartiesSIM::Person

Dependency

The Categories service maintains references to information managed in by the Parties for the role that performs a particular business activity that generated a record.

From Class: CategoriesService::Categories

To Class: PartiesService::Parties

Dependency

The Authorities service uses the Parties service to managed the data about the Authority.

From Class: AuthoritiesService::Authorities

To Class: PartiesService::Parties

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To AssociationClass: PartiesSIM::PartyRole

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To AssociationClass: PartiesSIM::OrganizationMembership

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To Class: PartiesSIM::Party

Realisation

From Class: PartiesService::Parties

To Interface: PartiesService::Parties

8.6.4.2.2 Interface: PartiesService::Parties

Attributes

Connections

Realisation

From Class: PartiesService::Parties

To Interface: PartiesService::Parties

8.6.4.2.3 Capabilities: PartiesService::Parties

getAssignedParties: PartyRoles[]

Scope Public

Description This operation returns the collection of PartyRoles, that have filled the Role designated by the roleId.

Parameters roleId: Id {in}

getAssignedParty: Party

Scope Public

Description This operation returns the Party that filled a particular Role as designated by the by the roleID on the effectiveDate.

Parameters roleId: Id {in}
effectiveDate: TimeStamp {in}

getParent: Party

Scope Public

Description This operation returns the Party which is the immediate Parent of the Party designate by the partyId.

Parameters partyId: Id {in}

getParty: Party

Scope Public

Description This operation returns the Party associated with the particular partyId.

Parameters partyID: Id {in}

getSubordinates: Party[]

Scope Public

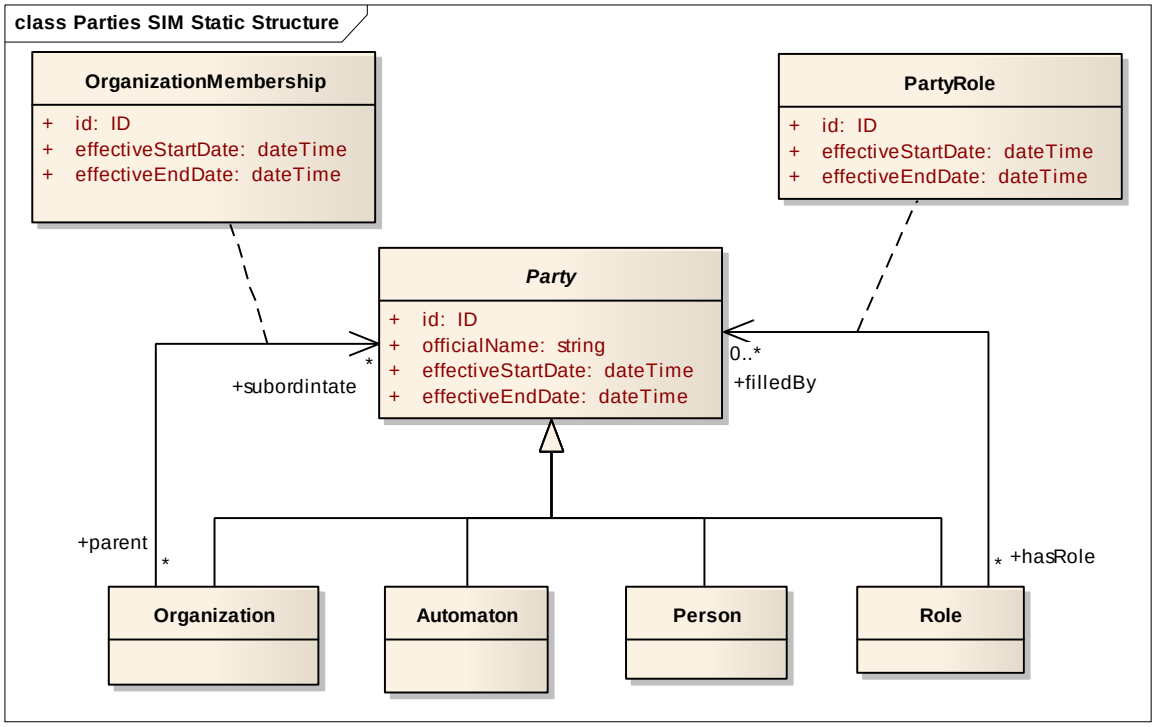
Description This operation returns the collection of Party's which are subordinate to the Party designated by the partyId on the effectiveDate.

Parameters partyId: Id {in}
effectiveDate: TimeStamp {in}

8.6.4.2.4 Package: PartiesSIM

The PartiesSIM package contains the elements used to define the parameters and information management responsibilities of the Parties service.

Parties SIM Static Structure



The Parties Service SIM Static Structure diagram shows the information elements that comprise parameters or information that the Parties Service service must manage.

See the Parties Package of the RMS Domain model for definitions of these elements.

8.6.4.2.4.1 Class: PartiesSIM::Automaton

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Generalization

From Class: PartiesSIM::Automaton

To Class: PartiesSIM::Party

8.6.4.2.4.2 AssociationClass: PartiesSIM::OrganizationMembership

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: OrganizationMembership.id

Type: ID

Description: See the Party package of the RmsDomainModel for a definition of this element.

Attribute: OrganizationMembership.effectiveStartDate

Type: dateTime

Description: See the Party package of the RmsDomainModel for a definition of this element.

Attribute: OrganizationMembership.effectiveEndDate

Type: dateTime

Description: See the Party package of the RmsDomainModel for a definition of this element.

Connections

Dependency

From Issue: PartiesSIM::Are these IDs auto-generated or do they need to be set?

To AssociationClass: PartiesSIM::OrganizationMembership

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To AssociationClass: PartiesSIM::OrganizationMembership

8.6.4.2.4.3 AssociationClass: PartiesSIM::PartyRole

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: PartyRole.id

Type: ID

Description: See the Party package of the RmsDomainModel for a definition of this element.

Attribute: PartyRole.effectiveStartDate

Type: dateTime

Description: See the Party package of the RmsDomainModel for a definition of this element.

Attribute: PartyRole.effectiveEndDate

Type: dateTime
Description: See the Party package of the RmsDomainModel for a definition of this element.

Connections

Dependency

From Issue: PartiesSIM::Are these IDs auto-generated or do they need to be set?

To AssociationClass: PartiesSIM::PartyRole

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To AssociationClass: PartiesSIM::PartyRole

8.6.4.2.4.4 Class: PartiesSIM::Organization

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Generalization

From Class: PartiesSIM::Organization

To Class: PartiesSIM::Party

AssociationClass

See the Party package of the RmsDomainModel for a definition of this association.

From Class: PartiesSIM::Organization
In the Role of: parent

Multiplicity: *

To Class: PartiesSIM::Party
In the Role of: subordintate
Multiplicity: *

8.6.4.2.4.5 Class: PartiesSIM::Party

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Attribute: Party.id
Type: ID
Description: See the Party package of the RmsDomainModel for a definition of this element.

Attribute: Party.officialName
Type: string
Description: See the Party package of the RmsDomainModel for a definition of this element.

Attribute: Party.effectiveStartDate
Type: dateTime
Description: See the Party package of the RmsDomainModel for a definition of this element.

Attribute: Party.effectiveEndDate
Type: dateTime
Description: See the Party package of the RmsDomainModel for a definition of this element.

Connections

Generalization

From Class: PartiesSIM::Organization

To Class: PartiesSIM::Party

Generalization

From Class: PartiesSIM::Person

To Class: PartiesSIM::Party

AssociationClass

See the Party package of the RmsDomainModel for a definition of this association.

From Class: PartiesSIM::Role
In the Role of: hasRole
Multiplicity: *

To Class: PartiesSIM::Party
In the Role of: filledBy
Multiplicity: 0..*

AssociationClass

See the Party package of the RmsDomainModel for a definition of this association.

From Class: PartiesSIM::Organization
In the Role of: parent
Multiplicity: *

To Class: PartiesSIM::Party
In the Role of: subordintate
Multiplicity: *

Generalization

From Class: PartiesSIM::Automaton

To Class: PartiesSIM::Party

Generalization

From Class: PartiesSIM::Role

To Class: PartiesSIM::Party

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To Class: PartiesSIM::Party

Dependency

The Annotations service maintains reference information related to parties associated with annotations.

From Class: AnnotationsService::Annotations

To Class: PartiesSIM::Party

Dependency

From Issue: PartiesSIM::Are these IDs auto-generated or do they need to be set?

To Class: PartiesSIM::Party

8.6.4.2.4.6 Class: PartiesSIM::Person

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To Class: PartiesSIM::Person

Generalization

From Class: PartiesSIM::Person

To Class: PartiesSIM::Party

8.6.4.2.4.7 Class: PartiesSIM::Role

See the Party package of the RmsDomainModel for a definition of this element.

Attributes

Connections

Dependency

The Parties service is responsible for managing the information regarding organizational structure used for the purpose of establishing provenance within the records management services.

From Class: PartiesService::Parties

To Class: PartiesSIM::Role

AssociationClass

See the Party package of the RmsDomainModel for a definition of this association.

From Class: PartiesSIM::Role
In the Role of: hasRole
Multiplicity: *

To Class: PartiesSIM::Party
In the Role of: filledBy
Multiplicity: 0..*

Generalization

From Class: PartiesSIM::Role

To Class: PartiesSIM::Party

Issue: PartiesSIM::Are these IDs auto-generated or do they need to be set?

Attributes

Connections

Dependency

From Issue: PartiesSIM::Are these IDs auto-generated or do they need to be set?

To AssociationClass: PartiesSIM::OrganizationMembership

Dependency

From Issue: PartiesSIM::Are these IDs auto-generated or do they need to be set?

To AssociationClass: PartiesSIM::PartyRole

Dependency

From Issue:	PartiesSIM::Are these IDs auto-generated or do they need to be set?
To Class:	PartiesSIM::Party

9 Platform Specific Models

Two sets of Platform Specific Models are defined for the specification

- RMS Information Exchange XSD's
- WSDL's for RMS Web Service Definitions

These machine-readable files are considered normative parts of this specification.

9.1 RMS Information Exchange XSD's

Records Management XML Schemas are based on the Document, Party, RmsCore, and AttributeProfile packages in the RmsPIM. They have two purposes:

1. They are used to import and export Managed Records via a compliant XML file.

2. They are used as the basis for forming XQuery/XPath query statements and responses to those statements.

9.1.1 Partitioning of XSD's

An XSD is provided for every domain package partitioned among the four containing packages:

- Rms-Core
  - Annotation.xsd
  - Attributeprofile.xsd
  - Authenticity.xsd
  - Category.xsd
  - Dispositions.xsd
  - Managedrecords.xsd
  - Role.xsd
- Document
  - Document.xsd
- AttributeProfile
  - AttributeProfile.xsd
- Party
  - Party.xsd

The correlation between the model packages and the XSD files is as follows:

Schema	**Model Pack1age**	**XSD File**
Annotation	RmsPim/RmsCore/Annotations	Annotations.xsd
AttributeProfile	RmsPim/AttributeProfile	AttributeProfile.xsd

Authenticity	RmsPim/RmsCore/Authenticity	Authenticity.xsd
Category	RmsPim/RmsCore/Category	Category. xsd
Dispositions	RmsPim/RmsCore/Dispositions	Dispositions. xsd
Document	RmsPim/Document	Document.xsd
ManagedRecords	RmsPim/RmsCore/ManagedRecords	ManagedRecords. xsd
Role	RmsPim/RmsCore/RmsRoles	Role.xsd
Party	RmsPim/Party	Party. xsd
RmsSet	N/A	RmsSet.xsd

- The namespaces for each domain package XSD are as follows:
  - http://www.omg.org/spec/rms/20101201/Party
  - http://www.omg.org/spec/rms/20101201/rms-core/Role
  - http://www.omg.org/spec/rms/20101201/Document
  - http://www.omg.org/spec/rms/20101201/rms-core/Annotation
  - http://www.omg.org/spec/rms/20101201/rms-core/Category
  - http://www.omg.org/spec/rms/20101201/rms-core/ManagedRecords
  - http://www.omg.org/spec/rms/20101201/rms-core/Dispositions
  - http://www.omg.org/spec/rms/20101201/rms-core/Authenticity
  - http://www.omg.org/spec/rms/20101201/AttributeProfile
- Each of the domain packages XSD's contains a complexType which is named by concatenating the domain package name with "Set". Its purpose is to serve when necessary as root element that is not semantically part of the domain package(s). Each is defined to have 0 to unbounded number of members of class type elements allowing the transfer of unrelated items should the business case require it.
- One exception to "one …Set" per Domain Package is the Dispositions Package which, due to its size, was partitioned within the XSD itself with five …Set complexType's
  - DispositionInstructionSet
  - DispositionPlanSet
  - DispositionRecordSets
  - DispositionEventSet
  - DispositionSuspendSet
- An additional XSD, RmsSet.xsd, is provided
  This XSD is a "collector" for the domain package XSD's. The primary purpose of this XSD is to provide a container (root element) for exchanging RMS information that is not a semantic element, but an "abstract" container.
  - Its namespace is targetNamespace=http://www.omg.org/spec/rms/20101201/Rms-Set

- It imports all of the above domain package namespaces.
- It has a single complexType named “RmsSet” which contains a sequence of optional elements
  - There is one element for each domain package.
  - The element is of the …Set complexType from each of the domain packages.

9.1.2 UML to XSD Transformation Heuristics

The XSD’s were derived from the UML model using the following heuristic guide:

9.1.2.1 UML Classes

UML Classes are converted to xs:complexType statements the name of which is the classes name. Additionally an element is created with the same name (the class’s name) with the type of the class’s complexType, enabling any class to serve as a root element if it makes sense for the purpose of the information exchange.

Each Class contains an xs:sequence of xs:element’s for

- attributes with clauses for
  - name=”attribute-name”
  - type=”xs:attribute-type”
  - minOccurs and maxOccurs statements to reflect multiplicity
- associations
  - name=”roleName”+”ID”
    - *[The convention is to append the class name to which the association extends with ID forming, in effect, a “foreign key” to the class instance]*
  - type=xs:NCName
    - *[type=NCName is used in lieu of IDREF since under many business scenarios the target instance may not be present in the same XML document as the instance being transferred.]*
  - minOccurs and maxOccurs statements to reflect multiplicity on the terminating end of the association.

9.1.2.2 UML Association

The opposite end role of each UML Association becomes an element contained within a class (xs:complexType) participating in the association.

These elements are “peers” to the attributes (xs:elements) in the classes (xs:complexTypes) and are foreign keys pointing to the class instance(s) at the other end of the association, as explained in “Classes” above.

9.1.2.3 Association Classes

UML Association Classes are converted to Classes as defined above. UML Associations (defined above) are created between the Association Class and each of the classes participating in the association. Note there are no direct pointers between the associated classes.

9.1.2.4 Enumerations

Enumerations are converted to simpleType's with name="NameOfEnumerationClass". A base for the type is declared via an xs:restriction that contains the xs:enumeration values.

9.2 WSDL's for RMS Web Service Definitions

Ten WSDL files are provided; one for each of the following services.

- Annotations
- AttributeProfiles
- Authorities
- Categories
- Dispositions
- Documents
- ManagedRecords
- Query
- RecordAuthentications
- Parties

The WSDL files were created based on the interface operations in the corresponding service packages in the RmsServices package of the RmsPim. The correlation is as follows:

Service: Annotations
Package: RmsPim/RmsServices/RmsCoreServices/AnnotationsService
Annotations.wsdl

Service: Authorities
RmsPim/RmsServices/RmsCoreServices/ AuthoritiesService
Authorities.wsdl

Service: Categories
RmsPim/RmsServices/RmsCoreServices/ CategoriesServic
Categories.wsdl

Service: Dispositions
RmsPim/RmsServices/RmsCoreServices/ DispositionsService
Dispositions.wsdl

Service: Documents

RmsPim/RmsServices/RmsCoreServices/ DocumentsService
Documents.wsdl

Service: ManagedRecords
RmsPim/RmsServices/RmsCoreServices/ ManagedRecordsService
ManagedRecords.wsdl

Service: Query
RmsPim/RmsServices/RmsCoreServices/ Query Service
Query.wsdl

Service: RecordAuthentications
RmsPim/RmsServices/RmsCoreServices/ RecordAuthenticationsService
RecordAuthentications.wsdl

Service: AttributeProfiles
RmsPim/RmsServices/RmsUtilityServices/AttributeProfiles Service
AttributeProfiles.wsdl

Service: Parties
RmsPim/RmsServices/RmsUtilityServices/PartiesService
Parties.wsdl

Appendix A – Model Package Descriptions

A.1 Package: RMS Submission

A.1.1 Package: RmsPim

The RmsPim package contains the general Platform Independent Model of the RMS Specification. This model is used to capture a structural and behavioral specification in a manner that can be implemented in a variety of technologies.

A.1.1.1 Package: Overview

The Records Management Services Packages consist of a Platform Independent Model (PIM) and a Platform Specific Model (PSM).

The Platform Specific Model contains four packages describing the Domain Model, and one describing the Service Model. Each of these major packages are comprised of further subpackages. (See Appendix for the package structure.)

The Domain Model Packages are:

1. The RmsCore class structure of Records Management based on the work of the Records Management Services Component Interagency Project Team.

2. The AttributeProfile package provides the capability of specifying attribution by class type for the major RECORDS MANAGEMENT classes. This enables attribution based on the business context of the RECORDS MANAGEMENT environment allowing attribution according to such standards as DoD 5015.2, Dublin Core, etc.

3. The Document package collects the elements needed to support records that are one or more electronic "bit streams". Each bit stream is represented by a Document.

4. The Party package collects the elements necessary for modeling organizational structure. In the case of records management, however, the purpose is to capture assignment of responsibility of actions and custodianship, not represent the overall organizational structure.

The Service Model

A.1.1.2 Package: AttributeProfile

The AttributeProfile package provides the capability of specifying attribution by class type for the major RECORDS MANAGEMENT classes. This enables attribution based on the business context of the RECORDS MANAGEMENT environment allowing attribution according to such standards as DoD 5015.2, Dublin Core, etc.

A.1.1.3 Package: Document

The Document package collects the elements needed to support records that are one or more electronic "bit streams". Each bit stream is represented by a Document.

A.1.1.4 Package: Party

The Party package collects the elements necessary for modeling organizational structure. In the case of records management, however, the purpose is to capture assignment of responsibility of actions and custodianship, not represent the overall organizational structure.

A.1.1.5 Package: RmsCore

The core class structure of Records Management based on the work of the Records Management Services Component Interagency Project Team.

A.1.1.5.1 Package: Annotation

The Annotation package collects the elements needed to support the records management concept of annotated records.

A.1.1.5.2 Package: Authenticity

The Authenticity package collects the elements needed to support the records management concept of authentic records, i.e., providing assurance that what is retrieved from a record management environment is identical to that which was put there.

A.1.1.5.3 Package: Category

The CaseFile package collects the elements needed to support the records management concept of record categories.

A.1.1.5.4 Package: Dispositions

The Document package collects the elements needed to support records that are one or more electronic "bit streams". Each bit stream is represented by a Document.

A.1.1.5.5 Package: ManagedRecord

The ManagedRecord package collects the elements needed to support the basic concepts of a ManagedRecord. It is shown here in its full context of many of the key concepts associated with managing a record.

A.1.1.5.6 Package: RmsRoles

The RmsParty package extends the Party package to include the roles necessary for assigning responsibility of actions and custodianship in a records management environment. It is not intended to represent the overall organization structure.

A party model in a records management context is related to the organizational structure of the organization in which records are being managed, but is not identical to it. The

purpose of the model is to be able to express Provenance and to identify the Roles in the organization that attribute aspects of the Managed Record.

A.1.1.6 Package: RmsServices

The RmsServices package contains subpackages for each service provided by RMS. RmsSolution maps the capabilities available to RMS clients. RmsProcessServices is a placeholder for future process oriented services (corresponds to the empty "process services layer" of the RMS Service Capability Layering diagram that opens the RmsServices section. The RmsCoreServices

A.1.1.6.1 Package: RmsSolution

The RmsSolution package contains elements that represent clients of the RMS services. These are generally referred to as RMS Clients and RMS Applications.

A.1.1.6.2 Package: RmsProcessServices

The RMS Process Services package is an architectural placeholder for process-related services. The current version of the specification is meant to be independent of business processes that generate records. Future versions of the specification may include process services that manage records management functions.

A.1.1.6.3 Package: RmsCoreServices

The RMS Core Services package contains the specifications of the RECORDS MANAGEMENT Services in the Core Business Layer of the RMS architecture.

A.1.1.6.3.1 Package: AnnotationsService

The AnnotationsService package contains the model elements that together define the Annotations service.

Package: AnnotationsSIM

The AnnotationsSIM package contains the elements used to define the parameters and information management responsibilities of the Annotations service.

A.1.1.6.3.2 Package: AuthoritiesService

The AuthoritiesService package contains the model elements that together define the Authorities service.

Package: Authorities SIM

The Authoress package contains the elements used to define the parameters and information management responsibilities of the Authorities service.

A.1.1.6.3.3 Package: CategoriesService

The CategoriesService package contains the model elements that together define the Categories service.

Package: CategoriesSIM

The CategoriesSIM package contains the elements used to define the parameters and information management responsibilities of the Categories service.

A.1.1.6.3.4 Package: DispositionsService

The DispositionsService package contains the model elements that together define the Dispositions service.

Package: DispositionsSIM

The DispositionsSIM package contains the elements used to define the parameters and information management responsibilities of the Dispositions service.

A.1.1.6.3.5 Package: DocumentsService

The DocumentsService package contains the model elements that together define the Documents service.

Package: DocumentsSIM

The DocumentsSIM package contains the elements used to define the parameters and information management responsibilities of the Documents service.

A.1.1.6.3.6 Package: ManagedRecordsService

The ManagedRecordsService package contains the model elements that together define the ManagedRecords service.

Package: ManagedRecordsSIM

The ManagedRecordsSIM package contains the elements used to define the parameters and information management responsibilities of the ManagedRecords service.

A.1.1.6.3.7 Package: QueryService

Provides the capability of returning RECORDS MANAGEMENT entities base on their RmsDomainModel as specified in the query string. The returning parameter is also based upon the RmsDomainModel.

Package: QuerySIM

Refer to the RmsDomainModel for the elements used to define the parameters and information management responsibilities of the Queries service. See 'RmsDomainModel Package Structure'.

A.1.1.6.3.8 Package: RecordAuthenticationsService

The RecordAuthenticationsService package contains the model elements that together define the RecordAuthentications service.

Package: AuthenticationsSIM

The AuthenticationsSIM package contains the elements used to define the parameters and information management responsibilities of the Authentications service.

A.1.1.6.4 Package: RmsUtilityServices

The RmsUtilityServices package contains services that are used in many of the other packages.

A.1.1.6.4.1 Package: AttributeProfiles Service

The AttributeProfilesService package contains the model elements that together define the AttributeProfiles service.

Package: AttributeProfileSIM

The AttributeProfileSIM package contains the elements used to define the parameters and information management responsibilities of the AttributeProfiles service.

A.1.1.6.4.2 Package: PartiesService

The PartiesService package contains the model elements that together define the Parties service.

Package: PartiesSIM

The PartiesSIM package contains the elements used to define the parameters and information management responsibilities of the Parties service.

A.1.2 Package: RmsPsm

The sub-packages contained in the UML model are non-Normative.

In the case of XSD’s hese packages are interim results from an auto-generation transform. This transform makes one step toward achieving the desired XSD’s for the RMS. The rest of the operations were performed manual with an XSD editing tool. The heuristic used for transforming UML Domain Model Elements in the PIM to corresponding XSD’s is documented in the main body of the specification in the section

“Platform Specific Model”, which also documents the domain packages on which each XSD is based.

Similarly there are sub-packages for generation of the WSDL and traceability to the service packages on which each is based.

Appendix B – Use Case Scenarios

Use case scenarios were prepared by records management subject matter experts on the submission team. The following is a list of the scenarios kept, followed by the scenarios themselves. Some of the scenarios were deemed redundant or out of scope, hence the non sequential scenario numbers. The scenarios were used to test the information and service models to assure completeness.

Scenario 01 – Capture MS Word Document as a Record

John Doe is an action officer in the ABC agency. He is specifically responsible for conducting government acceptance tests for new software products. At the conclusion of the test, he prepares a written report using MS Word as his document processor. The report has been approved for release and publication. He now wants to manage the report as a record.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Organization has a records schedule.
2. Organization has a retention schedule for the records schedule.
3. Organization provides the user with the capability to search the records schedule for a record category.

B. Script of Operations

Scenario 1 – Capture Word Document as a Record			
Step	**User Action**	**System Response**	**Comments**
1.	Action Officer creates and saves a report to his "c" drive	Document is written to the user's C drive	
2.	Action Officer retrieves and opens the document to declare it as a record.	Systems displays document on the user's desktop	
3.	Action Officer reviews the document and clicks the "Save as Record" option	System displays the prompt "Save as Record Yes/No?	
4.	User clicks the "Yes" option	System displays the following options: 1. Choose from a list of the user's favorite (pre-defined) categories 2. Choose from a list of recently used categories by the user 3. Search for	1. It is conceived that a user will have a pre-determined sub-set of the agencies records schedule, commonly referred to as a file plan

Scenario 1 – Capture Word Document as a Record			
Step	**User Action**	**System Response**	**Comments**
		category/folder	2. As is often scene and made as a function of an application most recently used, last accessed are kept and presented to the user as a function.
5.	User clicks the "Search for Category" option	System displays the category search interface	The interface would have the capability to search by word(s) and/or phrase(s)
6.	User enters search criteria and clicks OK	System displays a list of categories that matched the user's search criteria.	
7.	User selects the most appropriate category	System enters the category value in the category attribute field.	
8.	User may review the auto- populated attribute fields for completeness	1. Record ID (system assigned) 2. Date/Time Set Aside (system assigned) 3. Set Aside by Action Officer (system assigned) 4. Agency Name, Current (system assigned) 5. Agency Name, Current Date (system assigned system date) 6. Record Keeper Name, Current (system assigned) 7. Record Keeper Name, Current Date (system	

Scenario 1 – Capture Word Document as a Record			
Step	**User Action**	**System Response**	**Comments**
		assigned) This must be the same as Agency Name Current Date (above) 8. Agency Name (sub-ordination structure if required by business rules; system assigned) 9. Agency Name Date (sub-ordination structure date system assigned by) 10. Record Category, Current (selected by user in Step 8) 11. Record Category, Current Date (system assigned) 12. Disposition Instruction, Current (from record category : disposition instruction relationship) 13. Disposition Instruction, Current Date (system assigned)	
16.	User saves his/her work	Profile is closed and document is now a managed record.	

C. Post-Conditions

1. A unique identifier is assigned to the record
2. The record's profile is associated with the record.
3. The record is now managed as a managed record

Scenario 02 – Capture PDF Document as a Record

John Doe is an action officer in the ABC agency. He is specifically responsible for conducting government acceptance tests for new software products. At the conclusion of the test, he prepares a written report using MS Word as his document processor. The report has been approved for release and publication. He converts the report to PDF for external distribution. He now wants to manage the PDF version of the report as a record.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Organization has records schedule.
2. Organization has a retention schedule. Dispositions have been assigned to record categories.
3. Organization provides the user with the capability to search records categories.

B. Script of Operations

Scenario 2 – Capture PDF Document as a Record			
Step	**User Action**	**System Response**	**Comments**
1.	Action Officer creates an after action report and wants to save a PDF copy of the document as a record. The AO selects save from a menu list	The System prompts "Save as Document" "Save as Record"	
2.	AO selects "Save as Record"	System prompts "Save as Record Yes/No?"	
3.	AO selects "No"	System closes human interface for saving the PDF no further action by system	
4.	AO selects "Yes"	System provides user with an interface that allows the user to type in a record category, search for a record category with key word and/or phrase and cancel action.	`
5.	AO types in "A1400"	System responds "Place into	

Scenario 2 – Capture PDF Document as a Record			
Step	**User Action**	**System Response**	**Comments**
		A1400 After Actions Reports Yes/No”	
6.	AO selects “No”	System returns AO to #4 interface above	
7.	AO selects “Yes”	PDF document is set aside as a record in “A1400 After Action Reports”	
8.	AO types in search for a record category area “after action reports”	System responds “A1400 After Action Reports” and “A5000 After Action Reports Engineers”	
9.	AO selects “A1400 After Action Reports”	System responds “Place into A1400 After Actions Reports Yes/No”	
10.	AO selects “No”	System returns AO to the user interface in #4 above.	
11.	AO selects “Yes”	PDF document is set aside as a record in “A1400 After Action Reports”	
12.	AO selects “cancel”	System responds “Cancel Action Yes/No”	
13.	AO selects “Yes”	System closes human interface for saving the PDF no further action by system.	
14.	AO selects “No”	System returns AO to #4 above.	

C. Post-Conditions

4. A unique identifier is assigned to the PDF record
5. The PDF record is now managed as a managed record
6. The PDF record’s profile is associated with the PDF record.

Scenario 10a – Establish Record Authenticity

The ABC agency has a record authenticity service. Each time a document is captured as a record the service will automatically authenticate the record (hash or digital signature are example services). Or the records staff could manually select a record or set of records to authenticate on an ad hoc basis..

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Organization has a service or utility that will establish the hash value (has value is one way of many ways to accomplish this activity and is used as an example only and is not to be considered directive in nature) of a managed record.
2. The organization has the capability to run the authentication service as a scheduled service.
3. Members of the records staff have the capability to manually select records to be authenticated.

B. Script of Operations

Scenario 1 – Capture Word Document as a Record			
Step	**User Security**	**System Response**	**Comments**
1.	User sets aside a document as a record.	System recognizes a document has been set aside as a record and initiates the authentication service. The system runs a hash routine against the record and populates the authenticity base and authenticity base date attributes.	

C. Post-Conditions:

Scenario 10b – Validate Record Authenticity

The ABC agency has a record authenticity service. At intervals pre-determined the agency may have the validate authenticity service validate records, however, this scenario deals with a user requesting a copy of a record to work with.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Organization has a service or utility that has established the hash value (has value is one way of many ways to accomplish this activity and is used as an example only and is not to be considered directive in nature) of a managed record.
2. The organization has the capability to run the authentication service.
3. Members of the records staff have the capability to manually select records to be authenticated.

B. Script of Operations

Scenario 1 – Validate Record Authenticity			
Step	**User Security**	**System Response**	**Comments**
1.	User searches, selects a record for use.	1. The system initiates the validate authenticity service on the record. 2. The system selects the authenticity type used to set the authenticity base attribute. 3. The system runs the authenticity type used to set the authenticity base attribute and populates the Authenticity Current attribute and the Authenticity Current Date attribute. 4. The system compares the Authenticity Base (AB) attribute with the Authenticity Current (AC) attribute. 5. If AB=AC then populate the Authenticity	This is the first request for the record sent it was set aside. Ensuring the correct authenticity type is being used is up to the solution provider, it is simple to think a record of what authenticity type used at the time the authenticity base date was set by the system would be the simplest, but not the only way to accomplish this.

Scenario 1 – Validate Record Authenticity			
Step	**User Security**	**System Response**	**Comments**
		Validation attribute with a 1. The system identifies the Authenticity Validation attribute is populated with a "1" and makes a copy of the record available to the user.	
2.	User receives copy of the record		
3.		If AB does not equal AC populates the Authenticity Validation attribute with a "0." The system makes identifies the Authenticity Validation attribute is set with "0" and creates a notification the validation of the records authenticity has failed. System notifies user the record cannot be authenticated at this time and a message has been sent (to: help desk, RM, security?) therefore a copy cannot be provided at this time.	This notification would normally go to at least the records manager and system security, maybe the administrator, but final decision on the notification will be in accordance with the agency rules.

C. Post-Conditions:

1. A copy of the record has been provided to the requester
2. A copy of the record was not provided to the requester and notifications of the same were created and sent to the appropriate parties.

Scenario 11 – Link Associate Records

John Henry is an action officer in the ABC agency. He has filed a new record that supersedes a previously filed record. His task is to link the superseded record with the superseded by record.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has a Link function associate one or more records with another.
2. Agency has a Search and Retrieve records function that permits the user to search for and retrieve records that he has access to.
3. Agency has predefined link types such as Supporting/Supported; Superseded/Superseded By; Cross Reference; Email/Attachment, etc.
4. Agency has two or more declared records.

B. Script of Operations

Scenario 11 –Link/Associate Records			
	Actor Action	**System Response**	**Comments**
1.	User launches the "Link" function	Systems displays the "Link" function user interface with the following data windows: • Link From button • Link To button • Link Type button	
2.	User selects *Link From* button	Systems launches the Search for Records function	
3.	User enters search criteria as appropriate to retrieve the desired record	Systems displays the desired record in a search results display	
4.	User selects the desired record in the search results display	System highlights the specified record	
5.	User selects the *Link To* button	Systems launches the Search for Records function	

Scenario 11 –Link/Associate Records			
	Actor Action	**System Response**	**Comments**
6.	User enters search criteria as appropriate to retrieve the desired record	Systems displays the desired record in a search results display	
7.	User selects the desired record in the search results display	System highlights the specified record	
8.	User selects the *Link Type* button	System displays a pre-defined list of Link Types (*Record_Association_Id* and *Records_Assoication_Description* such as: • Cross-Reference • Supported, Supporting • Superseded, Superseded By	
9.	User selects the appropriate Link Type	System associates the two records with the specified link type and populates (on each record) the • *Record_ Association_ Id* attribute • *Record_ Association* description attribute • *Record_Association_Date*	

C. Post-Conditions

1. The two records are linked with the appropriate link type..

Scenario 12 – Change Record Provenance

Jane Doe is a records officer in the ABC agency. The agency has three divisions and three record keepers (one per division). Her agency has been subsumed by agency XYX. In addition, what was the ABC agency will move form three divisions to two divisions.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic hierarchical category structure in place (Otherwise known as a records schedule)
2. Agency has a retention schedule associated with the records schedule in place. The disposition rules are assigned to the categories.
3. Agency provides the user with the capability to search the category structure when selecting which category and folder to file his/her record(s).
4. The attributes relevant to the agency's provenance are populated.
5. User has the capability to retrieve a group of records by category or subject.

B. Script of Operations

Scenario 12 –Change Record Provenance			
Step	**User Action**	**System Response**	**Comments**
1.	Jane Doe has received noticed that her agency has been consumed by the XYZ agency, effective today and will be going from three divisions to two divisions. Jane Doe must update the provenance of all of the agencies records and the recordkeeper of the records for each of the divisions	Waiting on user input	
2.	Jane Doe requests access to the agencies records and access to the provenance service	System receives request and if authentication of user is valid then make the provenance update service available	
3.	Jane Doe indicates to the service to update the	System prompts the user with "Are You Sure? Yes	

Scenario 12 –Change Record Provenance			
Step	**User Action**	**System Response**	**Comments**
	provenance attribute to XYZ agency for all of the agencies records.	or No".	
4.	4.a1 Jane Doe selects "Yes"	System initiates provenance service and when action complete, display "Complete" and prompts "OK"	
	4.a2 Jane Doe selects "OK"	System closes the provenance update service	
	4.b.1 Jane Doe Selects "No"	System closes the provenance update service	
5.	Jane Doe has to replace the division 2 recordkeeper, Billy Jean, with the division 1 recordkeeper, Bobby Riggs, because division 1 and division 2 were aggregated together, as division 1.		
6.	Jane Doe requests access to the provenance service to update the RecordKeeper for all records Bill Jean is the RecordKeeper.	System receives request and if authentication of user is valid then make the provenance service available	
7.	Jane Doe makes the request to update the recordkeeper attribute for the records of division 2 to indicate that Bobby Riggs as the recordkeeper	System receives request and if authentication of user is valid for division2, then system prompts the user with "Are You Sure? Yes or No".	
	7.a1 Jane Doe selects "Yes"	System initiates provenance service and when action complete, display "Complete" and	

Scenario 12 –Change Record Provenance			
Step	**User Action**	**System Response**	**Comments**
		prompts “OK” and “More Changes?”	
	7.a2 Jane Doe selects “OK”	System closes the provenance service	
	7.a3 Jane Doe selects more changes	System repeats steps 6 and 7 until user selects “OK” or “No”	
	7.b.1 Jane Doe Selects “No”	System closes the provenance update service	

Scenario 13 – Change Record Keeper

Jane Doe is a records officer in the ABC agency. She is specifically responsible for the management and maintenance of the agency's records. The "record keeper" is defined as the administrative entity, unit, office, or person responsible for the custody and ongoing management of the records during their active business use.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic hierarchical category structure in place. File plan consists of subject categories and folders created to aggregate and store digital records.
2. Agency has a retention schedule associated with the category structure in place. The disposition rules are assigned to the subject categories and are inherited by the folders and records.
3. Applicable subject categories and folders have been marked Vital. Vital marking is inherited by folders and records.
4. User profile information includes name, office, agency, etc.
5. User has compiled a list of "favorite" folders that he/she is most likely to file his/her record(s).
6. Agency has the capability to track and display the folders most recently used by the user to file his/her record(s).
7. Agency provides the user with the capability to search the category structure when selecting which category and folder to file his/her record(s).
8. The attributes relevant to the agency's electronic record holdings provenance were populated at the time of record capture.
9. User has the capability to retrieve a group of records by category or subject.

B. Script of Operations

Scenario 12 –Change Record Keeper			
Step	**User Action**	**System Response**	**Comments**
1.	Records Officer searches for and retrieves records where the record keeper is assigned to the previous record keeper's name.	• System displays the list of records to a search results screen with the previous record keeper's name populated in the *Record Keeper, Current* attribute field	
2.	Record Officer selects all of the records listed in the search result screen	Systems highlights all records in the search results screen	
3.	Record Officer launches a global update utility.	System displays the utility window. Providing a drop down list of all fields that can be globally updated.	
4.	Records Officer User selects the *Record Keeper, Current* data field	Systems displays a data window in which the new value can be entered	
5.	Records Officer enters a new value for the *Record Keeper, Current*	System displays the new value in the *Record Keeper, Current* attributes	
6.	Records Officer saves his/her work	System updates the following attributes on each selected record: • *Record Keeper, Current* • *Record Keeper*	

Scenario 12 –Change Record Keeper			
Step	**User Action**	**System Response**	**Comments**
		Current Date • *Record Keeper, Previous* • *Record Keeper Previous Date*	
7.		The system updates the Provenance on all selected records.	

Scenario 14 - Change Modifiable Attribute

Jane Doe is a records officer in the ABC agency. She is specifically responsible for monitoring the accuracy of metadata applied to records by the agency's action officers. The agency uses a country attribute to identify the country that is the source of the record's content. The country code for the Soviet Union (USSR) has been changed RUSS for Russia.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic hierarchical category structure in place. File plan consists of subject categories and folders created to aggregate and store digital records.
2. Agency has a retention schedule associated with the category structure in place. The disposition rules are assigned to the subject categories and are inherited by the folders and records.
3. Applicable subject categories and folders have been marked Vital. Vital marking is inherited by folders and records.
4. User profile information includes name, office, agency, etc.
5. User has compiled a list of "favorite" folders that he/she is most likely to file his/her record(s).
6. Agency has the capability to track and display the folders most recently used by the user to file his/her record(s).
7. Agency provides the user with the capability to search the category structure when selecting which category and folder to file his/her record(s).
8. The attributes relevant to the agency's electronic record holdings provenance were populated at the time of record capture.
9. User has the capability to retrieve a group of records by category or subject.

B. Script of Operations

Scenario 14 –Change Modifiable Attribute			
Step	**User Action**	**System Response**	**Comments**
1.	Records Officer searches for and retrieves a single or a collection of electronic records. The search is by category, or by "subject".	System displays the list of records to a search results screen	
2.	Record Officer launches a global update utility.	System displays the utility window.	

Scenario 14 –Change Modifiable Attribute			
Step	**User Action**	**System Response**	**Comments**
		Providing a drop down list of all fields that can be globally or singularly updated.	
3.	Records Officer User selects the *Agency_Official_Name_Current* data field	Systems displays a data window in which the new value can be entered	
4.	Records Officer enters a new value for the *Agency_Official_Name_Current*	System updates the following attributes on each selected record: • *Agency Official Name, Current* • *Agency Official Name Current Date* • *Agency Official Name, Previous* • *Agency Official Name Previous Date*	
5.	Records Officer selects the *Record Keeper, Current* data field	Systems displays a data window in which the new value can be entered	
6.	Records Officer enters a new value for the *Record Keeper, Current*	System updates the following attributes on each selected record: • *Record Keeper, Current* • *Record Keeper Current Date*	

Scenario 14 –Change Modifiable Attribute			
Step	**User Action**	**System Response**	**Comments**
		• *Record Keeper, Previous* • *Record Keeper Previous Date*	
7.	Records Officer clicks the appropriate button to save her work.	The system updates the Provenance on all selected records.	

Scenario 15 - Change Authority Attribute

Jane Doe is a records officer in the ABC agency. She is specifically responsible for monitoring the accuracy of metadata applied to records by the agency's action officers. The agency maintains a list of source authorities for the retention schedule. In this use case, the authority for the retention assigned to a particular record is changed from GRS 20 a2 to GRS 20 a4.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic hierarchical category structure in place. File plan consists of subject categories and folders created to aggregate and store digital records.
2. Agency has a retention schedule associated with the category structure in place. The disposition rules are assigned to the subject categories and are inherited by the folders and records.
3. Applicable subject categories and folders have been marked Vital. Vital marking is inherited by folders and records.
4. User profile information includes name, office, agency, etc.
5. User has compiled a list of "favorite" folders that he/she is most likely to file his/her record(s).
6. Agency has the capability to track and display the folders most recently used by the user to file his/her record(s).
7. Agency provides the user with the capability to search the category structure when selecting which category and folder to file his/her record(s).
8. The attributes relevant to the agency's electronic record holdings provenance were populated at the time of record capture.
9. User has the capability to retrieve a group of records by category or subject.
10. An Authority data attribute exists in the system with a predefined pick list of values.

B. Script of Operations

Scenario 15 –Change Authority Attribute			
Step	**User Action**	**System Response**	**Comments**
1.	User logs in as an records officer with system permissions to add new and change *Authorities* assigned to records.	System recognizes user's privileges and grants access to this function	

Scenario 15 –Change Authority Attribute			
Step	**User Action**	**System Response**	**Comments**
2.	If the Authority value does not exist in the system, the Records Officer launches the *Add New Authority* function.	System displays	
3.			
4.	Record Officer launches a global update utility.	System displays the utility window providing a drop down list of all fields that can be globally updated.	
5.	Records Officer User selects the *Authority, Current* data field	Systems displays a data window in which the new value can be entered	
	Records Officer enters a new value for the *Authority, Current*	Systems displays a data window with the new value	
	Records Officer saves her work	System updates the following attributes on each selected record: • *Authority, Current* • *Authority, Current Date* • *Authority, Previous* • *Authority, Previous Date*	
		The system updates the *Authority* attribute on all effected records.	

Scenario 18 – Validate Record Authenticity

Jane Doe is a records officer in the ABC agency. She is specifically responsible for the management and maintenance of the agency's records. The "record keeper" is defined as the administrative entity, unit, office, or person responsible for the custody and ongoing management of the records during their active business use.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic recordkeeping system in place. .
2. Agency has a document authentication service in place in conjunction with the electronic recordkeeping system. Applicable subject categories and folders have been marked Vital. Vital marking is inherited by folders and records.
3. Agency has one or more authenticated records.

B. Script of Operations

Scenario 18 –Validate Record Authenticity			
Step	**User Action**	**System Response**	**Comments**
1.	Records Officer searches for and retrieves a record that was previously authenticated.	1. System locates the authenticated record 2. System authentication service establishes a new authentication result for the retrieved record 3. System compares the new result with the previous result. 4. If the results match, then the system will display or list the record in the user's search result screen. 5. If the results do not match, then the system shall	

Scenario 18 –Validate Record Authenticity			
Step	**User Action**	**System Response**	**Comments**
		display a warning message to the user stating that the record is no longer authenticate. 6. The message will prompt the user to cancel his/her request to retrieve the record or to continue the request to retrieve the record.	

Scenario 19 – Suspend (Freeze) Disposition

Jane Doe is a records officer in the ABC agency. She is specifically responsible for applying a "hold" to freeze or exempt the disposition of records that match specified search criteria as relevant to an audit, investigation, or litigation.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic recordkeeping system with the capability to search and retrieve records.
2. Agency has a records hold function that will mark records needed for a specific hold and suspend (freeze) their disposition.
3. Agency has a list of names/positions that are authorized to initiate a hold.
4. Agency has a records officer role with the permissions necessary to search, retrieve, and mark records to suspend their disposition
5. The record officers received a notification from an authorized individual to place a collection of records that matched specified criteria on hold. The notification included a description/purpose of the hold and the search criteria.
6. Agency has one or more managed records.

Note: Suspend Disposition Authority is a legally binding order, notice, or freeze on the execution of the established disposition instruction of an established disposition authority. This could be a person or office.

B. Script of Operations

Scenario 19 –Suspend (Freeze) Disposition			
Step	**User Action**	**System Response**	**Comments**
	Create Suspend Disposition Authority		
1.	Upon receipt of Suspend Disposition order, the Records Officer launches the "Suspend Disposition" or "Hold" function	System displays the create "Hold" data window with the following data fields: • *Disposition_Authority_Suspend_Number* • *Disposition_Authority_Suspend_Title* • *Disposition_Authority_Suspend_Descript*	Added user defined attributes to disposition_suspend: Number Title Ordered by

Scenario 19 –Suspend (Freeze) Disposition			
Step	**User Action**	**System Response**	**Comments**
		ion • *Disposition_Authority_Suspend_Ordered By* • *Disposition_Authority_Suspend_Search Criteria*	Search criteria
2.	Records Officer tabs to the *Disposition_Authority_Suspend_Number* data field and enters the number of the Hold following the agency's numbering schema.	System displays the user entered value in the *Disposition_Authority_Suspend_Number* data field.	
3.	Records Officer tabs to the *Disposition_Authority_Suspend_Title* data field and enters the name of the Hold.	System displays the user entered value in the *Disposition_Authority_Suspend_Title* data field.	
4.	Records Officer tabs to the *Disposition_Authority_Suspend_Description* data field and enters a descriptive information about the Hold.	System displays the user entered value in the *Disposition_Authority_Suspend_Description* data field.	
5.	Records Officer tabs to the *Disposition_Authority_Suspend_Ordered_By* data field and enters the name and title of the person ordering the Hold.	System displays the user entered value in the *Disposition_Authority_Suspend_Ordered_By* data field.	Associated with a role/entity authorized to order a suspension (prevents transfer and/or destruction and changes to editable attributes). This might involve editing of

Scenario 19 –Suspend (Freeze) Disposition			
Step	**User Action**	**System Response**	**Comments**
			the organization's party model
6.	Records Officer tabs to the *Disposition_Authority_Suspend_Search_Criteria* data field and enters the criteria used to search for applicable records	System displays the user entered value in the *Disposition_Authority_Suspend_Search_Criteria* data field.	
7.	Records Officer saves her work	System assigns and appends the following data fields to the Hold: • *Disposition_Authority_Suspend_Identifier* • *Disposition_Authority_Suspend_Created By* • *Disposition_Suspend_Date_Created*	Created by (operator) and date created are audit attributes. Date hold was created and date that records were actually frozen could be different dates.
	Assign Hold to applicable managed records		
8.	Records Officer searches for and retrieves a single or a collection of records that meet search criteria.	System displays the list of records to a search results screen	
9.	Record Officer launches the Suspend Disposition utility.	System displays a pick list of existing Holds with the following data: • *Disposition_Authority_Suspend_Number* • *Disposition_Authority_Suspend_Title*	
10.	Records Officer selects	System displays a	

Scenario 19 –Suspend (Freeze) Disposition			
Step	**User Action**	**System Response**	**Comments**
	the appropriate Hold by *Disposition_Authority_Suspend_Number*	confirmation dialog box: "Do you want to apply this hold to the selected records?" Yes/No	
11.	Records Officer selects Yes to confirm	• System links each the Hold to all selected records metadata: • System populates the following metadata attribute on the Hold: *Disposition_Authority_Suspend_Is_Active*	The date applied in this step should be the system date when the records were put on Hold.
12..	If first instance has occurred, then	System creates a link for each additional Hold and appends to record	

C. Post-Conditions

1. A Disposition Suspension is added to the database
2. The id of the Disposition Suspension is linked to each selected record
3. The *Disposition_Authority_Suspend* attribute is populated on each selected record and this action freezes or exempts the records from destruction.

Scenario 20 – Re-Instate (Unfreeze) Disposition

Jane Doe is a records officer in the ABC agency. She is specifically responsible for applying a "hold" to freeze or exempt the disposition of records that match specified search criteria as relevant to an audit, investigation, or litigation.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic recordkeeping system with the capability to search and retrieve records.
2. Agency has a records hold function that will mark records needed for a specific hold and suspend (freeze) their disposition.
3. Agency has a list of names/positions that are authorized to initiate and to lift a hold.
4. Agency has a records officer role with the permissions necessary to search, retrieve, and mark records to lift suspended dispositions.
5. The record officers received a notification from an authorized individual to lift a disposition suspension. Agency has one or more managed records

B. Script of Operations

Scenario 20 – Re-Instate (Unfreeze) Disposition			
Step	**User Action**	**System Response**	**Comments**
1.	Record Officer launches the Re-Instate Disposition utility.	System displays a pick list of existing Holds with the following data: • Disposition_Authority_Suspend_Number • Disposition_Authority_Suspend_Title	
2.	Records Officer selects the appropriate Hold(s) by Disposition_Authority_Suspend_ Number	System displays the Disposition_Authority_Suspend_Number(s) for each desired hold	
3.	Records Officer selects Release Hold	System displays the following menu options:	

Scenario 20 – Re-Instate (Unfreeze) Disposition			
Step	**User Action**	**System Response**	**Comments**
		• Release Hold • Release All Objects • Release Selected Objects	
4.	Records Officer selects the Release Hold menu option	System displays a confirmation dialog box: "Do you want to release selected Hold(s)?" Yes/No	
5.	Records Officer selects Yes to confirm	1. System populates the following attributes on the affected Holds: • Disposition_Authority_Suspend_Revocation • Disposition_Authority_Suspend_Revocation_Date • Disposition_Authority_Suspend_Revocation_Identifier • Disposition_Authority_Suspend_Revocation_Description 2. System removes the links to the records frozen by the Hold 3. System changes the status of the Hold to Inactive. 4. For all records without any other Holds, system will change the Suspend flag to inactive	

C. Post-Conditions

1. A Disposition Suspension is added to the database
2. The id of the Disposition Suspension is linked to each selected record
3. The *Disposition_Authority_Suspend* attribute is populated on each selected record and this action freezes or exempts the records from destruction.

Scenario 21 – Disassociate Linked Records

John Henry is an action officer in the ABC agency. He has filed a new record that supersedes a previously filed record. His task was to link the superseded record with the superseded by record. After realizing that he linked the wrong records he

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has a Link function associate one or more records with another.
2. Agency has a Search and Retrieve records function that permits the user to search for and retrieve records that he has access to.
3. Agency has predefined link types such as Supporting/Supported; Superseded/Superseded By; Cross Reference; Email/Attachment, etc.
4. Agency has two or more declared records.

B. Script of Operations

Scenario 21 –Disassociate Records			
	Actor Action	**System Response**	**Comments**
1.	User launches the Search for Records function	Systems displays the user interface to the Search for Records function	
2.	User enters search criteria as appropriate to retrieve the desired record on which to remove the link	Systems displays the desired record in a search results display	
3.	User selects the desired record in the search results display	System highlights the specified record	
4.	User navigates to the Display Links function	Systems display the list of links associated with the selected record	
5.	User selects the Link Type to be removed	Systems highlights the selected Link Type	
6.	User selects the delete link option	System displays a conformation dialogue box	

Scenario 21 –Disassociate Records			
	Actor Action	**System Response**	**Comments**
7.	Users selects the Yes button	System enters a null value in the attributes for this specific link on both record: • *Record_ Association_ Id* attribute • *Record_ Association_ Description* attribute • *Record_Association_Date*	

C. Post-Conditions

1. The link has been removed from the incorrectly associated records.

Scenario 26 – Cutoff Records

Jane Doe is a records officer in the ABC agency. She is specifically responsible for scheduling two records management services. The first service is used to calculate the retention of those records whose due dates can be calculated. The second service is used to find the records whose retention periods have expired and to aggregate those records into a session for further processing.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has a retention calculation service that examines each declared record, folder, and box (where applicable) and determines if the retention due date can be calculated (is the trigger date populated).
2. Agency has a disposition session service that will aggregate records, folders, and boxes based upon expired retention dates and disposition action. It will aggregate disposal objects (session) by disposal action (destroy, transfer, review, cutoff, archive, accession to NARA, etc.)
3. Agency has a email notification service that will automatically compose, address, and send an email notification to member(s) of the records staff, providing notification of new sessions to be processed. The notification will identify the session number and disposal action. The notification will also include a web link to the session which provides the capability to view the list of objects included in the session and their details (attributes).
4. Agency has records that are due for cutoff.

B. Script of Operations

Scenario 26 –Cutoff Records			
	Actor Action	**System Response**	**Comments**
1.	System executes the Find Candidate service and identifies the records that are eligible for cutoff.	• System generates a Cutoff session. • System generates and sends email notification to member(s) of the records staff	
2.	Records Officer(s) launches email system and opens email notification.	Email system displays the notification message and a link to the Cutoff session.	
3.	Records Officer clicks the	The system displays a listing of	

Scenario 26 –Cutoff Records			
	Actor Action	**System Response**	**Comments**
	link to the Cutoff session.	the records and folders aggregated in the Cutoff session.	
4.	Records Officer schedules the session for final disposition processing.	System will execute the Cutoff disposition service at the scheduled time	
5.		System cutoff (closes) the folders listed in the session. The folders are closed from further filing.	
6.		System assigns a date to the Date Cutoff attribute on each folder and loose (folder less) records.	Need a *Date Cutoff* attribute
7.		System updates the status of effected records to Closed.	

C. Post-Conditions

1. The *Date Cutoff* attribute is updated/assigned.
2. Folders are closed to further filing.
3. The status of the effected records is changed to Closed.

Scenario 27 – Find Disposition Candidates

Jane Doe is a records officer in the ABC agency. She is specifically responsible for scheduling two records management services. The first service is used to calculate the retention of those records whose due dates can be calculated. The second service is used to find the records whose retention periods have expired and to aggregate those records into a session for further processing.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has an electronic hierarchical category structure in place. File plan consists of subject categories and folders created to aggregate and store digital records.
2. Agency has a retention schedule associated with the category structure in place. The disposition rules are assigned to the subject categories and are inherited by the folders and records.
3. Agency has a retention calculation service that examines each declared record, folder, and box (where applicable) and determines if the retention due date can be calculated (is the trigger date populated).
4. Agency has a disposition session service that will aggregate records, folders, and boxes based upon expired retention dates and disposition action. It will aggregate disposal objects (session) by disposal action (destroy, transfer, review, cutoff, archive, accession to NARA, etc.)
5. Agency has a email notification service that will automatically compose, address, and send an email notification to member(s) of the records staff, providing notification of new sessions to be processed. The notification will identify the session number and disposal action. The notification will also include a web link to the session which provides the capability to view the list of objects included in the session and their details (attributes)

B. Script of Operations

Scenario 27 –Find Disposition Candidates			
	User Action	**System Response**	**Comments**
1.	Records Officer launches the Retention Calculation Service	System displays the retention calculation service data window with the following menu options: • Run At (user specifies a time to execute the	

Scenario 27 –Find Disposition Candidates			
	User Action	**System Response**	**Comments**
		service on a daily, weekly, monthly, or annual basis) • Run From (user specifies a periodic time, such as every 5 minutes, between 7:00 am and 6:00 pm on a daily basis)	
2.	Records Officer selects the *Run At* option	System displays the *Run At* data window	
3.	Records Officer enters run every 5 minutes between the hours of 7:00 am and 6:00 pm daily	System updates the Run At service execution schedule. System calculates retention due dates for records, folders, and boxes at the scheduled intervals and updates their retention due dates.	Instead of re-calculating disposition eligibility date every time • Calculate when a records category is changed, or the disposition instruction associated with it. These events would trigger re-calculation of disposition eligibility date • The government uses event based eligibility a lot, e.g. 5

Scenario 27 –Find Disposition Candidates			
	User Action	**System Response**	**Comments**
			years after official leaves office.
4.	Records Officer launches the Retention Scheduling Service	System displays a user interface to the Retention Scheduling Service with the following menu options: • Disposal Action • Schedule • Department/Office • Category • SQL • Recipient	
5.	Records Officer selects *Disposal Action*	The system displays the Disposal Action dropdown selection list with the following actions: • Archive • Cutoff • Destroy • Review • Transfer • Accession to NARA	These dispositions are not in our model & were not specified by the RMSC IPT. Review would be used when the disposition is not known (category not yet determined).
6.	Records Officer selects the *Archive* disposal action service	System populates the Disposal Action attribute with *Archive* and then displays the menu options data window	This is in the disposition instruction. Isn't this a pre-condition? This doesn't actually set the attribute, the disposal action service uses it to

Scenario 27 –Find Disposition Candidates			
	User Action	**System Response**	**Comments**
			aggregate the records to be Archived, or Destroyed, or whatever the disposition dictates.
7.	Records Officer selects s *Schedule*	The system displays the Schedule data window with the following menu options: • Run At (user specifies a time to execute the service on a daily, weekly, monthly, or annual basis) • Run From (user specifies a periodic time, such as every 5 minutes, between 7:00 am and 6:00 pm on a daily basis)	This implies a service to collect the records based on eligibility date.
8.	Records Officer selects *Run At*	System displays the *Run At* data window	
9.	Records Officer enters run at 10:00 pm on the last day of every month	System updates the Archive Service execution schedule	This implies a service to execute each disposition.
10.	Records Officer selects *Department/Office*	System displays the *Department/Office* data window	These are optional filters if we want to address only a subset of the records.
11.	Records Officer, optionally enters a department or office filter to limit candidates to records from the specified department or office	System updates the *Department/Office* filter	These are optional filters if we want to address only a subset of the records.

Scenario 27 –Find Disposition Candidates			
	User Action	**System Response**	**Comments**
12.	Records Officer selects *Category*	System displays the *Category* data window	These are optional filters if we want to address only a subset of the records.
13.	Records Officer, optionally enters a subject category filter to limit candidates to records from the specified category	System updates the *Category* filter	These are optional filters if we want to address only a subset of the records.
14.	Records Officer selects *SQL*	System displays the *SQL* data window	…or some query-by-example or other query capability.
15.	Records Officer, optionally enters a SQL statement filter to limit candidates to records from the specified SQL statement. Such as all records filed by Jane Doe between a specified period of time)	System updates the *SQL* filter	
16.	Records Officer selects *Recipients*	System displays the *Recipients* data window	This implies that there is a report generator that pushes reports to users (so their emails must be known) This list might be filtered by role. The system must have a list of

Scenario 27 –Find Disposition Candidates			
	User Action	**System Response**	**Comments**
			individuals authorized by action (archive, cutoff, transfer, …)
17.	Record Officer selects the *Recipients* drop down list	System displays a dropdown list of records staff members	
18.	Records Officer selects member(s) of the records staff to receive them email notifications of new sessions to be processed	System updates the *Recipients* list and populates their email address(es) in the service	
19.	Repeat steps 4 – 18 for each desired disposal actions		
20.	System executes the Find Disposition Candidates services	System: • Aggregate all Archive Candidates into an Archive session, affix a name and number to the session, affix the date and time session was created and compose and send email notification to the recipients • Aggregate all Cutoff Candidates into an Cutoff session, affix a name and number to the session, affix the date and time session was created and compose and send email notification to the recipients • Aggregate all Destroy Candidates into an	The list would not be sent via eMail but a link to a place in the system where it can be accessed or acted on. Records that are on "suspend" will be filtered out of the list.

Scenario 27 –Find Disposition Candidates			
	User Action	**System Response**	**Comments**
		Destroy session, affix a name and number to the session, affix the date and time session was created and compose and send email notification to the recipients • Aggregate all Review Candidates into an Review session, affix a name and number to the session, affix the date and time session was created and compose and send email notification to the recipients • Aggregate all Transfer Candidates into an Transfer session, affix a name and number to the session, affix the date and time session was created and compose and send email notification to the recipients • Aggregate all Accession to NARA Candidates into an Accession to NARA session, affix a name and number to the session, affix the date and time session was created and compose and send email notification to the recipients	

C. Post-Conditions

1. Retention is calculated, re-calculated, and updated at each prescribed interval as a service.

2. Retention candidates are found and aggregated per the filter configurations as a service
3. Emails are composed for each new session and sent to the appropriate recipients
4. Records officers receive email notifications of new disposition sessions to be processed.

Scenario 28 – Transfer Records

Jane Doe is a records officer in the ABC agency. She is specifically responsible for scheduling two records management services. The first service is used to calculate the retention of those records whose due dates can be calculated. The second service is used to find the records whose retention periods have expired and to aggregate those records into a session for further processing.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has a retention calculation service that examines each declared record, folder, and box (where applicable) and determines if the retention due date can be calculated (is the trigger date populated).
2. Agency has a disposition session service that will aggregate records, folders, and boxes based upon expired retention dates and disposition action. It will aggregate disposal objects (session) by disposal action (destroy, transfer, review, cutoff, archive, accession to NARA, etc.)
3. Agency has a email notification service that will automatically compose, address, and send an email notification to member(s) of the records staff, providing notification of new sessions to be processed. The notification will identify the session number and disposal action. The notification will also include a web link to the session which provides the capability to view the list of objects included in the session and their details (attributes).
4. Agency has records that are due to be transferred to an external transfer location.

B. Script of Operations

Scenario 28 –Transfer Records			
	Actor Action	**System Response**	**Comments**
1.	System executes the Find Candidate service and identifies the records that are eligible for transfer	• System generates a Transfer session. • System generates and sends email notification to member(s) of the records staff	Transfer packet? PISM needed, xml schema for import/export
2.	Records Officer(s) launches email system and opens email notification.	Email system displays the notification message and a link to the Transfer session.	
3.	Records Officer clicks the	The system displays a listing of	Note: Need a

Scenario 28 –Transfer Records			
	Actor Action	**System Response**	**Comments**
	link to the Transfer session.	the records aggregated in the Transfer Records session by transfer location.	*transfer location* attribute
4.	Records Officer schedules the session for final disposition processing.	System will execute the Transfer disposition service at the scheduled time	
5.		System exports (writes out) a copy of each electronic record in the session to a specified path	
6.		System exports the metadata and metadata change history to a specified path	
7.		System changes the status of the records to “Transferred”.	
8.	Records Officer transfers the records with metadata and history to the appropriate transfer location	Records custodian at the transfer location receives, reviews, and accepts the transfer of the records and assumes custodial responsibilities.	
9.	Records custodian at the transfer location notifies the agency that the transfer action was successfully completed	Agency records officer retrieves the local copy of the transferred records and: • Destroy (expunge) the copies of the transferred records • Delete record metadata and change history of the transferred records or • Optionally, retain the record metadata showing that the details of the transfer (who, what, when, and where)	

Scenario 28 –Transfer Records			
	Actor Action	**System Response**	**Comments**

C. Post-Conditions

1. Copies of electronic records and their metadata (optionally, with change history) have been transferred to the specified location.
2. The local copy of the transferred records has been expunged.
3. The agency optionally retains the metadata of the expunged records and the records status is changed to transferred.

Scenario 29 – Accession to NARA

Jane Doe is a records officer in the ABC agency. She is specifically responsible for scheduling two records management services. The first service is used to calculate the retention of those records whose due dates can be calculated. The second service is used to find the records whose retention periods have expired and to aggregate those records into a session for further processing.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has a retention calculation service that examines each declared record, folder, and box (where applicable) and determines if the retention due date can be calculated (is the trigger date populated).
2. Agency has a disposition session service that will aggregate records, folders, and boxes based upon expired retention dates and disposition action. It will aggregate disposal objects (session) by disposal action (destroy, transfer, review, cutoff, archive, accession to NARA, etc.)
3. Agency has a email notification service that will automatically compose, address, and send an email notification to member(s) of the records staff, providing notification of new sessions to be processed. The notification will identify the session number and disposal action. The notification will also include a web link to the session which provides the capability to view the list of objects included in the session and their details (attributes).
4. Agency has records that are due to be accessioned to NARA.

B. Script of Operations

Scenario 29 –Accession to NARA			
	Actor Action	**System Response**	**Comments**
1.	System executes the Find Candidate service and identifies the records that are eligible for accession by NARA	• System generates Accession to NARA session. A separate accession will be generated for each media type. • System generates and sends email notification to member(s) of the records staff	
2.	Records Officer(s)	Email system displays the	

Scenario 29 –Accession to NARA			
	Actor Action	**System Response**	**Comments**
	launches email system and opens email notification.	notification message and a link to the Accession to NARA sessions	
3.	Records Officer clicks on one of the Accession to NARA session link	The system displays the contents of the selected Accession to NARA session.	
4.	Records Officer generates and submits a report of the records to NARA for approval (repeat this step for each individual session)		There is a NARA specified format/form for this report
5.	Upon receipt of NARA's approval to accession the records, the Records Officer schedules the sessions for final disposition processing.	System will execute the Accession to NARA disposition service at the scheduled time	
6.		System exports (writes out) a copy of each electronic record for each specified session to a specified path	
7.		System exports the metadata and metadata change history to the specified path	
8.		System changes the status of the records to "Transferred to NARA".	
9.	Records Officer transfers the records with metadata and history to NARA	NARA receives, reviews, and accepts the transfer of the records and assumes custodial responsibilities.	
10.	NARA notifies the	Agency records officer	

Scenario 29 –Accession to NARA			
	Actor Action	**System Response**	**Comments**
	agency that the transfer action was successfully completed	retrieves the records that were successfully accessioned by NARA and: • Destroy (expunge) the electronic records • Delete record metadata and change history • Optionally, retain the record metadata showing that the details of the accession to NARA process (who, what, when, and where)	

C. Post-Conditions

1. Copies of electronic records and their metadata (optionally, with change history) have been transmitted to NARA.
2. Agency copy of the transferred records, metadata, and change history are permanently deleted from the agency's recordkeeping system.

Scenario 30 – Destroy Records

Jane Doe is a records officer in the ABC agency. She is specifically responsible for scheduling two records management services. The first service is used to calculate the retention of those records whose due dates can be calculated. The second service is used to find the records whose retention periods have expired and to aggregate those records into a session for further processing.

A. Pre-Conditions.

This section defines the "state" of things before the scenario starts – conditions that have to be true for the scenario to take place.

1. Agency has a retention calculation service that examines each declared record, folder, and box (where applicable) and determines if the retention due date can be calculated (is the trigger date populated).
2. Agency has a disposition session service that will aggregate records, folders, and boxes based upon expired retention dates and disposition action. It will aggregate disposal objects (session) by disposal action (destroy, transfer, review, cutoff, archive, accession to NARA, etc.)
3. Agency has a email notification service that will automatically compose, address, and send an email notification to member(s) of the records staff, providing notification of new sessions to be processed. The notification will identify the session number and disposal action. The notification will also include a web link to the session which provides the capability to view the list of objects included in the session and their details (attributes).
4. Agency has records that are due to be destroyed.

B. Script of Operations

Scenario 30 –Destroy Records			
	Actor Action	**System Response**	**Comments**
1.	System executes the Find Candidate service and identifies the records that are eligible for destruction	• System generates a Destroy session. • System generates and sends email notification to member(s) of the records staff	Note: Records on hold are excluded
2.	Records Officer(s) launches email system and opens email notification.	Email system displays the notification message and a link to the Destroy session.	
3.	Records Officer clicks the	The system displays a listing of	Note: Need a

Scenario 30 –Destroy Records			
	Actor Action	**System Response**	**Comments**
	link to the Destroy session.	the records aggregated in the Destroy session by record owner.	*record owner* (same as record keeper?) attribute
4.	Records Officer(s) generate a Records Destruction Approval Request	A Records Destruction Approval Request is generated and forwarded to each record owner	
5.	Upon receipt of the Records Destruction Approval forms, the Records Officer schedules the session for final disposition processing.	System will execute the Destroy disposition service at the scheduled time	
6.		System destroys (expunges) each electronic record in the session, including all renditions	
7.		System deletes the metadata and the metadata change history	
8.		System generates a Destruction Certificate that captures the details of the destruction action.	Is this a function of the RMS?

C. Post-Conditions

1. Copies of electronic records have been expunged and their metadata (with change history) have been deleted from the recordkeeping system.

.