|  |
| --- |
| OAIS-IF Model Report  Class Model  Version ● Proposed |
| |  | | --- | |  | | Date/Time Generated: | 05/11/2018 12:42:01 | | Author: | David |   EA Repository : C:\Users\David\EA\OAIS-IF-DG.EAP |
| |  |  | | --- | --- | | CREATED WITH |  | |

Table of Contents

1 Class Model 3

1.1 Class Model diagram 3

1.2 $help://class\_model\_pattern.htm 3

1.3 $help://classdiagram.htm 3

1.4 AccessRightsInfo 4

1.5 Archival Information Collection 4

1.6 ArchivalInfoPackage 6

1.7 AssociatedDescriptors 8

1.8 Billing 8

1.9 BitSequence 8

1.10 CollectionDescription 8

1.11 ContextInfo 9

1.12 DataObject 9

1.13 FixityInfo 10

1.14 InfoObject 10

1.15 InfoPackage 13

1.16 ObjectId 14

1.17 OtherRepInfo 15

1.18 PackageDescription 16

1.19 PreservationDescriptionInfo 17

1.20 ProvEntryInfo 17

1.21 ProvenanceInfo 20

1.22 Query 21

1.23 ReferenceInfo 22

1.24 RepInfo 22

1.25 Repository 23

1.26 SearchResult 24

1.27 SemanticRepInfo 25

1.28 StructureRepInfo 26

1.29 URI 27

1.30 User 27

1.31 Repository 27

# Class Model

Package in package 'Model'

Class Model

Version Phase 1.0 Proposed

created on 18/10/2018. Last modified 18/10/2018

## Class Model diagram

Class diagram in package 'Class Model'

Class Model

Version 1.0

David created on 18/10/2018. Last modified 04/11/2018



1. Class Model

## $help://class\_model\_pattern.htm

Text in package 'Class Model'

$help://class\_model\_pattern.htm

Version 1.0 Phase 1.0 Proposed

created on 18/10/2018. Last modified 18/10/2018

Alias Read about Class Modeling

Extends

## $help://classdiagram.htm

Text in package 'Class Model'

$help://classdiagram.htm

Version 1.0 Phase 1.0 Proposed

created on 18/10/2018. Last modified 18/10/2018

Alias View Further Examples

Extends

## AccessRightsInfo

Class in package 'Class Model'

AccessRightsInfo

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from AccessRightsInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |
| Generalization from AccessRightsInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |

## Archival Information Collection

Class in package 'Class Model'

Archival Information Collection

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

Extends ArchivalInfoPackage

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from Archival Information Collection to ArchivalInfoPackage  [ Direction is 'Source -> Destination'. ] |
| Generalization from Archival Information Collection to ArchivalInfoPackage  [ Direction is 'Source -> Destination'. ] |
| Generalization from Archival Information Collection to ArchivalInfoPackage  [ Direction is 'Source -> Destination'. ] |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Bi-Directional) derived from/described by | |
| Source: Public (Class) CollectionDescription | Target: Public (Class) Archival Information Collection |

| OPERATIONS |
| --- |
| getcontentInfo () : InfoObject Public  Properties:  attribute\_name = contentInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getcontentObj () : InfoObject Public  Properties:  attribute\_name = contentObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getdataObj () : DataObject Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getident () : ObjectId Public  Properties:  attribute\_name = ident  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getpdi () : PreservationDescriptionInfo Public  Properties:  attribute\_name = pdi  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getpdi () : PreservationDescriptionInfo Public  Properties:  attribute\_name = pdi  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getrepInfo () : RepInfo Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setcontentInfo (newVal : InfoObject ) : void Public  Properties:  attribute\_name = contentInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setcontentObj (newVal : InfoObject ) : void Public  Properties:  attribute\_name = contentObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setdataObj (newVal : DataObject ) : void Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setident (newVal : ObjectId ) : void Public  Properties:  attribute\_name = ident  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setpdi (newVal : PreservationDescriptionInfo ) : void Public  Properties:  attribute\_name = pdi  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setpdi (newVal : PreservationDescriptionInfo ) : void Public  Properties:  attribute\_name = pdi  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setrepInfo (newVal : RepInfo ) : void Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## ArchivalInfoPackage

Class in package 'Class Model'

ArchivalInfoPackage

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

Extends InfoPackage

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from ArchivalInfoPackage to InfoPackage  [ Direction is 'Source -> Destination'. ] |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from Archival Information Collection to ArchivalInfoPackage  [ Direction is 'Source -> Destination'. ] |
| Generalization from Archival Information Collection to ArchivalInfoPackage  [ Direction is 'Source -> Destination'. ] |
| Generalization from Archival Information Collection to ArchivalInfoPackage  [ Direction is 'Source -> Destination'. ] |

| ATTRIBUTES |
| --- |
| contentInfo : InfoObject Private  [ Is static False. Containment is Not Specified. ] |
| pdi : PreservationDescriptionInfo Private  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getcontentInfo () : InfoObject Public  Properties:  attribute\_name = contentInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getpdi () : PreservationDescriptionInfo Public  Properties:  attribute\_name = pdi  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setcontentInfo (newVal : InfoObject ) : void Public  Properties:  attribute\_name = contentInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setpdi (newVal : PreservationDescriptionInfo ) : void Public  Properties:  attribute\_name = pdi  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## AssociatedDescriptors

Class in package 'Class Model'

AssociatedDescriptors

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from AssociatedDescriptors to CollectionDescription  [ Direction is 'Source -> Destination'. ] |
| Aggregation from AssociatedDescriptors to PackageDescription  [ Direction is 'Source -> Destination'. ] |

## Billing

Class in package 'Class Model'

The charge, if any, for retrieving the ob ject is provided here.

Billing

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

## BitSequence

Class in package 'Class Model'

Sequence of bits

BitSequence

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

## CollectionDescription

Class in package 'Class Model'

CollectionDescription

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from AssociatedDescriptors to CollectionDescription  [ Direction is 'Source -> Destination'. ] |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Bi-Directional)  derived from/described by | |
| Source: Public (Class) CollectionDescription | Target: Public (Class) Archival Information Collection |

## ContextInfo

Class in package 'Class Model'

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

ContextInfo

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from ContextInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Aggregation from ContextInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from ProvenanceInfo to ContextInfo  [ Direction is 'Source -> Destination'. ] |

## DataObject

Class in package 'Class Model'

Either a Physical Object or a Digital Object.

**Digital Object**:An objectcomposed of a set of bit sequences.

**Physical Object**:An object (such as a moon rock, bio-specimen, microscope slide) with physically observable properties that represent information that is considered suitable for being adequately documented for preservation, distribution, and independent usage.

DataObject

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

| ATTRIBUTES |
| --- |
| bits : BitSequence Private  Multiplicity: ( [0..\*], Allow duplicates: 0, Is ordered: False )  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getbits () : BitSequence Public  Properties:  attribute\_name = bits  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setbits (newVal : BitSequence ) : void Public  Properties:  attribute\_name = bits  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## FixityInfo

Class in package 'Class Model'

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

FixityInfo

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from FixityInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Aggregation from FixityInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |

## InfoObject

Class in package 'Class Model'

InfoObject

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from InfoObject to InfoPackage  [ Direction is 'Source -> Destination'. ] |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from PreservationDescriptionInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from InfoPackage to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from ProvEntryInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from ContextInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from FixityInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from ProvenanceInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from PackageDescription to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from ProvenanceInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from ReferenceInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from AccessRightsInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from RepInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |

| ATTRIBUTES |
| --- |
| dataObj : DataObject Private  [ Is static False. Containment is Not Specified. ] |
| ident : ObjectId Private  [ Is static False. Containment is Not Specified. ] |
| repInfo : RepInfo Private  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getdataObj () : DataObject Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getident () : ObjectId Public  Properties:  attribute\_name = ident  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getrepInfo () : RepInfo Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setdataObj (newVal : DataObject ) : void Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setident (newVal : ObjectId ) : void Public  Properties:  attribute\_name = ident  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setrepInfo (newVal : RepInfo ) : void Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## InfoPackage

Class in package 'Class Model'

InfoPackage

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from InfoPackage to InfoObject  [ Direction is 'Source -> Destination'. ] |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from InfoObject to InfoPackage  [ Direction is 'Source -> Destination'. ] |
| Generalization from ArchivalInfoPackage to InfoPackage  [ Direction is 'Source -> Destination'. ] |
| Aggregation from PreservationDescriptionInfo to InfoPackage  [ Direction is 'Source -> Destination'. ] |

| ATTRIBUTES |
| --- |
| contentObj : InfoObject Private  Multiplicity: ( [0..1], Allow duplicates: 0, Is ordered: False )  [ Is static False. Containment is Not Specified. ] |
| pdi : PreservationDescriptionInfo Private  Multiplicity: ( [0..\*], Allow duplicates: 0, Is ordered: False )  [ Is static False. Containment is Not Specified. ] |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Unspecified) | |
| Source: Public (Class) PackageDescription | Target: Public (Class) InfoPackage |

| OPERATIONS |
| --- |
| getcontentObj () : InfoObject Public  Properties:  attribute\_name = contentObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getpdi () : PreservationDescriptionInfo Public  Properties:  attribute\_name = pdi  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setcontentObj (newVal : InfoObject ) : void Public  Properties:  attribute\_name = contentObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setpdi (newVal : PreservationDescriptionInfo ) : void Public  Properties:  attribute\_name = pdi  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## ObjectId

Class in package 'Class Model'

This is an identifier which allows an object to be retrieved, for example It could be a URI

ObjectId

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from ReferenceInfo to ObjectId  [ Direction is 'Source -> Destination'. ] |

| ATTRIBUTES |
| --- |
| object : InfoObject Private  [ Is static False. Containment is Not Specified. ] |
| uri : URI Private  Multiplicity: ( [0..\*], Allow duplicates: 0, Is ordered: False )  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getobject () : InfoObject Public  Properties:  attribute\_name = object  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| geturi () : URI Public  Properties:  attribute\_name = uri  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setobject (newVal : InfoObject ) : void Public  Properties:  attribute\_name = object  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| seturi (newVal : URI ) : void Public  Properties:  attribute\_name = uri  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## OtherRepInfo

Class in package 'Class Model'

OtherRepInfo

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from OtherRepInfo to RepInfo  [ Direction is 'Source -> Destination'. ] |

| OPERATIONS |
| --- |
| getdataObj () : DataObject Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getident () : ObjectId Public  Properties:  attribute\_name = ident  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getrepInfo () : RepInfo Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setdataObj (newVal : DataObject ) : void Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setident (newVal : ObjectId ) : void Public  Properties:  attribute\_name = ident  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setrepInfo (newVal : RepInfo ) : void Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## PackageDescription

Class in package 'Class Model'

PackageDescription

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from PackageDescription to InfoObject  [ Direction is 'Source -> Destination'. ] |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from AssociatedDescriptors to PackageDescription  [ Direction is 'Source -> Destination'. ] |

| ASSOCIATIONS | |
| --- | --- |
| Association (direction: Unspecified) | |
| Source: Public (Class) PackageDescription | Target: Public (Class) InfoPackage |

## PreservationDescriptionInfo

Class in package 'Class Model'

PreservationDescriptionInfo

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from PreservationDescriptionInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Aggregation from PreservationDescriptionInfo to InfoPackage  [ Direction is 'Source -> Destination'. ] |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from ProvenanceInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |
| Aggregation from ContextInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |
| Aggregation from ReferenceInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |
| Aggregation from AccessRightsInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |
| Aggregation from FixityInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |

## ProvEntryInfo

Class in package 'Class Model'

Based on the PROV Data Model for provenance interchange on the webhttps://www.w3.org/TR/2013/NOTE-prov-primer-20130430/

Ontology is

Entity

wasDerivedFrom Entity

wasAtttibutedTo Agent

wasGeneratedBy Activity

Activity

used Entity

wasAssociatedWith Agent

2. Intuitive overview of PROV

2.1 Entities

In PROV, physical, digital, conceptual, or other kinds of thing are called entities. Examples of such entities are a web page, a chart, and a spellchecker. Provenance records can describe the provenance of entities, and an entity’s provenance may refer to many other entities. For example, a document D is an entity whose provenance refers to other entities such as a chart inserted into D, and the dataset that was used to create that chart. Entities may be described as having different attributes and be described from different perspectives. For example, document D as stored in my file system, the second version of document D, and D as an evolving document, are three distinct entities for which we may describe provenance.

2.2 Activities

Activities are how entities come into existence and how their attributes change to become new entities, often making use of previously existing entities to achieve this. They are dynamic aspects of the world, such as actions, processes, etc. For example, if the second version of document D was generated by a translation from the first version of the document in another language, then this translation is an activity.

2.3 Usage and Generation

Activities generate new entities. For example, writing a document brings the document into existence, while revising the document brings a new version into existence. Activities also make use of entities. For example, revising a document to fix spelling mistakes uses the original version of the document as well as a list of corrections. Generation does not always occur at the end of an activity, and an activity may generate entities part-way through. Likewise, usage does not always occur at the beginning of an activity.

2.4 Agents and Responsibility

An agent takes a role in an activity such that the agent can be assigned some degree of responsibility for the activity taking place. An agent can be a person, a piece of software, an inanimate object, an organization, or other entities that may be ascribed responsibility. When an agent has some responsibility for an activity, PROV says the agent was associated with the activity, where several agents may be associated with an activity and vice-versa. Consider a chart displaying some statistics regarding crime rates over time in a linear regression. To represent the provenance of that chart, we could state that the person who created the chart was an agent involved in its creation, and that the software used to create the chart was also an agent involved in that activity. An agent may be acting on behalf of others, e.g. an employee on behalf of their organization, and we can express such chains of responsibility in the provenance.

We can also describe that an entity is attributed to an agent to express the agent's responsibility for that entity, possibly along with other agents. This description can be understood as a shorthand for saying that the agent was responsible for the activity which generated the entity.

One may want to describe the provenance of an agent. For example, an organization responsible for the creation of a report may evolve over time as the report is written as some members leave and others join. To make provenance assertions about an agent in PROV, the agent must be declared explicitly both as an agent and as an entity.

2.5 Roles

A role is a description of the function or the part that an entity played in an activity. Roles specify the relationship between an entity and an activity, i.e. how the activity used or generated the entity. Roles also specify how agents are involved in an activity, qualifying their participation in the activity or specifying for what aspect of it each agent was responsible. For example, an agent may play the role of "editor" in an activity that uses one entity in the role of "document to be edited" and another in the role of "addition to be made to the document", to generate a further entity in the role of "edited document". Roles are application specific, so PROV does not define any particular roles.

2.6 Derivation and Revision

When one entity's existence, content, characteristics and so on are at least partly due to another entity, then we say that the former was derived from the latter. For example, one document may contain material copied from another, and a chart was derived from the data that it illustrates.

PROV allows some common, specialized kinds of derivation to be described. For example, a given entity, such as a document, may go through multiple revisions over time. Between revisions, one or more attributes of the entity may change. In PROV, the result of each revision is a new entity. PROV allows one to relate those entities by making a description that one was a revision of another. Another kind of derivation is to say that one entity, a quotation, was quoted from another entity, commonly a document.

2.7 Plans

Activities may follow pre-defined procedures, such as recipes, tutorials, instructions, or workflows. PROV refers to these, in general, as plans, and allows the description that a plan was followed, by agents, in executing an activity.

2.8 Time

Time is often a critical aspect of provenance. PROV allows the timing of significant events to be described, including when an entity was generated or used, or when an activity started and finished. For example, the model can be used to describe facts such as when a new version of a document was created (generation time), or when a document was edited (start and end of the editing activity).

2.9 Alternate Entities and Specialization

There is often more than one way to describe something in a record of provenance. Each perspective will be referred to by a separately identified entity, and PROV provides a mechanism for linking the different descriptions of the same thing together through the mechanism of specialization. One entity is a specialization of another entity if it shares the same fixed attributes, with the possible addition of further fixed attributes. This concept is best illustrated through a few use cases.

Entities can be mutable things. For example, a webpage is a single entity, W, despite being edited over time. Each version of the webpage is also an entity, W1, W2... To connect an individual version to the webpage in general, we say that the former is a specialization of the latter: W1 is a specialization of W, W2 is a specialization of W, and so on.

Two individuals may create provenance referring to the same thing from different perspectives. For example, the author of an article may attribute that article to themselves using PROV while, independently, a reader might quote a fact from that article elsewhere and document this in PROV. If the author later changes the fact, then from the reader's perspective there are now two versions of the article, and the reader had quoted from the version before the change. From the author's perspective, there is a single article, attributed to the author. If the author, the reader, or a third party, were to connect the two PROV records, that party would say that the article as referred to by the reader is a specialization of the same article as referred to by the author.

The above illustrates where we may want to connect entities by saying that they refer to the same thing, but at different levels of specialization. PROV also allows us to more generally draw a connection between two descriptions of the same thing, even if not at different levels of specialization, describing the entities as alternates of each other. For example, two versions of the webpage above, W1 and W2, are alternates of each other because they describe the same webpage.

As another example, if a file is copied from one directory to another to create a backup, we may say that the copies are alternate versions of the same, location-independent, file. Specifically, we may say that the file in the first directory, entity F1, is an alternate of the file in the second directory, entity F2. Note that it is the context (location) rather than content of the file that differs between the entities in this case.

ProvEntryInfo

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from ProvEntryInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |

| OPERATIONS |
| --- |
| getdataObj () : DataObject Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getident () : ObjectId Public  Properties:  attribute\_name = ident  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getrepInfo () : RepInfo Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setdataObj (newVal : DataObject ) : void Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setident (newVal : ObjectId ) : void Public  Properties:  attribute\_name = ident  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setrepInfo (newVal : RepInfo ) : void Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## ProvenanceInfo

Class in package 'Class Model'

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

ProvenanceInfo

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

Extends ContextInfo, InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from ProvenanceInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |
| Generalization from ProvenanceInfo to ContextInfo  [ Direction is 'Source -> Destination'. ] |
| Generalization from ProvenanceInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |
| Generalization from ProvenanceInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |

| ATTRIBUTES |
| --- |
| provEntry : ProvEntryInfo Private  Multiplicity: ( [0..\*], Allow duplicates: 0, Is ordered: True )  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getprovEntry () : ProvEntryInfo Public  Properties:  attribute\_name = provEntry  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setprovEntry (newVal : ProvEntryInfo ) : void Public  Properties:  attribute\_name = provEntry  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## Query

Class in package 'Class Model'

Test string containing the search, following agreed query syntax

Query

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

| ATTRIBUTES |
| --- |
| queryText : String Private  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getqueryText () : String Public  Properties:  attribute\_name = queryText  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setqueryText (newVal : String ) : void Public  Properties:  attribute\_name = queryText  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## ReferenceInfo

Class in package 'Class Model'

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

ReferenceInfo

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

Extends InfoObject, ObjectId

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from ReferenceInfo to PreservationDescriptionInfo  [ Direction is 'Source -> Destination'. ] |
| Generalization from ReferenceInfo to ObjectId  [ Direction is 'Source -> Destination'. ] |
| Generalization from ReferenceInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |

## RepInfo

Class in package 'Class Model'

RepInfo

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

Extends InfoObject

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Generalization from RepInfo to InfoObject  [ Direction is 'Source -> Destination'. ] |

| INCOMING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from StructureRepInfo to RepInfo  [ Direction is 'Source -> Destination'. ] |
| Aggregation from SemanticRepInfo to RepInfo  [ Direction is 'Source -> Destination'. ] |
| Aggregation from OtherRepInfo to RepInfo  [ Direction is 'Source -> Destination'. ] |

| ATTRIBUTES |
| --- |
| id : ObjectId Private  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getid () : ObjectId Public  Properties:  attribute\_name = id  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setid (newVal : ObjectId ) : void Public  Properties:  attribute\_name = id  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## Repository

Class in package 'Class Model'

Repository

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

| OPERATIONS |
| --- |
| getObject (objId : ObjectId ) : InfoObject Public  The InfoObject may be any of the types of InfoObject or InfoPackages, including an ArchivalOnfoPackage  [ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getRepositoryId (rep : URI ) : Repository Public  [ Is static True. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| login () : User Public  [ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| search (query : Query ) : SearchResult Public  [ Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## SearchResult

Class in package 'Class Model'

SearchResult

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018

| ATTRIBUTES |
| --- |
| billing : Billing Private  Multiplicity: ( [0..\*], Allow duplicates: 0, Is ordered: True )  [ Is static False. Containment is Not Specified. ] |
| resultObjects : ObjectId Private  Multiplicity: ( [0..\*], Allow duplicates: 0, Is ordered: True )  [ Is static False. Containment is Not Specified. ] |

| OPERATIONS |
| --- |
| getbilling () : Billing Public  Properties:  attribute\_name = billing  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getresultObjects () : ObjectId Public  Properties:  attribute\_name = resultObjects  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setbilling (newVal : Billing ) : void Public  Properties:  attribute\_name = billing  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setresultObjects (newVal : ObjectId ) : void Public  Properties:  attribute\_name = resultObjects  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## SemanticRepInfo

Class in package 'Class Model'

SemanticRepInfo

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from SemanticRepInfo to RepInfo  [ Direction is 'Source -> Destination'. ] |

| OPERATIONS |
| --- |
| getdataObj () : DataObject Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getident () : ObjectId Public  Properties:  attribute\_name = ident  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getrepInfo () : RepInfo Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setdataObj (newVal : DataObject ) : void Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setident (newVal : ObjectId ) : void Public  Properties:  attribute\_name = ident  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setrepInfo (newVal : RepInfo ) : void Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## StructureRepInfo

Class in package 'Class Model'

StructureRepInfo

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

| OUTGOING STRUCTURAL RELATIONSHIPS |
| --- |
| Aggregation from StructureRepInfo to RepInfo  [ Direction is 'Source -> Destination'. ] |

| OPERATIONS |
| --- |
| getdataObj () : DataObject Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getident () : ObjectId Public  Properties:  attribute\_name = ident  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| getrepInfo () : RepInfo Public  Properties:  attribute\_name = repInfo  [ Stereotype is «property get». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setdataObj (newVal : DataObject ) : void Public  Properties:  attribute\_name = dataObj  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setident (newVal : ObjectId ) : void Public  Properties:  attribute\_name = ident  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |
| setrepInfo (newVal : RepInfo ) : void Public    Properties:  attribute\_name = repInfo  [ Stereotype is «property set». Is static False. Is abstract False. Is return array False. Is query False. Is synchronized False. ] |

## URI

Class in package 'Class Model'

Uniform Resource Identifier

A Uniform Resource Identifier (URI) is a string of characters that unambiguously identifies a particular resource. To guarantee uniformity, all URIs follow a predefined set of syntax rules[1], but also maintain extensibility through a separately defined hierarchical naming scheme (e.g. "http://").

It could be a URL or URN.

URI

Version 1.0 Phase 1.0 Proposed

David created on 22/10/2018. Last modified 22/10/2018

## User

Class in package 'Class Model'

User default value is Anon - the anonymous user which has not privileges

User

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 22/10/2018

## Repository

Interface in package 'Class Model'

Repository

Version 1.0 Phase 1.0 Proposed

David created on 21/10/2018. Last modified 21/10/2018