Changes from AIM 3.0, Revision 11 to AIM Foundation

You can print and export wiki pages

Unable to render {include} The included page could not be found.

This document describes changes applied to the Annotation and Image Markup (AIM) model between version 3.0, revision 11 and the AIM Foundation model.

Contents of this Page

Introduction

The most significant changes from version 3.0, revision 11 to the AIM Foundation model is the use of ISO 21090 data types, a caBIG® mandate, and the introduction of AIM Statements. An AIM Statement describes a relationship of a finding found on an image or series of images. It represents a relationship between the subject and object entities in the AIM foundation model.

You can download the AIM 3.0 model and the AIM Foundation model.

Why Use AIM Statements?

The AIM 3.0 model reflects relationships between classes using containment of a source class by a target class, and inheritance such as IS-A relationships. The expressive power of the model is limited by these two types of relationships. No other types of relationships in the AIM 3.0 model are possible and not all necessary relationships are present. For instance, there is no direct relationship between instances of the *AnatomicEntity* class, e.g. right-upper lobe of lung, and the *ImagingObservation* class, e.g. mass. Such classes can be indirectly linked to each other only via the *Annotation* class.

The desire to improve the expressiveness of the AIM model and specific use cases prompted us to create a flexible model of AIM statements. Use cases that led us to change the AIM model from the containment association approach to explicitly stated relationships between two classes follow.

> A. Justin Kirby at the NCI's Cancer Imaging Program has a use case for storing information from a mammography case report form (CRF). The CRF has "Associated Findings" or imaging observation characteristics that are associated with the entire breast and are not specific to a mass. Dr. David Channin at Guthrie Clinic also has a similar use case.
>
> B. Dr. Lior Weizman, a research fellow working with Dr. Daniel Rubin, wants to associate calculation results with a DICOM segmentation object.
>
> C. Several AIM users want to capture imaging observations and calculations related directly to image markup.
>
> D. A user wants to capture a measurement of a liver volume from a CT scan to facilitate clinical assessment of liver disorders, to improve decision making in liver transplant surgery and to avoid donor-recipient graft mismatch.

Adding an association relationship between the *AnatomicEntity* and *ImagingObservationCharacteristic* resolved the CRF issue. Adding an association between segmentation and calculation satisfied Dr. Weizman's comment. Applying the same approach could satisfy use cases C and D.

As the AIM model is used by an increasing number of users, additional requests will arise that add relationships between classes or create new classes to store other important information related to AIM. Managing and rearranging associations with classes will be too complex to manage without the AIM *State ment* class and its subclasses.

Summary of Changes Between AIM 3.0 and 4.0 Models

AIM 4.0 has eighty new classes. Twelve were renamed and four were deleted. Changes were also made to the data types, as explained below.

ISO 21090 Data Types

The following table shows the renamed data types.

AIM 3.0 Data Type	ISO 21090
boolean	BL
CalculationResultIdentifier	Not Applicable
ComparisonOperators	Not Applicable
Date	TS
Double	REAL
Integer	INT or II
String	ST or Uid

The following four AIM 3.0 attributes are now mapped to a single ISO 21090 CD data type.

AIM 3.0 Data Type	ISO 21090
codeValue	CD
codeMeaning	CD
codingSchemeDesignator	CD
codingSchemeVersion	CD

New AIM 4.0 Classes

AnnotationCollection

The AIM 3.0 model does not support a collection concept. In AIM 3.0, each AIM annotation is stored as a single AIM XML document or AIM DICOM SR. A typical imaging study generates more than one AIM annotation. Managing AIM annotations of the same study becomes an extra activity that AIM implementers have to deal with. The AIM 4.0 foundation model has adopted the ability to store the same type of related AIM annotations as a single source. AIM has two types of annotations: image annotation and annotation-of-annotation. Image annotations annotate images. Annotation-of-annotations annotate other annotations, including image annotations and annotation-of-annotations. AIM 4.0 foundation, therefore, has a mechanism to manage collections of the same type of AIM annotations.

The *AnnotationCollection* abstract class is the parent of *ImageAnnotationCollection* and *AnnotationofAnnotationCollection*. It provides the general concept that AIM may contain one or more instances of the same type. It associates with two optional classes used to capture information about a person, software, or software manufacturer that is generating AIM instances.

ImageAnnotationCollection

This class is one of two root classes in the model. It inherits all *AnnotationCollection* properties. This class signifies that all members of a collection are of the *ImageAnnotaton* type.

AnnotationofAnnotationCollection

This class is the second root class in the model. It inherits all *AnnotationCollection* properties and signifies that all members of a collection are of type *Annot atonOfAnnotation*.

AnnotationOfAnnotationStatement can be used to create statements about image annotations and annotation-of-annotations.

AnnotationStatement

Relationships in the AIM 3.0 model are represented by inheritance or IS-A relationships and association relationships. These two expressions do not always precisely represent what AIM users want to state in annotations. AIM 4.0 foundation introduces subject-predicate-object statement constructs, called AIM statements, to precisely define relationships between two entities: a subject and an object. A subject and an object can come from a set of particular classes in the AIM model. The introduction of AIM statements requires structural and class name changes from the AIM 3.0 model. The following classes in AIM 3.0 model were renamed and can be used as subjects and objects in AIM 4.0 foundation.

- *Annotation* was renamed as *AnnotationEntity*.
- *AnnatomicEntity* was renamed as *ImagingPhysicalEntity*.
- *ImagingObservation* was renamed as *ImagingObservationEntity*.
- *Inference* was renamed as *InferenceEntity*.
- *AnnotationRole* was renamed as *AnnotationRoleEntity*.
- *GeometricShape* was renamed as *GeometricShapeEntity*.
- *TextAnnotation* was renamed as *TextAnnotationEntity*.
- *ImageReference* was renamed as *ImageReferenceEntity*.
- *DicomImageReference* was renamed as *DicomImageReferenceEntity*.
- *WebImageReference* was renamed as *UriImageReferenceEntity*.
- *Calculation* was renamed to *CalculationEntity*.

The classes above are derived from the *Entity* class. Prior relationships between many classes in the AIM 3.0 model have been deleted; see section 4 below for more information. Instead, an AIM statement can be created using the above classes as the subject and object of an AIM statement. The naming convention used to create an AIM statement is a concatenation between subject, predicate, and object. The current predicates in AIM are as follows.

- Excludes
- Has
- IsComparedWith
- IsCompriseOf
- IsFoundIn
- IsIdentifiedBy
- References
- Uses

AnnotationStatement class is the parent of *ImageAnnotationStatement* and *AnnotationOfAnnotationStatement* classes. It represents a general concept about a statement used to describe a finding that is addressed on an image or images in a series. A statement concept expresses the most granular information that an AIM annotation can have. An AIM annotation presents its content in a collection of semantic statements. Statements describe things found, measured, and/or graphically annotated on an image. There are three types of statements: *AnnotationStatement*, *AnnotationOfAnnotationStatement* and *ImageAnnotationStatement*.

The *AnnotationStatement* class has nine different subtypes of annotation statements that can be applied to both *AnnotationOfAnnotationStatement* and *Ima geAnnotationStatement*. The classes derived from *AnnotationStatement* are in the AIM 4.0 foundation UML diagram section called *AnnotationStatement (common to both types of annotations)*.

A *Statement* class represents relationships via the use of the predicates listed above, inserted between subject and object. Subject and object associations can link to *ImagingPhysicalEntity*, *ImagingObservationEntity*, *InferenceEntity*, *ReferencedAnnotationEntity*, *GeometricShapeEntity*, *TextAnnotationEntity*, *Ur iImageReferenceEntity*, *SegmentationEntity*, *DicomImageReferenceEntity*, *ImageAnnotation*, or *AnnotationOfAnnotation*. Not all combinations between these classes are valid AIM statements. See the *AnnotationStatement*, *AnnotationOfAnnotationStatement*, and *ImageAnnotationStatement* sections in the AIM 4.0 UML diagram for all valid AIM statements.

ImageAnnotationStatement

This class is a parent class of AIM statements that can only be applied to *ImageAnnotation*.

AnnotationOfAnnotationStatement

This class is a parent class of AIM statements that can only be applied to *AnnotationOfAnnotation*.

Entity

The *Entity* abstract class represents the existence of a thing, concept, observation, calculation, measurement, or graphical drawing in AIM. It is a parent class of all entities that are subjects and objects of AIM statements.

MarkupEntity

This abstract class represents graphical drawing and textual description that can be placed on an image.

CompactCalculaitonResult

The result of a calculation captured in a string format. The type attribute of the base class defines the data format in which the string is captured.

A string value of a calculation and its type are determined by the type attribute in this class, which inherits from the *CalculationResult* class. A type can be an array, binary, histogram, matrix, scalar, URI, or vector. An encoding method is applied to the content of the value attribute. A compression method attribute can be used to define the compression algorithm of the value attribute to reduce the size of the value attribute.

ExtendedCalculationResult

This class stores a result of a calculation explicitly with the precise location of each element in the result. It supports sparse matrix type results.

Algorithm

The class provides well-defined instructions for arriving at results.

Parameter

The class represents a set of structured data to be used in the calculation or computation of an algorithm.

AdjudicationObservation

An observation is made about the comparison between two or more performers' clinical results (i.e., "adjudication"). It describes a specific time point or all previous time points as well as those explicitly called out.

AuditTrail

It is used to capture any activity, in coded terms, that requires an entry in the audit trail log, including general and time point lesion observations, time point observations, etc.

SegmentationEntity

This is an abstract class representing the result of a segmentation process where digital images are segmented into a set of pixels. These pixels are stored in a separate file from the original images. Segmentation typically represents a region of interest.

TaskContextEntity

This class contains identifying and descriptive attributes of the reading session and the reading subtask that result in clinical environment or trial results. The class consists of the overall task and the specific subtask. A task represents a unit of overall work, e.g. "Read all of the available time points for the subjects". It may have one or more subtasks. It can be used to capture planned (scheduled) and to record performed tasks.

ThreeDimensionGeometricShapeEntity

This abstract class represents three-dimensional coordinates of a graphical drawing that can be placed on an image to identify a region of interest (ROI).

TwoDimensionGeometricShapeEntity

This abstract class represents two-dimensional coordinates of a graphical drawing that can be placed on an image to identify a region of interest (ROI).

TwoDimensionPoint

This class was originally named Point. A point is defined by a single pixel denoted by a single (column,row) pair.

TwoDimensionCircle

This class was originally named Circle. A circle is defined by two (column,row) pairs. The first point is the central pixel. The second point is a pixel on the perimeter of the circle.

TwoDimensionEllipse

This class was originally named Ellipse. An ellipse is defined by four pixel (column,row) pairs; the first two points specify the endpoints of the major axis and the second two points specify the endpoints of the minor axis of an ellipse.

TwoDimensionPolyline

This class was originally named Polyline. A polyline is defined by a series of connected line segments with ordered vertices denoted by (column,row) pairs; if the first and last vertices are the same, it is a closed polygon.

TwoDimensionMultipoint

This class was originally named Multipoint. A multipoint is defined by multiple pixels each denoted by a (column,row) pair.

ThreeDimensionPoint

This class contains a single location denoted by a single (x,y,z) triplet.

ThreeDimensionMultipoint

This class contains multiple locations each denoted by an (x,y,z) triplet. The points need not be coplanar.

ThreeDimensionPolyline

This class stores a series of connected line segments with ordered vertices denoted by (x,y,z) triplets; the points need not be coplanar.

ThreeDimensionPolygon

This class stores a series of connected line segments with ordered vertices denoted by (x,y,z) triplets, where the first and last vertices shall be the same forming a polygon; the points shall be coplanar.

ThreeDimensionEllipse

An ellipse is defined by four (x,y,z) triplets; the first two triplets specify the endpoints of the major axis and the second two triplets specify the endpoints of the minor axis.

ThreeDimensionEllipsoid

A three-dimensional geometric surface whose plane sections are either ellipses or circles and contains three intersecting orthogonal axes, “a”, “b”, and “c”. The ellipsoid is defined by six (x,y,z) triplets; the first and second triplets specify the endpoints of axis “a”, the third and fourth triplets specify the endpoints of axis “b”, and the fifth and sixth triplets specify the endpoints of axis “c”.

ImagePlane

The class contains common imaging attributes in the DICOM Image Plane module. DICOM modalities that share the same module are CT, MR, RT, and PET.

GeneralImage

General Image specifies the Attributes that identify and describe an image within a particular series.

Classes Derived from Annotation Statement that are Common to Both ImageAnnotation and AnnotationOfAnnotation

CalculationEntityReferencesCalculationEntityStatement

A calculation result can reference another calculation result without using its referenced calculation outcome.

A use case:

A user wants to store measurement results of left and right ventricular parameters from a cardiac MRI study.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create a calculation result to store the left ventricular measurement.
4. Create a calculation result to store the right ventricular measurement.
5. Create *CalculationEntityReferencesCalculationEntityStatement*. It links left and right ventricular measurements.

CalculationEntityUsesCalculationEntityStatement

A calculation result can use another calculation result for its own computation purposes.

A use case:

A user wants to store mean and standard deviation measurement results from a region of interest in a CT study.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create a markup, which is a region of interest of type polygon.
4. Create a calculation result to store the mean value.
5. Create a calculation result to store the standard deviation (SD) result. Note that the mean value is used to calculate SD.
6. Create *CalculationEntityUsesCalculationEntityStatement*. It uses SD calculation result as a subject and mean calculation result as an object of the statement.

ImagingObservationEntityHasCalculationEntityStatement

An image observation can have a calculation result associated with it.

A use case:

A user wants to measure a mass found on an image.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create an imaging observation containing mass, RID3874, RadLex.
4. Create a makeup of type line measurement. The line measurement has a length as a result.
5. Create a calculation result to store the length of the mass.
6. Create *ImagingObservationEntityHasCalculationEntityStatement* to link between the imaging observation (subject) and calculation results (object).

ImagingObservationEntityIsFoundInImagingPhysicalEntityStatement

An image observation can be found in an imaging physical entity.

A use case:

A user wants to state that a mass is found on the left upper lobe of the lung.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series and image instances.
3. Create an imaging observation containing mass, RID3874, RadLex.
4. Create an imaging physical entity containing left upper lobe, RID1327, RadLex.
5. Create *ImagingObservationEntityIsFoundInImagingPhysicalEntityStatement* to link the imaging observation (subject) and imaging physical entity (object).

ImagingPhysicalEntityHasCalculationEntityStatement

An imaging physical entity can have a calculation result.

A use case:

A user wants to store a diagnostic measurement of liver size.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create an imaging physical entity containing liver, RID58, RadLex.
4. Create a calculation result to store the size of the liver.
5. Create *ImagingPhysicalEntityHasCalculationEntityStatement* to link the imaging physical entity (subject) with the calculation result (object).

ImagingPhysicalEntityHasImagingObservationEntityStatement

An image physical entity can have an imaging observation.

A use case:

A user wants to state that there is a mass in the left upper lobe of a patient.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series and image instances.
3. Create an imaging physical entity containing left upper lobe, RID1327, RadLex.
4. Create an imaging observation containing mass, RID3874, RadLex.
5. Create *ImagingPhysicalEntityHasImagingObservationEntityStatement* to link imaging physical entity (subject) with imaging observation (object).

AnnotationEntityHasPerformedTaskContextEntityStatement

The class is used to record an activity or a performed task used to create an AIM instance. The AIM instance may contain zero or more performed tasks. Either the *ImageAnnotation* or *AnnotationOfAnnotation* class can be a subject of this statement.

A use case:

A user wants to record activities performed to create an AIM instance.

Working with AIM:

For example, a user wants to record lesion activity.

1. Create *ImageAnnotation*.
2. Create *GeometricShapeEntity* and record user makeup.
3. Create *TaskContextEntity* and fill in the required information; e.g., task name and subtask. The task information must be related to markup activity.
4. Create *AnnotationEntityHasPerformedTaskContextEntityStatement* to link *ImageAnnotation* (subject) with *TaskContextEntity* (object).

AnnotationEntityHasPlannedTaskContextEntityStatement

The class is used to record a planned activity or task. The task is used to inform users or computer programs of what activity needs to be performed in order to create an AIM instance. The instance may be created from one or more planned tasks. Either the *ImageAnnotation* or *AnnotationOfAnnotation* class can be a subject of this statement.

A use case:

A user wants to plan or define activities to be performed in order to create an AIM instance. For example, users want to create a planned activity to guide imaging interpreters in creating an AIM instance.

Working with AIM:

1. Create *ImageAnnotation*.
2. Create *TaskContextEntity*and fill in the required information; e.g., task name and subtask. Both subtasks below have the same task information; e.g., search for a lesion in an organ.
    a. Identify an imaging observation found on an image.
    b. Identify an anatomic entity location associated with the imaging observation.
3. Create *AnnotationEntityHasPerformedTaskContextEntityStatement* to link *ImageAnnotation* (subject) with *TaskContextEntity* (object).

Classes Derived from AnnotationOfAnnotationStatement

AnnotationOfAnnotationHasAnnotationOfAnnotationStatement

An instance of *AnnotationOfAnnotation* has another instance of *AnnotationOfAnnotation*. An *AnnotationOfAnnotation* may be used to compare, evaluate or reference one or more instances of *AnnotationOfAnnotation* and/or *ImageAnnotation*.

A use case:

An adjudicator wants to compare a result in an annotation-of-annotation instance with another image annotation instance.

Assumption:

1. There is an annotation-of-annotation instance with an additional image annotation instance of the same lesion.
2. There is a system capable of reading and extracting information from the annotation-of-annotation instance and image annotation.

Working with AIM:

1. Create a new annotation-of-annotation instance.
2. Extract calculation result from annotation-of-annotation and other information to display to the user.
3. Create *AnnotationOfAnnotationHasAnnotationOfAnnotationStatement* to link the newly created annotation-of-annotation first (subject) with the existing annotation-of-annotation (object).
4. Compare the results of the existing annotation-of-annotation with the user's own measurement and evaluation.
5. Create and store the calculation entity after comparing the difference between the two sizes from existing annotation-of-annotation and new measurements.
6. Create an inference entity to store the result, such as *smaller* or *larger* to the question: "Is the size getting larger or smaller?"
7. Create *AnnotationOfAnnotationHasCalculationEntityStatement* to associate the annotation-of-annotation (subject) with the calculation entity (object).

AnnotationOfAnnotationHasAnnotationRoleEntityStatement

A given instance of type *AnnotationOfAnnotation* can have an assigned role. Examples of roles can be baseline, follow-up, referenced case, etc. They are captured in the model as coded terms in the *AnnotationRoleEntity* class. Some of these roles have been defined in the DICOM standard part 16, Content Mapping Resource. For example:

Baseline Category:

DCM 112074 Target Lesion at Baseline

DCM 112075 Non-Target Lesion at Baseline

DCM 112076 Non-Lesion at Baseline

A use case:

A user wants to assign a baseline role to an annotation-of-annotation consisting of image annotations of a non-target lesion.

Assumption:

1. Image annotations for a non-target lesion were created earlier.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create an annotation role entity containing Baseline Category, 112016, DCM.
3. The annotation-of-annotation has a statement referencing image annotations with the non-target lesion.
4. Create *AnnotationOfAnnotationHasAnnotationRoleEntityStatement* to link the annotation-of-annotation (subject) to the annotation role entity (object).

AnnotationOfAnnotationHasCalculationEntityStatement

An instance of *AnnotationOfAnnotation* can have one or more calculation results. The instance can reference one calculation at a time in a statement. For example, if there are three calculation results, there must be three AnnotationOfAnnotationHasCalculationEntityStatements.

A use case:

An adjudicator wants to compute an average size of the mass from the three markups created by three readers.

Assumptions:

1. Three image annotations for a target lesion were created earlier from three different readers.
2. There is a system capable of reading and extracting information from the three annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create three *AnnotationOfAnnotationHasImageAnnotationStatements* linking annotation-of-annotation (subjects) to image annotations (objects).
3. Extract the mass size from each image annotation and calculate an average.
4. Create *CalculationEntity* and store the average result calculation and other required information.
5. Create *AnnotationOfAnnotationHasCalculationEntityStatement* to link the annotation-of-annotation (subject) to the calculation entity (object).

AnnotationOfAnnotationHasImageAnnotationStatement

An instance of *AnnotationOfAnnotation* can reference existing image annotations. Results from image annotations can be used for further analysis, computation, comparison, reference, etc.

A use case:

An adjudicator wants to annotate a study read by three different readers.

Assumption:

1. Three image annotations for a target lesion were created earlier from three different readers.
2. There is a system capable of reading and extracting information from the three annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create three *AnnotationOfAnnotationHasImageAnnotationStatements* to link the annotation-of-annotation (subject) with each instance of image annotation (object).
3. The adjudicator may want to create additional imaging physical entity statements and imaging observation entity statements, etc.

AnnotationOfAnnotationHasImagingObservationEntityStatement

An instance of *AnnotationOfAnnotation* can have one or more imaging observations associated with the instance. *AnnotationOfAnnotationHasImagingObservationEntityStatement* expresses a relationship between the instance of *AnnotationOfAnnotation* and an imaging observation.

A use case:

An adjudicator wants to annotate imaging observations on a study read by three different readers.

Assumption:

1. Three image annotations for a target lesion were created earlier from three different readers.
2. There is a system capable of reading and extracting information from the three annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create three *AnnotationOfAnnotationHasImageAnnotationStatements* linking annotation-of-annotation (subject) to image annotations (objects).
3. Create an *ImagingObservationEntity* and store the related question, the imaging observation result, and optional information. See *ImagingObservationEntity*.

4. Create an *AnnotationOfAnnotationHasImagingObservationEntityStatement* to link annotation-of-annotation (subject) to the imaging observation entity (object).

AnnotationOfAnnotationHasImagingPhysicalEntityStatement

An instance of *AnnotationOfAnnotation* can reference an imaging physical entity, an anatomical part, or a physical object that can be identified on an image.

A use case:

An adjudicator wants to annotate an imaging physical entity on a study read by three different readers.

Assumption:

1. Three image annotations for a target lesion were created earlier by three different readers.
2. There is a system capable of reading and extracting information from the three annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create three *AnnotationOfAnnotationHasImageAnnotationStatements* linking annotation-of-annotation (subject) to image annotations (objects).
3. Create *ImagingPhysicalEntity* and store the related question, the imaging physical entity result, and optional information. See *ImagingPhysicalEntit*
.
4. Create an *AnnotationOfAnnotationHasImagingPhysicalEntityStatement* to link the annotation-of-annotation (subject) to the imaging physical entity (object).

AnnotationOfAnnotationHasInferenceEntityStatement

An instance of *AnnotationOfAnnotation* can have a conclusion derived by interpreting images and/or other supplemental information related to the images. The conclusion is stored in *InferenceEntity*.

A use case:

An adjudicator wants to provide a medical conclusion to a study read by three different readers.

Assumption:

1. Three image annotations for a target lesion were created earlier from three different readers.
2. There is a system capable of reading and extracting information from the three annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create three *AnnotationOfAnnotationHasImageAnnotationStatements* linking annotation-of-annotation (subject) to image annotations (object).
3. Create an *InferenceEntity* and store the related question and medical conclusion. See *InferenceEntity*.
4. Create *AnnotationOfAnnotationHasInferenceEntityStatement* to link the annotation-of-annotation (subject) to the inference entity (object).

AnnotationOfAnnotationIsComparedWithAnnotationOfAnnotationStatement

An instance of *AnnotationOfAnnotation* can be compared to another instance of *AnnotationOfAnnotation*. AIM users can further create *CalculationEntityIsC omparedWithCalculationEntityStatement* to compare a calculation result from the subject *AnnotationOfAnnotation* instance with a calculation result from the object *AnnotationOfAnnotation*.

A use case:

An adjudicator wants to compare annotation-of-annotations' calculation results from two different time points--baseline and first follow-up.

Assumption:

1. There are two annotation-of-annotations from two time points, baseline and the first follow-up.
2. There is a system capable of reading and extracting information from the annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create two *AnnotationOfAnnotationHasAnnotationOfAnnotationStatements* linking a newly created annotation-of-annotation (subject) to the two existing annotation-of-annotations (objects).
3. Create an *AnnotationOfAnnotationIsComparedWithAnnotationOfAnnotationStatement* to link baseline (subject) and follow-up (object) instances.
4. Extract a calculation result from the baseline and the first follow-up annotation-of-annotation instances.
5. Compare results.
6. Create and store the calculation entity if comparing the difference between two sizes.
7. Create an inference entity to store the result, such as *smaller* or *larger*, to the question "Is the size getting larger or smaller?"
8. Create an *AnnotationOfAnnotationHasCalculationEntityStatement* to associate the annotation-of-annotation (subject) with the calculation entity (object).
9. Create an *AnnotationOfAnnotationHasInferenceEntityStatement* to associate the annotation-of-annotation (subject) with an inference entity (object).

AnnotationOfAnnotationIsComparedWithImageAnnotationStatement

An instance of *AnnotationOfAnnotation* can be compared with an instance of *ImageAnnotation*. AIM users can further create a *CalculationEntityIsComparedWithCalculationEntityStatement* to compare a calculation result from the subject *AnnotationOfAnnotation* instance with a calculation result from the object *ImageAnnotation*.

A use case:

An adjudicator wants to compare an annotation-of-annotation calculation result from the baseline and an image annotation created by a reader from the first follow-up.

Assumption:

1. There is an annotation –f-annotation from the first time point and an image annotation from the first follow-up.
2. There is a system capable of reading and extracting information from the annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create an *AnnotationOfAnnotationHasAnnotationOfAnnotationStatement* linking the annotation-of-annotation (subject) to the baseline annotation-of-annotation (object).
3. Create an *AnnotationOfAnnotationHasImageAnnotationStatement* linking the annotation-of-annotation (subject) to the first follow-up image annotation (object).
4. Create *AnnotationOfAnnotationIsComparedWithImageAnnotationStatement* to link the baseline annotation-of-annotation (subject) with the follow-up image annotation (object).
5. Extract a calculation result from the baseline annotation-of-annotation and first follow-up image annotation.
6. Compare results.
7. Create and store the calculation entity if comparing the difference between two sizes.
8. Create an inference entity to store the result, such as *smaller* or *larger*,to the question "Is the size getting larger or smaller?"
9. Create an *AnnotationOfAnnotationHasCalculationEntityStatement* to associate the annotation-of-annotation (subject) with a calculation entity (object).
10. Create an *AnnotationOfAnnotationHasInferenceEntityStatement* to associate the annotation-of-annotation (subject) with inference entity (object).

CalculationEntityIsComparedWithCalculationEntityStatement

When an AIM user wants to compare two calculation results, the user can use *CalculationEntityIsComparedWithCalculationEntityStatement* to identify a subject and object of calculation results.

This statement should not exist alone. There should be a statement such as *AnnotationOfAnnotationIsComparedWithAnnotationOfAnnotationStatement* or *AnnotationOfAnnotationIsComparedWithImageAnnotationStatement* existing alongside the *CalculationEntityIsComparedWithCalculationEntityStatement*.

A use case:

An adjudicator wants to compare an annotation-of-annotation calculation result from the baseline and an image annotation created by a reader from the first follow-up.

Assumption:

1. There is an annotation-of-annotation from the first time point and image annotation from the first follow-up.
2. There is a system capable of reading and extracting information from the annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create *AnnotationOfAnnotationHasAnnotationOfAnnotationStatement* linking the annotation-of-annotation (subject) to the baseline annotation-of-annotation (object).
3. Create an *AnnotationOfAnnotationHasImageAnnotationStatement* linking the annotation-of-annotation to the first follow-up image annotation (object).
4. Create an *AnnotationOfAnnotationIsComparedWithImageAnnotationStatement* to link the baseline annotation-of-annotation (subject) with the follow-up image annotation (object).
5. Create *CalculationEntityIsComparedWithCalculationEntityStatement* to link the calculation from the baseline annotation-of-annotation (subject) with the calculation from the image annotation (object).
6. Extract a calculation result from the baseline annotation-of-annotation and the first follow-up image annotation.
7. Compare the results.
8. Create and store the calculation entity if comparing the difference between two sizes.
9. Create an inference entity to store the result, such as *smaller* or *larger*, to the question "Is the size getting larger or smaller?"
10. Create an *AnnotationOfAnnotationHasCalculationEntityStatement* to associate the annotation-of-annotation (subject) with a calculation entity (object).
11. Create an *AnnotationOfAnnotationHasInferenceEntityStatement* to associate the annotation-of-annotation (subject) with an inference entity (object).

ImageAnnotationHasAnnotationRoleEntityStatement

This class is used to assign an annotation role to an image annotation. For example, an image annotation can be a baseline in one study but a follow-up in another study.

A use case:

A user wants to assign a baseline role to an image annotation of a non-target lesion.

Assumption:

- An image annotation for a non-target lesion was created earlier.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create an annotation role entity containing Baseline Category, 112016, DCM.
3. Create an *AnnotationOfAnnotationHasImageAnnotationStatement* to link an annotation-of-annotation (subject) with an image annotation (object).
4. Create an *ImageAnnotationHasAnnotationRoleEntityStatement* to link the image annotation (subject) to the annotation role entity (object).

ImageAnnotationIsComparedWithImageAnnotationStatement

An instance of *ImageAnnotation* is compared with another instance of *ImageAnnotation*. AIM users can further create *CalculationEntityIsComparedWithCalculationEntityStatement* to compare a calculation result from the subject *ImageAnnotation* instance with a calculation result from the object *ImageAnnotation*.

A use case:

An adjudicator wants to compare image annotation calculation results from two different time points, baseline and the first follow-up.

Assumption:

1. There are two image annotations from two time points--baseline and the first follow-up.
2. There is a system capable of reading and extracting information from the annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create two *AnnotationOfAnnotationHasImageAnnotationStatement* annotation-of-annotation statements to link the annotation-of-annotation (subject) to each image annotation instance (object).
3. Create an *ImageAnnotationIsComparedWithImageAnnotationStatement* to link baseline (subject) and follow-up (object) instances.
4. Extract calculation results from the baseline and the first follow-up image annotation instances.
5. Compare the results.
6. Create and store the calculation entity if comparing the difference between two sizes.
7. Create an inference entity to store the result, such as *smaller* or *larger*, to the question "Is the size getting larger or smaller?", PRV-SIZE01 and Private.
8. Create an *AnnotationOfAnnotationHasCalculationEntityStatement* to associate the annotation-of-annotation (subject) with a calculation entity (object).
9. Create an *AnnotationOfAnnotationHasInferenceEntityStatement* to associate the annotation-of-annotation (subject) with an inference entity (object).

ImageAnnotationIsComparedWithAnnotationOfAnnotationStatement

An instance of *ImageAnnotation* is compared with another instance of *AnnotationOfAnnotation*. AIM users can further create *CalculationEntityIsComparedWithCalculationEntityStatement* to compare a calculation result from the subject *AnnotationOfAnnotation* instance with a calculation result from the object *ImageAnnotation*.

A use case:

A user wants to compare an image annotation calculation result with an annotation-of-annotation calculation result created by an adjudicator.

Assumption:

1. There is an image annotation with a calculation result.
2. There is an annotation-of-annotation with a calculation result.
3. There is a system capable of reading and extracting information from the annotations for further computational and manipulation purposes.

Working with AIM:

1. Create an annotation-of-annotation instance.
2. Create an *AnnotationOfAnnotationHasAnnotationOfAnnotationStatement* to link the annotation-of-annotation (subject) with the referenced annotation-of-annotation (object).
3. Create an *AnnotationOfAnnotationHasImageAnnotationStatement* to link the annotation-of-annotation (subject) with the instance of image annotation (object).
4. Create an *ImageAnnotationIsComparedWithAnnotationOfAnnotationStatement* to link the image annotation (subject) with the referenced annotation-of-annotation (object).
5. Extract the calculation results from annotation-of-annotation and image annotation.
6. Compare results.
7. Create and store the calculation entity after comparing the difference between the two sizes.
8. Create an inference entity to store the result, such as smaller or larger, to the question "Is the size getting larger or smaller?"
9. Create an *AnnotationOfAnnotationHasCalculationEntityStatement* to associate the annotation-of-annotation (subject) with calculation entity (object).
10. Create an *AnnotationOfAnnotationHasInferenceEntityStatement* to associate the annotation-of-annotation (subject) with inference entity (object).

Classes derived from ImageAnnotationStatement

DicomImageReferenceEntityHasCalculationEntityStatement

A DICOM image can have a calculation associated with the image.

A use case:

A user wants to store a calculation result associated with a DICOM image. The calculation comes from a region of interest (ROI) of a mass. The user uses a freehand drawing tool to outline the mass region.

Working with AIM:

1. Create an image annotation instance.
2. Create *DicomImageReferenceEntity* containing an image with a mass.
3. Create *ImageAnnotationHasDicomImageReferenceEntityStatement* to link the image annotation (subject) to the DICOM image reference entity (object).
4. The user creates the ROI of the mass.
5. Compute area of the mass.
6. Create *CalculationEntity* and store the ROI result.
7. Create *DicomImageReferenceEntityHasCalculationEntityStatement* to link the DICOM image (subject) with the calculation entity (object).

DicomImageReferenceEntityHasImagingObservationEntityStatement

A DICOM image, captured in *DicomImageReferenceEntity*, can be associated with an imaging observation, captured in *ImagingObservationEntity*, to describe an observation on the image.

A use case:

A user wants to store an imaging observation associated with a DICOM image.

Working with AIM:

1. Create an image annotation instance.
2. Create *DicomImageReferenceEntity* containing an image with a mass.
3. Create an *ImageAnnotationHasDicomImageReferenceEntityStatement* to link the image annotation (subject) to the DICOM image reference entity (object).
4. The user creates ROI of the mass.
5. The user provides an imaging observation with the values of mass, RID3874, RadLex.
6. Create *ImagingObservationEntity* and store the value above.
7. Create *DicomImageReferenceEntityHasImagingObservationEntityStatement* to link the DICOM image (subject) with the imaging observation entity (object).

DicomImageReferenceEntityHasImagingPhysicalEntity

A DICOM image, captured in *DicomImageReferenceEntity*, can associate with an image physical entity, captured in *ImagingPhysicalEntity*, to describe an observation on the image.

A use case:

A user wants to store an imaging physical entity associated with a DICOM image.

Working with AIM:

1. Create an image annotation instance.
2. Create *DicomImageReferenceEntity* containing an image with a mass.
3. Create *ImageAnnotationHasDicomImageReferenceEntityStatement* to link the image annotation (subject) to the DICOM image reference entity (object).
4. The user creates the ROI of the mass.
5. The user provides an imaging physical entity with the values of left upper lobe, RID1327, RadLex.
6. Create *ImagingPhysicalEntity* and store the value above.
7. Create *DicomImageReferenceEntityHasImagingPhysicalEntity* to link DICOM image (subject) with the imaging physical entity (object).

DicomSegmentationEntityHasImagingObservationEntityStatement

A DICOM segmentation object can have an imaging observation to further describe the segmentation object.

A use case:

A user wants to associate a DICOM segmentation with an imaging observation.

Assumption:

- There is a system capable of creating a DICOM segmentation object based on a user ROI for a set of images in a series.

Working with AIM:

1. Create an image annotation instance.
2. Create *DicomImageReferenceEntity* containing images with a mass.
3. Create *ImageAnnotationHasDicomImageReferenceEntityStatement* to link the image annotation (subject) to the DICOM image reference entity (object).
4. The user designates the ROI of the mass.
5. The system creates a DICOM segmentation object.
6. Create *DicomSegmentationEntity* with the required information from step 5.

7. The user provides an imaging observation with values of mass, RID3874, RadLex.
8. Create *ImagingObservationEntity* and store the value above.
9. Create *DicomSegmentationEntityHasImagingObservationEntityStatement* to link DICOM segmentation entity (subject) with the imaging observation entity (object).

ImageAnnotationHasCalculationEntityStatement

An image annotation can have a calculation result associated with it.

A use case:

A user wants to compute and store a size of the mass from an image.

Working with AIM:

1. Create an image annotation instance.
2. Create a markup to measure the size of a mass.
3. Extract the mass size.
4. Create *CalculationEntity* and store the result and other required information from step 3.
5. Create *ImageAnnotationHasCalculationEntityStatement* to link the image annotation (subject) to the calculation entity (object).

ImageAnnotationHasChildImageAnnotationStatement

An image annotation can have a child image annotation. This represents a hierarchical structure between images. A child image implicitly has a parent image. A use case of this type of statement is a diagnostic mammogram image that has another magnification image associated with the diagnostic image. Another use case is a whole slide image with a 10X magnification factor that may have a section of the image magnified and stored as a separate image. Both images can have a parent-child relationship.

A use case:

A user wants to link a mammography image with a corresponding magnification image.

Assumption:

- There is a *DicomImageReferenceEntity* containing the original image.

Working with AIM:

1. Create an image annotation instance.
2. User views a left mediolateral oblique (MLO)
3. User creates a magnification view of the ROI and save it as a DICOM secondary captured image.
4. Create *DicomImageReferenceEntity* containing the secondary captured image.
5. Create *ImageAnnotationHasChildImageAnnotationStatement* to link the image annotation (subject) to an instance of *DicomImageReferenceEntity* containing secondary captured image (object).

ImageAnnotationHasDicomImageReferenceEntityStatement

An instance of image annotation can have DICOM images of the same series.

A use case:

A user views a set of CT axial series.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create *ImageAnnotationHasDicomImageReferenceEntityStatement* to link image annotation (subject) and DICOM image reference (object).

ImageAnnotationHasDicomSegmentationEntityStatement

An image annotation can have one or more DICOM segmentation objects. An *ImageAnnotationHasDicomSegmentationEntityStatement* statement represents a direct relationship between an image annotation and a DICOM segmentation entity. If the image annotation has three DICOM segmentation entities, there needs to be three *ImageAnnotationHasDicomSegmentationEntityStatements*.

A use case:

A user found two nodules in a CT axial series. The user used a workstation to create two DICOM segmentation objects. The user annotated and stored the findings in AIM format with segmentation information.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. An image annotation has the set of CT images in the series. We need to create a statement that associates image annotation with DICOM images.

** Use *ImageAnnotationHasDicomImageReferenceEntityStatement*

** The statement has the image annotation instance as subject and DICOM image reference instance as object.

1. Create two DICOM segmentation entity instances. Each instance references DICOM segmentation object from step 2.
2. Create two statements of type *ImageAnnotationHasDicomSegmentationEntityStatement*. Each statement has the same image annotation (subject) but different DICOM segmentation entity (object).

ImageAnnotationHasImagingObservationEntityStatement

An image annotation can have an imaging observation associated with it.

A use case:

A user wants to state that a mass is found on an image.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create an imaging observation containing mass, RID3874, RadLex.
4. Create *ImageAnnotationHasImagingObservationEntityStatement* to link the imaging annotation (subject) with imaging observation (object).

ImageAnnotationHasImagingPhysicalEntityStatement

An image annotation can have an imaging physical entity associated with it.

A use case:

A user wants to label the left upper lobe of lung on an image.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create an imaging physical entity containing left upper lobe, RID1327, RadLex.
4. Create *ImageAnnotationHasImagingPhysicalEntityStatement* to link the imaging annotation (subject) and imaging physical entity (object).

ImageAnnotationHasInferenceEntityStatement

An image annotation can have a conclusion derived by interpreting images and/or other supplemental information related to the images.

A use case:

A user wants to provide medical conclusion for a study.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create *InferenceEntity* and store the related question and medical conclusion. See *InferenceEntity*.
4. Create *ImageAnnotationHasInferenceEntityStatement* to link an image annotation (subject) to the inference entity (object).

ImageAnnotationHasTextAnnotationEntityStatement

An image annotation can have a text annotation associated with it.

A use case:

A user creates a text markup on an image from a set of CT axial series.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create a text markup on a mass.
4. Create *TextAnnotationEntity* to store information in step 3.
5. Create *ImageAnnotationHasTextAnnotationEntityStatement* to link the image annotation (subject) with the text annotation entity (object).

ImageAnnotationHasThreeDimensionGeometricShapeEntityStatement

One or more three-dimensional graphic drawings may be associated with an instance of an image annotation. *ImageAnnotationHasThreeDimensionGeometricShapeEntityStatement* represents a direct relationship between an instance of image annotation and a drawing. If you have two drawings on an image, you will need to create two statements.

A use case:

A user creates a three-dimensional markup on an image from a set of CT axial series.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create a three-dimensional polyline markup on a mass.
4. Create *ThreeDimensionPolyline* to store information in step 3.
5. Create *ImageAnnotationHasThreeDimensionGeometricShapeEntityStatement* to link the image annotation (subject) with the *ThreeDimensionPolyline* entity (object).

ImageAnnotationHasTwoDimensionGeometricShapeEntityStatement

An instance of image annotation may have one or more two-dimensional graphical drawings associate with it. *ImageAnnotationHasTwoDimensionGeometricShapeEntityStatement* represents a direct relationship between an instance of image annotation and a drawing. If you have two drawings on an image, create two statements.

A use case:

A user creates a two-dimensional markup on an image from a set of CT axial series.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create a two-dimension polyline markup on a mass.
4. Create *TwoDimensionPolyline* to store information in step 3.
5. Create *ImageAnnotationHasTwoDimensionGeometricShapeEntityStatement* to link the image annotation (subject) with the *TwoDimensionPolyline* entity (object).

ImageAnnotationHasUriImageReferenceEntityStatement

An image annotation can have a URI reference that directs to the actual storage location of the image used in the image annotation.

A use case:

A user views a set of CT axial series from URI sources.

Working with AIM:

1. Create an image annotation instance.

2. Create *UriImageReferenceEntity* for each image in the CT series.

3. Create *ImageAnnotationHasUriImageReferenceEntityStatements* to link the image annotation (subject) with each *UriImageReferenceEntity* (object).

ImagingObservationEntityIsIdentifiedByThreeDimensionGeometricShapeEntityStatement

An instance of imaging observation entity can be identified by one or more three-dimensional graphical drawings associate with it. *ImageObservationEntityIsIdentifiedByThreeDimensionGeometricShapeEntityStatement* represents a direct relationship between an instance of imaging observation entity and a markup. If you have two markups on an image, create two statements.

A use case:

A user creates a three-dimensional markup on an image from a set of CT axial series.

Working with AIM:

1. Create an imaging observation entity instance.
2. Create a DICOM image reference instance and its associated image study, image series and image instances using *DicomImageReferenceEntity*.
3. Create a three-dimensional polyline markup on a mass.
4. Create *ThreeDimensionPolyline* to store information in step 3.
5. Create *ImagingObservationEntityIsIdentifiedByThreeDimensionGeometricShapeEntityStatement* to link the imaging observation entity (subject) with the *ThreeDimensionPolyline* entity (object).

ImagingObservationEntityIsIdentifiedByTwoDimensionGeometricShapeEntityStatement

An instance of imaging observation entity can be identified one or more three-dimensional graphical drawings associated with it. *ImageObservationEntityIsIdentifiedByThreeDimensionGeometricShapeEntityStatement* represents a direct relationship between an instance of an imaging observation entity and a markup. If you have two markups on an image, create two statements.

A use case:

A user creates a three-dimensional markup on an image from a set of CT axial series.

Working with AIM:

1. Create an imaging observation entity instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create a two-dimensional polyline markup on a mass.
4. Create *TwoDimensionPolyline* to store information in step 3.

5. Create *ImagObservationIsIdentifiedByTwoDimensionGeometricShapeEntityStatement* to link the imaging observation entity (subject) with the *Two DimensionPolyline* entity (object).

ImagingObservationEntityIsIdentifiedByTextAnnotationEntityStatement

An imaging observation can be identified by a text annotation.

A use case:

A user wants to create a text markup for a mass found on an image.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances.
3. Create an imaging observation containing mass, RID3874, RadLex.
4. Create a text markup on a mass.
5. Create a text annotation entity based on information from step 4.
6. Create an *ImagingObservationEntityIsIdentifiedByTextAnnotationEntityStatement* to link the imaging observation (subject) with the text annotation entity (object).

ImagingPhysicalEntityHasThreeDimensionGeometricShapeEntityStatement

An instance of imaging physical entity may have one or more three-dimensional drawings associate with it. *ImagePhysicalEntityHasThreeDimensionGeometricShapeEntityStatement* represents a direct relationship between an instance of an imaging physical entity and a markup. If you have two markups on an image, you create two statements.

A use case:

A user creates a three-dimensional markup on an image from a set of CT axial series.

Working with AIM:

1. Create an imaging physical entity instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create an imaging physical entity containing left upper lobe, RID1327, RadLex.
4. Create a three-dimensional polyline markup on a mass.
5. Create *ThreeDimensionPolyline* to store information in step 3.
6. Create *ImagingPhysicalEntityHasThreeDimensionGeometricShapeEntityStatement* to link the imaging physical entity (subject) with the *ThreeDimensionPolyline* entity (object).

ImagingPhysicalEntityHasTwoDimensionGeometricShapeEntityStatement

An instance of imaging physical entity may have one or more two-dimensional drawings associate with it. *ImagePhysicalEntityHasTwoDimensionGeometricShapeEntityStatement* represents a direct relationship between an instance of an imaging physical entity and a markup. If you have two markups on an image, you create two statements.

A use case:

A user creates a two-dimensional markup on an image from a set of CT axial series.

Working with AIM:

1. Create an imaging physical entity instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create an imaging physical entity containing left upper lobe, RID1327, RadLex.
4. Create a two-dimensional polyline markup on a mass.
5. Create a *TwoDimensionPolyline* to store information in step 3.
6. Create *ImagingPhysicalEntityHasTwoDimensionGeometricShapeEntityStatement* to link the imaging observation entity (subject) with the *TwoDimensionPolyline* entity (object).

ImagingPhysicalEntityHasTextAnnotationEntityStatement

An image physical entity can have a text annotation.

A use case:

A user creates a text markup to label the imaging physical entity from a set of CT axial series.

Working with AIM:

1. Create an image annotation instance.
2. Create a DICOM image reference instance and its associated image study, image series, and image instances using *DicomImageReferenceEntity*.
3. Create an imaging physical entity containing left upper lobe, RID1327, RadLex.
4. Create a text markup on a mass.
5. Create *TextAnnotationEntity* to store information from step 4.
6. Create *ImagingPhysicalEntityHasTextAnnotationEntityStatement* to link the imaging physical entity (subject) with the text annotation entity (object).

ThreeDimensionGeometricShapeEntityIsComprisedOfThreeDimensionGeometricShapeEntityStatement

Two or more drawings captured as *ThreeDimensionGeometricShapeEntity* instances have a direct relationship to other drawings when these drawings are placed on the same physical entity or thing.

A use case:

A user draws two different markups on the same lesion. This lesion is the same thing that appears on the same image.

Working with AIM:

1. Create an image annotation instance.
2. Create a markup of the first area of a mass.
3. Create *ThreeDimensionGeometricShapeEntity* to store information in step 2.
4. Calculate area from step 3 (optional).
5. Create a markup of the second area of the same mass.
6. Create a *ThreeDimensionGeometricShapeEntity* to store information in step 5.
7. Calculate the area from step 5.
8. Create *ThreeDimensionGeometricShapeEntityIsComprisedOfThreeDimensionGeometricShapeEntityStatement* to link the first markup (subject) to the second markup (object).

ThreeDimensionGeometricShapeEntityExcludesThreeDimensionGeometricShapeEntityStatement

Two or more graphical drawings captured as *ThreeDimensionGeometricShapeEntity* instances have a direct relationship with other drawings when these drawings are placed on the same physical entity or thing.

A use case:

A user draws two different markups on the same lesion. This lesion is the same thing that appears on the same image.

Working with AIM:

1. Create an image annotation instance.
2. Create a markup of the first area of a mass.
3. Create *ThreeDimensionGeometricShapeEntity* to store information from step 2.
4. Calculate the area from step 3 (optional).
5. Create a markup of the second area of the same mass.
6. Create *ThreeDimensionGeometricShapeEntity* to store information in step 5.
7. Calculate the area from step 5.
8. Create *ThreeDimensionGeometricShapeEntityExcludesThreeDimensionGeometricShapeEntityStatement* to link the first markup (subject) to the second markup (object).

TwoDimensionGeometricShapeEntityIsComprisedOfTwoDimensionGeometricShapeEntityStatement

Two or more graphical drawings captured as *TwoDimensionGeometricShapeEntity* instances have a direct relationship with other graphical drawings when these drawings are placed on the same physical entity or thing.

A use case:

A user draws two different markups on the same lesion. This lesion is the same thing that appears on the same image.

Working with AIM:

1. Create an image annotation instance.
2. Create a markup of the first area of a mass.
3. Create *TwoDimensionGeometricShapeEntity* to store information from step 2.
4. Calculate the area from step 3 (optional).
5. Create a markup of the second area of the same mass.
6. Create *TwoDimensionGeometricShapeEntity* to store information in step 5.
7. Calculate the area from step 5.
8. Create *TwoDimensionGeometricShapeEntityIsComprisedOfTwoDimensionGeometricShapeEntityStatement* to link the first markup (subject) to the second markup (object).

TwoDimensionGeometricShapeEntityExcludesTwoDimensionGeometricShapeEntityStatement

Two or more graphical drawings captured as *TwoDimensionGeometricShapeEntity* instances have a direct relationship with other drawings when these drawings are placed on the same physical entity or thing.

A use case:

A user draws two different markups on the same lesion. This lesion is the same thing that appears on the same image.

Working with AIM:

1. Create an image annotation instance.
2. Create a markup of the first area of a mass.
3. Create *TwoDimensionGeometricShapeEntity* to store information from step 2.
4. Calculate the area from step 3 (optional).
5. Create a markup of the second area of the same mass.

6. Create *TwoDimensionGeometricShapeEntity* to store information from step 5.
7. Calculate the area from step 5.
8. Create *TwoDimensionGeometricShapeEntityExcludesTwoDimensionGeometricShapeEntityStatement* to link the first markup (subject) to the second markup (object).

UriImageReferenceEntityHasImagingObservationEntityStatement

A URI image reference can have an imaging observation. It is used in conjunction with *UriImageReferenceHasImagingPhysicalEntityStatement*.

A use case:

A user wants to state that a mass is found on an image.

Working with AIM:

1. Create an image annotation instance.
2. Create *UriImageReferenceEntity* instance with the image location.
3. Create an imaging observation containing mass, RID3874, RadLex.
4. Create *UriImageReferenceEntityHasImagingObservationEntityStatement* to link the *UriImageReferenceEntity* (subject) with imaging observation (object).

UriImageReferenceEntityHasImagingPhysicalEntityStatement

An URI image reference can have an imaging physical entity associated with it.

A use case:

A user wants to label left upper lobe of lung on an image.

Working with AIM:

1. Create an image annotation instance.
2. Create a *UriImageReferenceEntity* instance with the image location.
3. Create an imaging physical entity containing left upper lobe, RID1327, RadLex.
4. Create *UriImageReferenceEntityHasImagingPhysicalEntityStatement* to link the *UriImageReferenceEntity* (subject) with the imaging physical entity (object).

UriImageReferenceEntityHasCalculationEntityStatement

A URI image reference can have a calculation associated with it.

A use case:

A user wants to compute and store a size of the mass from an image.

Working with AIM:

1. Create an image annotation instance.
2. Create a *UriImageReferenceEntity* instance with the image location.
3. Create a markup to measure the size of a mass.
4. Create *CalculationEntity* and store the size of the mass.
5. Create *UriImageReferenceEntityHasCalculationEntityStatement* to link the *UriImageReferenceEntity* (subject) with the calculation entity (object).

Renamed Classes

AnatomicEntity was renamed as ImagingPhysicalEntity

ImagingPhysicalEntity is an entity or object that can be identified on an image. For medical imaging, it may represent anatomical location of an organ or body structure. Terms from controlled vocabulary such as RadLex, SNOMED CT, DCIOM, etc. are used to record the type of entity.

AnatomicEntityCharacteristic was renamed as ImagingPhysicalEntityCharacteristic

An *ImagingPhysicalEntityCharacteristic* is a characteristic of imaging physical entities. It is in contradistinction to *ImagingObservationCharacteristic*. In a medical area, for example, "dilated" might be an imaging physical entity characteristic of the "colon" imaging physical entity.

ImagingObservation was renamed *ImagingObservationEntity*

An *ImagingObservationEntity* is the description of things that are seen in an image. "Mass", "Pleural Effusion", "Foreign Body", and "Artifact", are all examples of imaging observation entity.

Inference was renamed *InferenceEntity*

A conclusion derived by interpreting images and/or other supplemental information related to the images such as medical history, geographic history, etc.

AnnotationRole was renamed *AnnotationRoleEntity*

AnnotationRoleEntity describes the role of referenced annotation.

ImageReference was renamed *ImageReferenceEntity*

ImageReference is an abstract class that references the image which is being annotated.

WebImageReference was renamed *UriImageReferenceEntity*

UriImageReferenceEntity is a source image for the annotation. It can be accessed via Intranet, Internet, local computer and/or file sharing systems.

DicomImageReference was renamed *DicomImageReferenceEntity*

DicomImageReferenceEntity is a source image for the annotation.

Annotation was renamed *AnnotationEntity*

AnnotationEntity captures information that can be described, measured, calculated, and drawn on images either by a human or machine observer.

TextAnnotation was renamed *TextAnnotationEntity*

TextAnnotationEntity represents the text and the markup of text intended to be rendered on the image.

GeometricShape was renamed *GeometricShapeEntity*

GeometricShapeEntity is the shape of a region of interest (ROI).

Segmentation was renamed *DicomSegmentation*

DicomSegmentation is a multi-frame image representing a classification of pixels in one or more referenced images. Segmentations are either binary or fractional. See *DICOM part 3 Segmentation IOD* for more information.

PresentationState was renamed *ReferencedDicomObject*

ReferencedDicomObject is a collection of related DICOM objects created from the image(s), e.g. presentation state, SR document, radiotherapy objects, waveform, encapsulated document, etc.

The old association between *PresentationState* and *DicomImageReferenceEntity* was removed. It was replaced by an association between *ReferencedDicomObject* and *ImageStudy*.

Deleted Classes

Four classes were deleted since AIM statements and new classes cover the concepts represented by the deleted classes.

- *ReferencedGeometricShape* is represented by *ImagingObservationEntityIsIdentifiedByGeometricShapeEntityStatement*.
- *AimStatus* is represented by *AuditTrail*.
- *ReferencedAnnotation* is represented by *AnnotationOfAnnotationHasAnnotationRoleEntityStatement* and *ImageAnnotationHasAnnotationRoleEntityStatement*.
- *ReferencedCalculation* is represented by *CalculationEntityReferencesCalculationEntityStatement*.

New Attributes

New attributes have been added to the following classes.

Inference

- *questionTypeCode* is used to collect coded entry data that describes the question being asked that is related to the *typeCode* attribute.
- *isPresent*, a Boolean value indicating whether or not an inference exists in the observed images.
- *label*, a human readable description of the inference.
- *questionIndex* is used to store an index value that identifies the order of the question in an AIM Template.
- *comment* is a free text about inference.

GeometricShapeEntity

- *label* is a human readable description of the geometric shape.
- *description* is a free text about the geometric shape, not intended for rendering.
- *questionTypeCode* is used to collect coded entry data that describes the question being asked that is related to the *typeCode* attribute.
- *questionIndex* is used to store an index value that identifies the order of the question in an AIM Template.

- *interpolationMethod* is used to create new data points from a discrete set of data points, such as linear, polynomial, spline and piecewise constant interpolation.
- *comment* is free text about a geometric shape entity.

ImageSeries

- *modality* is the equipment used to acquire images of subjects or things, such as human and animal bodies.

CalculationEntity

- *questionTypeCode* is used to collect coded entry data that describes the question being asked that is related to the *typeCode* attribute.
- *imageIdentifierReference* is a number that refers to an existing image identifier from *ImageReference*.
- *questionIndex* is used to store an index value that identifies the order of the question in an AIM Template.

ImageStudy

- *procedureDescription* is information about the procedure being performed on a subject.

Scale

- *type* represents different types of scales that are Nominal, Ordinal or Ratio.

AnnotationRole

- *questionTypeCode* is used to collect coded entry data that describes the question being asked that is related to the *typeCode* attribute.

ImageReference

- *imageIdentifier* is a unique number within an *ImageAnnotation* object that uniquely identifies this object.

ImagingObservation

- *imageIdentifierReference* is a number that refers to an existing image identifier from *ImageReference*.
- *questionIndex* is used to store an index value that identifies the order of the question in an AIM template.

Annotation

- *templateUid* is a UID that references to an AIM template used to capture semantic meaning of pixel data, markup and calculation.

CalculationResult

- *dataType* is coded entry data used to describe or capture a type of parameter. A coded data type can be a primitive programming data type such as integer, double, etc. as well as other data types such as URI.

Equipment

- *deviceSerialNumber* is manufacturer’s serial number of the equipment that produced the sources.

ImagingPhysicalEntity

- *questionIndex* is used to store an index value that identifies the order of the question in an AIM Template.

Quantile

- *bins* is a number representing the dividing ordered data into an equal-sized data subsets. For example, maxValue is 100. It's divided by four. Four is the value for the bins.
- *selectedBin* is an integer value of the selected bin.
- *minValue* is a minimum value of the lowest value in a range.
- *maxValue* is a maximum value of the largest value in a range.

CharacteristicQuantification

- *comment* is free text about characteristic quantification.
- *valueLabel* is a value that associates with the label.
- valueDescription is a human readable description.

ImagingPhysicalEntity

- *comment* is free text about imaging physical entity.

ImagingPhysicalEntityCharacteristic

- *comment* is free text about imaging physical entity characteristic.

Attribute Name Change

The *typeCode* attribute replaces a set of coded term attributes as a single attribute in a class. The replaced attributes include:

- *codeValue*
- *codeMeaning*
- *codingSchemeDesignator*
- *codingSchemeVersion*

The attributes are mapped to a single CD ISO 21090 data type. The affected AIM classes include:

- *AnnotationEntity* (formerly known as *Annotation*)
- *AnnotationRoleEntity* (formerly known as *AnnotationRole*)
- *CalculationEntity* (formerly known as *Calculation*)
- *ImagingPhysicalEntity* (formerly known as *AnatomicEntity*)
- *ImagingPhysicalEntityCharacteristic* (formerly known as *AnatomicEntityCharacteristic*)
- *ImagingObservationEntity* (formerly known as *ImagingObservation*)
- *ImagingObservationEntityCharacteristic* (formerly known as *ImagingObservationCharacteristic*)
- *InferenceEntity* (formerly known as *Inference*)
- *NonQuantifiable*

The *name* attribute in *CharacteristicQuantification* has been changed.