Integrating the Healthcare Enterprise

IHE Radiology White Paper

AI Interoperability in Imaging

Revision 1.1 – Published

Date: October 12, 2021

Authors: Brad Genereaux, Kevin O'Donnell, Brian Bialecki, Karl Diedrich,
Christopher J Roth, Antje Schroeder, Neil Tenenholtz,
Khaled Younis, Harald Zachmann, and the IHE Radiology Community

Email: radiology@ihe.net

Please confirm you have the most recent version of this document. See here for Published versions and here for Public Comment versions.

Foreword

This is a white paper of the IHE Radiology domain.

This white paper is published on October 12, 2021. Comments are invited and can be submitted at Radiology Public Comments.

General information about IHE can be found at IHE.

Information about the IHE Radiology domain can be found at IHE Domains.

Information about the organization of IHE Technical Frameworks and Supplements and the process used to create them can be found at Profiles and IHE Process

The current version of the IHE Radiology Technical Framework can be found at Radiology Technical Framework.

Navigating this Document

Given the length of the white paper, some readers may wish to focus on sections of interest.

Section 2 describes the breadth of potential applications of AI in Imaging, with a particular goal of recognizing applications beyond generating candidate clinical findings from image analysis. The people, systems and constructs that are referenced in the rest of the white paper are also introduced.

Section 3 describes the steps and interoperability needs when developing and using AI for purposes such as those described in Section 2. These "use cases" are organized in roughly chronological order, from the establishment and operation of dataset repositories, to the preparation of datasets and their use in training and testing AI models, to the packaging, deployment, testing, and clinical usage of trained AI models. A "common mechanics" section addresses topics that recur in the context of multiple use cases.

Section 4 is a first cut at modelling the metadata associated with some key entities, such as Datasets, Data Repositories, and AI Models, that appear in the use cases.

CONTENTS

1 Introduction

This document, the IHE Radiology AI Interoperability in Imaging White Paper, describes an organizing framework and roadmap for creating profiles to support the creation, lifecycle, and use of AI datasets, AI Models, and AI Applications.

1.1 Purpose of the AI Interoperability in Imaging White Paper

This white paper is intended to document what "AI in Imaging" encompasses, and provide a comprehensive map of interoperability needs, problems, and challenges that must be addressed to achieve an ecosystem of interoperable products that support all the processes and tasks that make up AI in Imaging.

The paper uses "imaging" as a shorthand intended to cover all forms of medical imaging.

This document will be used by the IHE Planning and Technical Committees to identify logical groups of needs that would constitute functional Profiles. The enumerated needs, problems, and challenges will help ensure the Profiles are properly scoped and do not overlook issues that might only be apparent in the "big picture" or after more careful consideration.

Having this roadmap will make it much easier to select and properly scope profile proposals and may also help to prioritize/sequence the work. Hashing out some basic points of agreement will likely lead to faster progress when the profiles are developed. Vendors and users will benefit from any degree of alignment/harmonization.

The white paper does not describe specific actors or transactions. Such definition will be the job of subsequent Profiles. Generally, the focus here is on determining the key questions and needs so the profile work is well scoped. From that foundation, profile development will undertake to develop consensus-based interoperable answers that meet those needs.

1.2 Intended Audience

The intended audience of the IHE Radiology AI Interoperability in Imaging White Paper is:

- Members of the IHE Planning and Technical Committees
- Members of other standards bodies who create and support the standards necessary for AI (e.g., DICOM®[1], HL7®[2], LOINC®[3], etc.)

Many user communities that are also attempting to understand the scope of AI in Imaging and the various needs, problems, and challenges, should find this white paper useful (and are strongly encouraged to contribute insights), including:

[1] DICOM® is the registered trademark of the National Electrical Manufacturers Association for its standards publications relating to digital communications of medical information.

[2] HL7®, is the registered trademark of Health Level Seven International and the use of this trademark does not constitute an endorsement by HL7

[3] LOINC® is registered United States trademarks of Regenstrief Institute, Inc.

- Consumers of AI products (radiologists, technologists, informaticists, referring/attending physicians)
- Administrators of AI products (PACS admins, IT operations, radiology leadership, health records and quality team)
- Developers of AI products (including analysts, and data scientists)
- Vendors of imaging products (Image Managers, Image Displays, RIS, Imaging Modalities)
- Vendors of AI products and services (AI Orchestrator and AI Performer products)
- Developers of AI-related standards
- Strategic planning members, product managers, R&D engineering leads
- Health policy agencies and experts, national societies

2 Landscape

Artificial intelligence in medical imaging encompasses an exceedingly broad spectrum of topics. This section describes the problems that AI is poised to solve and lays out the applications and personas in greater detail.

AI can be used to perform or support a wide variety of tasks in the realm of medical imaging, including image analysis, broader workflows beyond the images themselves (like ordering and reporting), and broader imaging workflows beyond radiology (like pathology, dentistry, and ophthalmology), and other imaging domains (like pathology).

Creating, training, validating, deploying, monitoring the performance of, and using artificial intelligence components in medical imaging is a large problem space with emerging technologies. Those AI components need to be integrated into a variety of clinical settings which includes hospitals, imaging departments, private medical practices, skilled nursing facilities, clinics, and the patient's home.

2.1 Applications of AI in Imaging

This section identifies and categorizes applications to which AI and Machine Learning are applied in medical imaging. It is not intended to be exhaustive, but is intended to provide a robust list that establishes the broad scope of AI so that:

- The scope of subsequent IHE Profiles can be deliberately considered and decided.
- Use cases, mechanics, and entities in later sections of this white paper can be vetted to confirm their applicability to this range of applications.

Many AI Applications involve performing a task that would otherwise be performed by a human. AI Applications may fully automate the task, automate the task but the result is subject to oversight by a human, or semi-automate the task where the AI Application and the human collaborate on the task or support the task by assessing the results generated by the human.

AI Applications may interact with humans with different degrees of "immediacy".

- AI Applications may work completely in the background, in the sense that a human seldom or never interacts directly with the application or its output.
- AI Applications may perform tasks ahead of time (either scheduled or proactively addressing expected future needs) so the output is available for human use at an appropriate time.
- AI Applications may be directly invoked by a human interaction with output available promptly or at some time later depending on the data collection and processing requirements.
- Finally, some may be “interactive” in the sense of cyclically taking user input, performing inference, and going back to the user for more input.

Users may migrate from one pattern to another (e.g., start with a proactively developed result which the user then starts interacting with by adjusting input information). Each of the preceding patterns may have interoperability implications for Profiles to consider in terms of scheduling and coordinating task performance and delivery of results at appropriate times and places.

AI Applications may be triggered to perform tasks in a variety of ways, including

- by availability of data of a form that the AI Application can process
- by completion of a preceding task (daisy-chain)
- by defined workflow logic that calls for the AI Application to perform a task
- by direct command from a user

Benefits driving the adoption of AI can include one or more of making the task faster, easier, less costly, more accurate, or more consistent/repeatable. AI also brings uniformity to derived physiologic and anatomic measurements across a potentially large number of human performers for a given task. Without it, individuals are left all doing their own thing. AI makes it easier for a group of people to collectively do the right thing easily.

Lastly, it should be noted that much of this white paper applies similarly to algorithms that are not based on neural networks and machine learning. Profile development should consider addressing conventional algorithms in tandem with AI-based algorithms. Using common mechanisms where it makes sense would likely benefit developers and users of both technologies.

2.1.1 Ordering and Scheduling

Ordering and Scheduling refers to the process of selecting and ordering imaging procedures and scheduling resources (scanners, rooms, staff, the patient, other supplies, or devices, etc.) for the execution of those procedures.

Tasks to which AI could be applied include:

- Ranking appropriate procedures to order
  - May be based on the patient history, patient location, historical patterns of the ordering physician, similar patient population outcomes, national guidelines, user free text.
  - See IHE Clinical Decision Support Order Appropriateness Tracking [STD-CDS-OAT]
- Scheduling and prioritizing scanners, rooms, staff, etc. for ordered procedures.
- Participating in chatbot patient or staff customer service interactions.
- Suggesting the right imaging clinic/physical location.
- Predicting and mitigating “no shows”
- Identifying that a patient is due (or overdue) to have a particular imaging study (or other procedure) ordered.

    - May be based on guidelines and patient specific details. E.g.,
        - Due for a screening mammogram based on age, date of last exam, and result of last mammogram.
        - Due for a screening lung CT based on patient risk factors.
- Prompting staff to request follow-up exams recommended in reports. [STD-SDC]
- Proposing appropriate values for procedure codes, body part codes, billing codes, etc. and cross-matching with appropriate ontologies for fully populating metadata

2.1.2 Procedure Guidance

Procedure Guidance refers to interventional imaging procedures and to the use of imaging for the purpose of guiding a non-imaging procedure. Imaging for diagnostic purposes is covered in later sections.

Tasks to which AI could be applied include:

- Detecting that a device such as a breathing tube has been positioned (or remains) in the proper location and informing clinicians.
- Providing guidance to the direction of a clinical approach during a minimally invasive interventional or open surgical procedure.
- Directing surgeons or interventionalists toward the optimal percutaneous location to begin a procedure, or to identify tissue remaining to be treated.
- Positioning a biopsy needle properly within the target tissue.

2.1.3 Image Acquisition

Image Acquisition refers to the acquisition of images and associated data by an imaging modality. It includes interactions with the patient and associated devices by the staff acquiring images and the imaging modality as part of the acquisition process.

Tasks to which AI could be applied include:

- Setting acquisition parameters appropriate to the patient and diagnostic task
- Optimizing radiation exposure and dose (in balance with image quality)
- Positioning the patient appropriately for the imaging procedure
    - E.g., ensuring the primary anatomic structure for the requested procedure is properly in view in the acquired images.
- Detecting image acquisition triggers like cardiac waveforms or lung inspiration level.
- Guiding the acquisition process to get adequate or optimal images. E.g.,
    - a suitable ultrasound projection for assessing cardiac function.

  - providing instructions to the clinician, or interacting with the patient
  - at-home devices or standalone devices in public settings (a melanoma mole scanner in a pharmacy).
- Detecting image metadata details that are unlabeled or mislabeled.
  - E.g., image laterality, body part imaged, etc.
  - This task is intended for labeling that would be done at acquisition time. Other tasks related to segmentation and labelling can be found under Image Analysis.
- Detecting significant differences in patient demographics (ensuring that the correct patient is being imaged)
- Detecting and/or compensating for patient motion or warning the degree of motion is beyond a quality threshold so the technologist can communicate with the patient for compliance and/or trigger re-imaging.
- Notifying the tech if required views or series were not acquired (e.g., a missing view for an X-ray or ultrasound procedure) so those images could be acquired before the patient leaves, thus avoiding having to call the patient back later.

2.1.4 Image Generation

Image generation refers to the reconstruction of images from acquisition data (e.g., generating CT images from sinograms, or generating digital subtraction angiographic images from planar XA images) and related processing to improve the images, such as removing noise or increasing contrast.

Reconstruction and processing are commonly embedded in image acquisition modalities although it can be performed on separate workstations.

Tasks to which AI could be applied include:

- Generating images from “less complete” data than would otherwise be used, e.g., partial CT rotation, or shorter MR acquisition.
- Generating images from an incomplete acquisition, e.g., discontinued acquisition due to medical event.
- Reducing noise in images
- Increasing contrast or conspicuity of important features
- Decreasing or removing artifacts from patient motion, metal objects, air pockets, etc.
- Suppressing or removing materials or anatomical structures (e.g., bone removal on x-rays, vessel suppression on CT chest exams)
- Increasing the resolution of images or clarity of small details
- Assessing/scoring the quality of the images (perhaps relative to the reason for imaging)

- Excluding or sequestering images that are redundant or non-diagnostic to reduce downstream clutter, processing load, and reader fatigue.
- Prioritizing or triaging reconstruction and post-processing worklists

2.1.5 Image Analysis

Image Analysis refers to processing the content of images and associated data to generate new information, such as measurements, observations, and content for the image interpreter. Image Analysis is commonly performed separate from image capture and thus is often performed on a system other than the imaging modality.

The "**target**" of the image analysis might be a device (like a pacemaker, a stent, or central line), a part of a device (like the tip of a needle), a condition (like pneumothorax, or osteopenia), a pathological structure (like a tumor or aneurysm), or an anatomical structure (like a kidney, aortic valve, colon, or blood vessel)

Tasks to which AI could be applied include:

- Detecting the presence of a target
  - The purpose of detection tasks may be to support the reason for the study, to detect less obvious findings or those unrelated to the imaging request, or to detect findings that may influence events prior to interpretation of the study (see Section 2.1.6 Image Interpretation on prioritizing reading worklists).
  - Detection may also serve a safety check function, catching undesirable but possible situations like medical apparatus left in a patient.
  - A detection result may be true/false, or may be a probability
- Ruling out the presence of a target
  - Like the detection task, but with an emphasis on triaging and excluding medical emergencies at the point of care.
    - The focus on things like the rate of false positives and true negatives may differ between detection tasks and rule-out tasks.
- Locating a target
  - Locations may be in the form of a point, an arrow, a rough outline, etc.
  - Locations may include the orientation as well as the position.
  - Locations may be relative to another feature (like the position of an endotracheal tube relative to the carina)
- Identifying the same target in multiple images
  - The target may have been imaged using the same modality at different times (e.g., longitudinal correlation of tumors), different modalities at roughly the same time (e.g., a target in coincident PET and CT images), or in multiple views in the same

study (e.g., a lesion appearing in both the LCC view and the LMLO view of a mammography study)

- Segmenting a target
  - Segmentations may be in the form of a bounding contour, probability map, etc.
- Measuring a quantitative characteristic of targets
  - Anatomic measurements could include length, area, volume, gap, angle, RECIST measurement, etc.
  - Physiologic measurements could include ejection fraction, tissue blood perfusion, lesion enhancement characteristics, etc.
- Determining the appropriate "score/grade/code" from a grading or classification system
  - Examples include characterizing a tumor, grading stenosis, grading hip joint osteoarthritis, etc.
- Characterizing a qualitative characteristic of a target
  - Characterizations may include shape, spiculation, heterogeneity, or density.
  - E.g., breast density reporting.
- Assessing changes between a current and prior study, across same or different modalities.
  - AI systems will need to distinguish between clinical change and changes due to acquisition parameters, normal aging, or intervention (radiation therapy).
- Spatially registering studies with current / prior study across same or different modalities.
  - Registration may be rigid or elastic (e.g., to account for malleable organs (such as the breast), MRI field inhomogeneity or changes to the shape of the target over time
- Assessing whether the image characteristics are adequate/appropriate for processing by another AI.
  - Assessment might include checking the view, the image quality, or other features. For example, a prescreen AI could confirm that a breast implant is not present prior to the image being processed by a breast AI that is not trained to process images with an implant in the view.
  - The assessment might result in alternate subsequent AI algorithms being used.
- Identifying/selecting which images amongst a set are the ones to be processed by another algorithm or reviewed by a radiologist.
  - E.g., selecting the images acquired with the appropriate sequence (such as T1-weighted) or of the appropriate anatomy in a large MR Study. This can be helpful when the corresponding tags or Series Descriptions are not consistently populated.

- Autogenerating teaching files.
    - E.g., "the typical COVID x-ray looks like this."

2.1.6 Image Interpretation

Image interpretation refers to supporting the radiologist in the review of the content of an imaging study, and associated data, to isolate relevant findings, impressions, and follow-up management recommendations, and capture those in a diagnostic report.

Tasks to which AI could be applied include:

- Prioritizing or triaging images to be viewed and interpreted, such as on reading worklists.
- Composing a summary of the medical history of the patient that includes details most relevant to the requesting physician and their current task.
- Selecting prior images from the patient's record that are most relevant to the reading of a current study.
- Selecting and optimizing the display protocols for display of the imaging dataset under review
- Extracting critical results from reports to drive notification & additional findings for follow-up workflows.
- Flagging lesions for review during image interpretation, or for quality assurance after human reading to identify any relevant additional findings.
- Detecting discrepancies between the dictated report body and the AI-detected observations from images (e.g., potential auto-detected findings), or known information about the patient (e.g., left-right discrepancies between report and procedure performed or a report that mentions ovaries for a male patient).
- Suggesting the diagnosis or relevant differential diagnosis
- Predicting the progression of a condition or the likely outcome of an intervention
- Distinguishing "meaningful" findings (whether part of the intended reason for exam or not) from less meaningful/significant
    - E.g., a benign mass unrelated to the reason for exam, otherwise known as an "incidentaloma."
- Recommending appropriate clinical follow-up steps
- Recommending appropriate patient staging for disease progression
- Identifying or preparing an effective presentation ("best" slice, useful cut planes, optimized windowing) to understand a given finding.

- Rendering anatomic structures
    - E.g., blood vessels net, bones, kidney, tumor, etc.
- Creating draft sentences or sections of reports based on imaging inputs.
    - May provide material for findings, impressions, diagnosis, and follow-up recommendations.
- Incorporating Image Analysis results (see Section 2.1.5 Image Analysis) into the report and/or medical record in coded and/or narrative form.
- Transcribing and redistributing natural language dictation into structured terms and formatted reports using natural language processing (NLP)
- Determining appropriate people to alert of specific results.
- Identifying appropriate billing codes for performed procedures.
- Recognizing details required for a given billing code that are missing from the report
    - E.g., expected quantitative values for a certain procedure
- Recognizing components of the study that are missing from the report
    - E.g., a contrast series exists but contrast not mentioned yet in the report

2.1.7 Patient Management and Treatment Planning

Patient Management refers to activities leading to care decisions for individual patients.

See also “Recommending appropriate clinical follow-up steps” in 2.1.6 Image Interpretation. These use cases might also relate to 2.1.1 Ordering and Scheduling.

Treatment plans might include surgery, interventional radiology procedures, radiation therapy, chemotherapy, referrals to other specialists, and other patient management.

Tasks to which AI could be applied include:

- Proposing care or treatment plans to a human, concordant with relevant guidelines.
    - A care plan may consist of appropriate assessments and diagnostic tests being conducted at specific intervals, appropriate interventions driven from those assessments, and appropriate follow-up steps.
- Assessing ongoing conformance of a human generated treatment plan with relevant guidelines, and whether completed treatment was in conformance with relevant guidelines.
- Predict patient outcome(s) for a proposed treatment plan
- Performing patient risk stratification
- Alerting personnel to (potentially) critical patient conditions such as pneumothorax, brain bleed or stroke.

2.1.8 Population Health

Population Health refers to activities related to the assessment of an aggregate patient population. Tasks to which AI could be applied include:

- Selecting relevant patients and data from their records (filtering for a patient cohort)
    - E.g., for clinical trials, or for training AI Models
- Identifying trends or patterns for certain pathologies and co-morbidities
    - E.g., “Tuberculosis is up sharply in the last two weeks", or "The prevalence of thyroid cancer in smokers in the north end of the city has been increasingly trending above the state levels over the last 5 years."
- Identifying trends or patterns for certain treatments and outcomes
    - E.g., “Patients with COVID respond better when they get drug X.”
- Determining efficacy and adverse events for drugs or other treatments
- Monitoring ongoing data to identify deviations from trends of data (e.g., outbreaks)
- De-identifying imaging content for inclusion in aggregate datasets
    - E.g., using NLP to identify patient identifying details in report body prose or private DICOM tags, comment fields, image overlays, etc.

2.1.9 Departmental Analysis and Operations

Departmental Analysis refers to activities related to the assessment of the operational performance of a department such as radiology.

Tasks to which AI has been, or could be, applied include:

- Computing and tracking clinical performance indicators against departmental standards and accepted benchmarks
    - E.g., re-admission rates, lesion identification and peer review
- Identifying trends or patterns in throughput or resource utilization
    - E.g., scanner duty cycle, report turnaround time (TAT), exam length, etc.
- Assessing the utilization and effectiveness of clinical tools such as AI Applications
- Assessing the performance of individual imaging equipment to predict possible near-term machine breakdown and trigger proactive preventative maintenance.
- Assessing the rate of “correctness” of ordering physician and departmental Reasons for Exam
    - E.g., the fraction of emergency department pulmonary embolus protocol CT requests where imaging demonstrates pulmonary embolus.

- Supporting departmental analysis tasks conducted in a real-time, daily, quarterly, or annual basis.
- Analyzing information access patterns to protect systems from cyber-attack or unapproved data sharing.
- Monitoring and ordering device supplies and service
  - E.g., chemical reagents, medical supplies

2.2 Personas, Systems, and Entities

This section introduces people (“personas”), products (“systems”), and digital entities (“entities”) that will appear in the Use Cases section that follows.

2.2.1 Personas

Personas describe the role, responsibility, or primary interest of a person in the context of a use case and provide readers with insights into motivations and challenges that may be relevant to addressing the use case. Identifying personas up-front enables use case developers a critical look at all layers of the interoperability boundary.

The personas listed below are not intended to imply a specific job title. In practice, a wide variety of people might be involved in the use cases in Section 3, and those people might take on various roles in different use cases. These different personas might also work together in groups to collectively complete tasks identified in use cases.

Figure 2.2.1-1 maps the following personas to the use cases in Section 3 in which they appear.

Clinical

Radiologist: Provides differential diagnoses on imaging studies and perform image-guided therapeutic procedures. Increasingly asked to integrate AI into image interpretation tasks.

Radiology Leadership: Governs the overall operation of a radiology department, including assessing tools to aid the radiologist team. Responsible for authorizing capital and operational funds for AI software and hardware.

Clinician: Provides direct care to a patient. May interact directly or indirectly with imaging AI applications or the results they produce. They may also use enterprise imaging-related services.

Health Information Management Team: Manages medical imaging record quality by ensuring that records are complete and free of discrepancies, using software tools potentially augmented by AI to aid them in this work.

AI Lifecycle

Data Scientist: Responsible for conducting AI research and supporting the design the scope and parameters of imaging AI initiatives to craft models from transformed imaging data. Responsible for assessing models for inclusion into clinical practice. May be clinically or technically trained.

Principal Investigators are a type of data scientist responsible for leading research projects and determines research project goals and plans to attain them.

Biostatistician: Supports the data scientist by evaluating that the breadth and depth of datasets statistically support their intended use, e.g., to establish a model's claim of efficacy and appropriateness.

Annotator: Provides data record annotations such as ground truth (I.e., correct results for tasks) to be used for training, testing, and validating AI Models.

Oversight Committee: A group of clinical users, AI lifecycle users, and hospital operations users that oversees safety, vets new products, and prioritizes projects.

Hospital Operations

ML Ops Analyst: Monitors AI Application operations and clinical and technical performance over time. Monitors for updates that may impact AI Applications and participates in go-live activities (from development environments to pre-production environments to production environments). Monitors for operational QA and tests models running in production.

Repository Administrator: Maintains the collection of all imaging data and related healthcare metadata and supports data scientists in supplying study cohorts with data.

Repository Contributor: Provides imaging data, annotations, related healthcare metadata, and expertise in the population of repositories.

IT Operations: Maintains and administrates the systems that run the radiology department, including test, development, pre-production, and production environments, as well as working with data, workflows, and assessing tools that get integrated into workflows. This could include, for example, a PACS Administrator or a Vice-Chair of Informatics.

Information Security and Privacy Officer: Responsible for data security operations and architecture, preventing data loss and fraud, and identity and access management.

Product Vendors

Product Manager: Determines the features necessary for medical software applications.

Vendor Validator: Takes software developed by engineers and requirements from product managers and determines whether it is performing as expected.

Product Support Team: Supports deployments of systems that run the radiology department, including test, development, pre-production, and production environments, and works with their customers (e.g., their customers' informaticists and data scientists) on deploying software for evaluation.

Governance

Health Policy Agency: Oversees how healthcare is delivered in a region and may provide guidance on how to incorporate imaging and AI Applications and provide oversight in how it is used.

Payor: Responsible for renumerating the hospital for the costs incurred for a radiology exam and collecting premiums from patients that they serve.

System Regulator: Provides approval for market claims when selling a product in a particular jurisdiction, by assessing claims and supporting data.

Table 2.2.1-1: Personas in Use Cases

	3.1 Repository Use Cases	3.2 Dataset Assembly Use Cases	3.3 Model Creation Use Cases	3.4 Application Distribution Use Cases	3.5 Functional Validation Use Cases	3.6 Clinical Usage Use Cases	3.7 Feedback Use Cases
Clinical							
Radiologist						*	
Radiology Leadership			*	*	*		
Clinician					*	*	*
Health Information Management Team						*	
AI Lifecycle							
Data Scientist	*	*	*	*			*
Biostatistician			*		*		
Annotator		*					
Oversight Committee			*		*		
Hospital Operations							
ML Ops Analyst				*	*	*	*
Repository Administrator	*						
Repository Contributor	*						
IT Operations				*		*	
Information Security and Privacy Officer				*	*		
Product Vendor							
Product Manager	*	*	*		*		*
Vendor Validator					*		
Product Support Team				*	*	*	
Governance							
Health Policy Agency						*	
Payor					*		*
System Regulator					*		

2.2.2 Systems

Systems represent significant hardware or software that appear in use cases. Recognizing systems helps to identify interoperability boundaries.

This list is not all-inclusive and does not reflect all possible configurations or functionality. In practice, for example, the role of a PACS might be fulfilled by a VNA, a viewer and workflow software integrated together.

PACS: Picture Archive and Communication System. Responsible for viewing, storing, and workflow of medical imaging. System may be converged or deconstructed.

EMR: Electronic Medical Record. Responsible for discrete health data, such as patient demographics, allergies, diseases, and reports. Strictly speaking, an Electronic Health Record (EHR) contains more information than an EMR and can work across health organizations; however, this white paper uses the term EMR generically and does not distinguish between the two.

VNA: Vendor Neutral Archive. Responsible for archiving imaging studies and related data.

AI Model Zoo: Responsible for storing and distributing AI Applications.

Image Display: Responsible for visualizing medical images and results from AI processing.

AI Application Host: A dedicated system that runs specific AI Models or AI Applications.

Imaging Modality: Responsible for the acquisition and generation of patient images.

Section 2.2.3 also contains a few entities that might be considered Systems.

2.2.3 Entities

Common entities that appear in use cases are described in Section 4 Entities, which also captures initial ideas about what information would go into a digital encoding of those entities.

The definitions from Section 4 are replicated here to clarify the intended use of the entities in the use cases.

Data Element: An individual piece of data. The granularity may vary. E.g., an individual image (DICOM or PNG), the patient age, a reason for admission, an annotation (RTSTRUCT or Condition Present), a medication, or a lab value.

Data Record: A set of related data elements. The relationship is often that they are all associated with the same patient encounter, but this can vary. E.g., an x-ray, a lab result, and a reason for admission, all associated with a given patient encounter.

Dataset: A set of related data records. Typically, all the data records in a dataset contain roughly the same data elements. Datasets containing data records with patient data will typically span many patients.

Data Repository: An infrastructure that hosts one or more datasets for discovery and retrieval.

Transform: A process that produces one or more output data elements from one or more input data elements, typically by transcoding or resampling.

AI Model: A neural network architecture and a set of weights that has been trained to produce appropriate outputs when supplied certain types of input.

Note: Given the current focus on deep learning technologies, this white paper will refer frequently to AI Model; however, statistical or other machine learning techniques are often applicable in most places where AI Model is referenced in this document.

AI Application: A package of components (including algorithms, necessary data transport interfaces, and input / output transforms) to be executed in a target environment to perform AI tasks. The algorithm(s) may be based on deep learning, conventional machine learning, or other techniques.

Service: A running instantiation of an AI Application.

Feedback: An assessment by a human or system of the output produced by an AI Model from a given data record.

3 Use Cases

This section describes activities (use cases) involved in the development and use of AI in Imaging for purposes such as those described in Section 2. The activities described here span from research and development, through deployment to after-market monitoring and maintenance.

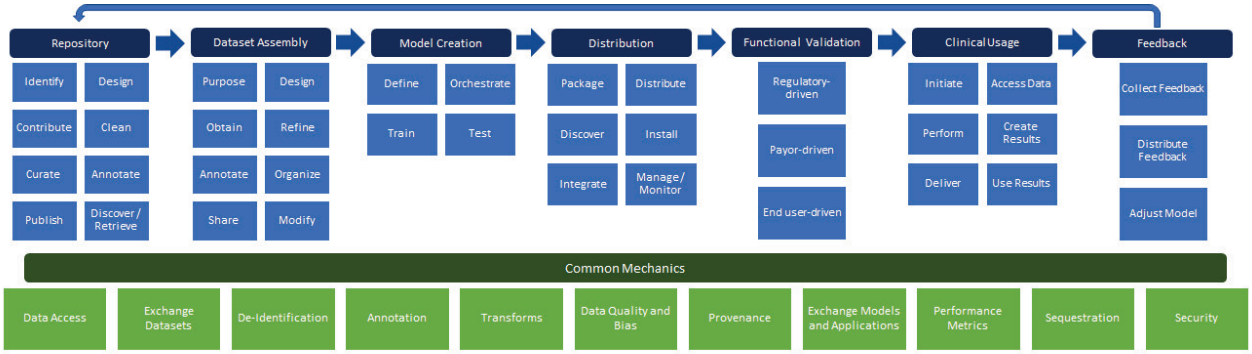


Figure 3-1: Map of AI Use Cases

Each Use Case describes a particular activity (such as contributing data to a repository, annotating data, or creating cohorts of data from existing repository data records) and notes important questions, topics, and considerations that might need to be addressed in a Profile which includes that Use Case.

The Use Cases are grouped into significant stages, such as Repository Use Cases that covers all use cases related to the preparation and management of datasets.

The Use Case Groups, and the Use Cases within them, are presented as a sequence; however, as with the Waterfall Model [BIB3-1], significant feedback and iteration is expected, both within Use Case Groups and between Use Case Groups.

Many use cases involve taking data elements, data records, and/or datasets (see Personas and Systems: Entities), using them, and often creating new or revised data elements, data records, or datasets. To distinguish between these entities of the same type but different source or purpose, the descriptions will often prefix a distinguishing adjective that describes the purpose or origin of the entity, e.g., a *contributed* data record, a *repository* dataset, a *validation* dataset, or a *training* dataset.

Some examples of how these entities are manipulated in the use cases include:

- Populating a repository may involve taking a contributed data record, then normalizing and validating several of the data elements, removing or de-identifying several other data elements, and adding the resulting data record to the repository dataset.

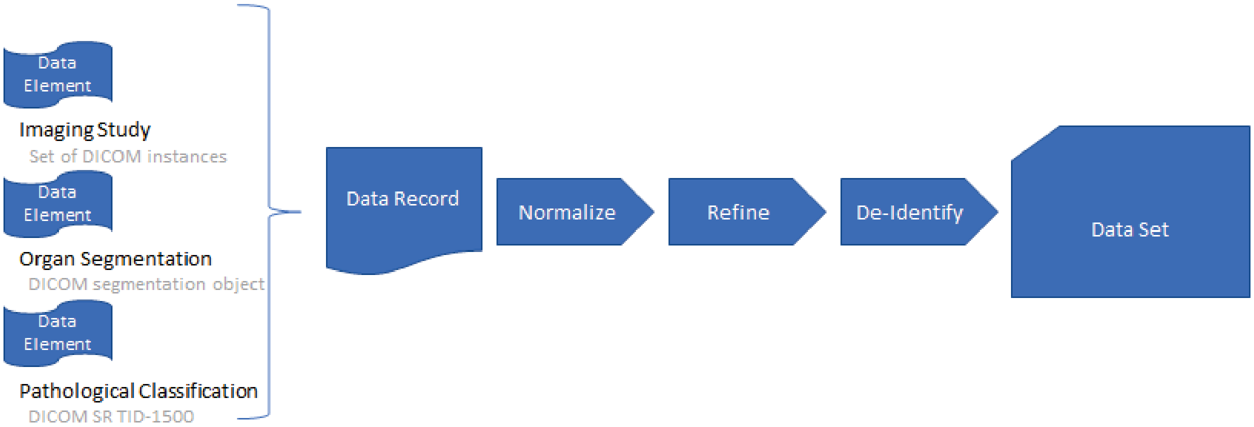


Figure 3-2: Aggregating Data Elements into Data Records and Datasets

- Creating a training **dataset** might involve taking several repository **datasets**, selecting certain **data records** based on inclusion/exclusion criteria, dropping several **data elements** as irrelevant to the planned model purpose, and storing the resulting training **data records** as a single training **dataset**.
- Performing testing might involve taking a **data record** from a test **dataset**, setting aside the **data elements** that contain the result, presenting the rest of the **data elements** to an AI Model, and then comparing the output of the AI Model to the result **data elements**.
- An annotation task might involve taking a **data record** and adding an annotation **data element** to that **data record**. This is often done for all the **data records** in a **dataset**. This is commonly done when preparing training **data records**, validation **data records**, and test **data records**.

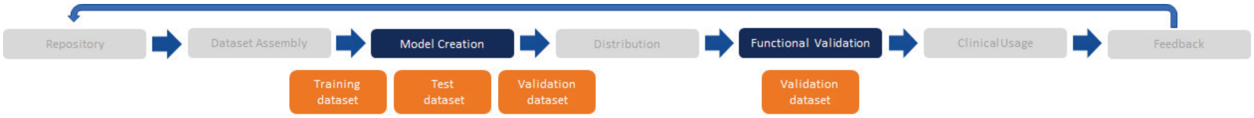


Figure 3-3: Training, test, and validation datasets

- An inference task involves presenting one prepared inference **data record** as an input to an AI model which produces one (or more) **data elements** as an output/result.

The following scenarios are intended to provide a sense of the overall process. They do not delve into all relevant details.

Scenario 1:

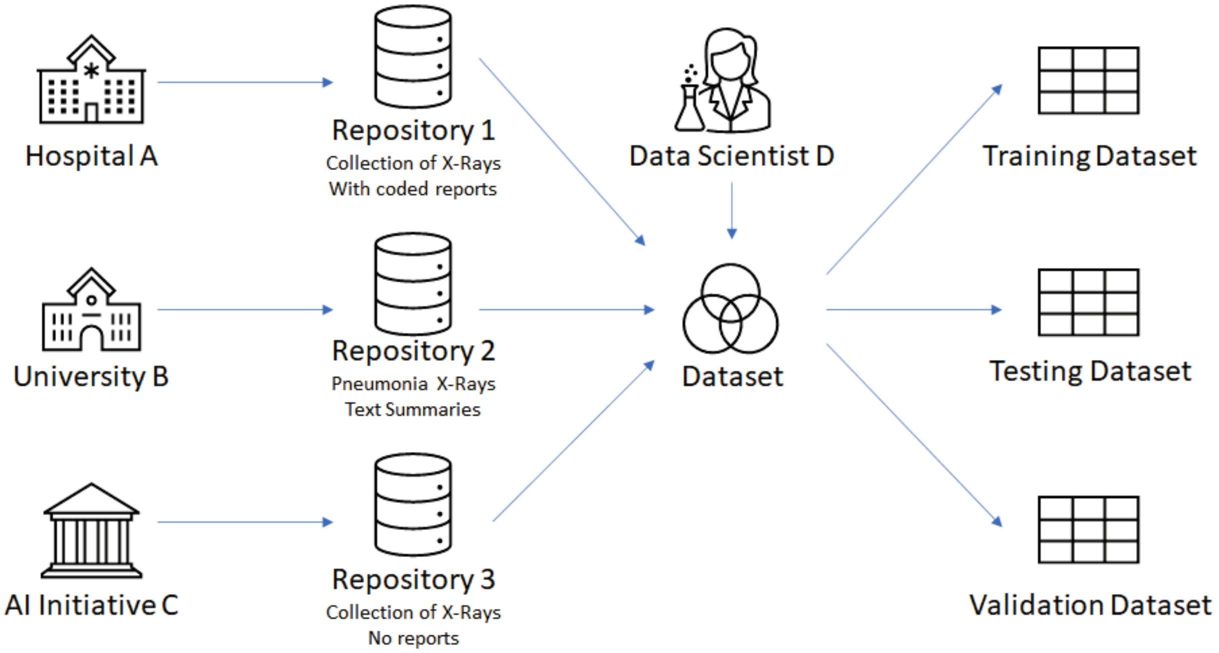


Figure 3-4: Example Scenario: Assembling Datasets from Repositories

Hospital A has created Repository Dataset 1 based on their collection of chest x-rays. Their clinical practice is to create fully coded structured reports. The Repository Administrator has arranged for each data record to contain the chest radiograph image, data elements for each of 6 key demographic details (sex, approximate age, etc.) and each individual finding, impression and recommendation broken out from the coded report as separate data elements. Each record has been de-identified to obscure the patient identity and the dataset has been published in the public repository operated by the research arm of the hospital.

University Research Group B has created a similar Repository Dataset 2 which was intentionally rich in positive pneumonia cases, however the report content is included as paragraph formatted text in a single data element.

AI Initiative C has created a Repository Dataset 3, with chest radiographs but no reports.

Data Scientist D plans to train an AI model to detect the presence of pneumonia in chest radiograph images. In designing the primary dataset, Dr. D requires each data record to contain input data elements for the image, the patient sex, the approximate patient age, and an output data element for the presence or absence of pneumonia.

Dr. D discovers Datasets 1, 2, & 3 and obtains all the data records. For all the data records in Dataset 1, the required data elements are retained, and the rest are discarded. For the data records in Dataset 2, the required input data elements are retained, and the text report is passed through a Natural Language Processor to extract an appropriate pneumonia value for the output data element. For the data records in Dataset 3, the image headers are automatically parsed to populate the sex and approximate age input data elements, and a radiologist is recruited to perform an annotation task to populate the output data element for each record.

Realizing that the make and model of x-ray device and several key imaging parameters might be important, Dr. D runs all the records in the primary dataset through an automatic parser to extract the make, model, and several imaging parameters from the image header into additional input data elements.

Dr. D organizes the data records from the primary dataset into secondary datasets for training, validation, and testing, maintaining a balance of records derived from each of the 3 repository datasets and balancing demographics, rates of presence of pneumonia, and imaging device/protocol variants.

Scenario 2:

Data Scientist E plans to train an AI model to propose report recommendations for chest radiographs at the end of interpretation based on the findings and impressions. Ms. E designs the primary dataset to have data records with input data elements containing findings, impressions, and a broad set of patient demographics, and output data elements containing recommendations.

The data records from Repository Dataset 1 can be used with little refinement, discarding the images, assigning the findings and impressions encoded in the reports to be input data elements, and the report recommendations as output data elements. Like Scenario 1, Ms. E uses a Natural Language Processor to extract input and output data elements from the reports in Repository Dataset 2. As a refinement step, Ms. E has a graduate student compare the extracted data elements to the Dataset 2 narrative text and fix any observed errors.

Since, aside from a few demographics in the image headers, Dataset 3 lacks most of the key inputs and outputs, and resources are not available to do full reporting on all those images, Dataset 3 is not incorporated at this time.

Ms. E is, however, able to discover a repository with a dataset of radiology reports (but no images). The data records from this Repository Dataset 4 are unencoded text and can be processed and incorporated into Ms. E's primary dataset in a similar fashion to the data records from Repository Dataset 2.

Organizing the primary data records into training, validation, and test datasets is like Scenario 1.

3.1 Repository Use Cases

Repository Use Cases encompass the creation and management of repository datasets.

Typically, repository datasets are established to serve as the basis for datasets used to train, test, and validate AI Models.

Notes: 1. It is important to recognize that the creation and management of repositories that include patient data involves significant policy questions and ethical considerations that go beyond the interoperability related topics which are the focus of this document.

2. Use cases later in this document involve storing and distributing models or applications. To avoid confusion, those use cases will avoid using the term "repository", and instead refer to model zoos, or application libraries.

3.1.1 Identify the Repository Purpose

A Repository Administrator establishes the purpose(s) a repository is intended to serve and the corresponding scope of data it will contain.

While some repositories may be created with the development of a specific model in mind (e.g., a repository of chest x-rays for training algorithms to detect the presence of lung cancer), the purpose of most repositories is model-independent (e.g., a repository of imaging, labs, reports, and clinical notes for patients admitted with chest pain) or at least applicable to a broad variety of questions.

The purpose details will influence decisions in many of the subsequent use cases in this section.

Considerations that may have technical interoperability aspects include:

- Encoding the intended purpose(s)
    - Need to facilitate the Discover Repository Content use case
    - Note that Data Assembly use cases may well go beyond the original intended purpose of the repository and that determining suitability is the job of the model Data Scientist not the Repository Administrator).
    - Consider codes identifying the specific model tasks the data records are intended for, or in the case of Model-independent repositories, codes identifying the clinical conditions or imaging procedures in the repository scope
- Intended Audience
    - "Public" access, or "In-house" by users within the institution, or something in between.
    - This choice can influence later choices, including access control, de-identification, data formats, and consent management.
    - In-house repositories can enable local testing of newly-installed AI models to confirm their performance on local patient mixes and image characteristics before the model is put into clinical use. The nature of the various models might not be known when the repository is established. "Baseline performance data" may be needed to support outcome improvement comparisons during such testing

Profile writers may consider some of these questions for further discussion:

- How do we account for the repository purpose shifting over time?
- If you have two different repository purposes (being met by one infrastructure), is that one repository or two?
- How to locate and engage data sources for whom the intended purpose is compelling?

3.1.2 Design Repository

A Repository Administrator designs the data structures, rules, policies, and necessary infrastructure to host the datasets.

Creating a repository involves setting up a technical infrastructure and designing the databases and schemas that will store the dataset(s). Many of these choices will be driven by the intended purpose(s) and scope of the repository and by practical constraints of the infrastructure.

In some cases, an existing infrastructure (e.g., the partial content of a VNA) is being exposed as a repository. In that case, this step is partly about documenting what exists, since there may be limits on how the existing data records can be modified.

The other use cases in this group also include considerations that will affect the repository design. Rather than reiterate those here, the reader is encouraged to review the rest of the Repository Use Cases with this in mind.

Considerations that may have technical interoperability aspects include:

- Defining the specific data elements expected or required in each data record.
- Inclusion and exclusion criteria for data records
  - E.g., specific populations, conditions, modalities
  - Being pre-de-identified may be criteria for inclusion or exclusion.
  - These criteria are sometimes referred to as “eligibility criteria.”
  - The Design Dataset use case also addresses inclusion/exclusion criteria; however, Repository criteria often differ due to having purposes which may be broader or narrower.
- Encoding format(s) of each data element
- Standardized code sets or common terminologies to be used.
- Whether data element storage is centralized or distributed
  - Centralized storage makes it easier to normalize access protocols and mechanisms.
  - Distributed storage allows data to be left in their source systems.
- Whether data is separate from production clinical systems
  - Using separated systems requires more storage, and may necessitate extra annotations to incorporate other clinical and research information.
  - Using production clinical systems may risk impacting performance in a way that affects patient care.
- The degree to which data is de-identified.

    - A privacy officer may ascertain that the repository needs to be de-identified, which may influence how the dataset will be structured (e.g., how can linked datasets and related records be captured).
    - Certain data purposes may require certain potentially identifying details to be retained (e.g., approximate dates of datasets, anonymous patient identification to link data from the same patient)
- Data provenance mechanisms
    - See Common Mechanics: Provenance
- Providing information to, and performing, an Institutional Review Board (IRB) review
- Access control
- Patient consent
    - Whether consent prevents use of de-identified data varies between jurisdictions.

Profile writers may consider some of these questions for further discussion:

- Is sharing a common design (data elements, inclusion criteria, etc.) across different repositories worthwhile to facilitate federating their datasets?
    - Broad/federated datasets are an important method to diversify the dataset and avoid potential sources of bias. See Common Mechanics: Data Quality and Bias.
- How, and to what degree, should the Repository Design incorporate characteristics driven by the FAIR Principles?
    - Some, such as unique IDs for data are already incorporated in the Provenance mechanics etc.
    - The FAIR Principles (Findability, Accessibility, Interoperability, and Reusability) are intended to facilitate the effective management and sharing of datasets. [BIB3.1.2-1]
- Should repository design be generalized to be another example of data assembly?

3.1.3 Contribute Data

A Repository Contributor contributes data records and/or data elements to a repository.

The first “contributor” may be the Repository Administrator harvesting an initial set of data records to which they have obtained access as part of planning for the repository in the first place.

Incorporating the first data records into the repository dataset often serves as an initial test of the repository design and design assumptions so some iteration with the Design Repository use case may be expected.

Repository Contributors may be internal or external to the organization hosting the repository. The same considerations typically need to be addressed in either case, but the onus for addressing them may shift. t

Considerations that may have technical interoperability aspects include:

- Obtaining and tracking the consent of patients who are the subject of a contributed data record.
    - Some regulations (such as GDPR) allow for unconsented use of data under specific “common good” conditions [BIB3.8.3-3].
- Confirming compliance with any IRB approval conditions
- Reviewing any licensing or usage constraints on the contributed data records
    - E.g., identified data records may be restricted from being re-released publicly.
- Tracking the provenance of each data element
    - E.g., who contributed it, how it was created, when it was created, if was clinically reviewed and vetted, whether it was derived, etc.
- Constraining who is permitted to contribute
    - The Repository Contributor may be outside the Repository Administrator organization, and may be an unknown entity, subject to security and inclusion/exclusion concerns.
    - Contributed data records may be unsolicited and/or of unknown provenance.
- See Common Mechanics: Data Access for mechanisms to obtain data element values.
    - (For distributed repositories) Mechanisms to convey references to data, such as URIs, and associated metadata to support indexing.
- Contributing data using push (source-driven) mechanisms and/or pull (repository-driven) mechanisms.

3.1.4 Clean Data Record

A Repository Administrator processes contributed data records and data elements before they are formally incorporated into repository dataset(s).

Considerations that may have technical interoperability aspects include:

- Cleansing and normalizing the data record
    - Fixing up problematic DICOM tags
        - Handling private DICOM tags (either by transforming them into standard attributes if available or at least converging different private tag representations from different vendors into normalized form)

    - Vendors may document private tags in the product DICOM Conformance Statement, or in DICOM PS3.15 Table E.3.10-1, or in a Private Data Element Definition Sequence (0008,0310) in the header of the DICOM instance where the private tags are used.
  - Scaling the resolution into a standard size used for all studies
    - See Entities: Transform
  - Normalizing the color sets and pixel values.
  - Normalizing the uniformity of images in terms of degree of compression and single / multi-frame SOP class representation
    - Conversely, training an AI Model on variety may make it more robust in the wild
  - Coercing values into standard code sets and common terminologies.
    - E.g., RadLex, RadElement, LOINC, SNOMED CT®[4]
  - Normalizing DICOM Study Descriptions, Series Descriptions, Anatomy, Procedure Codes, etc.
  - Confirming required/minimum elements are present (and handling exceptions by fixing, flagging, or dropping)
  - Confirming inclusion and exclusion criteria have been met (and handling exceptions)
  - Organizing clinical data into encounter-oriented records
    - Clinical data may need to be presented per data element and not as a complete dataset depending on use case. E.g., presented as SAS extracts, R, BIDS, FHIR®[5] bundles, or bulk spreadsheets.
  - De-identifying data as needed depending on the design and policies of the repository
    - See Common Mechanics: De-identification
      - How much de-identification to do at each stage in the data pipeline is a potentially complex issue that is discussed in the Common Mechanics section.
- Augmenting data records by deriving new data elements
  - Using NLP to analyze reports and expose findings or observations as coded data elements or FHIR Observation [STD-FHIR-OBS] resources.

[4] Some IHE Profiles incorporate SNOMED® CT, which is used by permission of the International Health Terminology Standards Development Organisation. SNOMED CT© was originally created by the College of American Pathologists. SNOMED CT is a registered trademark of the International Health Terminology Standards Development Organisation, all rights reserved

[5] FHIR is the registered trademark of Health Level Seven International and the use of this trademark does not constitute an endorsement by HL7.

        - Augmented data may be perfectly accurate or mostly accurate (e.g., weakly label imaging studies)
    - Using data transform tooling to prepare normalized versions of the data to expedite usage of the data.
        - E.g., Creating NumPy [BIB3.1.4-1] arrays for pixel data

3.1.5 Curate Repository

A Repository Administrator reviews the dataset(s) within a repository to ensure that its contents are appropriate for the repository's stated purpose.

Considerations that may have technical interoperability aspects include:

- Quarantining new datasets/data records to prevent their use until necessary curation has been completed.
- Sequestering specific data records to constrain access to them in accordance with a sequestration strategy
    - See Common Mechanics: Sequestration Strategies
- Detecting and managing duplication of data
    - Anonymizing data makes this more challenging.
    - Data identification and provenance may help with this.
        - See Common Mechanics: Provenance.
    - Supporting users of the repository who will be trying to avoid duplication of data records across multiple repositories.
- Confirming appropriate consent
    - Depends on the policies of the repository, the conditions of use of the source data, and the ability to identify the subject of the source record.
- Assessing the balance/bias of the dataset
    - Identifying the criteria for when obtaining additional data records is appropriate.
    - See Common Mechanics: Bias.
- Confirming the contents of the repository is appropriate for the stated purpose.
    - Assessing data for representation of edge cases and balancing of difficult cases (or cases with confounding factors).
    - Assessing data records for inclusion of relevant clinical details, as determined by clinical subject matter experts.

3.1.6 Annotate Repository Data

A Repository Contributor augments data records with new annotation data elements.

Typically, the annotation data elements are observations that an AI Model would be expected to produce given the corresponding data record as an input. Since the repository may serve a variety of AI Models, the annotations might not be of use to all users of the repository, but often the Repository Administrator will have a good sense of the kinds of annotations that may be in demand.

An advantage of adding annotation data elements in the repository dataset is that it may avoid many users of the dataset having to duplicate the annotation work. It is also possible that the repository may be able to bring more resources to bear to do a better job than some users. In a sense, this is a refinement of the Clean Data Record and Curate Repository use cases and a number of the considerations in those sections apply.

Considerations that may have technical interoperability aspects include:

- See Common Mechanics: Annotation.
- Transforming annotation data elements to match the Repository Design
  - o See Entities: Transform

Profile writers may consider some of these questions for further discussion:

- What quality criteria and/or credential checking for contributed annotation data elements would it be useful for the repository to establish?
  - o Such criteria and credential checking might use corresponding attributes in the provenance record and include quality parameters in accepted (or rejected) annotations or data records.

3.1.7 Publish Repository

A Repository Administrator exposes the existence of the repository along with descriptive metadata and prepares the dataset(s) for access.

While some repositories may be published with a limited intended use, others will be broader, and applicable to a great number of AI Model applications. Repositories may serve AI goals beyond those envisioned by the administrator. While the Repository Purpose identified by the Repository Administrator is certainly useful information, an emphasis in this use case should be on documenting the nature of the repository content and leave it to clients to assess the applicability to their model/goal.

See Repository: Discover Repository Content (which is the primary user of the metadata described here) and Repository: Retrieve Repository Content (which is the primary user of the prepared dataset).

Considerations that may have technical interoperability aspects include:

- Encoding metadata used in the repository, including:

    - The intended purpose of the repository (both what type of application; see Section 2.1: Applications of AI in Imaging) and perhaps whether it is appropriate for training, validation, etc.
    - Descriptors of the size and scope of the repository
    - Inclusion and exclusion criteria used to select data records
        - Informs analyses of potential bias in the dataset. See Common Mechanics: Data Quality and Bias.
        - Including clinical considerations behind the criteria might help related decisions by model developers using the dataset.
    - The core set of common data elements in each data record.
    - Endpoints for the discovery and retrieval use cases.
    - Terms of use, licensing and/or limitations on who may access the data for what purposes.
        - This may get into sequestration strategy topics like whether the data is permitted to be shared or to be used for training.
        - Any IRB restrictions on the use of any data elements must be made clear.
- Communicating metadata about the repository
    - This covers the publication mechanisms and top-level metadata. Data record level metadata is covered in Repository: Discover Repository Content.

Profile writers may consider some of these questions for further discussion:

- Where might the "existence" notification be sent/published to? How?
    - Where is the metadata published to or is it retrieved from the repository itself as part of initial interactions by a client?
- Which details are included in the "publication" (this repository exists) versus which details are handled as part of the "discovery and retrieval" during data assembly?
    - I.e., sort of like the difference between what you get in a DIMSE C-FIND vs the full header returned when you retrieve an object.
- Will repositories want/need to publish only specific data elements or data records that meet certain criteria, and might that depend on the authorization of the user or group?
    - E.g., Only the results are provided back to the developer to preserve data elements or eliminate the need for de-identification.

3.1.8 Discover Repository Content

A Repository Administrator prepares mechanisms to support discovery of repository data records by Data Scientists.

This use case is paired with the Dataset Assembly: Obtain Dataset use case. It exists here in the Repository Use Cases group because to a large degree it is the repository administrator that controls the metadata that will be made available for discovery and the transfer protocols supported for metadata retrieval.

Considerations that may have technical interoperability aspects include:

- Query based on presence or absence of specific data elements.
  - The Data Scientist may wish to confirm the presence of input data elements that are essential to their model or confirm that their model's output data elements (I.e., ground truth) are present so they will not have to do annotation to generate them locally.
- Query based on the values of specific data elements.
  - The Data Scientist may need data records that support certain diversity targets or that fit certain criteria of their model.
  - Some data elements that might commonly be the basis for such criteria include patient age, gender, imaging modality, device model/version, body part imaged, performed procedure, imaging protocol, contrast usage, and diagnosis (possibly derived from a codeset like RadLex or Gamuts)
- Querying may include de-identification of elements in the matches being returned or may disallow certain types of queries based on readily identifying information.
  - See Common Mechanics: De-Identification
- Access controls and security of the data need to be addressed to ensure that sequestered or quarantined data elements are not made available to users by the repository's interfaces.

Profile writers may consider some of these questions for further discussion:

- What existing query standards may be appropriate?

3.1.9 Retrieve Repository Content

A Repository Administrator prepares mechanisms to support retrieval of repository data records by Data Scientists.

This use case is paired with the Dataset Assembly: Obtain Dataset use case. It exists here in the Repository Use Cases group because to a large degree it is the repository administrator that controls the transfer protocols supported for dataset retrieval.

Considerations that may have technical interoperability aspects include:

- Bulk retrieval versus fine-grained retrieval
  - A repository may support providing data either as singular requests for specific data records (e.g., through providing an index of records with URIs to retrieve further

data), or through bulk retrieval (as a ZIP file or data volume). These mechanisms likely rely on a query/discovery mechanism.

  - See Common Mechanics: Data Access.
  - Bulk retrieval of clinical data might put undue load onto clinical systems.

- Location and latency of retrieval of data
- Retrieving may include de-identification on-the-fly
  - See Common Mechanics: De-Identification
    - How much de-identification to do at each stage in the data pipeline is a potentially complex issue that is discussed further in the Common Mechanics section.
- Authentication, authorization, and access control
  - See Common Mechanics: Security

Profile writers may consider some of these questions for further discussion:

- To what extent should requests for data be throttled? How is that communicated?

3.2 Dataset Assembly Use Cases

Dataset Assembly Use Cases encompass the collection of data records to assemble a specific dataset to be used for the training, validation, or testing of a particular AI Model.

3.2.1 Identify Dataset Purpose

A data scientist, sometimes with a Product Manager, defines the purpose for which the dataset will be used.

Typically, the purpose is to train, test, and/or validate an AI Model that performs a specific task, so the Identify Dataset Purpose use case is closely linked to the Define Model Task use case (see Section 3.3.1).

The purpose will drive how repository datasets are selected and data records assembled into cohorts, and how training datasets are derived (e.g., a subset of a repository or federated repositories that are appropriate for the question being asked). The purpose may also guide the target performance and robustness of the resulting model which may also influence dataset design and selection criteria. Radiology leadership teams and Oversight Committees may also provide insight to clinical or operational aspects of the data that should be considered.

Considerations that may have technical interoperability aspects include:

- Aspects of the purpose may need to be encoded for later use in inclusion / exclusion criteria when performing queries and selection for relevant datasets and data records.
- Including testing and/or validation in the purpose of the dataset invokes concerns about sequestration and contamination of validation datasets with data records used in training

- Mapping expected data elements in the dataset to AI Model inference inputs and the expected result.

Profile writers may consider some of these questions for further discussion:

- Given the model purpose, what sorts of factors might constitute bias in the training, validation, and test datasets?
    - E.g., patient age, race, insurance status, socio-economic status, scanner type, scanner protocol, image quality, etc.
- What sort of metadata is captured when a model is instantiated?
- What is the fundamental output(s) of the AI Model?
- What is the fundamental input information the AI Model will need?
- Do we need to define the type of data that we want to get feedback on?

3.2.2 Design Dataset

A data scientist defines the dataset(s) they wish to create.

As part of this process, the data scientist provisions the dataset, and assigns the notable metadata to the dataset. Creating robust generalizable models requires a well-populated dataset with data records representative of what the model will be exposed to.

As an example, the data scientist designs a dataset to train a model to classify pneumothorax. They target collecting 100 data records containing a chest radiography study. Their acceptance criteria include that the imaging modality is digital radiography, and the body part is chest. The dataset will be collected and used internally, so the data records will not be de-identified. The data records will contain an output data element for pneumothorax present/pneumothorax absent.

The data scientist considers potential sources of bias in their inclusion and exclusion criteria and plans to assess the dataset during assembly for areas not covered appropriately (for example, skewing toward a subset of their patient population).

Considerations that may have technical interoperability aspects include:

- Defining the specific data elements expected or required in each data record.
- Setting inclusion and exclusion criteria based on:
    - Imaging studies and metadata
        - Imaging Modality type (e.g., include CT but exclude MR, US, and PET)
        - Anatomic coverage of imaging (e.g., must include lung apex and base) and laterality (if relevant)
        - Procedure type (e.g., include use of contrast agent)
        - Series characteristics (rules may be vendor specific)

        - Presence of devices and ground truth of proper position
        - Radiation exposure and dose details
        - Hanging protocols and view settings (e.g., which LUT was applied, whether KOS is available for best slice)
        - Studies that have similar types of prior studies
        - Image quality (whether images are blurred / has artifacts)
        - Use of image compression and whether it is lossy or lossless.
        - Presence of raw pixel data rather than presentation or machine LUT applied.
    - Imaging study outputs
        - Findings (e.g., presence of pneumothorax)
        - Measurements of specific findings (e.g., a nodule with a volume greater than a specific amount)
        - Image scores (quality, grade/codes of grading or classification systems)
        - Severity of findings
    - Billing codes
    - Whether an exam is associated with a re-admission or adverse event
    - Acquisition modality characteristics
    - Metadata of the patient, order, ordering physician, and/or imaging location
    - Diseases or known co-morbidities
    - Belonging to a specific care plan, treatment plan, or clinical trial
    - Timestamps and durations of events
    - Presence of adjudicated truth (data to be used for training, validation, or testing)
    - Being pre-de-identified may be criteria for inclusion or exclusion.
- Encoding format(s) of each data element (which may be vendor specific)
- Standardized code sets or common terminologies to be used
- Whether data element storage is centralized or distributed
    - Centralized storage makes it easier to normalize access protocols and mechanisms.
    - Centralized storage may make data element changes, such as anonymization, easier to manage.
    - Distributed storage allows data to be left and managed in their source systems.

    - Data elements in distributed storage may have their own governance policies, requiring conformance before data can be communicated to the repository.
- Whether data is separate from production clinical systems
    - Separation avoids impacting patient care (safety and performance).
    - Accessing production clinical systems directly may reduce storage and avoid extra annotations (although clinical "annotations" serve a different purpose than AI training, so they may be in a different format, might not be present, or might need to be augmented).
- The degree to which data is de-identified.
    - A privacy officer may ascertain that the dataset needs to be de-identified, which may influence how the dataset will be structured (e.g., how can related records be captured).
    - Certain data purposes may require certain potentially identifying details to be retained (e.g., approximate dates of datasets, anonymous patient identification to link data from the same patient)
- Data provenance mechanisms
    - See Common Mechanics: Provenance
- Providing information to, and performing, an Institutional Review Board (IRB) review
- Accounting for bias and statistical relevance
    - See Common Mechanics: Data Quality and Bias
    - Consider procedures that will be executed during Obtain Data Records, Refine Data Records and Organize Datasets to ensure that datasets are free of sources of bias and enable generalizability.
    - For example, when obtaining, refining, and organizing datasets, search information available to the AI model during training for details that might be correlated with the task result. An AI model that used/depended on these would demonstrate erroneously high performance that would not be reproduced in the real world when these "hints" are not available.
        - A DICOM tag, or image marker, or overlay might indicate an image came from the ICU (as did all the data records showing pathology) which might correlate with the presence of pathology.
        - A subset of patients with asthma that received early and intensive care could well correlate with better outcomes and thus a better prognosis. This might bias the dataset and train the AI model to predict those asthma patients as low risk.
- Patient consent

Profile writers may consider some of these questions for further discussion:

- What kinds of data elements do we expect in data records?
  - Images, Reports, Annotations, Spatial Registrations, Demographics, Lab Results, other Clinical info, Family History, Admitting Diagnosis, Past Procedures, Current Drugs, …
- How, and to what degree, should the Dataset Design incorporate characteristics driven by the FAIR Principles? Some, such as unique IDs for data are already incorporated in the Provenance mechanics etc.
  - The FAIR Principles (Findability, Accessibility, Interoperability, and Reusability) are intended to facilitate the effective management and sharing of datasets. [BIB3.1.2-1]

3.2.3 Obtain Data Record(s)

A data scientist discovers repositories and explores the metadata of the datasets they contain to identify and retrieve data records that would be appropriate for the dataset they are assembling.

This use case interacts directly with several Repository Use Cases. See Repository Use Cases: Publish Repository, Discover Repository Content, and Retrieve Repository Content.

Comparing the published purpose of a repository to the intended purpose of the dataset is often a useful first step. Note that while the dataset being assembled may have a narrow purpose, like training a model to detect the presence of a certain class of liver tumor, repositories often have broad purposes such as a large collection of chest and abdominal CTs with a wide variety of lesions.

In addition to obtaining records from repositories, a data scientist might search local image archives and EMRs directly for relevant patient data from which to construct data records.

Considerations that may have technical interoperability aspects include:

- Tracking the provenance of obtained data records
  - See Common Mechanic: Provenance
- Selecting specific data records from a repository dataset
  - Selection criteria will be driven by the dataset purpose and design.
  - Some repositories may have search capabilities allowing retrieval of a matching subset of records.
  - Even if a repository does support record metadata queries, retrieving an entire dataset and filtering locally allows greater detail and flexibility of filtering.
  - Presence of existing annotation data elements in a data record may be highly desirable since it avoids the cost of creating those annotations.
    - Conversely, externally produced annotation data elements may or may not meet the accuracy or consistency criteria determined by the data scientist for this dataset.

        - Annotation data elements may sometimes be available from a different repository than the one hosting the rest of the data record.

- An important variant case is the situation where the Repository does not allow the data scientist to have access to the data records.
    - In this variant, the data scientist would convey the intended dataset design and record selection criteria to the Repository Administrator. The Repository Administrator would prepare datasets on behalf of the data scientist. During the Train Model step (see Section 3.3.3), the data scientist and Repository Administrator would coordinate a way for the (possibly containerized) model training environment to access the datasets.
    - This case depends on communicating the dataset design, inclusion/exclusion criteria, required data elements (Section 3.2.2), needed data record refinements (Section 3.2.4) and annotations (Section 3.2.5) so that the "external" curator of the dataset can successfully complete those activities on behalf of the data scientist.
    - Training provenance information will also need to be managed in some way between the data scientist and Repository Administrator.
- Seeking out a diversity of data records to avoid sources of bias.
    - See Common Mechanics: Data Quality and Bias.
    - May require obtaining data records from a diversity of healthcare institutions, a diversity of acquisition systems (different makes, models, versions), a diversity of acquisition techniques (protocols), and a diversity of patient populations.

Profile writers may consider some of these questions for further discussion:

- How do they find and access the descriptions and metadata made available in the Publish Repository use case?
- How is consent handled?
- How is data licensing handled?
- What details should the query model include? What would a researcher want to know about data(sets)?

3.2.4 Refine Data Record(s)

A data scientist processes obtained data records and data elements before they are formally incorporated into their own dataset(s).

Considerations that may have technical interoperability aspects include:

- Data records obtained from repositories may include many data elements that are not relevant to the dataset purpose and may be dropped to reduce the bulk of the dataset.

    - Removing unneeded Series e.g., remove “with contrast” series and keep “without contrast” series.
- Manually inspecting data records and disqualifying based on quality criteria.
- Transforming data elements to match the Dataset Design
    - See Entities: Transform
- De-identifying data records
    - A dataset receiver might skip de-identification when the sender claims the data records have already been de-identified, however as part of due care and liability mitigation requirements, the dataset receiver may decide to perform de-identification anyway or do a de-identification check of received data records, especially when the sender and receiver may have different de-identification policies or strategies.
    - Note the potential complexities if de-identification is performed repeatedly on the same data element. E.g., several “minor” date adjustments might aggregate into a major change.
    - See Common Mechanics: De-Identification
        - How much de-identification to do at each stage in the data pipeline is a potentially complex issue that is discussed further in the Common Mechanics section.

Profile writers may consider some of these questions for further discussion:

- Given some of the expected uses of the repository, what sources of bias may be of interest to the model Data Scientist, and thus the repository could help by considering during data harvesting and contribution?
- How does de-identification differ between object types and how are patient record sets correlated? E.g.:
    - Multiple timepoints for same patient
    - Different object types CDA®[6] and DICOM in the same study/timepoint)
- What kind of data gaps or errors might be expected in the data records?
- When/how should data scientists communicate local modifications to datasets back to source Repositories from which the local dataset was derived, in whole or in part.
- Are there means to sub-type the dataset that is meaningful further down in the dataset annotation process? (e.g., contrast or no contrast)

[6] CDA is the registered trademark of Health Level Seven International and the use of this trademark does not constitute an endorsement by HL7.

3.2.5 Annotate Data Record(s)

An Annotator augments data records with new annotation data elements.

Typically, annotation data elements are results that an AI Model would be expected to produce given the corresponding data record as an input, also known as "**ground truth**". Annotation data elements may also be added to conform to data record requirements established during dataset design (see Dataset Assembly Use Cases: Design Dataset).

The ground truth is provided to the AI Model during training to drive the weight adjustments of the training process. The ground truth data elements are held back from the AI Model during testing and validation and are compared against the results produced by the AI Model.

Considerations that may have technical interoperability aspects include:

- See Common Mechanics: Annotation.
- Transforming annotation data elements to match the Dataset Design
    - See Entities: Transform

Profile writers may consider some of these questions for further discussion:

- Do annotation data elements received in data records from the repository need to be tweaked or replaced based on the details of this dataset purpose?

3.2.6 Organize Datasets

A data scientist organizes data records in a primary dataset into smaller sets for training, validation, and testing.

Data records are sub-divided into smaller datasets for training models (training datasets), validating the models as part of the training process (validation datasets), and testing models that have been completed (test datasets). See Appendix B: Glossary. Note that model validation performed as part of training is a different process than functional validation of a medical product. See Functional Validation Use Cases for more details.

Considerations that may have technical interoperability aspects include:

- Dividing available data
    - Proportion of available data allocated to each dataset, e.g.:
        - Training dataset: 65% of data
        - Validation dataset: 15% of data
        - Test dataset: 20% of data
    - Ensuring similar balanced representation in each dataset. See [BIB3.2.6-1].
- Sequestering datasets to avoid mixing training, validation, and test datasets.
    - Analyze for records not being duplicated between those datasets.

    - See Common Mechanics: Sequestration.
    - Consider use of a third-party to prepare test datasets to increase independence and avoid problematic assumptions and biases affecting both the training and test datasets.
- Assessing the balance of data records in the dataset
    - Datasets are balanced with data elements based on type and finding evenly between training, validation, and testing. Unbalanced datasets may represent sources of bias or may limit the robustness of the algorithm across a variety of data.
        - Balance may require each dataset including a diversity of sites and acquisition equipment, diversity of patient characteristics, diversity of pathology, appropriate proportions of normal/abnormal, diversity of image quality, etc.
        - See Common Mechanics: Data Quality and Bias.
    - When organizing datasets, all imaging of a given patient may need to reside in a single dataset to prevent the model from “memorizing” patient-specific features for the prediction task.
    - May require obtaining additional data records
        - See Dataset Assembly Use Cases: Obtain Data Record(s)
    - Analysis of test results may be able to detect bias present in the trained model. This may depend on appropriate population of the test dataset.

Profile writers may consider some of these questions for further discussion:

- To what extent are data records grouped/classified? For example, imaging data with contrast versus no contrast.
- To what extent do training, validation, and test datasets get its own “autonomy”, meaning, that they are not intrinsically connected? For example, is a validation dataset required to be connected to one and only one training dataset?

3.2.7 Share Dataset

A data scientist makes the dataset(s) available to others who will be contributing to the development of the model.

The recipients may include other data scientists who may be collaboratively training this model (or training competing models in the case of an AI competition), repository administrators who archive training datasets for provenance purposes, or who act as custodians for sequestered test datasets, and biostatisticians who would assess datasets for potential biases.

Considerations that may have technical interoperability aspects include:

- Accessing underlying data
    - A dataset could be federated using XDS [STD-XDS] or similar architectural style of registry/repository.

- Uniquely identifying patient and data records
    - Patient and record identity management should be well understood. Lab results or external exams may have different patient identifiers that may need to be managed. Techniques like PIX [STD-PIX] can be employed.
    - De-identification may involve creating new identifiers that may need to be maintained in other data elements within a data record.
- Delivering data to various types of destinations, e.g.,
    - A destination for this data could be a system in the same hospital, perhaps in a different department, or in a central repository.
    - It could be a different hospital (perhaps in the same organization, such as a satellite hospital sending to an academic medical center).
    - It could be a centralized location operated by a cloud service.
- Supporting means to update shared datasets when data records are added, modified, or deleted after initial delivery.
    - See Dataset Assembly: Modify Dataset.
    - Data distribution may need to be tracked to find the specific dataset to be updated.
    - Data may have additional elements added such as annotations. The format and distribution mechanisms need to be considered, e.g., additional DICOM series added to a study may create bias for some use cases.
    - Once shared to a remote location, updates and changes may need to be reported back to the initial sharing entity. When this happens, care must be taken to ensure data is not duplicated in some future use case.
- Auditing and data security
    - As is with the case whenever data is shared, care must be taken to address network and security components.
    - Audit logs should denote that data was shared with other actors.
    - The foreign system must be reachable on the network.
- Data licensing and distribution rights
    - The dataset author should have the rights to share the data.
    - Appropriate permissions for patient consent need to be addressed.
    - If the dataset contains identifying information, distributing it to other organizations may require de-identification
        - See Common Mechanics: De-identification

    - How much de-identification to do at each stage in the data pipeline is a potentially complex issue that is discussed further in the Common Mechanics section.

Profile writers may consider some of these questions for further discussion:

- Must an entity that shares a dataset support reporting of changes by those using the dataset? What is considered a change and when must these changes be reported?
- How, and to what degree, should dataset-sharing incorporate characteristics driven by the FAIR Principles? Some, such as unique IDs for data are already incorporated in the Provenance mechanics etc.
    - The FAIR Principles (Findability, Accessibility, Interoperability, and Reusability) are intended to facilitate the effective management and sharing of datasets. [BIB3.1.2-1]

3.2.8 Modify Dataset(s)

A data scientist adds, removes, and changes data records in datasets.

This use case addresses modifications to datasets assembled for use in training, validation, and testing by the data scientist coordinating those activities as described in the earlier parts of Section 3.2. While some of those modifications will be driven by modifications to the source datasets in Repositories, the process of modifying Repository Datasets is discussed in Repository Use Cases: Contribute Data, Clean Data Record, Curate Repository, and Annotate Repository Data.

Considerations that may have technical interoperability aspects include:

- Reasons for modifications to a dataset may include:
    - **For Data removal:** Patient that is the subject of a data record withdraws consent; Data record is no longer on the relevant to the use case (e.g., the equipment used to acquire the data record is end of life). Patient merge removes patient from cohort (e.g., moves them to a different jurisdiction)
    - **For Data additions:** New relevant data records, and/or annotation data elements for existing data records, become available; New data records are collected to address underfitting. Data records distributed from the Feedback Use Case (see Section 3.7.2)
    - **For Data changes:** Fixing mislabeled data; Inconsistent presentation formats, data may undergo compression (e.g., compressing older images to save space as Scotland does in their 10-year-old national archive)
- Dataset Provenance
    - A training dataset is “frozen in time” when a model is instantiated – if a training set is changed, it really needs a new identity.
    - Retaining previous dataset versions can facilitate troubleshooting or correcting when bad data has been added to a dataset (e.g., incorrect labels).

  - A linear versioning strategy (version 1, version 2, …) does not work, as multiple derivatives could branch at any given time. For example, if Dataset 1 Version 1 is shared between two departments, and both add data records, they have not created two Version 2.
  - Once a version has been "saved" or "published", it should remain frozen, and derivatives must have a new identifier. A ledger (like Blockchain) might be applicable.
  - See Common Mechanics: Provenance.
- Dataset Versioning
  - The version ID of a dataset changes when data is added, modified, or removed. The dataset version ID may be used by AI Models to reference a specific point in time of a specific training instance.
  - Versioning history, with the nature and reason for each change to the dataset may be helpful.
- Relationship to models created from this dataset.
  - If multiple models are created from a single dataset, but the dataset itself has been changing and evolving, it would affect reproducibility.
- Scope changes over time
  - If the scope of a dataset changes significantly (for example, a pancreas cancer dataset is expanded to include normal pancreas cases), the dataset may need to be treated as a new dataset.
- Retiring a dataset
  - Datasets that are no longer used should be retained for audit and recordkeeping purposes but marked as inactive.

Profile writers may consider some of these questions for further discussion:

- How do we define and communicate the identity and provenance of an "instance" of a training dataset?
- How and when do you communicate modifications to datasets that are in use?
- Should data scientists monitor Repositories from which they have assembled datasets so they can propagate relevant Repository changes into their local datasets? If so, what mechanisms would be appropriate?
- When/how should data scientists communicate local modifications to datasets back to source Repositories from which the local dataset was derived, in whole or in part.

3.3 Model Creation Use Cases

Model Creation Use Cases encompass planning and executing the training of an AI Model using training, validation, and test datasets.

3.3.1 Define Model Task

A Data Scientist, sometimes with a Product Manager, identifies the task the AI Model will be trained to perform.

The task may be to answer a specific question, produce a specific piece of information, or perform a specific function. See Landscape: Applications of AI in Imaging for an extensive list of potential tasks throughout the medical imaging process, from ordering to population health, to which AI could be applied.

Initial consideration of the task will likely have taken place when establishing the purpose of the collected datasets, so the Model Creation Use Case: Define Model Task use case is closely linked to the Identify Dataset Purpose use case (see Dataset Assembly Use Cases: Identify Dataset Purpose).

This activity often includes stakeholders that represent clinical and operational aspects of the task the AI Model will perform such as Oversight Committees or Radiology Leadership.

Considerations that may have technical interoperability aspects include:

- Function of the AI Model
    - Identify the output that the AI Model is expected to produce
    - Identify the inputs that the AI Model is expected to have available to it
    - Identify the breadth of values the AI Model expected to be able to handle for its inputs
    - Identify any values or situations where the model is not expected to be able to function properly
    - Identify how the AI Model may access its inputs in a clinical workflow and what transforms may be applied
        - See Clinical Usage Use Cases: Access Inference Data Record.
    - Identify how the output will be applied in clinical usage
        - See Clinical Usage Use Cases: Use Result.
        - Consider also what format the outputs will be returned
    - If this is a previously distributed AI Model being re-trained, assess whether the expectations of the task or performance have changed. E.g., re-training for updated PI-RADS scoring rules.
- Sources of Bias

    - See Common Mechanics: Data Quality and Bias.
    - Assess the task, and possibly do a literature search, to identify aspects of the inference data that may potentially be sources of bias.
        - E.g., is it a diagnostic task that is known to, or could be suspected to, vary with patient age, sex, race, socioeconomic status, or other criteria?
    - These details will be applied in the use cases for collecting data records, assembling balanced datasets, and performing training, validation, and testing to ensure that potential sources of bias are considered, hopefully mitigated, and at the very least, transparently documented.
- Level of autonomy
    - Identify whether the AI Model is intended to perform the task autonomously, autonomously with oversight, or interactively/semi-autonomously with a human.
    - Greater levels of autonomy may require more rigorous testing, monitoring, and involved methods of integration.
- Identify how the AI Model will be incorporated into the clinical process
    - This may include computer-aided triage (CADt), computer-aided detection (CADe), computer-aided diagnosis (CADx), or computer-aided optimization (CADo). See [BIB3.3.1-1].
- Explainability of results
    - Explainability refers to the notion of being able to produce information that “explains” how an AI Model arrived at the task result that it generated; for example, which input values most influenced the result.
    - Explainability is of particular interest because the self-organizing, black-box nature of many AI Models makes it difficult to assess the “design” of the algorithm for validity, or potential flaws or weaknesses. With human-designed algorithms, the designer can explain the rationale behind the algorithm. With more traditional machine learning algorithms (e.g., regression models), the derived function is visible. For an extensive survey of methods for explaining black-box models, see [BIB3.3.1-4]
    - Considering at design time how to incorporate explainability will support several “downstream” use cases. The focus and explainability goals vary with different personas and use cases. See [BIB3.3.1-3] and [BIB3.3.1-5].
        - During Functional Validation: Test Model (by User) and Clinical Usage: Use Result, explainability may help users, such as Radiologists and Clinicians assess whether they trust the AI Model.
        - During Functional Validation: Test Model (by User), explainability may help the Oversight Committee and Radiology Leadership assess whether the AI Model

        training is transferable to their local patient population and identify boundaries for the AI Model performance.

        - During Functional Validation: Test Model (by Regulator), explainability may help the System Regulator assess that the efficacy demonstrated during validation is representative of it’s likely performance in the field
        - During Feedback: Collect Model Feedback and Feedback: Adjust Model, explainability may help the Data Scientist assess the nature of any performance issues exposed by a case where the AI Model generated an incorrect result.
    - Black-box AI models may require external XAI techniques (post-hoc analysis) to provide explainability and interpretability, such as:
        - Visual explanations (e.g., using heatmaps, saliency maps, or class activation methods)
        - Explanations by simplification and feature relevance explanations that can be plugged to any model (i.e., model-agnostic) with the intent of extracting some information from its prediction procedure.
- Target environment where AI Model will function
    - E.g., hardware. Specific considerations may apply in resource-restricted deployments like in embedded systems or on portable devices where memory and/or processing power may be limited.
- Target performance and necessary clinical performance
    - See Common Mechanics: Model Performance Metrics for potential metrics of relevance
        - Note: Depending on the use case and mode of integration, the required level of performance for the model may significantly differ from human performance on the same task.

Profile writers may consider some of these questions for further discussion:

- How does AI Model packaging and distribution be reflected in the model task definition phase?

3.3.2 Orchestrate Training

A Data Scientist designs the structure of the AI Model and plans the process that will be used to train the Model.

Considerations that may have technical interoperability aspects include:

- Selecting a model framework and/or toolkit

  - Developing training software from scratch can be very time consuming. Building training software on top of existing toolkits reduces time and takes advantage of training techniques and optimizations built into toolkits.
  - See Appendix D.4 for commonly used toolkits.
- Incorporating the designated training, validation, and test datasets
  - Planning for the detection (and mitigation) of overfitting to the training dataset during the training process depends crucially on avoiding overlap of data records between the training, validation, and test datasets during the dataset assembly process.
    - See Dataset Assembly Use Cases: Organize Datasets
  - The training orchestration should include a plan for how to incorporate new data records (while avoiding potential biases). Since the benefit of increasing the size and diversity of the datasets may merit ongoing efforts for data assembly, new data records may become available over the course of the training process.
    - See Repository Use Cases: Contribute Data and Dataset Assembly Use Cases: Refine Data Records.
  - Sequestering data in training, validation, and test datasets to avoid inappropriate re-use of data records or leakage of labels and other relevant correlates.
    - See Common Mechanics: Sequestration.
- Performing necessary transforms upon data ingestion
  - See Common Mechanics: Transform
- Augmentation of datasets to enrich model training.
  - This involves “virtual records” that are generated either beforehand, or on the fly during training, to encourage the model to be invariant to these transforms (e.g., rotate study 1 degree to create virtual datasets)
- Selecting model type and implementation
  - For a neural network, this may involve selecting the appropriate neural network architecture and implementation (e.g., a ResNet built in PyTorch located in the “Torchvision” package)
  - For a traditional learner such as a gradient-boosted tree, this may involve selecting one of many implementations which may contain subtle differences (e.g., XGBoost)
  - See Appendix A: AI Background
- Determining model hyperparameters, e.g.:
  - Number of layers
  - Number of channels

- Model Identification Strategy
    - Model identification includes global uniqueness, versioning, provenance, training history, and the model owner.
- Selecting initial weights
    - Models can be trained from scratch (randomly initialized weights) or fine-tuned (pretrained model weights) via transfer learning.
    - Models that are being fine-tuned do not have to originate from the same research organization.
- Determining training hyperparameters, e.g.:
    - Number of training epochs
    - Learning rate
- Organization of epochs and validation/testing patterns
    - Model testing methodology as the model is being trained.
- Distributing training across systems using the same dataset
    - Spreading computation of model training and training data storage across multiple machine nodes or GPUs to enable parallel computation to process larger models and larger training data sets [BIB3.3.2-1] [BIB3.3.2-2] [BIB3.3.2-3].
- Collaborative training across institutions using distinct datasets to update a global model
    - May partially improve a global model using part of an institution's data, and then do institution specific fine-tuning using a held-out dataset to refine the model
    - There are several types of collaborative training [BIB3.3.2-7], which scale from learning between individual nodes all the way to learning across enterprises.
        - Model ensembling occurs when each institution trains their own model, and those weights are then averaged to create a final model.
        - Federated learning occurs when updates to model weights from each institution are combined [BIB3.3.2-4] [BIB3.3.2-5] [BIB3.3.2-6].
        - Cyclical weight transfer occurs when a model is trained at one site, then transferred to another site and further tuned, and then transferred to yet another site and further tuned.
        - Split learning occurs when partial training of models occurs at each institution and further training is done without access to raw data
    - Transfer of collaborative training assets, e.g., training frameworks, training libraries and packages, configuration data, expected data to be used, resulting model weights
    - See Common Mechanics: Exchanging Models

- Fine-tuning or starting from scratch
    - Fine-tuning increases performance of an existing model with additional data. See [BIB3.3.2-8].
    - In a clinical environment, a model trained elsewhere may be fine-tuning for locally acquired data.
        - This decision may depend on regulatory requirements; for example, the specific class of a regulated medical device to which the model belongs may have constraints on re-training.
- Hyperparameter optimization
    - Once a model has been trained, the model and training hyperparameters may be changed, triggering another training of the model. This process allows for the discovery of improved model configurations and weights.

Profile writers may consider some of these questions for further discussion:

- In collaborative training, how does one ensure that the training and test datasets assembled at each site conform to the dataset design (see Design Dataset) in terms of inclusion/exclusion criteria, data elements, format, or other factors?
- Can/should collaborative training use the same mechanisms as the Application Distribution and Management Use Cases?
- When conveying model hyperparameters, how much needs to be interoperable?
- When different institutions are communicating model weights and deltas, should interoperable technologies be considered?
    - See Common Mechanics: Exchanging Models
- What further differences need to be described if “the training is taken to the data” or “the data is taken to the training location”?

3.3.3 Train Model

A data scientist trains an AI Model using data records from a training dataset.

Model training uses the supplied training dataset to adjust the weights and biases of the model nodes to fit the transformed outputs of the models to match expected ground truth as closely as possible. The training process also makes use of validation dataset(s) as described below.

Considerations that may have technical interoperability aspects include:

- Loading training data
    - Expedient fetching, parsing, and transforming of data elements is a consideration to not inhibit model training.

        - Some aspects may be performed in real-time; however, this may also be performed in advance. See Model Creation Use Cases: Orchestrate Training.
    - Efficient transfer of the data elements to where training occurs in a format it “understands.”
    - The training dataset can be transferred to where the training framework is running, or the training framework can be transferred to where the training dataset is located.
    - Training can be run on local datasets or on remote datasets in a collaborative fashion.
- Executing training
    - The steps that occur inside the training framework can be distributed amongst different institutions (known as collaborative learning). See Model Creation Use Cases: Orchestrate Training.
    - Processing power and storage for training can be local or distributed across multiple servers.
    - See Appendix A: AI Background.
- Handling error conditions
    - Overfitting: Occurs when the model function fits limited training data too closely and performance on novel data is lower as a result [BIBA.1.14].
    - Bias: Occurs where assumptions are made on the completeness of the training data [BIBA.1.15]. See Common Mechanics: Data Quality and Bias.
    - Predictive error: Occurs due to systematic prejudice from faulty assumptions. [BIBA.1.16]
    - Variance error: Occurs when there are changes in predictive estimates of models with different training data.
    - Not having access to training datasets or the ability to perform parsing or transforming of data elements.
    - When performing collaborative learning, not being able to communicate with other participants as required.
- Completing training
    - Capturing model performance during training and once it is complete.
    - Exporting model weights once training is complete.
        - See Common Mechanics: Exchanging Models
    - Capturing model identity and other metadata
        - The provenance of the resulting model includes the identity of the seed model and dataset used for training

        - Each model generation should have its own identity
        - See Common Mechanics: Provenance
    - Selecting a trained AI Model
        - Using the validation dataset, the "best" candidate model is identified based on task-specific metrics
            - See Model Creation Use Cases: Define Model Task
            - See Common Mechanics: Model Performance Metrics
            - Note: this use of validation corresponds to the AI usage, not the medical device usage. See Functional Validation Use Cases for further details.
    - The selected model then proceeds through the model lifecycle for further testing and evaluation
        - See Test Model and Functional Validation Use Cases

Profile writers may consider some of these questions for further discussion:

- How is the dataset represented in the model?
- How are training responsibilities shared amongst different enterprises?
- How do you decide when you are done training (e.g., also avoid overfitting)?
- What are the implications of subsequent training of an AI model using a training dataset with different characteristics than the earlier training datasets?
    - For example, consider an AI model trained on a dataset of CT studies from a specific scanner model of a specific vendor. If the AI model is then fine-tuned using a different vendor or a different scanner model, is this valid? How is the Data Scientist involved in that decision?
- What is the impact on the identity of a model when fine-tuning it?
- What is the impact of "changed annotation styles" around fine-tuning?
- Should models that fail during the training process be captured, so that data scientists can identify features with best or least performance in the AI Model with their statistical analysis?

3.3.4 Test Model

A data scientist and/or biostatistician evaluates the performance of a trained model using a test dataset.

Test datasets have been organized (see Dataset Assembly: Organize Datasets) to provide an independent and unbiased estimate of model performance, see the discussion of appropriate test datasets below.

Testing may be conducted by a separate team of data scientists and biostatisticians not involved in training. Pre-clinical validation, not to be confused with use of the validation dataset during model training, may be conducted by the same organization or a separate pre-clinical testing organization.

Considerations that may have technical interoperability aspects include:

- Using appropriate test dataset
  - Ensure data and/or patients from the training and validation datasets are not included in the test dataset. Test data records need to have been "held out" so the test is independent of the model training.
  - The test dataset should be large and diverse enough and have appropriate statistical distributions of data records to ensure a low-variance, unbiased estimate of model performance.
  - See Common Mechanics: Sequestration
- Assessing test performance via relevant metrics
  - See Common Mechanics: Model Performance Metrics
- Identifying that a model is ready for next activities (e.g., external functional validation)
- Recording and parsing results including model performance and confidence
- Tracking provenance of the test dataset and the model being tested
  - Indicate the combination of the unique model version and dataset version used
  - The training, validation, and testing process is often iterative and performed repeatedly, especially if the model does not perform as expected in subsequent functional validation. See Functional Validation Use Cases.
  - Capture details in audit logs
  - See Common Mechanics: Provenance.
- Communicating results and provenance
  - Subsequent testing and functional validation activities and Distribution and Clinical Usage use cases use the results and provenance to help assess the application performance and build trust and confidence in the results.
- Exporting test results and datasets for
  - Peer-reviewed publications.
  - AI Challenges
  - Collaborative model development

Profile writers may consider some of these questions for further discussion:

- Does testing differ (e.g., use different test datasets or performance criteria) depending on whether this test follows a "from scratch" training run, a "transfer learning" training run, or a "fine-tuning/localization" training run?
- How are versions assigned to each unique test model / dataset combination? How does this vary from production versioning in a continuous learning model? What kind of reporting is necessary?

3.4 Application Distribution and Management Use Cases

Application Distribution and Management Use Cases encompass making an AI Model and associated components into an AI Application that can be distributed, installed, integrated, managed, and used.

Five deployment patterns are considered in the following use cases. The deployment pattern affects aspects of distribution and clinical usage as noted in each use case in this section and in Clinical Usage Use Cases.

- Embedded – The model/application is developed as an inherent component of a parent product system to provide a particular function. E.g., an AI Model might be embedded in a Computerized Physician Order Entry (CPOE) system to suggest appropriate imaging procedures based on the details of the patient admission.
- Integrated – The model/application is developed as an embedded function but uses an external computing resource. E.g., an AI Model might be integrated with a CT scanner to perform image reconstruction of scan data which is done using a companion GPU server.
- Standalone – The model/application is deployed on its own in the target environment and interacts directly with the target environment. E.g., an organ segmentation server interacts with the PACS/VNA or a DICOM router, or a segmentation worklist, to identify and retrieve CT studies needing segmentation, and interacts with PACS/VNA to store segmentation objects to the study.
- Hosted – The model/application is designed to be executed on a compatible AI hosting system that was developed separately, but the model/application interacts with the environment to manage its activities and to transfer data. E.g., an AI Application that segments lung tumors might be hosted on an AI Platform. The AI Application claims lung tumor segmentation tasks posted to a worklist server, retrieves images and stores results to the local PACS.
- Mediated – Similar to Hosted but the model/application does not interact directly with its environment. The mediating host takes care of all task orchestration and data transfers. E.g., an AI Model that detects pneumonia in chest x-rays might be a mediated component on one of several compatible AI Platforms which take care of identifying tasks, determining which application to run, retrieving DICOM data, extracting/transforming pixels for the model, and encoding the model result into a medical format for inclusion in the patient record.

3.4.1 Package Application

A data scientist creates a distributable “application package” containing AI Model(s).

An AI Application could encompass just the AI Model (data-only packages), or include connectors, transforms, and services in self-contained executable containers (executable package), as well as possibly traditional algorithms. See Entities: AI Model and AI Application.

Shared models must be interoperable in the ecosystem to which they are delivered. The packaging must also be compatible with the architecture of the target environment. See Appendix B Architectural Considerations.

Sharing a package that contains just the AI Model in a common format makes most sense in a research environment as it allows the receiver to run, improve or re-define the model in their own development environment while using their own tools. On the other hand, sharing a complete executable AI Application package is the most common way for finished clinical applications ready for deployment in a clinical environment.

Considerations that may have technical interoperability aspects include:

- Supporting different deployment patterns (see Section 3.4)
    - Packaging Hosted and Mediated applications will need to consider machine readable descriptions of the application capabilities and needs to be compatible with the platform. Packaging Embedded and Integrated applications will likely have few, if any, interoperability aspects.
- Encoding and provisioning minimum hardware and software requirements
    - The AI Model and surrounding AI Application will need to be tested that they operate according to specifications.
    - E.g., hardware minimums for CPU processors, RAM memory, hard drive space, Graphic Processor (GPU) for running the model in a clinical production environment.
- Identifying packaging with other software or hardware
    - Once a model is functionally validated and has obtained required regulatory clearances it may be packaged with and sold as part of another medical software product, e.g., an Image Display, or packaged and deployed independently.
    - In this case a Docker file format may be well suited to contain a self-contained executable model package.
- Facilitating verification of correct installation in the target environment (see Section 3.4.4).
    - Including data for operational testing (sample data and corresponding correct output)
- Preparing model and application details useful for enumeration (this information is called “AI Model manifest” going forward)

  - Model function (e.g., classification or segmentation, for what body part, for what imaging modality)
  - Model identity, source, and format
  - Model provenance
    - See Common Mechanics: Provenance
  - Level of model testing performed
  - Model performance (for evaluating and building trust in the model)
  - Data input and output characteristics, including information related to explainability of the analysis outcome and help further trust in the AI
    - What inputs are required to be present and what inputs (if any) are optional
    - What the requirements and expectations are for each input, e.g., format and range of values for which the algorithm has been trained
    - What outputs are produced and in what format
  - Data transformation characteristics
    - What transforms may need to be done on each input and what transforms should not be done (e.g., lossy compression of image pixels)
    - What transforms are acceptable for each output (so that the semantics are preserved)
    - See Common Mechanics: Transform
  - AI Application Host characteristics (e.g., operating system and platform)
  - Appropriate use of the AI Application and contraindications

Profile writers may consider some of these questions for further discussion:

- What format should be used for encoding the AI Model manifest (see Appendix C.4 Reference Toolkits)?
- Which container formats should be used for distributing self-contained AI Applications (Docker is a common, Windows self-installable, RAR files)?
- What specific regulatory rules should be conveyed?
  - E.g., regarding labeling intended use and warnings.

3.4.2 Distribute Application

A data scientist distributes the packaged AI Application.

Considerations that may have technical interoperability aspects include:

- Supporting different deployment patterns

  - See Model Creation Use Cases: Package Application
- Standardized mechanisms for discovery and distribution
  - See Model Creation Use Cases: Discover Application
  - Hosted and Mediated applications may depend on standardized mechanisms for discovery and distribution.
  - Standalone applications will be less dependent.
  - Distributing Embedded and Integrated applications will likely have few, if any, interoperability aspects since they are part of a parent product.
- Identifying targets for sharing the AI Application
  - It could be shared between institutions, for research purposes, or for commercial purposes.
- Choosing a distribution method, e.g.:
  - **Model Zoos:** The data scientist could upload the model package to a publicly accessible platform to allow other data scientists to download the model. (Note: Model Zoo is preferred over Model Repository in this white paper to avoid confusion with Dataset Repositories.)
  - **File Sharing:** The data scientist could share the model directly using a common file sharing service or a dedicated server.
  - **Website Download:** Publishing a model or AI Application on a vendor website, or from a jurisdictional oversight agency. The application package might be stored in an Enterprise Resource Planning (ERP) system.
  - See Common Mechanics: Exchanging Models
- Sharing the distribution reason
  - For functional validation
  - For research use
  - For clinical deployment
  - For trial use
- Releasing and tracking subsequent versions of the AI Application
  - The AI Model packaged in the AI Application may be re-trained to improve performance, incorporate revised definitions of correct results (E.g., revised PI-RADS scoring rules), support a more diverse range of cases, etc.

Profile writers may consider some of these questions for further discussion:

- Should the distributed models be access-controlled, and by what means?

- What transport protocols and security mechanisms should be used for model transfer (e.g., HTTPS or SFTP)?

3.4.3 Discover Application

A data scientist and/or IT Operations finds and obtains an AI Application for local use, based on needs identified by Radiology Leadership.

This use case is most applicable to Hosted, Mediated, and Standalone model/applications.

Considerations that may have technical interoperability aspects include:

- Supporting different deployment patterns (see Section 3.4)
- Discovering Hosted and Mediated applications may depend on standardized mechanisms for discovery and distribution (see Section 3.4.2). Standalone applications will be less dependent. Discovering Embedded and Integrated applications will likely have few, if any, interoperability aspects since they are part of a parent product.
- Querying for AI Applications (e.g., based on attributes in the model manifest)
    - Intended purpose.
    - Supported modalities.
    - Status of the model
    - Target O/S
    - Application area
    - Body part
    - Official regulatory and release status of the model
    - AI Model performance for a given set of conditions
  - The definition of a standard format for a model manifest file would be helpful to store properties of each model in a structured form to allow for structured queries.
- Retrieving an AI Application
  - A retrieval location should be made available to retrieve the AI Application, such as an HTTP link.
  - There should be means to assert the authenticity of the AI Application, so the user has the confidence that they downloaded the expected component.
  - See Common Mechanics: Provenance.
- Delivering the AI Application onto an AI Application Host
  - Once the desired model is identified based on the query results, the user copies the model into their own work area (if working on a ML cloud platform) or downloads it to their local system.

Profile writers may consider some of these questions for further discussion:

- Do different types of users have access to different AI Applications for discovery?

3.4.4 Install Application

IT Operations unpackages, installs, and configures a packaged AI Application.

This may be done in collaboration with a local Data Scientist and/or ML Ops Analyst. It also leverages oversight from the Information Security and Privacy Officer.

Considerations that may have technical interoperability aspects include:

- Supporting different deployment patterns (see Section 3.4)
    - Installing Hosted and Mediated applications will be defined by the platform.
    - Installing Embedded applications will be inherent in the installation of the parent product, as will Integrated except for the external component.
    - Installing Standalone applications might involve many different commercial or proprietary software environments to run the model.
    - Note that except for Embedded applications, all the other patterns may be deployed as on-prem installations (workstation or server-based) or as a cloud-hosted service.
- Supporting different architectural patterns (see Appendix B Architectural Considerations).
- Conveying the general installation process; this typically includes:
    1. Running the installation program,
    2. Registering the AI Application within the host environment.
    3. Licensing of the AI Application.
    4. Configuration of the AI Application.
    5. Testing the proper installation of the AI Application, e.g., by using it on test data that was packaged with the application and comparing the inference output with the expected result.
- Automation of the AI Application installation
    - E.g., whether the installation fully automated (e.g., in case the model is deployed as a Docker file) or does it involve manual steps.
    - If a previous version of the same model is installed, an un-installer may have to be run first or multiple versions may be used in parallel.
- Incorporation of AI Application details into local management databases.
    - E.g., model identity, purpose, provenance, etc.

    - Mechanisms for confirmation of model ID and version after the download to ensure they got the correct model.
- Registering an AI Application in a platform (model orchestration framework)
    - This includes describe its inputs, outputs, and resource needs for running the models (e.g., CPU/GPU, memory, and temporary file storage requirements)
    - This helps the framework manage the resources when starting up running an inference job with the AI Application, e.g., when running the same application in parallel on a distributed set of compute nodes.
- Standardizing configuration settings for AI Applications
    - How to turn optional features on/off
    - Setting input and output paths
    - Identifying IP address, hostname, and port
    - Identifying SSL, authentication, and authorization
    - Selecting a data access protocol like DICOM (and accompanying data like AE title)
    - Fault tolerance behaviors like retries and timeouts. See also discussion in Section 3.6.3 Perform Inference.
    - Sources of additional data of the AI Application, e.g., access to priors or other non-imaging data like FHIR resources (which, as noted above, may also require connectivity data like IP address, etc.).
- Licensing mechanisms for AI Applications
    - E.g., seat-based licenses, per-use subscription models.
- Limiting access of the AI application to clinical resources, especially if they contain PHI, to avoid PHI leaks, e.g., in deployments spanning multiple networks.

Profile writers may consider some of these questions for further discussion:

- How to mitigate possible PHI leaks from within the deployed model?
    - E.g., preventing unauthorized access to a VM with the AI Application while it is processing patient data.
- How can the input data and resource needs of AI Applications be captured in a standardized way?

3.4.5 Integrate Application

An ML Ops Analyst prepares the environment and its users to perform inference with an installed AI Application.

Considerations that may have technical interoperability aspects include:

- Supporting different deployment patterns (see Application Distribution and Management Use Cases)
    - Integration will be particularly important for Standalone and Hosted applications which interact directly with their environment.
    - Integration of Mediated applications will be largely handled by the mediating platform.
    - Integration of Embedded and Integrated applications is essentially handled by the parent product.
- Supporting different architectural patterns (see Appendix B Architectural Considerations).
- Coordinating execution of the AI Application in response to requests for processing
    - For Standalone, Hosted, and Mediated applications it may be appropriate to take on the role of the AI Performer or be embedded into an AI Performer as a proxied AI Model Application following the IHE AIW-I Profile (see [STD-AIW-I])
- Coordinating the necessary transforms of incoming data to be suitable for the AI Application.
    - See Common Mechanics: Transform
    - Defining interfaces for data input and output to AI Models, in cases where the model comes in "data-only" package form and/or requires input data in a particular format rather than accepting a wide range of DICOM objects.
    - Providing and configuring appropriate software modules to perform transforms on input data to make them conform to whatever data format the model requires, e.g., decompression or resampling of pixel data.
- Coordinating the necessary transforms of outgoing data suitable for consuming applications.
    - See Common Mechanics: Transform
    - Providing and configuring post-processing functions that format output data of a model, e.g., wrapping raw pixel data or segmented volumes into DICOM compliant objects. Jupyter Notebooks are a common mechanism used for this purpose.
    - Mapping model outputs in form of simple string identifiers, e.g., "pneumo-thorax suspected", or numeric measurements, to coded concepts compatible with the consuming systems
        - DICOM Supplement 219 [STD-DICOM-JSONSR] may be used for transforming locally defined strings or measurements returned by the AI application in JSON format into coded concepts defined in public dictionaries for inclusion in DICOM SR results as per [STD-AIR]; Similarly, HL7 FHIR may be used for representing observations into standardized data structures.

    - Mapping an AI Application result from the DICOM format it provided into another DICOM format that is compatible with the consumer of the result, e.g., transforming a DICOM Surface segmentation object into a DICOM Segmentation object.
    - Mapping supportive elements like explainability documentation that goes with the outgoing data.
- Chaining AI Applications together to form processing pipelines.
    - E.g., feeding the output of a lung nodule *detection* application from vendor A into a lung nodule *characterization* application from vendor B.
    - Transport issues may need specific attention, e.g., it may require bridging between a cloud-based AI Model and another clinical app running on a local workstation.
    - Consider differing needs of AI Applications implemented using traditional image processing algorithms with AI Applications based on deep-learning networks
        - Although being combined, applications could have different needs, e.g., one requiring GPUs vs. the other requiring CPUs, or one is parallelizable and can be distributed to multiple compute nodes and the other is not.
- Registering AI Applications within a model execution framework to enable their automatic invocation based on the nature of incoming data, including information such as:
    - The types of data the AI Application can do inference on
        - E.g., individual imaging studies for AI Applications performing tasks to support a radiologist’s interpretation of images.
        - E.g., sets of imaging studies and other EMR data for AI Applications performing assessment of patient treatment regimens and population health.
        - See also Section 3.6.2 Access Inference Data Record
    - The types of analysis the AI Application performs
        - Mapping of AI Applications to procedure codes used in scheduling exams or to other types of codes used in the operating environment.
- Verify correct operation of the AI Application.
    - Using sample data and results if they were packaged with the AI Application
    - For initial verification before go-live
        - See also Application Distribution and Management Use Cases: Test Model (by User).
    - For ongoing surveillance after go-live by running regular checks with controlled input data and expected results.

        - See also Application Distribution and Management Use Cases: Monitor Application Service.
    - Maintaining a “status” of AI Applications within the operational environment to indicate whether it is verified, ready for production, etc.
    - This may require connectivity into IT monitoring systems and may use IHE ATNA [STD-ATNA] or SOLE [STD-SOLE].
- Connecting into the right environment
    - E.g., development, pre-production, staging and production environments; or where AI Applications would only be used for processing patient exams for one physician only.
- Promoting an AI Application from one environment to another
    - This switch-over may require reconfiguration of network settings, permissions, and licenses, and maybe facilitated by a hosting AI Orchestration framework.
    - Keep systems in a “pre-go-live” state, e.g., perhaps triggering an AI service is live, but delivery of results goes to an adjacent system.
- Serial and parallel workflows
    - In Serial Workflow, if the AI Application fails, the destination may not receive the original imaging study objects. In Parallel Workflow, the AI Application does not send original imaging study objects to avoid duplication, and the Destination must be prepared to receive and associate objects from separate transmissions and potentially out of order.

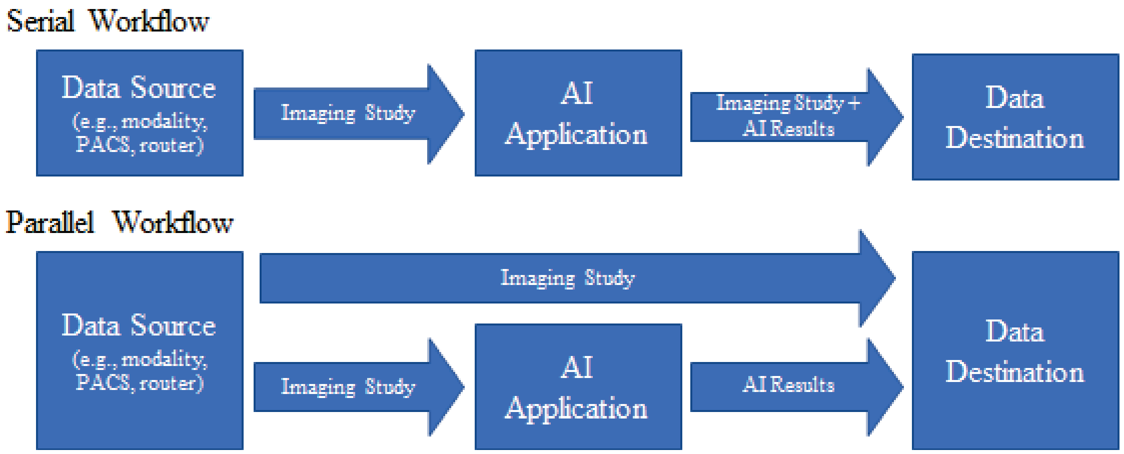


Figure 3.4.5-1: Serial versus Parallel workflow

- Supporting billing and reporting systems
    - These systems must be configured to allow for tabulating the exams on which a model is run to monitor proper operation, and to provide input to billing.

- Training clinicians and support personnel on operating the AI Application.
- Training clinicians on how to appropriately use the AI Application outputs
    - Product specialists need to train and educate potential users of the model to be aware of how the model output is provided to them.
    - E.g., the behavior of the AI Application that has been integrated in PACS Viewing stations or on dedicated workstations, and how they are expected to interact with them.
    - This training covers both the model itself and the other systems the user interacts with in the context of the model (e.g., result display).

Profile writers may consider some of these questions for further discussion:

- Are there mechanisms for twin pathways for the same processing task, e.g., a non-AI original path for a breast CAD application, and a path with an AI-based breast cancer detection application, that would allow to compare the performance of both applications across a larger collection of exams? How does this relate to functional validation steps?

3.4.6 Manage Application Service

An ML Ops Analyst makes the AI Application available as a Service and supports ongoing operation including control, monitoring, and scaling.

Considerations that may have technical interoperability aspects include:

- Supporting different deployment patterns
    - See the introduction section of Application Distribution and Management Use Cases.
    - Hosted and Mediated applications will depend on the platform to ensure the model is ready to perform assigned tasks, possibly using DICOM Service Discovery and Control [STD-DICOM-DISCOVERY]. It could do this by starting it up on a dedicated server or VM to allow for a warm start, or at least ensure there are enough CPU / GPU, and memory resources available to perform efficiently when it is started up to process data. To allow the proper allocation of resources for each model, it is vital that the resource requirements are discoverable in the model registry.
    - Standalone applications will depend on the IT Operations and/or automation provided by the vendor to handle startup, resource allocation and scaling.
    - Embedded applications will depend on their parent product, i.e., the proper sizing of the hardware is expected as part of the design phase of the device, and not the responsibility of operator action.
- Monitoring AI Applications operational conditions
    - Processes should be in place to monitor the proper operation of the model to identify:
        - resource bottlenecks

        - problems with input data
        - unusual environment conditions like low disk space and high temperature
        - any other unexpected behavior that would require corrective actions by an IT administrator.
    - Consider built-in monitoring capabilities of the infrastructure running the AI Application, e.g., Kubernetes.
    - Monitor turnaround times, i.e., the time it takes to process a study with a given model and make results available to downstream systems.
    - Monitoring may be done internally by ML Ops Analysts or remotely by Product Support Teams
    - System performance issues may be an indicator of possible clinical performance issues.
        - E.g., if a model is uncharacteristically taking hundreds of gigabytes of disk space, there may be clinical impacts.
    - Clinical monitoring is distinguished here from operational monitoring described above.
        - See Application Distribution and Management: Monitor Application Service
- Scaling AI Applications
    - Multiple instances of a Task Performer might be launched to improve turn-around time (at increased computing cost). Work could then be distributed across these instances.
    - In case the model is run in a virtualized environment with dynamic resource allocation, such as a Kubernetes cluster, the framework could dynamically increase or decrease the available hardware resources to control optimal hardware utilization based on recent and current workload.
    - There may be conditions that warrant “burstable” capability for AI Applications. For example, adding additional capacity during daytime clinic hours and scaling back during the night.

Profile writers may consider some of these questions for further discussion:

- How to automatically startup or shutdown AI Applications safely and without interruption to the clinical operations, e.g., in case of environmental issues.

3.4.7 Monitor Application Service

An ML Ops Analyst, Product Support Team and/or the Information Security and Privacy Officer monitors the clinical performance of an AI Application during use.

In contrast to operational performance (such as having adequate resources), clinical performance is monitored in terms of metrics such as the sensitivity and specificity of the inference over time. Detection of performance degradation during monitoring may inform decisions about interventions and/or remediation.

Considerations that may have technical interoperability aspects include:

- When an AI Application is monitored
    - In real-time
    - Regularly scheduled.
    - Triggered by events such as:
        - installation of a new scanner, replacement of hardware or firmware components, or new reconstructions algorithms
        - introduction of a new protocol or new mode of operation (e.g., multi-energy)
        - installation of a new transform / platform / operating system / hardware.
        - changes in local patient demographics (e.g., starting to serve pediatric patients).
- How an AI Application is monitored
    - Control data records with known outcomes can be used on a regular basis to detect when models skew from known truths.
        - This method detects issues with the behavior of the AI Model and Application but will not detect issues with the acquired data that goes into the inference data records during use (for example, a replacement x-ray plate might produce images that are slightly different in a way that throws off the AI Application; this may require human verification and should be triggered through events as noted above).
    - Automated monitoring works best when the "expected" and "actual" result data elements are in machine-readable formats.
        - For example, if the expected result for a segmentation task on a control data record is stored as a DICOM segmentation object, but the AI Application generates a secondary capture of a segmented image, comparison is not easy to automate and may depend on manual comparison.
    - Periodically, manual checks by the Oversight Committee could be performed on randomly selected cases.
        - This may utilize feedback mechanisms (see Feedback Use Cases).
- Aspects of an AI Application to monitor
    - Whether the AI Application is producing "correct" results
    - The computational resources being used by the AI Application.

    - The processing response time and length of the processing queue.
- Types of access to AI Application monitoring information
    - IT Analysts review logs and take appropriate action (e.g., take offline)
        - If permitted, AI Application Vendors may also review logs to recommend appropriate action.
        - See also Feedback Use Cases: Circulate Feedback.
    - Payors may have access for billing and reimbursement purposes.
    - Health Information Management Team may have access to supplement the patient health record.
    - Health Policy Agencies may have access for population health reporting requirements.
    - System Regulators may have access to track performance across institutions for recall and warning issuance
    - The types of data provided in monitoring information may differ depending on the needs and privileges of the recipient.
- Remediating AI Applications with potential clinical performance issues
    - When an AI Application has been flagged as potentially generating poor quality results, remediation might involve:
        - Notifying IT Operations, ML Ops Analysts, and/or the Oversight Committee.
        - Running additional tests, including full re-validation where appropriate.
        - Suspending use of the AI Application by unregistering from an AI Platform and/or unsubscribing from further work from the AI Orchestrator
        - Continuing use but marking results “suspect” or “confirm manually”
        - Sending KOS objects to Image Managers for studies already processed in the last x hours to mark the results as potentially suspect
        - Flagging the application for more detailed re-assessment and/or retraining

Profile writers may consider some of these questions for further discussion:

- How are results from monitoring communicated back to the data scientist?
    - What are the differences in interoperability if monitoring is performed by:
        - the AI Application hosting institution?
        - the AI Application manufacturer?
        - the System Regulator that approved the AI Application?
        - the payor that reimburses based on observations from the AI Application?

- What are the differences when monitoring is performed from a central system, or when it is performed in a de-centralized way?
    - If monitoring is decentralized, how does it access datasets?
- Does automated monitoring follow the same AIW-I [STD-AIW-I] workflow or does it use more specialized means to manage the associated tasks?
- Is there a type of dataset called a Monitoring/Surveillance Dataset? Is it managed using the same process as described in Dataset Assembly Use Cases?
- Does monitoring happen in production environments, or does it happen in some form of shadow environment that may be used?
    - What would trigger suspension of the algorithm in routine use?
- When performance is degrading, can an AI Application be disabled, ignored, adjusted, marked, or re-routed?
- Is feedback generated by monitoring activities treated differently from feedback generated by users (in Feedback Use Cases)?

3.5 Functional Validation Use Cases

Functional Validation Use Cases encompass validation that an AI Model performs its function as intended. Functional Validation by the original developer is included in Model Creation Use Cases: Test Model. This section addresses validation by parties “external” to those developing the model.

> Note: The usage of the term “validation” in the context of medical devices differs from the usage of the term “validation” in the context of general AI model training. To avoid this confusion, and since the validation being performed in these use cases is like the functional validation performed in Section 3.3.4 Test Model, the rest of this section will describe functional validation in terms of testing the model. Similarly, the datasets here meet the definition of test datasets and should not be confused with the validation datasets in Section 3.3.3 Train Model.

A model that fails to meet required performance metrics may be sent back to Model Creation Use Cases: Train Model with additional data records that address observed deficiencies. It may also expose gaps or biases in the training, validation, and test datasets which require going back to Dataset Assembly with a revised design to obtain, annotate and organize improved datasets.

Although the use cases in this group all involve testing the model, the different parties may have different goals and methods.

These use cases assume the model has been installed at the point of testing.

3.5.1 Test Model (for Regulator)

A Data Scientist demonstrates the safety and efficacy of an AI Model and AI Application using a test dataset, to submit to a System Regulator. This activity is typically coordinated with Biostatisticians and Product Managers.

The regulatory approval process typically involves collecting additional data to meet a statistical model for accuracy.

Considerations that may have technical interoperability aspects include:

- Intended use of the AI Application.
    - Specifying the types of data that the AI Application is appropriate for (body part, imaging modality, patient population).
    - Specifying the type of AI Application – computer-aided detection (CADe), computer-aided diagnosis (CADx), or computer-aided triage (CADt).
    - Can be derived partially from the model purpose and the dataset that was used to train the model.
    - Whether the application is novel functionality or substantially equivalent to another application
        - This is indicated, for example, by the FDA and 510(k) and Premarket Approval designations
- Assessing the quality of the test dataset
    - No data records can be included in the test dataset if they have been used by the developer to train, validate, or test the AI Application.
    - The data records should contain actual, valid clinical data.
    - See Dataset Assembly Use Cases: Organize Datasets.
- Understanding test results
    - Test results should be available as both human-readable and interoperable with other systems that comprise the solution.

Profile writers may consider some of these questions for further discussion:

- What AI Model explainability capabilities could support this testing activity?
    - See explainability discussion in Model Creation: Define Model Task

3.5.2 Test Model (by Payor)

A Payor tests that an AI Model and Application addresses their business and operational requirements using a test dataset.

Payors may be focused on demonstrating the cost benefit of using AI Models to perform clinical tasks as a justification for reimbursement for their use. Payors may also have an interest in AI

Applications that are used to drive hospital workflow and measure efficiencies and effectiveness. In some health systems, payors are fully separate from the care delivery organization, in other systems they overlap so some of this testing might be combined with clinical testing.

Considerations that may have technical interoperability aspects include:

- Collecting data to demonstrate return on investment
    - Return on investment is an important metric – e.g., how much money is saved through early detection or rapid triage of clinical conditions.
    - Collecting timestamps, using a structure akin to SOLE [STD-SOLE], to convey the time particular tasks take in a clinical environment, with measurements taken before the AI Application is deployed, and after
- Associating a billing code (e.g., CPT) with the task performed
    - The scope of a billing code definition may be narrower, the same, or broader than the scope of the task performed by the AI Application, and an AI Application may be able to perform a variety of tasks.

Profile writers may consider some of these questions for further discussion:

- What AI Model explainability capabilities could support this testing activity?
- See explainability discussion in Model Creation: Define Model Task

3.5.3 Test Model (by User)

A Clinician tests that an AI Model and Application is producing accurate, appropriate results. An Oversight Committee (Radiology Leadership) reviews test results and decides whether to approve the AI Application for local use.

AI Applications are evaluated clinically using data not seen by the application before, by using an external validation dataset, using similar mechanisms described in Dataset Assembly Use Cases. Once the AI Application processes the data, the Clinician reviews the results inside a clinical application (typically cordoned off from production systems) and compares the results with ground truth to determine if the AI Model is performing correctly.

This activity may also be referred to as acceptance testing. Vendor validators may be responsible for performing testing as part of solution development processes.

Considerations that may have technical interoperability aspects include:

- Whether testing is retrospective or prospective (with a pre-annotated dataset)
    - Retrospective testing is quicker to implement but can be subject to bias if the AI result is known.
    - Testing may utilize mechanisms from Feedback Use Cases.
- Whether testing assesses AI Model performance on its own or compares clinical performance with and without the AI Model.

- Whether testing performance is broken down, for example by subject sub-population (sex, age, race, etc.), input data source (different scanners) or disease sub-type.
- Whether the Source of the AI Application being tested is:
    - the local institution.
    - commercial, for example, as a research trial
    - part of a trial program prior to purchasing a solution.
- AI Application deployment environment
    - The AI Application may be deployed in a development environment, a pre-production environment, a testing environment, or in a production environment with appropriate controls.
- Confirming technical operation of the AI Application by IT Operations and ML Ops Analysts, and local data scientists before clinical testing.
- Communicating success
    - Scoring systems and AI Application metrics
    - Identifying when an AI Application is ready for production use.
    - Finding comparable models to score against
    - How to communicate scores – coordinate comparisons and benchmarking
    - See Common Mechanics: Data Quality and Bias.
- Addressing failures
    - When clinical testing fails, an AI Application and its associated model(s) may be flagged for local tuning. See Model Creation Use Cases: Train Model.
- Incorporating Multiple reviewers
    - The result assessment from any given reviewer may differ for a variety of reasons (e.g., two clinicians looking at an imaging study with multiple findings, or where one is aware of additional factors that lead to a different conclusion in a borderline case).
    - Clinicians that have had a hand in training the model might be biased or might suffer from the same unconscious assumptions during testing that were present during training. Testing by and external “third party” might avoid those issues.
    - Results from AI Applications that differ from ground truth should be reviewed to determine if there are clinical factors or artifacts that may have influenced the difference.
    - False positives and negatives need to be evaluated for clinical significance when comparing to ground truth.
- Reasons test results may differ from test results during development

  - Different imaging protocols being used
  - Different sample sizes
  - Different data handling pathways
  - Possible de-identification effects (e.g., jittered dates)
- Iterative design
  - In active or continuous learning frameworks, a Clinician may review an AI Application and determine that further fine-tuning of the AI Model is necessary. Locally annotated datasets may be used to establish local ground truth to enable re-training. This may impact other testing steps (by regulators and payors).

Profile writers may consider some of these questions for further discussion:

- What metrics are used to compare and rank different models with the same or remarkably similar goals? See [BIB3.5.3-1] and [BIB3.5.3-2] for examples.
- Are there assumptions made when performing clinical testing if an AI Application passed regulatory testing and/or payor testing?
- Aside from AI Applications themselves, is there clinical testing applied to various components within a clinical testing?
  - For example, a model produces an inference result, and that result is converted into a DICOM-SR TID1500 object. What value is there if the “transform” is tested outside testing the entire AI Application?
  - Can standardizing test tooling be used to help assemble testing / training datasets and doing transforms from clinical repositories?
- What AI Model explainability capabilities could support this testing activity?
- See explainability discussion in Model Creation: Define Model Task
- Does testing differ if an AI Application is created within the institution doing the testing, a different institution, or a commercial entity?

3.6 Clinical Usage Use Cases

Clinical Usage Use Cases encompass the everyday use of AI Applications to perform tasks in a clinical setting.

While many of today’s AI solutions aim to assist radiologists in reading and reporting on patient exams, there are many other applications where AI can improve clinical quality and outcomes. See Section 2.1 Applications of AI in Imaging for an extensive list of example tasks.

The following subsections cover use cases partially addressed in the existing IHE profiles, AI Workflow for Imaging (AIW-I) [STD-AIW-I] and AI Results (AIR) [STD-AIR]. Since some terms from those Profiles have been used here, Figure 3.6-1 has been taken from the AIW-I

Profile and provided here to introduce some potentially useful actor and transaction concepts.

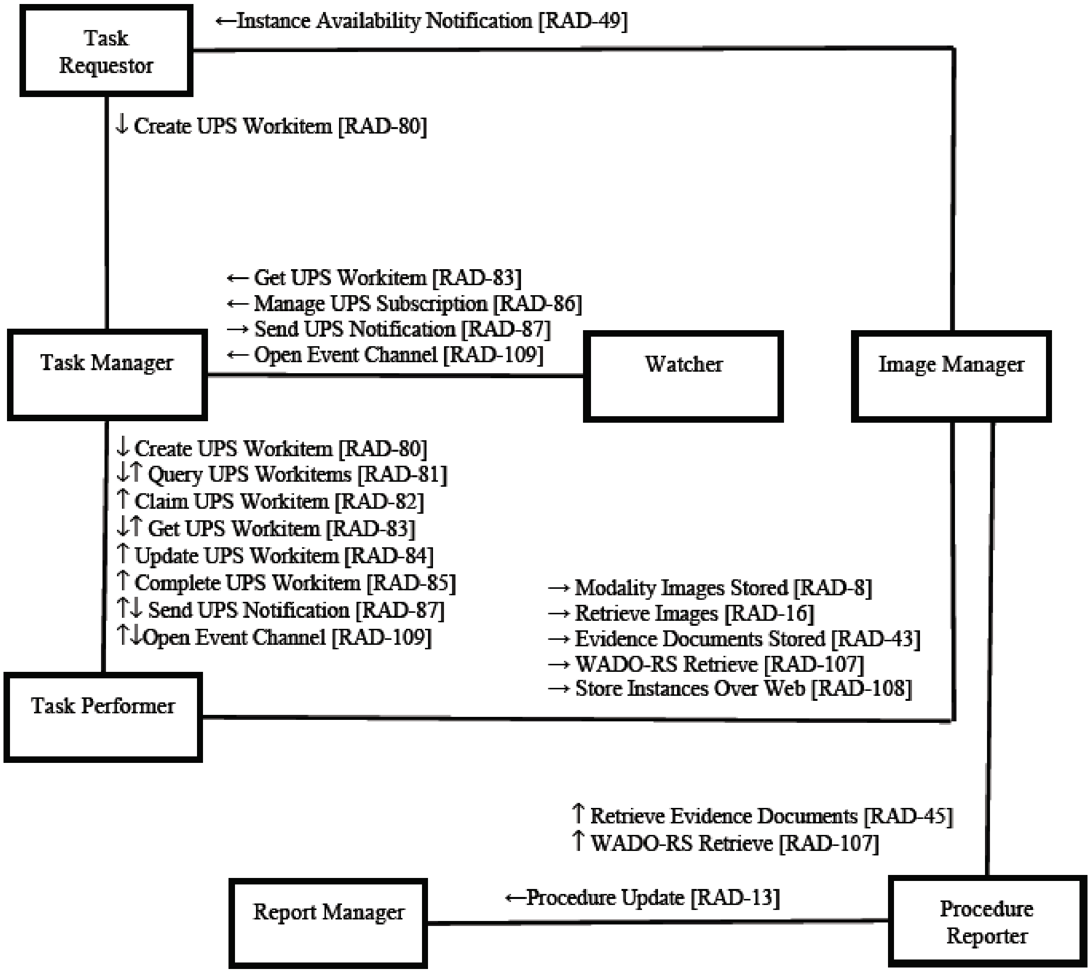

Task Requestor
←Instance Availability Notification [RAD-49]
↓ Create UPS Workitem [RAD-80]
← Get UPS Workitem [RAD-83]
← Manage UPS Subscription [RAD-86]
→ Send UPS Notification [RAD-87]
← Open Event Channel [RAD-109]
Task Manager
Watcher
Image Manager
↓ Create UPS Workitem [RAD-80]
↓↑ Query UPS Workitems [RAD-81]
↑ Claim UPS Workitem [RAD-82]
↓↑ Get UPS Workitem [RAD-83]
↑ Update UPS Workitem [RAD-84]
↑ Complete UPS Workitem [RAD-85]
↑↓ Send UPS Notification [RAD-87]
↑↓Open Event Channel [RAD-109]
→ Modality Images Stored [RAD-8]
→ Retrieve Images [RAD-16]
→ Evidence Documents Stored [RAD-43]
→ WADO-RS Retrieve [RAD-107]
→ Store Instances Over Web [RAD-108]
Task Performer
↑ Retrieve Evidence Documents [RAD-45]
↑ WADO-RS Retrieve [RAD-107]
Report Manager
←Procedure Update [RAD-13]
Procedure Reporter

Figure 3.6-1: AIW-I Actor Diagram (from Figure 50.1-1 of the AIW-I Profile)

3.6.1 Initiate Inference

A persona or system initiates inference on an inference data record.

A variety of different architectures may be applicable when considering how an initial trigger initiates data handling and execution control between one or more processes to ultimately perform inference and deliver a result. See Appendix B Architectural Considerations.

In AIW-I, inference is initiated by the “Task Requestor”. Systems that might incorporate (i.e., “be grouped with”) a Task Requestor include imaging modalities, EMRs, Report Managers, Order Fillers, or any other system needing a task performed.

Processing tasks are communicated as DICOM UPS-RS Workitems, which encapsulate references to input data, the type of task to perform (encoded as a processing procedure code), and optionally references to data element access locations, and the priority of the request (see [STD-AIW-I], transaction [RAD-80]).

In AIW-I, the Task Manager brokers tasks from Task Requestors to Task Performers. In a "pull model", the list of unclaimed tasks on the Task Manager are monitored by Task Performers which claim those appropriate to their capability. In an "orchestrated model", the Task Manager assigns tasks to specific Task Performers and to do so it is necessary to know all the performers that exist, know what tasks each can do, know that the input data record for the current inference task is adequate for, and accessible by the performer, and know the hardware resource needs of the performer. Mechanisms to support the Task Manager getting such information (referred to as "service discovery") are being explored and developed at the time of writing [STD-DICOM-DISCOVERY]. Configuring and initializing such mechanisms will be part of Section 3.4.4 "Install Application".

Considerations that may have technical interoperability aspects include:

- Initiation patterns (i.e., where the initiation logic resides, and what information/triggers it uses to determine when inference is to be performed). Examples include:
    - **User initiated (deliberate):** E.g., request ranking of appropriate procedures for known or suspected clinical condition, chatbot conversations, suggesting the right clinic,
    - **User triggered ("side-effect" of user event):** E.g., scheduling and prioritizing of exams
    - **Scheduled:** E.g., each afternoon predicts no-shows for the scheduled appointments of the next day
    - **Procedure triggered:** E.g., completion of a CT chest imaging procedure might trigger lung nodule detection
        - Imaging procedure completion can be complex and may involve waiting for sufficient time to elapse with no DICOM objects being added to the imaging study.
    - **Embedded function:** E.g., deep learning-based image reconstruction algorithm inside a CT, or deep learning-based irregular heartbeat rejection in a gated-acquisition mode on a scanner
    - **Event triggered:** E.g., an HL7 imaging order message might trigger a protocoling application at the RIS; or completion of one AI Application might trigger one or more AI Applications ("chaining")
        - A simple chaining use case example is when a first AI Application to segment lung nodules completes, a second AI Application is triggered to perform lung nodule classification.
        - A retrieval chaining use case might occur when an initial task to find, prefetch, and transform pre-requisite inputs (e.g., obtain and summarize key details from

        the patient's history) completes and that, in turn, triggers subsequent tasks to use those inputs.

        - More complex scenarios may involve execution "graphs" with one application providing input to multiple other applications, which are run in parallel, and then their results are all collected by another process.

    - **Nested:** E.g., a more complex pattern of cooperating applications might involve a primary AI Application reaching a point where it determines additional input is needed, and pauses its execution, triggers a secondary AI Application to perform a task to generate the needed input, and then resume when the input is available.

- Initiation conditions (i.e., determining whether the specific case merits performing inference). Some inferences may only be performed on a subset of cases.
    - The facts/details used in the condition logic will need to be communicated to where the logic is being evaluated.
    - Consider resource constraints (e.g., don't run a specific AI Application on non-indicated patients if processor loads are > 70% capacity), pricing (what incremental cost is incurred for running the inference), licensing, algorithm labelling (is the current case within the labelled use), local policies/rules (run stroke & hemorrhage detection on all unconscious patients), availability of required data elements, etc.
    - Some AI initiation choices might be built into the order, e.g., Head CT with Contrast and Stroke detection.
    - The many possible combinations of Initiation Patterns and Conditions, the variety of data that needs to be accessed and generated, and the differing granularity and variety of functions of AI algorithms has the potential to result in complex initiations with a web of dependencies.
- Determining which tasks are applicable to initiate on a given input data record.
    - For image analysis, this may be driven by requested procedure codes, imaging modality and body part or anatomical region associated with the exam, see AI Application Distribution and Management Use Cases: Integrate Application.
    - Additional information may be retrieved from other systems, e.g.,
        - Query the EMR to get the admission reason associated with a study
        - Obtain the list of radiologists on duty now and soon with their specialty/expertise to support prioritizing and assigning reading worklists
    - For tasks driven by clinical messages (such as order or report notification), this might be done by looking at the type of HL7 trigger event, or some of the specific fields (like inpatient versus outpatient, or reason for admission).
    - For the User Initiated pattern, tasks are selected by a human interacting with a UI.
- Identifying the resources in which to run the AI Application

  - See Clinical Usage Use Cases: Access Inference Data Record.
  - A “non-PHI-safe” AI Application, e.g., an app running on a wearable device evaluating the wearer’s biometric data by comparing it with other patient records in the hospital system, may need to have input data de-identified and pre-cached in a location accessible by the device.
- Communicating the task to the performing AI Application
  - In AIW-I, the details are all in the UPS Workitem. Consider adding any other information that may be useful to convey to know for the AI Application to run most effectively and efficiently.
  - In “Implicit Workflow”, the task is not explicitly communicated to the AI Application. Instead, data is pushed to the inference performer which presumes what task to perform, possibly upon inspection of the data. This is often done where the inference performer is only capable of a single task.

Profile writers may consider some of these questions for further discussion:

- How does AI Application initiation differ for applications that are not analyzing images to assist a radiologist in creating a report?
  - E.g., applications assisting image-guided interventions and surgery such as Cath lab procedures, image-guided biopsies, orthopedic interventions, or tumor ablations.
- How can inference tasks be prioritized? For example:
  - If ten tasks were submitted, 7 routine tasks, 2 urgent tasks, and 1 critical task, how are tasks performed in an appropriate order?
  - How does the availability/arrival of data (or subsets of the data) affect execution and prioritization? For example, a breast density model might average breast densities across all images in a study, but what if extra views arrive 5 minutes later? Does the model run multiple times, and if so, which result is presented to the clinician? See also data availability considerations in next section.
  - If, while performing a long-running routine task, an urgent or critical task comes in. How should it be handled?
- To what degree should series selectors be tolerant of more- or less-specific data?
  - E.g., an AI Application processes imaging studies for procedures containing “brain”, but a “head” procedure arrives (which contains brain imaging content). Would the inference be triggered?
- What is the role of IAN [STD-DICOM-IAN] in other, non-diagnostic workflows?
- What are alternate workflows / triggers that have not been considered in above use cases?
  - UPS Workitems can be created by a variety of actors as outlined in AIW-I.

    - Imaging Modality, EMR or Report Managers as initiators of AI inferences. How does scheduling and cancellation of inferences work in those cases?

3.6.2 Access Inference Data Record

An AI Application accesses the inputs that comprise an inference data record.

The inference data record, which holds the inputs to the inference, may include heterogeneous data elements (e.g., patient-specific and patient non-specific, current and prior records, or imaging and labs and medical history). The data record may be a single bundle, or the data elements may be distributed across multiple sources (e.g., the record references an image which may be retrieved from the PACS, and a lab result object which may be retrieved from a lab information system). It is possible that the inference data record does not contain everything needed for the AI Application to perform an inference.

There are many other scenarios were discovery and retrieval of additional information is needed for inference, including longitudinal imaging and patient record information, such as frequently stored in an EMR, as well as data stored outside of described healthcare data standard processes.

Considerations that may have technical interoperability aspects include:

- Accessing the data elements of the inference data record
    - Pull Model – the AI Application retrieves the data elements.
        - AIW-I assumes data elements are pulled using protocols like DICOMweb [STD-DICOMWEB], DICOM DIMSE [STD-DICOM], and XDS [STD-XDS].
    - Push Model – the data elements are pushed to the AI Application.
        - This is sometimes used in Implicit Workflow where, for example, data is stored into the “input folder” of the AI Application.
- Accessing data elements beyond the inference data record
    - Some AI Applications may be able to make use of information which might not have been included in the inference data record and may have mechanisms for exploring available data servers for relevant data.
        - E.g., obtain prior studies when tracking lesions across multiple time points. If the Task Requestor did not query the Image Manager for prior studies and include them in the inference data record, the AI Application may search for such data itself.
        - E.g., in abdominal CT studies, information about contrast administration and imaged body part(s) may be ambiguous or missing in the DICOM header.
        - Note: It can be adaptive/robust for the AI Application to search, but it requires query interfaces, more installation integration, and having the Task Requester do this means relevant priors can be pre-fetched (e.g., offline storage or another

network). If the task were triggered by the order, they could even be pre-fetched before the current study is acquired.

- Optimizing data access
    - Pre-caching input data elements to a low-latent location (such as to the disk of the machine running the AI Application) may improve turnaround times as the AI Application may not need to wait for data to be fetched
        - A coordinating application may need to ensure the entire input data record is available before triggering the AI Application.
    - Pre-caching may also be used to put de-identified data in a location accessible to a "non-PHI-safe" AI Application which is not permitted to access patient records or the hospital network directly.
- Access to data in remote repositories
    - Data which is indexed locally but stored remotely may require working through a proxy system to access the data.
    - A local data cache may simplify access by collecting potentially relevant data from remote sources and making them available using local mechanisms.
- Transforming clinical data before being passed to the AI Application.
    - Transforms may be applied to the retrieved data when, for example, an AI Application needs data in a format different than that in which it was stored or needs a subset of the data.
        - See Common Mechanics: Transform.
    - Transforms that modify clinical data (e.g., down sampling an image from 1024x1024 to 512x512, converting lab results into different units, or mapping codes between code sets) should be applied with caution and in a way consistent with the data used to train the AI Model, otherwise performance could be negatively affected.
- Retrieving information from clinical documents
    - E.g., an AI Model trying to detect breast cancer might incorporate knowledge about certain genetic dispositions, as well as relevant patient history such as previous occurrences of cancer and whether they were treated, or an AI Model for detecting different types of chest diseases may want to know what implants a patient has.
    - Clinical documents may be accessible using standard interfaces like XDS [STD-XDS] or FHIR [STD-FHIR].

Profile writers may consider some of these questions for further discussion:

- How can a "proxy archive" remain in sync?
- What is the impact of "chaining algorithms" where pixel data is converted from one format to another?

- Is it an issue if there is a gap on “notification of changes”?
- How can an AI Application’s needs for additional data be effectively delegated and communicated to the model execution framework?
- How can profiles be as simple as possible for AI performers (and shield complexity)?
- What sort of fault tolerance will be appropriate?
    - E.g., how often does a requester retry accessing the data? What sort of timeouts are reasonable? How are data elements flagged as being “optional” so that they are desirable in the inference data record, but if they are missing, the data record is “complete enough” and the inference can still proceed.

3.6.3 Perform Inference

An AI Application performs inference on an inference data record and generates outputs.

Considerations that may have technical interoperability aspects include:

- Determining where the inference will be performed and by which AI Application
    - The deployment pattern (see Application Distribution and Management Use Cases) being used will influence how an appropriate AI Application is selected and executed.
        - For Embedded deployments, the Application is predetermined and may already be running inside the parent system.
        - For a Standalone or Hosted deployment, applications might monitor a central worklist and claim tasks that are appropriate to their capabilities as they are posted.
        - For Mediated or some Hosted deployments, the application is selected by a central management process. That assignment could be communicated to the application in pull or push workflows by encoding the Scheduled Station attribute in a UPS Workitem. See [STD-AIW-I] for details.
    - Some related concepts on the conceptual layers between the initiation and performance of inference (and the associated completion and delivery of results) are discussed in Appendix B Architectural Considerations.
- Managing exceptions
    - See AIW-I [STD-AIW-I] which describes use cases for managing exceptions, e.g., rejecting, canceling, or re-assigning AI tasks using DICOM UPS Workitems, and notifying observers about error conditions. However, specific codes for common error conditions would aid in troubleshooting / resolution processes, e.g., “Input image did not meet AI specs”, “DICOM file corrupt”, “Could not access input data”.
    - See SOLE [STD-SOLE] for standardized workflow codes. See DICOMweb Error Codes [STD-DICOMWEB-ERRORS]

    - Some scenarios might require different mechanisms than DICOM UPS, e.g., a configuration to determine if and how many times an AI Application should retry processing a data set.
- Proxying performing of tasks
    - Instead of AI Models being wrapped by a Task Performer they could be proxied by a separate Task Performer (see AIW-I Profile, use case 4 in Section 50.4.2).
        - This allows for integration of data-only models packages (see Application Distribution and Management Use Cases: Package Application) into the AIW-I workflow without having to wrap it into a fully functional Task Performer.
        - In this case, each Task Performer could be dedicated and configured for a particular AI Model or a small set of AI Models, or it could be designed to act as a general-purpose Task Performer that can be configured for all kinds of models.
- Evaluating and reporting inference metrics
    - This may be valuable for troubleshooting and evaluating ongoing needs of updating AI Applications.
    - Data to capture includes model identity (manufacturer, appropriate use of model), model processing metadata (confidence, explainable AI), and model application state (failure behavior)

Profile writers may consider some of these questions for further discussion:

- What could be a good interface standard for how Task Performer proxies delegate work to AI models and how can they be chained together by a proxy?

3.6.4 Create Result

An AI Application encodes the output from an AI Model for consumption by the ecosystem.

Choices about how the results are encoded are typically influenced by the ways and systems in which they are used (see Clinical Usage Use Cases: Use Result).

Considerations that may have technical interoperability aspects include:

- Formats and encoding of results
    - For image analysis results, AIR identifies types of result data elements (“Primitives”) and specifies corresponding DICOM object encodings for each.
        - See “Results” under Entities: Data Element
    - Other results will likely be encoded in non-imaging standards like FHIR and HL7.
- Standardized (or at least converged) code sets
    - E.g., SNOMED-CT, RadLex, LOINC, Common Data Elements for Radiology (RDE), (see also AIR [STD-AIR], Section 49.4.1.4 “Code sets”)

- Explainability of results
    - Where relevant, AI Applications should not only convey the result ("this is pathological condition X"), but also provide evidence and rationale why. This may build clinician trust and provide data useful for troubleshooting. See Section 3.3.1.
- Result Confidence
    - Results may include a likelihood the result is correct or a confidence interval (range) for the result value.
    - In some cases, the result itself is a probability.
    - A calibration plot or calibration curve may be a useful way to communicate how the result of a classifier should be interpreted.
- Usability of result
    - Context of the result creation from the AI Application should be provided.
    - If the input data records provided to the AI Application was de-identified for some reason, some system will likely need to reliably re-identify the output data record for the result to be usable and properly integrated into the medical record.
- Reusability of results
    - Using results from one AI Model as data elements for training, validating, or testing another AI Model.
    - Results that have been flagged and/or edited as part of the Feedback process will also be used as inputs for subsequent training. See Feedback Use Cases.
- Multiple data element results
    - Applications may have multiple output data elements, e.g., a segmentation of the lung, several findings, an impression of pneumonia, and a heat map indicating the parts of the image most influencing the findings.

3.6.5 Deliver Result

An AI Application delivers the result(s) of the inference to their point of use. In the event of a failure to deliver results, ML Ops Analysts and Product Support Teams may be involved.

The destination depends on the context and type of analysis performed. It can be a DICOM archive, as described in AIR [STD-AIR] and AIW-I [STD-AIW-I], a FHIR-based repository, or any other information system or database. In AIW-I, this is the Task Performer storing the result to a target location that may be specified in the UPS Workitem.

Considerations that may have technical interoperability aspects include:

- Delivery mechanisms

    - For image analysis results (see Section 2.1.5), AIW-I and AIR specify the use of DICOM C-STORE or DICOMweb [STD-DICOMWEB] STOW-RS (transactions [RAD-43], [RAD-18], [RAD-29] and [RAD-108]).
        - If the specified target is an archive, other systems can query for those results. E.g., a DICOM Viewer could display results to a radiologist during image interpretation.
    - For non-DICOM output, e.g., a FHIR-encoded report [STD-FHIR-DXR], further standardization is needed (see also Clinical Usage Use Cases: Create Result).
    - For results of hospital tasks such as Ordering and Scheduling (see Section 2.1.1) or Patient Management and Treatment (see Section 2.1.7), the results will likely need to be encoded in HL7 messages or FHIR resources.
    - Inference results may need to be transcoded to be suitable for consumers.
        - This could happen before saving them to the manager/archive or upon retrieval.
    - For embedded AI, this step may not be needed since the inference result is used on the system where it is created.

Profile writers may consider some of these questions for further discussion:

- How are image analysis results encoded when consumed by non-DICOM-aware systems? How can they be transformed?
    - It is expected that future reporting systems will typically produce and consume data in FHIR format. EMR's currently use HL7 messages extensively. DICOM Working Group 20 is investigating transformations of DICOM SR content into FHIR Observation representations. Once it is decided which information needs to be transferred, effort will be required to maintain semantic equivalence of the information that is transcoded.
- For AI used in the context of interventional procedures and surgeries, will likely need to deliver its results to one of the systems used in the operating theatre, and only afterwards be sent to a DICOM Archive. More generally, what other forms of distribution of Inference Results are required in those scenarios?
- When humans and AI Applications perform the same task, for example both assess an image for pneumonia, how are “human-generated results” and “AI-generated results” combined and routed through the end-to-end workflow?
    - E.g., preliminary (“wet read”), blinded-reads, reads direct from imaging modality, resident reads, etc.
- More generally, what other forms of distribution of Inference Results are required in those scenarios?

3.6.6 Use Result

The user applies the AI inference result to the relevant task.

The user in this use case may be any persona for whom the results of the applications identified in Section 2.1 Applications of AI in Imaging are relevant: e.g., Radiologist, Radiology Leadership, Clinician, and Health Information Management Teams.

In the "daisy-chained" execution case, the result of one AI model may be used as the input to the next AI model, which might even be the only user of that result. E.g., the result of a lung nodule detection model might be the seed point fed into a lung nodule segmentation model, which in turn is the information used by a lung nodule classification model.

IHE AIR [STD-AIR] and AIW-I [STD-AIW-I] address a subset of the applications listed in Section 2.1. AIR is focused on the case where an Image Display presents image analysis results and imaging studies in an integrated manner to help the radiologist with the interpretation of images. For example, suspicious areas in an image may be outlined or highlighted or an identified lesion may be characterized in detail. AIW-I also covers the scenario where image analysis inference results are used to help prioritize worklists.

Considerations that may have technical interoperability aspects include:

- Levels of AI Autonomy
    - At one end of the spectrum, the AI Result is only provided to inform a human performing the task, at the other end of the spectrum the AI Result is applied automatically, and the task is performed with no human oversight.
- Presentation of AI Results
    - Depending on the task and the level of autonomy, the AI Results and relevant contextual information will need to be presented effectively to humans. The specific metadata elements that would be of interest to the human assessing and/or using the AI Results will largely depend on the nature of the result and the task the human and AI Application are performing together.
    - In the case of image analysis results, the AI Result may be presented as an annotation, appearing like manually created annotations.
    - Many encodings developed for robust storage, e.g., DICOM SR, leave presentation to display systems which some find challenging. The presence of multiple related AI Results can add to the challenge. Profiling should consider addressing this.
        - IHE AIR suggests some approaches (see Section 49.4.1.3 Result Presentation) but did not have time to flesh them out.
- Human approval and override
    - Depending on the level of autonomy, the AI Results will need to be approved or overridden by a human.

        - Need a mechanism for the human to indicate agreement or disagreement with the AI Result and in the case of disagreement, to indicate the "correct" result. Consider both GUI and voice-based mechanisms.
        - Need to record/store such agreement or disagreement (and correct result) - see also Feedback Use Cases

- Tracking activity and performance
    - The results, the outcome of the task the result affected, and the interactions between the human and the results may need to be logged to support retrospective review and performance evaluation.
- Applying the result to the intended task
    - See Section 2.1 Applications of AI in Imaging
- Realtime Interaction between the AI Application and a user
    - Some AI Applications generate results in advance for use or presentation to a user. Even with a single "verification" step, is a single sequential pass rather than interaction.
    - Other tasks involve direct interaction with a user.
        - A radiologist might provide a seed point in an image and invoke an AI Model to segment the underlying lesion. Similarly, the radiologist may disagree with a pre-generated segmentation and interactively adjust it.
        - The radiologist may segment a portion of the image and invoke an AI Model to characterize the region as malignant or benign.
- Communicating the result to systems performing subsequent tasks such as reporting.
    - Profile Proposals have been submitted in IHE Radiology to integrate the various systems involved in composing a radiology report and contributing data elements for incorporation into the report.
    - Some Reporting systems can use FHIR [STD-FHIR] resources generated by AI Application. Automatic conversion or mapping of information captured in DICOM objects to FHIR objects would be helpful, as some AI Models may output DICOM and not FHIR.
    - Results of one AI Application may also be used as inputs for another AI Application or some other type of analytics application.
- Operational effectiveness can be measured and improved by gathering information that feeds into metrics like in the following examples:
    - Capture turnaround time for AI Model inference to answer the question if the AI Application is performing fast enough to have maximum impact.

- Capture user interactions with AI Model results to measure how often user agrees or disagrees with AI Results.
- Capture what percentage of studies are processed by a model, and what percentage is rejected and for what reason.

IHE is extending the use of the SOLE [STD-SOLE] Profile to allow logging of events for the above-mentioned purposes.

Profile writers may consider some of these questions for further discussion:

- What AI Model explainability capabilities could support effective use of these results?
- See explainability discussion in Model Creation: Define Model Task
- Many of the applications in Section 2.1 outside of the radiologist's reading workflow use AI inference results in quite different ways. Those need to be explored further considering the considerations listed above.

3.7 Feedback Use Cases

Feedback use cases encompass getting information on the performance of an AI Application during routine clinical use, distributing the feedback, and, in the case of a neural net-based AI Model, potentially re-training it.

Note: Broader "feedback" about the application, such as its usability or the time it takes to launch, are out of scope here. This section is specifically focused on the "correctness" of the results produced by the AI during use. Similarly, automated monitoring of the AI Application is discussed in Section 3.4.7 Monitor Application Service, not in this section.

Since the deployed AI Model is packaged in an AI Application, in this section the terms may be used interchangeably.

3.7.1 Collect Model Feedback

A Clinician (or an AI Application user, such as the healthcare information management team or the Payor) provides feedback on the performance of an AI Application in routine usage.

Feedback will identify cases where the AI Application did not perform adequately, and cases with good performance. The latter is particularly important to gather evidence for broadening the scope of use of the model or during surveillance (see Application Distribution and Management Use Cases: Monitor Application Service).

Considerations that may have technical interoperability aspects include:

- Why feedback on an AI Application is being collected.
  - To confirm the AI Application continues to function appropriately, especially after an identifiable change such as an algorithm update
  - To detect poor result cases to better understand the boundaries of robust operation of the Model and to potentially drive improvement of the AI Model or AI Application.

- Content of feedback
    - Identity of the AI Application
        - See Common Mechanics: Provenance
        - Consider how the feedback mechanism collects this information (and the following) E.g., is the model identity extracted from the metadata of the result?
    - Identity of the Observer (e.g., clinician)
    - Identity or characterization/description of the case/input data record
        - May include order information, clinic location, demographic information, relevant imaging studies (possibly including raw data), care or treatment plans, raw data, reports, and/or observations.
        - This may be pointers to the data or include the data itself.
        - This data potentially could be added to repository and added to a dataset for inclusion in future training.
    - Original result (i.e., what the result was) and confidence metric
    - Expected "correct" result (i.e., what the result should have been)
        - May be provided as an output data element or a description of the expected result
        - May also include how the "correct" result was determined
        - The expected "correct" result may be the same as the original result, confirming the success of the algorithm
        - See Common Mechanics: Annotation
    - Assessment of the result produced (correct, incorrect)
        - Include additional detail where relevant such as degree of agreement, a confidence score, or other supplementary information
    - Severity of discrepancy
        - E.g., whether the difference is minor, or whether it would potentially result in a change of diagnosis or treatment
    - If a disagreement or failure occurred, assessment of the cause or type of failure
        - False negatives, false positives
        - Presence of some unusual/unexpected data or detail "distracted" the model. E.g., an unintended biological structure or medical apparatus was detected instead of the intended feature
        - Absence of some usual/expected data or detail hindered the model
    - The content may vary depending on the intended feedback destinations

        - Included data records alongside feedback: ranging from feedback only, all the way to the complete study with findings as created by the AI Application and corrected data elements.
        - E.g., regulatory bodies might require clinical data related to demonstrating safety and effectiveness; an Oversight Committee might want data from multiple feedback cases summarized by certain patient categories
        - See also Feedback Use Cases: Distribute Feedback
        - Data normalization may be needed prior to circulating feedback.
        - When feedback is being provided outside the organization, some elements may need to be de-identified. See Common Mechanics: De-identification

- Handling feedback from multiple sources
    - Determining if the feedback is concordant or discordant
    - Addressing discordant feedback, e.g., take the majority, consider relative experience of observers, apply common discrepancy resolution procedures, permitting discordant feedback in the retraining, etc.
- Impact of multi-step processing pipelines
    - When the result is produced by a multi-step processing pipeline, capturing the intermediate results and each intermediate output could be beneficial to check that the results are in the expected range.
- Assessing feedback before it is communicated (e.g., QA checks of the feedback, adjudication of feedback, discrepancy resolution between two readers and AI).
    - Feedback adjudication may be based on confidence measures using validation metrics derived from confusion matrix
    - For discrepancy resolution between AI and two readers, measures such as Intra Class Correlation (ICC) could help to evaluate the individual performance.
    - Leverage root-cause analysis to assess feedback
        - E.g., AI was not expecting to see a chest tube and thus was misinterpreted; or the AI Application reacted poorly to burnt in demographics
        - Root cause rationale may enable data scientists to make theories as to why the AI Application performed the way it did and may suggest additional measures to take (e.g., what additional data to gather) to improve performance of the AI Application.
    - Identify potential reasons why the AI Result was incorrect. For example, reasons an Image Analysis task failed might include:
        - Patient donated an organ and is entirely missing a relevant structure.

        - Patient had features that the model is not trained on (disease specific variations, e.g., running AI on a patient with hypertrophic cardiomyopathy)
        - Patient had surgery to address the condition being detected/assessed.
        - Patient had devices, lines or tubes that interfered with the analysis.
    - Feedback results in a new training record and "data completeness" may overlap with the design of the training dataset and should be considered.
    - Diagnostic reports might contain additional ground truth information and could be mined and measured against what the model predicts.
        - This could be "structured" checks – e.g., taking measurements from radiologist-generated DICOM SR and comparing against the generated AI Results in DICOM SR. FHIR Diagnostic Report resources [STD-FHIR-DXR] may also be mined for findings.
        - This could be "unstructured" checks with NLP – e.g., understanding the report and comparing against AI classification (e.g., AI reports presence of pneumonia, but the final report makes no mention of pneumonia)
- De-identification of feedback
    - See Common Mechanics: De-identification
    - Feedback may very often be generated from clinical usage, meaning the input data record and result data is fully identified.
    - Since feedback cases are clinically "current", it may be especially important for any required de-identification mechanisms to consider a method for associating feedback review/outcomes back to clinical records if notifying the patient/clinician of new information is appropriate.
- The encoding in which feedback is communicated.
    - Granularity of the feedback (e.g., the model failed on this case vs manually added segmentations and updated measurements)
    - Associating the feedback with a concrete finding (e.g., on which specific lung nodule did the model fail)
    - The encoded feedback entity (See Entities: Feedback) may contain an encoding of the "correct result", or may reference another object which encodes the "correct result"
    - Encoding the "correct result" the same way as the rest of the annotation data elements in the training datasets would facilitate incorporation in future training or testing.
        - The SR encoding from the IHE AIR Profile [STD-AIR] may be an appropriate format.
        - See Common Mechanics: Annotation

- Criteria in which to suspend usage of an AI Model.
  - See Application Distribution and Management Use Cases: Monitor Application Service.

Profile writers may consider some of these questions for further discussion:

- Do we differentiate between "real-time" versus "after the fact" (audit)?
  - Real-time feedback may be able to collect additional data that is available at "runtime". After-the-fact feedback may be able to include further analysis of the case or more accurate information (e.g., the biopsy result for an analyzed tumor).
- How does this relate to peer review / over reads / double reads? Can we use similar process patterns?
- Should a similar process be followed for providing feedback that is not intended for re-training models, but instead used for other purposes?
  - E.g., this may include system uptime details, error reports, throughput, product feedback on usability of the results, cases where the input data did not match to intended use of the AI Application, etc.
  - This may also include assessments by the payor, regulatory bodies, users, oversight committees, or the vendors themselves.
  - Could some of this collected feedback be aggregated and used to generate performance metrics of the AI Application? E.g., sensitivity / specificity / DICE / statistics on measurements, etc. See Functional Validation Use Cases.
- How can the model creator inform the feedback source about details specific to their model that should be collected during feedback?
  - Consider the fundamental inputs and outputs determined during Model Creation Use Cases: Define Model Task).
  - IHE QRPH Structured Data Capture (SDC) [STD-SDC] Profile may provide some insights.

3.7.2 Distribute Feedback

Feedback is communicated to necessary stakeholders, which may be filtered through a Data Scientist or automated. ML Ops Analysts monitors incoming feedback in the event of potential application failures. Product Managers and Data Scientists use feedback to make determinations about whether to adjust AI Models.

Considerations that may have technical interoperability aspects include:

- What to distribute
  - See Feedback Use Cases: Collect Model Feedback for the content of the feedback itself.

- Feedback bundling
    - Whether feedback submission is occurring on a case-by-case basis or as batch submission
- Feedback destination
    - Feedback may be distributed, in part or in whole, to multiple destinations. Depending on the destination, some information may be removed, or de-identified.
    - The identity of the AI Model is part of the provenance details and would facilitate distributing the feedback to the original model developers
    - Performance and outcomes information may be distributed locally to the people or systems managing the AI Application
    - Feedback could be added to a repository and assembled into a dataset for future training / tuning of AI Models
    - Some feedback might be appropriately communicated to actors involved in Functional Validation and Surveillance
        - See Application Distribution and Management Use Cases
- Feedback alerting thresholds.
    - Significant negative feedback may trigger alerts or alarms or discontinue use of the Model (either locally at the reporting site, or across all sites where the algorithm is in use) pending review/remediation.
    - This might not be needed if doing feedback inside the system that has the model and doing continuous re-training.
- Storage of feedback
    - Data records associated with feedback may be contributed to an appropriate repository (see Repository Use Cases: Contribute Data).
- Obtaining and distributing patient consent for feedback payloads where needed
- Push vs pull feedback distribution mechanisms.
- Billing and licensing implications
    - Incorrect or disputed AI Results may impact billing.
    - Efforts for providing feedback (positive or negative) may be billable.

Table 3.7.2-1 provides some examples for who feedback may be distributed to, what feedback data may be distributed, and how.

Table 3.7.2-1: Example Distribution Patterns

Destination/Recipient	Distributed Data	Distribution Mechanism
Data Scientist	Full data	AI Application logs
Oversight Committee	Summary data	System dashboards
Product Support Team	De-identified summary data	HTTPS for each event
System Regulator	De-identified roll-up data	HTTPS in batch

Profile writers may consider some of these questions for further discussion:

- What is the mechanism to inform the model developers about negative performance?
- Which repository is the recipient of the feedback?
- How does the feedback get conveyed from the point where it is created to where it will be used for re-training?
    - Does the imaging study get captured and stored back, or just the "yes/no" feedback? Is there an in-between (e.g., maybe the model only cares about the "with contrast series" and that is the only thing that we captured?)
    - In addition to yes/no feedback it could be helpful to train the radiologist to fine tune certain parameters in a tolerance range.
    - Further information described in Feedback Use Cases: Collect Model Feedback like intermediate results in a multi-step pipeline, validation metrics, and more specifically the Radiologist approved predefined checklist could be helpful to the model developer.
- In addition to circulating feedback to the model creator, should it also be circulated relevant repositories? Whose repositories (e.g., a local repository? A remote repository? A temporary repository until remote data scientists has a chance to review it?)
    - When feedback is going external, it likely needs to be de-identified
        - See Common Mechanics: De-identification
    - How to encode and communicate details that are important to maintain?
    - If feedback is not being shared externally, should there be a feedback availability notification message?
- Does collection and circulation of feedback overlap with details collected in IHE Teaching Files and Clinical Trials Export (TCE) Profile [STD-TCE]?

3.7.3 Adjust Model

A Data Scientist incorporates the feedback into data records in a training dataset and retrains the model.

Data records that are flagged as diverging from the AI Model can be added to a Dataset (see Dataset Assembly Use Cases: Obtain Data Record), with ground truth data elements identified, that can be sent back to the Obtain Data Records step and a decision might be made at the Organize Datasets step whether to use the feedback data record into training, validation, or testing. Models can then be fine-tuned or re-trained as described in Model Creation Use Cases: Train Model.

Considerations that may have technical interoperability aspects include:

- Applying updates to the AI Model
    - Re-training (to a larger or smaller degree) based on the new feedback.
    - Active (or continuous) learning occurs when data records are immediately fed into a re-training or fine-tuning step. Active learning may not be appropriate for some AI Applications, as changing models may impact the performed functional validation.
- Re-validation and re-distribution of updated AI Application / AI Model
    - Mechanisms to take an updated AI Model into production use may follow similar or same use case patterns as described in Application Distribution and Management Use Cases.
    - Re-training may require re-validation review, from clinicians, regulatory bodies, and/or payors. See Functional Validation Use Cases.
- Feedback as additional ground truth:
    - In Dataset Assembly Use Cases: Refine Data Record and Annotate Data Record, feedback payloads may look different than originally provided ground truth. It is up to the data scientists and model training process to reconcile how to interpret ground truth.
- Update model provenance as model adjustment is performed.
    - See Common Mechanics: Provenance

Profile writers may consider some of these questions for further discussion:

- Would requests to update models from Feedback Workflow be triggered via UPS-RS?
- How frequently would model adjustment occur? After each feedback record? After 100 records? After a certain number of bad results? How confident was the AI in the bad result (if low confidence, might not need to retrain)? How severely incorrect were the bad results? Does the user opinion of the severity get factored in?
- What kind of "data quality" and provenance checking do we expect the Data Scientist to perform on the received feedback records?

- What AI Model explainability capabilities could support this analysis and adjustment activity?
- See explainability discussion in Model Creation: Define Model Task
- Are there differences, when the model is adjusted, if the feedback was human-generated (e.g., through agreement or disagreement), or machine-generated (e.g., an NLP review of the final report detected that no mention of a flagged classification identified by an AI model)?
    - Does the cadence of re-training change when it is human-generated or AI-generated feedback?
    - Is the provided feedback/case assessed for suitability for re-training of the current AI Model?
    - Is there assessment that feedback that has been provided improved the performance of the model? To what extent should we keep adjusting the model?

3.8 Common Mechanics

This section describes capabilities that apply in multiple Use Case sections above. Because the active persona can vary depending on the "parent" use case, the initial descriptive sentence uses the passive voice here, rather than the active voice used above.

When addressing these "sub-use cases" in profiles, it will be important to consider the benefits of using common mechanics for these capabilities. Doing so effectively, and designing a robust mechanism, will depend on recognizing the multiple use cases to which the sub-use case applies and the different ways a selected mechanism would be used. The various applications can be found by searching the Use Cases sections for the name of the Common Mechanics section since they have been referenced as appropriate.

3.8.1 Data Access

Data is located and retrieved to populate data elements.

Often, this data exists in operational healthcare systems such as EMRs, PACS, VNAs, and laboratory information systems. Due to the breadth of sources and types of data, a variety of data access protocols will likely be needed. At the same time, encouraging some convergence where practical will reduce the cost of implementation and potential interoperability issues.

Considerations that may have technical interoperability aspects include:

- Data element encoding and transport. Example elements and standards include:
    - Imaging
        - DICOM DIMSE [STD-DICOM]
        - DICOMweb [STD-DICOMWEB]
    - Annotations

    - See Common Mechanics: Annotation.
  - Patient medical records and demographics
    - HL7 v2 [STD-HL7] ADT and ORU
    - FHIR Patient [STD-FHIR-PT] and Observation [STD-FHIR-OBS]
  - Reports
    - HL7 v2 [STD-HL7] ORU
    - FHIR DiagnosticReport [STD-FHIR-DXR]
    - DICOM SR [STD-DICOM-SR]; Indicate structured versus unstructured reports
    - Clinical notes (e.g., discharge summaries, nursing notes)
    - Radiation Dose Recording [BIB3.8.1-1]
  - Schedules (timebound) of entities/resources, such as scanners, rooms, staff, and patients
    - FHIR Appointment [STD-FHIR-APPT] and Schedule [STD-FHIR-SCHED]
  - Worklists (order and priority bound) such as reading and processing worklists.
    - DICOM MWL [STD-DICOM-MWL]
    - DICOMweb UPS-RS [STD-DICOMWEB-UPSRS]
  - Orders
    - HL7 v2 [STD-HL7] ORM
    - FHIR ServiceRequest [STD-FHIR-SVCREQ]
    - DICOM MWL [STD-DICOM-MWL]
  - Procedures and Care Plans
    - FHIR Procedure [STD-FHIR-PROC]
    - HL7v2 [STD-HL7] OMG
    - DICOM MPPS [STD-DICOM-MPPS]
    - RadLex [STD-RADLEX]
    - FHIR CarePlan [STD-FHIR-CAREPLAN]
  - Protocols
    - DICOM Procedure Protocol [STD-DICOM-PROTOCOL]
  - Patient Diagnoses and Conditions
    - HL7 v2 [STD-HL7] ADT and ORU
    - FHIR Condition [STD-FHIR-COND]

        - ICD [STD-ICD]
        - SNOMED [STD-SNOMED]
        - LOINC [STD-LOINC]
    - Available equipment / modalities
        - DICOM DNS Service Discovery [STD-DICOM-SVCDISCOVER]
        - DICOM Modality Worklist [STD-DICOM-MWL]
    - Appropriateness criteria
        - e.g., updated regularly from authoritative source to inference products.
        - Lung-RADS [STD-ACR-LUNG], BI-RADS [STD-ACR-BIRAD], etc.
    - Appropriateness recommendations
        - IHE CDS-OAT [STD-CDS-OAT]
    - Practice guidelines / clinical pathways
    - Audit Events
        - IHE ATNA [STD-IHE-ATNA]
        - IHE SOLE [STD-IHE-SOLE]
    - Consumer / Public Services
        - Traffic and weather conditions / forecasts, e.g., impact to no show prediction, icy conditions for increases to ER admissions.
        - Taxi ordering and availability for patient transport.
        - Census data
        - Public health data
    - Anatomy atlas
        - FMA [STD-FMA]
    - Care team members, with credential / role / job and contact information.
        - IHE DCTM [STD-DCTM]
    - Billing transactions
        - CPT [STD-CPT]
            - Procedure costs (including costs for running inferences)
- Common codes and value sets reduce the need for transcoding and support common understanding when data is accessed in multiple places
    - See also the FAIR criteria relating to Interoperability [BIB3.1.2-1]

- Self-description of data
  - E.g., when extensions are needed, or transforms are performed.
- Security implications (authentication, authorization, and access control)
  - See also 3.8.3 De-Identification
- Data transforms
  - Data is represented in a specific format and need to be converted to another format prior to use.
  - See Common Mechanics: Transforms.

Profile writers may consider some of these questions for further discussion:

- Should access/retrieval also address query/discovery, or is that better handled as a separate service?
- In each use case, will systems need to support push or pull or both?
- When using data, does the user keep a persistent copy (which raises synchronization questions) or does it discard and re-retrieve if/when needed (which raises versioning questions)?
- Are there implications in using live clinical data?
  - For example, retrieving a sizable volume of records in the middle of a busy clinic may negatively impact the performance of clinical operations. In addition, taking clinical data before it has been finalized (e.g., additional DICOM series being acquired and delivered, or corrections to a DICOM series being re-sent) may result in inaccuracies.
- Does a data record incorporate data element values or include data elements by reference?

3.8.2 Exchanging Datasets

The content of an entire dataset is accessed or moved from one location to another.

Considerations that may have technical interoperability aspects include:

- Encoding of the data elements
  - See Common Mechanics: Data Access.
- Encoding of data records (bundle of elements or vector of references to elements)
  - Data records will need to handle data elements encoded in different formats in the same data record.
  - Some use cases may call for some data elements in a data record to be included by reference/access path rather than incorporating the data directly.

- Packaging and bundling of datasets.
- Protocols for exchanging datasets.
  - Protocols used for accessing individual data elements or records should be adequate for accessing moderately large datasets.
  - DICOM Sup223 Inventory IOD and Services (public comment draft as of 2021.04) provides inventory of very large sets of DICOM content to facilitate efficient migration. See [BIB3.8.2-1]. It should be noted that research such as that described in this white paper is identified as a secondary use case.
  - IHE XDS [STD-XDS] handles cross-enterprise sharing of data elements and datasets
  - FHIR Bulk Data Access (Flat FHIR) [early draft]. See also FHIR Import/Export.
- Caching datasets versus on-demand access
  - Use cases may call for protocols to handle both push and pull, streaming and batch transfer, synchronous and asynchronous, pre-cached and on-demand transfers
  - Keeping a persistent copy may create the need for synchronization, whereas discard and re-retrieving data if/when needed may create versioning issues.
- Security implications (authentication, authorization, and access control)
  - See Common Mechanics: De-Identification.
- Consent management
  - Individual contributed data records may be subject to consent requirements.

Profile writers may consider some of these questions for further discussion:

- Does a dataset incorporate data records or include data records by reference?
- At what point does the size of the dataset raise new considerations?
  - Is there a need to exchange/access part of a dataset?
  - Should we consider the ability to exchange all the data elements for some but not all the data records? Which end controls which records are included in each part?
  - Should we consider the ability to exchange all the data records with some but not all the data elements? Which end controls which elements are included in each part?
  - Should we consider the ability to exchange both some of the data elements for some of the data records?

3.8.3 De-identification

Identifying information is modified or removed from data elements, data records or datasets.

The data used in AI is often data from real patients. To protect the identity and privacy of the patients, information which could expose their identity to individuals outside their chain of care needs to be processed so that is no longer reasonably possible. The words anonymization and pseudonymization are sometimes also associated with this process.

Considerations that may have technical interoperability aspects include:

- Determining what de-identification is needed based on where the data is being sent and for what purpose.
  - In clinical use inside a hospital, patient data remains identified. If using onsite AI Models, there might be no need for de-identification.
  - Arrangements like HIPAA BAAs (Business Associate Agreements) engage a hospital partner in the responsibility to protect identified patient data and data sharing between the Associates might not require de-identification.
  - If the security of the data sent to an external service or device cannot be protected, or to increase the level of security, the data might be de-identified before sending to the external service.
  - Training and testing of AI Models is typically outside of the individual care of specific patients and so de-identification is generally the rule.
  - Data being stored and published in repositories is likely expected to protect the privacy of the subjects of the data, depending on consent, terms or use, and/or local regulations and policies.
- Determining when (in the pipeline of Use Cases in Section 3) de-identification should take place.
  - The details that need to be retained for processing are often dependent on the model and its selected task, so preemptively deidentifying data in the repository may be less functional.
  - De-identification may be a multi-phase process occurring before data is contributed to a repository, as part of repository curation, during retrieval from a repository, as part of dataset refinement, and/or when accessing the inference data record.
  - Conversely, certain training methods that involve sending the model to the source of training data where the data "custodian" administers the training step might not require de-identification of the training data at all.
  - Such decisions may require discussion and agreements between stakeholders such as the Repository Administrator(s), the data scientist(s) assembling training datasets, the data scientist(s) developing the AI model, and Oversight Committees relating to the policies of the institution(s).
- De-identification mechanisms
  - Information such as demographics that has been "burnt into" images may need to be detected and obfuscated

    - Facial or other identifying features that may be recognizable in surface renderings of 3D CT or MR scans of the patient's head may need to be obscured.
    - Identifying information that has been replicated into free text metadata fields (such as comment fields in DICOM headers) may need to be detected and removed.
    - Some UIDs may be generated using algorithms that encode the date into the UID string.
    - See DICOM Basic Application-Level Confidentiality Profile [STD-DICOM-DEIDENT] for relevant information and named options such as the Clean Pixel Data Option, the Clean Recognizable Visual Features Option, the Retain Patient Characteristics Option, and the Retain Longitudinal Temporal Information Option.
- Handling the need to coordinate de-identification across multiple data elements or data records.
    - E.g., by using the same artificial patient ID for a given patient so that a pre-treatment record and a post-treatment record can be linked.
    - Exact dates can often be changed, to obscure identification, but the relative time between dates may need to be preserved if that is relevant to the AI task. Coordinating such "preserving changes" can be challenging if it needs to be done at different times and the number of different related records is large. See related birthdate discussion below.
- Considering whether there is a need to later re-identify the patient.
    - E.g., by maintaining a protected lookup table so that a patient can be notified if an unexpected health problem is uncovered during use of the data.
- Finding a way to maintain useful provenance despite performing de-identification
    - A trade-off of performing particularly complete de-identification is that it may result in duplicate records being included in a dataset, or allow the same record to be included in both the training and test dataset, when the same record is encountered in two different sources but was not identifiable as the same record.
- De-identification should be performed in a deliberate fashion, ideally based on relevant specifications.
    - DICOM Basic Application-Level Confidentiality Profile [STD-DICOM-DEIDENT]
        - Using associated options such as retaining patient characteristics or retaining longitudinal temporal information.
    - IHE White Paper analysis [BIB3.8.3-1]
    - Health Insurance Portability and Accountability Act (HIPAA) [BIB3.8.3-2]
    - General Data Protection Regulation (GDPR) [BIB3.8.3-3]
- Open-source and commercial tools exist to support de-identification.

  - Some providers have imaging datasets with artificial identities that can be used to test/evaluate de-identification tools and processes against their recommended results.
  - An AI Model could be trained to obfuscate burnt-in demographics.
  - See Appendix D.4: Reference Toolkits.

Profile writers may consider some of these questions for further discussion:

- How to deal with unusual anatomy or pathology that might itself be rare and thus readily identifiable on its own or with only a few other details.
- What methods to retain partially identifying data elements that might otherwise be de-identified are needed?
  - The task being performed by the AI Model may depend on certain details as inputs. E.g., patient age may be a diagnostic factor, or the relative time between two sets of images may be significant.
  - Assessment of the data records in a repository and/or in a training or test dataset for bias or balance may depend on related details being present for the data elements. E.g., sex, race, age, geographical region, healthcare facility, scanning equipment model, etc.
  - In some cases, it may be necessary to remove identifying details but retain relevant information by doing things like removing the birthdate but leaving the approximate age intact or modifying the study dates to obscure the actual date the patient encounters occurred but maintain the relative times between them. A challenge with such approaches will be to make them work on large amounts of varied data while producing reliable results. Some cases may be semantically difficult, but a manual process may not be feasible.
  - See [STD-DICOM-DEIDENT] for discussion of options that preserve aspects of the dataset.

3.8.4 Annotation

Data records are supplemented with data elements. The supplemental data elements are referred to as annotations.

Because of the obvious symmetries between what the AI Model is intended to generate (ground truth for learning) and what the AI Model generates during clinical use (Results), much of this section also applies to Results. Similarly, since Feedback (see Feedback Use Cases: Collect Model Feedback and Entities: Feedback) often incorporates an annotation representing a corrected result, much of this section also applies to Feedback.

Considerations that may have technical interoperability aspects include:

- Types of things that are the subject of annotations.

  - Ground truth annotations represent the correct result that the AI Model is intended to learn to generate. See 2.1 Applications of AI in Imaging for a broad set of examples.
    - A chest x-ray data record may have annotations for presence of pneumonia, presence of pneumothorax, presence of COPD, presence of scarring, etc.
    - A mammography data record may have annotations for breast composition, according to BI-RADS [BIB3.8.4-1], BI-RADS lesion classifications, and descriptors for lesion morphology and texture.
    - An abdominal CT data record may have segmentation annotations identifying liver tissue, liver lesions, and non-liver tissue.
    - Annotations may be needed to encode the absence of a device, condition, or pathology. For example, a radiology report may not mention all things that are absent if it is not deemed relevant/useful by the reading radiologist. Conversely, if the report does not mention the presence of a tube in the medical images, one should not infer that there is definitively no tube present.
    - Distinguishing between different reasons a piece of information is not present may also be relevant. See Null Flavor [STD-DICOM-NULL] for some considerations when conveying null data.
  - Observations or assessments of the input data elements in the data record
    - E.g., the image quality and whether it is adequate for the task, the patient positioning and whether it is adequate, whether devices like chest tubes that might confound image analysis are present, etc.)
  - Additional input data elements such as observations that may help an AI Model distinguish cases
    - E.g., demographics, laboratory values or coded report findings
- Form of annotations (depends on the type of information)
  - Coded Labels can be used for a wide variety of observations such as identifying pathologies, anatomy, objects, and characteristics.
    - Use standard codes (RadLex, LOINC, SNOMED, ICD-10, FMA)
    - Decide on, and consistently use, a level of granularity that is appropriate to the task result (e.g., encoding anatomy as temporal lobe->cerebral cortex -> cerebrum -> brain -> head)
  - Segmentations indicate the spatial extent of a target.
  - Bounding Boxes indicate the rough spatial location and extent of a target.
  - Spatial Measurements (area, volume, angle)
  - IHE AI Results Profile [STD-AIR] proposes “primitives” that correspond to several forms of annotations for Image Analysis applications.

- Generating annotation data elements
  - Expert human performers of the task the AI Application is intended to perform are a common source.
  - In some cases, annotations are definitive "ground truth"; in other cases, annotations represent the best effort of a human to determine the answer and may not be definitively correct.
- Assessing the quality of annotations
  - Arguably, the quality of the ground truth data elements has more of an impact on the quality of the resulting model than the quality of some other data elements
  - One practice is to have a second expert review the first experts' annotations (rather than submitting a second blinded annotation)
  - Having multiple experts annotate the same data record can mitigate intra-observer variability and increase the quality of annotations
    - Consider a discrepancy resolution method like radiology over-reads
    - Consider quality filters like requiring 3 observers to agree before the corresponding data record is added to the dataset (and the potential side effects of not exposing the algorithm to the omitted cases)
    - Consider aggregating repeat annotations statistically
- Information that might be relevant to record in an annotation data element:
  - The identity of the observer that generated the annotation.
  - Indications of the expertise/skill of the observer (e.g., formal qualifications or credentials)
  - Tools used by the observer (in case the tool induces a bias or quality issue that skews the data)
  - Whether the annotation is based on the observer applying specific standard criteria (e.g., BIRADs score criteria), local criteria, or the observer's personal judgement.
  - Confidence or quality of ground truth (e.g., absolute versus confidence scores, level of ambiguity or hedging)
  - Considerations for that ground truth (causes that support ground truth or additional observations)
  - Observations about the input data
  - See also Common Mechanics: Provenance
- Encoding annotation data elements

- Encoding annotation data elements similarly to other data elements reduces complexity and increases compatibility. E.g., when findings are encoded differently by different applications, or findings that serve as annotations are encoded differently from findings that serve as feedback and differently than findings that serve as results, implementation effort increases and compatibility decreases.
- IHE AI Results Profile [STD-AIR] proposes encodings for several annotation data elements of Image Analysis applications. The proposed encodings (defined in Section 6.5.3 of AIR) include:
    - Comprehensive 3D SR Storage [STD-DICOM-SR] stores:
        - qualitative image findings in form of coded concepts such as presence or absence of conditions
        - measurements, locations, ROIs, and tracking identifiers of image findings
        - Specifically, DICOM Template TID 1500 “Measurement Report” provides a single common encoding for a wide variety of imaging study observations.
    - Segmentation Storage [STD-DICOM-SEG] stores voxel-based spatial segmentations and the associated labels
    - Parametric Map Storage [STD-DICOM-PM] stores floating point pixel images that encode things like saliency heat maps.
    - Key Object Selection Document Storage [STD-DICOM-KOS] is a type of SR that applies a label to a list of images or other DICOM objects.
- Other DICOM encodings of potential interest include:
    - RT Structure Set Storage [STD-DICOM-RTSS] for contour-based segmentations of organs and target volumes for biopsy or radiotherapy targets.
    - Surface Segmentation Storage [STD-DICOM-SSEG] stores segmentations as a polygonal surface mesh.
    - Secondary Capture Image Storage (and its multi-frame and color variants) [STD-DICOM-SECCAP] are “screenshots”. Although readily displayed, this lowest common denominator lacks spatial information and is not machine readable. As such, its use is discouraged, except as an adjunct display for information already encoded in a proper format.
    - Grayscale Softcopy Presentation State Storage (“GSPS” and its color variants) [STD-DICOM-GSPS] store a selected presentation of one or more images and can specify graphic and text overlays. As with Secondary Capture, it is widely supported by PACS, but is not designed as a method of storing primary data so it is best limited to use as an adjunct display.

  - DICOM Supplement 219 [STD-DICOM-JSONSR] specifies a simplified JSON encoding of SR objects that is fully equivalent/transcodable. It is intended to facilitate easy processing by data scientists who are more familiar with JSON.
  - HL7 FHIR [STD-FHIR] includes a variety of Resources that will likely be increasingly supported by deployed HIT systems. Those resources can be relevant sources and sinks for inference task data elements and annotations and data for training those AI Models. A few resources of interest include:
    - FHIR Observation [STD-FHIR-OBS] captures clinical observations and will likely become the preferred format for things like lab results, vital signs, clinical assessments, and readings from monitoring devices.
    - FHIR DiagnosticReport [STD-FHIR-DXR] aggregates multiple related observations.
    - FHIR Patient [STD-FHIR-PAT] captures patient identity and demographics like sex and age.
    - FHIR Encounter [STD-FHIR-ENC] captures information about an encounter between the patient and a healthcare provider like the reason for admission and the department being visited.
    - FHIR Procedure captures information about a procedure that has been, or will be, performed on the patient.
  - HL7 V2 Message Segments [STD-HL7-V2] are currently the most deployed encoding for a variety of observation and operational data.
  - Machine-readable encodings are generally preferable so annotations can be used in training and automated testing. Non-machine-readable encodings (e.g., secondary capture), might be usable for human interpreted testing to compare results visually, but not for training.
  - For some annotations there may be multiple roughly equivalent encodings that are suitable. Failure to converge on standard encodings can create a significant obstacle to data sharing and interoperability. A transcoding that is accurate, robust, and automated can mitigate some of the difficulties.
    - E.g., segmentations may be encoded as a polygon surface, contours on slices, RT structure sets, pixel/voxel masks, etc.
  - Annotations may be informally recorded in some research groups using alternative formats (such as textual labels, NIFTI, and NumPy). Care needs to be taken as these formats often are destructive in nature to metadata and original data cannot be retrieved.
- Transporting annotation data elements
  - Annotation data elements are often simply transported as part of the data record/dataset to which they belong.

    - See Common Mechanics: Exchanging Datasets
  - Annotation activities may be performed outside the organization hosting/managing the data records in which case they need to be conveyed back.
    - See Common Mechanics: Data Access for some possibilities.

3.8.5 Transforms

A data element is produced by processing one or more other data elements.

Transforms are typically performed to make a data element that has been provided by a source acceptable to the needs or preferences of the destination. Transforms are sometimes referred to as “pre-processing”.

Transforms are used during Repository curation to normalize data and create data elements that meet the repository design. Transforms are used during Dataset Assembly to, again, normalize data and create data elements that meet the dataset design. Transforms are used during Model Creation to present data elements to the AI Model in a form it can process, and potentially to enrich the training dataset by creating records that are variants of other records. Transforms are used during Clinical Usage to put AI Model outputs into a form desired by the HIT systems that will use it.

More than one transform could be strung together; capturing which transforms were run as part of these processes is important to demonstrate conformance, compliance, and to aid troubleshooting when something goes wrong.

Transforms can be a useful tool to keep core components converged on a standard set of data formats, while allowing integration with a wider variety of formats used in sites.

Considerations that may have technical interoperability aspects include:

- Transforms that transcode (convert), while maintaining (in principle) the information content
  - Mapping between coding systems (e.g., local procedure codes to RadLex Playbook)
  - Rendering a DICOM image as a PNG file
  - Extracting DICOM metadata or DICOM SR content as XML or JSON
  - Converting a point in a device-based coordinate system to a patient-based coordinate system
- Transforms that extract a relevant subset of the information
  - Extracting and isolating order, observation, and report details from an HL7 message to create a CSV file of pertinent values
  - Creating data elements from fields in the DICOM header
  - Extracting an SR coded concept into a FHIR Observation
  - Cropping an image

  - Extracting information on a target set of allergies from the allergy list of one or more patients
- Transforms that modify information
  - Rotating and/or flipping an image
  - Adjusting the window level/center of an image
  - Resampling an image or waveform (e.g., turning a 1024x1024 mammogram into a 256x256 image consumable by an AI engine, or extracting multiple 256x256 files for processing)
  - Masking or anonymizing demographics or other patient identifying information. See Common Mechanics: De-identification.
  - Modification of information may be lossy or lossless
    - Some data losses may occur on transformed objects, and thus, care should be taken to protect the original data or be able to source it again when needed.
    - Source data may be needed after inference has occurred to relate the results, for example, of a segmentation (where both the original image and the segmentation mask is present).
- Transforms that aggregate information
  - Converting a set of frames into a three-dimensional matrix
  - Normalization of coding and coding concepts using natural language processing to process free text to derive a value
  - Compiling a set of individual procedure records into statistical metrics of departmental performance
- Where the transform is implemented
  - Inside the AI Application package itself
  - Inside the AI Platform
  - Inside an HL7 engine, or a DICOM router
  - Inside a VNA or other data source
- When transforms are utilized
  - Transforms are used in Clinical Usage Use Cases: Access Inference Data Record to make the inputs as provided by the environment meet the input expectations or requirements of the AI Application
  - Transforms are used in Clinical Usage Use Cases: Create Result to make the outputs as provided by the AI Application meet the expectations or requirements of the environment.

  - Transforms in Repository Use Cases: Clean Data Record and Repository Use Cases: Curate Repository may be used to normalize a repository dataset (or might even be performed by the Contributor in Repository Use Cases: Contribute Data if requirements are provided and adhered to)
  - Transforms used as part of Dataset Assembly: Organize Datasets and Model Creation Use Cases: Orchestrate Training may be used to augment datasets and generate synthetic data and create additional synthesized datasets.
    - E.g., taking an image with a ground truth and rotating it one degree 360 times.
- Provenance of transforms performed
  - The list of transforms performed and their results as part of processing pipelines may aid in troubleshooting, and in some cases, may need to be part of the clinical record.
  - See Common Mechanics: Provenance.
- Determinism of transform
  - There may be differences in mathematical calculations for certain transform operations that may impact AI operations.
  - E.g., when shrinking images, a transform from Library 1 and a transform from Library 2 might average pixels differently, which may have a negative effect on the outcomes.

Profile writers may consider some of these questions for further discussion:

- Are there use cases where transforms packaged in an AI Application are conditionally called, depending on the input data?
  - E.g., imaging data from a specific manufacturer may require additional tag transforms prior to processing.
  - How does this impact overall performance monitoring? How does this impact the validation described in Functional Validation Use Cases?

3.8.6 Data Quality and Bias

The content of an artifact is assessed for correctness, completeness (e.g., not missing data elements in a data record), conformance, appropriateness, and bias.

Considerations that may have technical interoperability aspects include:

- Quality criteria
  - **Repository quality criteria** evaluate whether containing datasets align with the stated repository purpose, that they are complete with annotated ground truth made by and verified by trained annotators and use strongly typed ontologies for representing codable concepts.

    - **Dataset quality criteria** evaluate whether include containing a balanced set of data records that support the defined purpose and contain appropriate annotations.
    - **Data record quality criteria** is dependent on the type of data element and record it contains. Imaging data records, for example, could use metrics like image quality, image noise and sharpness, and severity of motion artifacts.
    - **Feedback quality criteria** evaluate responses that are specific, accurate (e.g., generated by someone with appropriate expertise), and depth of detail on how the result was correct or incorrect so that models can be appropriately retrained.
- Assessing quality
    - Aggregating feedback from experts
    - Human vs computational assessments
    - Qualitative vs quantitative assessments
- Conveying quality
    - How to indicate repositories / datasets / models / feedback have good or questionable quality
- Identifying and mitigating bias
    - Bias present in a dataset that could bias performance of the intended task.
    - Datasets need enough data records to ensure bias is accounted for and statistical relevance is present within the dataset for its purpose. Data records that represent situations that would be rarely encountered by the model should be carefully distributed.
    - Consider diversity of sites and equipment from which the data is collected, diversity of patient characteristics, diversity of pathology, appropriate blends of normal/abnormal, diversity of image quality, etc.
    - Account for bias and statistical considerations based on populations and expected occurrence of conditions.

Profile writers may consider some of these questions for further discussion:

- How do characterize and communicate quality?
    - Of a dataset, of annotators (and skill level), of a model, …
- How is fit for purpose assessed?
    - What details might be amenable to quantitative measurements/thresholds vs qualitative assessments?
- How is diversity and bias of the dataset conveyed?
    - Sometimes this depends on the specific task.

    - Factors in the assessment of diversity

3.8.7 Provenance

The origin and history of an identifiable digital entity is recorded and communicated.

Digital entities include AI Applications, AI Models, datasets, data records and possibly individual data elements.

Being able to manage and trust digital entities depends on being able to identify and differentiate them, recognize different versions, and understand how they were created.

Considerations that may have technical interoperability aspects include:

- Uniquely identifying each entity
    - Defining (for each class of entity) what changes to an entity instance require assigning a new identity.
- Describing how the entity was created.
    - Distinguish between "real data" versus artificial/derived/synthetic data, e.g., generated by enrichment processes.
- Detecting and describing changes to the entity
- Describing relationships between an entity and its predecessors
- Recording the change "log"
- Version control
- Tamper proofing (protecting digital entities from undetected modification)
    - Controlling access digital entities both at rest and in-transit is key to developing healthcare systems.
    - The security of other data elements in the repository and available through the interface needs to be factored into the repository creation as well.

Profile writers may consider some of these questions for further discussion:

- How are entities traced to their source?
- Could blockchain be used to convey provenance?
- How is duplication (where two entities that are identical have different identifiers) detected and resolved (de-duplication)?
    - Consider scenarios that may create duplication, such as publicly available data that is independently harvested by two different projects which each render the data into data records in their "independent" datasets.

3.8.8 Exchanging Models and Applications

The information necessary to replicate the operation of an AI Model or Application is accessed or moved from one location to another.

Considerations that may have technical interoperability aspects include:

- Shareable AI Model format
    - See Appendix D.4 for examples.
    - If a dominant format for AI Models is identified and if image archives would be a useful infrastructure to store/distribute AI Models, a DICOM wrapper could be defined (as was done for 3D Print Files).
    - Binary format for loading into binary applications to make the fastest and smallest applications.
        - See Appendix D.4 for examples.
        - Additional information is captured in "model manifest" to help others understand how to use the model.
- Model Metadata
    - These are key pieces of information necessary to execute the AI Application / AI Model, such as description of the task that the application performs, required inputs and outputs, and how the model performs under specific sets of conditions
    - AI Applications will have specific environmental requirements to reproduce the trained results.
    - See Entities: AI Model and Entities: AI Application, and Common Mechanics: Model Performance Metrics.
- Shareable Applications vs Sharable Models
    - Model packages may need additional infrastructure consideration to allow them to function as expected; application packages may have the connectors and services that allow deployment in such environments.
    - Production environments may be heavily application focused while research environments may have more model deployments
    - See Application Distribution and Management Use Cases: Package Application
    - See Entities: AI Model and AI Application

Profile writers may consider some of these questions for further discussion:

- Is the profile for production or research deployment?
- What standard deployment methods can be used to package for distribution? Are there any emerging standards that should be called out?

- How can metadata be standardized to ensure it is properly communicated to Model or Application users?

3.8.9 Performance Metrics

A quantitative benchmark of specific aspects of the model is derived, recorded, and communicated.

Considerations that may have technical interoperability aspects include:

- Performance types
    - Task performance is assessed in terms of the ability of the model to perform its defined task.
    - Outcome performance may be assessed in terms of the impact that use of the model has on the outcome of the workflows in which the AI model is being used
    - Operational performance may be assessed in terms of uptime, resource usage, and error message volume.
- Performance assessment context
    - A Product Manager may assess the task performance prior to release of the model-based product
    - A System Regulator may review assessments of the task performance during product review
    - An ML Ops Analyst may assess the continued model efficacy during operation
    - Radiology Leadership may assess performance as a basis for establishing trust in the AI Model
    - A Data Scientist may assess model-driven improvements in the clinical workflow
- Criteria to measure
    - Some model tasks have recognized performance metrics
        - Classification tasks – sensitivity, specificity [BIB3.8.9-1], accuracy, precision, F1, Area Under the Curve (AUC), etc.
        - Object Detection tasks – precision, recall, mean average precision, etc.
        - Segmentation tasks – Dice coefficient, intersection over union, etc.
    - Relating to outcomes
        - Number of inferences with agreement and disagreement
        - Amount of time saved compared to manual task
    - Relating to operation
        - Percentage of resources used and remaining

        - Number of inferences processed
        - Number of errors and rejections recorded
    - Multiple performance values may need to be estimated and captured for a given Application to communicate how its performance varies for different sub-populations, or in different operational settings.
        - Metrics may affect which Application is invoked for inference, e.g., Application A performs better on patient population A and Application B performs better on patient population B.
- Metric provenance
    - Metrics apply to a specific “version” of an AI Model.
    - See Common Mechanics: Provenance
    - Metric values themselves may need provenance in terms of the version and details of the assessment procedure and the dataset used to generate the metric.
        - Metric provenance may be required for regulatory testing (e.g., in the FDA AI/ML-based SaMD (Software as a Medical Device) Action Plan)
        - How many test cases in the assessment, case distribution, etc.

Profile writers may consider some of these questions for further discussion:

- Where will logging be done and who will have access?
- Should the model result confidence be considered/incorporated into performance metrics? E.g., it may be relevant if the model had a 50% confidence in the 3 results it got wrong, vs having a 95% confidence
- What’s the relationship between model performance metrics (aggregate) and confidence scores for individual case results from the model?

3.8.10 Sequestration

Certain data elements, data records, and/or datasets, are designated and access to them is controlled/restricted.

Datasets that have been, or are intended to be, used for a certain purpose may need to be isolated to prevent their use for a potentially conflicting purpose. For example, a test dataset would be biased if it contained data records that had been used for training the model being tested.

Individuals assembling datasets for training, validation, or testing should consider a strategy for how they will avoid “cross-contamination” of their datasets.

Considerations that may have technical interoperability aspects include:

- The framework for designating and tracking

    - One approach is that each data element that is used as part of a dataset must be accounted for based on that specific use case and point in time.
- An inventory of data elements must be part of the dataset.
- Any data elements added to a dataset after its initial creation need to be specified and the dataset version/provenance updated.
- Provenance is an important tool for sequestration since it can support reliably tracking the identity of data records and their source and chain of custody, which would help detect if the same data record has been used in different datasets for conflicting purposes.
- Private and federated repositories must have mechanisms to ensure data elements only appear once in the model development lifecycle. This can become a challenge when data is obtained from multiple sources and has gone through de-identification.
- How the potential “directionality” of “contamination” affects sequestration strategies
    - Validation and test datasets need to be concerned that their data records have not previously been used in training. But training has no particular concern about whether data records have been previously used for testing/validation.
    - One strategy focuses on collecting "fresh" (recently acquired and otherwise uncirculated) data records for testing. Once the data has been used for testing it could be released for use in validation and not used again for testing. After it's been used for validation, it might be added to the training pool. And training can make use of any data available since a modest degree of duplication of training records is a lesser concern.
- Sequestered data may be from different physical sites to test if models can be generalized to new sites.
- Repositories should have sequestration / locking functions which allow access to sequestered data for appropriate use cases.
- For regulatory purposes, the same data may be used to test multiple models to ensure consistency.
    - Consider creating access logs on the repository and making them available to users or otherwise incorporating them into provenance information to support detection of duplication and re-use

Profile writers may consider some of these questions for further discussion:

- Should test data ever be completely deidentified since UIDs cannot be confirmed as not having been used for training?
- Does making test data available in a shared repository raise insurmountable reliability challenges?

3.8.11 Security

Sensitive data is secured from inappropriate access both at rest and in-flight.

Security is very broad, and this section does not cover all aspects necessary for securing AI applications. IHE has published profiles and white papers on security topics [BIB3.8.10-1].

Considerations that may have technical interoperability aspects include:

- Goals for security of systems and data
  - Privacy for identified data
  - Enforcement of data use agreements
  - Prevent tampering with AI models and results
- Methods for security
  - De-identification of data. See Common Mechanics: De-identification.
  - Authentication and authorization controls for access to repositories and datasets.
  - The use of encryption (e.g., using SSL certificates) when querying, retrieving, and storing data and models.
  - Networking controls to protect endpoints from denial-of-service attacks, entity-in-the-middle attacks, etc.
  - Controls to safeguard the content and prevent AI models, applications, and data from being compromised.
- Boundaries of security in AI solutions
  - Since applications may be composed of multiple components, security may need to be addressed between components as well as interactions with the rest of the enterprise.
  - E.g., transforms, data communication endpoints, model trainers, data fetchers, etc.

Profile writers may consider some of these questions for further discussion:

- What are the impacts on security when in the workflow an AI Application is being invoked?
  - E.g., an AI Application being invoked as images are being acquired may have different security implications than an AI Application being involved across a patient population.
- What data exposed in log messages during the AI processes need to be anonymized and/or secured?

4 Entities

This section provides a starting point for scoping what key entities described in the use cases represent and what associated metadata would need to be captured. Ultimately, such work will be done during profile development, so this section is not intended to be definitive or complete.

It is intended that the attributes in the entity metadata tables below are driven by the implied and explicit requirements of all the use cases that refer to these entities; however, this work has not yet been completed. Input is welcome.

4.1 Data Element

The data element entity represents a data element and its associated metadata.

A **data element** is an individual piece of data.

The primary use of data elements is to represent the **inputs and outputs of an inference task**. Training, validation, and test data records may also contain supportive data elements that are neither passed as inputs nor are not outputs but are needed for administrative tasks. For example, the patient race might not be passed to the AI model as an inference input but might be maintained in the data record to support the assessment and avoidance of racial bias in the development of the AI model.

The **granularity** of data elements may vary. E.g., a data element may be an individual image (DICOM or PNG), the patient age, a reason for admission, a segmentation (contours or RTSTRUCT), a label (Condition Present), a medication, or a lab value (creatinine level).

Some types of data elements are often referred to by specific terms.

- When a human performs the task that the AI Model is intended to learn to perform, the human will generate the “correct” output data element. This process is referred to as **Annotation**. Annotation might also be performed by a different piece of software. Annotation might also be performed to generate input data elements that are missing in a data record.
  - See Common Mechanics: Annotation
  - See Entities: Annotation
- When the AI Model performs an inference on a data record in actual use, the output data element is referred to as a **Result**. See Clinical Usage Use Cases: Create Result.

Data elements often pass through several **processing steps** in the course of being created, contributed, cleaned, shared, refined, and used.

- Data elements may be de-identified at various stages
  - See Common Mechanics: De-identification
- Data elements may be transformed to change the encoding, or extract smaller data elements from a larger composite entity

  - See Common Mechanics: Transform
  - Transform processes may occur during the original clinical usage, during contribution, cleaning, curation, annotation, and sharing of the repository Dataset, during obtaining, refining, annotating, organizing, and sharing of the training, validation, or test dataset, and just before passing the data element to the AI Model during training, validation, testing, or clinical use.

- In principle, the above could be captured in the provenance details for the data element and the data record but doing so is often challenging and impractical.

Table 4.1-1: Data Element Metadata

Attribute	Description
identity	Uniquely identifies the data element.
provenance	A record of: • the origin of the data element (where it came from, who/what created it, the quality/skill of the creator, etc.) • changes to the data element (transforms, normalizations, de-identifications, assessments, etc.). Further modelling to map out a common data structure for provenance details has not yet been done. See Common Mechanics: Provenance for further details.
format	The encoding of this data element. Note: in a uniform dataset, this is the same for all occurrences of the data element so it could be coded at the data record or dataset level.
content	The data that comprises the data element. The granularity may be elemental, or compound as noted above.

The **format/representation** of data elements will vary widely due to:

- the wide variety of applications (see Applications of AI in Imaging) and the corresponding variety of information that data elements represent images, demographics, device characteristics, clinical observations, billing details, workflow details, diagnoses, procedures, spatial segmentations, measurements, scan protocols, classifications, etc.
- the variety of encoding formats available for the various types of information. Such differences represent an interoperability challenge. Some of these may be addressed with Transforms (see Common Mechanics: Transform). Profiling will also attempt to encourage convergence on an effective subset of the possible formats.

Annotations of images are of significant interest in this white paper, both as input data elements and output data elements.

4.2 Data Record

The data record entity represents a collection of data elements and its associated metadata.

A data record is a set of related data elements. The relationship is often that they are all associated with the same patient encounter, but this can vary. E.g., a data record might consist of an x-ray, a lab result, and a reason for admission, all associated with a given patient encounter.

A training data record may identify which of the data elements are input data elements for the inference and which are the "result" output data elements, or that may be configured into the training engine. See the "role" attribute in Table 4.2-1.

A simple data record might contain two data elements, an input image (e.g., a chest x-ray image) and an output text label (e.g., pneumonia). A data record can include data from different time points, such as a current image and a corresponding prior image from 6 months earlier. Some AI Applications may make use of data elements beyond those in the data record provided to them, for example by searching available data sources for additional data elements.

Table 4.2-1: Data Record Metadata

Attribute	Description
identity	Uniquely identifies the data record.
provenance	A record of: • the origin of the data record (where it came from, who/what created it, the quality/skill of the creator, etc.) • changes to the data record (addition or deletion of data elements, etc.). Further modelling to map out a common data structure for provenance details has not yet been done. See Common Mechanics: Provenance for further details.
Data Elements	References to the specific Data Elements that constitute this data record.
>role	The role played by this data element in this data record. The role reflects the purpose of the dataset the data record is in. A data element that is an output in one dataset, might be an input in another. Input: This data element is an input to the AI Model. Output: This data element is an output of the AI Model.

Attribute	Description
	Note: this presumes a "push" dataflow where the environment learns the needs of the AI Model and pushes appropriate data. The alternative "pull" dataflow would tag the concept encoded in each data element at the AI Model would use that to locate the inputs it needs in each data record. Also, since the role would be the same for all data records in a dataset, it probably makes sense to have this at the dataset level rather than each data record.

The FHIR Bundle resource [STD-FHIR-BUNDLE] combines related results into a group and may be a relevant encoding method for data records.

4.3 Dataset

The dataset entity represents a dataset and its associated metadata.

A dataset is a set of related data records. Typically, all the data records in a dataset contain roughly the same data elements. Datasets containing data records with patient data will typically span many patients.

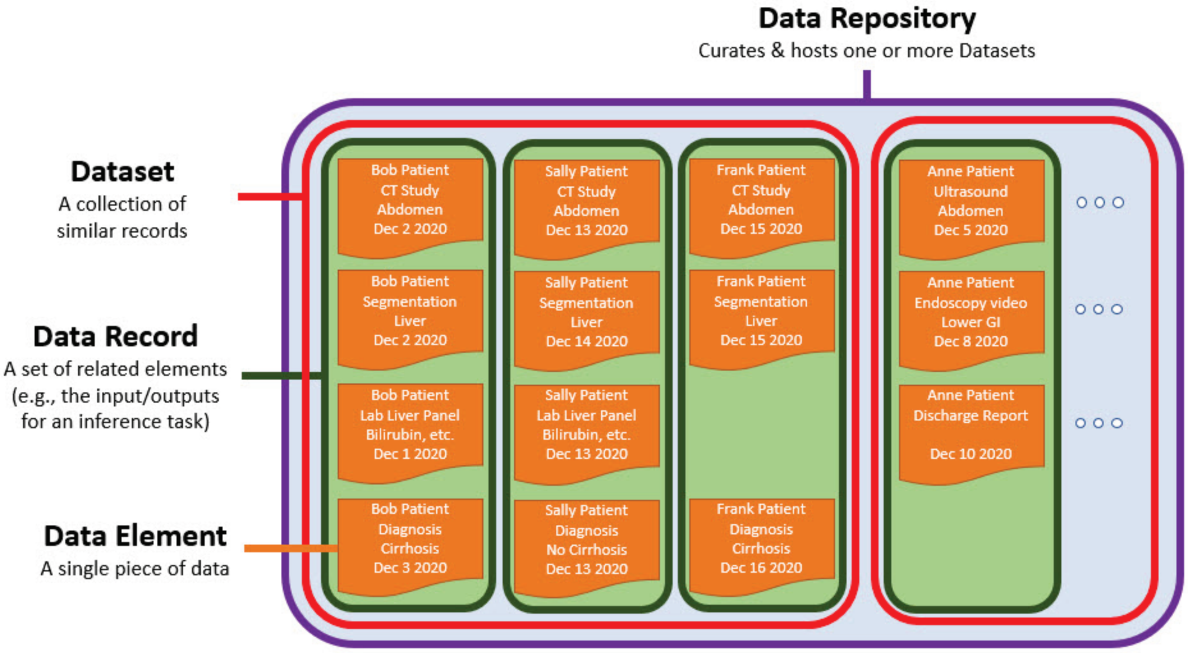

Data Repository
Curates & hosts one or more Datasets
Dataset
A collection of similar records
Data Record
A set of related elements (e.g., the input/outputs for an inference task)
Data Element
A single piece of data
Bob Patient CT Study Abdomen Dec 2 2020
Bob Patient Segmentation Liver Dec 2 2020
Bob Patient Lab Liver Panel Bilirubin, etc. Dec 1 2020
Bob Patient Diagnosis Cirrhosis Dec 3 2020
Sally Patient CT Study Abdomen Dec 13 2020
Sally Patient Segmentation Liver Dec 14 2020
Sally Patient Lab Liver Panel Bilirubin, etc. Dec 13 2020
Sally Patient Diagnosis No Cirrhosis Dec 13 2020
Frank Patient CT Study Abdomen Dec 15 2020
Frank Patient Segmentation Liver Dec 15 2020
Frank Patient Diagnosis Cirrhosis Dec 16 2020
Anne Patient Ultrasound Abdomen Dec 5 2020
Anne Patient Endoscopy video Lower GI Dec 8 2020
Anne Patient Discharge Report Dec 10 2020
○○○
○○○
○○○

Figure 4.3-1: Data Element, Record, and Set Hierarchy

Figure 4.3-1 shows the hierarchical relationship between data elements (orange boxes) and data records (rows of boxes) within a dataset (blue box).

Table 4.3-1 lists some possible metadata attributes for a dataset.

3880 **Table 4.3-1: Dataset Metadata**

Attribute	Description
identity	A globally unique identifier for the dataset.
provenance	Source and history of the dataset
status	The status of the dataset
title	A human-readable description of the dataset
purpose	Whether the dataset contains training, validation, testing, or simply repository data records
author	Who created the dataset
reviewer	Users who have looked at the dataset to ensure it is of sufficient quality. See Common Mechanics: Data Quality and Bias.
organization	Healthcare institution that created the dataset
inclusion criteria	Constraints on characteristics used to include data records in the dataset. All records in the dataset conform to these criteria. See Repository Use Cases: Retrieve Repository Content for further discussion.
exclusion criteria	Criteria used to exclude data records from the dataset. Records with these characteristics will not be present.
data records	References to the data records included in this dataset.
deidentified	Whether all the records in this dataset have been deidentified

Profile writers may consider some of these questions for further discussion:

- Does identity change when the content of the dataset changes or do we have layered versioning?
- 3885 How is model "bias" accounted for in the assembly of datasets?
- How can problems be identified and mitigated?
  - E.g., "faulty burnt-in demographics in a dataset which lead to a faulty model."
- How do we organize inclusion and exclusion criteria?
- Are git hashes something we would consider on fixing the dataset in time?

- What is the impact of dataset compilation on model reproducibility?
- Does the size of the dataset need to be conveyed, in number of records or sheer size on disk?
- How are any IRB restrictions conveyed?
- How are data elements and data records of a dataset described in the identity and any other identities they may be part of?
- How should sequestration of datasets be conveyed? See Common Mechanics: Sequestration.

4.4 Data Repository

The repository entity represents a repository and its associated metadata.

A data repository is an infrastructure that hosts one or more datasets for discovery and retrieval.

Table 4.4-1: Data Repository Metadata

Field	Considerations
identity	Globally unique identifier for the repository. It does not change when the datasets change.
title	A human-readable description of the repository
purpose	A description of the intended purpose of the repository
endpoints	Endpoints for the discovery and retrieval use cases
licensing	A brief description of the licensing and terms of use of the repository. May include a link to more detail and/or how to obtain a license.
terms	Terms of use of the repository
author	The identity and contact information for the Repository Administrator
organization	The organization responsible for the repository
datasets	UIDs of the datasets contained in the repository

4.5 Transform

The transform entity represents a transform and its associated metadata.

A transform is a process that produces one or more output data elements from one or more input data elements, typically by transcoding or resampling. See Common Mechanics: Transform.

Table 4.5-1: Transform Metadata

Attribute	Description
identity	Globally unique identifier of the implementation of the transform
title	Human readable label describing the transform performed
input	What does the transform expect as input, including details such as the expected number of inputs, the semantic content of each input, the acceptable encodings/formats of each input, limits on acceptable values/sizes, which may be optional, what defaults are assumed, etc.
output	What does the transform produce as output, including the number of outputs, the semantic content of each output, the encoding/format of each output, limits on the content that can be produced, etc.
version	Human readable version number of the implementation
library	Human readable name of the implementation

Several example instantiations of a Transform follow:

Table 4.5-2: Example Transforms

Attribute	#1	#2
identity	2.28.7687648762398	2.28.276879468927
title	"Rotate Image"	"Make NIFTI"
input	Parameter 1: a single DICOM instance (with or without a frame reference) Parameter 2: degree of rotation to the right	Parameter 1: a DICOM series
output	A single DICOM instance containing the rotated image or frame	A NIFTI matrix of points
version	1.0	1.0
library	DicomRotater	Dicom2NIFTI

3910

4.6 AI Model

The AI Model entity represents an AI Model and its associated metadata.

An AI Model is a neural network architecture and a set of weights that has been trained to produce appropriate outputs when supplied certain types of input. It is represented by data-only, i.e., it does not contain code as opposed to AI Applications covered in the next section. It requires a compatible machine learning framework to be executed (see [BIB3.4.1]). 3915

Table 4.6-1: AI Model Metadata

Attribute	Description
identity	Globally unique identifier for this instance of the model Consider whether UIDs or Digital Object Identifiers (DOI) might be more appropriate.
version	Model version number
family	Globally unique identifier of a family of models this instance belongs to (e.g., the family might collect all instances as they evolve over time)
parent model	Parent model from which this model was derived
title	Name of the model as a human readable label
status	A variety of status flags, including Draft, published, validated, cleared for clinical use (by whom), deprecated, obsoleted
task	Description of the task the model performs. May describe both the type of task (e.g., procedure recommendation, tumor detection, classification, segmentation, measurement) and the intended context (e.g., a particular organ or body region, a particular type of imaging procedure, a particular type of patient, etc.). See Section 2 for examples.
contraindication	When should this model not be used
bias	Known limitations and potential considerations for bias
target	Other target constraints (e.g., run only for a specific clinician or group)
input	Required inputs and optional inputs that will be used if available
input defaults	Default values and behaviors used when optional inputs are absent
output	Outputs the model produces to complete the assigned task
explainability output	Outputs the model produces to explain or provide supporting information for the task output (e.g., heatmaps, meta data, simplification method)
created	Nominal date and time that this instance of the model was created
author	Name, institution, and contact information for the author of the model.
data provenance	Sources of training data and validation data
training data	References to source training data

Attribute	Description
validation data	References to data the model was validated on
score	How the model performed with test data, and through what means was that score achieved. See Common Mechanics: Data Quality and Bias.
model provenance	Information on the internal architecture of the model, including what standard models it is based on

Profile writers may consider some of these questions for further discussion:

- How are diversity zones of data conveyed and summarized?
    - E.g., types of patients, types of equipment
- What other aspects of provenance need to be considered?

4.7 AI Application

The AI Application entity represents an AI Application and its associated metadata.

An AI Application is a package of components (including algorithms, necessary data transport interfaces, and input / output transforms) to be executed in a target environment to perform AI tasks. The algorithm(s) may be based on deep learning, conventional machine learning, or other techniques.

The number and type of executable components in the AI Application may depend on what infrastructure (hardware, operating system, software libraries) is present at the target environment. It must contain everything that is needed to simply deploy and install the AI Application in the target environment and start running inferences on it without the need to write any transform software or figure out how to configure the model in the target system. Software components used in an AI application in addition to AI models may include:

- the ML framework package matching the model definition file, e.g., TensorFlow or PyTorch (although this is most likely available in the target environment and therefore not necessary to be part of the package)
- code to transform input data into a suitable format to be supplied as input to the model, typically written in Python, Java, JavaScript, Go for Kubernetes, and C++.
- traditional algorithms, e.g., rule-based expert systems, which may run individual AI Models as part of individual rules or be completely based on non-machine-learning code.
- additional software libraries needed to run the transform code, e.g., Python libraries.

All these files may be combined into an executable package, such as one of the following:

1. A common “notebook” format such as Jupyter Notebook which can be imported into cloud services ([BIB4.7.2-2] [BIB4.7.2-3] [BIB4.7.2-4], [BIB4.7.2-5]) to be deployed and run on the applicable cloud platform ([BIB4.7.2-1] [BIB4.7.2-6]).
2. A self-contained Docker file that can be deployed in a container environment like Kubernetes and executed on cloud services.
3. An executable, possibly packaged with an installer, for local deployments, that can be downloaded and run natively on a base operating system.

Optionally, the executable package may require a license and/or compliance to an End User License Agreement to run.

The following table contains a list of metadata that could be used to describe an AI application in a machine-readable format on top of the metadata used to describe AI Models. Some attributes may be omitted depending on the contents of the AI Application package.

Table 4.7-1: AI Application Metadata

Attribute	Description
identity	Globally unique identifier of this version of this AI Application
title	Human readable label
status	A variety of status flags, including draft, published, validated, clinical use, bad
model	AI Model(s) packaged in the application (note: the metadata exposed through the model is likely important; body site, condition, etc.)
application input	Data elements the application expects as input
application output	Data elements the model produces as output
input transforms	Types of transforms that can be applied to the AI Application input before processing by the AI Model
output transforms	Types of transforms that can be applied to the AI Model output and/or explainability output to produce the application output
run conditions	Environment in which the application should be run in (minimum requirements)
created	Nominal date and time that this instance of the application was created
creator	Name, institution, and contact information for the creator of the application package.

Attribute	Description
score	How the model performed with test data, and through what means was that score achieved. See Common Mechanics: Data Quality and Bias.

4.8 Service

The Service entity represents a service and its associated metadata.

A Service is a running instantiation of an AI Application.

Table 4.8-1: Service Metadata

Attribute	Description
AI Application	The specific AI Application that is hosted by the Service
state	The state of the Service, such as running or offline
endpoint	The interface used to interact with the Service

4.9 Feedback

The Feedback entity represents feedback and its associated metadata.

Feedback is an assessment by a human or system of the output produced by an AI Model from a given data record.

This entity will likely reference Entities: Data Element, Entities: Data Record, and Entities: AI Model and AI Application.

Table 4.10-1: Feedback Metadata

Concept	Definition
model reference	Uniquely identifies the model that produced the subject of the feedback.
data record reference	Input data record from which the model produced the result/output for which feedback is provided In the event of a study reference, this may include references to the series, instance, frame, coordinates of issue (of a label, of the pixels, of the markup, etc.)
assessment	How well the result produced by the model represents the correct result for the task

Concept	Definition
contributing factors	Any known considerations that indicate why the AI Model may have performed poorly (e.g., distorted image, secondary presence of unexpected finding)
feedback source	Identifies the human or process that provided this feedback
original result	Identifies the value that was generated by the AI Application
original result confidence	Provides how confident the AI Application determined was correct for the result
correct result	Optional, provide result data element(s) considered correct by the feedback source
corrected result mechanism	How the correct result was attained
observer	Who provided the feedback, and their level of responsibility for the patient (e.g., referring physician, radiologist, specialist, or technician)

Considerations that may have technical interoperability aspects include:

- The specific study reference may be optional, as the feedback may be going to an external 3rd party system.
  - This might also be a “multi-step” process; might be internally collected (where patient details are included) and then aggregated for anonymized analysis (where patient details are excluded)
- Feedback might also be implicit, e.g., when there is no feedback, the model may not be used at all, or the model is performing great. It is unlikely clinicians will provide feedback on well-working models all the time.
  - This is important to differentiate, as a model should not be reinforced if there is no implicit or explicit feedback.

Profile writers may consider some of these questions for further discussion:

- When feedback is being sent externally, how can it be de-identified but not lose purpose?
  - E.g., should patient population or general demographics be transmitted instead?
  - Does the institution or the specific scanner / version / firmware be included?
- What are the different types of feedback that would warrant different interventions?
  - E.g., a study failing because the imaging study is blurred should be treated differently than when the study has failed because it missed a finding.
  - Both types of data should be useful input to model training.

- When a model fails (with an error message), should this be treated the same as a feedback message?
- How is model feedback weighted?
- Is IHE SOLE [STD-SOLE] or IHE ATNA [STD-ATNA] a better construct for providing feedback?

Appendices

Appendix A – AI Background

A.1 Deep Learning

A neural network is a computational learning system that uses a network of functions to understand and translate a data input of one form into a desired output and was inspired by neurobiology. [BIBA.1.10] Deep learning neural network models are the primary model structure used because of their top scoring performance on computer vision object detection, and segmentation benchmark tasks on data sets such as COCO (Common Objects in Context) [BIBA.1-1] [BIBA-2].

At a high level, a deep learning model uses the concept of the neuron to describe how it reacts in a specific layer of a particular region of a 2D or 3D image. These neurons will take an input, multiply the value with a model weight, and provide the output to the next layer of the model. These weights are critical to producing a deep learning result and are created as part of the model training process.

The recent performance gains of deep neural networks were made possible by increases in computing power and storage capacity to train these deep neural networks. Distributed learning increases computing power and storage for model training by spreading training across multiple server nodes or GPUs training the model in parallel [BIB3.3.2-3].

Neural network (NN) models are structured in multiple layers of nodes that use matrix multiplications to transform input data and pass output to adjacent layers. The layers are designed to learn increasingly higher-level imaging features [BIBA.1-3].

The nodes of the layers have a weight and an offset bias that transform input data to output. The weight determines the contribution of the input data to the model and the offset bias sets an activate threshold for the node to pass the transformed input data to the next layer. The nodes can be fully connected to the nodes of adjacent layers or only to spatially adjacent data by convolutions that match templates to the image data [BIBA.1-4].

Convolutional neural networks (CNN) are commonly used in medical imaging tasks because they only link data spatially close together. Data spatially close would have more relation than data far apart in medical images. CNNs have proven high performing in computer vision benchmarks.

Other, fully connected, architectures such as transformer networks [BIBA.1-5] can also be used in computer vision and therefore medical image tasks.

Transfer learning is reusing a previously trained model as a starting point for a second task [BIBA.1.11] by re-training the last layers [BIBA.1.12]. Offset bias is the learned constant additional inputs to nodes of neural network models that adjusts how easily a node will activate that are learned during model training [BIBA.1.13].

Four errors that could occur as part of model training include overfitting, bias, predictive errors, and variance errors. Overfitting a model is a type of error where the model function fits limited training data too closely and performance on novel data is lower as a result [BIBA.1.14]. Bias, a type of model error, is important to communicate, as clinical users of the model would want to know about to understand what kind of errors the model may make in clinical practice. Predictive error occurs due to systematic prejudice from faulty assumptions. Simpler models tend to have from bias [BIBA.1.15] [BIBA.1.16]. Variance errors occur when there are changes in predictive estimates of models with different training data. More complex models with more parameters tend to have higher variance.

A.2 Training Methods

Training (sometimes also called Learning) involves determining good values for all weights and biases of a model from labeled examples [BIBA.2-1].

Transfer Learning or "fine-tuning" involves small adjustments often with a smaller learning rate to model weights and biases to improve predictive performance of an existing model. [BIBA.2-2]

In Collaborative Learning (such as Federated Learning), the model collaboratively learns a shared prediction model with distributed training data while keeping the data holders' data private. [BIBA.2-3]

Distributed Learning involves spreading computation of model training and training data storage across multiple machine nodes or GPUs to enable parallel computation to process larger models and larger training data sets.

A.3 Training Process

Training data composed of input and expected ground truth output samples are split into sets for training and testing. Back propagation [BIBA.3-1] is used to iteratively adjust the node weights and biases of the model to fit the transformed outputs to expected training data outputs over epochs.

The error between the transformed output and expected ground truth is measured as loss. The model weights and offset biases are adjusted to minimize loss.

At the end of training epochs, the validation dataset is transformed by the partially trained model and compared to the expected output. Validation dataset loss growing larger than training dataset loss indicates model overfitting to the training portion of the data and the model does not generalize to data it was not trained on. Overfitting is reduced by randomly dropping out some nodes of layers each epoch, reducing the parameters of models by reducing nodes or layers, and cutting off training epochs when the validation dataset loss exceeds the training set loss.

Model weights are often saved after multiple epochs so the performance characteristics of the trained models after different epochs can be compared for selecting the best trained models for different tasks.

At the end of the training and testing runs model hyper-parameters such as number of layers, nodes in layers, initial weights, and offset biases, training epochs, learning rates may be adjusted. The training and validation datasets are seen repeatedly by the model during training cycles. The hyper-parameters for the model are optimized for performance on the validation dataset. Models can be trained, starting from random weights and offset biases or from pretrained base models.

An alternative to model pre-training is self-training [BIBA.3-2]. In self-training models are trained with unlabeled data to perform data transforms and restorations such as local pixel shuffling and restoring [BIBA.3-3]. The unlabeled data is more common than labeled data increasing the self-training sets size. Then after self-training labeled data specific to the medical tasks are used to fine-tune unlocked layers of the model as in transfer learning.

A.4 Fine-Tuning Models

Models that have been fine-tuned should be able to function independently from the source model; it should have its own identity (with provenance tracked).

An existing pre-trained model, trained either at the same location or at a different location, can have the model weights and offset biases adjusted to address a new dataset, whether it was produced on different types of equipment, a different population, or other factors.

Pretrained models may come from online collections; see Appendix D.4: Reference Toolkits.

Existing models could have been pre-trained with non-medical images, or on more common medical images. The existing trained model layers can be transferred to address new datasets through a process called transfer learning. In transfer learning individual layers of trained models are locked or frozen from additional training or unlocked for fine-tuning of the weights and offset biases. More unlocked layers will require more fine-tuning epochs and more training samples in an iterative training process like training from scratch.

A.5 Training Models in Healthcare

Healthcare AI has a specific problem in that the data used is highly confidential and anonymizing that data may have damaging effects (especially when multiple pieces of data need to be linked together, or uses textual reports where anonymizing is an exceedingly difficult problem).

Medical image data’s different characteristics from the common computer vision datasets used to train model architectures can require training from scratch. Training from scratch starts with randomly initialized weights and offset biases of the nodes. Training from scratch requires sufficient ground truth, annotated positive and negative training, and a test dataset.

To train a model and derive the correct weights, a large dataset is needed to guide the training process. There may not be enough ground truth annotated medical imaging data to train deep neural networks from scratch. Especially for positive cases of rare diseases. Starting with trained model layers and fine-tuning model weights and bias offsets with annotated medical image training sets can reduce the required training data.

Models can optionally fine-tune to individual institutions locally. The weights and offset biases of nodes in model layers can be unlocked and further trained on local image data to tune the model for the institution. This would change the model and earlier validation results may no longer apply to this model. Local tuning would be more commonly used in research trials though the FDA is proposing regulatory frameworks for continuous tuning of models [BIBA.5-1].

Model architectures for medical imaging tasks can be selected by finding high performing architectures for similar computer vision tasks on benchmark image datasets. The selected base model architecture will need further adaptation for medical imaging tasks. Computer vision model architectures are often structured for two-dimensional color image data. Model architecture should be adapted for grayscale medical images and three dimensions for CT, MRI, and ultrasound. Trained base architecture layers can be selected as a starting point and further trained for medical imaging use cases.

AI Models and Applications often require significant hardware resources to analyze large volumes of images efficiently and produce results in time for clinicians to have them available when they need them. For example, AI Applications analyzing imaging exams taken in an ER will need to have shorter turnaround times than AI Applications looking for less urgent conditions like cancer or dementia. It is a significant challenge to provision and monitor resources to satisfy the many needs different departments in a hospital or similar institutions in an efficient and reliable manner.

The element of “time” plays a key role in developing AI Applications. Some data elements are related to other data elements in that they capture an observation about the same entity at a different point in time. For example, a volume measurement of the same tumor in the same patient, taken from two different studies at different time points.

Appendix B – Architectural Considerations

This appendix attempts to abstract the business logic, data handling, and process execution associated with performing AI inferences in Clinical Usage.

Figure B-1 shows an abstraction of functional layers between the actual execution of the AI Model (bottom layer) and the conditions in the ecosystem were the initial trigger that led to the execution of the AI and where the output of the model was ultimately used (top layer).

The layers and the functions performed in each layer may provide a useful abstraction when assessing and comparing different architectures to be supported during the development of subsequent profiles. For example, each layer may be performed by a different executable, or several layers may be combined in one executable. It is also possible that multiple architectures may be simultaneously in use in the same environment since and executable at one layer might interact with different instances in the next layer down.

In the diagrams, there are grey bubbles on the boundaries between each pair of layers. Currently, it is assumed that those are things performed by the blue function boxes in the upper layer as part of invoking/interacting with the layer below it, but a conceivable alternative would be to assign that work to the lower layer at each boundary. And in the case of initial selection of relevant inputs, it was not clear whether that belonged to the interface between the first two layers or the second two layers.

Note: The labels for the grey bubbles between the Task Performance Layer and the Model Execution Layer have been simplified to read "Unwrap Inputs" and "Wrap Outputs" but may encompass several data preparation steps. "Unwrapping" may involve extracting a specific element of interest to the Model from the input object, as well as possible transforms or transcoding to fit what the Model expects. "Wrapping" may involve augmenting the output of the Model with additional contextual metadata and other transforms and transcoding to make the output object usable by the receiving systems, e.g., encoding a binary TRUE result into a finding of "Pneumonia is present" and attaching metadata identifying the patient, while composing the result object.

This white paper does not intend these diagrams to be normative, but rather to provide a useful starting point for subsequent analysis. Since several of the architectures that have been proposed in the past may be diagrammed using this as a common representation, it should facilitate comparing.

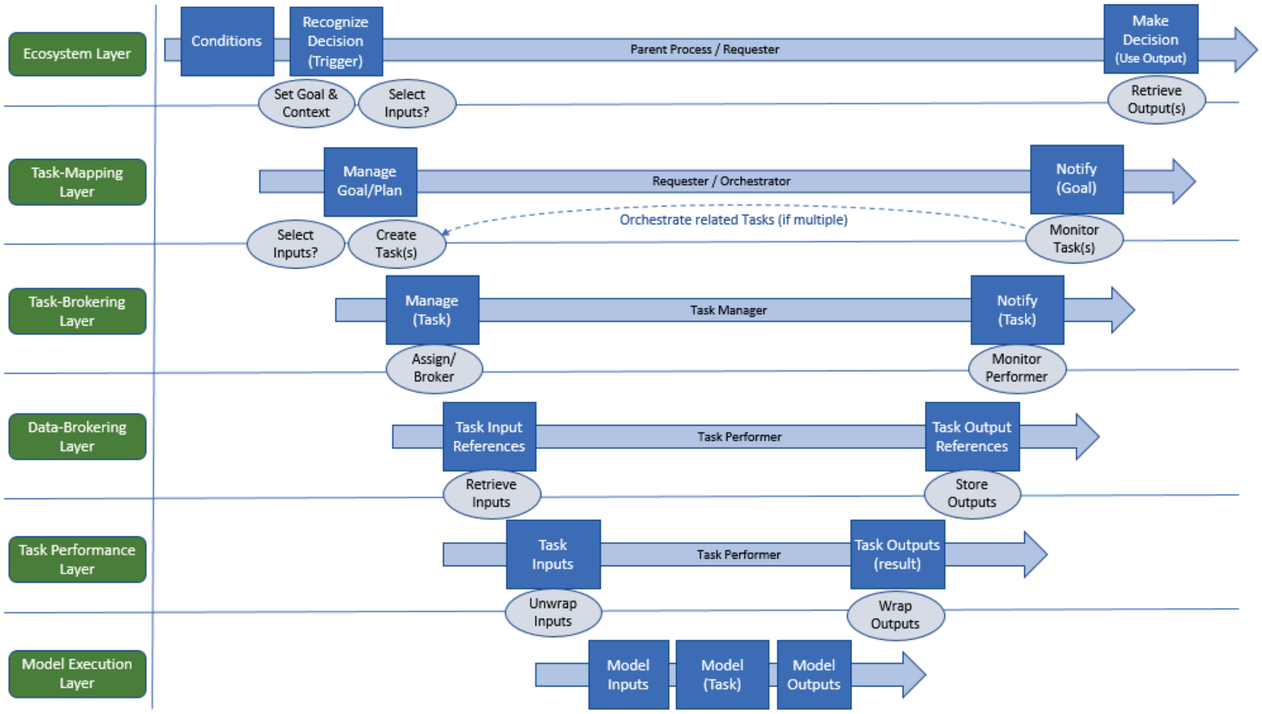


Figure B-1: AI Execution Abstraction Layers

4165 In Figure B-2, many of the upper layer functionality is folded into a relatively complex Requester.

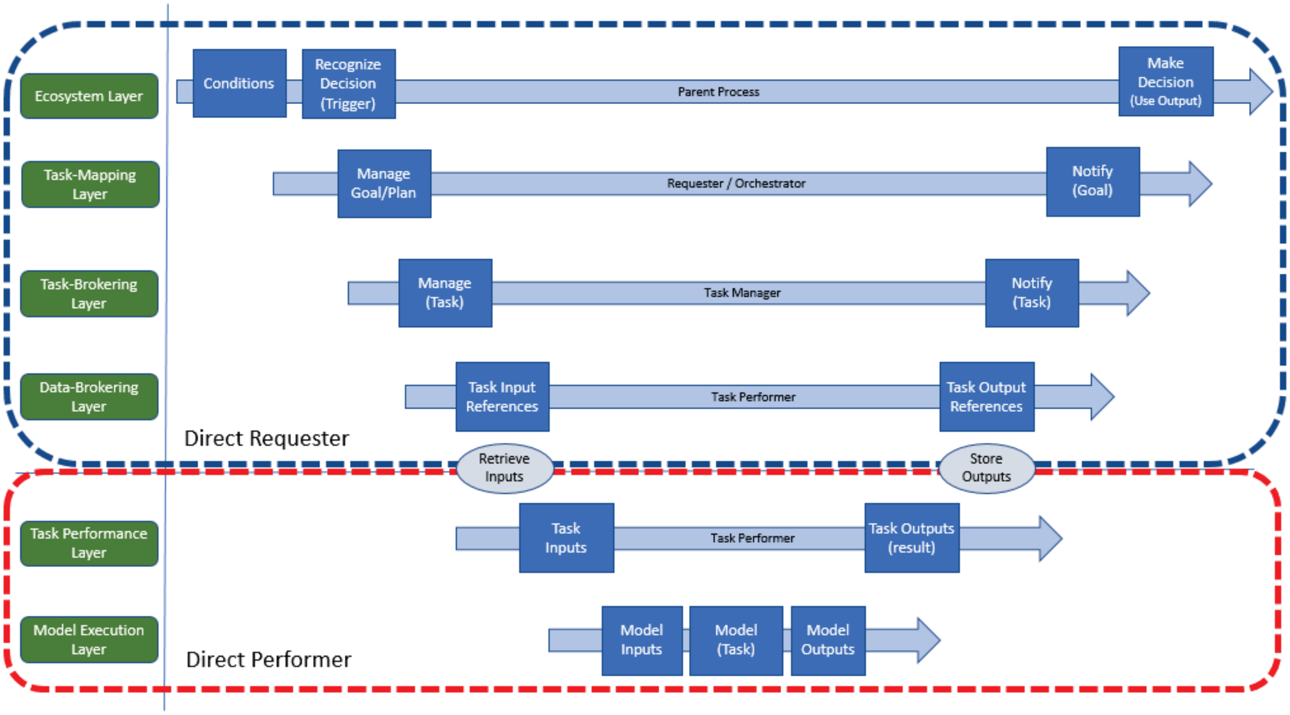


Figure B-2: Complex Requester Example

In Figure B-3, each abstraction layer is addressed by a separate focused executable. This 4170 corresponds to what is described in the IHE Radiology AI Workflow in Imaging (AIW-I) profile [STD-IHE-AIWI].

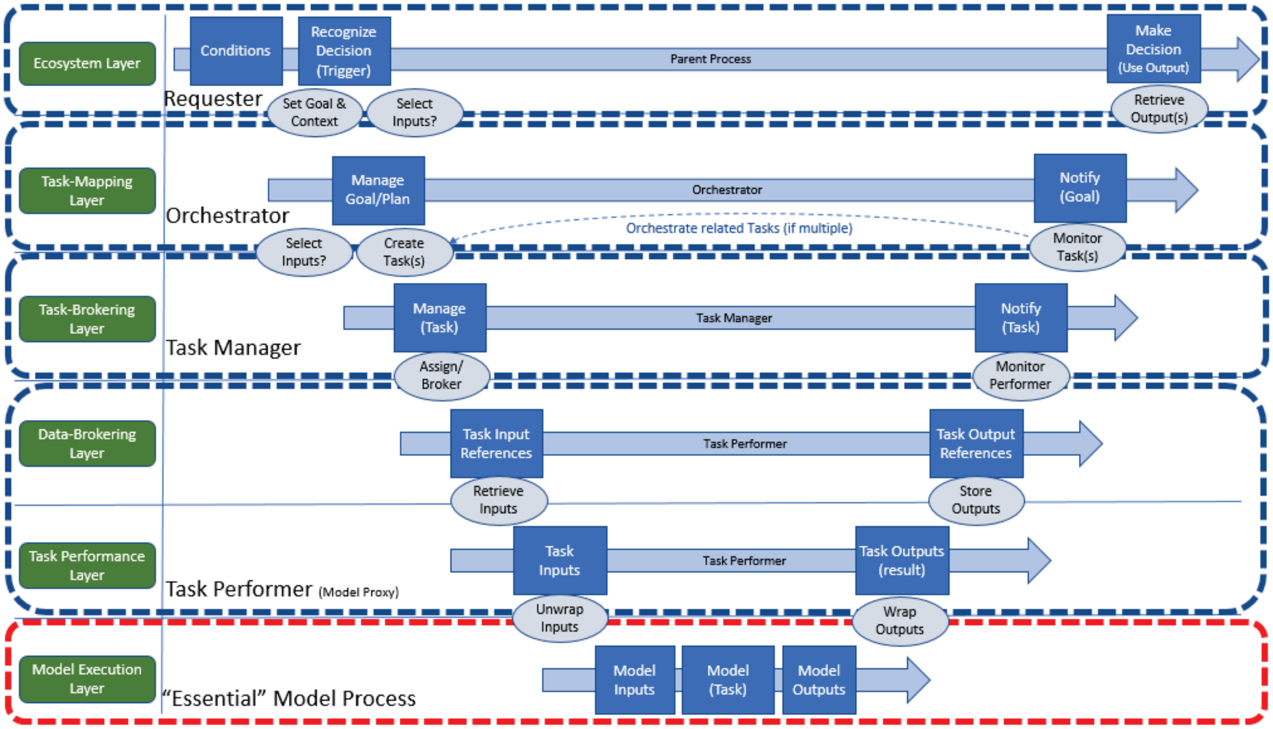


Figure B-3: IHE AIW-I-based Example

In Figure B-4, a more elaborate Orchestration Platform merges several of the intermediate layers.

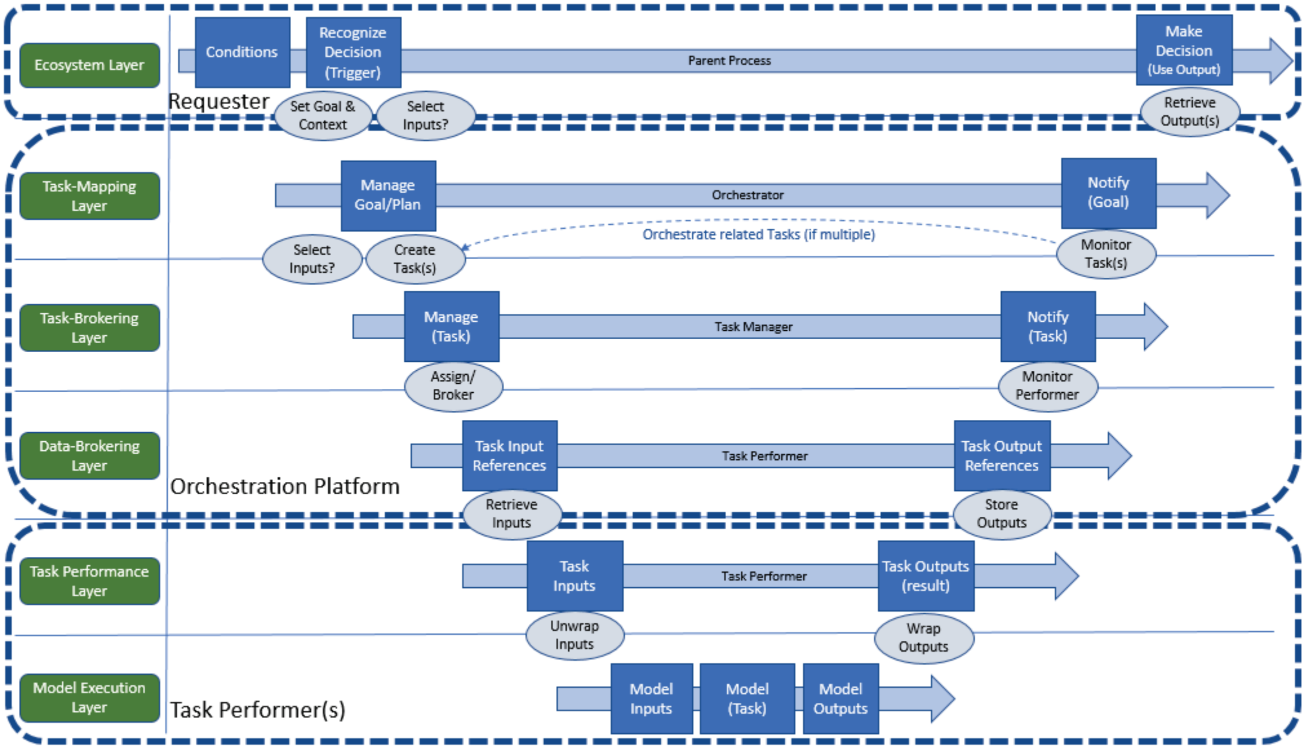
4190 

Figure B-4: Orchestration Platform Example

Appendix C – Glossary / Definitions

The complete IHE Glossary for IHE Technical Framework terms is available here.

4195 For common terms used in AI and Machine Learning, several good references are:

- “Glossary of AI Terms.” MITA https://www.medicalimaging.org/about-standards/glossary-of-ai-terms
- Ranschaert, Erik R., Morozov, Sergey, Algra, Paul R. Artificial Intelligence in Medical Imaging. https://www.springer.com/fgp/book/9783319948775

4200 Table C-1 consists only of terms that are used in this white paper in a more specific way than their "common" meaning.

Table C-1: Definitions for Terms in this White Paper

Term	Definition
data element	An individual piece of data. The inputs and outputs of an inference task are represented as data elements.
data record	A collection of related data elements.
annotation	A data element that has been added to supplement a data record. Annotations can serve several purposes. A key purpose is to represent the correct output an AI model is expected to generate when provided the input data elements from that data record, or additional observations of other data elements.
dataset	A collection of related data records. Typically, all the data records in a dataset contain similar data elements.
training dataset	A dataset used to train an AI model (i.e., derive values for the model’s weights/parameters).
validation dataset	A dataset used during training to validate that the performance of a candidate AI Model generalizes outside of, and is not overfitting to, the training dataset. This is typically used to evaluate the performance of multiple candidate models and select the best one for subsequent training and/or testing.
test dataset	A dataset used to estimate/predict the performance of a selected model on unseen data.

Term	Definition
	This is used by the algorithm developer to report model performance prior to clinical testing/evaluation. Note: Performance on a test dataset may not reflect real-world performance if the data it contains is not representative of the clinical population.
AI Model	A neural network architecture and a set of weights that has been trained to produce appropriate outputs when supplied certain types of input.
AI Application	A package of components (including algorithms and necessary data transport interfaces, and input / output transforms) to be executed in a target environment to perform AI tasks. The algorithm(s) may be based on deep learning, conventional machine learning, or other techniques.
Explainable AI	An AI Application is an Explainable Artificial Intelligence (XAI) if it can provide details that “explain” which inputs influenced the AI Result or expose the functions that generated the AI Result in an interpretable way. For image analysis, this may include saliency heat maps.

Appendix D – Bibliography and Further Reading

The following textbooks, publications, and websites are intended to provide future profile writers with greater background and context prior to profile development.

As machine learning is a rapidly maturing field, these sources may become dated and therefore should be viewed as an initial point of reference for further investigation rather than a series of authoritative sources.

These references were read and recommended by authors of the white paper. Readers may find that some references listed below, while not scientifically peer-reviewed in nature, are helpful explanations, but should refer to authoritative sources where appropriate.

D.1 Bibliography

These sources are directly referenced in the white paper.

- [BIB3-1] “Waterfall Model.”, Wikipedia, accessed Feb 27, 2021. https://en.wikipedia.org/wiki/Waterfall_model.
- [BIB3.1.2-1] Wilkinson MD, Dumontier M, Aalbersberg IJ, et al. “The FAIR Guiding Principles for scientific data management and stewardship.” Sci Data. 2016; 3:160018. ISSN: 2052-4463. https://doi.org/10.1038/sdata.2016.18.
- [BIB3.1.4-1] “NumPy: The fundamental package for scientific computing with Python,” Accessed April 12, 2021, https://numpy.org/
- [BIB3.2.6-1] Jason Brownlee. “What is the Difference Between Test and Validation Datasets?”. July 14, 2017. https://machinelearningmastery.com/difference-test-validation-datasets/
- [BIB3.3.1-1] Nicholas Petrick. “Pre- and Post-Market Evaluation of Autonomous AI/ML: Lessons Learned from Prior CAD Devices”. https://www.fda.gov/media/135712/download
- [BIB3.3.1-2] Ekin Tiu. “Metrics to Evaluate your Semantic Segmentation Model.” August 9, 2019, https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2.
- [BIB3.3.1-3] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, et al, Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI, Information Fusion, Volume 58, 2020, Pages 82-115. https://arxiv.org/pdf/1910.10045.pdf
- [BIB3.3.1-4] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, D. Pedreschi, A survey of methods for explaining black box models, ACM Computing Surveys 51 (5) (2018) 93:1–93:42. https://dl.acm.org/doi/pdf/10.1145/3236009

- [BIB3.3.1-5] L.H. Gilpin, D. Bau, B.Z. Yuan, A. Bajwa, M. Specter, L. Kagal, Explaining Explanations: An Overview of Interpretability of Machine Learning, 2018. https://arxiv.org/abs/1806.00069
- [BIB3.3.2-1] "Distributed training with TensorFlow". TensorFlow, accessed February 27, 2021, https://www.tensorflow.org/guide/distributed_training.
- [BIB3.3.2-2] Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenberg, Tim Verbelen, Jan S. Rellermeyera. "A Survey on Distributed Machine Learning". CoRR, 2019. https://arxiv.org/pdf/1912.09789.pdf
- [BIB3.3.2-3] Hao Zhang. "Intro to Distributed Deep Learning Systems". February 6, 2018. https://medium.com/@Petuum/intro-to-distributed-deep-learning-systems-a2e45c6b8e7
- [BIB3.3.2-4] Brendan McMahen, Daniel Ramage. "Federated Learning: Collaborative Machine Learning without Centralized Training Data". April 6, 2017. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.
- [BIB3.3.2-5] Wenqi Li, Fausto Milletari, Daguang Xu, et al. "Privacy-preserving Federated Brain Tumour Segmentation". October 2, 2019. https://arxiv.org/pdf/1910.00962.pdf.
- [BIB3.3.2-6] Maarten G. Poirot, Praneeth Vepakomma, Ken Chang, Jayashree Kalpathy-Cramer, Rajiv Gupta, Ramesh Raskar. "Split Learning for collaborative deep learning in healthcare". December 27, 2019. https://arxiv.org/abs/1912.12115.
- [BIB3.3.2-7] Ken Chang, Niranjan Balachandar, Carson Lam, Darvin Yi, James Brown, Andrew Beers, Bruce Rosen, Daniel L Rubin, Jayashree Kalpathy-Cramer. "Distributed deep learning networks among institutions for medical imaging." March 29, 2018. https://academic.oup.com/jamia/article/25/8/945/4956468
- [BIB3.3.2-8] "Transfer Learning". Accessed August 17, 2021, https://www.tensorflow.org/tutorials/images/transfer_learning.
- [BIB3.5.3-2] "Semantic Segmentation." Papers with Code, accessed February 21, 2021, https://paperswithcode.com/task/semantic-segmentation.
- [BIB3.5.3-1] "Object Detection." Papers with Code, accessed February 21, 2021, https://paperswithcode.com/task/object-detection.
- [BIB3.8.1-1] "Radiation Dose Reporting." DICOM, accessed August 19, 2021, https://www.dicomstandard.org/using/radiation.
- [BIB3.8.2-1] "Minutes: WG-33 Data Archive and Management", DICOM, January 20, 2021, http://dicom.nema.org/dicom/minutes/wg-33/2021/WG33-2021-01-20-tcon-minutes.pdf.
- [BIB3.8.3-1] "Analysis of Optimal De-Identification Algorithms for Family Planning Data Elements." Integrating the Healthcare Enterprise, December 2, 2016,

https://www.ihe.net/uploadedFiles/Documents/ITI/IHE_ITI_WP_Analysis-of-DeID-Algorithms-for-FP-Data_Elements.pdf.

- [BIB3.8.3-2] "Health Information Privacy." HHS.gov, accessed February 28, 2021, https://www.hhs.gov/hipaa/index.html.
- [BIB3.8.3-3] "Complete Guide to GDPR Compliance." GDPR.eu, accessed February 28, 2021, https://gdpr.eu/.
- [BIB3.8.4-1] "ACR BI-RADS Atlas – Mammography", American College of Radiology, https://www.acr.org/-/media/ACR/Files/RADS/BI-RADS/Mammography-Reporting.pdf.
- [BIB3.8.5-1] "dcm2nii DICOM to NIfTI conversion." Dcm2nii, accessed February 28, 2021, https://people.cas.sc.edu/rorden/mricron/dcm2nii.html.
- [BIB3.8.9-1] Parikh, R., Mathai, A., Parikh, S., Chandra Sekhar, G., & Thomas, R. (2008). Understanding and using sensitivity, specificity, and predictive values. Indian journal of ophthalmology, 56(1), 45–50. https://doi.org/10.4103/0301-4738.37595. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2636062/
- [BIB3.8.10-1] "IHE IT Infrastructure (ITI) Domain". IHE, accessed Sept 30, 2020, https://profiles.ihe.net/ITI/.
- [BIB4.2-1] "Introduction to Data in Machine Learning". GeeksforGeeks, May 17, 2020. https://www.geeksforgeeks.org/ml-introduction-data-machine-learning/.
- [BIB4.2-2] "What is annotation in machine learning?" Quora. https://www.quora.com/What-is-annotation-in-machine-learning
- [BIB4.2-3] "What is Data Annotation?" Appen, July 10, 2020, https://appen.com/blog/data-annotation/
- [BIB4.3.1] "Framing: Key ML Terminology." Machine Learning Crash Course, Google. https://developers.google.com/machine-learning/crash-course/framing/ml-terminology.
- [BIB3.4.1] "Comparing Machine Learning as a Service: Amazon, Microsoft Azure, Google Cloud AI, IBM Watson." Altexsoft, Sept 27, 2019. https://www.altexsoft.com/blog/datascience/comparing-machine-learning-as-a-service-amazon-microsoft-azure-google-cloud-ai-ibm-watson/
- [BIB4.7.2-1] "Learn more about all the notebooks experiences from Microsoft and GitHub." Microsoft. https://notebooks.azure.com/.
- [BIB4.7.2-2] "AI Platform Notebooks." Google. https://cloud.google.com/ai-platform-notebooks
- [BIB4.7.2-3] "Jupyter." Jupyter. https://jupyter.org/
- [BIB4.7.2-4] "Amazon Sagemaker." Amazon. https://aws.amazon.com/sagemaker/

- [BIB4.7.2-5] “IBM Watson Studio”. IBM. https://www.ibm.com/ca-en/cloud/watson-studio
- [BIB4.7.2-6] “Use Notebooks with Azure Machine Learning”. Use a Jupyter Notebook with Microsoft offerings, Microsoft. https://docs.microsoft.com/en-us/azure/notebooks/quickstart-export-jupyter-notebook-project#use-notebooks-with-azure-machine-learning.
- [BIBA.1-1] “Computer Vision.” Papers with Code, accessed February 28, 2021, https://paperswithcode.com/area/computer-vision.
- [BIBA.1-2], “What is COCO?” Common Objects in Context, accessed February 28, 2021, https://cocodataset.org/#home.
- [BIBA.1-3] Hosny, A., Parmar, C., Quackenbush, J. et al. Artificial intelligence in radiology. Nat Rev Cancer 18, 500–510 (2018). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6268174/
- [BIBA.1-4] Mazurowski, Maciej A et al. “Deep learning in radiology: An overview of the concepts and a survey of the state of the art with focus on MRI.” Journal of magnetic resonance imaging: JMRI vol. 49,4 (2019): 939-954. doi:10.1002/jmri.26534. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6483404/
- [BIBA.1-5] “End-to-end Object Detection with Transformers.” Facebook AI, May 27, 2020, https://ai.facebook.com/research/publications/end-to-end-object-detection-with-transformers.
- [BIBA.1-10] “What is a Neural Network?” DeepAI, accessed February 28, 2021, https://deepai.org/machine-learning-glossary-and-terms/neural-network.
- [BIBA.1-11] “A Gentle Introduction to Transfer Learning for Deep Learning” Machine Learning Mastery, September 16, 2019, https://machinelearningmastery.com/transfer-learning-for-deep-learning/.
- [BIBA.1-12] Mohamed Elgendy. “Deep Learning for Vision Systems”, Chapter 6. https://www.manning.com/books/deep-learning-for-vision-systems
- [BIBA.1-13] “Weights and Biases.” AI Wiki, Accessed February 28, 2021, https://docs.paperspace.com/machine-learning/wiki/weights-and-biases.
- [BIBA.1-14] Will Kenton. “Overfitting.” Investopedia, November 28, 2020, https://www.investopedia.com/terms/o/overfitting.asp.
- [BIBA.1-15] Jaspreet. “Understanding and Reducing Bias in Machine Learning.” April 5, 2019, https://towardsdatascience.com/understanding-and-reducing-bias-in-machine-learning-6565e23900ac.
- [BIBA.1-16] Jason Brownlee. “Gentle Introduction to the Bias-Variance Trade-Off in Machine Learning.” Machine Learning Mastery, October 25, 2019,

https://machinelearningmastery.com/gentle-introduction-to-the-bias-variance-trade-off-in-machine-learning/.

- [BIBA.2-1] “Descending into ML: Training and Loss.” Machine Learning Crash Course, Google, Accessed February 28, 2021, https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss.
- [BIBA.2-2] “Fine-Tuning.” Dive into Deep Learning, accessed February 28, 2021, https://d2l.ai/chapter_computer-vision/fine-tuning.html.
- [BIBA.2-3] Brisimi TS, Chen R, Mela T, Olshevsky A, Paschalidis IC, Shi W. Federated learning of predictive models from federated Electronic Health Records. Int J Med Inform. 2018 Apr; 112:59-67. DOI: 10.1016/j.ijmedinf.2018.01.007. Epub 2018 Jan 12. PMID: 29500022; PMCID: PMC5836813. https://pubmed.ncbi.nlm.nih.gov/29500022/
- [BIBA.3-1] Geoff Currie, K. Elizabeth Hawk, Eric Rohren, Alanna Vial, Ran Klein. “Machine Learning and Deep Learning in Medical Imaging: Intelligent Imaging.” Journal of Medical Imaging and Radiation Sciences. October 7, 2019. Vol 50 Issue 4 P477-487. https://www.jmirs.org/article/S1939-8654(19)30504-1/fulltext
- [BIBA.3-2] Barret Zoph and Golnaz Ghiasi and Tsung-Yi Lin and Yin Cui and Hanxiao Liu and Ekin D. Cubuk and Quoc V. Le. “Rethinking Pre-training and Self-training.” 2020. https://arxiv.org/abs/2006.06882
- [BIBA.3-3] Zongwei Zhou and Vatsal Sodha and Md Mahfuzur Rahman Siddiquee and Ruibin Feng and Nima Tajbakhsh and Michael B. Gotway and Jianming Liang. “Models Genesis: Generic Autodidactic Models for 3D Medical Image Analysis”. 2019. https://arxiv.org/abs/1908.06912
- [BIBA.5-1] “Proposed Regulatory Framework for Modifications to Artificial Intelligence/Machine Learning (AI/ML)-Based Software as a Medical Device (SaMD).” FDA, https://www.fda.gov/files/medical%20devices/published/US-FDA-Artificial-Intelligence-and-Machine-Learning-Discussion-Paper.pdf
- [BIBB-1] “Design Controls”, Joseph Tartal, FDA. https://www.fda.gov/media/116762/download

D.2 Recommended Reading

The following papers were recommended by contributors to this white paper.

D.2.1 Medical Imaging AI Papers

- Mongan, Moy, Kahn. Checklist for Artificial Intelligence in Medical Imaging (CLAIM): A Guide for Authors and Reviewers, Radiology: Artificial Intelligence, Vol. 2, No. 2, https://pubs.rsna.org/doi/10.1148/ryai.2020200029
- Ranschaert, Erik R., Morozov, Sergey, Algra, Paul R. Artificial Intelligence in Medical Imaging. https://www.springer.com/gp/book/9783319948775

- J. Raymond Geis, Adrian P. Brady, Carol C. Wu, et al. Ethics of Artificial Intelligence in Radiology: Summary of the Joint European and North American Multi-Society Statement. https://pubs.rsna.org/doi/full/10.1148/radiol.2019191586
- Martin J. Willemink, MD, PhD, Wojciech A. Koszek, MS, Cailin Hardell, MS, et al. "Preparing Medical Imaging Data for Machine Learning.", Radiology, April 2020, pp 4-15, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7104701/.
- Levine AB, Schlosser C, Grewal J, Coope R, Jones SJM, Yip S. Rise of the Machines: Advances in Deep Learning for Cancer Diagnosis. Trends Cancer. 2019 Mar;5(3):157-169. DOI: 10.1016/j.trecan.2019.02.002. Epub 2019 Feb 28. PMID: 30898263. https://pubmed.ncbi.nlm.nih.gov/30898263/
- Kohli, M.D., Summers, R.M. & Geis, J. Medical Image Data and Datasets in the Era of Machine Learning—Whitepaper from the 2016 C-MIMI Meeting Dataset Session. J Digit Imaging 30, 392–399 (2017). https://doi.org/10.1007/s10278-017-9976-3
- Marc Kohli, Luciano M. Prevedello, Ross W. Filice and J. Raymond Geis Implementing Machine Learning in Radiology Practice and Research: American Journal of Roentgenology: Vol. 208, No. 4 (AJR) https://www.ajronline.org/doi/full/10.2214/AJR.16.17224
- Curtis P. Langlotz, MD, PhD Bibb Allen, MD Bradley J. Erickson, MD, PhD, et al, A Roadmap for Foundational Research on Artificial Intelligence in Medical Imaging: From the 2018 NIH/RSNA/ACR/The Academy Workshop Radiology 2019; 291:781–791 https://doi.org/10.1148/radiol.2019190613
- Harvey H., Glocker B. (2019) A Standardized Approach for Preparing Imaging Data for Machine Learning Tasks in Radiology. In: Ranschaert E., Morozov S., Algra P. (eds) Artificial Intelligence in Medical Imaging. Springer, Cham. https://doi.org/10.1007/978-3-319-94878-2_6
- Li, Tian and Sahu, Anit Kumar and Talwalkar, Ameet and Smith, Virginia. "Federated Learning: Challenges, Methods, and Future Directions." IEEE Signal Processing Magazine, Vol 37 No 3 May 2020. https://arxiv.org/abs/1908.07873
- Xue-Li Du, Wen-Bo Li, Bo-Jie Hu. Application of artificial intelligence in ophthalmology, Sep 2018. https://pubmed.ncbi.nlm.nih.gov/30225234/
- Linda Roach, Artificial Intelligence: An overview of the field with selected applications in ophthalmology. Nov 2017, Eyenet Magazine. https://www.aao.org/eyenet/article/artificial-intelligence

D.2.2 Medical Imaging AI Websites

- "Federated Learning: Collaborative Machine Learning without Centralized Training Data." Google, April 2017, https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

- "Artificial Intelligence (AI) and Machine Learning (ML) in Medical Devices." FDA, October 20, 2020, https://www.fda.gov/media/142998/download.
- "Request for Information (RFI) in the Federal Register." Federal Register, https://www.federalregister.gov/documents/2019/05/01/2019-08818/artificial-intelligence-standards
- "New guidance for AI in screening." Public Health England, March 14, 2019. https://phescreening.blog.gov.uk/2019/03/14/new-guidance-for-ai-in-screening/
- "ACR's Platform-Model Communication for AI." American College of Radiology, Accessed February 27, 2021, https://www.acrdsi.org/-/media/DSI/Files/ACR-DSI-Model-API.pdf.

D.2.3 Deep Learning Books and Papers

- Ian Goodfellow, Yoshua Bengio, Aaron Courville. "Deep Learning". MIT Press, 2016. https://www.deeplearningbook.org/
- Christopher Bishop. "Pattern Recognition and Machine Learning". Springer, 2006. https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf
- Pullum, Laura, Steed, Chad, Jha, Sumit Kumar, and Ramanathan, Arvind. Mathematically Rigorous Verification & Validation of Scientific Machine Learning. United States: N. p., 2018. https://www.osti.gov/servlets/purl/1474476
- Mohamed Elgendy. "Deep Learning for Vision Systems". Chapter 6 Transfer Learning. October 2020. https://www.manning.com/books/deep-learning-for-vision-systems
- R.J. Brooks, A.M. Tobias. "Choosing the best model: Level of detail, complexity, and model performance." Mathematical and Computer Modelling, Volume 24, Issue 4, 1996, Pages 1-14, ISSN 0895-7177. https://www.sciencedirect.com/science/article/pii/0895717796001033

D.2.4 Deep Learning Websites

- "Papers with Code." Papers with Code, accessed February 21, 2021, https://paperswithcode.com/.
- "Fine-Tuning." Dive into Deep Learning. https://d2l.ai/chapter_computer-vision/fine-tuning.html
- "Object Detection." Papers with Code, https://paperswithcode.com/task/object-detection
- "What is a machine learning model?" Microsoft. https://docs.microsoft.com/en-us/windows/ai/windows-ml/what-is-a-machine-learning-model

D.3. Inventory of Standards

The following standards were discussed during the development of this white paper. The list is not comprehensive. The job of assessing the appropriateness of the standards will be part of the development of any profiles based on this white paper. Additional standards may be identified and during that process.

D.3.1. IHE Profiles

IHE Radiology Profiles

- [STD-AIR] "AI Results (AIR)." Integrating the Healthcare Enterprise, July 16, 2020, https://www.ihe.net/uploadedFiles/Documents/Radiology/IHE_RAD_Suppl_AIR.pdf
- [STD-AIW-I] "AI Workflow for Imaging (AIW-I)." Integrating the Healthcare Enterprise, August 6, 2020, https://www.ihe.net/uploadedFiles/Documents/Radiology/IHE_RAD_Suppl_AIW-I.pdf.
- [STD-FUNC] "Follow-Up of Non-Critical Actionable Findings (FUNC)." Integrating the Healthcare Enterprise, August 11, 2017, https://docs.google.com/document/d/1pEQAIWDuD0HPQisBLlzF_FaovG8aWuIKgl163I3kr8E/edit.
- [STD-CDS-OAT] "Clinical Decision Support Order Appropriateness Tracking (CDS-OAT)." Integrating the Healthcare Enterprise, April 25, 2019, https://www.ihe.net/uploadedFiles/Documents/Radiology/IHE_Rad_Suppl_CDS-OAT.pdf.
- [STD-SOLE] "Standardized Operational Log of Events (SOLE)." Integrating the Healthcare Enterprise, July 27, 2018, https://www.ihe.net/uploadedFiles/Documents/Radiology/IHE_RAD_Suppl_SOLE.pdf.
- [STD-TCE] "Teaching File and Clinical Trial Export (TCE)." Integrating the Healthcare Enterprise, September 14, 2020, https://wiki.ihe.net/index.php/Teaching_File_and_Clinical_Trial_Export.

IHE IT Infrastructure Profiles

- [STD-XDS] "Cross-Enterprise Document Sharing (XDS)." Integrating the Healthcare Enterprise, June 20, 2019, https://wiki.ihe.net/index.php/Cross-Enterprise_Document_Sharing.
- [STD-ATNA] "Audit Trail and Node Authentication (ATNA)." Integrating the Healthcare Enterprise, March 19, 2020, https://wiki.ihe.net/index.php/Audit_Trail_and_Node_Authentication.
- [STD-PIX] "Patient Identifier Cross-Referencing (PIX)." Integrating the Healthcare Enterprise, July 2, 2018, https://wiki.ihe.net/index.php/Patient_Identifier_Cross-Referencing.

IHE Quality, Research and Public Health Profiles

- [STD-SDC] "Structured Data Capture (SDC)." Integrating the Healthcare Enterprise, March 19, 2019, https://ihe.net/uploadedFiles/Documents/QRPH/IHE_QRPH_Suppl_SDC.pdf.

IHE Patient Care Coordination Profiles

- [STD-DCTM] "Dynamic Care Team Management (DCTM)." Integrating the Healthcare Enterprise, December 11, 2020, https://wiki.ihe.net/index.php/Dynamic_Care_Team_Management_(DCTM).

D.3.2. DICOM Standards

- [STD-DICOM] "Digital Imaging and Communications in Medicine." DICOM, https://www.dicomstandard.org/.
- [STD-DICOMWEB] "DICOMweb." DICOM, https://www.dicomstandard.org/dicomweb
- [STD-DICOM-DEIDENT] "Attribute Confidentiality Profiles." DICOM PS3.15 E, http://dicom.nema.org/medical/dicom/current/output/chtml/part15/chapter_E.html
- [STD-DICOM-MPPS] "Modality Performed Procedure Step Information Object Definition." DICOM PS3.3 B.17, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_B.17.html
- [STD-DICOMWEB-UPSRS] "Worklist Services and Resources." DICOM PS3.18 11, http://dicom.nema.org/medical/dicom/current/output/chtml/part18/chapter_11.html
- [STD-DICOMWEB-ERRORS] "Status Codes" DICOM PS3.18 8.5, http://dicom.nema.org/medical/dicom/current/output/html/part18.html#sect_8.5
- [STD-DICOM-MWL] "Modality Worklist SOP Class." DICOM PS3.4 K.6.1, http://dicom.nema.org/medical/dicom/current/output/chtml/part04/sect_K.6.html#sect_K.6.1
- [STD-DICOM-SR] "Structured Report Document Information Object Definitions." DICOM PS3.3 A.35, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_A.35.html
- [STD-DICOM-SEG] "Segmentation IOD." DICOM PS3.3 A.51, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_A.51.html
- [STD-DICOM-PM] "Parametric Map IOD." DICOM PS3.3 A.75, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_A.75.html
- [STD-DICOM-KOS] "Key Object Selection Modules." DICOM PS3.3 C.17.6, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_C.17.6.html

- [STD-DICOM-RTSS] "RT Structure Set IOD." DICOM PS3.3 A.19, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_A.19.html
- [STD-DICOM-SSEG] "Surface Segmentation IOD." DICOM PS3.3 A.57, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_A.57.html
- [STD-DICOM-SECCAP] "Secondary Capture Modules." DICOM PS3.3 C.8.6, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_C.8.6.html
- [STD-DICOM-GSPS] "Greyscale Softcopy Presentation State IOD", DICOM PS3.3 A.33.1, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_A.33.html#sect_A.33.1
- [STD-DICOM-JSONSR] "Sup 219 – JSON Representation of DICOM Structured Reports", DICOM, https://www.dicomstandard.org/News-dir/ftsup/docs/sups/Sup219.pdf
- [STD-DICOM-NULL] "Null Flavor", DICOM, http://dicom.nema.org/medical/dicom/current/output/chtml/part20/sect_5.3.2.html.
- [STD-DICOM-IAN] "Instance Availability Notification Module", DICOM, http://dicom.nema.org/medical/dicom/current/output/chtml/part03/sect_C.4.23.html.
- [STD-DICOM-SVCDISCOVER] "DNS Service Discovery", DICOM, http://dicom.nema.org/medical/dicom/current/output/chtml/part15/sect_H.2.html.
- [STD-DICOM-PROTOCOL] "Procedure Protocol Information Object Definition", DICOM, http://dicom.nema.org/medical/dicom/current/output/html/part03.html#sect_A.82.
- [STD-DICOM-DISCOVERY] "Sup224: Service Discovery and Control", DICOM, ftp://dicom.nema.org/MEDICAL/Private/Dicom/WORKGRPS/Wg06/2021/2021-06-21/Sups/Sup224_ServiceDiscoveryControl/dicom-wg-23-supplement-service_discovery_and_control.docx

D.3.3. HL7 Standards

- [STD-HL7] "HL7 Version 2 Product Suite." HL7, https://www.hl7.org/implement/standards/product_brief.cfm?product_id=185
- [STD-FHIR] "Fast Healthcare Interoperability Resources." HL7, https://www.hl7.org/fhir/overview.html
- [STD-FHIR-OBS] "Observations." FHIR, HL7, https://www.hl7.org/fhir/observations.html.
- [STD-FHIR-DXR] "DiagnosticReport." FHIR, HL7, https://www.hl7.org/fhir/diagnosticreport.html.
- [STD-FHIR-PT] "Patient." FHIR, HL7, https://www.hl7.org/fhir/patient.html.

- [STD-FHIR-APPT] “Appointment.” FHIR, HL7, https://www.hl7.org/fhir/appointment.html.
- [STD-FHIR-SCHED] “Schedule.” FHIR, HL7, https://www.hl7.org/fhir/schedule.html.
- [STD-FHIR-SVCREQ] “ServiceRequest.” FHIR, HL7, https://www.hl7.org/fhir/servicerequest.html.
- [STD-FHIR-PROC] “Procedure.” FHIR, HL7, https://www.hl7.org/fhir/procedure.html.
- [STD-FHIR-CAREPLAN] “CarePlan.” FHIR, HL7, https://www.hl7.org/fhir/careplan.html.
- [STD-FHIR-COND] “Condition.” FHIR, HL7, https://www.hl7.org/fhir/condition.html.

D.3.4. Other References

- [STD-CPT] “CPT Codes Lookup”, Codify by AAPC, Accessed February 28, 2021, https://www.aapc.com/codes/cpt-codes-range/.
- [STD-ICD] “International Statistical Classification of Diseases and Related Health Problems”, World Health Organization, Accessed February 28, 2021, https://www.who.int/standards/classifications/classification-of-diseases.
- [STD-SNOMED] “SNOMED International”, SNOMED, Accessed February 28, 2021, https://www.snomed.org/.
- [STD-LOINC] “LOINC”, LOINC, Accessed February 28, 2021, https://loinc.org/.
- [STD-ACR-LUNG] “Lung CT Screening Reporting & Data System”, American College of Radiology, Accessed March 4, 2021, https://www.acr.org/Clinical-Resources/Reporting-and-Data-Systems/Lung-Rads.
- [STD-ACR-BIRAD] “ACR BI-RADS Atlas 5th Edition”, American College of Radiology, Accessed March 4, 2021, https://www.acr.org/Clinical-Resources/Reporting-and-Data-Systems/Bi-Rads.
- [STD-RADLEX] “RadLex Term Browser”, RSNA Informatics, Accessed August 19, 2021, http://radlex.org/.
- [STD-FMA] “Foundational Model of Anatomy”, University of Washington School of Medicine, Accessed February 28, 2021, http://sig.biostr.washington.edu/projects/fm/AboutFM.html.

D.4. Reference Toolkits

The following toolkits were recommended by contributors of this white paper. There was no intent to make this list comprehensive. No commercial tools were included. These tools may inform use cases and future profile development. Feel free to suggest others.

Deep Learning Toolkits

- MXNet: https://mxnet.apache.org/
- PyTorch: https://pytorch.org/
    - MONAI: https://monai.io/
- TensorFlow: https://www.tensorflow.org/
    - DLTK: https://dltk.github.io/

Model Formats

- HDF5: (https://www.hdfgroup.org/solutions/hdf5/)
- ONNX: (https://onnx.ai/)
- Python pickle used by PyTorch: (https://pytorch.org/tutorials/beginner/saving_loading_models.html, https://docs.python.org/3/library/pickle.html).
- TensorFlow "SavedModel": (https://www.tensorflow.org/guide/saved_model)
- Model Card Toolkit: https://github.com/tensorflow/model-card-toolkit
- TensorFlow lite and "xxd": https://www.tensorflow.org/lite/microcontrollers/build_convert
- TorchScript: https://pytorch.org/tutorials/advanced/cpp_export.html

Model Sources

- GitHub: https://github.com/.
- Kaggle: https://www.kaggle.com/.
- Model Zoo: https://modelzoo.co/.
- Papers with Code: https://paperswithcode.com/.

De-Identification

- PixelMed DICOM Cleaner: http://www.dclunie.com/pixelmed/software/webstart/DicomCleanerUsage.html