Maintenance Management Information Exchange Model (MMIXM)

Modeling Guidelines for MMIXM Release Version 3.0.0

11 August 2023

MMIXM Development Team

Prepared for:
Federal Aviation Administration
CLMRS Portfolio Programs (AJM-233)
800 Independence Avenue
Washington, D.C. 20591

U.S. Department of Transportation
John A. Volpe National Transportation Systems Center

Volpe

Notice

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for the contents or use thereof.

The United States Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the objective of this report.

Date	MMIXM Release Version	Description of Revision
1 Sept 2017	1.0.0	MMIXM v1.0.0 Official Release version
31 March 2020	2.0.0	MMIXM v2.0.0 Official Release version
31 Dec 2021	2.1.0	MMIXM v2.1.0 Official Release version
11 August 2023	3.0.0	MMIXM v3.0.0 Official Release version

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 MMIXM Background

The Federal Aviation Administration (FAA) is responsible for maintaining National Airspace System (NAS) assets. Data pertaining to these assets, along with maintenance-related coordination and transaction histories, and technician training and qualifications, are managed via numerous disparate systems and databases. These systems often employ vendor-specific technology and data formats to manage data. Furthermore, organizations may use different nomenclature, or use different identifiers, to uniquely identify an asset. These differences in data formats, nomenclature and semantics pose a challenge to effectively share data between systems and stakeholders. As such, sharing of data in the current environment often involves manual processes. The FAA is addressing similar data exchange challenges in other domains by employing standardized data models. These standards include AIXM (Aeronautical Information Exchange), FIXM (Flight Information Exchange), and IWXXM (Weather Information Exchange). However, there is no equivalent standard for standardizing data within the FAA Operations and Maintenance (O&M) environment.

The Maintenance Management Information Exchange Model (MMIXM) is a data standard for exchanging information between various FAA O&M systems. The current version represents the version 3.0.0 release.

1.2 MMIXM Scope

There are a large number of FAA O&M systems and stakeholders. MMIXM v3.0.0 focuses on a subset of these O&M systems. Table 1 below captures the MMIXM Asset Management concept area, in scope FAA systems, level of maturity for implementing the system's requirements, and additional notes about implementation status. The level of maturity in the table represents the MMIXM Development Team's estimate based on knowledge in that concept area and requirements implemented in the model.

MMIXM Asset Management FAA Systems	***Level of Maturity***
Automated Maintenance Management System (AMMS)	Implementation of AMMS needs are largely based on AMMS FID system requirements. Maintenance logging, expected to be consumed by AMMS, is not currently included in the scope of MMIXM.
Facility, Service, and Equipment Profile (FSEP)	FSEP was one of the first FAA systems evaluated by the MMIXM Development Team. Since then, FSEP has been incorporated into the Remote Monitoring and Logging System (RMLS). RMLS maintains the original FSEP asset hierarchy. Likewise, the original FSEP data requirements remain unchanged in MMIXM.
Remote Monitoring and Logging System (RMLS) National Logging Network (NLN)	RMLS NLN consumed FSEP in 2019. Current MMIXM implementation considers the asset portion of RMLS.

Automated Inventory Tracking System (AITS)	The implementation of AITS requirements is largely based on documentation and system research.
Logistics Center Support System (LCSS)	The implementation of LCSS requirements is largely based on documentation and system research.

Table 1: MMIXM Asset Management Systems and Level of Maturity

MMIXM Qualification (Certifications and Credentials) FAA Systems	***Level of Maturity***
Air Traffic Safety Oversight Service (AOV)	MMIXM Team worked extensively with AOV and RMLS to define elements required an AOV-RLMS data exchange.
Learning Management System (LMS)	MMIXM Team worked extensively with LMS and RMLS to define elements required an LMS-RLMS data exchange.

Table 2: MMIXM Qualification (Certifications and Credentials) Systems and Level of Maturity

MMIXM Monitoring FAA Systems	***Level of Maturity***
Enterprise Service Monitoring (ESM)	MMIXM Team worked extensively with ESM (and tangentially with ESM stakeholders) to define elements required for enterprise service monitoring use cases.
National Remote Maintenance Monitoring (RMM) Network (NRN)	RMM NRN uses Logical Units and Data Points to monitor lower-level asset components. The MMIXM model supports RMM NRN use cases, including: • Associating the RMLS Logical Unit Identifier (Hex ID) with asset identifiers • Associating RMLS Data Point information to a Monitoring Event • Associating more than one Monitoring Event to an asset.
Communications Remote Maintenance Monitoring (CRMM)	CRMM is currently testing MMIXM v3.0.0 for use transmitting monitoring and telemetry data.

Table 3: MMIXM Monitoring Systems and Level of Maturity

1.3 MMIXM Modeling Assumptions and Constraints

One of the many goals of the MMIXM model and development process is to accommodate current data exchange use cases while also accommodating the future evolution of the FAA maintenance data exchange needs. Below are a list of foundational modeling assumptions and constraints that helped to guide the development of MMIXM to date.

- The MMIXM v3.0.0 modeling effort focuses on the systems previously described in the Scope section.

- The MMIXM v3.0.0 Logical Model is largely based on the current structure of TechOps systems[1] for maintenance and monitoring.
- Development continues to harmonize FSEP capabilities/work areas and AOV credential ratings. MMIXM v3.0.0 accommodates the proposed harmonization described in the Order 6000.15J, General Maintenance Handbook for NAS Facilities.
- AOV credentials and LMS certifications status elements should be based on the current RMLS-AOV-LMS integration requirements defined in AMMS.
- The MMIXM v3.0.0 Logical Model does not incorporate proposed scheduling applications and associated data structures.
- Logging requirements, established by the RMLS NLN system, have been temporarily removed from the model due to AMMS development and the potential evolution of FAA maintenance event logging concepts. Logging requirements will be re-introduced into MMIXM when the future requirements of maintenance logging in the FAA are finalized.
- There is no consistent use of unique asset identifiers across FAA systems. MMIXM allows for the exchange of the commonly used asset identifiers currently in use by FAA systems.

1.4 MMIXM Logical Data Model

A logical data model represents an abstract structure of an information domain and is independent of the technology used for implementation. This structure includes entities, attributes, and relationships. The logical model development process includes gathering information about an organization's business requirements and business processes, for both the legacy operation as well as the future operation. The logical model should capture detailed information such as the following:

- Data types
- Data type restrictions
- Cardinality of associative relationships (i.e., zero-to-one, one-to-many)
- Navigability of associative relationships (i.e., one-way, bi-directional)
- Specification of optional and required attributes
- Definitions on data elements and associations
- Reused data elements (if available) from other logical models

1.5 MMIXM Development Process

The MMIXM Logical Model was developed using Sparx Enterprise Architect. It was developed with the Unified Modeling Language (UML), and utilizes object-oriented concepts such as classes, attributes, and associations. Business requirements were gathered from the following:

- Stakeholder interviews
- System documentation
- Sample data extracted from legacy systems
- Legacy database implementation

[1] Data exchange requirements related to the integration of AOV, CTS and RMLS were also incorporated. Additional systems outside of TechOps will be included in future MMIXM iterations.

- Stakeholder feedback on Logical Model iterations

The Logical Model serves as the basis for the Physical Model. In this case, the Physical Model schema is described with XML Schema Documents (XSD). For more details on the MMIXM development process please refer to the MMIXM Process and Governance document on the MMIXM website (https://mmixm.aero/content/documents.html).

2. Structure

Classes within the MMIXM Logical Model are organized into UML Packages. A UML package is used to group classes. Packages can be contained within other packages, and associations can be defined between them; thus, they can also be used to describe hierarchical or dependency relationships.

2.1 MMIXM Packages

A snapshot of the high-level packages in the MMIXM Logical Model are shown below.

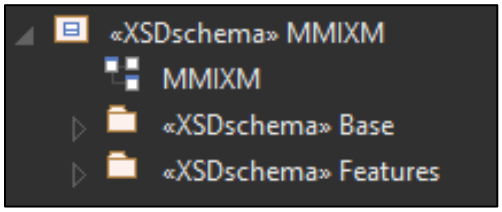

Figure 1: MMIXM High-Level Packages

The MMIXM package contains the *Base* and *Features* package. The *Base* package contains classes that are not domain-specific and could potentially be reused in other parts of the model; conversely, the *Features* package contains classes that are domain specific. This general organization of classes into Base and Features packages is similar to the organization seen in the FIXM and AIXM models as well as other industry models.

2.2 MMIXM Base Package

The *Base* package contains additional packages with the following types of classes:

- Concepts that are not domain-specific
  - Contact Information: Contains elements describing information used to contact a person or organization (e.g., mailing address, phone number, etc.).
  - Location: Contains elements describing locations (e.g., geographical coordinates, address, or even virtual locations such as IP address).
  - Measures: Contains measure types and related units of measure.
  - Message: Defines message metadata elements and provides a framework for sending a single message or message collections.
  - Person/Organization: Contains elements describing a person or an organization.
- Data type restrictions
  - Enumerations: Enumerations are a base simple type in the XSD specification containing a list of possible values.
  - Simple type restrictions: A simple type restriction restricts its base type by applying facets to restrict its values.

The sub-package structure within the *Base* package is shown below.

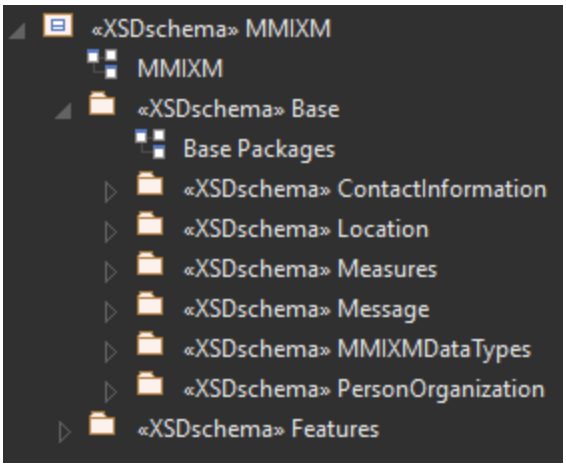

Figure 2: MMIXM "Base" Packages

2.3 MMIXM Features Package

The *Features* package contains additional packages containing the following types of domain-specific classes:

- **Asset Package:** The Asset Package accounts for data exchange requirements for the various organizations within the FAA that manage assets. In MMIXM, an asset is broadly defined to include inventoried parts, operationally deployed systems, and even enterprise web services.

- **Monitoring Package:** The Monitoring Package represents data exchange requirements for real-time (or near real-time) monitoring of an asset's state. These messages typically include information about whether the asset is operational or not.

- **Qualifications Package:** The Qualification Package focuses on data elements describing the credentials and certifications belonging to maintenance personnel.

The sub-package structure within the *Features* package is shown below.

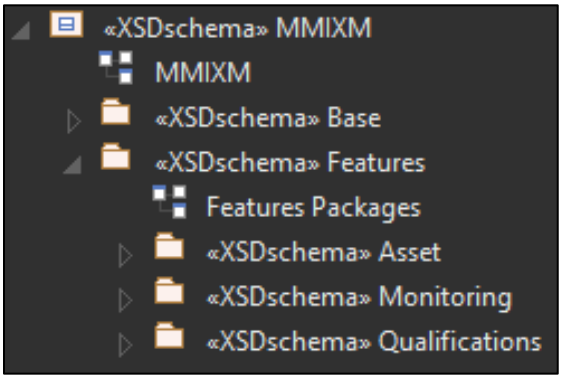

Figure 3: MMIXM "Features" Packages

3. Data Modeling Best Practices

This section describes general data modeling principles that were followed during model development, to ensure consistency.

3.1 Model Development Principles

- General Development Principles
    - Information being modeled should not be influenced by how the information is packaged, transmitted, or used.
    - Content must be in scope. All content added must trace back to a change request (CR) which was approved by the MMIXM development team with agreement from MMIXM stakeholders.
    - MMIXM should avoid unnecessary complexity. Complexity inhibits understanding and increases implementation effort. The less complex the model, the less likely information exchange results in different or ambiguous interpretation.
    - The MMIXM model should not be influenced by the concerns of a single or small group of stakeholders.
    - The MMIXM model should limit redundancy of information. Redundant information is a source of model inconsistencies that can result in misunderstanding, ambiguity, and incorrect information usage. Redundancy also increases object size and entails business rules external to the model.
    - The MMIXM model should limit element name size where appropriate.
    - MMIXM elements are organized alphabetically to improve usability and navigability of the schemas. The exception to this principle is the asset relationship elements. The MMIXM Development Team determined that those relationships should be located towards the end of the messages.
- MMIXM-Identified Development Principles
    - System names should not be used in the model if possible. However, in the FAA there are systems with specific elements that only make sense in the context of that system. In scenarios like this, using a system name is appropriate to ensure transparency for users of the model (Note: a current example of this in the model is the use of “fsep” before elements where their meaning and definition would not make sense outside the context of that system).
    - Similar to the point above, acronyms should be avoided if possible. However, there are cases in MMIXM where acronyms are used because they are common, stakeholder-recognized acronyms (e.g., FSEPs FIC, FAC, and LOCID nomenclature).
    - In the Logical Model connector target names should mirror class names where possible.
    - MMIXM, where appropriate, should borrow concepts and best practices from existing standards.
    - Additional Information elements are used throughout the model to allow flexibility for early adopters. This allows data producers to publish custom data that is unstructured. The MMIXM Development Team will continually evaluate how these elements are used in order to incorporate stakeholder data elements in future model releases.

- In many data models, abstract data structures have often been used to represent choice structures. Abstract structures, at times, have been known to be challenging to implement. MMIXM should use choice structures rather than abstract structures where it is acceptable to do so.

3.2 Naming Conventions

3.2.1 Camel Case Notation

Class names and package names are in upper camel case; this is represented as several words joined together, and the first letter of each word is capitalized. The figure below shows the classes in the Asset package, and the upper camel case representation of their associated class names.

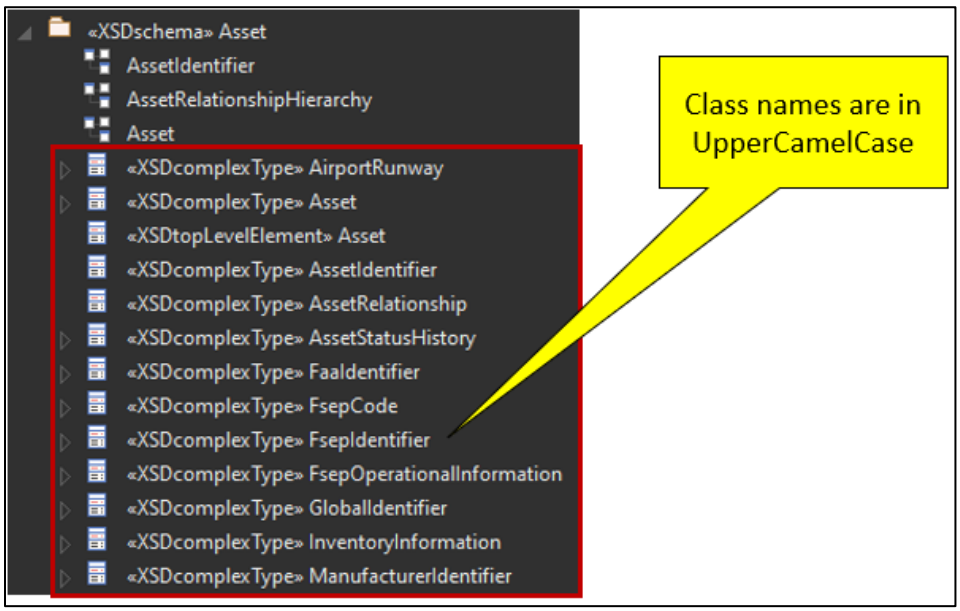

Figure 4: Examples of MMIXM Upper Camel Case Notation

The names of the attributes contained within a class are in lower camel case. This is similar to upper camel case, except that the first letter of the first word is not capitalized. If the attribute name contains more than one word, then the first letter associated with all remaining words is in upper camel case (e.g., condition versus assetInventoryStatus). Figure 5 below shows attributes of a class, and their names in lower camel case.

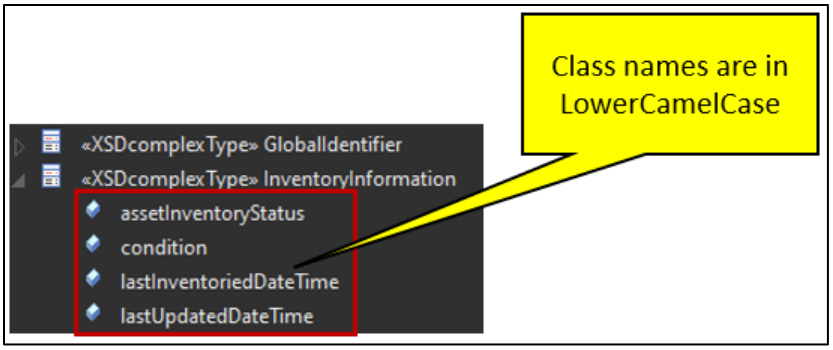

Figure 5: Examples of MMIXM Lower Camel Case Notation

3.2.2 Element Names

Element names are spelled out in full whenever practical. Exceptions are made for some acronyms if they are either very long or well-known (e.g., FAA). Figure 6 below shows spelled-out element names within Asset package.

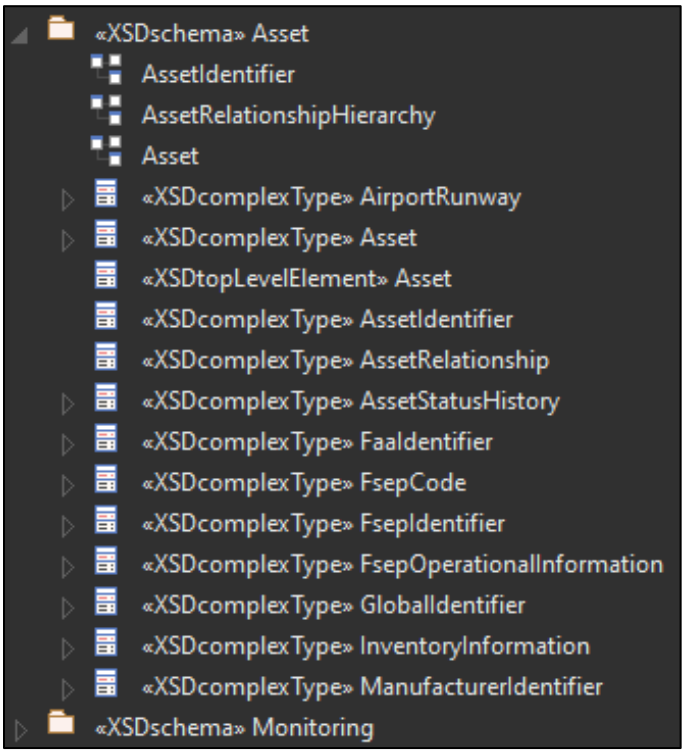

Figure 6: Examples of MMIXM Element Names

3.3 Deprecated Elements

MMIXM, being at a very early stage of development, does not have deprecated elements. As the model matures, the MMIXM Development Team will continually revaluate if a formal deprecation policy is required.

3.4 Versioning

MMIXM follows the Semantic Versioning 2.0.0 format (see https://semver.org/), with the number sequence represents MAJOR.MINOR.PATCH.

The definition of each release type for MMIXM is as follows:

- A MAJOR version introduces significant conceptual changes. Examples of changes that constitute a major release include the refactoring of a large part of the model, deletion of model elements without prior deprecation or replacement, or withdrawal of a given physical realization from the previous model release.
- A MINOR version introduces new, optional model elements, deprecation of model elements (with or without replacement), and deletion of model elements that were deprecated in the previous release.
- A PATCH version is limited to bug fixes, such as correcting spelling mistakes, clarifying definitions and documentation, and updating external references.

The aforementioned definitions of the release type are aligned with the currently-proposed approach of the AIXM/FIXM/IWXXM community, and may evolve as their approach evolves.

3.5 Namespaces

XML namespaces are used to provide uniquely named elements and attributes. Identically-named elements and attributes can therefore be resolved if each element is assigned to a separate, unique namespace. The MMIXM standard aims to maintain unique element and attribute names, regardless of namespace, thus minimizing the need for having many namespaces. Starting in MMIXM v2.1.0, MMIXM adopted a new namespace policy. MMIXM namespaces only contain the one-digit major version number rather than the full three digits (e.g., 2 rather than 2.1.0). This policy ensures backwards compatibility allowing for more predictability for stakeholders across releases.

The MMIXM v3.0.0 model currently uses the following three namespaces[2]:

- https://mmixm.aero/3
- https://mmixm.aero/base/3
- https://mmixm.aero/features/3

3.6 Data Type Restrictions

Data type restrictions, such as enumerations and XML simple type restrictions, are contained in the model. Enumerations represent an allowable set of values for a particular data element. All enumerations are represented upper camel case to align with the agreed upon convention in XML for enumerations. Simple type restrictions contain the pattern facet describing the restriction, i.e., the

[2] AIXM and FIXM have a similar number of namespaces

allowable alphanumeric characters, the length of the string, etc. The implementation of a simple type restriction is shown below for illustration.

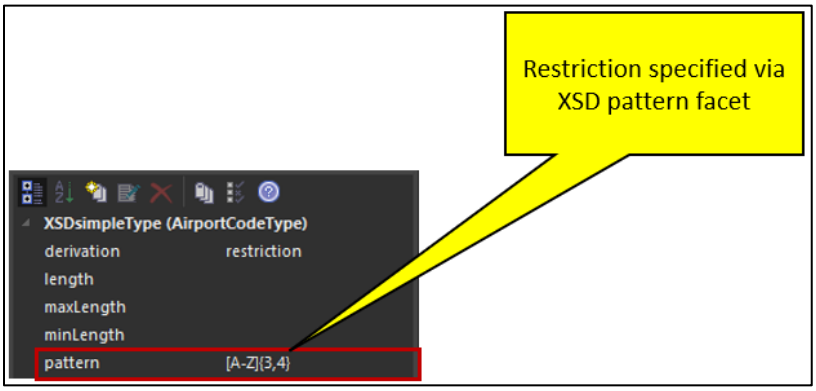

Figure 7: Example of MMIXM DataType Restriction

The MMIXM data type restrictions are contained within the *MMIXM Data Types* package.

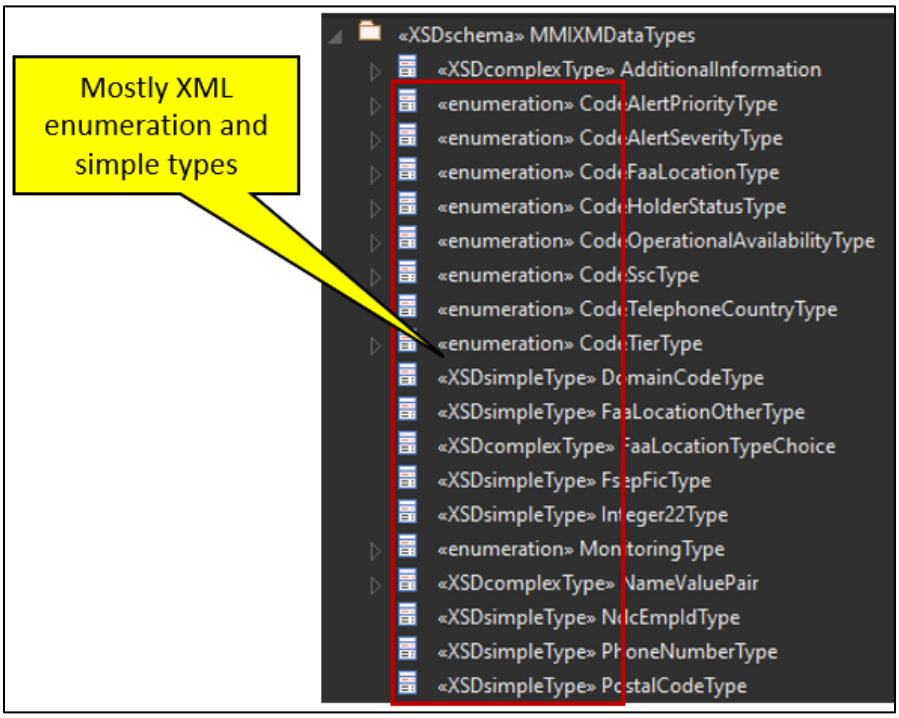

Figure 8: Example of MMIXM SimpleTypes and Enumerations

A snapshot of some of the classes within the *MMIXM Data Types* is shown above in Figure 8. AIXM and FIXM both have a similar organization: data types are contained in a dedicated package within the model.

3.7 Reuse of Existing Standards

Existing standards were considered during the development of the model. These existing standards include aviation standards such as AIXM, FIXM and IWXXM, as well as other standards such as National

Information Exchange Model (NIEM[3]), and standards maintained by the Machinery Information Management Open System Alliance (MIMOSA[4]). In many cases, the concepts within the FAA Operations and Maintenance domain were unique and thus reusing existing standards was not sufficient.

However, some implementations from existing standards were reused. An example of standards reuse is the Telephone Number. There are various database implementations of a phone number amongst the various legacy FAA O&M systems. The MMIXM model should standardize - where possible - these concepts that are not domain-specific. AIXM provides a pattern facet for constraining telephone number data elements, whereas NIEM represents a telephone number as a complex type with constituent parts (e.g., country code, telephone number, and telephone number extension). A combination of both implementations was used in MMIXM, as illustrated below.

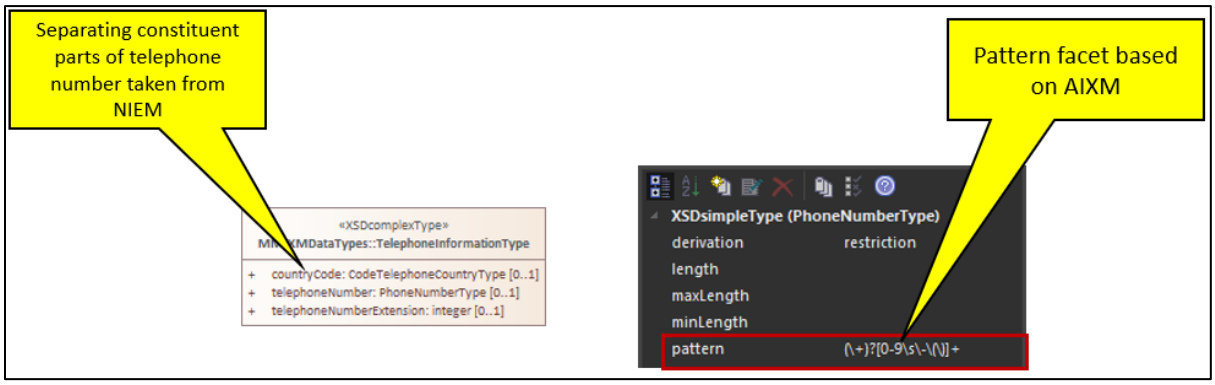

Figure 9: Example of MMIXM Simple Type Restriction using a Pattern

3.8 Optionality for Most Class Associations and Attributes

Most of the class associations and attributes in MMIXM v3.0.0 are optional. This aligns with the FIXM modeling approach[5]. The reasoning behind this approach was to provide flexibility to users during this early stage of the model. It is planned that, as more specific data exchange requirements are solicited and provided by stakeholders and users, some fields will become mandatory in future MMIXM iterations.

3.9 Standardization of Terminologies

One of the main purposes of a data standard is to provide consistent and unambiguous definitions to terminology. Heterogeneous organizations within the FAA may use different terms for the same thing -- or similar terms for different things. This makes data exchange between organizations challenging. An attempt was made to define terms consistently in the model. Work will continue between the MMIXM modeling team and FAA O&M stakeholders to increase standardization of semantics in future model releases. One area where standardization has been difficult is for terms that were originally defined by FSEP and are now part of RMLS NLN. Some of these terms when used to describe assets are not

[3] https://www.niem.gov/
[4] http://www.mimosa.org/
[5] In FIXM, all associations and attributes are optional

consistent with common FAA usage outside of the maintenance community (e.g., the term *facility*), but their understanding within the FAA maintenance community is ubiquitous. In these cases, the term Fsep is included within the data element types and data element names (see examples in the left portion of the figure below) to make this distinction clear to users and developers.

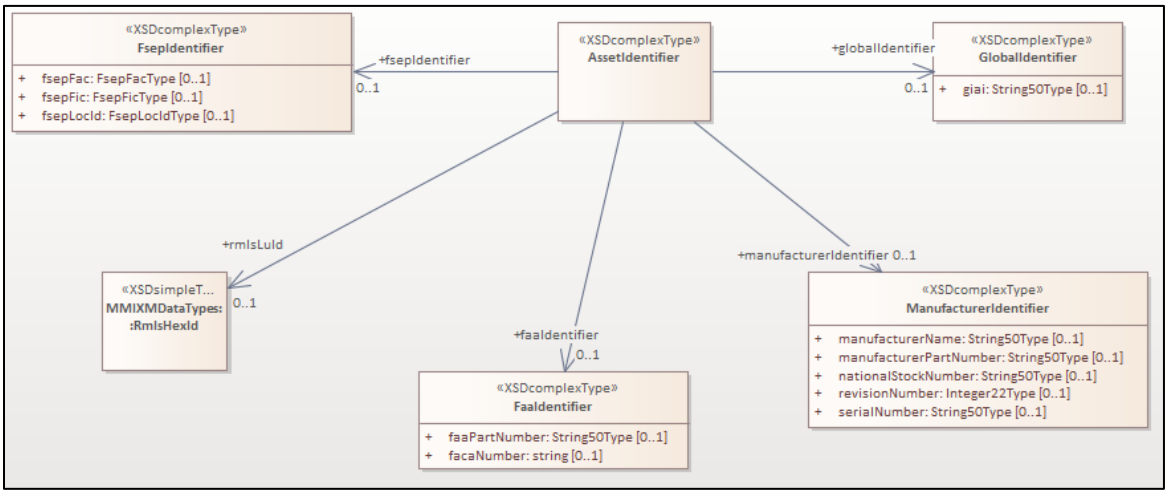

Figure 10: Example of MMIXM Use of Acronyms

3.10 Schema Extensions

There are currently no schema extensions associated with MMIXM v3.0.0. Future releases of MMIXM may accommodate extensions to the model, in which case, guidelines governing the development of extensions will be included in those future releases.

3.11 Message Wrapper

The message schema contains two high-level classes, Message and MessageCollection. The Message class allows data producers to generate information into a single message. MessageCollection class allows data producers to wrap many messages in a single message. Both Message and MessageCollection contain metadata or header information (see figure below). The message element can contain multiple features, thus providing the flexibility to publish information about multiple features in the same message.

Message elements are also used in AIXM and FIXM, hence aligning with other aviation data modeling principles.

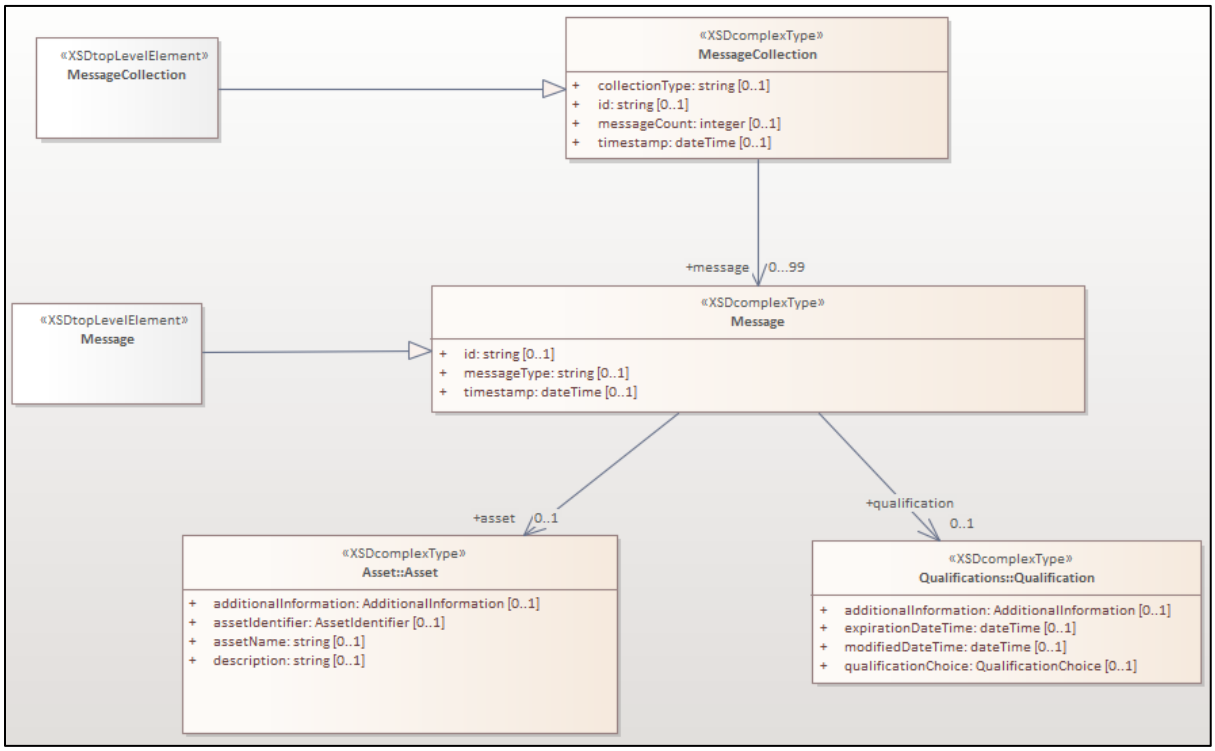

Figure 11: MMIXM Message Package

3.12 Name-Value Pairs

MMIXM provides an element that allows the publication of additional information. This additional information is presented as name-value pairs. This allows data producers to publish custom data that is unstructured. In the figure below, this is represented by the *additionalInformation* attribute in the Qualification class. The *additionalInformation* attribute is only inherent in elements that are at high levels in the XML hierarchy.

Custom data published via the name-value pair also provides the data modelers' insight into other data exchange requirements that may need to be incorporated to accommodate the needs of stakeholders. The concept of name-value pairs was also used in older versions of FIXM.

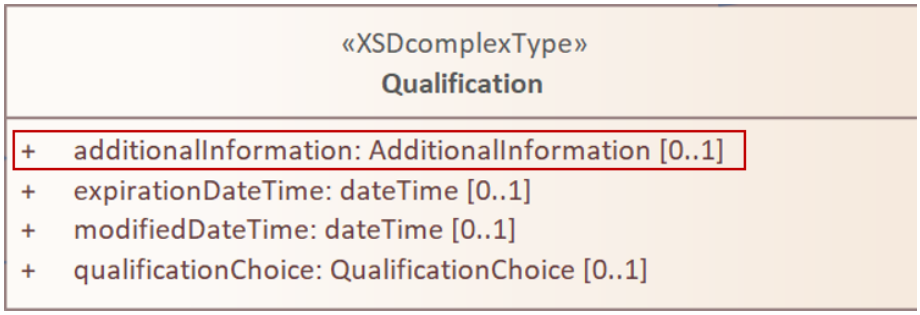

Figure 12: Example of MMIXM Name-Value Pair

3.13 Units of Measure

This package defines the units of measurement used within the MMIXM model. Units of measure are defined as simple enumerations of units appropriate to a specific measure. The data types defined are Flight Level, Distance, Volume, Weight, Speed, Electric Current, Temperature, Frequency, Power, Percentage, Electric Potential, Time, Energy, Pressure, Light Intensity, Flow Rate, Computer Information, Computer Information Rate, Angle, and Angular Rate. These data types are not specific to any one domain and could be reused in other models.

A. Acronyms

Acronym	Meaning
AITS	Automated Inventory Tracking System
AIXM	Aeronautical Information Exchange Model
AJM-233	CLMRS Portfolio Programs
AMMS	Automated Maintenance Management System
AOV	Air Traffic Safety Oversight Service
CLMRS	Configuration, Logistics, and Maintenance Resource Solutions
CR	Change Request
CRMM	Communications Remote Maintenance Monitoring
ESM	Enterprise Service Monitoring
FAA	Federal Aviation Administration
FAC	Facility Type
FIC	Facility Identification Code
FID	Final Investment Decision
FIXM	Flight Information Exchange Model
FSEP	Facility, Service, and Equipment Profile
ICAO	International Civil Aviation Organization
ID	Identifier
IP	Internet Protocol
IWXXM	ICAO Meteorological Information Exchange Model
LCSS	Logistics Center Support System
LMS	Learning Management System
LOCID	Location Identifier
MMIXM	Maintenance Management Information Exchange Model
NAS	National Airspace System
NIEM	National Information Exchange Model
NLN	National Logging Network
NRN	National Remote Maintenance Monitoring (RMM) Network
O&M	Operations and Maintenance
RMLS	Remote Monitoring and Logging System
RMM	Remote Maintenance Monitoring
UML	Unified Modeling Language
XML	eXtensible Markup Language
XSD	XML Schema Definition