Maintenance Management

Information Exchange Model (MMIXM)

Modeling Guidelines

MMIXM Development Team

**Version 1.1.1**

**14 January 2019**

Prepared for:

**Federal Aviation Administration**

**NAS Integration Support Group, Operations Support**

**800 Independence Avenue**

**Washington, D.C. 20591**

**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.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **REPORT DOCUMENTATION PAGE** | | | | *Form Approved*  *OMB No. 0704-0188* | | |
| Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503. | | | | | | |
| 1. AGENCY USE ONLY (Leave blank) | | 2. REPORT DATE  January 14, 2019 | | 3. REPORT TYPE AND DATES COVERED  Draft Report | | |
| 4. TITLE AND SUBTITLE  Maintenance Management Information Exchange Model (MMIXM) Modeling Guidelines | | | | | 5a. FUNDING NUMBERS  FB34B4 | |
| 6. AUTHOR(S)  Volpe MMIXM Development Team | | | | | 5b. CONTRACT NUMBER | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)  U.S. Department of Transportation  John A Volpe National Transportation Systems Center  Organizational Performance Division  55 Broadway  Cambridge, MA 02142-1093 | | | | | 8. PERFORMING ORGANIZATION REPORT NUMBER  N/A | |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)  Federal Aviation Administration  NAS Integration Support Group, Operations Support  800 Independence Avenue  Washington, D.C. 20591 | | | | | 10. SPONSORING/MONITORING  AGENCY REPORT NUMBER | |
| 11. SUPPLEMENTARY NOTES | | | | | | |
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT  The contents of this document may be export-controlled and are not currently available for public distribution. | | | | | 12b. DISTRIBUTION CODE | |
| 13. ABSTRACT (Maximum 200 words)  TBW | | | | | | |
| 14. SUBJECT TERMS  TBW | | | | | | 15. NUMBER OF PAGES  21 |
| 16. PRICE CODE |
| 17. SECURITY CLASSIFICATION  OF REPORT  Unclassified | 18. SECURITY CLASSIFICATION  OF THIS PAGE  Unclassified | | 19. SECURITY CLASSIFICATION  OF ABSTRACT  Unclassified | | | 20. LIMITATION OF ABSTRACT  Same As Report |
| NSN 7540-01-280-5500 |  | |  | | | Standard Form 298 (Rev. 2-89)  Prescribed by ANSI Std. 239-18  298-102 |

|  |  |  |
| --- | --- | --- |
| Date | Version | Description of Revision |
| 16 May 2017 | 1.0.0 RC | Release Candidate draft |
| 1 Sept 2017 | 1.0.0 | Official Release version |
| 29 Sept 2017 | 1.1.0 Beta | Beta Release draft |
| 8 Dec 2017 | 1.1.0 RC | Release Candidate draft |
| 16 March 2018 | 1.1.0 | Official Release version |
| 14 Jan 2019 | 1.1.1 | Official Release version |

# Table of Contents

[Table of Contents ii](#_Toc508976105)

[1. Introduction 1](#_Toc508976106)

[1.1 Background 1](#_Toc508976107)

[1.2 MMIXM Logical Data Model 1](#_Toc508976108)

[1.3 Development Process 1](#_Toc508976109)

[1.4 Scope 2](#_Toc508976110)

[2. Structure 3](#_Toc508976111)

[2.1 MMIXM Packages 3](#_Toc508976112)

[2.2 MMIXM Base Package 3](#_Toc508976113)

[2.3 MMIXM Features Package 4](#_Toc508976114)

[3. Data Modeling Principles 5](#_Toc508976115)

[3.1 Model Development Assumptions 5](#_Toc508976116)

[3.2 Naming Conventions 6](#_Toc508976117)

[3.2.1 Camel Case Notation 6](#_Toc508976118)

[3.2.2 Element Names 7](#_Toc508976119)

[3.3 Deprecated Elements 7](#_Toc508976120)

[3.4 Versioning 7](#_Toc508976121)

[3.5 Namespaces 8](#_Toc508976122)

[3.6 Data Type Restrictions 8](#_Toc508976123)

[3.7 Reuse of Existing Standards 9](#_Toc508976124)

[3.8 Optionality for Most Class Associations and Attributes 10](#_Toc508976125)

[3.9 Standardization of Terminologies 10](#_Toc508976126)

[3.10 Schema Extensions 10](#_Toc508976127)

[3.11 Message Wrapper 11](#_Toc508976128)

[3.12 Name-Value Pairs 11](#_Toc508976129)

[3.13 Abstract Classes 12](#_Toc508976130)

[3.14 Units of Measure 13](#_Toc508976131)

[4. MMIXM Logical Model (UML) Design Notes 13](#_Toc508976132)

[4.1 Asset Design Notes 13](#_Toc508976133)

[4.2 Monitoring Design Notes 13](#_Toc508976134)

[4.3 Event Coordination Design notes 13](#_Toc508976135)

[4.4 Qualification Design Notes 14](#_Toc508976136)

[4.5 Reference Material Design Notes 14](#_Toc508976137)

[5. References 15](#_Toc508976138)

# Introduction

## Background

The 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 with effectively sharing 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 WXXM (Weather Information Exchange). However, there is no equivalent standard for managing data within FAA Operations and Maintenance (O&M) systems.

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 1.1.1 release. The purpose of this release is to support early adopters and the MMIXM proof-of-concept activities.

## 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

## Development Process

The MMIXM Logical Model was developed using Enterprise Architect Version 13. 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
* 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).

## Scope

There are a large number of FAA O&M systems and stakeholders. MMIXM v1.1.1 focused on a subset of these O&M systems. This subset included:

* Facility Service Equipment Profile (FSEP)
* Remote Monitoring and Logging System National Logging Network (RMLS NLN)
* Air Traffic Safety Oversight Service (AOV)
* Automated Inventory Tracking System (AITS)
* Certification Tracking System (CTS)[[1]](#footnote-2)
* Logistics Center Support System (LCSS)
* Life Cycle Asset Tracking System (LCATS)
* Enterprise Service Monitoring (ESM)

# 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.

## MMIXM Packages

A snapshot of the high-level packages in the MMIXM Logical Model is shown in Figure 1.

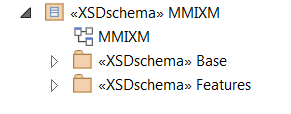


Figure . MMIXM Logical Model Packages

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.

## MMIXM Base Package

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

* Concepts that are not domain-specific
  + Location
  + Contact Information
  + Person
* Data type restrictions
  + Enumerations
  + Simple type restrictions
* Message wrapper classes

The sub-package structure within the *Base* package is shown in Figure 2.

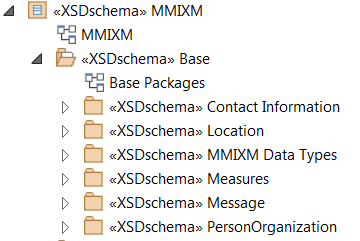


Figure . MMIXM Base Sub-Package Structure

## MMIXM Features Package

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

* Asset
* Monitoring
* Event Coordination
* Qualifications
* Reference Material

The sub-package structure within the *Features* package is shown in Figure 3.

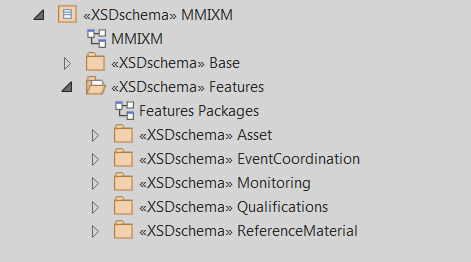


Figure . MMIXM Features Sub-Package Structure

# Data Modeling Principles

This section describes general data modeling principles that were followed during model development, to ensure consistency. These principles involve naming conventions and model organization.

## Model Development Assumptions

1. The MMIXM v1.1.1 modeling efforts should focus on the systems previously described in the *Scope* section.
2. The MMIXM v1.1.1 Logical Model should be largely based on the current structure of Tech Ops systems[[2]](#footnote-3).
3. MMIXM v1.1.1 modeling was based on a standalone FSEP system.
4. There is a known inconsistency between FSEP capabilities/work areas and AOV credential ratings. Work is underway to harmonize them. MMIXM v1.1.1 should accommodate the proposed harmonization described in the draft *Order 6000.15H, General Maintenance Handbook for NAS Facilities*.
5. AOV credentials and CTS certifications status elements should be based on the current RMLS-AOV-CTS integration requirements.
6. The MMIXM v1.1.1 Logical Model should not incorporate proposed future work scheduling applications and associated data structures.

## Naming Conventions

### 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. Figure 4 shows the classes in the Asset package, and the upper camel case representation of their associated class names.

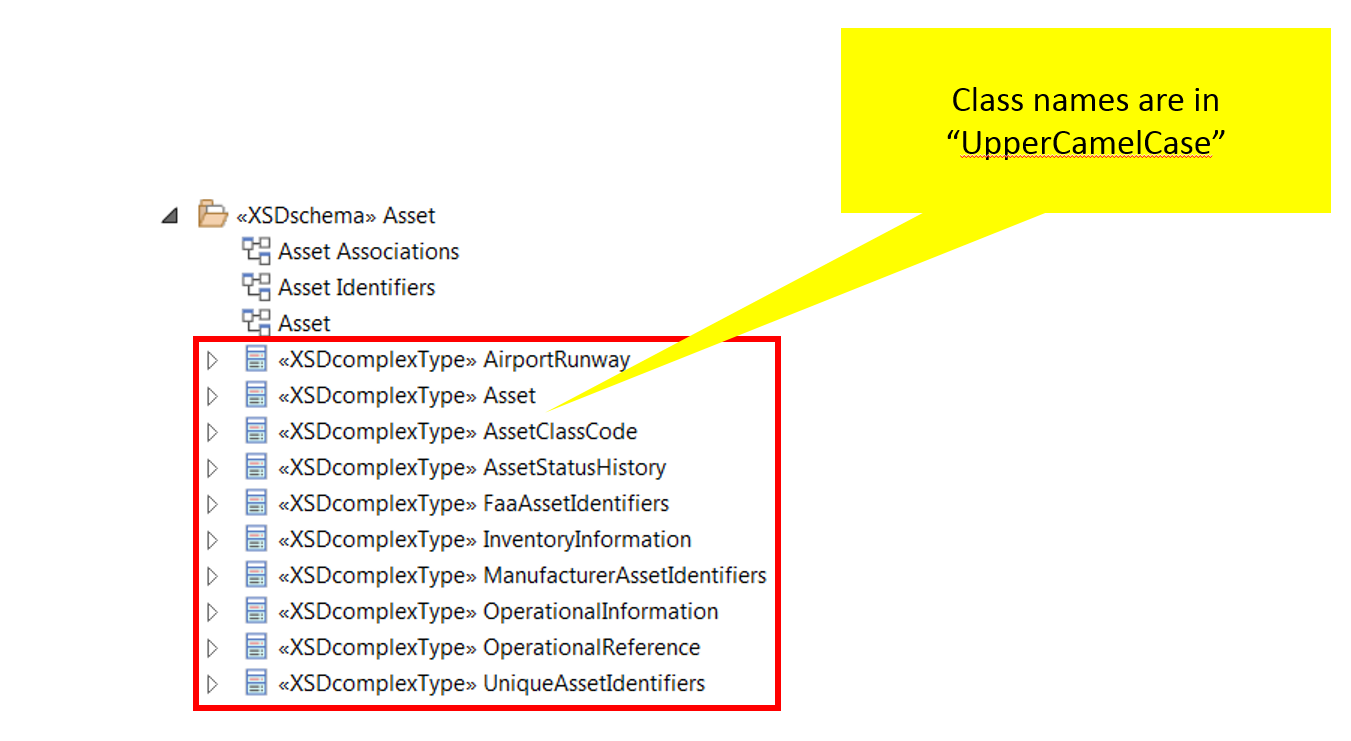


Figure . MMIXM Class Camel Case Convention

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. Figure 5 shows attributes of a class (InventoryInformation), and their names in lower camel case.

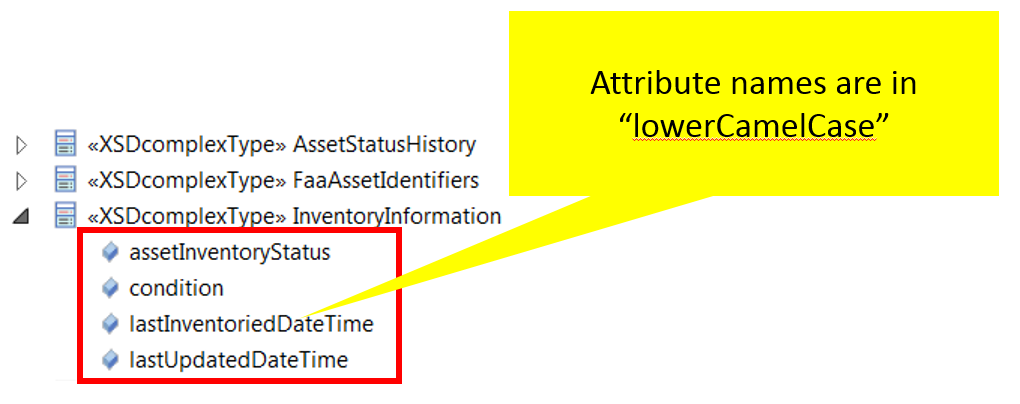


Figure . MMIXM Attribute Camel Case Convention

### Element Names

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

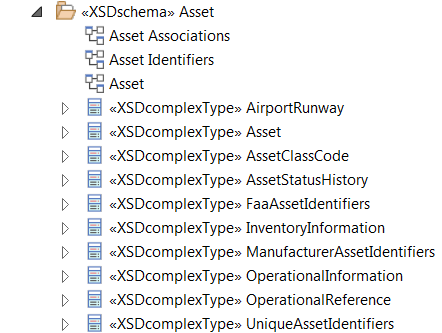


Figure . MMIXM Element Name Convention

## Deprecated Elements

MMIXM, being at a very early stage of development, does not have deprecated elements. However, as the model matures, a formal deprecation policy will be developed.

## Versioning

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

## 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. The full version number will be included in MMIXM namespaces. The MMIXM v1.1.1 model currently uses the following three namespaces[[3]](#footnote-4):

* [https://mmixm.aero/1.1.1](https://mmixm.aero/1.0.0)
* [https://mmixm.aero/base/1.1.1](https://mmixm.aero/base/1.0.0)
* [https://mmixm.aero/features/1.1.1](https://mmixm.aero/features/1.0.0)

## 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 allowable alphanumeric characters, the length of the string, etc. The implementation of a simple type restriction is shown in Figure 7 for illustration.

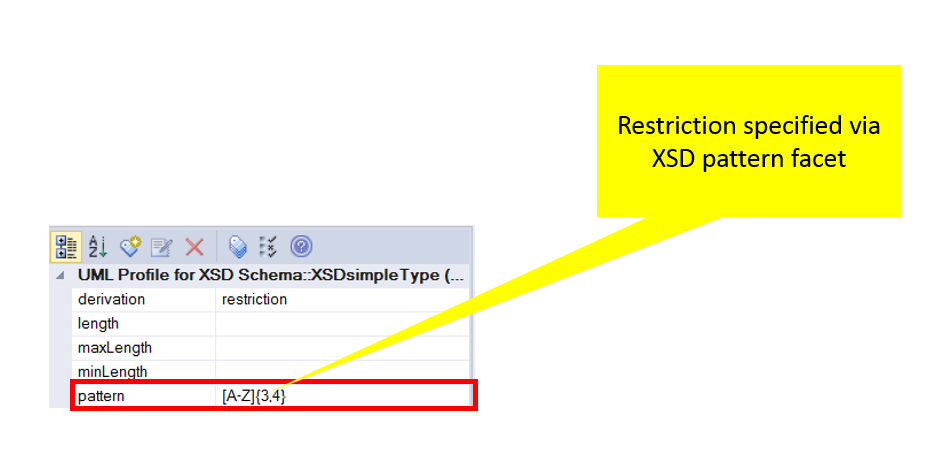


Figure . Example of Simple Type Restriction

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

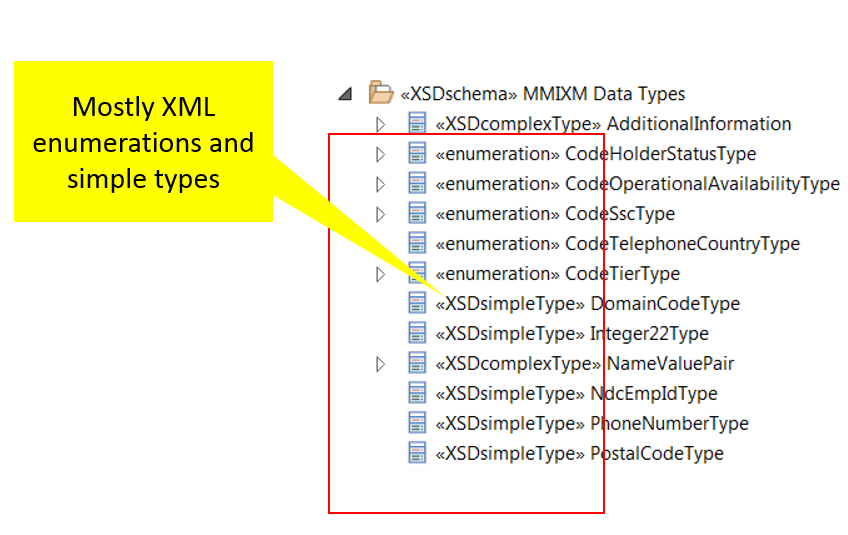


Figure . MMIXM Data Types

## 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), and standards maintained by the Machinery Information Management Open System Alliance (MIMOSA). 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 in Figure 9.

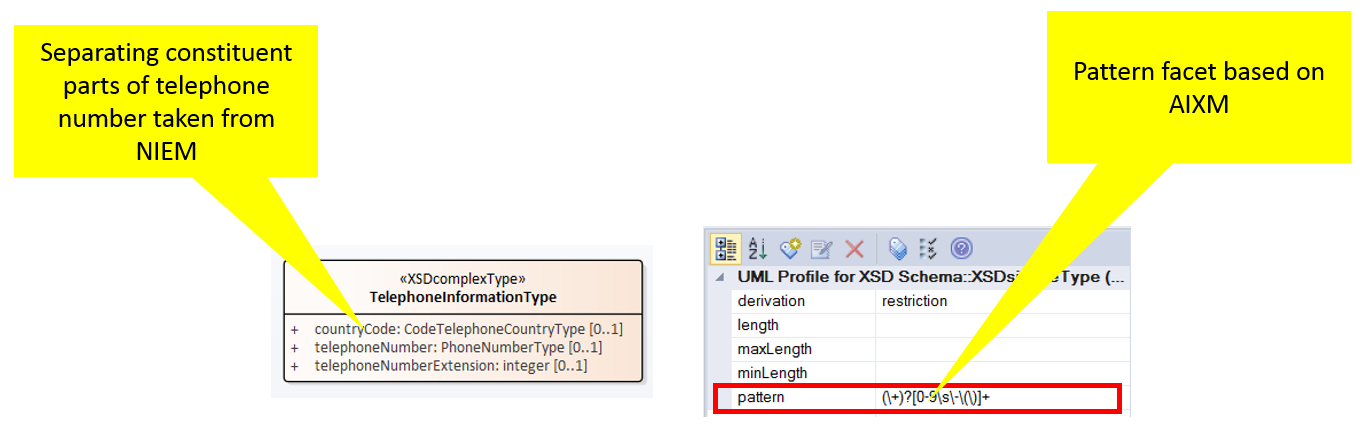


Figure . Examples of Reuse of Existing Standards

## Optionality for Most Class Associations and Attributes

Most of the class associations and attributes in the MMIXM v1.0 Logical Model are optional. This aligns with the FIXM modeling approach[[4]](#footnote-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.

## 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.

## Schema Extensions

There are currently no schema extensions associated with MMIXM v1.1.1. 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.

## Message Wrapper

The highest level element is the message class, which allows data producers to publish metadata or header information. 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.

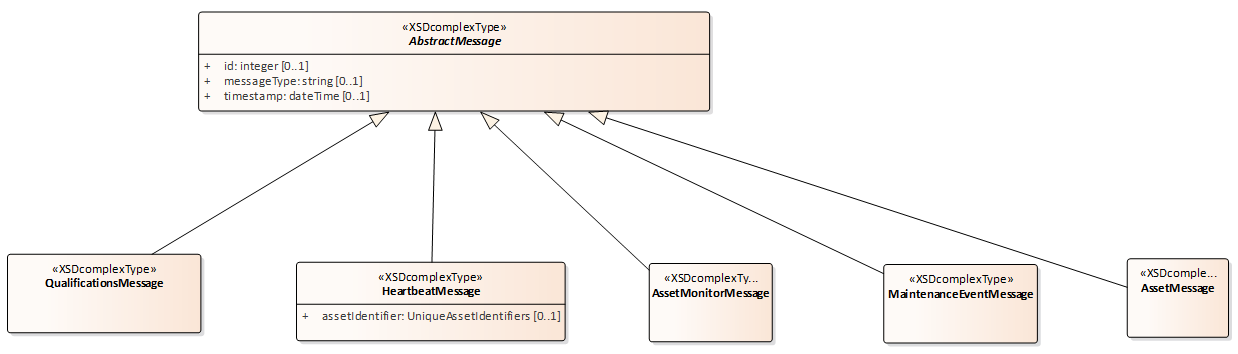


Figure . MMIXM Message Wrapper Examples

## 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.

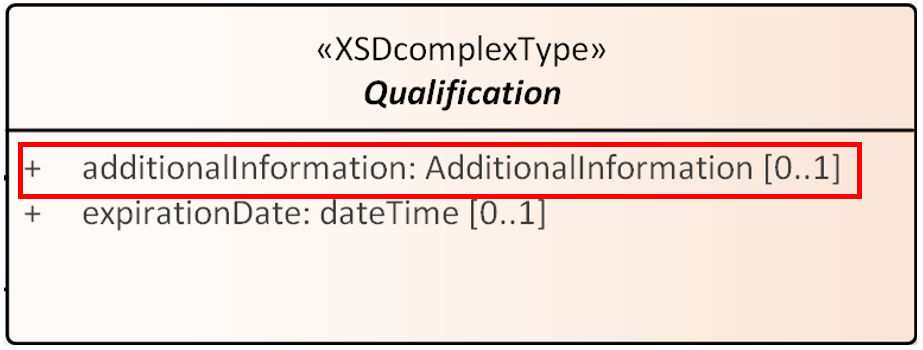


Figure . MMIXM Name-Value Pair Example

## Abstract Classes

Abstract classes are used in the model to reflect concepts that can be described in multiple ways. For example, the concept of a *Location* can be described as geographical coordinates, a postal address, a relative location inside a building structure, or a domain-specific address type such as a GSA address. It can even be described as a virtual address, e.g., an IP address. Figure 12 shows the abstract Location class, depicted with a red border, along with classes that derive from Location. Abstract classes will not show up in XML instances; however, they form the basis for which derived classes are constructed.

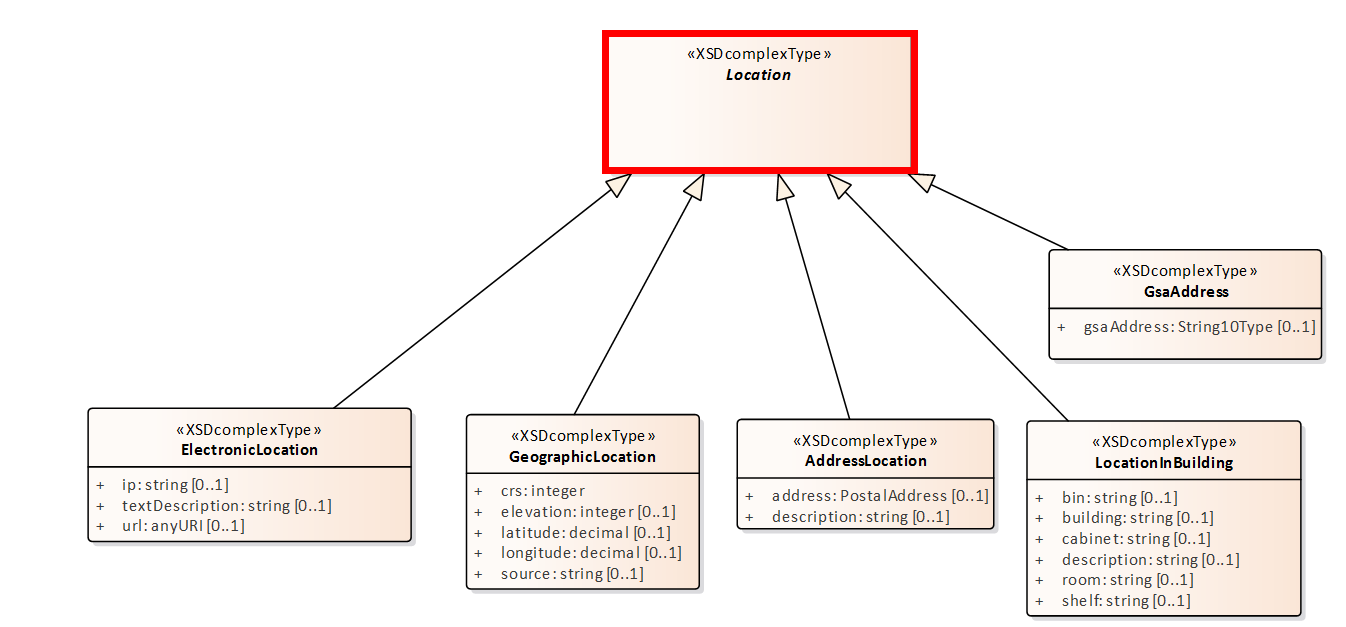


Figure . MMIXM Abstract Class Example

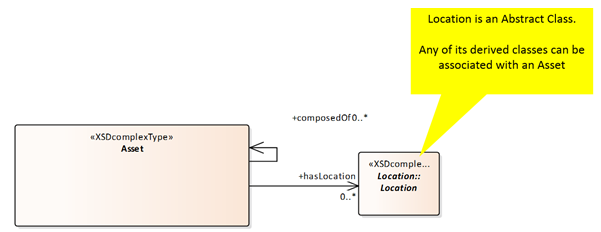
Defining associations with Abstract classes provides flexibility to stakeholders by allowing various ways to describe that association. Figure 13 provides an illustrative example. In the figure, the Asset class has an association with the Abstract class *Location*. As mentioned, Location cannot be in an XML instance because it is an Abstract class. However, any of its derived types can be in an XML message. Thus, an Asset could be associated with a Geographic Location and an Address Location in the same XML instance. 

Figure . MMIXM Abstract Class Association Example

## 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.

# MMIXM Logical Model (UML) Design Notes

Schema documentation pertaining to all MMIXM classes is provided at:

<https://www.mmixm.aero/releases/mmixm-1-1-1/schema_documentation/mmixm.html>

## Asset Design Notes

There are various organizations within the FAA that manage assets, and each manages them differently. For example, two organizations may use the same terminology for different types of assets -- and vice versa. In addition, organizations can have different ways of uniquely identifying an instance of an asset, but these asset identifiers are not globally unique. Building a customized submodel for each FAA asset management system runs contrary to the purpose of a data standard, thus in MMIXM the asset is broadly defined to include inventoried parts, operationally deployed systems, and even enterprise web services.

## Monitoring Design Notes

Monitoring represents real-time (or near real-time) monitoring of an asset’s current operational state. Since a monitoring message should contain a very limited amount of information, e.g., the current operational availability of the asset and the asset identifier, the monitoring message includes a class that is derived from the Asset class, but restricted to only provide this limited subset of information.

## Event Coordination Design notes

The event coordination package within MMIXM v1.1.1 largely reflects the logging structure of the RMLS-NLN database. A maintenance event contains logs describing the coordination of maintenance activities on an asset in need of maintenance or repair. There are associated logs depending upon whether the maintenance event represents a corrective action (repair), preventive maintenance action, coordination activities, or an alarm. The MMIXM modeling team engaged with RMLS stakeholders to identify which fields to incorporate in the model.

## Qualification Design Notes

The qualification package focuses on data elements describing the credentials and certifications belonging to maintenance personnel. The credentials are managed by AOV and the certifications are managed by CTS. There is a plan to integrate these two systems with RMLS. The MMIXM modeling team worked with the stakeholders of these systems to incorporate in the model data exchange requirements for this eventual integrated system.

## Reference Material Design Notes

The reference package contains data elements describing reference documentation associated with an asset. The package represents a very generalized structure; however, based on data exchange requirements provided to the MMIXM modeling team this may be further refined in future MMIXM releases.

# References

[1] Maintenance Management Information Exchange Model (MMIXM) Model Description Document (v1.0.0) – 1 September 2017

[2] Maintenance Management Information Exchange Model (MMIXM) Primer (v1.1.1) – 14 January 2019

[3] <https://www.mmixm.aero/>

[4] <https://semver.org/>

[5] http://www.aixm.aero/

[6] https://www.fixm.aero/

[7] http://www.wxxm.aero/

[8] <https://www.niem.gov/>

[9] http://www.mimosa.org/

1. This is a module within the Comprehensive Management Resource Information System (CMRIS) [↑](#footnote-ref-2)
2. Data exchange requirements related to the integration of AOV, CTS and RMLS were also incorporated. Additional systems outside of Tech Ops will be included in future MMIXM iterations. [↑](#footnote-ref-3)
3. AIXM and FIXM have a similar number of namespaces [↑](#footnote-ref-4)
4. In FIXM, all associations and attributes are optional [↑](#footnote-ref-5)