ARCHITECTURE REFERENCE

ODT-MIM

MIP Information Model (MIM) — Standard Reference

Overview of Requirements

HEADQUARTERS
UNITED STATES ARMY EUROPE AND AFRICA
(USAREUR-AF)
Wiesbaden, Germany

20 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

MIP INFORMATION MODEL (MIM) — STANDARD REFERENCE

OVERVIEW OF REQUIREMENTS

This chapter provides an overview of the information exchange requirements that underpin the MIP Information Model and its predecessors, the JC3IEDM and the ATCCIS Generic Hub.

Requirements in ATCCIS Phase III

Modelling work started early in Phase III (in 1992) without a formal statement of information exchange requirements. The Data Subgroup was staffed by a combination of serving military officers and technical experts.

Summary of Information Requirements

Major Topic	Information Category
Forces (friendly and enemy)	Force composition, Force disposition, Force sustainment, Mobility and transportation, Weapons systems, C4I and other information systems
Environmental conditions—physical	Land, Sea, Air, Space
Environmental conditions—civil	Political, Cultural, Economic
Situational information	Mission, C3 conditions, Intelligence, Targeting, Deployment/movement/manoeuvre, Force security, Sustainment

Categories of Operational Information

Category	Definition
1. Friendly or Enemy Forces	
1.1 Force Composition	Types and numbers of military and non-military forces
1.2 Force Disposition	Locations of military forces
1.3 Force Sustainment	Capabilities for logistical support (supply, maintenance, medical, etc.)

Category	Definition
1.4 Mobility and Transportation	Capability for inter- and intra-theatre movement of forces and materiel
1.5 Weapon Systems	Type, number, capabilities, and limitations of weapon systems
1.6 C4I and Other Information Systems	Type, number, capabilities, and limitations of C4I and other information processing systems
2. Environmental Conditions	
2.1 Physical	Factors arising from nature and the physical environment as modified by man (land, sea, air, space)

General Design Considerations

- **Object Identification**: Objects of military significance need to be identified, including physical things (units, equipment, stores, personnel, facilities, geographic features) and non-physical concepts (coordination points, lines, areas)
- **Class vs Instance**: Individual objects must be distinguished from the classes of objects to which they belong
- **Descriptive Characteristics**: Objects and their types need to be described with characteristics sufficient for supporting command and control tasks
- **Display Support**: Information elements must permit suitable display of the operational situation
- **Historical Records**: Selected information about certain characteristics needs to be retained for tracking and planning purposes

NAMESPACES

Introduction

Many organisations and standardisation bodies use the same names for different things. For instance, 'Unit' defined in the MIP Information Model may not be syntactically and semantically equivalent to 'Unit' in other standards such as MAJIIC or APP-9.

Namespaces in UML

In UML, each package defines a namespace so that elements defined in different packages do not conflict, even if they have the same name. The qualified name of an element is made of the qualified name of the containing package, followed by `::` and the element name itself.

Example: The class `Line` located in the package `Location`, which is contained in package `Classifiers`, has the qualified name `Classifiers::Location::Line`.

MIM Namespace Pattern

```
<org>::mim::<ver>
```

Where: - `<org>` denotes the Internet domain: `www.mimworld.org` - `<ver>` denotes the version number (e.g., `4.0.1`)

Example: The namespace of MIM 4.0.1 is `www.mimworld.org::mim::4.0.1`

Namespaces in XML

When using XML, a namespace is identified by a Unique Resource Identifier (URI). Rules:

- URIs are specified as resolvable URLs following IETF RFC 1738
- Generic URL format for namespaces: `https://<org>/mim/<ver>(/<pkg>)*(/<elem>)*`
- Generic URL format for concepts within namespaces: `https://<org>/mim/<ver>(/<pkg>)*(/<elem>)*#<elem>`

Examples: - Package namespace: `https://www.mimworld.org/mim/4.0.1/Classifiers/Object/Facility` - Class URI: `https://www.mimworld.org/mim/4.0.1/Classifiers/Object/Facility#Bridge` - Attribute URI: `https://www.mimworld.org/mim/4.0.1/Classifiers/Object/Facility/Bridge#spanQuantity`

SEMANTIC ANNOTATIONS

Object Type, Instance and Status Information

There are three types of information for Objects:

1. **Type information** — characteristics that apply to the class of object
2. **Instance information** — characteristics unique to a specific instance
3. **Status information** — administrative, medical, physical, and procedural states at a given time

The distinction between type and instance information is paramount. By default, attributes characterise a specific instance. Stereotypes indicate: - `«type»` — attribute carries information related to the type of Object - `«status»` — attribute specifies status information

Type Information

Many objects are of interest primarily in terms of a class or category rather than as individually identified items (e.g., a tank, a helicopter, an armoured brigade). Type information tends to be more static, as it must hold for all objects of a particular class.

Example: For a *Leopard-2* tank, calibre of main gun, track width, and load class are type characteristics as they apply to all instances.

Status Information

Status attributes capture administrative, medical, physical, and procedural states or conditions of Objects at a given time (past, present, or predicted). This includes:

- **Operational status**: Capability to perform roles (e.g., damage level, mobility, firing capacity)
- **Hostility status**: Classification as friend or enemy

Status information is not inherent to a specific Object and may have multiple reports from independent observers.

ROLES

During their lifetime, objects can play many different roles. An organisation may serve as a resource for performing a task or as a reporter of an incident. A facility may be considered an obstacle or a candidate target.

Role Characteristics

- A role may have its own properties different from the class playing the role
- Role instances may have access to some (but not all) properties of the object playing the role
- The existence of a role instance depends on the object playing the role
- An object may play several different roles simultaneously
- An object may acquire and relinquish roles dynamically
- Roles can play roles
- Each role instance has a unique identifier
- Several unrelated classes may play the same role

Roles as Classes

For complex needs, a role is represented by a separate class derived from class `Role`, connected via an association with stereotype `«isRoleOf»`. Role classes may have their own attributes, associations, and subclasses.

Adopted Attributes

When attributes of a natural class are needed for a specific role, they are adopted. Adopted attributes are declared as *derived* (indicated by a leading `/`). The mapping of adopted attributes is specified via OCL constraints.

Multiple Natural Classes

A role class may be associated with multiple natural classes. The `«isRoleOf»` associations are mutually exclusive (xor) — any role instance can only be associated with exactly one natural class instance.

DISCRIMINATOR CODES FOR COMPACT TAXONOMIES

The MIM defines thousands of Objects and Actions, resulting in deep and broad class hierarchies. Most leaf classes have no attributes of their own.

Attribute-free leaf classes can be collapsed into a `«discriminator»` code one level up the hierarchy, creating: - An enumeration with values for each collapsed subclass - A new `categoryCode` attribute marked as `«discriminator»`

This approach supports maintenance, comprehension, and visualization, and is lossless (can be transformed to/from regular UML hierarchies).

CODED ASSOCIATIONS

Some associations have an association class with a `«discriminator»` attribute, subsuming multiple concepts.

Example: A Person can have several family relationships with another Person. Rather than many individual associations, all are combined into a single association where specific types (`is cousin of`, `is grandparent of`, etc.) are encoded as code values.

The `«codedAssoc»` stereotype on code values specifies source and target role names for "unrolling" these associations.

CODE TYPES

Extensible and Complete Codes

The MIM includes hundreds of codes specified as UML enumerations. All codes must be specified as either:

- **Complete** (`complete=true`): User can only select from the given list
- **Extensible** (`complete=false`): User can indicate no appropriate value was found and provide a free-text description via `LackingCodeValue`

Managed Codes

Some codes cannot be reasonably defined at modelling time (e.g., fake nations for exercises). These are declared as *managed* (`managed=true`).

Managed codes: - Are defined and maintained by a military authority before/during operations - Literals in the MIM are suggestions only - Require technical solutions for sharing value sets among parties

Ordered Codes

For enumerations where code values have an ordering relationship:

- Stereotype `«orderedEnu»` is assigned
- Tagged value `relationship` specifies the ordering (a total strict order)
- Values are listed from lowest to highest

Example: `PrimacyCode` with values `tertiary < secondary < primary`

DESIGN PRINCIPLES

Missing Information / 'Nillable' Properties

During operations, specific information may not be exchangeable for various reasons:

- No appropriate code value exists
- The information doesn't make sense in context
- The correct value isn't available to the sender
- The value will be available later
- The value is unknown but probably exists
- The value is not divulged

The MIM defines `NilReason` (and subclass `LackingCodeValue`) to indicate why information is not provided.

Key principle: Any property is considered 'nillable' — there's no need for special literals like *Other*, *Unknown*, or *Not Otherwise Specified* in enumerations.

Semantic ID

Each model element is uniquely identified by a qualified name mapping to a URI. Each MIM version defines its own namespace.

Purpose: Capture semantic equivalence across MIM versions, as the model may be restructured — elements may be renamed, moved between packages, or literals may become classes.

Semantic IDs: - Are type 4 UUIDs in 8-4-4-4-12 format - Trace concepts between model versions - Support mappings to other standards (e.g., APP-6)

> **NOTE**
>
> Semantic IDs differ from Sparx Enterprise Architect GUIDs — they track semantic equivalence, not element identity.

Deprecated Elements and Associations

Elements or associations may become obsolete or ambiguous. They are marked with: - `«deprecatedClass»` - `«deprecatedAttribute»` - `«deprecatedAssociation»`

Deprecation Process: 1. Justification required for deprecation 2. Impact analysis on existing MIP capability packages 3. Formal CCB approval 4. Published list distributed to stakeholders 5. Removal after four working groups (~12 months) unless need is justified 6. Formal change proposal for removal

REFERENCES

- IETF RFC 1738 (URL specification)

- IETF RFC 4122 (UUID URN Namespace)
- STANAG 5525 (JC3IEDM)
- ADatP-5644 (Web Service Messaging Profile)