Testbed-12 FIXM GML Engineering Report

Table of Contents

Publication Date: 2017-06-19

Approval Date: 2016-09-15

Posted Date: 2016-11-13

Reference number of this document: OGC 16-028r1

Reference URL for this document: http://www.opengis.net/doc/PER/t12-F011

Category: Public Engineering Report

Editor: Thomas Disney

Title: Testbed-12 Aviation FIXM GML ER

OGC Engineering Report

COPYRIGHT

WARNING

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER’S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by

destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Abstract

The FAA and EUROCONTROL, in conjunction with multiple other international partners, are currently in the process of developing the Flight Information Exchange Model (FIXM). FIXM is an exchange model capturing Flight and Flow information that is globally standardized. The need for FIXM was identified by the International Civil Aviation Organization (ICAO) Air Traffic Management Requirements and Performance Panel (ATMRPP) in order to support the exchange of flight information as prescribed in Flight and Flow Information for a Collaborative Environment (FF-ICE).

FIXM is the equivalent, for the Flight domain, of Aeronautical Information Exchange Model (AIXM) and Weather Information Exchange Model (WXXM), both of which were developed in order to achieve global interoperability for, respectively, Aeronautical Information Systems (AIS) and Meteorological Information (MET) exchange. FIXM is therefore part of a family of technology independent, harmonized and interoperable information exchange models designed to cover the information needs of Air Traffic Management. Previous OGC IP initiatives developed an architecture that supports the exchange of AIXM and WXXM data. This report shall describe the integration of Geography Markup Language (GML) profile elements into FIXM, specifically, the Feature, Time, Geometries and Units of Measure (UOM), into FIXM version 3.0.1 and drafts of FIXM 4.0. The purpose of this report is to provide recommendations and change requests (CR) on the implementation of GML elements for use by the FIXM development community.

Business Value

FIXM is a developing standard that requires input from the aviation community. The sponsors of the Aviation Thread of the OGC Testbed have required an engineering report to describe the integration of GML into FIXM for the use case of implementing a FIXM sourced Web Feature Service (WFS). The business value is that with the adoption of OGC web service standards, producers and consumers of FIXM can reuse WFS COTS implementations for the distribution of FIXM sourced data.

What does this ER mean for the Working Group and OGC in general

This ER supports the overall objective of the OGC Aviation DWG to deliver recommendations on how to best use OGC standards in the context of SWIM / ATM information exchanges. The content of this ER is therefore expected to be used as appropriate by standardization organizations and/or governance bodies

in charge of the development of Air Traffic Management (ATM) standards for System Wide Information Management (SWIM).

Keywords

ogcdocs, testbed-12, FIXM, GML

Chapter 1. Introduction

1.1. Scope

The purpose of this report is to provide recommendations for the implementation of GML elements into FIXM. The report shall:

- Describe the integration of GML into the FIXM physical model (XML schemas);
- Document the verification and validation (through tests and demonstrations) of:
  1. the FIXM design, and
  2. the capability of the OGC Testbed-12 Aviation Architecture [OGC 16-018] to support interoperable exchange of FIXM data, and document the results (especially any improvements for FIXM); and
- Document potential enhancements and identified issues for relevant documents and references. For OGC documents, change requests (especially for Standard and Best Practice documents) will be created.

1.2. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors.

Table 1. Contacts

Name	Organization
Thomas Disney	Harris Corporation
Thomas Forbes	Snowflake Software
Volker Diels-Grabsch	m-click.aero
Timo Thomas	m-click.aero
Charles Chen	Skymantics

1.3. Future Work

It is expected that this document may result in changes in the FIXM standard, version 4.0 or later. Several change requests for the current FIXM 4.0 draft will be raised at the FIXM working group to be further discussed and refined.

The GML standard may wish to develop a UOM system more suitable for FIXM, so the decision not to use GML's UOM types for FIXM as described in section GML elements not *yet* to introduce into FIXM could be reconsidered. The goal would be to develop a common set of UOM types throughout AIXM, FIXM, WXXM and possibly other GML application schemas.

GML may also play a strong role when defining how to include the actual flight trajectory into FIXM. The actual route is essentially a list of points with additional attributes such as altitude and

timestamps. This is a well-covered topic of GML and similar structures already exist in other GML application schemas. For the comparatively dense point set of an actual flight trajectory, a compact encoding is especially important.

Although not directly related to FIXM, it should be investigated in how far the current concept of the required and document-wide unique gml:id attribute still serves its original purpose. For example, making this attribute optional would not only solve certain conflicts between the gml:id concept and the WFS standard, but may improve the overall acceptance of GML.

The special requirements of the FIXM XML Schema demonstrate that the tools for generating XML Schema from UML model should be joined and standardized. FIXM is not the only model that uses specialized mapping rules. For example, AIXM has such rules as well, which, due to AIXM temporality, avoids modeling a separate TimeSlice class for every Feature class in the UML model. It may be desirable to capture these mapping rules of AIXM, FIXM, and others in the future in a unified form. This unified form may be a directly usable XSLT script, a configuration for the ShapeChange tool, or a specialized to-be-developed mapping language.

In the August 2016 release of FIXM 4.0, a concept of FIXM messaging was included. Due to the timing of the FIXM 4.0 release relative to Testbed 12, Testbed 12 investigations could not extend to the integration of GML within the FIXM messaging concept. Therefore, it is recommended future work investigate the integration of GML within the concept of FIXM messaging in FIXM 4.0.

1.4. Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

Chapter 2. References

- OGC: OGC 16-018 - Testbed-12 Aviation Architecture Engineering Report, 2017
- ISO: ISO 19109:2015 – Geographic information – Rules for Application Schema, 2015
- OGC: OGC 14-037 – OGC Testbed 10 Flight Information Exchange Model GML Schema, 2014
- FAA: SWIM-002 - "SOFTWARE SPECIFICATION Syntax and Processing of XML-Based Documents in the Context of SWIM-Enabled Services", http://www.faa.gov/nextgen/programs/swim/governance/standards/media/Syntax%20and%20Processing%20of%20XML-Based%20Documents%2006162015.pdf, 2015
- Braeckel, A.: Efficient XML and Weather, Briefing to WXXM/AIXM Conference, http://www.aixm.aero/sites/aixm.aero/files/imce/library/AIXM_WXXM_Conf_2010/25_day_2_-*wx*-_05_efficientxml-and-weather-case-study-2010-05-04.pdf, 2010
- OGC: [OGC 12-028r1] Use of GML for aviation data", https://portal.opengeospatial.org/files/?artifact_id=62061, 2016
- OGC: [OGC 14-037] OGC Testbed 10 FIXM GML Schema, https://portal.opengeospatial.org/files/?artifact_id=58956, 2014
- Interactive Instruments: ShapeChange, http://shapechange.net/
- Eurocontrol and FAA: AIXM 5 - Feature Identification and Reference - use of xlink:href and UUID, version 1.0, http://www.aixm.aero/sites/aixm.aero/files/imce/AIXM51/aixm_feature_identification_and_reference-1.0.pdf, 2011
- Airservices Australia, DSNA, EUROCONTROL, IATA, JCAB, NATS Limited, NAV CANADA, SESAR Joint Undertaking & US FAA: FIXM Change Management Charter, https://fixm.aero/documents/FIXM%20Change%20Management%20Charter%20v1.0.pdf, 2014

Chapter 3. Terms and Definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard [OGC 06-121r9] shall apply. In addition, the following terms and definitions apply.

3.1. Abbreviated Terms

AIXM - Aeronautical Information Exchange Model
AMDB - Aerodrome Mapping Database
CCB - Change Control Board
COTS - Commercial Off the Shelf
DNOTAM - Digital NOTAM
FAA - Federal Aviation Administration
FIXM - Flight Information Exchange Model
GML - Geography Markup Language
HTTP - HyperText Transfer Protocol
ICAO - International Civil Aviation Organization
ISO - International Organization for Standardization
NAS - National Airspace System
NEMS - NAS Enterprise Messaging Service
NOTAM - Notice to Airmen
OGC - Open Geospatial Consortium
RDF - Resource Description Framework
SBVR - Semantics of Business Vocabulary and Business Rules
SQL - Structured Query Language
SWIM - System Wide Information Management
UUID - Universally Unique Identifier
WCS - Web Coverage Service
WFS - Web Feature Service
WFS-T - WFS-Transactional
WMTS - Web Map Tile Service
WPS - Web Processing Service
WXXM - Weather Information Exchange Model
XMI - XML Metadata Interchange
XML - Extensible Markup Language
XSLT - Extensible Stylesheet Language Transformations

Chapter 4. Overview

The Flight Information Exchange Model (FIXM) is an exchange model capturing Flight and Flow information that is globally standardized. The need for FIXM was identified by the International Civil Aviation Organization (ICAO) Air Traffic Management Requirements and Performance Panel (ATMRPP) in order to support the exchange of flight information as prescribed in Flight and Flow Information for a Collaborative Environment (FF-ICE). FIXM is the equivalent, for the Flight domain, of AIXM (Aeronautical Information Exchange Model)and WXXM (Weather Information Exchange Model), both of which were developed in order to achieve global interoperability for, respectively, AIS and MET information exchange. FIXM is therefore part of a family of technology independent, harmonized and interoperable information exchange models designed to cover the information needs of Air Traffic Management.

The OGC Testbed 12 FIXM Engineering Report seeks to harmonize FIXM with AIXM, and to permit the exchange of FIXM via a WFS. Testbed 12 seeks to build upon previous OGC IP initiatives and to demonstrate the integration of the ISO/OGC standard GML in a manner that meets the requirements of the FIXM community. Work undertaken in previous OGC IP initiatives seeks to integrate GML to FIXM via a model-driven approach.

Chapter 5. Status Quo & New Requirements Statement

5.1. Status Quo

The FAA, Eurocontrol, and multiple international stakeholders have developed the current FIXM. FIXM is an exchange model capturing Flight and Flow information that is globally standardized.

To harmonize FIXM with AIXM and WXXM, and to permit the exchange of FIXM via a WFS, Testbed 12 sought to build upon previous OGC IP initiatives and to demonstrate the integration of the ISO/OGC standard GML in a manner that meets the requirements of the FIXM community.

Work undertaken in previous OGC IP initiatives sought to integrate GML to FIXM via a model-driven approach. The FIXM UML was modified to ensure ISO 19109 compliance, before a UML to GML Application Schema (UGAS) tool model drove a GML-conformant FIXM 2.0 version. Feedback of this GML compliant FIXM has since been received from the FIXM community; most pertinently feedback highlighted the following:

- Data bloating due to use of GML object-property model (see 7.1 Object-Property Model); and
- Data bloating due to removal of XML attributes (see 7.3.1 Temporal Values).

5.2. Requirements Statement

Testbed 12 built upon FIXM community feedback to ensure that the implementation of GML elements is usable by the FIXM development community. The GML integration activity requirements were:

- Describe the integration of GML into the FIXM physical model (XML schemas); and
- Document the verification and validation (through tests and demonstrations) of:
  1. the FIXM design; and
  2. the capability of the OGC Aviation Architecture to support interoperable exchange of FIXM data, and document the results (especially any improvements for FIXM).

Chapter 6. Solutions

6.1. Targeted Solutions

Building upon previous work to increase interoperability with FIXM, various targeted solutions were investigated and discussed during Testbed 12. At a high-level, these targeted solutions were:

- Chapter 7 - Implementing GML in FIXM; and
- Chapter 8 - Reducing FIXM XML File Size.

These approaches are not mutually exclusive; each approach is explained in detail in the sections below.

6.2. Recommendations

The following recommendations describe which parts of FIXM should be changed, but also which parts of FIXM should not, or at least not yet, be changed.

6.2.1. GML/GCO elements not to introduce into FIXM

Using the following elements from GML as well as ISO/TS 19139:2007 GCO would not provide any benefits, neither for FIXM itself nor for interoperability. Also, these would impose additional complexity on the FIXM model and increase the size of FIXM documents. So it is recommended not to introduce these into FIXM:

- `gco:CharacterString` (keep using `xs:string` instead)
- `gco:Decimal` (keep using `xs:decimal` instead)
- `gco:Integer` (keep using `xs:int` instead)
- `gml:TimePeriodType` (keep using `fb:DurationType` = `xs:duration` instead)
- `gml:TimePositionType` (keep using `fb:TimeType` = `xs:dateTime` instead)

6.2.2. GML elements not *yet* to introduce into FIXM

The following element from GML is not recommended for FIXM, because the GML system for units of measurements is not as fine grained as the UOM system of FIXM:

- `gml:UomIdentifier` (keep using `fb:UomAirspeedType`, `fb:UomAltitudeType`, etc. instead)

However, if a future version of the GML standard would develop a UOM system more suitable for FIXM, this recommendation should be reconsidered. Using a common set of UOM types throughout all GML application schemas (AIXM, FIXM, WXXM, ...) would provide an interoperability benefit on its own, especially with regard to generic GIS clients or generic GIS processing tools.

6.2.3. FIXM Data Usability

During integration of the FIXM GML compliant WFS with the data broker, a data usability issue was

raised. The issue stemmed from the concepts described in Section 7.7 and Section 7.8, and highlights a use case for implementing the related change requests.

The FIXM standard permits a point to be represented using one of multiple encodings: (1) by bearing, distance and reference point (`fb:IntersectionType`), (2) by latitude-longitude coordinate pair (`fb:PositionPointType`), or (3) by reference point (`fb:DesignatedPointOrNavaidType`).

The original FIXM data sourced for the Testbed 12 scenario represented different points within a single flight route using either approach 2 or 3. The use of referenced points led to issues when visualizing and querying the data. For example, a user requesting the aforementioned FIXM data would be unable to visualize or spatially query the data without resolving the points to latitude-longitude coordinate pairs. Compounding the issue is the fact that fixes representing points are not globally unique. Therefore, in order to resolve reference points to latitude-longitude coordinate pairs, resolution must be done in context of the preceding and/or following points if using a global lookup table of reference points.

Change requests contained within Section 7.7 and Section 7.8 discuss encoding points using a combination of the aforementioned representation approaches. For example, a point may be represented using both the reference point and also a latitude-longitude coordinate pair. Using this approach a user is able to spatially locate the data whilst maintaining the point reference without incurring the complexities of referenced point resolution.

6.2.4. Change Requests for FIXM

The following changes are recommended for FIXM. These are to be submitted as change requests (CR) for approval by the FIXM Change Control Board (CCB). The change requests are numbered with a "TB12" prefix (TB12.1, TB12.2, ...) to facilitate referencing these change requests during the process through the FIXM CCB.

- FIXM CR TB12.1: Minimize set of Feature Types
- FIXM CR TB12.2: Switch to GML Feature Types
- FIXM CR TB12.3: Relocate top-level Attributes of Feature Types
- FIXM CR TB12.4: Define Business Rule for `gml:id` in `<fx:Flight>`
- FIXM CR TB12.5: Introduce `gml:Point`
- FIXM CR TB12.6: Unrestricted URI Referencing
- FIXM CR TB12.7: Unrestricted Natural Key Refs for Aerodrome
- FIXM CR TB12.8: Unrestricted Natural Key Refs for Points
- FIXM CR TB12.9: Make use of the GML Aviation Profile
- FIXM CR TB12.10: Use `<gml:identifier>` instead of `<fx:gufi>`
- FIXM CR TB12.11: Integration of ISO 19115 / 19139 Metadata

Chapter 7. Implementing GML in FIXM

The implementation of GML in FIXM provides benefits at both the data and service levels. At the data level, GML integration allows for greater harmonization between FIXM and related exchange models that are based on GML, such as AIXM and WXXM/IWXXM [1: The International Civil Aviation Organization (ICAO) standardized a version of WXXM called ICAO-WXXM, or IWXXM for short]. Whilst at the service level, GML integration means FIXM can leverage globally accepted data service standards such as the WFS.

To implement GML in FIXM whilst maintaining a compact data encoding, the following areas have been investigated:

- Object-Property Model (including the UML to XSD process);
- GML Abstract Feature;
- Use of `gml:id` Attribute;
- Use of `<gml:identifier>` Element;
- Temporal Values;
- Use of `xsi:type` Attribute;
- Harmonize FIXM references with AIXM and WXXM using XLink; and
- Remove restrictions on providing natural keys and coordinates.

7.1. Object-Property Model

Feedback from the OGC Testbed 10 efforts to integrate GML into FIXM highlighted "data bloating" caused primarily by GML object-property model implementation. [2: GML Object-Property Model Description - The Resource Description Framework (RDF) class-property model partially forms the basis of the GML object-property model. The GML object-property model works by declaring a type and then assigning properties to that type. The XML properties are called child elements of the type. The GML object-property model uses a convention whereby GML types are denoted with UpperCamelCase and properties denoted with LowerCamelCase. A GML object cannot directly contain another GML object, but must have a property whose value us the other object.]

The use of the GML object-property model adds additional XML nesting when implemented - therefore, conflicting the desire to maintain compact encoding.

When integrating GML within FIXM during Testbed 12, the following GML object-property model implementations were strategized and discussed.

1. Stay strictly within the GML object-property model, and apply ISO 19109 encoding rules for generating XML schemas from UML.
2. Modify the GML object-property model and encoding rules slightly. Apply minimal modifications to provide a compact encoding while keeping the object-property model.
3. Discard the object-property model entirely for FIXM. Modify the existing FIXM schema on as few places as possible, just enough to work within existing WFS implementations.

7.1.1. Keep Object-Property Model

By default, the object-property model is instantiated by UML to GML Application Schema (UGAS) tools such as Interactive Instruments ShapeChange [3: Interactive Instruments ShapeChange - http://shapechange.net/] or inbuilt Enterprise Architect GML Application Schema output functionality.

These UGAS tools are typically built upon standard encoding rules (see GML 3.2.1 Annex E) that specify the mapping between an ISO 19109 conformant UML model to a GML application schema. In general, UML to GML application schema encoding rules are based on the concept that UML class definitions are mapped to type and element declarations in XML schema.

Interactive Instruments ShapeChange UGAS tool was used in OGC Testbed 10 to output the FIXM GML XSDs from an ISO 19109 conformant FIXM 2.0 UML model. Using this standard approach, the outputted GML application schema was subject to the GML object-property model which lead to additional XML nesting in the data. This XML nesting lead to data bloating, for example:

Example of FIXM 2.0 Encoding:

```
...
<route routeText="KATL.PNUTT7.MCN..AMG.J45.OMN..KDAB">
    <segment airway="S1">
        <routePoint>
            <point xsi:type="base:LocationPointType">
                <location srsName="urn:ogc:def:crs:EPSG::4326">
                    <pos>33.63333333 -84.41666667</pos>
                </location>
            </point>
        </routePoint>
    </segment>
    ...
</route>
...
```

Example of GML Conformant FIXM 2.0 Encoding:

```
...
<route>
    <Route routeText="KATL.PNUTT7.MCN..AMG.J45.OMN..KDAB">
        <segment>
            <RouteSegment airway="S1">
                <routePoint>
                    <RoutePoint>
                        <point>
                            <fb:LocationPoint>
                                <fb:location>
                                    <gml:Point gml:id=
"KATL.PNUTT7.MCN..AMG.J45.OMN..KDAB..S1" srsName="urn:ogc:def:crs:EPSG::4326">
                                        <gml:pos>33.63333333 -84.41666667</gml:pos>
                                    </gml:Point>
                                </fb:location>
                            </fb:LocationPoint>
                        </point>
                    </RoutePoint>
                </routePoint>
            </RouteSegment>
        </segment>
        ...
    </Route>
</route>
...
```

The above examples demonstrate the GML conformant FIXM version introduces object levels inspired by the GML object-property model (for example, `RouteSegment` and `RoutePoint`). This increased XML nesting is undesirable to FIXM stakeholders who want an efficient and compact XML encoding. Therefore, during Testbed 12 the decision was made to maintain a compact encoding and not adopt this approach of instantiating the GML object-property model.

7.1.2. Modify Object-Property Model

This can be achieved by adding another element `foo` to the meta structure which is similar to "object", but has a more compact XML encoding. Of course, `foo` then will not have certain properties which "objects" usually have:

- `foo` cannot be replaced with a reference.
- `foo` cannot be an abstract type but must be a concrete type (for example, cannot be "point" which may be the name of a designated point, or may contain directly the coordinates).

Using off-the-shelf UGAS tools, this approach is not possible due to the deviation from standard UGAS encoding rules. Custom development would be required to support such additional encoding rules and functionality. Due to the development required coupled with the limitations above, this approach was not adopted in Testbed 12.

7.1.3. Discard Object-Property Model

By discarding the use of the object-property model, data bloating concerns are negated. This can be achieved by providing minimal modification to the existing FIXM XML schemas to allow them to function with existing WFS implementations.

Following the results of GML integration into FIXM during OGC Testbed 10 [4: OGC Testbed 10 FIXM GML Schema - https://portal.opengeospatial.org/files/?artifact_id=58956], FIXM stakeholders investigated integration of GML without object-property model implementation. The Testbed 12 sponsor provided the XML schema produced from the aforementioned investigation. This provided a starting point for the Testbed 12 requirement - the XML schema was based on FIXM 3.0.1, with specific modifications to include GML elements only where needed.

Producing such an XML schema with discarded object-property model from UML requires a custom UGAS tool that does not fully implement UML to GML Application Schema encoding rules. Producing such a custom tool during Testbed 12 was beyond the remit, therefore, this was achieved by applying XML schema modification directly.

7.1.4. GML Abstract Feature

To process FIXM with common GML tools, such as serving FIXM through a WFS, all FIXM feature types must be marked as GML feature types. That means:

1. All GML feature types must extend directly or indirectly from `gml:AbstractFeatureType`; and
2. All GML feature elements must be in the substitution group `gml:AbstractFeature`.

Currently FIXM defines many feature types that are not intended to be used as stand-alone features - therefore, should not be converted to GML features. For each FIXM feature converted to GML features, two modifications would be encountered:

- The WFS standard states that the properties of a GML feature must be XML elements, not XML attributes (except `gml:id`); and
- The `gml:id` attribute is required on each GML feature.

These modifications contradict the desired compactness of FIXM. To prevent the above complications, there should be a few feature types in FIXM as possible. That is, only the central `fx:FlightType` should become a feature type. For `fx:FlightType` to be realised as a GML feature, the following changes have been applied:

- `fx:FlightType` extended directly or indirectly from `gml:AbstractFeatureType`; and
- `fx:Flight` must be in the substitution group `gml:AbstractFeature`.

Note that this adds a new required XML attribute, `gml:id`, on the `fx:Flight` element. It also requires the following attributes directly defined in `fx:FlightType` to be converted to XML elements:

- `flightType` (type `fx:TypeOfFlightType`); and
- `remarks` (type `fb:FreeTextType`).

In summary, the following FIXM document:

```
<fx:Flight
    ...
    flightType="S"
    remarks="Some Remarks">
  ...
</fx:Flight>
```

is changed to:

```
<fx:Flight
    ...>
  <fx:flightType>S</fx:flightType>
  <fx:remarks>Some Remarks</fx:remarks>
  ...
</fx:Flight>
```

While this increases the size of FIXM documents by a few bytes, this is a one-time increase per `<fx:Flight>` feature. Moreover, this affects only datasets where these properties are actually set.

Alternatively, it is also possible to introduce a new property element that carries the attributes `flightType` and `remarks`.

This leads to the following FIXM change requests:

- FIXM CR TB12.1: Minimize set of Feature Types
- FIXM CR TB12.2: Switch to GML Feature Types
- FIXM CR TB12.3: Relocate top-level Attributes of Feature Types
- FIXM CR TB12.5: Introduce `gml:Point`
- FIXM CR TB12.9: Make use of the GML Aviation Profile

7.2. Use of `gml:id` Attribute

Introducing GML adds a new required attribute `gml:id` to all features and geometries. Although this goes slightly contrary to the compactness goal of FIXM, the `gml:id` is currently unavoidable (as of GML 3.2) and important for a variety of WFS operations, filter expressions and so on. A business rule should be established on how to populate the required `gml:id` attributes in FIXM on both the feature and geometry levels.

At feature level, the following strategies have been investigated:

A. Fill `gml:id` with a short dummy identifier:

```
<fx:Flight gml:id="ID_1">
  ...
  <fx:gufi codeSpace="urn:uuid">fe14ebcb-ff7a-4d20-bcaa-4ca15fa1d2ab</fx:gufi>
  ...
</fx:Flight>
```

B. Use `gml:id` as a replacement for `<fx:gufi>`, as it serves almost the same purpose:

```
<fx:Flight gml:id="uuid.fe14ebcb-ff7a-4d20-bcaa-4ca15fa1d2ab">
  ...
</fx:Flight>
```

C. Derive `gml:id` from `<fx:gufi>`, which is redundant but works well with compression:

```
<fx:Flight gml:id="uuid.fe14ebcb-ff7a-4d20-bcaa-4ca15fa1d2ab">
  ...
  <fx:gufi codeSpace="urn:uuid">fe14ebcb-ff7a-4d20-bcaa-4ca15fa1d2ab</fx:gufi>
  ...
</fx:Flight>
```

At geometry level, the following strategies have been investigated:

1. Fill `gml:id` with a short dummy identifier:

```
<gml:Point ... gml:id="ID_3">
  ...
</gml:Point>
```

2. Derive `gml:id` from `<fx:gufi>` and a serial number (here: `3`), which is redundant but works well with compression:

```
<gml:Point ... gml:id="uuid.fe14ebcb-ff7a-4d20-bcaa-4ca15fa1d2ab.3">
  ...
</gml:Point>
```

These are 6 possible recommendations (`A1`, `A2`, `B1`, `B2`, `C1`, `C2`) in total.

In AIXM, almost the same questions arose, with just two differences. First, AIXM reuses an existing GML element (`<gml:identifier>`), while FIXM introduces a new element (`<fx:gufi>`). Second, AIXM introduces GML objects at all levels, so there are `gml:id` attributes not just for features and geometries, but also for all complex properties.

In AIXM, the current recommendation to use strategy `C` on feature level, while there is no recommendation on geometry level. Although there are popular AIXM systems which do not follow

this recommendation, such as NM ADR implementing A1, most often the strategy combination C1 is seen in AIXM datasets, sometimes also a variant of C2.

However, the condition above is about to change. In the upcoming WFS Temporality Extension, it is discussed whether not to store gml:id values at all, even though that would degrade certain WFS filters which operate directly on gml:id values. There are two reasons for considering storage of the values. First, the <gml:identifier> element already provides the UUIDs, and filtering by <gml:identifier> is only slightly more verbose than filtering directly by gml:id. Second, it is possible to formulate WFS queries that return different feature representations for the same feature or objects. For these queries, a conformant WFS server would have to return an XML dataset that is formally schema-invalid due to duplicate gml:id values. Note that this contradiction already exists in the current WFS 2.0 specification, but has not been thoroughly discussed until the work on the WFS Temporality Extension.

In light of these latest discussions, it is recommended that FIXM doesn't follow AIXM here, but goes one step ahead and implements A1. Future versions of GML and WFS may fix the issues with gml:id, perhaps by making it optional. Until then it is best to consider the required gml:id attributes as technical noise and not to use it for any identification purposes (for which <gml:identifier> or <fx:gufi> are sufficient).

This leads to the following FIXM change request:

- FIXM CR TB12.4: Define Business Rule for gml:id in <fx:Flight>

7.3. Use of <gml:identifier> Element

It was investigated whether FIXM should make use of <gml:identifier> instead of their custom <fx:gufi> field. On the one hand, right now there is no pressing issue to be solved. On the other hand, both have the same structure and serve the same purpose: identifying a single GML feature through a globally unique identifier. The <gml:identifier> is already directly available to FIXM through gml:AbstractFeatureType. Moreover, the <gml:identifier> may receive some additional semantics in future WFS versions and related standards (possibly related to generic UUID resolution or other UUID topics). Such improvements would then be inapplicable to FIXM due to its custom <fx:gufi> identifier field.

This leads to the following FIXM change request:

- FIXM CR TB12.10: Use <gml:identifier> instead of <fx:gufi>

7.4. Temporal Values

The concept of time in FIXM is extensive; occurrences of timestamps are typically represented by XML attributes using xs:dateTime. The GML concept of time was implemented in the aforementioned sponsor-provided FIXM GML schemas used as a starting point in Testbed 12; instead of xs:dateTime, gml:TimePositionType was used. The use of gml:TimePositionType:

- allows the use of alternative temporal reference systems (i.e. calendars and time keeping systems other than ISO 8601, such as GPS time);

- permits the expression of inexact or "fuzzy" positions using now, before, after qualifiers; and
- accommodates time positions of varying precision (e.g. year, month, date).

The functionality offered by `gml:TimePositionType` was deemed unnecessary for FIXM. Furthermore, `gml:TimePositionType` is a complex element, therefore timestamps could no longer be modelled as XML attributes, thus causing data bloating. For example:

7.4.1. `xs:dateTime` in FIXM example

```
...
<fx:runwayPositionAndTime runwayName="27R">
    <fb:runwayTime>
        <fb:actual time="2016-02-24T20:26:00.000Z"/>
        <fb:target time="2016-02-24T22:10:49.456Z"/>
    </fb:runwayTime>
</fx:runwayPositionAndTime>
...
```

7.4.2. `gml:TimePositionType` in FIXM example

```
...
<fx:runwayPositionAndTime runwayName="27R">
    <fb:runwayTime>
        <fb:actual>
        <fb:time>2016-02-24T20:26:00.000Z</fb:time>
    </fb:actual>
    <fb:target>
        <fb:time>2016-02-24T22:10:49.456Z</fb:time>
    </fb:target>
    </fb:runwayTime>
</fx:runwayPositionAndTime>
...
```

7.5. Use of xsi:type Attribute

Testbed 12 investigated the use of the `xsi:type` attribute in FIXM because typically in GML this construct is not used, but instead every XML element has a clear type. Moreover, this is actively forbidden in some GML application schemas such as AIXM [5: See "Use of GML for aviation data", page 63, section "12.6.5. Use of xsi:type is forbidden", https://portal.opengeospatial.org/files/?artifact_id=62061].

The `xsi:type` attribute is one of several attributes defined by the schema instance namespace [6: http://www.w3.org/2001/XMLSchema-instance] that are used in instance documents with special meaning. The `xsi:type` attribute may be used to explicitly assert an element's type in the instance document.

The xsi:type attribute is used extensively throughout FIXM, to the point that elements defining aerodromes, significant points and constraints cannot be expressed without it. The examples below demonstrate how the type of the fx:point element may be asserted using the xsi:type attribute, thus enabling a location to be encoded in multiple ways:

7.5.1. Example 1 - xsi:type="fb:FixPointType"

```
...
<fx:routePoint>
    <fx:point xsi:type="fb:FixPointType" fix="DETOX"/>
</fx:routePoint>
...
```

7.5.2. Example 2 - xsi:type="fb:LocationPointType"

```
...
<fx:routePoint>
    <fx:point xsi:type="fb:LocationPointType">
        <fb:location srsName="urn:ogc:def:crs:EPSG::4326">
            <ff:pos>35.0 -74.0</ff:pos>
        </fb:location>
    </fx:point>
</fx:routePoint>
...
```

7.5.3. Example 3 - xsi:type="fb:RelativePointType"

```
...
<fx:routePoint>
    <fx:point xsi:type="fb:RelativePointType" fix="DETOX">
        <fb:distance uom="NAUTICAL_MILES">100.0</fb:distance>
        <fb:radial ref="TRUE">90.0</fb:radial>
    </fx:point>
</fx:routePoint>
...
```

Because the xsi:type attribute explicitly asserts an element's type, some tools cannot match the name of the type instead of the name of the element – in this case knowing the element name and the context in which it occurs is not enough to process the XML. Conversely, a "type-aware" application can process XML containing xsi:type attributes – therefore, for an application processing FIXM being "type-aware" would be a requirement.

Schema modifications to remove the xsi:type attribute were discussed during Testbed 12, however lead to increased XML nesting. In summary, following investigation use of the xsi:type attribute within a GML conformant FIXM was deemed acceptable.

7.6. Referencing using XLink

While not directly linked to the integration of GML, Testbed 12 also investigated potential areas of harmonization between FIXM and related standards such as AIXM.

FIXM currently makes reference to aeronautical information modeled within the AIXM model using natural key information. For example, the departure aerodrome of a flight is referenced using the airport ICAO code (a four character alphanumeric code). This domain-specific way of linking, works well for highly specialized services and clients, but is inappropriate for generic services and clients.

In AIXM, being based on GML, only generic links are used to reference between features. [7: AIXM 5 - Feature Identification and Reference - use of xlink:href and UUID, version 1.0, http://www.aixm.aero/sites/aixm.aero/files/imce/AIXM51/aixm_feature_identification_and_reference-1.0.pdf] Among others, this is the key enabler for automatic link resolving in Web Feature Services (WFS) through the `resolveDepth="…"` parameter.

This is only possible if all links follow the URI standard and are encoded in XML using the XML Linking Language (XLink) as specified by the W3C, which defines standardized XML attributes for linking purposes. XLink is a generic standard used across a variety of schemas such as GML, GML application schemas linke AIXM, SVG, MathML, RDDL and many others. The most important XLink attribute is `xlink:href` which contains the target URI, but XLink also defines additional attributes such as `xlink:title` and `xlink:role`. All XLink attributes are optional, so adding XLink capabilities to an existing XML Schema is always a backwards compatible change.

XLink references and natural keys are not mutually exclusive. While specialized FIXM systems may continue to use natural keys, XLink references enable generic services to add globally understood links on behalf of their clients, so that clients are able to resolve these links in a generic way.

The following FIXM types would benefit from XLink capabilities:

`fb:AerodromeReferenceType`
: may wish to link to an `aixm:AirportHeliport` feature.

`fb:SignificantPointType`
: may wish to link to an `aixm:DesignatedPoint` or `aixm:Navaid` feature.

The XLink standard does not provide any restriction on the kind of URI to be linked to. Depending on the context, different kinds of links may be appropriate.

- Local references such as `xlink:href="#uuid.89863…"` are useful if the FIXM flight data and AIXM basic data are provided within the same document or set of documents.
- Abstract references such as `xlink:href="urn:uuid:89863…"` are useful if just the identifier (UUID) of the corresponding AIXM feature is to be communicated and the receiver is expected to be capable of URN resolving.
- External absolute references such as `xlink:href="https://example.com/get/89863…"` are useful if the AIXM features are served through a different service and the FIXM service wants to direct the receiver to a specific authoritative basic data service such as an AIXM WFS. The link then describes an HTTP GET request to that other service to receive the referenced AIXM feature.

- External relative references such as `xlink:href="./89863⋯"` are useful if the FIXM flight data and AIXM basic data are served through the same service, but do not appear in the same response document. The link then describes an HTTP GET request to the same service to receive the referenced AIXM feature.

This leads to the following FIXM change request:

- FIXM CR TB12.6: Unrestricted URI Referencing

It should be noted, during the proposed implementation of the XLink standard in FIXM, XLink security concerns were considered. The use of the XLink standard has been identified as a potential source of multiple security vulnerabilities by the FAA [8: http://www.faa.gov/nextgen/programs/swim/governance/standards/media/Syntax%2%0A0and%20Processing%20of%20XML-Based%20Documents%2006162015.pdf]. Security vulnerabilities identified include allowing a malicious consumer to insert into an otherwise valid message a modified XLink value that contains an exploit. As a result, the document outlaws the use of the XLink standard in XML schema development internally within the FAA. However, as stated within the document, this restriction does not apply to external organizations developing XML schema - including FIXM development in Testbed 12.

7.7. Remove Restrictions on Natural Keys and Coordinates

During the testbed it was observed that FIXM is too restrictive in how aerodrome (`fb:AerodromeReferenceType`) and points (`fb:SignificantPointType`) can be specified.

First, it is not allowed to provide the natural key (ICAO designator) and the geometry (aerodrome reference point) at the same time. This is especially problematic since the natural key is provided in a highly FIXM-specific way. For example, aerodrome can only be resolved if the receiver has intimate knowledge about:

- how natural keys are encoded in FIXM, and
- how to resolve these to aerodrome name and reference point.

Second, the same issue exists for points, where a service is prevented from performing the resolution from a point designator to the actual coordinates, and providing both to the client.

Third, if a point is given by intersection parameters (bearing, distance and reference point) and the service calculates the resulting position on behalf of the client, it is unable to communicate both the position and intersection parameters to the client.

This leads to the following FIXM change requests:

- FIXM CR TB12.7: Unrestricted Natural Key Refs for Aerodrome
- FIXM CR TB12.8: Unrestricted Natural Key Refs for Points

7.7.1. FIXM Aerodrome Reference

```
...
<fx:departure>
  <fx:departureAerodrome xsi:type="fb:IcaoAerodromeReferenceType" code="KATL"/>
</fx:departure>
...
```

This ICAO designator, KATL, identifies a specific aerodrome, Hartsfield-Jackson Atlanta International Airport. A consumer of this data may want further information about the referenced feature, for example to visualize this data a consumer would need to resolve this ICAO designator to a geographic location.

To harmonize FIXM with AIXM and allow consumers to locate additional information for referenced features, the use of XLink in FIXM was investigated. By using XLink, internal or external links can be added to XML documents in a standardized format.

For example, the below example presents the same departure aerodrome as the previous example, however uses the xlink:href attribute to reference an external location. In this example, the external location is a WFS endpoint.

7.7.2. FIXM Aerodrome Reference with XLink External Reference

```
...
<fx:departure>
  <fx:departureAerodrome xsi:type="fb:IcaoAerodromeReferenceType" code="KATL"
xlink:href="https://service.example.com/airports?id=234"/>
</fx:departure>
...
```

The use of XLink also offers the opportunity to add human-readable descriptions using the xlink:title attribute. In the example below, the name attribute has been replaced with xlink:title.

7.7.3. FIXM Aerodrome Reference with XLink Human-Readable Description

```
...
<fx:departure>
  <fx:departureAerodrome xsi:type="fb:UnlistedAerodromeReferenceType" xlink:title=
"Hartsfield-Jackson Atlanta International Airport"/>
</fx:departure>
...
```

Chapter 8. Reducing FIXM XML File Size

Maintaining compact encoding of FIXM formed a major component of the feedback from previous OGC interoperability work. The rationale driving this requirement is the need for lightweight FIXM message transmission.

Although not directly investigated during Testbed 12, previous investigations of both (1) Binary XML, and (2) Generic XML compression provide insight into possible methods of maintaining lightweight FIXM transmission [9: Efficient XML and Weather, Briefing to WXXM/AIXM Conference - http://www.aixm.aero/sites/aixm.aero/files/imce/library/AIXM_WXXM_Conf_2010/25_day_2_-*wx*-_05_efficientxml-and-weather-case-study-2010-05-04.pdf] [10: Efficient XML Report, v2.0 - https://wiki.ucar.edu/download/attachments/23364539/Efficient%20XML%20Report%20v2.docx?version=1&modificationDate=1364227447000&api=v2]. Binary XML and Generic XML compression is summarized below.

8.1. Binary XML

Verbose XML encoding impacts both bandwidth and latency, especially in context of ground-to-air communications, and data archival and storage. Using a binary XML format reduces the verbosity of standard XML documents and will reduce the cost of parsing, transmitting, and storing the data. Previously, binary XML compression has been investigated for addressing issues surrounding XML verbosity in IWXXM [11: https://wiki.ucar.edu/download/attachments/23364539/Efficient%20XML%20Report%20v2.docx?version=1&modificationDate=1364227447000&api=v2]. Similar XML verbosity concerns have been expressed by FIXM stakeholders.

8.2. Generic Compression

An alternative to using binary XML, discussed above, is to use conventional XML document compression technologies, for example, GZIP [12: http://www.gzip.org/]. Whilst the use of binary XML offers reduced file size, decreased parsing time and random access, traditional compression offers only reduced file size.

Chapter 9. Change Requests for FIXM

The FIXM has an established process of governance, administered by a Change Control Board (CCB). Under the governance rules, there exists a formal process for implementing change requests. Any proposed change to the FIXM artifacts must be submitted for approval by the FIXM CCB. Reasons for change requests include but not limited to alignment with international standards such as ICAO and IATA, and compliance with the FF-ICE implementation guidance. Any changes outside of this scope must be justified by providing sufficient value as to be approved by the FIXM CCB (see FIXM Change Management Charter).

With this in mind, numerous FIXM change requests have been proposed following the investigation and discussion during the testbed. Each of the following change requests for the FIXM standard solves one or more specific interoperability issues identified during the testbed. However, not all change requests are concerned with introducing GML types into FIXM. The issues also reached into related topics such as the WFS standard, the flexibility of referencing and the XLink standard.

For clarity, the change requests are kept formally independent from each other to the most possible extent. Nevertheless, it is recommended to apply all of them. Most change requests are relevant to each other, such connections are explained in the respective descriptions.

The change requests are numbered with a "TB12" prefix (TB12.1, TB12.2, ...) to facilitate referencing these change requests during the process through the FIXM CCB.

The schema changes are based on the FIXM 4.0 Pre-Release Draft version 2 (`FIXM_4.0_Pre-Release_Draft_v2.zip`). The FIXM NAS extension is considered as well, but needs adjustment only in change request TB12.5.

All change requests have been verified through the setup of Web Feature Services serving FIXM 4.0-GML datasets by Snowflake and m-click. Moreover, it has been verified through the m-click Validator that the GML Aviation Profile works not just with AIXM 5.1, but also flawlessly with the proposed FIXM 4.0-GML schema (see TB12.9).

9.1. FIXM CR TB12.1: Minimize set of Feature Types

9.1.1. Description

NOTE	This issue has already been addressed in the latest FIXM 4.0 draft by removing `fb:FeatureType`.

This change request is a preparation for the change requests TB12.2 and TB12.3. It ensures that the scope of these change requests is limited as much as possible, which vastly reduces the total number of required changes to the FIXM XML Schema.

The current FIXM 4.0 draft defines many feature types that aren't actually meant to be used as stand-alone features. When switching to GML feature types as of TB12.2, this would impose restrictions on these types as described in TB12.3. Moreover, this would add a required `gml:id` attribute to all these types, contrary to the desired compactness of FIXM.

To prevent these complications, the following types should no longer be feature types. That is, these types should no longer extend from `fb:FeatureType` and their elements should be removed from the substitution group `fb:Feature`:

- `fb:ExtensionType`
- `fx:AircraftType`
- `fx:DangerousGoodsType`
- `fx:EnRouteType`
- `fx:AircraftPositionType`
- `fx:FlightArrivalType`
- `fx:FlightDepartureType`
- `fx:FlightEmergencyType`
- `fx:FlightRouteType`
- `fx:FlightStatusType`

The single remaining feature type is `fx:FlightType`, which is the only real feature type in FIXM.

Note that in particular, `fb:ExtensionType` should no longer be a feature type. Otherwise, all future extension types would have to take care of the restrictions as described in TB12.3. This is in line with the FIXM documentation, which clarifies that the extension type is meant to be used for objects, not features:

> The Extension type is the base type from which extension (non-core) objects inherit.
>
> Extension objects can be attached to a Flight through the "extensions" element.
>
> — FIXM 4.0 draft XML Schema, annotation of `fb:ExtensionType` in `base/Extension.xsd`

This also fits well with the current use of `fb:ExtensionType` in existing FIXM extensions. For

instance, the FIXM NAS extension *does not* define its feature type `nas:NasFlightType` to extend from `fb:ExtensionType`, but instead from `fx:FlightType`, which is perfectly in line with this change request.

9.1.2. Schema Changes

None. This issue has already been addressed in the latest FIXM 4.0 draft.

9.2. FIXM CR TB12.2: Switch to GML Feature Types

9.2.1. Description

To process FIXM with common GML tools, such as serving FIXM through a WFS, all FIXM feature types must be marked as GML feature types. That means:

1. All feature types must extend directly or indirectly from `gml:AbstractFeatureType`.
2. All feature elements must be in the substitution group `gml:AbstractFeature`.

Assuming that TB12.1 is accepted, hence `fx:Flight` being the only FIXM feature type, these reduce to the following changes:

1. `fx:FlightType` must extend directly or indirectly from `gml:AbstractFeatureType`.
2. `fx:Flight` must be in the substitution group `gml:AbstractFeature`.

Note that this adds a new required attribute `gml:id` on the `<fx:Flight>` element, more to that in TB12.4.

9.2.2. Schema Changes

Schema Change TB12.2.1

Import GML namespace where needed:

flight/flight/FlightData.xsd

```
<schema ... xmlns:gml="http://www.opengis.net/gml/3.2" ...>
  ...
  <import namespace="http://www.opengis.net/gml/3.2"
          schemaLocation="http://schemas.opengis.net/gml/3.2.1/gml.xsd"/>
  ...
</schema>
```

Schema Change TB12.2.2

Change definition of type `fx:FlightType` from:

flight/flightdata/FlightData.xsd

```
<complexType name="FlightType">
  <annotation>...</annotation>
  <sequence>...</sequence>
  <attribute ...>...</attribute>
  <attribute ...>...</attribute>
</complexType>
```

to:

flight/flightdata/FlightData.xsd

```
<complexType name="FlightType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>...</sequence>
      <attribute ...>...</attribute>
      <attribute ...>...</attribute>
    </extension>
  </complexContent>
</complexType>
```

Schema Change TB12.2.3

Change definition of element `fx:Flight` from:

flight/flight/FlightData.xsd

```
<element name="Flight" type="fx:FlightType"/>
```

to:

flight/flight/FlightData.xsd

```
<element name="Flight" type="fx:FlightType" substitutionGroup="gml:AbstractFeature"/>
```

9.3. FIXM CR TB12.3: Relocate top-level Attributes of Feature Types

9.3.1. Description

The WFS standard states that the properties of a feature must be XML elements, not XML attributes. (Here the `gml:id` does not count as "property".) This means that all features types should have no top-level XML attributes besides `gml:id`. Otherwise, compliant WFS implementations are allowed to refuse such datasets, or worse, to silently ignore these attributes (except for `gml:id`). Note that this applies only to XML attributes on feature type level, while sub elements may have XML attributes.

Assuming that TB12.1 is accepted, hence `fx:Flight` being the only FIXM feature type, this means that only the direct attributes of `<fx:Flight>` have to be converted to elements.

This affects the following attributes:

- `flightType` (type `fx:TypeOfFlightType`)
- `remarks` (type `fb:FreeTextType`)

In summary, the following FIXM document:

```
<fx:Flight
    ...
    flightType="S"
    remarks="Some Remarks">
  ...
</fx:Flight>
```

is changed to:

```
<fx:Flight
    ...>
  <fx:flightType>S</fx:flightType>
  <fx:remarks>Some Remarks</fx:remarks>
  ...
</fx:Flight>
```

While this increases the size of FIXM documents by a few bytes, this is a one-time increase per `<fx:Flight>` feature. Moreover, this affects only datasets where these properties are actually set.

9.3.2. Schema Changes

Schema Change TB12.3.1

Change definition of type `fx:FlightType` from:

flight/flight/FlightData.xsd

```
<complexType name="FlightType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        ...
      </sequence>
      <attribute name="flightType" ...>...</attribute>
      <attribute name="remarks" ...>...</attribute>
    </extension>
  </complexContent>
</complexType>
```

to:

flight/flight/FlightData.xsd

```
<complexType name="FlightType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="flightType" type="..." minOccurs="0" maxOccurs="1">...</ele...>
        <element name="remarks" type="..." minOccurs="0" maxOccurs="1">...</element>
        ...
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Schema Change TB12.3.2

It should be documented at a prominent place within the FIXM standard that for interoperability reasons, future revisions of FIXM should not add attributes directly to feature type elements.

Such a prominent place may be the documentation annotation of the type `fx:FlightType`.

9.4. FIXM CR TB12.4: Define Business Rule for `gml:id` in `<fx:Flight>`

9.4.1. Description

As a consequence of TB12.2 and TB12.5, there is a new required attribute `gml:id` on the `<fx:Flight>` element and all point geometries. Although this goes slightly contrary to the compactness goal of FIXM, the `gml:id` is currently unavoidable (as of GML 3.2) and important for a variety of WFS operations, filter expressions and so on. A business rule should be established on how to fill the `gml:id`. That is, the FIXM standard should state a policy such as the following:

> At feature as well as geometry level, the `gml:id` should be filled with a short dummy identifier that is document-wide unique:
>
> ```
> <fx:Flight gml:id="ID_1">
>   ...
>     <gml:Point ... gml:id="ID_3">
>       ...
>     </gml:Point>
>   ...
> </fx:Flight>
> ```
>
> The `gml:id` attribute should not be used for global identification purposes, for which only `<gml:identifier>` should be used.

In case TB12.10 is not accepted, replace `<gml:identifier>` with `<fx:gufi>` in the above statement.

9.4.2. Schema Changes

None. This change request is about establishing an informal policy in FIXM which cannot be expressed in XML Schema.

9.5. FIXM CR TB12.5: Introduce `gml:Point`

9.5.1. Description

The use of GML geometries plays a central role in improving the interoperability of FIXM, where `gml:PointType` is the most relevant for FIXM.

All point geometries in FIXM use or extend from `fb:GeographicalPositionType`, whose definition is already very similar to `gml:PointType`. To finish the transition to `gml:PointType`,

1. Remove `fb:GeographicalPositionType` and use `gml:PointType` instead.
2. Use `gml:Point` instead of `fb:GeographicalPosition` as substitution group.

Note that this adds a new required attribute `gml:id` to all point elements, which is currently unavoidable in GML. However, this slight increase in size can be turned into an overall more compact notation through TB12.7 and TB12.8.

9.5.2. Schema Changes

Schema Change TB12.5.1

Import GML namespace where needed:

base/Airspace.xsd, base/Organization.xsd, flight/flightroutetrajectory/RouteTrajectory.xsd, NasPosition.xsd and NasRoute.xsd

```
<schema ... xmlns:gml="http://www.opengis.net/gml/3.2" ...>
  ...
  <import namespace="http://www.opengis.net/gml/3.2"
          schemaLocation="http://schemas.opengis.net/gml/3.2.1/gml.xsd"/>
  ...
</schema>
```

Schema Change TB12.5.2

Change definition of type `fb:PositionPointType` from:

base/Airspace.xsd

```
<complexType name="PositionPointType">
  ...
    <element name="position" type="fb:GeographicalPositionType" ...>
  ...
</complexType>
```

to:

base/Airspace.xsd

```
<complexType name="PositionPointType">
  ...
    <element name="position" type="gml:PointType" ...>
  ...
</complexType>
```

Schema Change TB12.5.3

Change definition of type `fb:OtherUnitReferenceType` from:

base/Organization.xsd

```
<complexType name="OtherUnitReferenceType">
  ...
    <element name="position" type="fb:GeographicalPositionType" ...>
  ...
</complexType>
```

to:

base/Organization.xsd

```
<complexType name="OtherUnitReferenceType">
  ...
    <element name="position" type="gml:PointType" ...>
  ...
</complexType>
```

Schema Change TB12.5.4

Change definition of type `fx:TrajectoryPoint4DType` from:

flight/flightroutetrajectory/RouteTrajectory.xsd

```
<complexType name="TrajectoryPoint4DType">
  ...
    <extension base="fb:GeographicalPositionType">
  ...
</complexType>
```

to:

flight/flightroutetrajectory/RouteTrajectory.xsd

```
<complexType name="TrajectoryPoint4DType">
  ...
    <extension base="gml:PointType">
  ...
</complexType>
```

Schema Change TB12.5.5

Change definition of element `<fx:TrajectoryPoint4D>` from:

flight/flightroutetrajectory/RouteTrajectory.xsd

```
<element name="TrajectoryPoint4D" type="..." substitutionGroup="fb:GeographicalP..."/>
```

to:

flight/flightroutetrajectory/RouteTrajectory.xsd

```
<element name="TrajectoryPoint4D" type="..." substitutionGroup="gml:Point"/>
```

Schema Change TB12.5.6

Change definition of type `nas:NasAircraftPositionType` from:

NasPosition.xsd

```
<complexType name="NasAircraftPositionType">
  ...
    <element name="targetPosition" type="fb:GeographicalPositionType" ...>
  ...
</complexType>
```

to:

NasPosition.xsd

```
<complexType name="NasAircraftPositionType">
  ...
    <element name="targetPosition" type="gml:PointType" ...>
  ...
</complexType>
```

Schema Change TB12.5.7

Change definition of type `nas:NasRouteInformationType` from:

NasRoute.xsd

```
<complexType name="NasRouteInformationType">
  ...
    <element name="tfmsNextEvent" type="fb:GeographicalPositionType" ...>
    <element name="tfmsNextPosition" type="fb:GeographicalPositionType" ...>
  ...
</complexType>
```

to:

NasRoute.xsd

```
<complexType name="NasRouteInformationType">
  ...
    <element name="tfmsNextEvent" type="gml:PointType" ...>
    <element name="tfmsNextPosition" type="gml:PointType" ...>
  ...
</complexType>
```

Schema Change TB12.5.8

Remove the whole definition of type `fb:GeographicalPositionType` and element `<fb:GeographicalPosition>`:

base/Airspace.xsd

```
<complexType name="GeographicalPositionType">
  ...
</complexType>
<element name="GeographicalPosition" .../>
```

9.6. FIXM CR TB12.6: Unrestricted URI Referencing

9.6.1. Description

The goal of this change request is to reduce the gap between AIXM and FIXM through the XLink standard, so that FIXM flights have the possibility to reference AIXM features as they see fit.

NOTE	The change described here is fully backwards compatible. Existing FIXM documents do not need to be converted to account for this change.

This is an important improvement in terms of interoperability. Among others, this is the key enabler for automatic link resolving in Web Feature Services (WFS) through the `resolveDepth="…"` parameter.

Moreover, using XLink references enables unambiguous references from FIXM flights to basic data. Currently, FIXM links to basic data solely through the use of natural keys, which is sufficent for most use cases, but does not provide referential integrity in the strict sense. For example, the names of designated points are only unique within a certain radius. This is not a problem within a single country, but provides challenges when linking datasets globally.

NOTE	This change requests solves some, but not all issues with referencing. The remaining issues are the topic of TB12.7 and TB12.8.

The following FIXM types would benefit from XLink capabilities:

`fb:AerodromeReferenceType`
: may wish to link to an `aixm:AirportHeliport` feature.

`fb:SignificantPointType`
: may wish to link to an `aixm:DesignatedPoint` or `aixm:Navaid` feature.

Adding XLink support means appending the attribute group `xlink:simpleAttrs` to these types. This adds several optional attributes from the `link:` namespace, of which `xlink:href` is the most important one.

The XLink standard does not provide any restriction on the kind of URI to be linked to. Depending on the context, different kinds of links may be appropriate:

- Local references such as `xlink:href="#uuid.89863…"` are useful if the FIXM flight data and AIXM basic data are provided within the same document or set of documents.
- Abstract references such as `xlink:href="urn:uuid:89863…"` are useful if just the identifier (UUID) of the corresponding AIXM feature is to be communicated and the receiver is expected to be capable of URN resolving.
- External absolute references such as `xlink:href="https://example.com/get/89863…"` are useful if the AIXM features are served through a different service and the FIXM service wants to direct the receiver to a specific authoritative basic data service such as an AIXM WFS. The link then describes an HTTP GET request to that other service to receive the referenced AIXM feature.
- External relative references such as `xlink:href="./89863…"` are useful if the FIXM flight data

and AIXM basic data are served through the same service, but do not appear in the same response document. The link then describes an HTTP GET request to the same service to receive the referenced AIXM feature.

9.6.2. Schema Changes

Schema Change TB12.6.1

Import XLink namespace where needed:

base/Aerodrome.xsd and base/Airspace.xsd

```
<schema ... xmlns:xlink="http://www.w3.org/1999/xlink" ...>
  ...
  <import namespace="http://www.w3.org/1999/xlink"
          schemaLocation="http://www.w3.org/1999/xlink.xsd"/>
  ...
</schema>
```

Schema Change TB12.6.2

Append attribute group `xlink:simpleAttrs` to type `fb:AerodromeReferenceType`:

base/Aerodrome.xsd

```
<complexType name="AerodromeReferenceType">
  ...
  <attributeGroup ref="xlink:simpleAttrs"/>
</complexType>
```

Schema Change TB12.6.3

Append attribute group `xlink:simpleAttrs` to type `fb:SignificantPointType`:

base/Airspace.xsd

```
<complexType abstract="true" name="SignificantPointType">
  ...
  <attributeGroup ref="xlink:simpleAttrs"/>
</complexType>
```

9.7. FIXM CR TB12.7: Unrestricted Natural Key Refs for Aerodrome

9.7.1. Description

The goal of this and the following TB12.8 change request is to make the FIXM XML Schema better suited for usage in generic OGC services such as WFS or WPS, while at the same time making FIXM documents more compact, because the verbose explicit `xsi:type="⋯"` attributes are no longer required for aerodrome (TB12.7) and point (TB12.8) references. So in contrast to the other TB12 FIXM CRs, this is not a trade-off between standards compliance and compactness, but a win-win for both interoperability and compactness.

NOTE	The change described here is fully backwards compatible. Existing FIXM documents do not need to be converted to account for this change.

FIXM is very restrictive in how aerodrome (`fb:AerodromeReferenceType`) can be specified. These are either:

- by ICAO designator (`fb:IcaoAerodromeReferenceType`), or
- by point geometry and name (`fb:OtherReferenceType`)

In particular, it is not allowed to provide the natural key (ICAO designator) and the geometry (aerodrome reference point) at the same time. This is especially problematic since the natural key is provided in a highly FIXM-specific way, meaning these can only be resolved if the receiver has intimate knowledge about:

- how natural keys are encoded in FIXM, and
- how to resolve these to aerodrome name and reference point.

This is not an issue when two highly specialized systems communicate with each other, but for generic OGC services such as WFS or WPS an important interoperability aspect is that the server is able to resolve these references on behalf of the client, to facilitate displaying of FIXM datasets in generic clients.

For example, a flight from the Hartsfield–Jackson Atlanta International Airport (`KATL`) can be communicated to the client either as pure natural key:

```
<fx:aerodrome
    xsi:type="fb:IcaoAerodromeReferenceType"
    locationIndicator="KATL"/>
```

or as resolved information:

```
<fx:aerodrome
    xsi:type="fb:OtherReferenceType"
    name="HARTSFIELD-JACKSON ATLANTA INTERNATIONAL AIRPORT">
  <fb:referencePoint xsi:type="fb:PositionPointType">
    <fb:position srsName="urn:ogc:def:crs:EPSG::4326" gml:id="ID_2">
      <gml:pos>33.6 -84.4</gml:pos>
    </fb:position>
  </fb:referencePoint>
</fx:aerodrome>
```

Also note the verbose specification of `xsi:type`, whose sole purpose is to ensure that exactly one of the two sets of information can ever be provided — the natural key or the resolved information.

A simpler solution, which is at the same time more compact, would be to merge these types.

That way, the previous examples would remain valid, but in addition, a generic service would be able to provide all information it has:

```
<fx:departureAerodrome
    locationIndicatorIcao="KATL"
    name="Hartsfield–Jackson Atlanta International Airport">
  <fb:aerodromeReferencePoint>
    <fb:position srsName="urn:ogc:def:crs:EPSG::4326" gml:id="ID_2">
      <gml:pos>33.6 -84.4</gml:pos>
    </fb:position>
  </fb:aerodromeReferencePoint>
</fx:departureAerodrome>
```

For the sake of backwards compatibility, the types `fb:IcaoAerodromeReferenceType` and `fb:OtherReferenceType` should be kept as empty extensions from `fb:AerodromeReferenceType`, so that documents are still allowed to specify these `xsi:type` values, even though this is not necessary anymore. Future versions of FIXM may decide to remove these types.

NOTE

This change request does not affect the extensibility through `xsi:type` in any way. It is still possible for extensions to create new types which extend from `fb:AerodromeReferenceType`, and to use these as before:

```
<fx:departureAerodrome xsi:type="fb:NewAerodromeReferenceType">
  ...
</fx:departureAerodrome>
```

9.7.2. Schema Changes

Schema Change TB12.7.1

This schema change assumes that TB12.6 has been accepted. Otherwise, the same change applies

without the attribute group `xlink:simpleAttrs`.

Change definition of type `fb:AerodromeReferenceType` from:

base/Aerodrome.xsd

```
<complexType abstract="true" name="AerodromeReferenceType">
  <annotation>...</annotation>
  <attributeGroup ref="xlink:simpleAttrs"/>
</complexType>
<element name="AerodromeReference" type="fb:AerodromeReferenceType"/>
```

to:

base/Aerodrome.xsd

```
<complexType name="AerodromeReferenceType">
  <annotation>...</annotation>
  <sequence>
    <element name="referencePoint" ...>...</element>
  </sequence>
  <attribute name="iataDesignator" ...>...</attribute>
  <attribute name="name" ...>...</attribute>
  <attribute name="locationIndicator" ...>...</attribute>
  <attributeGroup ref="xlink:simpleAttrs"/>
</complexType>
<element name="AerodromeReference" type="fb:AerodromeReferenceType"/>
```

Schema Change TB12.7.2

Change definition of type `fb:IcaoAerodromeReferenceType` from:

base/Aerodrome.xsd

```
<complexType name="IcaoAerodromeReferenceType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="fb:AerodromeReferenceType">
      ...
    </extension>
  </complexContent>
</complexType>
<element name="IcaoAerodromeReference" .../>
```

to:

base/Aerodrome.xsd

```
<complexType name="IcaoAerodromeReferenceType">
  <annotation>
    <documentation>This type exists only for backwards compatibility.</documentation>
  </annotation>
  <complexContent>
    <extension base="fb:AerodromeReferenceType"/>
  </complexContent>
</complexType>
```

Schema Change TB12.7.3

Change definition of type `fb:OtherReferenceType` from:

base/Aerodrome.xsd

```
<complexType name="OtherReferenceType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="fb:AerodromeReferenceType">
      ...
    </extension>
  </complexContent>
</complexType>
<element name="UnlistedAerodromeReference" .../>
```

to:

base/Aerodrome.xsd

```
<complexType name="OtherReferenceType">
  <annotation>
    <documentation>This type exists only for backwards compatibility.</documentation>
  </annotation>
  <complexContent>
    <extension base="fb:AerodromeReferenceType"/>
  </complexContent>
</complexType>
```

9.8. FIXM CR TB12.8: Unrestricted Natural Key Refs for Points

9.8.1. Description

This change request is the continuation of TB12.7, simplifying the various point types in FIXM. This change request may be considered even more important than TB12.7, given that typical FIXM documents contain more point specifications than aerodrome specifications.

NOTE The change described here is fully backwards compatible. Existing FIXM documents do not need to be converted to account for this change.

FIXM is very restrictive in how points (`fb:SignificantPointType`) can be specified. These are either:

- by bearing, distance and reference point (`fb:RelativePointType`), or
- by just the position (`fb:PositionPointType`), or
- by the point designator (`fb:DesignatedPointOrNavaidType`).

Similar to the aerodrome case in TB12.7, a generic service is unable to provide all its knowledge to the client:

1. The service is prevented from performing the resolution from a point designator to the actual coordinates, and providing both to the client.
2. If a point is given by FRD parameters (bearing, distance and reference point) and the service calculates the resulting position on behalf of the client, it is unable to communicate both the position and FRD parameters to the client.

By merging the point types, a service would be able to provide all information it has to the client.

For the sake of backwards compatibility, the types `fb:RelativePointType`, `fb:PositionPointType` and `fb:DesignatedPointOrNavaidType` should be kept as empty extensions from `fb:SignificantPointType`, so that documents are still allowed to specify these `xsi:type` values, even though this is not necessary anymore. Future versions of FIXM may decide to remove these types.

NOTE This change request does not affect the extensibility through `xsi:type` in any way. It is still possible for extensions to extend from `fb:SignificantPointType` and to use it as before:

```
<fx:routePoint xsi:type="ext:NewPointType">
  ...
</fx:routePoint>
```

9.8.2. Schema Changes

Schema Change TB12.8.1

This schema change assumes that TB12.6 has been accepted. Otherwise, the same change applies without the attribute group `xlink:simpleAttrs`.

Change definition of type `fb:SignificantPointType` from:

base/Airspace.xsd

```
<complexType abstract="true" name="SignificantPointType">
  <annotation>...</annotation>
  <attributeGroup ref="xlink:simpleAttrs"/>
</complexType>
<element name="SignificantPoint" type="fb:SignificantPointType"/>
```

to:

base/Airspace.xsd

```
<complexType name="SignificantPointType">
  <annotation>...</annotation>
  <sequence>
    <sequence>
      <annotation>...</annotation>
      <element name="bearing" ...>...</element>
      <element name="distance" ...>...</element>
      <element name="referencePoint" ...>...</element>
    </sequence>
    <sequence>
      <annotation>...</annotation>
      <element name="position" ...>...</element>
    </sequence>
  </sequence>
  <attribute name="designator" ...>...</attribute>
  <attributeGroup ref="xlink:simpleAttrs"/>
</complexType>
<element name="SignificantPoint" type="fb:SignificantPointType"/>
```

Schema Change TB12.8.2

Change definition of type `fb:RelativePointType` from:

base/Airspace.xsd

```
<complexType name="RelativePointType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="fb:SignificantPointType">
      ...
    </extension>
  </complexContent>
</complexType>
<element name="RelativePoint" .../>
```

to:

base/Airspace.xsd

```
<complexType name="RelativePointType">
  <annotation>
    <documentation>This type exists only for backwards compatibility.</documentation>
  </annotation>
  <complexContent>
    <extension base="fb:SignificantPointType"/>
  </complexContent>
</complexType>
```

Schema Change TB12.8.3

Change definition of type `fb:PositionPointType` from:

base/Airspace.xsd

```
<complexType name="PositionPointType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="fb:SignificantPointType">
      ...
    </extension>
  </complexContent>
</complexType>
<element name="PositionPoint" .../>
```

to:

base/Airspace.xsd

```
<complexType name="PositionPointType">
  <annotation>
    <documentation>This type exists only for backwards compatibility.</documentation>
  </annotation>
  <complexContent>
    <extension base="fb:SsignificantPointType"/>
  </complexContent>
</complexType>
```

Schema Change TB12.8.4

Change definition of type `fb:DesignatedPointOrNavaidType` from:

base/Airspace.xsd

```
<complexType name="DesignatedPointOrNavaidType">
  <annotation>...</annotation>
  <complexContent>
    <extension base="fb:SignificantPointType">
      ...
    </extension>
  </complexContent>
</complexType>
<element name="DesignatedPointOrNavaid" .../>
```

to:

base/Airspace.xsd

```
<complexType name="DesignatedPointOrNavaidType">
  <annotation>
    <documentation>This type exists only for backwards compatibility.</documentation>
  </annotation>
  <complexContent>
    <extension base="fb:SignificantPointType"/>
  </complexContent>
</complexType>
```

9.9. FIXM CR TB12.9: Make use of the GML Aviation Profile

9.9.1. Description

A GML profile is a subset of the GML Schema that is sufficient for a certain domain. Usually, a GML profile is related to one or more GML application schemas. For restrictions that are not expressible with XML Schema, a GML profile may provide additional business rules to further reduce the subset of GML.

For the aviation domain such a GML profile has been developed in Use of GML for aviation data which was developed mainly with AIXM in mind, but may be applicable for FIXM, too. This GML Aviation Profile reduces the set of GML geometries, the allowed coordinate reference systems, and so on.

Since FIXM also just wants to use a subset of GML, it may choose to reuse at least the XML Schema provided by the GML Aviation Profile. This will work flawlessly, because all TB12 change requests introduce only parts of GML that are used by AIXM as well.

An alternative way for FIXM may be to define its own, even more reduced, GML profile. However, this may fragment the efforts to create a GML profile for the whole aviation domain.

9.9.2. Schema Changes

Schema Change TB12.9.1

Specifying the GML profile is not expressible directly via XML Schema, because GML profiles always use the same XML namespace as the full GML schema.

So this decision needs to be communited through documentation annotations in the FIXM Schema.

In addition, this decision should be communicated through in the FIXM standard itself.

9.10. FIXM CR TB12.10: Use <gml:identifier> instead of <fx:gufi>

9.10.1. Description

FIXM should make use of <gml:identifier> instead of their custom <fx:gufi> field. Both have the same structure and serve the same purpose: identifying a single GML feature through a globally unique identifier. So it is natural to reuse the existing <gml:identifier>. Also, the <gml:identifier> may receive some additional semantics in future WFS versions and related standards (possibly related to generic UUID resolution or other UUID topics). Such improvements would then be inapplicable to FIXM due to its custom <fx:gufi> identifier field.

The <gml:identifier> is already directly available to FIXM through gml:AbstractFeatureType as introduced by TB12.2. So just the <fx:gufi> element needs to be removed. However, its type fb:GloballyUniqueFlightIdentifierType should be kept as it is also used by <diversionRecoveryInformation>.

The existing restrictions on the GUFI should be provided through a business rule or through a GML Application Schema (see TB12.9).

For consistency with AIXM, the attribute value codeSpace="urn:uuid" should be changed to codeSpace="urn:uuid:", that is, with trailing colon.

A FIXM document will change from:

```
<fx:Flight ...>
  ...
  <fx:gufi codeSpace="urn:uuid">94ea0e57-8249-49d4-9fd9-5670e8357766</fx:gufi>
  ...
</fx:Flight>
```

to:

```
<fx:Flight ...>
  <gml:identifier
      codeSpace="urn:uuid:">94ea0e57-8249-49d4-9fd9-5670e8357766</gml:identifier>
  ...
</fx:Flight>
```

9.10.2. Schema Changes

Schema Change TB12.10.1

Remove definition of element fx:gufi in type fx:FlightType:

flight/flightdata/FlightData.xsd

```
<element name="gufi" ...>
  ...
</element>
```

9.11. FIXM CR TB12.11: Integration of ISO 19115 / 19139 Metadata

9.11.1. Description

Metadata is nowadays an essential part of data, enabling users to specify information about topics such as data identification, access, usage and quality. A widely used standard addressing such metadata requirements is the ISO 19115 model, together with its XML implementation, ISO 19139. AIXM 5 currently integrates this metadata standard by means of an optional metadata property on the feature collection, feature and time slice level.

The goal of this change request is to introduce a similar metadata property on a FIXM flight feature, enabling users to describe general metadata about the feature. One particular use case revealed in Testbed 12 is the ability to include lineage / provenance information for FIXM features in the Data Broker, to inform users about the origin of flight features in an aggregated flight data set.

NOTE	The change described here is fully backwards compatible. Existing FIXM documents do not need to be converted to account for this change.

The following FIXM type would benefit from an ISO 19115 metadata property:

`fx:FlightType`

This change request suggests to define a general metadata property in the Base XML Schema, with usage of this property in the Flight XML Schema. Having a general metadata property available in the Base XML Schema makes it easy to integrate metadata in other types in the future, depending on the need - e.g., at the message level, in extensions ...

9.11.2. Schema Changes

Schema Change TB12.11.1

Import the ISO metadata namespace where needed:

base/Base.xsd

```
<schema ... xmlns:gmd="http://www.isotc211.org/2005/gmd" ...>
  ...
  <import namespace="http://www.isotc211.org/2005/gmd"
          schemaLocation="http://www.isotc211.org/2005/gmd/gmd.xsd"/>
  ...
</schema>
```

Schema Change TB12.11.2

Define an optional metadata property, capable of holding an ISO 19115 MD_Metadata element.

base/Base.xsd

```
<complexType name="FeatureMetadataPropertyType">
  <complexContent>
    <extension base="gml:AbstractMetadataPropertyType">
      <sequence minOccurs="0">
        <element ref="gmd:MD_Metadata"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Schema Change TB12.11.3

Integrate the optional metadata property with a flight type.

flight/flightdata/FlightData.xsd

```
<complexType name="FlightType">
  ...
  <sequence>
    ...
    <element name="featureMetadata" type="fb:FeatureMetadataPropertyType"
             minOccurs="0">
      <annotation>
        <documentation>
        An ISO metadata property to describe general metadata about a flight feature.
        </documentation>
      </annotation>
    </element>
  </sequence>
  ...
</complexType>
```

Appendix A: Revision History

Table 2. Revision History

Date	Editor	Revision	Primary clauses modified	Descriptions
February 25, 2016	T. Disney	.1	all	initial commit
February 29, 2016	C. Chen	.2	all	initial version review
July 26, 2016	V. Diels-Grabsch & T. Forbes	1.0	all	Final Draft ER Release

Appendix B: Bibliography

[1] OGC: OGC Testbed 11 Demonstration. (2015).

[2] Lee, J., Zlatanova, S.: 3D geoinformation science. Springer (2009).