ETSI TS 103 689 V1.1.1 (2019-11)

Digital Audio Broadcasting (DAB);
Filtered Information Service (FIS);
Application specification

Reference

DTS/JTC-DAB-99

Keywords

broadcasting, DAB, digital, radio, XML

Contents

Intellectual Property Rights

Foreword

This Technical Specification (TS) has been produced by Joint Technical Committee (JTC) Broadcast of the European Broadcasting Union (EBU), Comité Européen de Normalisation ELECtrotechnique (CENELEC) and the European Telecommunications Standards Institute (ETSI).

NOTE 1: The EBU/ETSI JTC Broadcast was established in 1990 to co-ordinate the drafting of standards in the specific field of broadcasting and related fields. Since 1995 the JTC Broadcast became a tripartite body by including in the Memorandum of Understanding also CENELEC, which is responsible for the standardization of radio and television receivers. The EBU is a professional association of broadcasting organizations whose work includes the co-ordination of its members' activities in the technical, legal, programme-making and programme-exchange domains. The EBU has active members in about 60 countries in the European broadcasting area; its headquarters is in Geneva.

European Broadcasting Union
CH-1218 GRAND SACONNEX (Geneva)
Switzerland
Tel: +41 22 717 21 11
Fax: +41 22 717 24 81

The Eureka Project 147 was established in 1987, with funding from the European Commission, to develop a system for the broadcasting of audio and data to fixed, portable or mobile receivers. Their work resulted in the publication of European Standard, ETSI EN 300 401 [1], for DAB (see note 2) which now has worldwide acceptance.

NOTE 2: DAB is a registered trademark owned by one of the Eureka Project 147 partners.

The DAB family of standards is supported by World DAB, an organization with members drawn from broadcasting organizations and telecommunication providers together with companies from the professional and consumer electronics industry.

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document establishes an optional extension to the broadcast standard for Digital Audio Broadcasting (DAB) system.

The Filtered Information Service (FIS) is a data application that allows service providers to deliver information to groups of receivers with configurable filters, for example, text language, model number, date of registration, etc. The XML framework and transport specification are defined.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference/.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI EN 300 401 (V2.1.1): "Radio Broadcasting Systems; Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers".

[2] ETSI EN 301 234 (V2.1.1): "Digital Audio Broadcasting (DAB); Multimedia Object Transfer (MOT) Protocol".

[3] ETSI TS 101 756: "Digital Audio Broadcasting (DAB); Registered Tables".

[4] ISO 8601: "Data elements and interchange formats -- Information interchange -- Representation of dates and times".

[5] ISO/IEC 10646: "Information technology - Universal Coded Character Set (UCS)".

[6] IETF RFC 2045: "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies".

[7] ISO 3166-1: "Codes for the representation of names of countries and their subdivisions - Part 1: Country codes".

[8] "GeoRSS: Geographically Encoded objects for RSS feeds".

NOTE: Available at http://www.georss.org.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 101 499: "Hybrid Digital Radio (DAB, DRM, RadioDNS); SlideShow; User Application Specification".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI EN 300 401 [1] and the following apply:

display language: user defined language used for display of text elements

key: unique identifier used to address receiver devices allocated to a particular company

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

DAB	Digital Audio Broadcasting
FEC	Forward Error Correction
FIG	Fast Information Group
FIS	Filtered Information Service
GPS	Global Positioning System
HMI	Human Machine Interface
ISO	International Organization for Standardization
JFIF	JPEG File Interchange Format
JPEG	Joint Photographic Experts Group
MCI	Multiplex Configuration Information
MIME	Multimedia Internet Message Extensions
MJD	Modified Julian Date
MOT	Multimedia Object Transfer
MSC	Main Service Channel
OEM	Original Equipment Manufacturer
PNG	Portable Network Graphics
POI	Point-Of-Interest
URL	Uniform Resource Locator
UTF	Unicode Transformation Format
XML	eXtensible Markup Language
XSD	XML Schema Definition

4 Introduction

The Filtered Information Service (FIS)is an application used to send messages to suitably equipped DAB devices via the broadcast radio network. It may be used by a radio manufacturer, car manufacturer or third-party to disseminate information to their clients. The FIS allows the information to be identified by company defined filters in order to target particular information to particular device groups. The information is primarily text based, but images may also be provided.

FIS use cases can be very diverse, from kitchen radios to a company car fleet.

EXAMPLE 1: A kitchen radio maker provides news that updated software is available for download via an internet address, or that it has new models available.

EXAMPLE 2: An OEM wishes to reach vehicle owners about important information such as a dealer recall but has lost contact with some owners due to vehicle resale; the information may be targeted at a particular range of vehicles based on the model year, transmission system, etc.

EXAMPLE 3: A car rental company provides general messages to groups of vehicles based on their location, age, etc. and specific details of individual rental issues which are only displayed in the corresponding vehicle.

Since the FIS uses the DAB broadcast network, there is no need for special infrastructure to transmit the FIS information. The broadcaster or network operator embeds messages into the DAB broadcast according to the defintions provided by the present document. The receiver implements security and message filtering mechanisms in order to display the appropriate information. It is not necessary for the receiver to have continuous reception of the FIS, because the information is cyclically repeated and cached in the receiver.

Each FIS has a unique company identifier, called a key, which allows receivers to focus on only the messages of interest. Messages may be provided securely to end-user devices by the use of encryption. A FIS consists of an XML document and may include additional image files.

An overview of the FIS is provided in figure 1.

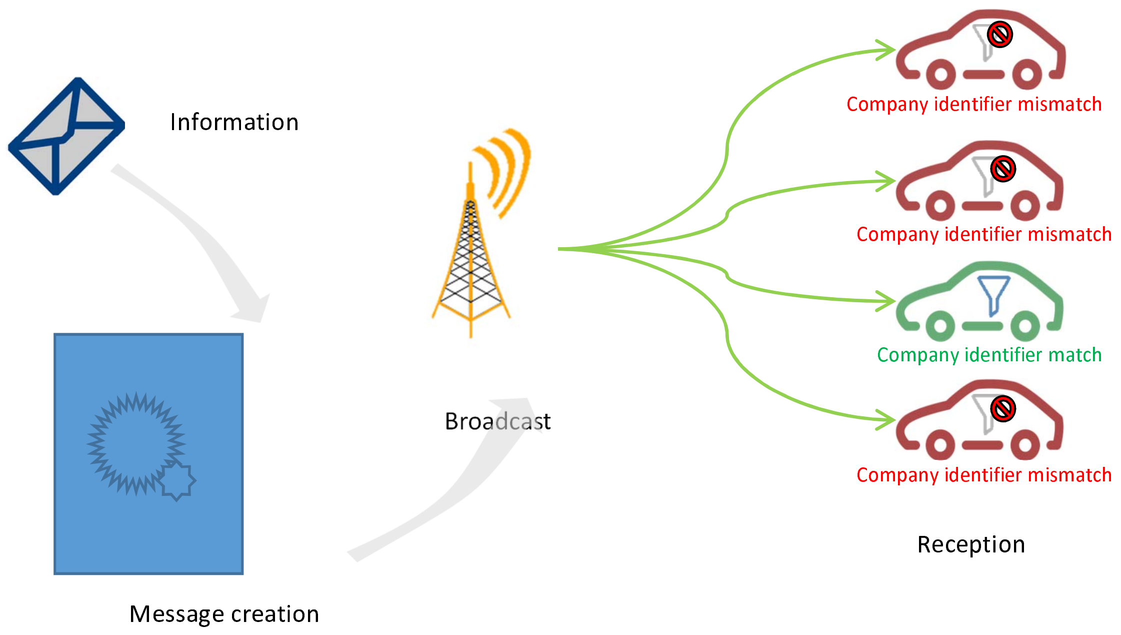

Information
Message creation
Broadcast
Company identifier mismatch
Company identifier mismatch
Company identifier match
Company identifier mismatch
Reception

Figure 1: Overview of the FIS

An entity can direct a message to a dedicated subset of its own receivers by defining and using filters linked to the product characteristics that the company wants to address.

EXAMPLE 4: An OEM can direct messages to specific car models from specific registration periods.

It is the company responsibility to define its own filters in a filter definition file. With this file, filters can be applied to messages in the broadcast XML file. The receiver will display the message only if associated filters are compliant with the receiver configuration file.

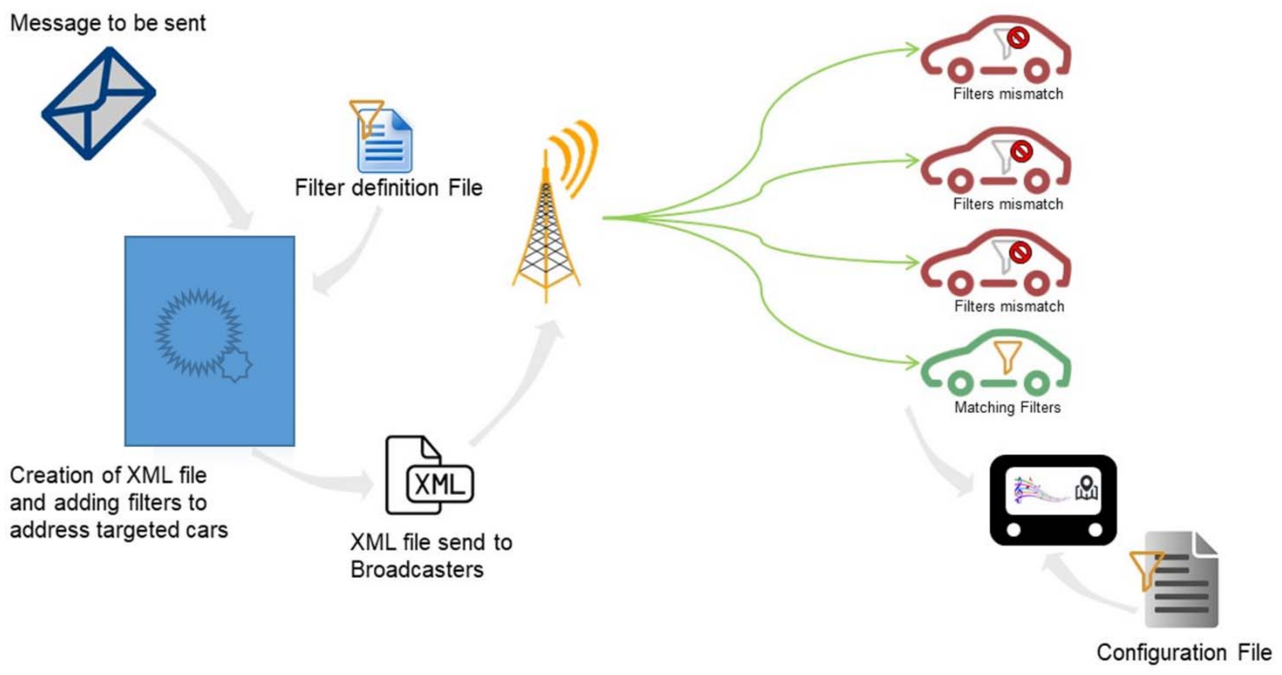


Figure 2: Example of use of the filter definition file

Messages from a particular service provider are grouped together into a FIS. The FIS may be encrypted to provide additional security, but the encryption method is not standardized. Encryption is not recommended when the FIS is directed to multiple receiver brands, for example, when a car rental company wants to send messages to the entire fleet; in this case different makes and models of cars with different receiver brands all need to be able to decode the FIS.

5 Filtered Information Service data

5.0 Introduction

A FIS consists of a set of messages, encoded as an XML file, and supporting data files, such as images, carried in an MOT carousel (see ETSI EN 301 234 [2]). The MOT carousel is transported using packet mode with additional FEC (see ETSI EN 300 401 [1]). Every file within the MOT carousel is identified with the company identifier to provide the first level of message selection in the receiver.

All XML documents defined in the present document shall use the ISO/IEC 10646 [5] character set using UTF-8 character encoding.

Text sections (attributes or elements) shall not use the following reserved XML characters:

& " '

These characters shall be encoded using the predefined entity references &amp &quot &apos respectively.

The reserved XML characters < > may be used in the `title` and `body` elements. In all other text sections they shall be encoded using the predefined entity references &lt &gt respectively.

5.1 FIS element

This is the root element of the FIS document and can contain zero or more of the following elements:

- message.

Its attributes are detailed below:

Attribute	Description	Type	Status
`key`	Company identifier for entity providing the information.	*xs: unsignedInt*	Required
`version`	Indicator for changed content; shall be incremented by one for every new version of the `fis` element, modulo 65536.	*xs:unsignedShort*	Required

5.2 Message element

The `message` element can contain the following elements:

- text
- picture
- validity
- geolocation
- filters

As a minimum, one `text` element and one `validity` element shall be specified for each message. The `validity` element specifies the lifetime of the message in the receiver.

Its attributes are detailed below:

Attribute	Description	Type	Status
`identifier`	Unique identifier	*xs:unsignedInt*	Required
`priority`	Message priority chosen from: **critical, important, major, normal, minor, low**	*xs:enumeration*	Optional, defaults to "normal"

The `identifier` attribute shall be set to a unique value for each message, in the range 0 to 4 294 967 295.

A message is created by using a new `identifier`.

A message is deleted in the receiver only when its `validity` has expired, even if it is no longer transmitted. The service provider shall therefore ensure that no `identifier` is reused until after the message has expired.

Only the `validity` element may be updated, see clause 5.5: this allows a message to have its validity extended or curtailed. If any other content is changed, a new message with a different `identifier` shall be created; the old message should continue to be transmitted with an expired validity to ensure its deletion in receivers.

Not all messages will have the same importance: some may be really important, for example, if they concern a safety issue, others may be less important, for example, those that concern only promotion.

The FIS has 6 levels of messages:

- 0: critical
- 1: important
- 2: major

- 3: normal (default)
- 4: minor
- 5: low

Messages with priority 0, 1 or 2 shall be displayed by the receiver within their validity period (subject to other selection criteria).

Messages with priority 3, 4 or 5 do not have to be displayed. It is up to the receiver whether these messages are displayed within their validity period, and this could be implemented as a user settings option in the receiver.

The priority of a message shall not be changed.

5.3 Text element

5.3.1 General

The `text` element shall contain one of each of the following elements:

- language
- title
- body

Language variants of the same message can be provided by including text elements with different language settings. Variants of the body in the same language but with different levels of detail may be provided by defining message filters; care should be taken to ensure that only one text element in each language is provided for each filter output.

5.3.2 Formatting

Some W3C HTML tags can be used to provide formatting for the title and body.

Style, font and font size tags shall not be used. No font and default font size are defined for the text display due to screen resolution.

The text formatting tags defined in table 1 can be used in both the title and body elements.

Table 1: W3C text formatting tags

Tag	Description
<small>, <strong>	Relative size
<b>	Bold text
<em>	Emphasis text
<i>	Italic text
<sub>	Subscripted text
<sup>	Superscripted text
<ins>	Inserted text
<del>	Deleted text
<mark>	Highlighted text

The paragraph formatting tags defined in table 2 can be used in the body element:

Table 2: W3C paragraph formatting tags

Tag	Description
<h1>, <h2>, <h3>, <h4>, <h5>, <h6>	Different heading format from h1, the most important, to h6, the least important.
<p>	Paragraph.
<hr>	Thematic change.

	Line break.
<a href=…>	Html link with no style. The link address is specified in the href attribute.
<ul>, <li>	List tags.

Receivers which do not actively manage HTML tags shall ignore them (i.e. they shall not be rendered as text).

5.3.3 Language element

The `language` child element defines the language for the other child elements of the parent element, that is the `title` and `body` of a `text` element, or the `text` of an `opening` element (see clause 5.6.3).

Its attributes are detailed below:

Attribute	Description	Type	Status
`xml:lang`	Language of the child element(s)	xml:lang	Required
`default`	Parent element to display if *display language* is not available and *second language* is defined	*xs:boolean*	Optional, default="false"
`mandatory`	Parent element to display if *display language* is not available	*xs:boolean*	Optional, default="false"

A `message` can be sent with textual content in several languages; for each language a parent element is created with child element(s) provided in that language.

The receiver configuration will include a *display language*, and may allow messages to be displayed in the *default language* set by the FIS service provider (typically these will be user settings).

The parent element corresponding to the *display language* shall be displayed if it exists.

In the case of messages with the priority set to **critical** or **important** (see clause 5.2), one parent element shall have the `mandatory` attribute set to `true`; in this case if no parent element is available in the *display language*, the message shall be displayed in the `mandatory` language.

It is also possible for messages with the priority set to **major**, **normal**, **minor**, or **low** to have one parent element with the `default` attribute set to `true`; in this case if no parent element is available in the *display language*, then the message shall be displayed in the `default` language if this is allowed by the receiver configuration.

If no parent element is available in the *display language*, and no parent element has the `mandatory` or `default` attributes set to "`true`", then the `message` shall not be displayed.

5.3.4 Title element

This parameter provides the title of the message. It has a maximum length of 32 characters.

5.3.5 Body element

This parameter provides the body of the message.

When the receiver population targeted by the service provider has a range of display screen sizes, it is recommended to use filters to provide different text lengths according to the screen capability.

EXAMPLE: A company has 3 types of screens:

- display can accommodate a maximum of 80 characters by scrolling text on a single line;
- display has three lines with a maximum of 40 characters;
- display has a high resolution with no limitation for the number of characters.

The company defines a filter for the screen type, configured for each display type. The content is then provided in 3 messages with different text bodies and a filter setting corresponding to the respective display screen.

5.4 Picture element

The inclusion of a `picture` element is optional. The image formats are identical to SlideShow, see ETSI TS 101 499 [i.1]. The aspect ratio of the image is 4:3.

Its attributes are detailed below:

Attribute	**Description**	**Type**	**Status**
`url`	MOT Content Name	*xs:string*	Required
`mimeValue`	PNG or JPG (see IETF RFC 2045 [6])	MIME type	Required
`width`	Width, in pixels, of image multimedia content.	*xs:positiveInteger*	Required
`height`	Height, in pixels, of image multimedia content.	*xs:positiveInteger*	Required

5.5 Validity element

The `validity` element describes the time period in which the message may be displayed, expressed as the calendar date based on the ISO 8601 [4] extended format: YYYY-MM-DD where "YYYY" is the year, "MM" the month and "DD" the day.

EXAMPLE: 2019-05-27.

Its attributes are detailed below:

Attribute	**Description**	**Type**	**Status**
`begin`	Beginning of message validity	xs:date	Optional
`end`	End of message validity	xs:date	Required

If the `message` should be immediately available, then the `begin` date does not need to be specified.

The `end` date shall be equal to or later than the `begin` date (whether specified or not), but no more than one year later.

The `validity` element may be updated to extend or curtail the period in which the message is valid for display. By setting the `end` date to a value in the past, the `message` shall be deleted by receivers. It is recommended that messages for deletion continue to be transmitted after the `end` date to ensure proper deletion in all receivers.

5.6 Geolocation

5.6.1 Introduction

The `geolocation` element allows messages to be localized and only displayed if the receiver is located within the specified localization.

The `geolocation` element shall contain one or more of the following child elements:

- country
- poi

- polygon

Its attributes are detailed below:

Attribute	Description	Type	Status
`xml:id`	A unique label to identify a geolocation definition	*xs:ID*	Optional
`ref`	A unique label (previously defined) to identify a geolocation definition	*xs:IDREF*	Optional

The `xml:id` attribute allows a geolocation definition to be provided once within the document and referenced elsewhere within the document by using the `ref` attribute.

The following restrictions apply:

- A `geolocation` element containing a cross-reference to another shall not define any child elements.
- If a `geolocation` element makes a non-existent reference, then the element shall be ignored.
- A `geolocation` element shall only make a cross-reference to another `geolocation` element within the same document.

If the receiver does not know its current position (either from a navigation aid like GPS or from a default or user configuration) then localization is not taken account in determining whether the message is displayed.

5.6.2 Country element

The `country` element specifies the country using the ISO 3166-1 alpha-2 country codes [7].

5.6.3 Point of Interest (poi) element

The localization zone is determined from the point specified and the distance set in the receiver configuration (either by default or by the user).

The `poi` element shall contain one point element and may contain one or more address, opening and uri elements:

Element	Description	Type	Status
`point`	Based on the georss:point type [8], this specifies one point by latitude and longitude in the WGS84 coordinate reference system, in the format: <latitude> <longitude>	*georss:doubleList*	Required
`address`	Physical address of the point of interest	*xs:string*	Optional
`opening`	Opening schedule	see below	Optional
`uri`	Information containing telephone numbers, fax numbers, emails, internet links, etc.	*xs:anyURI*	Optional

When the `opening` element is included, it should be provided in the same range of languages as the `text` element (see clause 5.3). Each `opening` element shall contain one language element and one text element:

Element	Description	Type	Status
`language`	Language of the text description	see clause 5.3.3	Required
`text`	Description of the opening schedule	*xs:string*	Required

5.6.4 Polygon element

The `polygon` element, based on the georss:polygon [8] type, specifies a space-separated series of points by latitude and longitude in the WGS84 coordinate reference system, forming an enclosed area, in the format:

<[<latitude> <longitude>]...>

The first pair and last pair shall be identical. A minimum of three pairs shall be given.

5.7 Filters element

5.7.1 Introduction

Language, geolocation and validity filters are integral to the operation of the FIS, but company defined filters are also provided to give virtually unlimited scope for directing messages to specific groups or even individual receivers.

The filters are defined using a filter definition file (see clause 7) and evaluated in the receiver against the receiver configuration file (see clause 8). The receiver will only display the subset of messages that correspond to the defined filter types, operations and values.

One or more `filters` elements may be defined for a message. Each `filters` element is evaluated and if the result is true, the message will be displayed. Therefore the logic applied is to OR the result of each `filters` evaluation.

The `filters` element shall contain one or more of the following child elements:

- filterEnum
- filterInt
- filterFloat
- filterDate

When more than one child element is declared, the logical AND of the results of each child element defines the result of the `filters` element.

EXAMPLE: The code fragment defines a filter that is true for a vehicle with a TYPE 1 gearbox AND registered during 2015 or 2016.

```
        <filters>
            <filterEnum filterName='gearBox'> <values><value>TYPE1</value></values></filterEnum>
            <filterDate filterName='regDate'> <comparisons gte="2015-01-01" lte="2016-12-31" />
</filterDate>
       </filters>
```

5.7.2 filterEnum element

A filter of type `filterEnum` can have either one `values` child element or one `ignores` child element.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter which has been defined in the filter definition file	*xs:token*	Required

A `values` child element conatins one or more `value` child elements, of type *xs:enumeration*, that if matched in the receiver configuration file will return a `true` value for the filter. An `ignores` child element contains one or more `ignore` child elements, of type *xs:enumeration*, that if unmatched in the receiver configuration file shall return a `true` value for the filter. In order to minimize the size of the transmitted XML file, the choice of using the `values` or `ignores` child element should be based on the operation that requires the smaller number of enumerated elements to be sent.

5.7.3 filterInt element

A filter of type `filterInt` can have either one `values` child element or one `ignores` child element or one `comparisons` child element.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter which has been defined in the filter definition file	*xs:token*	Required

A `values` child element conatins one or more `value` child elements, of type *xs:int*, that if matched in the receiver configuration file will return a `true` value for the filter. An `ignores` child element contains one or more `ignore` child elements, of type *xs:int*, that if unmatched in the receiver configuration file shall return a `true` value for the filter. In order to minimize the size of the transmitted XML file, the choice of using the `values` or `ignores` child element should be based on the operation that requires the smaller number of enumerated elements to be sent.

The `comparisons` child element attributes are detailed below:

Attribute	Description	Type	Status
`lt`	Available values are less than this defined value for the filter (operation <)	*xs:int*	Optional
`lte`	Available values are less than or equal to this defined value for the filter (operation ≤)	*xs:int*	Optional
`gt`	Available values are greater than this defined value for the filter (operation >)	*xs:int*	Optional
`gte`	Available values are less than or equal to this defined value for the filter (operation ≥)	*xs:int*	Optional

If all comparisons are matched by the value in the receiver configuration file, a `true` value shall be returned for the filter.

5.7.4 filterFloat element

A filter of type `filterFloat` can have either one `values` child element or one `ignores` child element or one `comparisons` child element.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter which has been defined in the filter definition file	*xs:token*	Required

A `values` child element conatins one or more `value` child elements, of type *xs:float*, that if matched in the receiver configuration file will return a `true` value for the filter. An `ignores` child element contains one or more `ignore` child elements, of type *xs:float*, that if unmatched in the receiver configuration file shall return a `true` value for the filter. In order to minimize the size of the transmitted XML file, the choice of using the `values` or `ignores` child element should be based on the operation that requires the smaller number of enumerated elements to be sent.

The `comparisons` child element attributes are detailed below:

Attribute	Description	Type	Status
`lt`	Available values are less than this defined value for the filter (operation <)	*xs:float*	Optional
`lte`	Available values are less than or equal to this defined value for the filter (operation ≤)	*xs:float*	Optional
`gt`	Available values are greater than this defined value for the filter (operation >)	*xs:float*	Optional
`gte`	Available values are less than or equal to this defined value for the filter (operation ≥)	*xs:float*	Optional

If all comparisons are matched by the value in the receiver configuration file, a `true` value shall be returned for the filter.

5.7.5 filterDate element

A filter of type `filterDate` can have either one `values` child element or one `ignores` child element or one `comparisons` child element.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter which has been defined in the filter definition file	*xs:token*	Required

A `values` child element conatins one or more `value` child elements, of type *xs:date*, that if matched in the receiver configuration file will return a `true` value for the filter. An `ignores` child element contains one or more `ignore` child elements, of type *xs:date*, that if unmatched in the receiver configuration file shall return a `true` value for the filter. In order to minimize the size of the transmitted XML file, the choice of using the `values` or `ignores` child element should be based on the operation that requires the smaller number of enumerated elements to be sent.

The `comparisons` child element attributes are detailed below:

Attribute	Description	Type	Status
`lt`	Available values are less than this defined value for the filter (operation <)	*xs:date*	Optional
`lte`	Available values are less than or equal to this defined value for the filter (operation ≤)	*xs:date*	Optional
`gt`	Available values are greater than this defined value for the filter (operation >)	*xs:date*	Optional
`gte`	Available values are less than or equal to this defined value for the filter (operation ≥)	*xs:date*	Optional

If all comparisons are matched by the value in the receiver configuration file, a `true` value shall be returned for the filter.

6 Transport mechanism

6.1 Introduction

The FIS is transported using the MOT protocol (see ETSI EN 301 234 [2]) in a packet mode sub-channel with FEC applied.

6.2 MOT Carousel

6.2.1 Basics

MOT in Directory Mode shall be used. The files may be compressed, if so the CompressionType parameter shall be used. The files may be encrypted, if so the CAInfo parameter shall be used. All MOT parameters shall be used as specified in ETSI EN 301 234 [2].

6.2.2 ContentTypes and ContentSubTypes

Only the ContentType/ContentSubType pairs (see ETSI TS 101 756 [3]) shown in table 3 are permitted:

Table 3: Permitted content type and content subtypes

Content type	Content subtype
text	HTML
image	JFIF (JPEG)
	PNG

6.2.3 ContentName

The character set for the ContentName shall be ISO Latin Alphabet No 1, see ETSI TS 101 756 [3]. Permitted characters are further restricted to a subset of this character set as follows: the lowercase Latin letters, the digits, the hyphen, the forward slash and the underscore ("a".."z", "0".."9", "-", "/", "_").

The file name shall start with the company identifier (key) encoded as an 8-digit hexadecimal number followed by the underscore character ("_") to form a unique prefix to allow files coming from different companies to be quickly identified.

6.2.4 Directory extension

The company identifier (key) shall be encoded as a 32-bit unsigned integer in the MOT directory with the extension parameter Id = 10 0000.

6.3 Signalling

6.3.1 Application type, FIG 0/13

The use of the FIS application within a DAB data channel shall be indicated by the use of FIG 0/13 with a UserApplicationType value of FIS as defined in ETSI TS 101 756 [3].

No user application specific data is defined.

6.3.2 Date and time, FIG 0/9 and FIG 0/10

The provision of a correctly broadcast reference time using FIG 0/10 and local time offset using FIG 0/9 is a mandatory requirement of this User Application. For further details on these parameters, see ETSI EN 300 401 [1].

7 Filter definition file

7.1 Introduction

To provide a greater level of message selection in the receiver, company specific filters can be defined. These filters are defined using a filter definition file and this file is used for the creation of the broadcast FIS (see clause 5) and the creation of the receiver configuration file (see clause 8).

Each filter definition is named and can be one of the four possible filter types: filterEnum, filterInt, filterFloat, or filterDate. The filter type filterEnum also defines the enumeration.

A filter definition file consists of a set of filter definitions, encoded as an XML file.

7.2 FisFilter element

This is the root element of the filter definition document and can contain one or more of the following elements:

- filterEnumDef
- filterIntDef
- filterFloatDef
- filterDateDef

Its attributes are detailed below:

Attribute	Description	Type	Status
`key`	Identifier for entity (company) providing the information	*xs:unsignedInt*	Required
`version`	Indicator for changed content; shall be incremented by one for every new version of the `fisFilter` element, modulo 65536	*xs:unsignedShort*	Required
`date`	Date of creation in the form YYYY-MM-DD	*xs:date*	Required

7.3 filterEnumDef element

A filter definition of type `filterEnumDef` contains the list of enumerated values, minimum 2, for the filter using the `enumDef` element.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter	*xs:token*	Required

Each `enumDef` element provides one enumerated value of type *xs:token*.

Additional company defined attributes may also be added without disturbing the generic FIS decoder.

7.4 filterIntDef, filterFloatDef, filterDateDef elements

A filter definition of type `filterIntDef, filterFloatDef` or `filerDateDef` has no child elements.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter	*xs:token*	Required

For each filter type, additional company defined attributes may also be added without disturbing the generic FIS decoder.

8 Receiver configuration file

8.1 Introduction

Different receivers have different capabilities and are fitted into different environments, for example, different makes and models of cars. To provide the specific data needed to evaluate the company defined filters, a receiver configuration file is used.

Each defined filter from the company's filter definition file, which can be one of the four possible filter types: filterEnum, filterInt, filterFloat, or filterDate, has its value assigned.

A receiver configuration file, encoded as an XML file, contains a value for the corresponding filter in the company filter definition file. The file has a version number and creation date. Each filter may have an optional expiration date.

Some company defined filters may be based on dynamic information, for example the odometer reading giving the distance the car has travelled since new. The value of these filters cannot be provided in the receiver configuration file, but shall be provided by an interface to the FIS decoder.

8.2 FisConf element

This is the root element of the receiver configuration document and can contain one or more of the following elements:

- filterEnumConf
- filterIntConf
- filterFloatConf
- filterDateConf

Its attributes are detailed below:

Attribute	Description	Type	Status
`key`	Identifier for entity (company) providing the information	*xs:unsignedInt*	Required
`version`	Indicator for changed content; shall be incremented by one for every new version of the `fisConf` element modulo 65536	*xs:unsignedShort*	Required
`date`	Date of creation in the form YYYY-MM-DD	*xs:date*	Required

8.3 filterEnumConf element

A filter definition of type `filterEnumConf` contains the enumerated value for this instance of the filter.

The enumerated value is of type *xs:token*.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter	*xs:token*	Required
`expiration`	Validity of the configuration value	*xs:date*	Optional

Additional company defined attributes may also be added without disturbing the generic FIS decoder.

8.4 filterIntConf element

A filter definition of type `filterIntConf` contains the enumerated value for this instance of the filter.

The enumerated value is of type *xs:Int*.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter	*xs:token*	Required
`expiration`	Validity of the configuration value	*xs:date*	Optional

Additional company defined attributes may also be added without disturbing the generic FIS decoder.

8.5 filterFloatConf element

A filter definition of type `filterFloatConf` contains the enumerated value for this instance of the filter.

The enumerated value is of type *xs:float*.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter	*xs:token*	Required
`expiration`	Validity of the configuration value	*xs:date*	Optional

Additional company defined attributes may also be added without disturbing the generic FIS decoder.

8.6 filterDateConf element

A filter definition of type `filterDateConf` contains the enumerated value for this instance of the filter.

The enumerated value is of type *xs:date*.

Its attributes are detailed below:

Attribute	Description	Type	Status
`filterName`	Name of the filter	*xs:token*	Required
`expiration`	Validity of the configuration value	*xs:date*	Optional

Additional company defined attributes may also be added without disturbing the generic FIS decoder.

8.7 Updating the receiver configuration file

The receiver configuration file can be updated to allow receiver makers or third-party companies to add or remove filters.

The update protocol implementation is chosen by the receiver maker. In the case of a third-party company, formal agreement should be found between the third-party and the receiver maker.

9 Receiver behaviour

9.1 Introduction

A receiver equipped with the FIS application shall only decode the FIS corresponding to its authorized company identifier (key). The FIS is identified in several ways: from the MOT directory extension parameter 10 0000; from the ContentName of the files in the MOT Carousel; from the `key` attribute of the `fis` element.

By using the receiver settings, the current receiver configuration file and any dynamic inputs, the FIS application will display only those messages that match the selection criteria. The FIS application shall store all received messages so that it can display those appropriate to the dynamically changing inputs, such as location, user selected language, odometer reading, date, etc.

9.2 Minumum functional requirements

The receiver shall be able to store one or more receiver configuration files. Without a configuration file, the receiver will not be able to display any FIS.

The FIS decoder shall only use files with a company identifier (key) corresponding to the key of the receiver configuration file(s).

The receiver shall manage a minimum of 20 messages in the FIS application database. When the database becomes full, stored messages with the lowest priority shall be deleted to make space for messages of higher priority. At the same priority level, the oldest messages shall be deleted to make space for newer messages.

The receiver shall be able to display the `text/title` of messages in full (32 characters); by default the `title` should be displayed in bold. It is recommended to be able to display a `text/body` of at least 128 characters, with scrolling if necessary.

The display of `picture` elements is optional. Images do not have to be displayed on the same screen as the text. When images are displayed, a minimum size of 160x120 pixels is required, but it is recommended to provide 320x240 pixels or higher. The image may be scaled from 50 % to 150 % to optimize the available display space.

9.3 User controls

It is recommended that the Carsion application sets its *display language* from the receiver HMI. The user could also be provided with the option to display messages in the *default language*.

It is recommended to permit the user to activate or deactivate messages with `priority` from 3 `(normal)` to 5 `(low)`.

The user could be provided with options regarding geolocation, for example, current location from a navigation system, user defined location, distance from a point-of-interest, etc. It is recommended that receivers without access to navigation should allow user defined areas.

9.4 Message display rules

9.4.1 Introduction

The minimum content of a message is a `validity` element and a `text` element. Typically, a message will contain multiple `filters` elements and multiple `text` elements in different languages. They may also contain a `picture` element and multiple `geolocation` elements.

Each element of each message (except the `picture` element, if present) shall be evaluated to determine whether the message should be displayed or not. The evaluation of messages shall be cyclical, because dynamically changing inputs will change the decision about whether the message should be displayed.

9.4.2 Validity element evaluation

Messages shall be valid to be displayed. To be valid, the *current date* shall be within the period of validity which is defined as starting on the `begin` date and ending on the `end` date; both the `begin` date and the `end` date are inclusive.

The *current date* shall be obtained from the MJD in the FIG 0/10, taking into account the local time offset provided by FIG 0/9.

When the `begin` date is not specified in the `validity` element, the `begin` date shall be the *current date* for this evealuation.

If the `begin` date is after the *current date* then the message is not valid, but the receiver retains the message for later evaluation.

If the `begin` date is after the `end` date then the message is malformed, and shall be deleted from the receiver.

If the `end` date is after the *current date* date then the message is expired, and shall be deleted from the receiver.

9.4.3 Filters element evaluation

If `filters` elements are present in the `message` element, then they shall be evaluated.

If any of the `filters` element evaluations produces a `true` result, then the message is a valid candidate for display. When multiple filters are present within the `filters` element, the result of each filter shall be evaluated and combined with logical AND to determine the overall result of that `filters` element.

For each filter within the `filters` element, the receiver configuration file with the same `key` as the `fis` element shall be searched for the `filterName`; if it is found, the result of the filter shall be evaluated from the associated value; if it is not found the result shall be `false`. The evaluation of each filter depends on its type and operation, see clause 5.7. If the filter requires a dynamic input (e.g. odometer reading), then the FIS application shall provide access to the correct data.

NOTE: Some filter values in the receiver configuration file may have expiry dates; the search for the filterName needs to manage this correctly.

9.4.4 Geolocation evaluation

If no `geolocation` elements are present in the `message` element, or if no location data is available, then the result of the `geolocation` evaluation is `true`.

If `geolocation` elements are present in the `message` element, and the receiver has access to location data (either static information entered by the user, or dynamic information from e.g. a connected navigation system) then they shall be evaluated:

- if a `country` element exists and the receiver is located within that country then the evaluation result shall be `true`;
- if a `poi` element exists and the receiver is located within the user defined distance then the evaluation result shall be `true`;
- if a `polygon` element exists and the receiver is located within that polygon then the evaluation result shall be `true`;
- else the evaluation result shall be `false`.

9.4.5 Text element evaluation

A message may contain several text elements in different langauges.

The `text` element with the `language` attribute that matches the *display language* shall be marked as a candidate for display.

If no `text` element with the `language` attribute that matches the *display language* exists and the `priority` of the message is **critical** or **important** (see clause 5.2), then the text element with the `mandatory` attribute set to `true` shall be marked as a candidate for display; else the text element with the `default` attribute set to `true` shall be marked as a candidate for display.

NOTE: It is possible that no `text` element is marked as a candidate for display.

9.4.6 Message display

If the `validity` evaluation is `true` AND the `filters` evaluation is `true` AND the `geolocation` evaluation is `true` then the `text` element marked as a candidate for display shall be presented. If a `picture` element exists for this `message` and the receiver is able to display it, then the `picture` element shall be presented. If a `geolocation/poi` element is present then the `address`, `opening/text` (in the *display language*) and `uri` elements should be presented, if present.

NOTE: Nothing will be presented if no `text` element has been marked as a candidate for display.

It is recommended that a mechanism is implemented to highlight the importance of messages with priority 0, 1 or 2 in order to capture the attention of the user quickly.

Annex A (informative): XML file examples

A.1 Transmission file

Example of xml transmission file to display four messages for the company with key = 564732.

```
<?xml version="1.0" encoding="UTF-8"?>
<fis key="564732" version="5" >
    <message identifier="1" >
        <text> <language xml:lang="en" /> <title content="Great cold Operation" /> <body
content="Great Cold Operation. Do not let winter spoil your weekends. Have your battery checked
every year." /> </text>
        <text> <language xml:lang="fr" /> <title content="Opération Grand Froid" /> <body
content="Opération Grand Froid. Ne laissez pas l'hiver vous gâchez le weekend. Faites vérifier votre
Batterie tous les ans." /> </text>
        <text> <language xml:lang="de" /> <title content="Sehr kalte Temperaturen" /> <body
content="Sehr kalte Temperaturen. Lassen Sie Ihr Wochenende nicht vom Winter ruinieren. Überpruefen
Sie Ihre Batterie jedes Jahr." /> </text>
        <picture url="Cold.png" mimeValue="image/png" width="120" height="90" />
        <validity begin="2018-07-12" end="2018-12-31" />
        <filters>
            <filterEnum filterName='engineType'>
<ignores><ignore>"ELECTRIC"</ignore></ignores></filterEnum>
       </filters>
    </message>

    <message identifier="2" priority="2" >
        <text> <language xml:lang="en" /> <title content="Winter tyres" /> <body content="Equip your
vehicle with winter tyres, this offers more grip, traction and safety. Contact one of our sales
points." /> </text>
        <text> <language xml:lang="fr" /> <title content="Pneus hiver" /> <body content="Equipez vos
4 roues en pneus hiver, cela offre plus d'adhérence, de motricité et de sécurité. Contactez un de
nos points de vente." /> </text>
        <text> <language xml:lang="de" /> <title content="Winterreifen" /> <body content="Benutzen
Sie Winterreifen, diese haben mehr Grip, bessere Traktion und bieten mehr Sicherheit. Kontaktieren
Sie unsere Verkaufspartner." /> </text>
        <picture url="Tyres.png" mimeValue="image/png" width="120" height="90" />
        <validity begin="2018-07-12" end="2018-12-31" />
    </message>

    <message identifier="3" priority="4" >
        <text> <language xml:lang="en" mandatory="true" /> <title content="New car" /> <body
content="You are invited to discover our new car." /> </text>
        <text> <language xml:lang="fr" default="true" /> <title content="Nouveau véhicule" /> <body
content="Vous êtes invités à venir découvrir notre nouveau véhicule." /> </text>
        <text> <language xml:lang="de" /> <title content="Neues Fahrzeug" /> <body content="Sie sind
herzlich eingeladen unser neues Auto zu entdecken." /> </text>
        <picture url="New_car.png" mimeValue="image/png" width="120" height="90" />
        <validity begin="2018-12-15" end="2018-12-31" />
        <geolocation>
            <country>FR</country>
            <poi>
                <point> 48.891087 2.3886613 </point>
                <address> "La Grande Halle de la Villette, 75019 Paris" </address>
                <opening> <language xml:lang="en" /> <text>"Every day, 9am to 9pm</text> </opening>
                <opening> <language xml:lang="fr" /> <text>"Ouvert tous les jours de 9h à 21h</text>
</opening>
                <uri> tel:+33(0) 1 23 45 67 89 </uri>
                <uri> mailto:mail@worlddab.org </uri>
                <uri> http:www.worlddab.org </uri>
            </poi>
        </geolocation>
    </message>

    <message identifier="4" priority="1" >
        <text> <language xml:lang="en" /> <title content="Vehicule callback" /> <body content="A
check is needed on your gearbox. Please, contact your garage" /> </text>
        <text> <language xml:lang="fr" /> <title content="Rappel véhicule" /> <body content="Un
contrôle est nécessaire sur votre boite de vitesse, veuillez contacter votre concessionnaire" />
</text>
```

```
        <text> <language xml:lang="de" /> <title content="Fahrzeugrueckruf" /> <body content="Eine
Kontrolle Ihres Getriebes ist notwendig. Bitte nehmen Sie Kontakt mit Ihrer Vertriebsniederlassung
auf" /> </text>
        <picture url="callback.png" mimeValue="image/png" width="120" height="90" />
        <validity begin="2018-12-15" end="2019-12-15" />
        <filters>
            <filterEnum filterName='gearBox'> <values><value>TYPE1</value></values></filterEnum>
            <filterDate filterName='registrationDate'> <comparisons gte="2015-01-01" lte="2016-12-
31" /> </filterDate>
        </filters>
    </message>
</fis>
```

A.2 Filter definition file

Example of xml filter definition file which defines 4 filters (gearbox, registrationDate, engineCapacity, gearNumber):

```
<?xml version="1.0" encoding="UTF-8"?>
<fisFilter key="564732" version="74" date="2018-07-01">
    <filterEnumDef filterName='gearBox'>
        <enumDef> TYPE1 </enumDef>
        <enumDef> TYPE2 </enumDef>
        <enumDef> TYPE3 </enumDef>
        <enumDef> TYPE4 </enumDef>
    </filterEnum>
    <filterDateDef filterName='registrationDate' />
    <filterFloatDef filterName='engineCapacity' />
    <filterIntDef filterName='gearNumber' />
</fisFilter>
```

A.3 Receiver configuration file

Example of xml configuration file which defines the value for 4 filters (gearbox, registrationDate, engineCapacity, gearNumber):

```
<?xml version="1.0" encoding="UTF-8"?>
<fisConf key="564732" version="38" date="2019-01-01">
    <filterEnumConf filterName='gearBox'> TYPE2 </filterEnum>
    <filterDateConf filterName='registrationDate'> 2014-10-25 </filterDate>
    <filterFloatConf filterName='engineCapacity'> 2.2 </filterFloat>
    <filterIntConf filterName='gearNumber'> 5 </filterInt>
</fisConf>
```

Annex B (normative): XSD defintions

B.1 Transmission file

To validate message transmission file, a XSD schema processor should be used.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.worlddab.org/schemas/fis/10"
    xmlns="http://www.worlddab.org/schemas/fis/10"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:georss="http://www.georss.org/georss"
    elementFormDefault="qualified">

    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="http://www.w3.org/2001/xml.xsd" />
    <xs:import namespace="http://www.georss.org/georss"
    schemaLocation="http://www.georss.org/xml/1.1/georss.xsd" />

    <xs:element name="fis">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="message" type="messageType" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="key" type="xs:unsignedInt" use="required" />
            <xs:attribute name="version" type="xs:unsignedShort" use="required" />
            <xs:anyAttribute namespace="##other" processContents="lax" />
        </xs:complexType>
    </xs:element>

    <!-- Definition of messageType -->
    <xs:complexType name="messageType">
        <xs:sequence>
            <xs:element name="text" type="textType" minOccurs="1" maxOccurs="unbounded" />
            <xs:element name="picture" type="pictureType" minOccurs="0" maxOccurs="1" />
            <xs:element name="validity" type="validityType" minOccurs="1" maxOccurs="1"/>
            <xs:element name="geolocation" type="geolocationType" minOccurs="0" maxOccurs="1" />
            <xs:element name="filters" type="filtersType" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="identifier" type="xs:unsignedInt" use="required" />
        <xs:attribute name="priority" type="priorityType" use="optional" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of textType -->
    <xs:complexType name="textType">
        <xs:sequence>
            <xs:element name="language" type="languageType" minOccurs="1" maxOccurs="1" />
            <xs:element name="title" type="titleType" minOccurs="1" maxOccurs="1" />
            <xs:element name="body" type="bodyType " minOccurs="1" maxOccurs="1" />
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of languageType -->
    <xs:complexType name="languageType">
        <xs:attribute ref="xml:lang" use="required" />
        <xs:attribute name="default" type="xs:boolean" default="false" />
        <xs:attribute name="mandatory" type="xs:boolean" default="false" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of titleType -->
    <xs:complexType name="titleType">
        <xs:attribute name="content" type="xs:string" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of bodyType -->
    <xs:complexType name="bodyType">
        <xs:attribute name="content" type="xs:string" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of pictureType -->
    <xs:complexType name="pictureType">
        <xs:attribute name="url" type="xs:string" use="required" />
        <xs:attribute name="mimeValue" type="mimeType" use="required" />
        <xs:attribute name="width" type="xs:positiveInteger" use="required" />
        <xs:attribute name="height" type="xs:positiveInteger" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of mimeType (Multipurpose Internet Mail Extension -->
    <xs:simpleType name="mimeType">
        <xs:restriction base="xs:string">
            <xs:whiteSpace value="collapse" />
            <xs:pattern value="([!-\.0-~]{1,}/[!-\.0-~]{1,})+" />
        </xs:restriction>
    </xs:simpleType>

    <!-- Definition of validityType -->
    <xs:complexType name="validityType">
        <xs:attribute name="begin" type="xs:date" use="optional" />
        <xs:attribute name="end" type="xs:date" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of geolocationType -->
    <xs:complexType name="geolocationType">
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="country" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            <xs:element name="poi" type="poiType" minOccurs="0" maxOccurs="unbounded" />
            <xs:element name="polygon" type="georss:doubleList" minOccurs="0" maxOccurs="unbounded"
/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
        </xs:choice>
        <xs:attribute ref="xml:id" use="optional" />
        <xs:attribute name="ref" type="xs:IDREF" use="optional" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of poiType -->
    <xs:complexType name="poiType">
        <xs:sequence>
            <xs:element name="point" type="georss:doubleList" minOccurs="1" maxOccurs="1" />
            <xs:element name="address" type="xs:string" minOccurs="1" maxOccurs="1" />
            <xs:element name="opening" type="openingType" minOccurs="0" maxOccurs="unbounded" />
            <xs:element name="uri" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of openingType -->
    <xs:complexType name="openingType">
        <xs:sequence>
            <xs:element name="language" type="languageType" minOccurs="1" maxOccurs="1" />
            <xs:element name="text" type="xs:string" minOccurs="1" maxOccurs="1" />
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of filtersType -->
    <xs:complexType name="filtersType">
        <xs:choice minOccurs="1" maxOccurs="unbounded">
            <xs:element name="filterEnum" type="filterEnumType" minOccurs="0" maxOccurs="1" />
            <xs:element name="filterInt" type="filterIntType" minOccurs="0" maxOccurs="1" />
            <xs:element name="filterFloat" type="filterFloatType" minOccurs="0" maxOccurs="1" />
            <xs:element name="filterDate" type="filterDateType" minOccurs="0" maxOccurs="1" />
        </xs:choice>
    </xs:complexType>

    <!-- Definition of filterEnumType -->
    <xs:complexType name="filterEnumType">
        <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="1">
                <xs:element name="values" type="valuesEnumType" minOccurs="0" maxOccurs="1" />
                <xs:element name="ignores" type="ignoresEnumType" minOccurs="0" maxOccurs="1" />
            </xs:choice>
        </xs:sequence>
        <xs:attribute name="filterName" type="xs:token" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of valuesEnumType -->
    <xs:complexType name="valuesEnumType">
        <xs:sequence>
            <xs:element name="value" type="xs:token" minOccurs="1" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- Definition of ignoresEnumType -->
    <xs:complexType name="ignoresEnumType">
        <xs:sequence>
            <xs:element name="ignore" type="xs:token" minOccurs="1" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- Definition of filterIntType -->
    <xs:complexType name="filterIntType">
        <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="1">
                <xs:element name="values" type="valuesIntType" minOccurs="0" maxOccurs="1" />
                <xs:element name="ignores" type="ignoresIntType" minOccurs="0" maxOccurs="1" />
                <xs:element name="comparisons" type="comparisonsIntType" minOccurs="0" maxOccurs="1"
/>
            </xs:choice>
        </xs:sequence>
        <xs:attribute name="filterName" type="xs:token" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of valuesIntType -->
    <xs:complexType name="valuesIntType">
        <xs:sequence>
            <xs:element name="value" type="xs:int" minOccurs="1" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- Definition of ignoresIntType -->
    <xs:complexType name="ignoresIntType">
        <xs:sequence>
            <xs:element name="ignore" type="xs:int" minOccurs="1" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- Definition of comparisonsIntType -->
    <xs:complexType name="comparisonsIntType">
        <xs:attribute name="lt" type="xs:int" use="optional" />
        <xs:attribute name="lte" type="xs:int" use="optional" />
        <xs:attribute name="gt" type="xs:int" use="optional" />
        <xs:attribute name="gte" type="xs:int" use="optional" />
    </xs:complexType>

    <!-- Definition of filterFloatType -->
    <xs:complexType name="filterFloatType">
        <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="1">
                <xs:element name="values" type="valuesFloatType" minOccurs="0" maxOccurs="1" />
                <xs:element name="ignores" type="ignoresFloatType" minOccurs="0" maxOccurs="1" />
                <xs:element name="comparisons" type="comparisonsFloatType" minOccurs="0"
maxOccurs="1" />
            </xs:choice>
        </xs:sequence>
        <xs:attribute name="filterName" type="xs:token" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of valuesFloatType -->
    <xs:complexType name="valuesFloatType">
        <xs:sequence>
            <xs:element name="value" type="xs:float" minOccurs="1" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- Definition of ignoresFloatType -->
    <xs:complexType name="ignoresFloatType">
        <xs:sequence>
```

```
                <xs:element name="ignore" type="xs:float" minOccurs="1" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <!-- Definition of comparisonsFloatType -->
        <xs:complexType name="comparisonsFloatType">
            <xs:attribute name="lt" type="xs:float" use="optional" />
            <xs:attribute name="lte" type="xs:float" use="optional" />
            <xs:attribute name="gt" type="xs:float" use="optional" />
            <xs:attribute name="gte" type="xs:float" use="optional" />
        </xs:complexType>

        <!-- Definition of filterDateType -->
        <xs:complexType name="filterDateType">
            <xs:sequence>
                <xs:choice minOccurs="0" maxOccurs="1">
                    <xs:element name="values" type="valuesDateType" minOccurs="0" maxOccurs="1" />
                    <xs:element name="ignores" type="ignoresDateType" minOccurs="0" maxOccurs="1" />
                    <xs:element name="comparisons" type="comparisonsDateType" minOccurs="0"
maxOccurs="1" />
                </xs:choice>
            </xs:sequence>
            <xs:attribute name="filterName" type="xs:token" use="required" />
            <xs:anyAttribute namespace="##other" processContents="lax" />
        </xs:complexType>

        <!-- Definition of valuesDateType -->
        <xs:complexType name="valuesDateType">
            <xs:sequence>
                <xs:element name="value" type="xs:date" minOccurs="1" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <!-- Definition of ignoresDateType -->
        <xs:complexType name="ignoresDateType">
            <xs:sequence>
                <xs:element name="ignore" type="xs:date" minOccurs="1" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <!-- Definition of comparisonsDateType -->
        <xs:complexType name="comparisonsDateType">
            <xs:attribute name="lt" type="xs:date" use="optional" />
            <xs:attribute name="lte" type="xs:date" use="optional" />
            <xs:attribute name="gt" type="xs:date" use="optional" />
            <xs:attribute name="gte" type="xs:date" use="optional" />
        </xs:complexType>

        <!-- Different values for message priority -->
        <xs:simpleType name="priorityType">
            <xs:restriction base="xs:token">
                <xs:enumeration value="0" /> <xs:enumeration value="critical" />
                <xs:enumeration value="1" /> <xs:enumeration value="important" />
                <xs:enumeration value="2" /> <xs:enumeration value="major" />
                <xs:enumeration value="3" /> <xs:enumeration value="normal" />
                <xs:enumeration value="4" /> <xs:enumeration value="minor" />
                <xs:enumeration value="5" /> <xs:enumeration value="low" />
            </xs:restriction>
        </xs:simpleType>

</xs:schema>
```

B.2 Filter definition file

To validate filter transmission file, a XSD schema processor should be used.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.worlddab.org/schemas/fisFilter/10"
    xmlns="http://www.worlddab.org/schemas/fisFilter/10"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">

    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="http://www.w3.org/2001/xml.xsd" />
    <xs:element name="fisFilter">
        <xs:complexType>
            <xs:sequence>
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                    <xs:element name="filterEnumDef" type="filterEnumDefType" minOccurs="0"
maxOccurs="1" />
                    <xs:element name="filterIntDef" type="filterIntDefType" minOccurs="0"
maxOccurs="1" />
                    <xs:element name="filterFloatDef" type="filterFloatDefType" minOccurs="0"
maxOccurs="1" />
                    <xs:element name="filterDateDef" type="filterDateDefType" minOccurs="0"
maxOccurs="1" />
                </xs:choice>
            </xs:sequence>

            <xs:attribute name="key" type="xs:unsignedInt" use="required" />
            <xs:attribute name="version" type="xs:unsignedShort" use="required" />
            <xs:attribute name="date" type="xs:date" use="required" />
            <xs:anyAttribute namespace="##other" processContents="lax" />
        </xs:complexType>
    </xs:element>

    <!-- Definition of filterEnumDefType -->
    <xs:complexType name="filterEnumDefType">
        <xs:sequence>
            <xs:element name="enumDef" type="xs:token" minOccurs="2" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="filterName" type="xs:token" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of filterIntDefType -->
    <xs:complexType name="filterIntDefType">
        <xs:attribute name="filterName" type="xs:token" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of filterFloatDefType -->
    <xs:complexType name="filterFloatDefType">
        <xs:attribute name="filterName" type="xs:token" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

    <!-- Definition of filterDateDefType -->
    <xs:complexType name="filterDateDefType">
        <xs:attribute name="filterName" type="xs:token" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>

</xs:schema>
```

B.3 Receiver configuration file

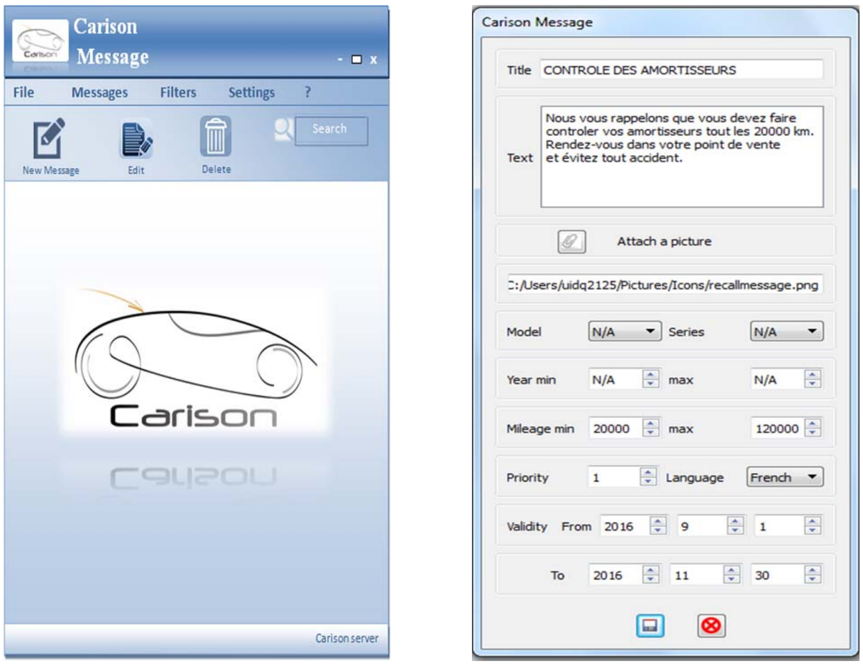
To validate configuration file, a XSD schema processor should be used.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.worlddab.org/schemas/fisConf/10"
    xmlns="http://www.worlddab.org/schemas/fisConf/10"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">

    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="http://www.w3.org/2001/xml.xsd" />

    <xs:element name="fisConf">
        <xs:complexType>
            <xs:sequence>
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                   <xs:element name="filterEnumConf" type="filterEnumConfType" minOccurs="0"
maxOccurs="1" />
                   <xs:element name="filterIntConf" type="filterIntConfType" minOccurs="0"
maxOccurs="1" />
                   <xs:element name="filterFloatConf" type="filterFloatConfType" minOccurs="0"
maxOccurs="1" />
```

```
                    <xs:element name="filterDateConf" type="filterDateConfType" minOccurs="0"
maxOccurs="1" />
                </xs:choice>
            </xs:sequence>

            <xs:attribute name="key" type="xs:unsignedInt" use="required" />
            <xs:attribute name="version" type="xs:unsignedShort" use="required" />
            <xs:attribute name="date" type="xs:date" use="required" />
            <xs:anyAttribute namespace="##other" processContents="lax" />
        </xs:complexType>
    </xs:element>

    <!-- Definition of filterEnumConfType -->
    <xs:complexType name="filterEnumConfType">
        <xs:simpleContent>
            <xs:extension base="xs:token">
                <xs:attribute name="filterName" type="xs:token" use="required" />
                <xs:attribute name="expiration" type="xs:date" use="optional" />
                <xs:anyAttribute namespace="##other" processContents="lax" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <!-- Definition of filterIntConfType -->
    <xs:complexType name="filterIntConfType">
        <xs:simpleContent>
            <xs:extension base="xs:int">
                <xs:attribute name="filterName" type="xs:token" use="required" />
                <xs:attribute name="expiration" type="xs:date" use="optional" />
                <xs:anyAttribute namespace="##other" processContents="lax" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <!-- Definition of filterFloatConfType -->
    <xs:complexType name="filterFloatConfType">
        <xs:simpleContent>
            <xs:extension base="xs:float">
                <xs:attribute name="filterName" type="xs:token" use="required" />
                <xs:attribute name="expiration" type="xs:date" use="optional" />
                <xs:anyAttribute namespace="##other" processContents="lax" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <!-- Definition of filterDateConfType -->
    <xs:complexType name="filterDateConfType">
        <xs:simpleContent>
            <xs:extension base="xs:date">
                <xs:attribute name="filterName" type="xs:token" use="required" />
                <xs:attribute name="expiration" type="xs:date" use="optional" />
                <xs:anyAttribute namespace="##other" processContents="lax" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

</xs:schema>
```

Annex C (informative): Example of transmission

This example demonstrates a generic use case for a product recall. The message data is entered into a FIS creation tool which formats the xml and generates the MOT carousel including related image files. The message creation tool ensures that the company identifier (key) is correctly provided.



Title: CONTROLE DES AMORTISSEURS

Text: Nous vous rappelons que vous devez faire controler vos amortisseurs tout les 20000 km. Rendez-vous dans votre point de vente et évitez tout accident.

:/Users/uidq2125/Pictures/Icons/recallmessage.png

Model	N/A	Series	N/A
Year min	N/A	max	N/A
Mileage min	20000	max	120000
Priority	1	Language	French

Validity	From	2016	9	1
	To	2016	11	30

Figure C.1: Message entry using the Carison™ FIS creation tool

NOTE: Carison is a trade mark owned by Continental Automotive GmbH.

For transmission by the ensemble provider, the following MCI fields are mandatory:

- FIG 0/2 TMId: 11 (MSC packet data);
- FIG 0/3 DSCTy: MOT (see ETSI TS 101 756 [3]);
- FIG 0/13 UATy: FIS (see ETSI TS 101 756 [3]);
- FIG 0/14 FEC Scheme: 01 (FEC scheme applied).

Depending on the number of messages, number and size of images, and desired carousel cycle time, the bit rate given to the FIS should be determined and the sub-channel size determined. More than one FIS may be combined in the same sub-channel, but each FIS will have its own MOT carousel.

For example, the ensemble provider may have two FIS which together have four files:

- AF673B01_carison_Ex1.xml;
- AF673B01_ImageMessageA1.jpg;
- AF673B01_ImageMessageB1.png;
- 00176EA5_carison_Ex2.xml.

The files "AF673B01_carison_Ex1.xml", "AF673B01_ImageMessageA1.jpg" and "AF673B01_ImageMessageB1.png" correspond to the company with key = AF673B01; they are placed in one MOT carousel and the MOT directory extension parameter with ParamId = 10 0000 is set to AF673B01. The file "00176EA5_carison_Ex2.xml" corresponds to the company with key = 00176EA5; it is placed in a second MOT carousel and the MOT directory extension parameter with ParamId = 10 0000 is set to 00176EA5.

History

Document history		
V1.1.1	November 2019	Publication