**3GPP TSG-SA5 Meeting #156 *S5-244583d1***

Maastricht, The Netherlands, 19 - 23 August 2024 Revision of S5-244323

**Source: Ericsson España S.A.**

**Title: DP on Real and Float data types**

**Document for: Endorsement**

**Agenda Item: 5.2**

# 1 Decision/action requested

***The group is asked to discuss and endorse the proposal.***

# 2 References

[1] 3GPP TS 32.156: “Telecommunication management; Fixed Mobile Convergence (FMC) model repertoire”.

[2] ITU-T X.680,"OSI networking and system aspects – Abstract Syntax Notation One (ASN.1)".

[3] 3GPP TS 28.622: “Telecommunication management; Generic Network Resource Model (NRM) Integration Reference Point (IRP); Information Service”

# 3 Rationale

# 3.1 Background

This section provides background information on existing material regarding Float and Real data types.

### 3.1.1 Float vs Real

Float and Real are data types used for representing floating-point numbers. Despite their common objective, these data types exhibit some differences. One of the most noticeable differences is precision (decimal digits). .

On the one hand, Real is a data type that allows storing approximate values with the specified mantissa of the floating-point number (see ITU-T X.680 [2]). The precision of this approximation depends on the number of bits that are used to store the mantissa; the higher the number, the higher the precision.



On the other hand, Float is a data type with the same properties and functionalities as Real, except that precision is limited to 7 decimal digits.



### 3.1.2 Material in existing 3GPP Technical Specifications

TS 32.160 [1] lists the different data types used in FMC repertoire. This list includes **eleven** data types:

* **Two** user-defined data types: <<dataType>> (clause 5.3.4) and ENUM (clause 5.3.5)
* **Nine** predefined data types (clause 5.4.3): **three** UML defined data types (Table 3) and **six** non-UML defined data types (Table 4).

As seen, TS 32.160 [1] only Real is listed as valid data type .

Table 1: UML defined data types (see table 5.4.3.1-1 from TS 32.160 [1])

|  |  |
| --- | --- |
| Name | Description and reference |
| Boolean | See Boolean type of ITU-T X.680 [2] |
| Integer | See Integer type of ITU-T X.680 [2] |
| String | See PrintableString type of ITU-T X.680 [2] |

Table 2: Non-UML defined data types (see table 5.4.3.1-2 from TS 32.160 [1])

|  |  |
| --- | --- |
| Name | Description and reference |
| AttributeValuePair | This data type defines an attribute name and the attribute’s value. |
| BitString | This data type is defined by Bit string of subclause 3 and subclause G.2.5 of ITU-T X.680 [2]. |
| DateTime | This data type defines Date/Time Format, and it is protocol specific. |
| DN | This data type defines the DN (see Distinguished Name of TS 32.300) of an object. It contains a sequence of one or more name components. The “initial sub-sequence” (note 1) of a DN is also a DN of an object.  Note 1: Suppose an object’s DN is composed of a sequence of 4 name components, i.e. 1st, 2nd, 3rd and 4th components. The “initial sub-sequence” of this DN is composed of the 1st, 2nd and 3rd components. |
| External | This data type is defined by another organization. |
| Real | This data type is defined by Real type of ITU-T X.680 [2] |

In Rel-19 version of TS 28.622 [3], the clause 5.2 was introduced. This clause, entitled “Simple Data Types”, defines **nineteen** data types. These data types (from now referred to as add-on data types) represent specializations of the data types defined in FMC model repertoire (from now, referred to as baseline data types). Table 3 captures this specialization; the first column specifies the add-on data type, and the second column the baseline data type.

Table 3: Simple data types (see Table 5.2-1 from 3GPP TS 28.622 [3])

|  |  |  |
| --- | --- | --- |
| Type Name | Type Definition | Description |
| FullTime | String | String with format "full-time" as defined in RFC 3339 [54] |
| DateMonth | String | String with format "date-month" as defined in RFC 3339 [54] |
| DateMonthDay | String | String with format "date-mday" as defined in RFC 3339 [54] |
| Float | Real | The type is Real with format "float" as defined in OpenAPI Specification [63]  Editor Note: format for YANG may need further study |
| Latitude | Real | The type is Real, the range is [-90, 90] |
| Longitude | Real | The type is Real, the range is [-180, 180] |
| DnList | array(DN) | List of DN |
| Mcc | String | Mobile Country Code, see clause 2.3 of TS 23.003 [5] for MCC,, String with pattern: '^[0-9]{3}$'  Editor Note: Pattern may need further study, e.g. alternatie pattern as '^ [02-79][0-9][0-9] $' |
| Mnc | String | Mobile Network Code, see clause 2.3 of TS 23.003 [5] for MNC, String with pattern: '^[0-9]{2,3}$' |
| Nid | String | This represents the Network Identifier, which together with a PLMN ID is used to identify an SNPN (see 3GPP TS 23.003 [5] and 3GPP TS 23.501 [8] clause 5.30.2.1).  Pattern: '^[A-Fa-f0-9]{11}$' |
| Tac | String | 2 or 3-octet string identifying a tracking area code as specified in clause 9.3.3.10 of 3GPP TS 38.413 [34], in hexadecimal representation. Each character in the string shall take a value of "0" to "9", "a" to "f" or "A" to "F" and shall represent 4 bits. The most significant character representing the 4 most significant bits of the TAC shall appear first in the string, and the character representing the 4 least significant bit of the TAC shall appear last in the string.  pattern: '(^[A-Fa-f0-9]{4}$)|(^[A-Fa-f0-9]{6}$)'  Examples:  A legacy TAC 0x4305 shall be encoded as "4305".  An extended TAC 0x63F84B shall be encoded as "63F84B"  Editor Note: Format may need further study |
| UtraCellId | Integer | UTRAN cells identified by UTRAN CGI  Editor Note: to add the limit number |
| EutraCellId | String | 28-bit string identifying an E-UTRA Cell Id as specified in clause 9.3.1.9 of 3GPP TS 38.413 [34], in hexadecimal representation. Each character in the string shall take a value of "0" to "9", "a" to "f" or "A" to "F" and shall represent 4 bits. The most significant character representing the 4 most significant bits of the Cell Id shall appear first in the string, and the character representing the 4 least significant bit of the Cell Id shall appear last in the string.  Pattern: '^[A-Fa-f0-9]{7}$'  Example:  An E-UTRA Cell Id 0x5BD6007 shall be encoded as "5BD6007". |
| NrCellId | String | 36-bit string identifying an NR Cell Id as specified in clause 9.3.1.7 of 3GPP TS 38.413 [34], in hexadecimal representation. Each character in the string shall take a value of "0" to "9", "a" to "f" or "A" to "F" and shall represent 4 bits. The most significant character representing the 4 most significant bits of the Cell Id shall appear first in the string, and the character representing the 4 least significant bit of the Cell Id shall appear last in the string.  Pattern: '^[A-Fa-f0-9]{9}$'  Example:  An NR Cell Id 0x225BD6007 shall be encoded as "225BD6007". |
| Fqdn | String | Fully Qualifed Domain Name, refere to clause 19.4.2 of TS 23.003[5]  Pattern: '^([0-9A-Za-z]([-0-9A-Za-z]{0,61}[0-9A-Za-z])?\.)+[A-Za-z]{2,63}\.?$'  minLength: 4  maxLength: 253 |
| Ipv4Addr | String | String identifying a IPv4 address formatted in the "dotted decimal" notation as defined in IETF RFC 1166 [60].  Pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$'  example: '198.51.100.1' |
| Ipv6Addr | String | String identifying an IPv6 address formatted according to clause 4 of IETF RFC 5952 [61]. The mixed IPv4 IPv6 notation according to clause 5 of IETF RFC 5952 [61] shall not be used.  Pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))$'  and  Pattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))$'  example: '2001:db8:85a3::8a2e:370:7334' |
| Ipv6Prefix | String | String identifying an IPv6 address prefix formatted according to clause 4 of IETF RFC 5952 [61]. IPv6Prefix data type may contain an individual /128 IPv6 address.  Pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))(\/(([0-9])|([0-9]{2})|(1[0-1][0-9])|(12[0-8])))$'  and  Pattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))(\/.+)$'  example: '2001:db8:abcd:12::0/64' |
| Uri | String | String providing an URI formatted according to IETF RFC 3986 [62]. |
| NOTE 1: The string Pattern in X.2-1 may have different variants with no “^” or “$” in the pattern string. | | |

As seen, Float data type is listed in the fourth row.

The **nineteen** data types defined in TS 28.622 [3] (which include Float) aim to complement the **eleven** data types defined in TS 32.160 [1] (which include Real), resulting in a total of **thirty** data types that can reused/imported across the different stage-2 Technical specifications, including (but not limited to): TS 28.622 (Generic NRM), TS 28.104 (MDA), TS 28.105 (AI/ML), TS 28.541 (5G NRM), etc.

### 3.2 Observations

Related to clause 3.1.1.

* **Observation #1:** Real and Float are two data types used for representing floating-point numbers. Their main difference is the achievable precision. While float allows representing numbers with precision up to 7 decimal digits, Real has no such limitation (i.e., higher precision can be achieved with higher number of bits to store the mantissa).

Related to clause 3.1.2.

* **Observation #2**: Table 5 associates Float (“type name” column) with Real (“type definition” column). This association, without further clarification, is confusing; it can be interpreted that Float and Real are interchangeable, which is not correct. It should be noted that the “type name” column lists add-on data types (introduced in TS 28.622 [3]), while “type definition” column lists baseline data types (introduced in TS 32.156 [2]).
* **Observation #3**: 3GPP SA5 specifications contain attributes using both data types; however, the main difference of these data types (see Table 2) is not reported in any 3GPP specification. This may have led to situations where the attribute is set to Float or Real with no criteria, i.e. without awareness on the impact that each option offers. For example, there may have been situations where 5-digit precision attributes are set to Real, when the Float type would have suffice.

### 3.3 Problem statement

The need of using/keeping both Real and Float to represent OAM and charging data has never been discussed nor agreed in 3GPP SA5. It is recommended to document the features of the selected data type(s), according to Table 2, so readers/developers become aware of these features when using the data type(s).

## 3.4 Potential solution

This section details the potential solution to address the problem statement in clause 3.3.











This solution applies for outcome 2, i.e. 3GPP SA5 agrees that both Real and Float data types are needed. Embracing this solution requires making changesin both 3GPP TS 32.156 [1] and 3GPP TS 28.622 [3]. The applicable changes are as follows:

* In Table 5.2-1 from 3GPP TS 28.622 [3], remove the row that describes Float data type.
* In Table 5.4.3.1-2 from TS 32.156 [1], add a new data type called Float.
* In the rows now used in Table 5.4.3.1-2 for Float and Real data types, use the “description and references” column to emphasize on their noteworthy differences, according to Table 2.

Table 5.2-1 in 3GPP TS 28.622 [3] would look like as pictured below.

|  |  |  |
| --- | --- | --- |
| Type Name | Type Definition | Description |
| FullTime | String | String with format "full-time" as defined in RFC 3339 [54] |
| DateMonth | String | String with format "date-month" as defined in RFC 3339 [54] |
| DateMonthDay | String | String with format "date-mday" as defined in RFC 3339 [54] |
| Float | Real | The type is Real with format "float" as defined in OpenAPI Specification [63]  Editor Note: format for YANG may need further study |
| Latitude | Real | The type is Real, the range is [-90, 90] |

Table 5.4.3.1-2 in TS 32.156 [1] would look like as pictured below.

|  |  |
| --- | --- |
| Name | Description and reference |
| AttributeValuePair | This data type defines an attribute name and the attribute’s value. |
| BitString | This data type is defined by Bit string of subclause 3 and subclause G.2.5 of ITU-T X.680 [2]. |
| DateTime | This data type defines Date/Time Format, and it is protocol specific. |
| DN | This data type defines the DN (see Distinguished Name of TS 32.300) of an object. It contains a sequence of one or more name components. The “initial sub-sequence” (note 1) of a DN is also a DN of an object.  Note 1: Suppose an object’s DN is composed of a sequence of 4 name components, i.e. 1st, 2nd, 3rd and 4th components. The “initial sub-sequence” of this DN is composed of the 1st, 2nd and 3rd components. |
| External | This data type is defined by another organization. |
| Real | This data type is defined by Real type of ITU-T X.680 [2].  Single-precision floating—point number. It has 4-byte size and 7-digit precision. In IEEE 754, it corresponds to binary32.  See NOTE 1. |
| Float | Double-precision floating—point number. It has 8-byte size and 15-digit precision. In IEEE 754, it corresponds to binary64.  See NOTE 1. |
| NOTE 1: <describe here the proof/need of keeping both data types>. | |

This solution proposes to update clause 5.2 in 3GPP TS 28.622 [3], by making the following changes:

* In the clause 5.2 description, clarify that data types captured in Table 5.2-1 specifies new data types (add-on data types) that aim to complement FMC model repertoire data types (baseline data types). Clarify that add-on data types (“type name” column ) represent specialization of baseline data types (“type definition” column).
* In Table 5.2.1, for the float and real data type, use “description” column to emphasize on their noteworthy difference, as marked in yellow below.

|  |  |  |
| --- | --- | --- |
| Type Name | Type Definition | Description |
| Float | Real | Float is a number with format sufficient for precision <=7 decimal digits. Real is a number with format sufficient for precision >7 decimal digits.  The type is Real with format "float" as defined in OpenAPI Specification [63]  Editor Note: format for YANG may need further study |

# 4 Detailed proposal

The group is asked to:

* Agree on the observations reported in clause 3.2.
* Endorse potential solution in clause 3.3.
* Submit a CR to SA5#157 to implement solution endorsed.