|  |
| --- |
| CHANGE REQUEST |
| Meeting ID:\* |  |
| Source:\* |  |
| Date:\* |  |
| Reason for Change/s:\* |  |
| CR against: Release\* |  |
| CR against: WI\* | [x]  Active <#WI#> [ ]  MNT maintenance / < Work Item number(optional)>Is this a mirror CR? Yes [ ]  No [x] mirror CR number: [ ]  STE Small Technical Enhancements / < Work Item number (optional)>Only ONE of the above shall be ticked |
| CR against: TS/TR\* |  |
| Clauses \* |  |
| Type of change: \* | [ ]  Editorial change[x]  Bug Fix or Correction[ ]  Change to existing feature or functionality[ ]  New feature or functionalityOnly ONE of the above shall be ticked |
| Other TS/TR(s) impacted | None |
| Post Freeze checking:\* | This CR contains only essential changes and corrections? YES [x]  NO [ ] This CR may break backwards compatibility with the last approved version of the TS? YES [ ]  NO [ ]  |
| Template Version: January 2019 (do not modify) |

**oneM2M Notice**

The document to which this cover statement is attached is submitted to oneM2M. Participation in, or attendance at, any activity of oneM2M, constitutes acceptance of and agreement to be bound by terms of the Working Procedures and the Partnership Agreement, including the Intellectual Property Rights (IPR) Principles Governing oneM2M Work found in Annex 1 of the Partnership Agreement.

GUIDELINES for Change Requests:

Provide an informative introduction containing the problem(s) being solved, and a summary list of proposals.

Each CR should contain changes related to only one particular issue/problem.

In case of a correction, and the change apply to previous releases, a separate “mirror CR” should be posted at the same time of this CR

Mirror CR: applies only when the text, including clause numbering are exactly the same.

Companion CR: applies when the change means the same but the baselines differ in some way (e.g. clause number).

Follow the principle of completeness, where all changes related to the issue or problem within a deliverable are simultaneously proposed to be made E.g. A change impacting 5 tables should not only include a proposal to change only 3 tables. Includes any changes to references, definitions, and acronyms in the same deliverable.

Follow the drafting rules.

All pictures must be editable.

Check spelling and grammar to the extent practicable.

Use Change bars for modifications.

The change should include the current and surrounding clauses to clearly show where a change is located and to provide technical context of the proposed change. Additions of complete clauses need not show surrounding clauses as long as the proposed clause number clearly shows where the new clause is proposed to be located.

Multiple changes in a single CR shall be clearly separated by horizontal lines with embedded text such as, start of change 1, end of change 1, start of new clause, end of new clause.

When subsequent changes are made to content of a CR, then the accepted version should not show changes over changes. The accepted version of the CR should only show changes relative to the baseline approved text.

## Introduction

Clean-up Annex C

<https://git.onem2m.org/specifications/ts/ts-0023/-/merge_requests/64>

<https://git.onem2m.org/specifications/ts/ts-0023/-/merge_requests/64/diffs?commit_id=d0426e84f7ba07c0a4b90be73fb9386b107126d6>

----------------------- Start of change 1 -----------------------

---a/TS-0023-SDT\_based\_Information\_Model\_and\_Mapping\_for\_Vertical\_Industries.md
+++b/TS-0023-SDT\_based\_Information\_Model\_and\_Mapping\_for\_Vertical\_Industries.md

@@ -6877,13 +6877,14 @@ Table B.2.2-1 captures the equivalency mapping between OCF defined Device Types

# Annex C (informative): Mapping to Content Attribute

## C.1 Introduction

Current SDT models are used only in form of &lt;flexContainer>s, and how to design content attribute of &lt;contentInstance> and &lt;timeSeriesInstance> is left to developers. There is no rule for design of content attribute, it means interoperability of content attribute is low. Then SDT can become one of the rules for design of content attribute, and the low interoperability problem will be solved.

The present clause explains how to use SDT as one of the rules for design of content attribute.

There are several benefits of using SDT in content attribute.

First, the resource architecture can be simpler than the one using &lt;flexContainer>s. When using &lt;flexContainer>s, universal attributes are mapped either into attributes of [deviceInfo] under a &lt;node> besides &lt;flexContainer>s, or into custom attributes of [[dmDeviceInfo]](#583-dmdeviceinfo) under a [[flexNode](#581-flexnode)] (see Rule 1-8 in clause [6.2.2](#622-resource-mapping-for-device-model)). Moreover, Action Class and DataPoint Class are the same layer in SDT, but Action Class is mapped to &lt;flexContainer> itself and DataPoint Class is mapped to attributes of &lt;flexContainer> expressing Module class. On the other hand, Using SDT in content attribute means using only one &lt;contentInstance> or &lt;timeSeriesInstance> so the resource architecture is simple.

First, the resource architecture can be simpler than the one using &lt;flexContainer>s. When using &lt;flexContainer>s, universal attributes are mapped either into attributes of [deviceInfo] under a &lt;node> besides &lt;flexContainer>s, or into custom attributes of [\[dmDeviceInfo\]](#583-dmdeviceinfo) under a [\[flexNode\]](#581-flexnode)] (see Rule 1-8 in clause [6.2.2](#622-resource-mapping-for-device-model)). Moreover, Action Class and DataPoint Class are the same layer in SDT, but Action Class is mapped to &lt;flexContainer> itself and DataPoint Class is mapped to attributes of &lt;flexContainer> expressing Module class. On the other hand, Using SDT in content attribute means using only one &lt;contentInstance> or &lt;timeSeriesInstance> so the resource architecture is simple.

Relating this benefit, it becomes easy to understand where to write information.

@@ -6895,9 +6896,10 @@ In addition, tools can generate validator of the data and converter among the su

## C.2 XML representation of SDT instances.

ModuleClasses, SubDevice models and DeviceClass models written in clause 5 are expressed another way with using each class names as the tag. This clause introduces this way.

Normative work for defining the mapping rules from SDT to XML/ JSON instance are defined by SDT4.0.

Normative work for defining the mapping rules from SDT to XML/JSON instance are defined by SDT4.0.

### C.2.1 Mapping Rules of XML representation

@@ -6966,6 +6968,7 @@ Below is the example for deviceThreeDPrinter:

## C.3 JSON representation of SDT instances

### C.3.1 Mapping Rules of JSON representation

This clause explains how to represent SDT instances inJSON format

Inclusion relationship of SDT instances are directly expressed as inclusion relationship of JSON hash({}). Key name is same as each class name of SDT. Value types are written in various types depending on SDT definition.

@@ -6988,6 +6991,7 @@ When a certain device does not any SubDevice Class, the tags about SubDevices Cl

### C.3.2 Example of JSON representation

Below is the example for deviceThreeDPrinter:

```JSON

@@ -7034,13 +7038,13 @@ Below is the example for deviceThreeDPrinter:

## C.4 How to write into content attribute

Any size of the SDT class cluster may be mapped to content attribute. For example, from only the DataPoint class to DataPoint, Action, Module, SubDevice and Device classes may be mapped to one content attribute at once.

Any size of the SDT class cluster may be mapped to content attribute. For example, from only the DataPoint class to DataPoint, Action, Module, SubDevice and Device classes may be mapped to one content attribute at once.

![Mapping SDT to content attribute](media/Mapping\_SDT\_to\_content\_attribute.png)

![Figure C.4-1: Image of Mapping SDT to content attribute](media/Mapping\_SDT\_to\_content\_attribute.png)

\*\*Figure C.4-1: Image of Mapping SDT to content attribute\*\*

An example that expresses a CREATE request for &lt;contentInstance> serialized into an XML document is shown below. This example is for the Module class and DataPoint class mapping and only content attribute and contentInfo attribute are shown as content parameter (pc).

```XML

@@ -7070,7 +7074,6 @@ In content attribute, a value of DataPoint may be written between tags named the

contentInfo attribute is able to be omitted because content attribute has tags named the certain Module class name (&lt;temp>&lt;/temp>).

An example for only DataPoint class mapping is shown below.:

```XML

@@ -7101,7 +7104,6 @@ If a contentInfo attribute is not used, content attribute may change as follows:

```

# Annex D (informative) Bibliography

- oneM2M TS-0001: "Functional Architecture"

----------------------- End of change 1 -----------------------