|  |  |
| --- | --- |
| CHANGE REQUEST | |
| Meeting ID:\* |  |
| Source:\* |  |
| Date:\* |  |
| Reason for Change/s:\* |  |
| CR against: Release\* |  |
| CR against: WI\* | Active <#WI0100#>  MNT maintenance / < Work Item number(optional)>  Is this a mirror CR? Yes  No  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  Bug Fix or Correction  Change to existing feature or functionality  New feature or functionality  Only 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  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

#wording corrections#

<https://git.onem2m.org/specifications/ts/ts-0041/-/merge_requests/9>

<https://git.onem2m.org/specifications/ts/ts-0041/-/merge_requests/9/diffs?commit_id=c8b1e2a020e58c9e729ecd27382b73208cb77b78>

========= Comment provided by Ingo Friese at 2025-02-11T09:08:30.931Z: =========

1.) check format here:"publish-subscribe functionality for the OGC side, as specified in the MQTT extension of the SensorThings..."

2.) typo: 'acording' to 'according' in 6.0

3.) change 'regarded' in "corresponding"

4.) change 'triggert' to 'triggered'

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

---a/TS-0041-oneM2M-SensorThings\_interworking.md  
+++b/TS-0041-oneM2M-SensorThings\_interworking.md

@@ -61,6 +61,7 @@ References are either specific (identified by date of publication and/or edition

- <a name="\_ref\_1">[1]</a> OGC SensorThings API "Part 1: Sensing Version 1.1" (http://www.opengis.net/doc/is/sensorthings/1.1)

- <a name="\_ref\_2">[2]</a> oneM2M TS-0033 (V3.0.0): "Interworking Framework"

- <a name="\_ref\_3">[3]</a> oneM2M TS-0001 (V4.23.0): "Functional Architecture"

## 2.2 Informative references

@@ -163,11 +164,11 @@ The Sensing Entities data model and the purpose of data within the data model di

# 6 Architecture Model of OGC/STA to oneM2M interworking

## 6.0 Introduction

## 6.0 Overview

Figure 6.0-1 shows an architecture approach for an Interworking Proxy Entity (IPE) between oneM2M and the OGC SensorThings API. The IPE is located between a oneM2M CSE and an OGC/SensorThings API (STA)-Server.

The basic interworking enables applications that are connected to an oneM2M-based system to get data from sensors that are connected to an OGC/STA server. Furthermore, an application that is connected to an OGC/STA server will be able to get data from sensors that are connected to an oneM2M-based system.

The basic interworking enables applications that are connected to an oneM2M-based system to get data from sensors that are connected to an OGC/STA server. Furthermore, an application that is connected to an OGC/STA server will be able to get data from sensors that are connected to an oneM2M-based system. The communication flow of the IPE shall rely on HTTP and MQTT. The MQTT protocol enables publish-subscribe functionality for the OGC side, as specified in the MQTT extension of the SensorThings API <a href="#\_ref\_i.1">[i.1]</a>.

![Figure 6.0-1: IPE architecture overview](media/STA\_oneM2M\_architekturbild\_01.svg)

@@ -179,12 +180,28 @@ According to oneM2M TS-0033 <a href="#\_ref\_2">[2]</a> a representation of a non-

- sensor (IoT-data); and

- administrative data (like historic locations or historic products IDs).

The IPE shall map the 'result' attribute of an OGC/STA 'Observation' to the 'content' attribute of a oneM2M &lt;contentInstance&gt;, and vice versa as shown in Figure 6.1-1.

The IPE shall map the 'result' attribute of an OGC/STA 'Observation' to the 'content' attribute of a oneM2M `<contentInstance>`, and vice versa as shown in Figure 6.1-1. The data type of the 'result' field of an "Observation" is according to SensorThings API <a href="#\_ref\_i.1">[i.1]</a> 'any' and depends on the 'observationType' defined in the associated "Datastream". The 'content' attribute of an oneM2M instance may be stringified data <a href="#\_ref\_3">[3]</a> understandable with the help of the 'contentInfo' attribute. The 'contentInfo' attribute on the oneM2M side may be added by the IPE. The original timestamps, present in the "Observation" as 'phenomenonTime' and in the `<contentInstance>` as "creationTime," shall be discarded. These timestamps are to be reset by the OGC /STA server and the CSE. They may be transmitted for informational purposes as part of the 'result' or the 'content' fields.

![Figure 6.1-1: OGC / STA-to-oneM2M data model mapping](media/data\_mapping.svg)

\*\*Figure 6.1-1: OGC / STA-to-oneM2M data model mapping\*\*

## 6.2 Communication Flow

Figure 6.2-1 shows the oneM2M-to-OGC/STA direction. In order to transfer data from a oneM2M sensor to the OGC/STA server the IPE creates a `<subscription>` to the `<container>` resource in the CSE containing the desired data. Triggered by a sensor event a new `<contentInstance>` is added to the `<container>` by the `<AE>`. The IPE gets a `<notification>` containing the `<contentInstance>` resource.

The IPE constructs an "Observation" creation request and copies the 'content' attribute of the `<contentInstance>` to the 'result' attribute of the "Observation" and sends it to a "Datastream" to be created as detailed in Section 6.3.1 at the OGC/STA server. The OGC/STA applcation gets the sensor data either by polling the OGC/STA server or subscribing to the corresponding "Datastream" at the MQTT broker of the OGC/STA server.

![Figure 6.2-1: Communication oneM2M-to-OGC/STA direction](media/com\_flow\_1.png)

\*\*Figure 6.2-1: Communication oneM2M-to-OGC/STA direction\*\*

Figure 6.2-2 shows the OGC/STA-to-oneM2M direction. The IPE subscribes to the desired "Datastream" of the MQTT-Broker at the OGC/STA server. The OGC/STA server publishes a new "Observation" via the MQTT broker triggered by a OGC/STA sensor. The IPE creates a `<contentInstance>` in a container, to be created as detailed in Section 6.3.2 in the CSE and copies the 'result' attribute of the "Observation" to the 'content' attribute of the `<contentInstance>`. The oneM2M applcation gets the sensor data either by polling the CSE or subscribing to the desired `<container>` at the CSE.

![Figure 6.2-2: Communication OGC/STA-to-oneM2M direction](media/com\_flow\_2.png)

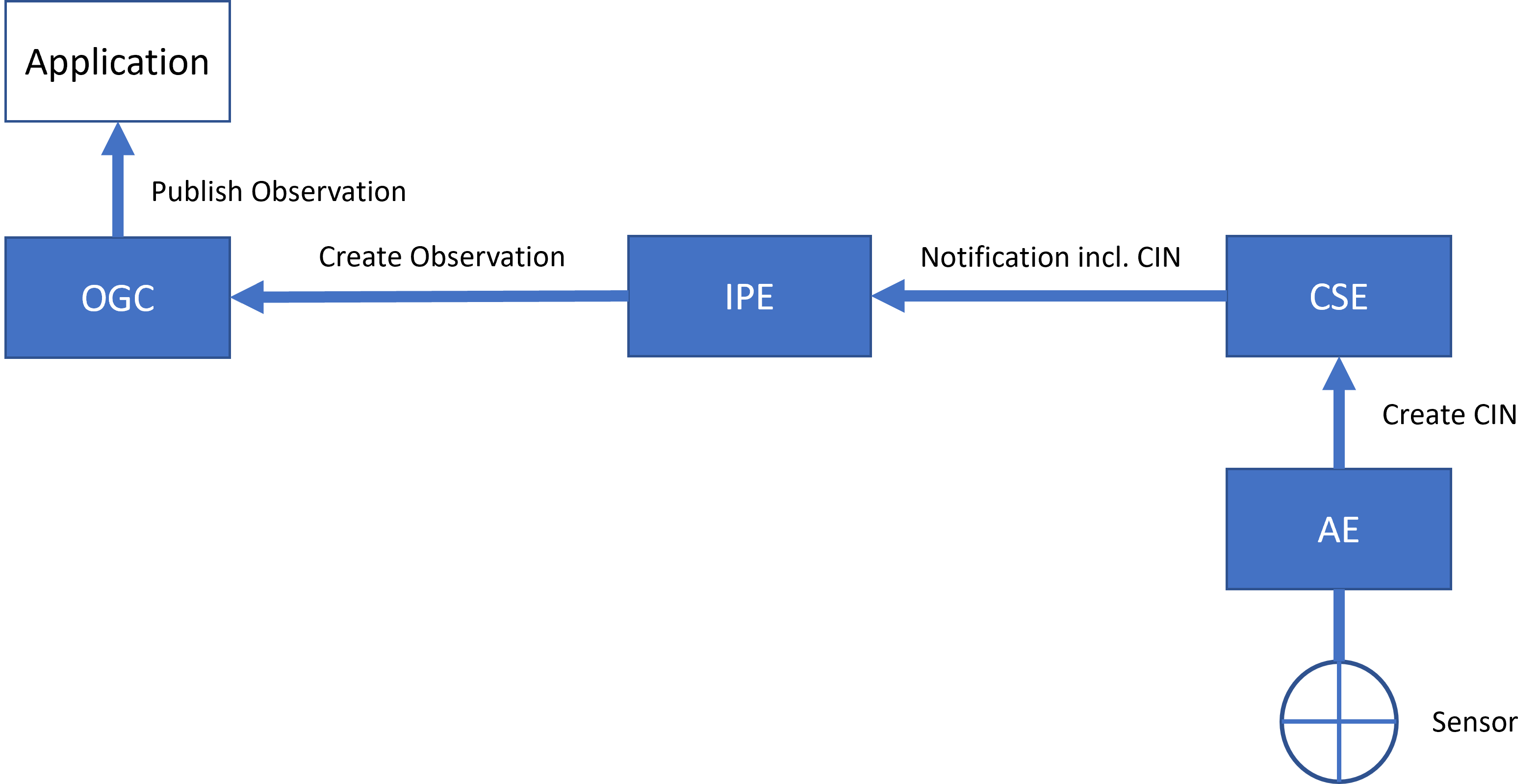
\*\*Figure 6.2-2: Communication OGC/STA-to-oneM2M direction\*\*

<mark>The following text is to be used when appropriate:</mark>

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

----------------------- Start of change 2 -----------------------

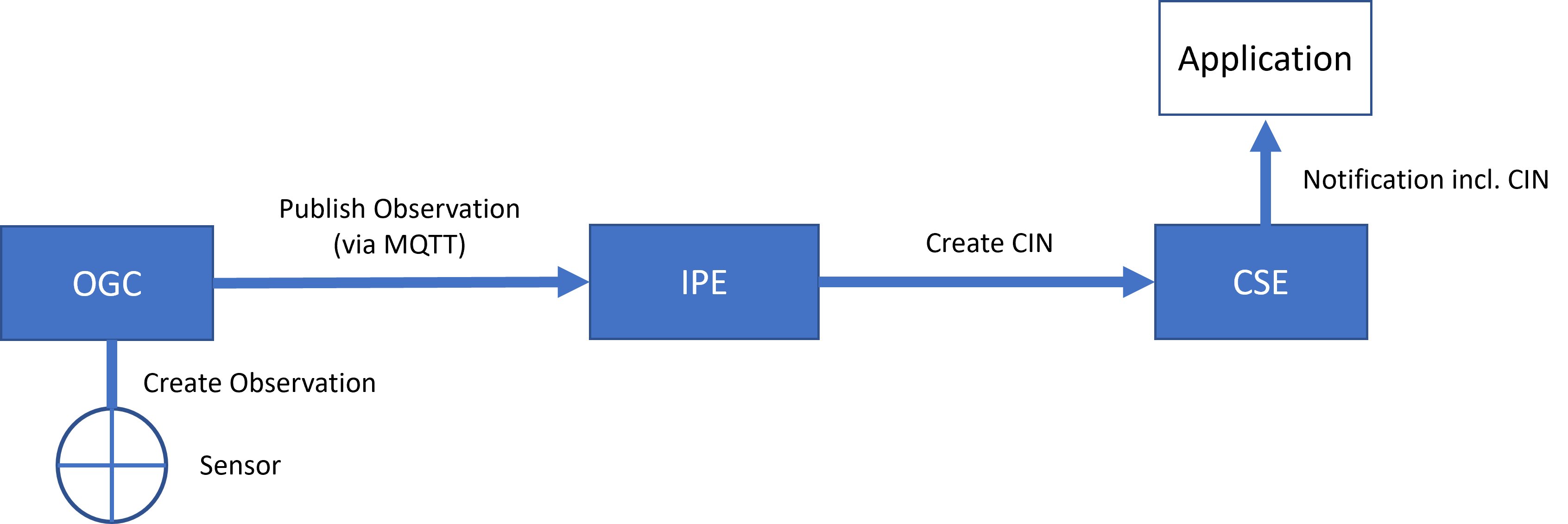
---/dev/null  
+++b/media/com\_flow\_1.png



----------------------- End of change 2 -----------------------

----------------------- Start of change 3 -----------------------

---/dev/null  
+++b/media/com\_flow\_2.png



----------------------- End of change 3 -----------------------

----------------------- Start of change 4 -----------------------

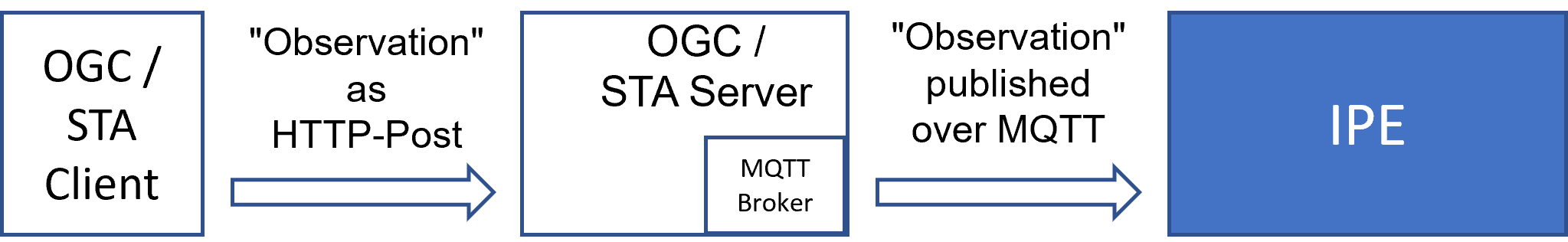
---/dev/null  
+++b/media/config.png



----------------------- End of change 4 -----------------------

----------------------- Start of change 5 -----------------------

---/dev/null  
+++b/media/config\_ogc.png



----------------------- End of change 5 -----------------------