|  |  |
| --- | --- |
| 3GPP TR 23.700-97 V18.1.0 (2023-06) | |
| Technical Report | |
| 3rd Generation Partnership Project;  Technical Specification Group Services and System Aspects;  Study on Application Capability Exposure for IoT Platforms;  (Release 18) | |
|  | |
|  |  |
|  | |
| The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) and may be further elaborated for the purposes of 3GPP. The present document has not been subject to any approval process by the 3GPPOrganizational Partners and shall not be implemented. This Specification is provided for future development work within 3GPPonly. The Organizational Partners accept no liability for any use of this Specification. Specifications and Reports for implementation of the 3GPP TM system should be obtained via the 3GPP Organizational Partners' Publications Offices. | |

|  |
| --- |
|  |
| ***3GPP***  Postal address  3GPP support office address  650 Route des Lucioles - Sophia Antipolis  Valbonne - FRANCE  Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16  Internet  http://www.3gpp.org |
| ***Copyright Notification***  No part may be reproduced except as authorized by written permission. The copyright and the foregoing restriction extend to reproduction in all media.  © 2023, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).  All rights reserved.  UMTS™ is a Trade Mark of ETSI registered for the benefit of its members  3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners LTE™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners  GSM® and the GSM logo are registered and owned by the GSM Association |

Contents

Foreword 5

1 Scope 6

2 References 6

3 Definitions of terms, symbols and abbreviations 7

3.1 Terms 7

3.2 Symbols 7

3.3 Abbreviations 7

4 Key Issues 8

4.1 Key issue #1: Background Data Transfer negotiation 8

4.2 Key issue #2: Application server monitoring and control of traffic 8

4.3 Key issue #3: IoT Platform PSM monitoring and configuration 9

4.4 Key issue #4: Configuration of Communication Patterns 9

4.5 Key issue #5: NIDD configuration 10

4.6 Key issue #6: Device Triggering services. 10

5 Solutions 11

5.1 Solution #1: Application Service Management Service 11

5.1.1 Functional model for application service management 11

5.1.1.1 General 11

5.1.1.2 On-network functional model description 11

5.1.2 Functional entities description 11

5.1.2.1 General 11

5.1.2.2 Application service management client 11

5.1.2.3 Application service management server 12

5.1.3 Configuring VAL server 12

5.1.4 Request-response model 12

5.1.4.1 ASM server requesting for on-demand status 12

5.1.4.2 ASM Client pushes service experience report to the ASM Server 13

5.1.4.3 ASM Server pulls service experience report from the ASM Client 14

5.1.4.4 ASM Server determines corrective action 15

5.1.5 Subscribe-notify model 15

5.1.5.1 ASM server subscribes to the monitoring status information 15

5.1.5.2 VAL server notifies the monitoring status information 16

5.1.6 Reference Points 16

5.1.6.1 General 16

5.1.6.2 ASM-C 16

5.1.6.3 ASM-UU 16

5.1.6.4 ASM-S 16

5.2 Solution #2: IoT Platform deployment options 17

5.2.1 General 17

5.2.2 Deployment models in single PLMN operator domain 17

5.2.2.1 General 17

5.2.2.2 Single network exposure access model 17

5.2.2.3 Distributed network exposure access model 18

5.3 Solution #3: IoT Platform Functional Models 19

5.3.1 General 19

5.3.2 On-network functional models 19

5.4 Solution #4: Device triggering 22

5.4.1 General 22

5.4.2 Procedures and information flows 22

5.4.2.1 Procedure 22

5.4.3 Evaluation 23

5.5 Solution #5: Application Server status monitoring via CAPIF 23

5.5.1 Description 23

5.5.1.1 AS service status monitoring 23

5.5.2 Evaluation 24

5.6 Solution #6: BDT configuration 24

5.6.1 General 24

5.6.2 Procedures and information flows 24

5.6.2.1 General 24

5.6.2.2 Request and Select Background Data Transfer Policy 24

5.6.3 Evaluation 26

5.7 Solution #7: UE unified traffic pattern and monitoring management 27

5.7.1 General 27

5.7.2 Procedures and information flows 27

5.7.2.1 General 27

5.7.2.2 UE unified traffic pattern and monitoring management subscription procedure 27

5.7.2.4 Traffic pattern configuration request procedure 29

5.7.2.5 Information Flows 30

5.7.2.5.1 UE unified traffic pattern and monitoring management subscription request 30

5.7.2.5.2 UE unified traffic pattern and monitoring management subscription response 31

5.7.2.5.3 UE unified traffic pattern update notification 31

5.7.2.5.4 Traffic pattern configuration request 31

5.7.2.5.5 Traffic pattern configuration response 32

5.7.3 Management and 5GC exposure procedures 32

5.7.3.1 General 32

5.7.3.2 CP configuration procedure 32

5.7.3.3 UE unified traffic pattern management procedure 33

5.7.3.4 Network parameter coordination procedure 34

6 Overall Evaluation 35

6.1 Architecture evaluation 35

6.2 Solution evaluations 36

6.2.1 General 36

6.2.2 Key issue #2: Application server monitoring and control of traffic 36

7 Conclusions 37

7.1 Solution conclusions 37

Annex A: Change history 38

# Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

In the present document, modal verbs have the following meanings:

**shall** indicates a mandatory requirement to do something

**shall not** indicates an interdiction (prohibition) to do something

The constructions "shall" and "shall not" are confined to the context of normative provisions, and do not appear in Technical Reports.

The constructions "must" and "must not" are not used as substitutes for "shall" and "shall not". Their use is avoided insofar as possible, and they are not used in a normative context except in a direct citation from an external, referenced, non-3GPP document, or so as to maintain continuity of style when extending or modifying the provisions of such a referenced document.

**should** indicates a recommendation to do something

**should not** indicates a recommendation not to do something

**may** indicates permission to do something

**need not** indicates permission not to do something

The construction "may not" is ambiguous and is not used in normative elements. The unambiguous constructions "might not" or "shall not" are used instead, depending upon the meaning intended.

**can** indicates that something is possible

**cannot** indicates that something is impossible

The constructions "can" and "cannot" are not substitutes for "may" and "need not".

**will** indicates that something is certain or expected to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**will not** indicates that something is certain or expected not to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**might** indicates a likelihood that something will happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

**might not** indicates a likelihood that something will not happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

In addition:

**is** (or any other verb in the indicative mood) indicates a statement of fact

**is not** (or any other negative verb in the indicative mood) indicates a statement of fact

The constructions "is" and "is not" do not indicate requirements.

# 1 Scope

The present document is a technical report which identifies SEAL functionality to support application capability exposure to general purpose servers or 3rd party IoT applications via IoT Platforms. Key issues and corresponding solutions necessary to ensure efficient use and deployment of IoT Platforms in the 5G network are included.

The study takes into consideration the existing work for Core Network exposure in 3GPP TS 23.502 [2] and provides recommendations for normative work.

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 23.502: "Procedures for the 5G System; Stage 2".

[3] 3GPP TR 23.503: "Policy and charging control framework for the 5G System (5GS)".

[4] 3GPP TR 22.101: "Service aspects; Service principles".

[5] 3GPP TS 29.522: "5G System; Network Exposure Function Northbound APIs"

[6] 3GPP TS 23.434: "Service Enabler Architecture Layer for Verticals (SEAL); Functional architecture and information flows"

[7] 3GPP TS 23.682: "Architecture enhancements to facilitate communications with packet data networks and applications"

[8] 3GPP TS 23.222: "Functional architecture and information flows to support Common API Framework for 3GPP Northbound APIs; Stage 2".

[9] 3GPP TR 23.700-98: "Study on Enhanced architecture for enabling Edge Applications".

[10] 3GPP TS 29.122: " T8 reference point for Northbound APIs".

# 3 Definitions of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms given in 3GPP TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in 3GPP TR 21.905 [1].

Internet of Things (IoT): Network supporting interconnection of devices, machines, and low complexity entities.

IoT Application: An application catering to one or more vertical domains which includes communication functionality for the Internet of Things.

IoT Client: An entity that provides the client-side functionalities corresponding to a specific IoT Application.

IoT Platform: An entity hosting a collection of services and enabling capabilities that supports IoT Applications.

IoT Platform service: A generic name for a common service or enabling capability provided by an IoT Platform.

IoT Platform Provider: A mobile network operator or a 3rd party service provider offering IoT Platform services to multiple 3rd party service providers or ASPs.

IoT Server: An entity that provides the server side functionalities corresponding to a specific IoT Application.

NOTE: The normative phase work will determine the mappings between these entities and SEAL entities.

## 3.2 Symbols

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

<symbol> <Explanation>

## 3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in 3GPP TR 21.905 [1].

ASM Application Service Management

ASP Application Service Provider

BDT Background Data Transfer

CAPIF Common API Framework for northbound APIs

CIoT Cellular IoT

CP Control Plane

CPP Communication Patterns Parameters

MIoT Massive IoT

NEF Network Exposure Function

NIDD Non-IP Data Delivery

SEAL Service Enablement Application Layer

SCEF Service Capability Exposure Function

TAU Tracking Area Update

VAL Vertical Application Layer

# 4 Key Issues

## 4.1 Key issue #1: Background Data Transfer negotiation

The cellular network provides background data transfer (BDT) capabilities which save network resources and reduce device energy consumption. The 5GS provides procedures for BDT negotiation services as described in 3GPP TS 23.502 [2] clause 4.16.7. The corresponding NEF services (see 3GPP TS 23.502 [2] clause 5.2.6.6) allow the servers to provide parameters for requesting or optimizing the background data traffic for a set of UEs.

Scenarios in which an IoT Platform interfaces with the Core Network to request future Background Data Transfer (BDT) Policies on behalf of IoT Servers need to be investigated further. In such scenarios the IoT Platform Provider leverages for its services a different business relationship with the MNO than the ASPs providing the individual IoT Applications. For example, a Smart City Platform may be configured to understand the reference to a charging rate based on the agreement with the operator (see 3GPP TS 23.503 [3] clause 6.1.2.4). At the same time, the ASP providing green building IoT applications hosted by the IoT Platform may rely on no special operator agreements.

In such scenarios, the IoT Servers provide the IoT Platform information such as expected data volume per UE, preferred time window or optional location, for a set of targeted IoT Clients. The IoT Platform negotiates future BDT policies with the Core Network as detailed in 3GPP TS 23.502[2] clause 4.16.7. Then the relevant information from the selected BDT policy is provided to the initiating IoT Servers, in order to initiate the data transfer.

Hence, it is required to study:

- How the SEAL functional model and deployment options may be leveraged for the implementation of IoT Platforms supporting such scenarios.

- Whether and how the IoT Platform in this scenario can:

- determine the reference to the charging rate (based on agreement with the operator) without exposure of this information to the IoT Servers.

- aggregate requirements/ requests from multiple IoT Servers resulting in a single future BDT negotiation request, e.g., when the same group of UEs is targeted.

- select a BDT Policy from a set of Possible Transfer Policies provided by the Core Network, based on IoT Server input.

## 4.2 Key issue #2: Application server monitoring and control of traffic

Clause 31 of 3GPP TS 22.101 [4] has provided requirements for control of traffic from UE-based applications toward associated server. When an application on a third-party server or the third-party server itself becomes congested or fails, the traffic towards that server needs to be controlled to avoid/mitigate potential issues caused by resulting unproductive use of 3GPP network resources. Following are some of the requirements:

*The 3GPP network shall be able to control (i.e., block and/or prioritize) traffic from UEs to an application on a third-party server or the third-party server itself without affecting traffic to other applications on the third-party server or to other third-party servers.*

*The 3GPP network shall be able to receive a status indication from the third-party server when an application on it is experiencing congestion or failure, and when normal operation resumes. Such a status indication may be sent periodically, and/or when the status of the application changes.*

*The 3GPP network shall be able to detect and monitor a third-party server's operational status e.g., congestion levels, failure, and unavailability of the third-party server.*

Consider a scenario where an IoT application server serves millions of devices. In an event where the application server's response time is increased due to high traffic congestion at the application server, it is required to able to control (i.e., block and/or prioritize) the traffic from the UEs or IoT devices towards the application server.

Hence, it is required to study:

1) How to enhance the enabler layer (e.g., SEAL, CAPIF) to support service monitoring of third-party application servers?

2) How to monitor a third-party server's operational status e.g., congestion levels, failure, and unavailability of the third-party server? How to control/influence the traffic when the third-party application server is experiencing congestion or failure, and when normal operation resumes?

## 4.3 Key issue #3: IoT Platform PSM monitoring and configuration

The cellular network exposes the capabilities for IoT Servers to provide network parameters such as: maximum response time, maximum latency, and suggested number of downlink packets. These parameters may then be used by the Core Network to derive periodic TAU Timer and Active time for the PSM mode, or to configure in-network extended buffering.

IoT Platforms provide services to multiple IoT Servers with different services for the same UEs using PSM mode. Each IoT Sever is aware only of its own communication requirements, e.g., communication schedules, delay tolerance for regular or high-priority messaging. Based on these communication requirements, some IoT Servers may be able to derive how long the UE should be unreachable for power saving purposes, or how long the UE should stay active before returning to PSM. In the meantime, other IoT Servers may be agnostic of the of underlying network used, therefore unable to derive such information.

If separate Core Network parameter configurations are derived individually by each IoT Server for the same UE, the CN determines the final value. However, this determination does not have any service-level context information. In some cases, the greatest optimizations are achieved when the calculations take in consideration the current network parameter configuration, which currently can be inferred only through the use of device monitoring.

When extended buffering is employed, the Network Parameter configuration can also be used by the application layer for retransmission timer configuration.

Scenarios in which an IoT Platform interfaces with the Core Network to provide optimal configuration of network parameters determined based on the communication characteristics from multiple IoT Servers need to be investigated further.

Hence, it is required to study whether and how the needs of multiple IoT servers can be satisfied via a single SEAL IoT Platform for the following:

- Capability to determine optimal network parameter values based on communication characteristics (e.g., communication schedules, delay tolerance) of services from multiple IoT Servers.

- Capability to configure device monitoring and receive Monitoring Event notifications via exposure APIs.

- Capability to provide the network parameter values (e.g., maximum response time) via exposure APIs, to the Core Network.

- Capability to expose the derived network parameter configuration (e.g., from monitoring events) to the application layer.

## 4.4 Key issue #4: Configuration of Communication Patterns

Many IoT devices have predicable communication behaviour. The Communication Pattern Parameters (CPP) Provisioning network exposure API has been introduced to make this information available to the network for resource planning and optimizations (see 3GPP 29.522 [5] clause 4.4.5). This capability provides optimizations for the devices as well, by way of reduction of signalling, energy saving, fewer sleep/awake transitions, etc.

IoT Servers should be able to provide the IoT Platform information on the communication behavior of the application(s) supported. In turn, the IoT platform should be able to aggregate the communication patterns of multiple applications interacting with the same UE before providing them to the underlying network.

Given the scale and the importance of this functionality, the CPP Provisioning capability also supports providing single configurations for UE groups, e.g., for support of MIoT. An IoT platform implemented using SEAL services should be enabled to implement CPP Provisioning for groups of UE, along with performing group management. However, the IoT Platform needs to be enabled to be provided with corresponding input by the individual IoT Servers.

Hence, it is required to study:

- whether and how use of the CPP Provisioning API can be enabled on IoT Platforms implemented using SEAL services, using configurations provided by multiple IoT Servers for the same UE.

- whether and how use of CPP Provisioning API use can be enabled on IoT Platforms implemented using SEAL services to provide configurations for UE groups managed via the IoT Platform.

## 4.5 Key issue #5: NIDD configuration

Since Release 13, 3GPP introduced the ability to send data to and from the UE in NAS messaging though the "Control Plane (CP) CIoT Optimizations" feature. Since no data plane set up is required, using the CP CIoT optimizations results in a reduced total number of control plane messages that are required to send short data transactions. CIoT Optimizations include the option of exchanging NEF-anchored Non-IP Data.

Use of Non-IP Data Delivery (NIDD) requires an initial NEF configuration step, which includes MTC Provider Information and Reliable Data Service Configuration, both of which usually require a pre-established relationship with the MNO. In addition, for the purpose of sending mobile-terminated Group NIDD messages, the NIDD Configuration procedure is used by the NEF to resolve the mapping of External Group Identifier to individual UEs. Therefore, an IoT platform implemented using SEAL services should be enabled to perform NIDD configuration for UE groups on behalf of the IoT Servers.

Hence, it is required to study whether and how NIDD configuration services by the IoT Platform implemented using SEAL services, on behalf of one or more IoT Servers, may be enabled. Configuration of group NIDD services should be also included in the study.

## 4.6 Key issue #6: Device Triggering services.

Currently, the SEAL specification in 3GPP TS 23.434 [6] does not include functionality for leveraging the Device Triggering network exposure API.

Typically, commercial IoT Platforms store and maintain a "digital twin" of the device. Such digital twin is accessed and used by the device (e.g., to publish sensor readings to the IoT Platform or to receive commands from the IoT Platform) as well as by the IoT Applications to retrieve sensor readings of a device or to issue commands to a device. This digital twin has several benefits since it allows the IoT Platform to reduce communication loads and allows devices to go to sleep. This can be done without impacting the availability of device information to the IoT Applications.

The communication between the IoT Applications and the device is asynchronous and is managed by the IoT platform. The IoT platform needs to perform device triggering when synchronization between the digital twin is needed and the UE needs to establish connectivity.

Hence, it is required to study whether the IoT Applications should directly use the existing network exposure capability for device triggering or whether the IoT Platform should be enhanced to provide device triggering support on behalf of the IoT Applications.

# 5 Solutions

## 5.1 Solution #1: Application Service Management Service

### 5.1.1 Functional model for application service management

#### 5.1.1.1 General

The Application Service Management (ASM) service provides application service monitoring and traffic control service. The functional model for the application service management is based on the generic functional model specified in clause 6 of 3GPP TS 23.434 [6]. It is organized into functional entities to describe a functional architecture which addresses the support for application service management aspects for vertical applications. The on-network functional model is specified in this clause.

#### 5.1.1.2 On-network functional model description

Figure 5.1.1.2-1 illustrates the generic on-network functional model for application service management.



Figure 5.1.1.2-1: On-network functional model for application service management

NOTE: The normative phase work will determine whether and how service-based interfaces are used in this architecture.

The interface between ASM client and ASM server is ASM-UU interface. The ASM client interacts with VAL client using ASM-C interface and ASM server interacts with VAL server using ASM-S interface.

The ASM server communicates with the SCEF via T8 reference point or communicates with the NEF via N33 reference point to control application specific traffic from the underlying 3GPP network system.

### 5.1.2 Functional entities description

#### 5.1.2.1 General

The functional entities for application service management SEAL service are described in the following subclauses.

#### 5.1.2.2 Application service management client

The application service management client acts as a SEAL client for application service monitoring and traffic control function.

#### 5.1.2.3 Application service management server

The ASM server is a functional entity used to configure one or more application servers related to 3GPP system information as well as the network. It also acts as a co-ordination layer in subscribing to the VAL sever and 3GPP network for any changes in the configuration and inform the other in order to perform necessary actions (like blocking or controlling traffic).

### 5.1.3 Configuring VAL server

To monitor status of the application service, the ASM server can configure the VAL server to monitor status information.



Figure 5.1.3-1: Configuring VAL server

1) In order to configure the VAL server, the ASM server sends ASM Configuration request towards VAL server including all parameters to be monitored the VAL server. The parameters include average request queue length, average time to process request, CPU usage, memory usage, etc.

2) Upon received the request, the VAL server sends response and starts monitoring all activities.

NOTE: It is to be addressed in the normative phase whether configuring VAL server procedure is required or not.

### 5.1.4 Request-response model

#### 5.1.4.1 ASM server requesting for on-demand status

The ASM server can request the VAL server to provide its status information. The ASM server may choose to request status information periodically, which can give the ASM server the possible trend of the VAL server status. Figure 5.1.4.1-1 shows the procedure for the ASM server to request on-demand status request from the VAL server.



Figure 5.1.4.1-1: ASM server requesting for on-demand status

1) Based on configurations such monitoring status information request from other entities (e.g., VAL service provider), ASM server initiates the on-demand status request towards the VAL server.

2) The ASM server sends an on-demand status request to the VAL server. The request includes the configuration parameters whose status needs to be included in the response.

3) The VAL server immediately responds to the ASM server with a report containing status information identified by the ASM server and available to the VAL server.

4) Upon receiving the report, the ASM server updates the monitoring status of the VAL Server to the other VAL UE(s), VAL Client(s) or 3GPP network as required.

#### 5.1.4.2 ASM Client pushes service experience report to the ASM Server

The ASM client keeps monitoring the different service KPIs as experienced for the application service. The KPIs include application specific performance measurements like end-to-end response time, connection bandwidth, request rate, server availability time, etc. On request from user or VAL client or any other trigger conditions, the ASM client sends the service experience report about a VAL server to the ASM server. Figure 5.1.4.2-1 illustrates a scenario of ASM Client pushes service experience report to the ASM Server.

Pre-condition:

1). ASM client determines to send service experience report based on certain criteria (e.g. VAL User request, VAL client request, any preconfigured or explicit configured triggering event, periodic event, like so)



Figure 5.1.4.2-1: ASM Client pushes service experience report to the ASM Server

1). The ASM client sends Push service experience request to the ASM server. The request contains service experience report about a VAL server (e.g. end-to-end response time as experienced by client, connection bandwidth, request rate by client, VAL server availability, etc) and includes the VAL UE ID, VAL service ID, VAL server identity for which the report is being sent and time stamp of the report.

2). Upon receiving the Push service experience request from the ASM client, the ASM server stores the report in to the database or permanent storage.

3). The ASM server may take further actions based on the analysis of the report as shared by the ASM client.

a) Based on service experience report, the ASM server may decide to collect additional information from other UEs or ASM clients which use the same VAL server (e.g. by using pull service experience procedure as specified in clause 5.1.4.3).

4). The ASM server sends Push service experience response to the ASM client.

NOTE: Step 3 and 4 can be performed in parallel.

5). The ASM server may determine the corrective action as specified in clause 5.1.4.4. If the issue is identified with the VAL UE, the ASM server may inform the corrective actions to be taken by the VAL UE.

#### 5.1.4.3 ASM Server pulls service experience report from the ASM Client

Figure 5.1.4.3-1 illustrates the procedure for the ASM server to pull the service experience report from the ASM clients. The procedure can be initiated by the ASM server upon receiving a Push service experience request from an ASM client or upon receiving a request from application service provider (application server) to get the service experience report from the clients or any other event that requires the ASM server to determine the service experience data.



Figure 5.1.4.3-1: ASM Server pulls service experience report from the ASM Client

1). The ASM server sends Pull service experience request to the ASM client. The request contains identity of the specific VAL server for which the service experience report is required. The request includes the VAL service ID and VAL server identity for which the report is requested.

2). Upon receiving the Pull service experience request from the ASM server, the ASM client takes user consent to send the report if the user consent is not available already.

3). The ASM client sends the Pull service experience response to the ASM server. The response contains result indicating whether the report is available or not. If report is available, the response contains service experience report about a VAL server (e.g. end-to-end response time as experienced by client, connection bandwidth, request rate by client, VAL server availability, etc) and includes the VAL UE ID, VAL service ID, VAL server identity for which the report is being sent and time stamp of the report.

4). The ASM server stores the service experience report into the database or permanent storage.

5). The ASM server may takes the corrective action as specified in clause 5.1.4.4.

#### 5.1.4.4 ASM Server determines corrective action

Based on the collective analysis of all reports, the ASM server performs following actions:

1) Determining the entity producing the issue, i.e. whether the reported issue is at VAL client or VAL server or 3GPP network.

NOTE 1: How the ASM server determines whether the issue is at the VAL client, VAL server or the 3GPP network is implementation specific

2) Once the entity causing the issue is identified, the ASM Server need to determine the corrective action for that entity.

NOTE 2: The ASM server may subscribe for monitoring events (e.g. UE reachability, Communication failure, PDU Session Status) to 5GC as specified in clause 5.2.6.2 of 3GPP TS 29.502 [2] and may uses the information received via notification (along with the service experience reports received from ASM client) in determining the entity producing the issue and the possible corrective action for that entity.

NOTE 3: ASM server logic and algorithm to determine the corrective action based on the entity causing the issue (i.e. issue is at VAL client or VAL server or the 3GPP network), is out of scope of this specification and implementation specific.

3) ASM server informs the corrective action towards the entity causing the issue.

NOTE 4: How ASM server indicates the suggestion for action to VAL layer entities is implementation specific.

### 5.1.5 Subscribe-notify model

#### 5.1.5.1 ASM server subscribes to the monitoring status information

Figure 5.1.5.1-1 shows the procedure for the ASM server to subscribe to the monitoring status information of the VAL server. The ASM server does not need to configure the VAL server in advance.



Figure 5.1.5.1-1: ASM server requesting for on-demand status

1) Based on configurations request from other entities to provide monitoring status information periodically or based on event, ASM server decides to initiate the monitoring information subscribe request to the VAL server.

2) The ASM server sends a monitoring information subscribe request to the VAL server. The request includes all parameters to be monitored for UE and VAL server. The parameters include average request queue length, average time to process request, CPU usage, memory usage, etc. The request also includes events when the VAL server needs to send the notification.

3) The VAL server sends a monitoring information subscribe response to the ASM server containing success or failure of the request.

#### 5.1.5.2 VAL server notifies the monitoring status information

Figure 5.1.5.2-1 shows the notification from VAL server towards the ASM server on occurrence of the event.

Pre-conditions:

1) The ASM server is subscribed to the monitoring status information.



Figure 5.1.5.2-1: VAL server notifies the monitoring status information

1) When the event occurs as specified by the ASM server in the subscription request, the VAL server sends monitoring information notification to the ASM server with a report containing status information identified by the ASM server and available to the VAL server.

2) Upon receiving the report, the ASM server updates the monitoring status of the VAL to the other VAL UE(s), VAL Client(s) or 3GPP network as required.

### 5.1.6 Reference Points

#### 5.1.6.1 General

This clause describes the reference points of the architecture for application service management.

#### 5.1.6.2 ASM-C

This is the interface between ASM Client and VAL Client. It is used by ASM client to share information regarding application service monitoring and provisioning of UE clients.

#### 5.1.6.3 ASM-UU

This is the interface between ASM Client and ASM Server. It is utilized by ASM Server to configure UE and provide policies related to the application service. The ASM server uses this interface to provide notifications about possible actions to control the traffic.

#### 5.1.6.4 ASM-S

This is the interface between ASM Server and VAL Server. It is used for provisioning and monitoring of VAL AS as per the policies and profiles of the VAL service. The 3GPP network status for the VAL service is monitored by ASM Server and notified to the VAL server if changes for it to take suitable action. This is service-based interface where VAL server and ASM Server exposes services as APIs, to be invoked by other entities.

5.1.7 Evaluation

The solution addresses KI#2 related to Application server monitoring. The solution proposes functional model for application service management, where ASM server collects data from VAL server or ASM client (along with VAL client) and if required, it decides corrective action to take for VAL client or VAL server. ASM server logic and algorithm to determine the corrective action is implementation specific. The solution proposes data collection procedures using request-response model and also subscribe-notify model.

## 5.2 Solution #2: IoT Platform deployment options

### 5.2.1 General

IoT Platforms enable applications and services from multiple verticals by providing a set of common services for deploying dynamic, performant, scalable and resilient services which may be underlying-network agnostic.

Fig 5.2.1-1 depicts a generic IoT Platform with IoT Platform Common Services (IoT-PCS) servers enabling a set of applications deployed using corresponding servers (IoT-App), which may belong to different verticals. On the device side, corresponding IoT-PCS and IoT-App clients enable the client-side functionality.



Figure 5.2.1-1: Generic representation of services enabled via IoT Platforms

The following clauses introduce functional models for supporting a variety of IoT Platform deployments based on the generic functional model specified in clause 6.2 of 3GPP TS 23.434 [6].

NOTE: The IoT Platform functional and deployment models in this document focus only on SEAL functionality to support application capability exposure to general purpose servers or 3rd party IoT applications.

### 5.2.2 Deployment models in single PLMN operator domain

#### 5.2.2.1 General

IoT Platform deployment models for single PLMN operator domain case are described in this clause.

The following models are differentiated primarily based on which entities have network exposure access and may implement the necessary functionality. Therefore, for the purpose of network exposure, IoT-PCS servers are implemented as SEAL servers. While some IoT-PCS services may be implemented as VAL services, the representation of that implementation option is abstracted in the following models.

#### 5.2.2.2 Single network exposure access model

Figure 5.2.2.2-1 illustrates the single network exposure access deployment model in single PLMN operator domain



Figure 5.2.2.2-1: Single network exposure access deployment model in single PLMN operator domain

In this model network interfaces are available to the IoT-PCS servers providing SEAL services via the SEAL-S reference point to IoT Application (IoT-App) servers as VAL servers. The IoT-PCS servers communicate with the SCEF via T8 reference point or with the NEF via N33 reference point.

The interface between the IoT-PCS client and IoT-PCS server is an instance of a SEAL-UU reference point, e.g., CM-UU. The IoT-PCS client interacts with IoT-App client (as a VAL client) using an instance of a SEAL-C reference point. The IoT-PCS server interacts with IoT-App server(s) over instance(s) of SEAL-S reference point(s).

#### 5.2.2.3 Distributed network exposure access model

Figure 5.2.2.3-1 illustrates the distributed network exposure access deployment model in single PLMN operator domain



Figure 5.2.2.3-1: Distributed network exposure access deployment model in single PLMN operator domain

In this model network interfaces are available to the IoT-PCS servers as well as to the IoT-App severs, therefore they both implement SEAL services.

The IoT Application uses both SEAL and VAL servers and clients. The IoT-PCS server(s) provides additional SEAL services via the SEAL-S reference point to IoT-App VAL servers.

The IoT-PCS and IoT-App SEAL severs communicate with the SCEF via T8 reference point or with the NEF via N33 reference point. It is assumed that each SEAL service deployed may be provided to a given IoT Application by either or both SEAL servers and that the IoT Platform provider can configure the PCS and IoT Applications in the IoT Platform with policies determining unambiguously which SEAL service to be used for the IoT application operations. The IoT-PCS and IoT-App SEAL servers may interact over an instance of SEAL-X reference point.

The interface between the IoT-PCS SEAL client and IoT-PCS SEAL server is an instance of a SEAL-UU reference point, e.g., GM-UU. The interface between the IoT-App SEAL client and IoT-App SEAL server is also instance of a SEAL-UU reference point, e.g., CM-UU.

The SEAL clients interact with VAL clients using SEAL-C reference points. The IoT-PCS SEAL server interacts with IoT-App VAL server over an instance of a SEAL-S reference point.

## 5.3 Solution #3: IoT Platform Functional Models

### 5.3.1 General

The functional model for IoT Platform services is based on the generic SEAL functional model specified in 3GPP TS 23.434 [6] clause 6. It is organized into functional entities to describe a functional architecture which addresses the support for IoT Platforms.

### 5.3.2 On-network functional models

Figure 5.3.2-1 illustrates the generic on-network functional model for IoT Platform services in single network exposure access deployment mode.



Figure 5.3.2-1: Functional model for IoT-PCS (single network exposure access)

The IoT-PCS client communicates with the IoT-PCS server over the IP-UU reference point, which is an instance of the SEAL-UU generic reference point. The IoT-PCS client provides IoT platform common services functionality to the IoT-App client(s) over IP- C reference point, which is an instance of the SEAL-C reference point. The VAL IoT-App server communicates with the IoT-PCS server over the IP-S reference point, which is an instance of the SEAL-S generic reference point.

The IoT-PCS server communicates with the SCEF via T8 reference point. The IoT-PCS server communicates with the NEF via N33 reference point by mechanisms defined in clause 5.2.6.2 of 3GPP TS 23.502 [2].

When IoT Applications are deployed in a stand-alone mode, without the use of an IoT Platform service, the functional model in Figure 5.3.2-2 applies. In this case, SEAL services and network exposure may be implemented internally to IoT-App.



Figure 5.3.2-2: Functional model for stand-alone IoT Applications (without IoT-PCS services)

For stand-alone IoT applications, the IoT-App client communicates with the IoT-App server over the IP-UU reference point, which is an instance of the SEAL-UU generic reference point. The IoT-App client provides application-specific services to the IoT-App client(s) over IP- C reference point, which is an instance of the SEAL-C reference point. The VAL IoT-App server(s) communicate with the IoT-App SEAL server over the IP-S reference point, which is an instance of the SEAL-S generic reference point.

The IoT-App server communicates with the SCEF via T8 reference point and with the NEF via N33 reference point.

When an IoT Application designed based on a stand-alone model, is deployed in conjunction with an IoT platform, inter-service communication is required between the two SEAL servers. Figure 5.3.2-3 illustrates the functional model for inter-service communications between an IoT-PCS SEAL Server and an IoT-App SEAL Server.



Figure 5.3.2-3: Inter-service communication between IoT-App and IoT-PCS SEAL servers

For inter-service communications, an IoT-App SEAL Server communicates with the IoT-PCS server over the SEAL-X3 reference point. In this deployment, both SEAL servers provide network exposure access, resulting in a distributed network exposure access deployment. Figure 5.3.2-4 depicts the resulting deployment. Note that this deployment aligns with the distributed network exposure access model introduced by the solution in clause 5.2, while using the proposed IoT-PCS-specific instances of SEAL reference points.



Figure 5.3.2-4: Functional model for IoT-PCS (distributed network exposure access)

## 5.4 Solution #4: Device triggering

### 5.4.1 General

An IoT-PCS Server may initiate a device trigger to a UE to cause it to establish a connection to the IoT-PCS Server, to connect to another server on the platform, to provide updated information, etc. The IoT-PCS Server may initiate the device trigger itself (implicit) or it may be initiated by a request that the IoT-PCS Server receives from IoT-App VAL Servers (explicit).

### 5.4.2 Procedures and information flows

#### 5.4.2.1 Procedure



Figure 5.4.2.1-1: Device Triggering Procedure

1). The device triggering procedure may be initiated based on an optional interaction between the IoT-PCS Server and an IoT-App VAL Server, For example, the IoT-App VAL Server may send an explicit API request, or the IoT-App VAL Server may send another message based on which the IoT-PCS determines to start the device triggering procedure.

2). The IoT-PCS Server determines to initiate the device triggering. The IoT-PCS Server may use UE Availability and/or pre-configured information to determine the timing of the Device Triggering request, e.g. the trigger may be sent to ensure that a target UE in PSM mode is reachable when resuming communications.

3). The IoT-PCS Server performs the device triggering procedure described in 3GPP TS 23.682 [7] clause 5.2. The procedure requires that the UE Identifier, port number(s) and protocol information are available at the IoT-PCS Server.

NOTE: It is to be addressed in the normative phase how the port number(s) and protocol information are being made available to the IoT-PCS Server

As part of the procedure, the IoT-PCS Server receives a Device Triggering delivery status report from SCEF/NEF indicating the success of the delivery.

4). If a request was received in step 1, the IoT-PCS Server responds to the request.

Based on the trigger purpose derived from the payload, the targeted IoT-PCS Client or IoT-App Client performs the corresponding actions (e.g., connect to the IoT-App VAL Server).

### 5.4.3 Evaluation

This solution addresses Key Issue #6, enabling the IoT-PCS to provide device triggering support on behalf of the IoT Applications.

## 5.5 Solution #5: Application Server status monitoring via CAPIF

### 5.5.1 Description

This solution addresses KI#2 by providing enhanced CAPIF services for managing application server (AS) status. The CAPIF core function (CCF) takes the responsibility to monitor AS service status. The CCF may expose the monitored AS service status to CCF service consumers (e.g. API invoker) via enhanced service discovery and event exposure procedures.

NOTE: One AS can provide one or more AS services.

#### 5.5.1.1 AS service status monitoring

When a 3rd party Application Server provides its service (also see KI#2 and sol#8 in 3GPP TR 23.700-98 [9]), it acts as Application Exposure Function (AEF) in CAPIF.

The application services provided by the AS can be published in the CCF so that the services are discoverable by the API invoker. When an AS acting as an AEF publishes its service API to the CCF, the AS updates its service API status (active, inactive) at the CCF. Table 5.5.1.1-1 shows the impact (with **bold** text) on the existing service API publish information flows in 3GPP TS 23.222 [8] as example, the same addition applies for the service API update request (e.g. to update service API status) and interconnection service API publish request.

Table 5.5.1.1-1: Service API publish request

|  |  |  |
| --- | --- | --- |
| Information element | Status | Description |
| API publisher information | M | The information of the API publisher may include identity, authentication and authorization information |
| Service API information | M | The service API information includes the service API name, service API type, **service API status (e.g. active, inactive),** communication type, description, Serving Area Information (optional), AEF location (optional), interface details (e.g. IP address, port number, URI), protocols, version numbers, and data format. |
| Shareable information | O (see NOTE) | Indicates whether the service API or the service API category can be published to other CCFs. And if sharing, a list of CAPIF provider domain information where the service API or the service API category can be published is contained. |
| NOTE: If the shareable information is not present, the service API is not allowed to be shared. | | |

For service API discovery, the CCF consumer (e.g. API invoker) can discover the service API status via service API discovery procedure or be notified about the service API status change via CAPIF event exposure procedure. The Application publishing function (APF) can update the service API status to the CCF.

The service status exposed by CCF is supported by "Availability of service APIs" event in clause 8.8.6 of 3GPP TS 23.222 [8].

### 5.5.2 Evaluation

This solution provides monitoring of service API status monitoring for the AS using CAPIF, which addresses KI#2. The control of application traffic towards the AS is up to the consumer of the AS service API to decide after AS service API discovery.

## 5.6 Solution #6: BDT configuration

### 5.6.1 General

For an IoT Platform, the use of Background Data Transfer (BDT) allows IoT-App VAL Servers to use transmission time windows that are/cost and/or throughput favorable. The network provider is also enabled to provide better network resource management for predictable usage of downlink data.

### 5.6.2 Procedures and information flows

#### 5.6.2.1 General

The Background Data Transfer feature requires an initial step in which policies are requested and negotiated. BDT Policy requests to the 3GPP network are based on an expected time window and UE set, with additional optional information e.g., expected data volume per UE. The UE set may be indicated as an expected number of UEs, a group ID or geographical area.

The feature allows for the server involved to negotiate the policies proposed by the network. It also allows the server to enable notifications to be sent, should network conditions affect future BDT policies.

Based on the BDT policies obtained using the procedures detailed in this clause, an IoT-App VAL server can initiate a data transfer to the client at the negotiated time and with the negotiated charging rates. The data transfer between the IoT-App VAL Server and the IoT-App VAL Client is performed without IoT-PCS Server enablement, but the IoT-App VAL Server may utilize functionality exposed by SEALDD or 5GMSG Servers. Service layer functionality for the purpose of facilitating the data transfer with the negotiated policy is not in scope of this specification.

#### 5.6.2.2 Request and Select Background Data Transfer Policy

Figure 5.6.2.2-1 depicts a general procedure for the request and configuration of traffic policies for BDT initiated by a request from an IoT-App VAL Server.



Figure 5.6.2.2-1: General Procedure for configuration of Background Data Transfer

**Step 1:** An IoT-App VAL Server requests IoT-PCS Server to negotiate with the 3GPP network a background data transfer policy.

The request includes expected data volume, expected number of UEs, expected time window for the background data transfer. The request may also include group ID, geographic information for the UEs, a request expiration time, guidance for policy selection. If guidance for policy selection is not included, the IoT-App VAL Server indicates if IoT-PCS Server may choose independently from among multiple transfer policies.

**Step 2:** Based on the request expiration time and Service Provider policies, IoT-PCS Server may determine to delay interactions with the 3GPP network in order to negotiate on behalf of multiple IoT-App VAL Servers.

The IoT-PCS Server performs the resource management of background data transfer procedure described in 3GPP TS 23.502 [2] clause 4.16.7.2. The procedure requires that expected data volume, expected number of UEs, and expected time window are provided by the IoT-PCS Server. If the IoT-PCS Server determined to negotiate on behalf of multiple IoT-App VAL Servers, the parameters included reflects a superset of the individual IoT-App VAL Server requests.

NOTE 1: The IoT-PCS Server determines to negotiate on behalf of multiple IoT-App VAL Servers based on implementation options and local policies. For example, if the request expiration time and expected time window are sufficiently large and, respectively, far away in time, the IoT-PCS Server may be allowed to delay the negotiations with the 3GPP network in case another request is received, targeting the same group of UEs. If another request is received with expected time windows sufficiently close and if the guidance for policy selection allows, a single policy/time window may be negotiated instead. This allows the UE group to wake up only once for multiple background data transfers.

The 3GPP network determines one or more applicable transfer policies based on the requesting Background Data Transfer parameters. A list of transfer policies is provided to the IoT-PCS Server. Each transfer policy includes mandatory Reference ID, charging rating group reference and allocated time window and optional maximum UL and DL bandwidth. The IoT-PCS Server uses ASP policies and the transfer selection guidance (if available) to select a policy*.* The IoT-PCS Server informs the 3GPP Network of the selected transfer policy.

NOTE 2: Based on 3GPP TS 23.503[3] clause 6.1.2.4. it is assumed that the IoT-PCS server is configured to understand the charging rating group reference based on agreements with the operator.

NOTE 3: Policy selection guidance options such as "lowest cost", "highest throughput", etc. are to be determined in the normative phase.

NOTE 4: The IoT-PCS server sets the warning notification based on local policies.

**Step 3:** The IoT-PCS Server responds to the IoT-App VAL Server, providing the Reference ID and allocated time window of the background data transfer policy.

5.6.2.3 Reselect Background Data Transfer Policy

Figure 5.6.2.3-1 depicts a general procedure for reselecting BDT policies after BDT warning.



Figure 5.6.2.3-1: General Procedure for reselecting BDT policies after warning

**Step 1:** The 3GPP Network, via NEF, sends the BDT warning (BDT Policy negotiate) notification to the IoT-PCS server. The notification includes the affected BDT policy Reference ID and list of candidate BDT policies.

Each of the BDT policies in the candidate BDT list includes mandatory Reference ID, charging rating group reference and time window, as well as optional maximum UL and DL bandwidth.

**Step 2:** TheIoT-PCS Server checks the new BDT policies included in the candidate list of the BDT warning notification. The IoT-PCS Server determines whether the notification affects multiple IoT-App VAL Servers or not. The IoT-PCS Server uses ASP policies and the transfer selection guidance (if available) provided with the initial IoT-App VAL Server request to select a policy*.*

The IoT-PCS Server informs the 3GPP Network of the selected transfer policy or that no new policy has been selected by using steps 11-16 of the procedure for BDT warning notification in 3GPP TS 23.502[2] clause 4.16.7.3.

**Step 3:** The IoT-PCS Server ends a new response to the IoT-App VAL Server, providing information about the new policy, or that no policy is available. If a new BDT policy is available, the information provided to the IoT-App VAL Server includes the ID of the applicable policy and the time window.

### 5.6.3 Evaluation

This solution addresses Key Issue #1. The solution allows the IoT-PCS to aggregate requirements/ requests from multiple IoT-App Servers and to select a set of BDT policies after negotiation with the Core Network, based on all IoT-App Server inputs. The solution also enables the IoT-PCS to determine the reference to the charging rate (based on agreement with the operator) without exposure of this information to the IoT Servers. At the same time, the solution does not affect the ability of IoT-App VAL/SEAL Servers to directly interact with 5GS for BDT negotiation, if so configured.

The solution addresses only BDT policy negotiation, with the data transfer being performed without IoT-PCS Server enablement. IoT-App VAL Servers may utilize functionality exposed by SEALDD or 5GMSG Servers for facilitating the data transfer.

The interactions with the Core Network are fully specified and require no changes. The request and response for BDT configuration exchanged between the IoT-App VAL Server and IOT-PCS server (i.e., steps 1 and 3 in 5.6.2.2) require specification. Therefore, this is a viable solution.

## 5.7 Solution #7: UE unified traffic pattern and monitoring management

### 5.7.1 General

### 5.7.2 Procedures and information flows

#### 5.7.2.1 General

UE unified traffic pattern and monitoring management procedures allow IoT-PCS to offer services leveraging several CN exposure APIs: communication patterns configuration, network parameter values configuration and UE monitoring event management.

#### 5.7.2.2 UE unified traffic pattern and monitoring management subscription procedure

An IoT-App VAL or SEAL server can indicate to the IoT-PCS server its interest in receiving UE unified traffic patterns and monitoring management services by sending the UE unified traffic pattern and monitoring management subscription request.

The subscription requests from each IoT-App VAL or SEAL server also include the traffic pattern configuration of the requester, which refers to application-level patterns of data traffic. The IoT-PCS server aggregates the traffic patterns obtained from the requestors (and described in Table 5.7.2.5.1-2) to determine the UE unified traffic patterns per UE. The UE unified traffic patterns are described via Table 5.7.2.5.3-1 for the UE unified traffic pattern update notification. These aggregated traffic patterns per UE (termed UE unified traffic pattern) are updated/adjusted by the IoT-PCS Server based on information obtain from UE monitoring.



Figure 5.7.2.2-1: UE unified traffic pattern and monitoring management subscription procedure

1. In order to subscribe to the IoT-PCS Server services, the IoT-App VAL/ SEAL server sends the UE unified traffic pattern and monitoring management request, as detailed in clause 5.7.2.5. The subscription request may include IoT-App traffic pattern configuration, which provides the traffic patterns of the specific IoT-App VAL/SEAL Server. The request may also include Management subscription indications which indicate to the IoT PCS server which management and 5GC exposure procedures the IoT-App VAL/SEAL server allows the IoT-PCS Server to perform on its behalf.

2. Upon receipt of the request, the IoT-PCS server sends a UE unified traffic pattern and monitoring management subscription response.

3. The IoT-PCS Server aggregates UE unified traffic pattern and monitoring management subscription requests from different IoT-App VAL/SEAL servers and determines the UE unified traffic pattern per UE (using the traffic patterns of all the IoT-Apps communicating with the UE). If the IoT-PCS Server determines that additional or updated IoT-App traffic pattern configurations are needed, it requests them from the IoT-App Servers using the Traffic pattern configuration request procedure in clause 5.7.4.

4. Depending on the subscription requests received and local policies, the IoT-PCS Server executes one or more management and 5GC exposure procedure (per UE). Management and 5GC exposure procedures are detailed in clause 5.7.3.

The IoT-PCS Server determines the management procedures required to be executed on behalf of the IoT-App VAL/SEAL Servers .based on the received management subscription indications as follows:

- If the CP configuration indication is provided, the IoT-PCS executes the CP configuration procedure detailed in clause 5.7.3.2.

- If the UE unified traffic pattern monitoring management indication is provided, the IoT-PCS Server executes steps 1-3 of the UE unified traffic pattern monitoring procedure detailed in clause 5.7.3.3.

- If the UE unified traffic pattern monitoring update notification indication is provided, the IoT-PCS Server executes the steps 1-4 of the UE unified traffic pattern monitoring procedure detailed in clause 5.7.3.3.

- If the Network parameter coordination indication is provided, the IoT-PCS executes the network parameter coordination procedure detailed in clause 5.7.3.4.

NOTE 1: The IoT-PCS Server translates the management subscription indications received from different IoT-App VAL/SEAL Servers into per-UE management indications based on local policies and configurations. For example, an IoT-PCS Server may be configured to execute a management procedure for a UE if at least one IoT-App VAL/SEAL Server indicates it. Another IoT-PCS Server may be configured to provide all the management procedures for the UEs using the platform independent of IoT-App Server subscription indications.

NOTE 2: Corresponding subscription update request and unsubscribe request procedures will complement this functionality in the normative phase. These would allow the update of the subscription request parameters and the deletion of the entire subscription, respectively.

5.7.2.3 UE unified traffic pattern update notification procedure

An IoT-PCS Server can provide updated UE unified traffic pattern information to IoT-App VAL or SEAL servers by sending UE unified traffic pattern update notifications as shown in figure 5.7.2.3-1. The UE unified traffic pattern management procedure detailed in clause 5.7.3.3. is an example of procedure which may result in UE unified traffic pattern updates at the IoT-PCS server, based on which UE unified traffic pattern update notifications are provided.

Pre-conditions:

1) The IoT-App Val/SEAL server has subscribed for UE unified traffic pattern and monitoring management services, requesting to receive UE unified traffic pattern update notifications



Figure 5.7.2.3-1: UE unified traffic pattern update notification procedure

1. The IoT-PCS server sends the UE unified traffic pattern update notification when either of the following occurs:

- Monitoring events lead to updates in the UE unified traffic pattern (e.g., to schedule elements in Table 5.7.2.5.3-1) the IoT-PCS server sends a corresponding notification to the IoT-App VAL/SEAL server. Other notifications may be provided, e.g., if the stationary indication changes.

- An NP Configuration Notification is received with a new set of applied network parameters and if the IoT-PCS Server determines that the new configuration is incompatible with the current UE unified traffic pattern (see also clause 5.7.3.3 step 3).

#### 5.7.2.4 Traffic pattern configuration request procedure

To obtain information about service-specific traffic patterns for a UE, the IoT-PCS server can request traffic pattern configuration from the IoT-App VAL/SEAL server. This procedure may be used for example to request traffic patterns from IoT-App Servers which did not initiate UE unified traffic pattern and monitoring management, but which nevertheless communicate with the UE.



Figure 5.7.2.4-1: Traffic pattern configuration request

1. In order to obtain the service-specific traffic pattern configuration for a UE, the IoT-PCS Server sends traffic pattern configuration request to the IoT-App VAL/ SEAL server. The request parameters indicate the UE(s) for which the request is made.

2. Upon receipt of the request, the IoT-App VAL/SEAL server sends a response. The parameters include one or more traffic pattern elements.

#### 5.7.2.5 Information Flows

##### 5.7.2.5.1 UE unified traffic pattern and monitoring management subscription request

Table 5.7.2.5.1-1 describes the information flow for the UE unified traffic pattern and monitoring management subscription request from the IoT-App VAL/SEAL server to the IoT-PCS server.

Table 5.7.2.5.1-1: UE unified traffic pattern and monitoring management subscription request

|  |  |  |
| --- | --- | --- |
| Information element | Monitoring | Description |
| IoT-App VAL UE ID | M | UEs hosting IoT-App clients for which the subscription is requested |
| IoT-App VAL service ID | M | Identity of the VAL service for which the subscription is requested. |
| Management subscription indications | M | At least one of the following indications is to be provided |
| > CP configuration indication | O | Indicates whether CP configuration by the IoT-PCS Server with 5GC is requested. (see NOTE 1) |
| > UE unified traffic pattern monitoring management indication | O | Indicates that management of the UE unified traffic pattern is requested |
| > UE unified traffic pattern update notification indication | O | Indicates that notifications for updates of the UE unified traffic pattern monitoring is requested |
| > Network parameter coordination indication | O | Indicates whether network parameter coordination by the IoT-PCS with 5GC is requested (see NOTE 1) |
| IoT-App traffic pattern configuration | O | Traffic pattern configuration of the VAL service for this UE, as described in table 5.7.2.5.1-2. |
| NOTE 1: The CP configuration and the network parameter coordination functionality are also subject to policies available at the IoT-PCS Server. | | |

Table 5.7.2.5.1-2: IoT-App traffic pattern configuration

|  |  |  |
| --- | --- | --- |
| Information element | Monitoring | Description |
| Schedule elements | O | List of schedule elements applicable to the traffic patterns of the VAL service for this UE. Each schedule element is composed from seven fields: second, minute, hour, day of month, month, day of week and year. Each element indicates times or durations when the service traffic occurs. Multiple schedule elements can be used to create complex scheduling. (see NOTE 3) |
| Expiration time | O | Identifies when the IoT-App traffic pattern parameter configuration expire. If absent, it indicates that there is no expiration time. |
| Stationary indication | O | Identifies whether the UE is expected to be stationary or mobile while communicating using this traffic pattern configuration |

NOTE 3: The following is an example of a schedule element with the fields: second, minute, hour, day of month, month, day of week and year:

\*; 0-30 ; 2; \*; Jan-Sept; Tues; \*.

This schedule element, when used for IoT-App VAL traffic patterns translates to the following in the CpProvisioning API as described in 3GPP TS 29.122[10] clause 5.10:

- periodicCommunicationIndicator: TRUE

- communicationDurationTime: 30 min

- periodicTime: 1 week

- scheduledCommunicationTime: Tues, 2:00-2:30

- validityTime: calculated using the Jan-Sept range and the provided expiration time.

NOTE: The format of this IE is to be provided in stage 3. The purpose of this description is to clarify how the same element can contain multiple periodicities, specify start/stop times, etc.

##### 5.7.2.5.2 UE unified traffic pattern and monitoring management subscription response

Table 5.7.2.5.2-1 describes the information flow for the UE unified traffic pattern and monitoring management subscription response from the IoT-PCS server to the IoT-App VAL/SEAL server.

Table 5.7.2.5.2-1: UE unified traffic pattern and monitoring management subscription response

|  |  |  |
| --- | --- | --- |
| Information element | Monitoring | Description |
| Result | M | Indicates success or failure of the subscription request |

##### 5.7.2.5.3 UE unified traffic pattern update notification

Table 5.7.2.5.3-1 describes the information flow for the UE unified traffic pattern update notification from the IoT-PCS server to the IoT-App VAL/SEAL server.

Table 5.7.2.5.3-1: UE unified traffic pattern update notification

|  |  |  |
| --- | --- | --- |
| Information element | Monitoring | Description |
| UE ID | M | UE for which the UE unified traffic pattern update notification is provided for. |
| Schedule elements | O | Schedule element applicable to the unified traffic patterns of the UE. A schedule element is composed from seven fields: second, minute, hour, day of month, month, day of week and year. Each element indicates times or durations of UE availability. |
| Stationary indication | O | Identifies whether the UE is expected to be stationary or mobile while communicating using this UE unified traffic pattern, as determined by the IoT-PCS Server |
| Cause | O | This element is mandatory when the notification is provided to inform of a parameter configuration applied by the network which is incompatible with the existing Traffic Patterns. (see NOTE)  The element is optional when the notification informs of UE unified traffic pattern updates, providing additional information on the reason for the UE unified traffic pattern update (e.g. monitoring events received) |
| NOTE: For notifications of incompatible configurations, the normative phase can consider whether adding an optional element with a proposed IoT-PCS Server UE unified traffic pattern update is beneficial. | | |

##### 5.7.2.5.4 Traffic pattern configuration request

Table 5.7.2.5.4-1 describes the information flow for the Traffic pattern configuration request from the IoT-PCS server to the IoT-App VAL/SEAL server.

Table 5.7.2.5.4-1: Traffic pattern configuration request

|  |  |  |
| --- | --- | --- |
| Information element | Monitoring | Description |
| UE ID | M | UE for which the Traffic pattern configuration is requested. |

##### 5.7.2.5.5 Traffic pattern configuration response

Table 5.7.2.5.5-1 describes the information flow for the Traffic pattern configuration response from the IoT-App server to the IoT-PCS server.

Table 5.7.2.5.5-1: Traffic pattern configuration response

|  |  |  |
| --- | --- | --- |
| Information element | Monitoring | Description |
| UE ID | M | UE for which the Traffic pattern configuration is provided |
| IoT-App traffic pattern configuration | M | Traffic pattern configuration of the VAL service for this UE, as described in table 5.7.2.5.1-2. |

### 5.7.3 Management and 5GC exposure procedures

#### 5.7.3.1 General

#### 5.7.3.2 CP configuration procedure

The CP configuration procedure uses the information received by the IoT-PCS Server from the IoT-App regarding predictable communication behaviour of their services to provide information to 5GC for resource planning purposes, using an existing network exposure API.

Pre-conditions:

1. IoT-PCS Server determines to provide the service for a specific UE after receiving CP configuration indications in UE unified traffic pattern and monitoring management subscription requests, subject to policy.



Figure 5.7.2.2-1: CP configuration procedure

1. The IoT-PCS Server stores all Traffic pattern configurations received for the UE in subsequent UE unified traffic pattern and monitoring management subscription procedures and determines the UE unified traffic pattern. The IoT-PCS Server can also initiate Traffic pattern configuration requests to obtain additional configurations. The IoT-PCS Server uses the UE unified traffic pattern to determine a CpParameterSet, as defined in 3GPP TS 29.122[10], for the CpProvisioning API.

2. The IoT-PCS Server determines based on local policy when the CpProvisioning API is to be invoked and executes the CP parameter provisioning procedure described in 3GPP TS 29.122[10] clause 4.4.9.

#### 5.7.3.3 UE unified traffic pattern management procedure

The UE unified traffic pattern management procedure is used to determine and manage a unified traffic pattern applicable to a specified UE. The IoT-PCS Server then uses the 5GC exposure of UE monitoring events to update the UE unified traffic pattern.

Pre-conditions:

1. IoT-PCS Server determines to provide the service for a specific UE if either of the following conditions is true:

a) It receives UE unified traffic pattern monitoring management indications in UE unified traffic pattern and monitoring management subscription requests; or

b) It determines to provide Network parameter coordination services for the UE.



Figure 5.7.3.3-1: UE unified traffic pattern and monitoring management subscription procedure

The IoT-PCS Server determines an initial UE unified traffic pattern, e.g. by using all Traffic pattern configurations received for the UE .

2. The IoT-PCS Server determines, based on local policy that UE monitoring events are to be configured and executes the corresponding Monitoring procedure as described in 3GPP TS 29.122 [10] clause 4.4.2.

3. The IoT-PCS Server updates the UE unified traffic pattern based on the received monitoring events as follows:

- If a Monitoring Notification report for UE\_REACHABILITY is received, and *idleStatusInfo* information is provided in the report, the IoT-PCS Server changes the schedule element of the UE unified traffic pattern such that the duration of activity is set to the value of the *activeTime* parameter configured in the *idleStatusInfo*.

- If a Monitoring Notification report for AVAILABILITY\_AFTER\_DDN\_FAILURE is received after a UE transitions to idle mode, the IoT-PCS Server updates the schedule element of the UE unified traffic pattern such that: the start of an activity window is based on the Idle Timestamp, with a periodicity equal to the TAU/RAU Timer; the duration of the activity window indicates the Active Time value.

- If a Monitoring Notification report for COMMUNICATION\_FAILURE is received The IOT-PCS updates the schedule element of the UE unified traffic pattern to indicate that no communications are currently available (e.g. by using a keyword such as "NULL"). Local policies may specify events/ thresholds further defining when the IoT-PCS may provide a UE unified traffic pattern update based on monitoring events. For example, the update may be provided only after repeated communication failures are received within a timespan, or only if high reliability communications are expected. It is recommended that UE Reachability monitoring is also enabled in conjunction with the Communication Failure monitoring. This enables the IoT-PCS to provide updated timing information once the UE becomes reachable again.

- If a Monitoring Notification report for LOSS\_OF\_CONNECTIVITY is received, the IoT-PCS Server changes the schedule element of the UE unified traffic pattern to indicate that no communications are currently available

4. Conditional: The IoT-PCS Server notifies subscribers of the UE unified traffic pattern updates, as described in clause 5.7.2.3

#### 5.7.3.4 Network parameter coordination procedure

The network parameter coordination procedure uses UE unified traffic pattern information to influence aspects of UE/network behaviour such as the UE's PSM and extended idle mode DRX. For this purpose, parameter values may be suggested for Maximum Latency and Maximum Response Time for a UE. 5GC may choose to accept, reject or modify the suggested configuration parameter value.

Pre-conditions:

1. IoT-PCS Server determines to provide the service for a specific UE after receiving Network parameter coordination indications in UE unified traffic pattern and monitoring management subscription requests, subject to policy.

2. IoT-PCS Server determines and manages UE unified traffic patterns as described in clause 5.7.3.3.



Figure 5.7.3.4-1: Network parameter coordination procedure

1. The IoT-PCS Server determines to provide Network parameter configuration to 5GC. This determination can be based on updates to the UE unified traffic patterns resulting from interactions with IoT-App Servers (e.g. Traffic pattern configuration updates), on local policies, etc.

The IoT-PCS Server determines parameters the needed for NpConfiguration data structure as specified in 3GPP TS 29.122[10] from the UE unified traffic patterns as follows:

*- maximumLatency* – This value tells the network how long the UE is allowed to sleep. Setting it to 0 will disable PSM, extended idle mode DRX, and extended buffering. The IoT-PCS can extract the periodicity derived from the UE unified traffic pattern, which includes the schedule elements for the UEs communications with all IoT-Apps.The IoT-PCS Server sets *Maximum Latency* to be approximately the periodicity of the active periods derived from the schedule elementof the UE unified traffic pattern.

*- maximumResponseTime* – When the UE uses PSM, Maximum Response Time tells the network how long the UE should stay reachable after a transition to idle. When the UE uses eDRX, Maximum Response Time is used by the network to determine when to send a reachability notification before a UE's paging occasion. The IoT-PCS Server extracts a duration of activity from the schedule element of the UE unified traffic pattern and sets *Maximum Response Time* to reflect the duration of activity, indicating how long the UE should stay reachable for downlink communications.

2. The IoT-PCS Server performs the Network Parameter Configuration procedure as described in 3GPP TS 29.122[10] clause 4.4.12.

NOTE: The values provided by IoT-PCS Server to 5GC in the Network parameter configuration procedure may or may not be accepted by the network. If they are not accepted, 5GC responds accordingly and the previous values apply, or new values are provided. The new values are used by IoT-PCS Server as described in clause 5.7.3.3, when they were provided via monitoring event notifications.

5.7.3 Evaluation

The solution addresses Key Issues #3 and #4.

The solution captured in this clause except sub-clause 5.7.3.2 allows the IoT-PCS to aggregate scheduling information and monitoring requests from multiple IoT-App Servers. Aggregated monitoring in the application layer can greatly reduce the signalling burden on the exposure interfaces, i.e. T8/ N33, especially for MIoT.

In addition, the solution enables IoT Platforms to integrate network-agnostic ASs (i.e. without SCEF/NEF APIs) and to provide them with UE monitoring and scheduling features. IoT-PCS Server is also used to use application-level scheduling to derive 5GC PSM configurations on behalf of multiple IoT-App servers. At the same time, this solution does not affect the ability of IoT-App VAL/SEAL Servers to directly interact with 5GS for UE monitoring. Therefore, the UE unified traffic pattern management features, provide a viable solution recommended for the normative phase.

The UE unified traffic pattern management procedure can be considered in the normative phase as enhancement to NRM event monitoring (as an alternative to IoT-PCS functionality).The IoT-PCS server can aggregate network parameter configurations for its serviced IoT App servers. However, additional coordination from different IoT-PCS servers and other AFs (not served by IoT-PCS server) is done in the 3GPP CN.

The procedure for CP configuration in clause 5.7.3.2 addresses KI #4. However, given the SCEF/NEF capabilities to deal with multiple CP configurations per UE, such a solution does not provide significant advantages, therefore it is not proposed to be considered for the normative phase.

# 6 Overall Evaluation

## 6.1 Architecture evaluation

This clause provides an evaluation of the application architecture for enabling application capability exposure to general purpose servers or 3rd party IoT applications via IoT Platforms.

A summary of the architecture and key issues specified in this technical report are listed in Table 6.1-1. The architecture enhancements proposed in clauses 5.1 and 5.3 describe the.

Table 6.1-1 Architecture evaluation

| Architecture solution | Applicable key issues  (clause reference) | Dependency on other working groups |
| --- | --- | --- |
| 5.3 IoT Platform functional model | Supports all key issues | None |
| 5.1 Functional model for application service management | Supports key issue 2 | None |

## 6.2 Solution evaluations

### 6.2.1 General

All the key issues and solutions specified in this technical report are listed in Table 6.2.1-1. It includes the mapping of the key issues (clause 4) to the solutions and corresponding solution evaluations. It also lists the dependencies on other working groups that will need consideration during the Release 18 normative phase.

Table 6.2.1-1 Key issue and solutions

| Key issues (evaluation clause reference) | Solution | Solution  (clause reference) | Dependency on other working groups |
| --- | --- | --- | --- |
| #1: Background Data Transfer negotiation | #2: Provisioning of Edge Data Network configuration | 5.2 | None |
| #2: Application server monitoring and control of traffic | #1: Application Service Management Service | 5.1 | None |
| #5: Application Server status monitoring via CAPIF | 5.5 | None |
| #3: Key issue #3: IoT Platform PSM monitoring and configuration | #7: UE unified traffic pattern and monitoring management (except 5.7.3.2) | 5.7 | SA2 |
| #4: Key issue #4: Configuration of Communication Patterns | #7: UE unified traffic pattern and monitoring management (clause 5.7.3.2) | 5.7 | N/A |
| #5 Key issue #5: NIDD configuration | none | N/A | N/A |
| #6 Key issue #6: Device Triggering services. | Solution #4: Device triggering | 5.4 | None |

### 6.2.2 Key issue #2: Application server monitoring and control of traffic

Solution #1 (Application Service Management Service) and solution #5 (Application Server status monitoring via CAPIF) addresses the key issue#2.

Solution #1 proposes application service management using SEAL server. The solution proposes data collection procedures using request-response model and also subscribe-notify model. If required, the ASM server decides corrective action to take for VAL client or VAL server.

NOTE: For solution#1, whether configuring VAL server procedure is required or not – will be considered in normative work.

Solution #5 proposes to enhance CAPIF service to monitoring application server status. When an AS acting as an AEF publishes its service API to the CCF, the AS updates its service API status (active, inactive) at the CCF. The API invoker can discover the service API status via service API discovery procedure or be notified about the service API status change via CAPIF event exposure procedure.

Both solutions can be considered as a base for normative work.

# 7 Conclusions

## 7.1 Solution conclusions

This technical report completes the study on application architecture for enabling application capability exposure to general purpose servers or third party IoT applications via IoT Platforms, with the following considerations for normative work:

1. Definition of terms and abbreviations captured in clause 3 will be reused as needed.

2. The application architecture enhancements for enabling application capability exposure to general purpose servers or 3rd party IoT applications via IoT Platforms are summarized in clause 6.1.

3. The IoT platform deployment options detailed in clause 5.2 will be analysed in the normative phase, to determine whether a new IoT PCS SEAL service is necessary. Applicable IoT-PCS -related procedures are to be implemented in the normative phase using existing SEAL services (e.g. NRM) whenever there is relevance to other verticals or enablers.

4. The following individual solutions, corresponding to the key issues, will be considered as candidate solutions:

a. for Key issue #1 (Background Data Transfer negotiation):

i. Solution #6 (BDT configuration) as NRM functionality.

b. for Key issue #2 (Application server monitoring and control of traffic):

i. Solution #1 (Application Service Management Service) as ASM SEAL functionality; and

ii. Solution #5 (Application Server status monitoring via CAPIF) as new CAPIF functionality.

c. for Key issue #3 (IoT Platform PSM monitoring and configuration)

i. Solution #7 (UE unified traffic pattern and monitoring management) except clauses 5.7.3.2, 5.7.3.4, as NRM functionality.

d. for Key issue #4 (Configuration of Communication Patterns):

i. Clause 5.7.3.2 of Solution #7 (CP configuration procedure) addresses the key issue but does not provide enough enhancement over existing SCEF/NEF functionality. Therefore, no functionality is considered for normative phase.

e. for Key issue #5 (NIDD configuration):

i. No solutions introduced have been agreed, therefore no functionality is considered for normative phase.

Editor's Note: The conclusion for Key Issue #6 is FFS

Annex A:  
Change history

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **Tdoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
| 2021-07 | SA6#44-e | S6-211702 |  |  |  | TR Skeleton | 0.0.0 |
| 2021-07 | SA6#44-e |  |  |  |  | S6-211703, S6-211857. Editorials: renamed TBD sub-clauses 4.1, 5.1 as 4.x, 5.x, deleted change history guidance table from skeleton. Modified Annex A title from skeleton to fix table of contents formatting. | 0.1.0 |
| 2021-09 | SA6#45-e |  |  |  |  | S6-212113, S6-212121. Editorials: added one abbreviation, minor grammatical corrections | 0.2.0 |
| 2021-10 | SA6#45-bis-e |  |  |  |  | S6-212372, S6-212373, S6-212476. Editorials: added several abbreviations, changed one to avoid ambiguity, miscellaneous grammatical corrections. | 0.3.0 |
| 2021-11 | SA6#46-e |  |  |  |  | S6-212714, S6-212827 | 0.4.0 |
| 2021-11 | SA6#46-e |  |  |  |  | Editorial correction history box | 0.4.1 |
| 2022-03 | SA6#47-e |  |  |  |  | S6-220326, S6-220331, S6-220364, S6-220456. Editorials: corrected formatting for the new text; added 3GPP TS 23.682 reference; added "3GPP" to existing references for consistency. | 0.5.0 |
| 2022-05 | SA6#48-e |  |  |  |  | S6-220879, S6- 220948, Editorials: corrected a plural to singular in new text for 5.1.4.2, corrected 5.5.1.1.clause number, corrected reference to Table 5.5.1.1-1, applied formats as needed | 0.6.0 |
| 2022-06 | SA6#49-e |  |  |  |  | S6-221307, S6- 221394, S6-221395. Editorials: Corrected number for Figure 5.6.2.3-1, deleted a repeated "3GPP" in step 1 of 5.6.2.2, added a reference, removed double blanks between words. | 0.7.0 |
| 2022-07 | SA6#49-bis-e |  |  |  |  | S6-221968. Editorials: Formatting, corrected meeting numbers in Annex B. | 0.8.0 |
| 2022-09 | SA6#50-e |  |  |  |  | S6-222090, S6-222474, S6-222597, S6-222607. Editorials: Formatting, added reference [10] in clause 2 and text, corrected numbering for figures 5.7.3.3-1 and 5.7.3.4-1, changed KI #2 evaluation from being clause 6.2 to being clause 6.2.2 | 0.9.0 |
| 2022-09 | SA#97-e | SP-220946 |  |  |  | Presentation for information at SA#97-e | 1.0.0 |
| 2022-10 | SA6#51-e |  |  |  |  | S6-222974, S6-222975. Deleted old figures 5.7.2.2-1 and 5.7.3.4-1 as S6-222975 replaced them without deletion of the old. | 1.1.0 |
| 2023-03 | SA6#53 |  |  |  |  | S6-230851 | 1.2.0 |
| 2023-03 | SA#99 | SP-230274 |  |  |  | Presentation for approval at SA#99 | 2.0.0 |
| 2023-03 | SA#99 | SP-230274 |  |  |  | MCC Editorial update for publication after TSG SA approval (SA#99) | 18.0.0 |
| 2023-06 | SA#100 | SP-230705 | 0002 | 2 | D | TR 23.700-97 Conclusion update and clean-up | 18.1.0 |
| 2023-06 | SA#100 | SP-230705 | 0003 |  | D | TR 23.700-97 General clean-up | 18.1.0 |