IoT Fundamentals - Connectivity Efficiency

Version 1.0

I. Introduction

This document is intended to guide customers on the implementation of their IoT solutions to achieve greater interoperability, performance, and quality of service, and is published and distributed solely for informational purposes. It identifies best practice implementations that ensure an efficient and optimized use of the mobile network by IoT Services, including their connected devices. Another intent is to proactively prevent systemic failures that result in runaway situations, such as network or service platform overload.

The risks documented in this document are not specific to the Telefónica Germany GmbH & Co. OHG network (referred to as "**Telefónica Germany**") but apply generically to all 3GPP mobile networks worldwide. For this reason, the GSMA Association has developed and maintained the ***GSMA TS.34 IoT Device Connection Efficiency Guidelines***[1]. That guideline collects hundreds of requirements and prevention mechanisms contributed by the industry for an efficient use of 3GPP mobile networks. To better support its customers,

Telefónica Germany prioritizes delivering the highest mobile network quality to its customers and shares the most relevant content of the GSMA TS.34 specification in this document, in condensed format. Kindly refer to the GSMA TS.34 specification for additional information.

IoT Services which do not consider the aspects described in this document may encounter unwanted consequences, including:

- Reduced SIM card lifetime caused by an excessive amount of read/write cycles;
- Increased power consumption of the device due to continuous restarts, which may also shorten device lifetime; and
- Negative impacts on IoT Service performance, potentially resulting in an inability to communicate, delayed communication, degradation of service quality, or even outages.

Furthermore, overuse or inefficient communication over the mobile network not only affects IoT Devices causing the incident but also the service of other customers using the same IoT Service Platform and/or network. The impact may include, for instance:

- Regionalized issues within the mobile network such as cell congestion; and
- Capacity and performance problems within the Mobile Network Operator's core network, referred to as "signaling storms," leading to wide area service disruption.

The target audience for this document includes Telefónica Germany customers, inbound roaming customers of other mobile (virtual) network operators using the Telefónica Germany network, IoT Service Providers, IoT Device makers, IoT Device Application developers, IoT communication module vendors and radio baseband chipset vendors.

[1] https://www.gsma.com/iot/gsma-iot-device-connection-efficiency-guidelines/

This "IoT Fundamentals - Connectivity Efficiency" document consists of eight chapters, summarized as follows:

- Chapter 1: Introduction;
- Chapter 2: Definitions of Terms Used in this Document; and
- Chapter 3: Extract of the GSMA TS.34 Connection Efficiency Guidelines.

While reviewing the GSMA TS.34 guidelines in Chapter 3, please consider that the GSMA Association considers compliance to entries including the words "SHALL" or "SHALL NOT" as mandatory. In turn, the GSMA Association recommends the implementation of requirements including the words "SHOULD" or "SHOULD NOT".

Furthermore, the GSMA TS.34 specification stipulates that the following applies to stakeholders within the IoT value chain:

- IoT Service Providers SHALL ensure that their IoT Services and their IoT Device makers comply to the requirements stated within the GSMA TS.34 specification. The IoT Service Provider SHOULD reference the GSMA TS.34 specification in the supply contracts they place with their IoT Device makers.
- IoT Device makers SHALL implement the requirements contained within the GSMA TS.34 specification in the IoT Devices that they manufacture. The IoT Device maker SHOULD reference the GSMA TS.34 specification in the supply contracts they place with their IoT Application developer, IoT communication module vendor and radio baseband chipset vendor partners.
- IoT Device Application developers SHALL ensure that their IoT Device Application conforms to the requirements summarized within the GSMA TS.34 specification.
- Radio baseband chipset and communication module vendors SHALL ensure that their products conform to the requirements presented within the GSMA TS.34 specification.

Two "IoT Solution Layer" architectures are described in this document:

- IoT Device Applications which integrate the device-side Service Enablement logic (refer to *Figure 1*); this architecture is referred to as a "monolithic"; and

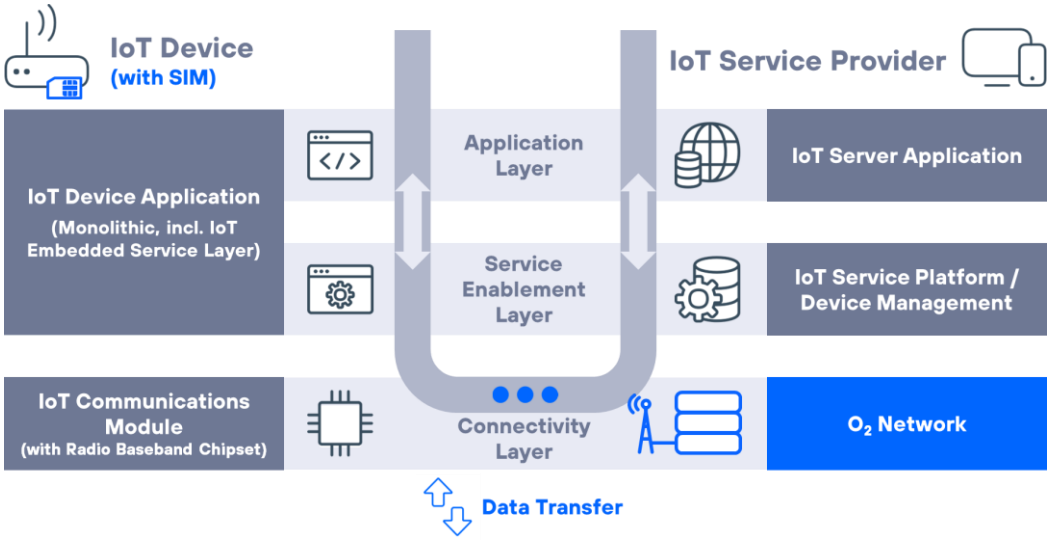

Figure 1: IoT Solution Layers, combined IoT Device Application and Service Enablement

- IoT Device Applications which separates device-side Service Enablement logic into a IoT Embedded Service Layer software, as is found, for instance in LwM2M clients, or oneM2M AE/CSEs (refer to *Figure 2*); this architecture is commonly referred to as a "tiered IoT Device Application".

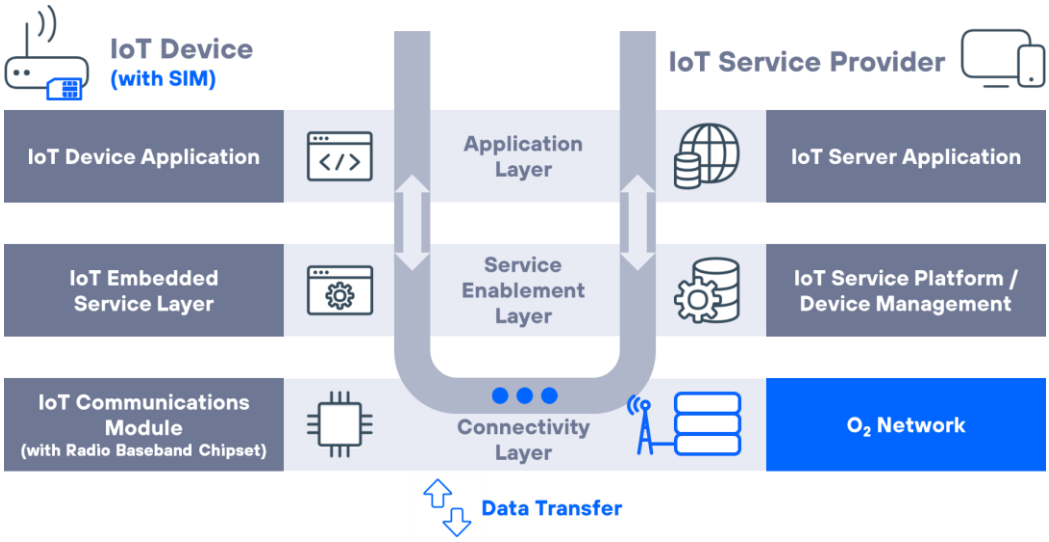

Figure 2: IoT Solution Layers, separate IoT Device Application and Service Enablement

II. Definitions of Terms Used in this Document

Note: The definitions provided below describe solution blocks of the IoT stack. They can also be visualized in Figures 1 and 2 (refer to Chapter 1).

IoT Service Provider

The provider of IoT service, using the Telefónica Germany network to provide an IoT Service. The Telefónica Germany network may be the home or roaming network of the SIMs inside the service's IoT Devices.

IoT Service

The IoT service provided by the IoT Service Provider; typically, this is an M2M vertical solution using IoT Devices such as actuators, sensors, or gateways to collect and communicate asset data or process predefined commands.

IoT Server Application

An application software that runs on a server and connected to the IoT Service Platform, exchanging data, and interacting with the IoT Device Application over said IoT Service Platform and the Telefónica Germany network.

IoT Service Platform

The platform connected to the IoT Server Application and maintaining data sessions with IoT Devices and their application. The Telefónica Germany network is used to send data between

the IoT Service Platform and the IoT Device's Communication Module. Application messages can be transmitted within the Telefónica Germany private network (including the air interface) using packet-switched, IP-based or Non-IP data bearers. Furthermore, the IoT Service Platform typically offers Service Enablement capabilities for the IoT Devices and the IoT Device Application, acting as a so-called Device Management Server. Finally, the IoT Service Platform typically offers APIs for IoT Server Applications to exchange data and interact with the IoT Device Applications over the IoT Service Platform.

Service Enablement

As defined by standardization bodies, this may include enabling functions such as device discovery, registration, management (individual devices or a group thereof), application and service management, communication management, data management, service charging and accounting, as well as subscription and notification. All IoT services support most of these enablement functions, which are coordinated between the IoT Devices and the IoT Service Platform inside of a logical Service Layer sitting between the Application Layer and the Connectivity Layer. Device-side, this may be an OMA LwM2M client or operating system. The TS.34 specification refers to the Service Enablement Layer on the IoT Device as the "IoT Embedded Service Layer". On the IoT Service Platform, a LwM2M server or IoT Connector may handle such operations. Service Enablement may be provided by the Mobile Network Operator, the IoT Service Platform provider, the IoT Service Provider, or the IoT Device maker.

IoT Device

A device, whose main function is to allow assets to be accessed, sensed and/or controlled remotely, primarily across existing mobile network infrastructures, such as the Telefónica Germany network. An IoT Device consists of multiple hardware and software components. A microcontroller (MCU) is typically used to host the IoT Device Application and Service Enablement logic. The IoT communication module contains a radio baseband chipset with 3GPP protocol stack which sets up, maintains, and tears down 3GPP connectivity with the mobile network, as well as a radio frequency front end (RFFE). A SIM, battery, GNSS solution, User Interface, antenna, sensory chip, or actuator are among the remaining components making up the IoT Device's architecture.

IoT Device Application

The application software component running on the IoT Device MCU, controlling the IoT communication module via AT commands, and interacting with an IoT Service Platform. The IoT Device Application may incorporate Service Enablement logic into a monolithic software package; otherwise, a tiered architecture may be used, which separates the software into an Application Layer and Service Enablement Layer.

III. Extract of the GSMA TS.34 Connection Efficiency Guidelines

Communication of the IoT Service (General)

These entries are defined by the GSMA Association within the **GSMA TS.34 specification**, the "*IoT Device Connection Efficiency Guidelines*."

Avoid Synchronized Network Access

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL avoid all synchronized behavior when performing device activation and communicating, employing instead a randomized pattern for communication (e.g., over period ranging from a few minutes to several hours, or days, as deemed tolerable by the IoT Service's use case). This is especially critical in the event of a service recovery upon removing a failure in the network, since the increased and concentrated signaling and traffic may further delay the network recovery and affect the performance of the IoT Service.

Furthermore, the IoT Service Platform which communicates to multiple IoT Devices SHALL avoid synchronized behavior and employ a randomized pattern for accessing the IoT Devices within the IoT Service Platform's domain.

The triggering of data transmissions, the rebooting of the IoT Device or any components therein (such as the IoT communication module or radio baseband chipset), or the sending of device management commands (including, but not limited to registrations and firmware updates) SHALL NOT be synchronized. (Ref: TS.34_4.0_REQ_003, TS.34_4.2_REQ_003, and TS.34_6.0_REQ_001)

Maximum Data Volume per Month

The monolithic IoT Device Application (or IoT Device Application and underlying IoT Embedded Service Layer) SHALL ensure that the amount of application data transmitted over the Telefónica Germany network complies with the maximum volume of data per single IoT Device, as specified in the Annex of this document (note: tariff-specific exceptions may apply). The maximum payload size per message allowed on average can be calculated as follows: (monthly data allowance / number of days per month) / maximum number of messages per day. Please note that Mobile IoT technologies (NB-IoT and LTE-M) are particularly impacted by non-compliant behavior. (Ref: TS.34_4.0_REQ_030, TS.34_4.1_REQ_005, and TS.34_4.2_REQ_030)

Monitoring of Data Volume Consumption

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL monitor the volume of data it sends and receives during a set period. If the volume of data exceeds on average the maximum volume of data per single IoT Device, as specified in the Annex of this document (note: tariff-specific exceptions may apply), the IoT Device Application sends a report to the IoT Service Platform and stops requesting mobile network connectivity until the necessary time has expired to balance the excess communication. Alternatively, it can adapt the application reporting period to send messages

less frequently, or reduce the size of subsequent messages, at least until the excess volume of communicated data is balanced out. Generally, the IoT Device Application SHALL compress data as much as possible and use optimized codification to transmit it. (Ref: TS.34_4.0_REQ_013 and TS.34_4.2_REQ_013)

Maximum Application Messages per Day

The monolithic IoT Device Application (or IoT Device Application and underlying IoT Embedded Service Layer) SHALL ensure that the application reporting period never exceeds on average Telefónica Germany's daily maximum number of messages per single IoT Device, as specified in the Annex of this document (note: tariff-specific exceptions may apply). Please note that Mobile IoT technologies (NB-IoT and LTE-M) are particularly impacted by non-compliant behavior. (Ref: TS.34_4.0_REQ_002.1, TS.34_4.0_REQ_030, TS.34_4.1_REQ_005, TS.34_4.2_REQ_002.1, and TS.34_4.2_REQ_030)

Monitoring of the Amount of Application Messages

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL monitor the number of application messages it attempts to send or receive over a period. If the number of messages exceeds on average the daily maximum number of messages per single IoT Device, as specified in the Annex of this document (note: tariff-specific exceptions may apply), the IoT Device Application sends a report to the IoT Service Platform and stops requesting mobile network connectivity until the necessary time has expired. Alternatively, it can adapt the application reporting period to send application messages less frequently, at least until the excess communication has been balanced out. (Ref: TS.34_4.0_REQ_012 and TS.34_4.2_REQ_012)

"Always-on" Connectivity with Frequent Communication

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL use an "always-on" connectivity mechanism (persistent PDP/PDN connection) in the case that data needs to be sent very frequently, instead of activating and deactivating network connections (i.e., performing a network attach) each time. (TS.34_4.0_REQ_001 and TS.34_4.2_REQ_001)

Use of a Private APN

If the IoT Service communicates with IoT Devices over the Telefónica Germany network, it SHOULD use a Private APN for direct addressing of IoT Devices with dedicated, static IP-address range. IoT Service Providers should take note of the fact that using Public APN may lead to a situation whereby IoT Devices cannot be reached on the Downlink after the TCP/IP or UDP/IP session expires. IoT Devices can however initiate Uplink communication at any point, setting up a new session, if necessary. (Ref: TS.34_4.0_REQ_031 and TS.34_4.2_REQ_031)

Triggering Devices only if Attached

The IoT Service Platform SHALL be aware of the IoT Device's state and only send "wake up" triggers whenever the IoT Device is known to be attached to the network. (Ref: TS.34_6.0_REQ_004)

Handling of "Keep Alive" Messages (except NB-IoT)

If the communication with the IoT Device within the home or roaming networks is TCP/IP-based, it will require the use of TCP "keep alive" polling messages to main the TCP session active, whenever the period of communication is longer than the TCP session timer. If the IoT Device uses UDP/IP protocol instead, it must either implement the shorter UDP session timer value or reestablish a new UDP session each time it wants to communicate on the Uplink.

In such cases, the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL automatically detect the session-specific timers and/or mobile network firewall NAT timers, such the TCP_IDLE value or UDP_IDLE value. This is achieved by increasing the polling interval dynamically until a mobile network timeout occurs, and then operating just below the timeout value. Please note that many roaming networks use a NAT timeout value below 30 minutes; TCP/IP session timers may be longer, up to 2 hours, for instance.

Fixed polling intervals ideally SHOULD NOT be used by the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device), as polling interval values may change or dynamically adapt with mobile network loading. If this cannot be avoided, the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD allow remote and/or local configuration of the interval of "keep alive" messages. (Ref: TS.34_4.0_REQ_006, TS.34_4.0_REQ_007, TS.34_4.2_REQ_006, and TS.34_4.2_REQ_007)

Prioritization of Application Data

The monolithic IoT Device Application (or IoT Device Application and underlying IoT Embedded Service Layer) SHOULD classify the priority (importance and urgency) of the data before deciding whether to send messages over the network, with the aim of minimizing congestion on the network. For example, it can distinguish between data that requires instantaneous transmission and delay-tolerant data that could be aggregated and/or sent during "off-peak" hours (e.g., during early morning hours). (Ref: TS.34_4.0_REQ_018, TS.34_4.1_REQ_003, and TS.34_4.2_REQ_018)

Off-Peak Communication

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD be designed to ensure the application's communication activity is not concentrated during periods of high network utilization (e.g., during early morning hours). Please note that "peak hours" may vary between applications. (Ref: TS.34_4.0_REQ_016 and TS.34_4.2_REQ_016)

Localized Heavy Communication

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL minimize any risk of geographical network loading problems, for instance caused by the triggering of operations in a small geographic area producing high network traffic, for example, firmware updates delivered at once to all devices in a city. Furthermore, the IoT Service SHALL tolerate any geographical network loading problems that may still occur due to external factors out of its control. (Ref: TS.34_4.0_REQ_017 and TS.34_4.2_REQ_017)

IoT Service Coordination

If the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) communicates with multiple IoT Server Applications using the same IoT communication module or radio baseband chipset, it SHOULD coordinate the payload transmission of each IoT Service in a way which makes efficient use of the network. (Ref: TS.34_4.0_REQ_002 and TS.34_4.2_REQ_002)

Authentication and Encryption of Communication

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD sent all application messages end-to-end encrypted and authenticate the IoT Service Platform prior to data communication. Likewise, the IoT Service Platform SHOULD authenticate the IoT Device prior to data communication. The strength of authentication used is to be dimension to be appropriate for the IoT Service.

These precautions ensure that IoT Devices and/or the IoT Service Platform are not compromised, potentially even leading to a Distributed-Denial-of-Service (DDoS) attack on the mobile network. (Ref: TS.34_4.0_REQ_020, TS.34_4.0_REQ_021, TS.34_4.2_REQ_020, TS.34_4.2_REQ_021, and TS.34_6.0_REQ_005)

Communication of the IoT Service (NB-IoT)

These entries are defined by the GSMA Association within the **GSMA TS.34 specification**, the "*IoT Device Connection Efficiency Guidelines.*"

Avoidance of MQTT or HTTP when using NB-IoT

If the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) communicates over a NB-IoT bearer, it SHOULD NOT use MQTT or HTTP messaging/management protocols, as their TCP/IP session-based connections are not optimized for NB-IoT. Connectionless protocols such as CoAP or MQTT-SN are ideal, as these allow for the underlying UDP/IP bearer to be removed. Similarly, unlike the case of TLS over TCP/IP, existing DTLS sessions can be generally continued over successive UDP/IP sessions. (Ref: TS.34_4.0_REQ_031 and TS.34_4.2_REQ_031)

Avoidance of "Keep Alive" Messages when using NB-IoT

If the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) communicates over a NB-IoT bearer, it SHOULD NOT implement TCP or UDP "keep alive" messages on the home or roaming network. Please consider the impact on battery life of recurring TCP/IP, TLS, MQTT, or HTTP session negotiation when communicating. (Ref: TS.34_4.0_REQ_006.1 and TS.34_4.2_REQ_006.1)

Error Handling & Recovery

These entries are defined by the GSMA Association within the **GSMA TS.34 specification**, the "*IoT Device Connection Efficiency Guidelines*."

Behavior when SIM Subscription is Temporarily Inactive

If the SIM subscription associated with an IoT Device is to be temporarily placed in an inactive state (i.e., the subscription is to be disabled for a fixed period), the IoT Service Provider SHALL first ensure that the IoT Device is temporarily disabled to restrict the device from trying to register to the network once the SIM is disabled. (Ref: TS.34_6.0_REQ_002)

Behavior when SIM Subscription is Permanently Disabled

Before the SIM subscription associated with an IoT Device is changed to a permanently terminated state, the IoT Service Provider SHALL ensure that the IoT Device is permanently disabled to stop the device from trying to register to the network once the SIM is permanently disabled.

The IoT Service Provider SHOULD carefully consider permanently terminating IoT Devices that are not easily serviceable, as it may require manual intervention (i.e., a service call) to re-enable the IoT Devices. (Ref: TS.34_6.0_REQ_002)

Behavior when Battery Power is Low or Power Failure Occurs

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD send a notification to the IoT Service Platform with relevant information when there is an unexpected battery problem. If a power failure occurs, IoT Devices may notify the IoT Service Platform in a non-synchronous manner, although this is to be implemented in a specific manner when communicating over NB-IoT (please refer to the corresponding requirement in the previous section). (Ref: TS.34_4.0_REQ_014 and TS.34_4.2_REQ_014)

Behavior when IoT Devices do not Respond to SMS Triggers

If the IoT Service Platform uses SMS triggers to "wake up" IoT Devices, it SHALL avoid sending multiple SMS triggers when no response is received within a certain period. (Ref: TS.34_6.0_REQ_003)

Behavior when Communication Fails on Home and Roaming Networks

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) is always be prepared to handle situations when communication requests fail on the home network or in roaming:

- It SHALL try to re-establish higher layer connectivity (e.g., VPN tunnels, SSH sessions);
- It SHALL diagnose if the communication issues to the server are caused by higher layer communication issues (within the MQTT(-SN), HTTP, CoAP, TCP/IP, or UDP/IP protocol);
- It SHALL try to re-establish the PDN connectivity or PDP context;
- It SHALL try to re-attach to the mobile network;
- It SHALL perform a network selection procedure;
- As a last resort, it SHALL reset in a randomized manner the IoT communication module or radio baseband chipset, which result in a re-establish of the RRC Connection. Retry schemas SHALL be exponentially delayed.
- Higher layers mechanisms SHALL thereafter attempt to re-establish the connection with the IoT Service Platform.

The monolithic IoT Device Application (or IoT Device Application and underlying IoT Embedded Service Layer) SHALL NOT frequently initiate a reboot of the IoT communication module or radio baseband chipset. It SHALL instead retry connection requests to the IoT Service Platform with an increasing back-off period. Such retry-mechanisms can vary and will depend on the importance and volume of downloaded data. Possible solutions can be:

- Simple counting of failed attempts since the data connection was first established (often, the easiest solution); or
- Monitoring the number of failed attempts within a certain period. For example, if the data connection is lost more than five times within an hour, then the request can be suspended. This can be a more reliable technique to avoid short but regular connection problems, such as when an IoT Device is moving away from one network cell to another. The data connection can be lost when the device switches between cells, but when the cell is providing good coverage; the request can be processed successfully.

Depending upon the IoT Service, no communication request by the IoT Device Application (or IoT Device Application and underlying IoT Embedded Service Layer) SHOULD ever be retried indefinitely – the request eventually times-out and is abandoned. (Ref: TS.34_4.0_REQ_011, TS.34_4.0_REQ_029, TS.34_4.1_REQ_002, TS.34_4.2_REQ_011, and TS.34_4.2_REQ_029)

Behavior when GNSS Coverage is Lost

When the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) detects that GNSS (e.g., GPS, GLONASS, BeiDou, or Galileo) coverage is lost, and the GNSS receiver is hosted on the IoT communication module or radio baseband chipset, it SHALL NOT reboot the IoT Device, IoT communication module, or radio baseband chipset.

Instead, the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL retry scanning to acquire mobile GNSS coverage with an increasing

back-off period. If that fails, it SHOULD perform diagnostics and send an alert to the IoT Server Application.

If the GNSS receiver is not hosted on the IoT communication module or radio baseband chipset, the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD reboot the affected hardware component (e.g., GNSS chip), perform diagnostics, and send an alert to the IoT Server Application. (Ref: TS.34_4.0_REQ_034, TS.34_4.0_REQ_035, TS.34_4.2_REQ_034, and TS.34_4.2_REQ_035)

Behavior when LAN Connectivity is Lost

When the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) detects that LAN connectivity with peripheral devices is lost, and the LAN connectivity function is hosted on the IoT communication module or radio baseband chipset, it SHALL NOT reboot the IoT Device, the IoT communication module, or radio baseband chipset. The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL retry scanning to acquire mobile LAN connectivity with an increasing back-off period. If that fails, it SHOULD perform diagnostics and send an alert to the IoT Server Application.

If the LAN connectivity function is not hosted on the IoT communication module or radio baseband chipset, the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD perform diagnostics, reboot the affected hardware component (e.g., Wi-Fi transceiver), and send an alert to the IoT Server Application. (Ref: TS.34_4.0_REQ_036, TS.34_4.0_REQ_037, TS.34_4.2_REQ_036, and TS.34_4.2_REQ_037)

Behavior when Sensors or Actuators Malfunction or are Triggered

When in-built sensors or actuators malfunction or are triggered, the Monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL NOT reboot the IoT Device, IoT communication module, or radio baseband chipset. It SHOULD perform diagnostics, reboot the affected hardware component (e.g., sensor or actuator component), and send an alert to the IoT Server Application. (Ref: TS.34_4.0_REQ_038, TS.34_4.0_REQ_039, TS.34_4.2_REQ_038, and TS.34_4.2_REQ_039)

Behavior when Device Memory Full

When the IoT Device's memory is full, for example due to the amount of collected data or an unwanted memory leak, the Monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL NOT reboot the IoT Device, IoT communication module, or radio baseband chipset. It SHOULD perform diagnostics, reboot the affected hardware component (e.g., on-board memory), and send an alert to the IoT Server Application. (Ref: TS.34_4.0_REQ_032, TS.34_4.0_REQ_033, TS.34_4.2_REQ_032, and TS.34_4.2_REQ_033)

IoT Device Requirements

These entries are defined by the GSMA Association within the **GSMA TS.34 specification**, the "*IoT Device Connection Efficiency Guidelines.*"

Radio Policy Manager

For large deployments of IoT Devices (e.g., >2,000 units within the same mobile network), the IoT Service Provider SHOULD employ IoT communication module in their IoT Devices whose radio baseband chipset supports the Radio Policy Manager (RPM) feature, as defined in the GSMA TS.34 specification, Section 8. This feature should be turned on for the Telefónica Germany network. (Ref: TS.34_5.2_REQ_001)

Avoidance of Mechanisms that Frequently Reboot Modem

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL never implement a mechanism that triggers the frequent reboot of the IoT communication module or radio baseband chipset. (Ref: TS.34_4.0_REQ_019 and TS.34_4.2_REQ_019)

Low Power Mode

When an IoT Device Application does not need to perform regular data transmissions and it can tolerate some latency for its IoT Service, the monolithic IoT Device Application (or IoT Device Application and underlying IoT Embedded Service Layer) SHOULD implement a "low power" operating mode where the IoT Device and its IoT communication module or radio baseband chipset are effectively powered down between data transmissions. This reduces power consumption of the IoT Device further and minimizes network signaling. (Ref: TS.34_4.0_REQ_020, TS.34_4.1_REQ_004, and TS.34_4.2_REQ_020)

Reset to Factory Settings

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD support a "reset to factory settings" via remote and local connection. (Ref: TS.34_4.0_REQ_024 and TS.34_4.2_REQ_024)

Reselection between 3GPP and non-3GPP Access

If the IoT Device supports more than one family of communications access technology (for example, 3GPP technologies and Wireless LAN) the monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD implement a protection mechanism to prevent a frequent "Ping-Pong" between these different families of communications access technologies. Introducing hysteresis or a randomized timer may be practical solutions. (Ref: TS.34_4.0_REQ_026, TS.34_4.0_REQ_027, TS.34_4.2_REQ_026, and TS.34_4.2_REQ_027)

Monitoring of Mobile Network Speed and Connection Quality

If data speed and latency is critical to the IoT Service, the monolithic IoT Device Application (or IoT Device Application and underlying IoT Embedded Service Layer) SHOULD constantly monitor mobile network speed and connection quality to request the appropriate quality of content from the IoT Service Platform. (Ref: TS.34_4.0_REQ_010, TS.34_4.1_REQ_001, and TS.34_4.2_REQ_010)

Adaption to Mobile Network Capabilities, Data Speed, and Latency

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL be capable of adapting to changes in mobile network feature capability and service exposure. Furthermore, it is designed to cope with variations in the mobile network's available throughput, data speed, and latency, especially when switching between different access bearers (i.e., NB-IoT, LTE-M, 2G, 3G, LTE, 5G-NSA). (Ref: TS.34_4.0_REQ_008, TS.34_4.0_REQ_009, TS.34_4.2_REQ_008, and TS.34_4.2_REQ_009)

IPv4/v6 Dual Stack Support

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHALL support IPv4/v6 dual stack (PDN Type = IPv4v6) so that it can properly roam onto mobile networks having support for either IPv4 only, IPv6 only, or dual stack only. (Ref: TS.34_5.3_REQ_006)

Device Time Resynchronization

The monolithic IoT Device Application (or the separated IoT Embedded Service Layer on the IoT Device) SHOULD support time resynchronization via remote and local connection. (Ref: TS.34_4.0_REQ_025 and TS.34_4.2_REQ_025)

IV. Release History

Date	Version	Author
04.09.2023	EN v1.0	Digital Services – IoT, Miguel Rodriguez