Telstra Wireless Application Development Guidelines

Version 9 Issue 1

Date: December 2018

TABLE OF CONTENTS

1. AIM

The goal of this document is to:

1. educate developers about Telstra's mobile network;
2. encourage developers to produce network "impact friendly" applications i.e. applications that don't impose an unnecessary strain on the mobile network; and
3. encourage more efficient use of the Telstra mobile network by developers;
   in order to enhance customers' experience of the Telstra mobile network.

It is important to understand how to best develop for our mobile network. Your design choices can have significant effects on how well your application works, as well as potentially give your product or solution a competitive edge and improve its performance, by improving things like its feature set and longevity.

2. SCOPE

These guidelines cover application development for smartphones, tablets and laptops using Telstra's mobile network (3G and 4G technologies). They also cover Machine to Machine (M2M) communication applications including embedded modules, devices integrating embedded modules and the related controlling software behaviour.

For those requiring additional detail on each topic, links are provided to explore further.

The scope of the document excludes the details of how to code applications and associated backend servers / cloud. It does not cover details of user interface visual design or programming languages.

This document draws from a wealth of existing industry associations, OS platform, wireless operator and other developer guidelines.

3. DISCLAIMER

These guidelines are general in nature and apply to the most common use-case scenarios. You should consider your own specific requirements where necessary. Reliance upon any representations made or information contained in this document is at your own risk.

Telstra will seek to update this document periodically. Please check the Telstra.com website for updated version. If you are building solutions specifically for Telstra or other customers seek guidance to ensure your solution delivers the product's desired performance.

4. INTRODUCTION

4.1. Why are the guidelines required?

The ongoing high performance and reliability of the Telstra mobile network relies heavily on the devices and applications that utilise it doing so in a manner that is both efficient and within the expected standards of the industry. These guidelines describe many methods that will help developers design applications in a way that doesn't impose unnecessary strain on the network. This is good for everyone: Telstra, developers, and customers.

Wireless Network Constraints
Wireless networks have some constraints that developers need to consider when developing their applications such as:

Power Source – Wireless devices typically don't have access to AC and rely on batteries whereas home/fixed network routers are AC powered.

Reliability/Robustness of network connection – Wireless networks deliver varying throughputs based on variables such as location and environmental factors.

Network Technology Speed and Latency Variability – Cellular throughputs vary considerably based on location, terrain, coverage, radio interference, geography, and technology. The latency of 4G is several times faster than that of 3G for example.

Capacity constraints – Apps must be designed to work well, and be responsive even when networks are heavily loaded.

4.2. Benefits of the guidelines?

Following the principles in these guidelines will provide benefits for the developer, app user and the network operator. Such benefits include:

- improved battery life for devices;
- lower data costs – if an application uses the network efficiently, it will use less data, resulting in lower data costs for the app user in connection with that application
- more responsive apps
- increased application / device longevity
- more robust / resilient applications
- reduced network signalling
- improved user security and privacy

Section 7 below (Application Development Techniques) sets out the techniques which developers can use to deliver the above benefits.

5. AN OVERVIEW OF THE TELSTRA NETWORK

5.1. Technologies and Features

Mobile networks evolve rapidly. Developers need to ensure they future proof their design and plan to take advantage of new features as they become available.

The following table presents the technologies and features available on the Telstra Network (as of December 2018) along with future technology trends. Included alongside are relevant considerations for designing your wireless application.

Technology/Feature	Present	Future	Considerations
Frequency Band Requirements			
3G Frequency Bands Uses HSPA and HSPA+ technologies	• 850 MHz (B5)	• 3G currently has a very large footprint, little network expansion is planned.	• No significant 3G design feature development is planned.
4G Frequency Bands Uses LTE technology	• 1800 MHz (B3) for coverage • 700 MHz (B28) for coverage • 2600 MHz (B7) for capacity and in-building coverage • Small pockets of 2100 MHz (B1) for capacity	• 900 MHz (B8) and 850 MHz (B5) may be supported in selected areas	• Note that the 700 MHz bands (B12, 13, 14, 17) used in the US are not compatible with the Australian 700 MHz band (B28) • Lower frequency bands work better in rural areas. • Refer link beneath table for coverage map **
LTE Carrier Aggregation (CA) Combines 2 or more bands together to allow greater throughput	• 2, 3, 4 and 5 Band CA Supported combinations: • 2CA: 7 + 7, 3 + 7, 3 + 28, 7 + 28, 1 + 3, 1 + 7, 1 +28, 3 + 3, 3 + 8 • 3CA: 3 + 7 + 7, 7 + 7 + 28, 3 + 7 + 28, 3 + 3 + 7, 3 + 3 + 28, 1 + 7 + 7, 1 + 3 + 7, 1 + 3 + 3 • 4CA: 3 + 7 + 7 +28, 3 + 3 + 7 + 7, 3 + 3 + 7 + 28, 1+ 7 + 7 + 28, 1 + 3 + 7 + 7, 1 + 3 + 3 +7, 1 + 3 + 7 + 28 • 5CA: 1 + 3 + 3 + 7 + 7, 1 + 3 + 3 + 7 + 28, 1 + 3 + 7 + 7 + 28, 3 + 3 + 7 + 7 + 28	• 6 Band CA: 1 + 3 + 3 + 7 + 7 + 28	• CA offers higher data rates that are suitable for large file downloads and video streaming applications • CA is limited to certain areas of the network.
MIMO Stands for multiple-in multiple-out. Using multiple antennas the up and downlink capacity of a radio transmission can be multiplied	In downlink network supports: • 2x2 on all bands • 4x2 on B1, B3 and B7 • 4x4 on B1, B3 and B7	MIMO support is in plan to be introduced on the following bands: • 4x2 on B28 •	

LTE-M/NB-IoT Low throughput and low power cellular technology for IoT solutions	• 700 MHz (B28) for Cat M1 and NB1	• 1800 MHz (B3) for Cat M1	
IPv6 The most recent version of the internet addressing protocol. It has a far larger number of unique addresses than IPv4	• IPv4 • Telstra has deployed Single Stack and Dual Stack IPv6 capability on handheld and tablet devices.	• IPv4v6 Dual Stack (Deployed on telstra.internet and telstra.wap APNs) • IPv6 Single Stack for Telstra.wap APN • Single Stack and Dual Stack IPv6 is expected to be rolled out to IOT devices in 2019.	• APNIC no longer has any IPv4 addresses and the industry is moving to IPv6

* B = 3GPP Band (e.g. B28 = Band 28, 700 MHz)
**Telstra LTE coverage footprint: https://www.telstra.com.au/coverage-networks/our-coverage

5.2. LTE Device Category Network Support

"Device categories" refer to different levels of maximum theoretical data rates supported by devices. Devices will have a maximum category that they can support. A throughput speed within that range can only be achieved if the network supports the same maximum category and is also dependant on various variables including signal strength & quality, number of concurrent users and other factors.

The following table lists the most common LTE categories currently supported by the Telstra Network as well as plans for the support of further categories in the future. This is not an exhaustive list. If your device supports a category not specifically mentioned in this table please contact Telstra so we can help you determine if that device category is supported on our network.

The new categories (Cat 1, M1, NB1) have been designed specifically for IoT (refer to section 5.4). A key difference between these and traditional LTE categories is that capability has been traded for lower power consumption, reduced device complexity and lower cost.

A summary of the key LTE categories supported on the Telstra network:

Category	Downlink (max)	Uplink (max)	3GPP Release*	Telstra Support
NB1	100 kbps	100 kbps	Rel. 13	Deployed in Network
M1	1 Mbps	1 Mbps	Rel. 13	Deployed in Network
1	10 Mbps	5 Mbps	Rel. 8	Deployed in Network
3	100 Mbps	50 Mbps	Rel. 8	Deployed in Network
4	150 Mbps	50 Mbps	Rel. 8	Deployed in Network
6	300 Mbps	50 Mbps	Rel. 10	Deployed in Network
9	450 Mbps	50 Mbps	Rel. 11	Deployed in Network
11	600 Mbps	50 Mbps	Rel. 11	Deployed in Network
13	390Mbps	N/A**	Rel. 12	Deployed in Network
16	1050 Mbps	N/A**	Rel. 12	Deployed in Network
13	N/A**	150Mbps	Rel. 12	Deployed in Network
18	1200 Mbps	N/A**	Rel. 13	Deployed in Network
19	1650Mbps	N/A**	Rel. 13	From 2019
20	2GBps	N/A**	Rel. 14	From 2019

*3GPP release refers to the release version of industry standard covering the device category.

** Note that as of 3GPP Release 12 the uplink and downlink category speeds have been split so that they can be paired in different combinations. What this means is that just because an area supports a particular categories downlink speed it doesn't necessarily support the same category uplink speed. Therefore UL speed is dependent on the UL category and is independent of the DL category.

5.3. 3G Category Network Support

Similar to the previous section, this following table lists instead the most commonly used 3G device categories as supported by the Telstra network. It should be noted that downlink (HSDPA) and uplink (HSUPA) categories are separate and can be supported in different combinations. As with the LTE table above, this is not an exhaustive list. If your device supports a category not specifically mentioned in this table please contact Telstra so we can help you determine if that device category is supported on our network.

Category	HSDPA Downlink (max)	HSUPA Uplink (max)	3GPP Release	Telstra Support
5	N/A	2 Mbps	Rel. 6	Deployed in Network
6	N/A	5.76 Mbps	Rel. 6	Deployed in Network
6	3.6 Mbps	N/A	Rel. 5	Deployed in Network
8	7.2 Mbps	N/A	Rel. 5	Deployed in Network
9	10.2 Mbps	N/A	Rel. 5	Deployed in Network
10	14.4 Mbps	N/A	Rel. 5	Deployed in Network
14	21.1 Mbps	N/A	Rel. 7	Deployed in Network
24	42.2 Mbps	N/A	Rel. 8	Deployed in Network

*R99 category is not permitted for new application devices on Telstra Network.

5.4. Internet of Things (IoT)

5.4.1. Introduction

The Internet of Things is an emerging technology trend based heavily on the M2M (Machine to Machine communication) market. IoT systems will typically comprise many (typically hundreds and even thousands) low cost and low power devices. They will tend to serve a singular function that requires very little data transfer rates and data usage, unlike the current popular mobile devices like smartphones, tablets, and mobile broadband modems.

For this reason there have been new LTE device categories created to address the Internet of Things. These new categories focus on addressing the previously unmet needs of these IoT solutions by providing low throughputs that require less energy consumption and extend the effective coverage area.

Telstra strongly recommends that developers use LTE technology for their IoT solutions. LTE supports the newly developed IoT specific device categories (Cat 1, Cat M1, and Cat NB1) which due to their simplicity have the advantage of being lower cost to purchase and develop for, as well as having significantly lower power requirements than higher category devices.

5.4.2. IoT Typical Usage by LTE Category

The following table is intended to help developers determine the most suitable device category for their IoT solution. You should choose a suitable device from the most appropriate device category to support the characteristics of your specific application.

LTE Category	> = 13	> = 1	1, M1	1, M1, NB1	M1, NB1
Device Data Usage (UL+DL bytes per day)	> 10 MB	> 1 MB	0.1 – 1 MB	< 0.1 MB (100 kB)	< 0.01 MB (10 kB)
LTE Bands Required	Triband (B28, B3, B7)	Dual Band (B28, B3)	Dual Band (B28, B3)	Dual Band (B28, B3)	Single Band (B28)
Typical Use Cases	• Video streaming • Connected Car	• Connected Home • Wearables	• Logistics • Remote Healthcare	• Smart City • Energy Metering	• Environmental Monitoring • Industrial Sensors

5.4.3. Telstra M2M Control Centre

The Telstra M2M Control Centre is a SIM and connectivity management platform. It allows our M2M business customers to easily manage their own M2M or IoT solutions. The Control Centre provides control and management of SIM deployments, device connectivity status, control of rate plans and automated diagnostic alerts.

5.4.4. Network Features

In support of IoT devices, several features of interest have been introduced.

5.4.4.1. Power Saving Mode (PSM)

Power Saving Mode is a feature designed for IoT devices to assist them to conserve battery power and potentially achieve a 10 year battery life.

Whilst it has always been possible for a device application to turn its radio module off to conserve battery power, the device would subsequently have to reattach to the network when the radio module was turned back on. The reattach procedure consuming a small but finite amount of energy. The cumulative energy consumption of reattaches can become significant over the life of a device and battery life could be extended if this procedure could be avoided.

When a device initiates PSM with the network, the network retains state information and a reattach procedure is not required if the relevant device awakes and sends data before the expiration of the time interval it agreed with the network.

For a monitoring application, the radio module in a device might be configured by an application to enable PSM, negotiate a 24 hour time interval with the network and provide a daily status update to a centralised monitoring point. If the device's monitoring application were to detect an alarm condition, irrespective of any agreed sleep interval, the application could wake the radio module instantly and send vital monitoring information to the centralised monitoring point without the need to execute a reattach procedure.

In a similar manner to a radio module that has been powered off, a radio module with PSM enabled cannot be contacted by the network whilst it is asleep. The inability to be contacted whilst asleep may preclude the use of PSM for some applications.

5.4.4.2. Extended Discontinuous Reception (eDRX)

Extended Discontinuous Reception is an extension of an existing LTE feature which can be used by IoT devices to reduce power consumption. eDRX can be used without PSM or in conjunction with PSM to obtain additional power savings.

Today, many smartphones use discontinuous reception (DRX) to extend battery life between recharges. By momentarily switching off the receive section of the radio module for a fraction of a second, the smartphone is able to save power. The smartphone cannot be contacted by the network whilst it is not listening but if the period of time is kept to a brief moment, the smartphone user will not experience degradation of service. E.g. If called, the smartphone might ring a fraction of a second later than if DRX was not enabled.

eDRX allows the time interval during which a device is not listening to the network to be greatly extended. For an IoT application it might be quite acceptable for the device to not be reachable for a few seconds or longer.

Whilst not providing the same levels of power reduction as PSM, for some applications eDRX may provide a good comprise between device reachability and power consumption.

5.4.4.3. Enhanced Coverage

Some IoT applications require devices to be positioned in areas that are not readily accessible by radio coverage. For example, underground parking garages and in ground pits. The 3GPP Enhanced Coverage feature has the potential to increase the depth of radio coverage to enable IOT devices to be placed and operate in locations that would otherwise not be possible.

The 3GPP Enhanced Coverage feature, also called Coverage Extension, increases the power levels of signalling channels together with the ability to repeat transmissions. Through repeated transmission the ability of receivers to correctly resolve the message sent is improved.

The trade-off is that repeating signal transmissions consumes additional power and the time between battery recharge or replacement may be reduced.

5.5. Telstra 2G and 3G Technology Closure

Telstra 2G network was shut down on the 1st of December 2016. Existing 2G-only products (handsets, M2M/IoT applications, etc) ceased to have mobile connectivity from this date. If they have not already, these applications will need to migrate across to devices that can support 4G technologies.

Current coverage of 3G technology will remain in service at least until early 2020. Limited closure activities may commence in 2020 and are expected to be well progressed by 2022. Telstra will provide a reasonable notice period of any shutdown activities in specific geographic areas that would impact end-customers.

To provide M2M/IoT devices with the longest possible support, Telstra recommends using the most recent technology available. Telstra considers LTE (4G) the technology of choice for M2M/IoT applications because it:

- Has a longer life expectancy
- Can support more devices per unit area
- Supports new M2M/IoT specific device categories (see section 5.4. above) that have better power consumption, lower cost and in some cases better coverage characteristics

5.6. Telstra Certified Devices

Telstra believes in customer first and puts the customer at the centre of everything we do. The Telstra certification and testing program is designed to make sure that your device is compatible with the network.

Telstra certification will help ensure that the device meets requirements on:

- Frequency band support
- Data throughput performance across all networks including 3G, 4G and Wi-Fi if applicable
- Network interoperability under stationary and mobile conditions
- Device performance under congested network environment
- Network reacquisition and retry algorithms
- Data and device stability
- Radio compliance
- Antenna sensitivity
- Over the air firmware and application upgrades
- IPv6 functionality
- Battery life for low power devices

A list of Telstra certified M2M/IoT modules and devices can be found at:
https://www.telstra.com.au/content/dam/tcom/business-enterprise/machine-to-machine/pdf/telstra-m2m-certified-devices-modules.pdf

6. GENERAL BEST PRACTICE APP DEVELOPMENT PRINCIPLES

The general principles that Telstra recommends when developing applications for use on our network are that apps should be designed to:

- Have interoperability / compatibility with the Telstra Network
- Minimize unnecessary data transfer
- Optimize necessary data transfer
- Reduce unnecessary network signalling
- Be resilient to changing network conditions
- Be responsive
- Be secure
- Be future proof
- Minimize harm to other network users
- Be serviceable
- Conserve Power

7. APPLICATION DEVELOPMENT TECHNIQUES

In this chapter a series of techniques will be described that can assist in implementing the general development principles outlined above. Table 1 below summarizes the techniques that will be described and their benefits. Table 2 maps the techniques to the development principles.

The tables provide a ready reference such that the developer can quickly find techniques that are most relevant to their development priorities.

Note the techniques have not been put in any particular order.

Techniques	Benefits							Section
	Improved Battery Life	Lower Data Costs	More Responsive apps	Increased Application / Device Longevity	Robust / Resilient application	Reduced Network Signalling	Improved User Security and Privacy	
Network Access								
Conservative Retry Algorithms	Y					Y		7.1
Avoid Synchronized access to network					Y	Y		7.2
Avoid Network Polling	Y				Y	Y		7.3
Avoid Network Pinging	Y				Y	Y		7.4
Avoid unnecessary network connections	Y					Y		7.5
Content Delivery								
Appropriate Media Coding	Y	Y	Y		Y			7.6
Data Compression		Y	Y					7.7
Content Caching	Y	Y	Y		Y	Y		7.8
Data Push	Y					Y		7.9
Download Resumption	Y	Y	Y		Y			7.10
Delta Downloads	Y	Y	Y					7.11
Data Batching	Y					Y		7.12
Use buffering / Data prefetching			Y		Y			7.13
Use of aggregators / hubs					Y	Y		7.14
Firmware over the air updates				Y			Y	7.15
User Interface Experience								
Careful use of Wakelocks	Y							7.16
Defer some activities until charging	Y							7.17
Scale app activity with battery charge	Y				Y			7.18
Provide user controls for app activities	Y	Y	Y					7.19
Manage activity of backgrounded apps	Y	Y	Y			Y		7.20
Use best practise for location services	Y						Y	7.21
Use asynchronous logic for app coding			Y					7.22
Careful App UI design for responsiveness			Y		Y			7.23
Provide Offline Functionality	Y		Y		Y			7.24
Scale app behaviour to network speed			Y		Y			7.25
Ensure app has diagnostic capabilities				Y				7.26
Network Technology Interactions								
Wi-Fi Usage		Y			Y	Y		7.27
Appropriate App Development Choices	Y	Y	Y			Y		7.28
Appropriate Technology Choices	Y			Y	Y			7.29
Deactivate unnecessary features	Y		Y					7.30
Follow security guidelines							Y	7.31
Follow privacy guidelines							Y	7.32
Design apps with IPv6 transition in mind				Y	Y		Y	7.33
Off Peak Transactions						Y		7.34

Table 1: Benefits of Development Techniques

Techniques	Principles											Section
	Interoperability / Compatibility with Telstra Network	Minimize unnecessary data transfer	Optimize necessary data transfer	Reduce unnecessary network signalling	Resilient to changing network conditions	Responsive	Secure	Future Proof	Minimize harm to other network users	Serviceable	Conserve Power	
Network Access												
Conservative Retry Algorithms		Y		Y					Y		Y	7.1
Avoid Synchronized access to network	Y		Y	Y	Y				Y			7.2
Avoid Network Polling	Y		Y	Y	Y				Y		Y	7.3
Avoid Network Pinging	Y		Y	Y	Y				Y		Y	7.4
Avoid unnecessary network connections				Y					Y		Y	7.5
Content Delivery												
Appropriate Media Coding		Y	Y		Y	Y			Y		Y	7.6
Data Compression			Y			Y			Y			7.7
Content Caching		Y		Y	Y	Y					Y	7.8
Data Push		Y		Y					Y		Y	7.9
Download Resumption		Y			Y	Y			Y		Y	7.10
Delta Downloads		Y	Y			Y			Y		Y	7.11
Data Batching			Y	Y					Y		Y	7.12
Use buffering / Data prefetching					Y	Y						7.13
Use of aggregators / hubs		Y	Y	Y	Y				Y			7.14
Firmware over the air updates	Y						Y	Y	Y	Y		7.15
User Interface Experience												
Careful use of Wakelocks											Y	7.16
Defer some activities until charging											Y	7.17
Scale app activity with battery charge					Y						Y	7.18
Provide user controls for app activities		Y				Y					Y	7.19
Manage activity of backgrounded apps		Y		Y		Y					Y	7.20
Use best practise for location services							Y				Y	7.21
Use asynchronous logic for app coding						Y						7.22
Careful App UI design for responsiveness					Y	Y						7.23
Provide Offline Functionality		Y			Y	Y						7.24
Scale app behaviour to network speed			Y		Y	Y						7.25
Ensure app has diagnostic capabilities	Y							Y	Y	Y		7.26
Network Technology Interactions												
Wi-Fi Usage		Y		Y	Y							7.27
Appropriate App Development Choices				Y		Y					Y	7.28
Appropriate Technology Choices	Y				Y			Y			Y	7.29
Deactivate unnecessary features						Y					Y	7.30
Follow security guidelines							Y					7.31
Follow privacy guidelines							Y					7.32
Design apps with IPv6 transition in mind	Y				Y		Y	Y	Y			7.33
Off Peak Transactions			Y	Y					Y			7.34

Table 2: Development Principles and techniques to achieve

7.1. Conservative Retry Algorithms

7.1.1. Description

Conservative retry algorithms are required to prevent apps from continually trying to upload or download content in the event of failures such as server issues, timed out activities or slow network speeds. This is to prevent rapid battery drain and reduce harm to other network users.

7.1.2. Methods

The idea is to limit aggressive retry attempts and to have a sensible back off timer and a finite retry algorithm. A sensible retry algorithm will have a randomized back off time (before retry), with increasing time between retries, and a finite number of retries before indicating failure to the user/application.

In an M2M application, conservative retry algorithms are critical. Consider the utility M2M monitoring case where thousands of M2M devices might concurrently try to reconnect to their server after a power outage to upload their measurements. Lack of a sensible retry algorithm can cause network access congestion, affecting not only the M2M specific application but other network users.

Another consideration for M2M applications is the effects an unsuitable retry algorithm could have on the battery life of the device. The importance of getting the device back online as soon as possible needs to be weighed up against the power requirements of having a more aggressive retry algorithm.

7.1.3. References

http://developer.att.com/static-assets/documents/library/best-practices-3g-4g-app-development.pdf (Best Practices for Battery Life section)

7.2. Avoid synchronized app access to the network

7.2.1. Description

Smartphones and tablet devices are often synchronized to a common central clock. If multiple apps use absolute times to synchronize simultaneously to perform actions like fetch email or news updates, they may cause blocking at the local cell level or excess load across the entire network.

M2M devices in particular typically exist in large numbers with many sharing common network access points in the form of the local mobile base station. If all these devices were to attempt to signal the network simultaneously it would create a lot of congestion that could potentially ripple out through the rest of the network.

7.2.2. Methods

To avoid synchronized app access to the network, activities shouldn't be scheduled for the exact same absolute time across large applications. In an M2M scenario, we would not want all meter readings for a utility to occur at exactly the same point in time across an entire city. Similarly we would not want an email client set to check email at exactly 8am each morning.

Developers should consider spreading /randomizing device access by random offsets that are relative to the nature of the activity. For instance for a periodic M2M activity that requires a small transfer to occur hourly, spread the accesses for the devices randomly across the hour.

For something large such as a large software/firmware update consider randomizing device updates over a longer period such as week or a month and consider doing the data transfer at an off peak time such as in the middle of the night between Midnight and 6am.

For M2M solutions make sure you consider the case where a power outage (for devices that use mains power with no battery back-up) results in all the devices powering back up at the same time. These devices should not all try to reconnect to the network at the same time. Stagger the network activity of all the M2M devices so as not to cause network congestion.

Note Telstra's terms and conditions also mandate conditions around synchronized access to Telstra's network for multiple M2M modem devices and contains the following clause "If your Wireless M2M application employs more than 50,000 modem devices, you must provide a facility to control data transmission intervals in real time. We may require you to increase data transmission intervals during periods of network congestion".

7.2.3. References

Refer *"Use of multiple modem devices"* in Telstra's *Our Customer Terms*.

https://www.telstra.com.au/content/dam/tcom/personal/consumer-advice/pdf/business-b/dataservices-m2m.pdf

7.3. Avoid Network Polling

7.3.1. Description

Network polling is regular periodic connection to the network – usually used by a device to check for newly available information from its server. Regular polling is discouraged as it wasteful of device energy and it can unnecessarily congest the network, reduce network performance and impact other network users.

7.3.2. Methods

Some methods to avoid or reduce the impact of regular polling are:

- Use Push instead of regular polling (refer Section 7.9)
- Code application to only retrieve new data from server when app is first loaded, and from then on, only when user requests screen / data to be refreshed (e.g. via a refresh button).
- Increase time between polls.

It is also very important that any regular polling between instances of the app or devices are not synchronized in time exacerbating the problem (refer Section 7.2). Note that Telstra's Customer Terms have some restrictions around regular polling – please ensure that your app conforms to these.

In future, Telstra may further restrict polling in our network, so developers need to consider an alternative means to achieve their goal.

7.3.3. References

Our Customer Terms

https://www.telstra.com.au/content/dam/tcom/personal/consumer-advice/pdf/business-b/dataservices-m2m.pdf

http://www.telstra.com.au/customer-terms/business-government/telstra-mobile/data-services/index.htm

7.4. Avoid Network Pinging

7.4.1. Description

Pinging is the transmission of a small amount of data to keep a connection open.

Some apps periodically ping the network just to keep and check for a live connection to avoid reconnecting when it needs to transfer information. This is also known as a "heart beat".

7.4.2. Methods

This behaviour is wasteful of network resources and drains the device battery.

Telstra does not permit pinging/polling the network or device at a periodicity greater than every 60 seconds.

If periodic pinging is required, make the pings as far apart as possible and randomized to prevent multiple devices trying to access the network at the same time.

Devices/applications should not use heartbeats (given that there are other techniques available to the developer as outlined in this guide) as it can unnecessarily congest the network, reduce network performance and impact other network users.

In future, Telstra may further restrict pinging/heartbeats in our network, so developers need to consider an alternative means to achieve their goal.

7.4.3. References

Our Customer Terms

https://www.telstra.com.au/content/dam/tcom/personal/consumer-advice/pdf/business-b/dataservices-m2m.pdf

http://www.telstra.com.au/customer-terms/business-government/telstra-mobile/data-services/index.htm

7.5. Avoid unnecessary network connections

7.5.1. Description

Unnecessary connections should be avoided to conserve energy consumption and network resources.

According to app testing done by AT&T a common app problem is that devices/apps keep connections open too long after they are no longer needed (refer reference below).

7.5.2. Methods

Developers should write application code so as to prevent unnecessary network connections.

App should only access the network when required and close connections promptly after they have been used.

7.5.3. References

http://developer.att.com/application-resource-optimizer/docs/best-practices/closing-connections

7.6. Appropriate media coding

7.6.1. Description

Coding media appropriately to cater for target device capabilities can save on unnecessary data transfer and improve user experience.

7.6.2. Methods

Content optimization can minimize data usage and reduce download times. Developers should utilize the OS platform APIs/function calls/User Agent information to determine the device capabilities with regard to screen display resolution and streaming capabilities, and serve media to the device accordingly. The lowest image resolution/ video frame rate / audio codec rate that gives a good user experience should be used. Application Server should also have video/audio clips encoded in a variety of bit rates and the app should choose the media rate that suits the radio network being used.

7.6.3. References

References on media coding:

https://webplatform.github.io/docs/concepts/Detecting_device_and_browser/

http://www.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/

http://www.html5rocks.com/en/mobile/cross-device/

References on device OS connectivity details:

IOS: Reachability

http://developer.apple.com/library/ios/#samplecode/Reachability/Introduction/Intro.html#//apple_ref/doc/uid/DTS40007324-Intro-DontLinkElementID_2

Windows Phone: Windows.Networking.Connectivity

http://msdn.microsoft.com/en-us/library/windows/apps/br207308.aspx

http://msdn.microsoft.com/en-au/library/windows/apps/hh700381

Android: ConnectivityManager

http://developer.android.com/reference/android/net/ConnectivityManager.html

7.7. Data Compression

7.7.1. Description

Data compression where possible can be used to minimize data transferred over the network and reduce costs for the user.

7.7.2. Methods

If applicable, objects being downloaded or uploaded should be compressed at the server/device respectively for data transfer reduction.

Image size compression which is lossy can impact user perception if the resulting image quality is poor on the screen size being used. It is important to tailor image resolution to the capabilities of the device.

Applications that are text based and use HTTP protocols such as news aggregators lend themselves to compression techniques. The HTTP protocol supports compression of content from server to device. Most common web servers, web browsers and mobile operating systems support HTTP compression.

If the app's server supports this feature it can indicate this to the server using the HTTP Header Accept-Encoding with HTML code.

7.7.3. References

Technical specifications on Accept-Encoding HTTP header

HTTP/1.1 Specification RFC2616 Section 14.3

http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.3

7.8. Content Caching

7.8.1. Description

Caching refers to keeping a copy of data that a user has already downloaded, in case it is needed again.

For instance a web page may be cached so that if the user needs to refer to that content again (e.g. If they navigate back to a previously viewed page), it can be displayed from the device's cache rather than having to be re-downloaded from the server.

Caching can reduce the need to reload images, web pages, style sheets, etc. which results in fewer data transfers, reducing network signalling and making apps appear faster and more responsive.

Caching can't be used for data that has a real time element to it such as dynamically changing content, streaming multimedia and where there may be security issues.

7.8.2. Methods

The HTTP protocol supports caching.

HTTP Caching Techniques:

- App checks the device's local cache to see if required data is already available and still current.
- Content version on server can be indicated by a variety of methods including date of last modification or ETag which is a unique reference number for the data on the server.

- If cache data is valid/current then the app can serve this local data on the device directly to the user without any further network communication /data transfer required
- If cache data is invalid / out of date or doesn't exist then the server will send the required data to the device to update its cache and display.

Things for the developer to consider:

- Expiry times for data in cache
- Device/OS cache size
- Type of server data to be cached

7.8.3. References

Caching in HTTP, World Wide Web Consortium (W3C), RFC2616 Section 13:

http://www.w3.org/Protocols/rfc2616/rfc2616-sec13.html

AT&T Best Practice Deep Dive – Duplicate Content (Application Resource Optimization)

http://developer.att.com/application-resource-optimizer/docs/best-practices/duplicate-content

AT&T Best Practice Deep Dive – Cache Control (Application Resource Optimization)

http://developer.att.com/application-resource-optimizer/docs/best-practices/cache-control

AT&T Best Practice Deep Dive – Cache Expiration (Application Resource Optimization)

http://developer.att.com/application-resource-optimizer/docs/best-practices/cache-expiration

7.9. Data Push

7.9.1. Description

Data Push, also known as Push or Server Push, is where the app server initiates communication with a client device.

Many smartphone apps attempt to deliver real time news, notifications and other data to users by periodically polling the network.

Regular polling is wasteful if there is no new information on the server to retrieve and causes unnecessary network signalling and drain on device battery.

The general technique that should be adhered to by developers is to push data to the device when there is actually new and relevant information available. A major consideration however is that a device taking advantage of PSM (refer to section 5.4.5.1) technology will not be in a state to receive push notifications.

7.9.2. Methods

Developers should utilize Push data services provided by the device' operating system if they exist. Most major phone OS's now support this functionality.

For Apple IOS the service is known as Apple Push Notification system.

Google's Android push platform is known as Firebase Cloud Messaging (FCM).

Microsoft Windows Phone has the Windows Push Notification Service (WNS).

In addition apps should have settings to allow the user to configure and control how data is pushed to the device.

7.9.3. References

For Apple IOS: APNs – Apple Push Notification Service

https://developer.apple.com/library/archive/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/APNSOverview.html#//apple_ref/doc/uid/TP40008194-CH8-SW1

For Google Android OS: GCM – Google Cloud Messaging for Android

https://firebase.google.com/docs/cloud-messaging/

For Microsoft Windows Phone OS: WNS – Windows Push Notification Service

https://docs.microsoft.com/en-us/windows/uwp/design/shell/tiles-and-notifications/windows-push-notification-services--wns--overview

7.10. Download Resumption

7.10.1. Description

For apps that may be downloading large files from a server for storage on device such as documents, binary files, digital magazines, e-books and so on, it is important that the app has some mechanism to recover from an interrupted file transfer rather than simply trying to download the entire file again.

Download resumption saves on unnecessary data transfer, and benefits the user experience in terms of time, data cost and robustness.

7.10.2. Methods

The HTTP protocol has support for downloading a specific portion or range of a file.

Developers can utilize HTTP protocols to download chunks of files rather than the whole file at once and check for successful downloading of each chunk. Developers should be careful to use a sensible retry algorithm to re-download after an unsuccessful event.

Note all retry mechanisms should have reasonably finite limits and retry attempts, so as not to overload the network or exhaust user's data allowances and wallet!

The main relevant HTTP header is: Range – which is used to specify the portion of file to be downloaded.

7.10.3. References

For more information on HTTP Range header refer http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.35

7.11. Delta Downloads

7.11.1. Description

Delta downloads refers to the technique of only downloading information that has changed, rather than all of the information on a server, web page or other application.

Delta downloads, reduces the amount of data needed to be downloaded and thus benefits the user in terms of lower data costs, more responsive applications and lower battery usage.

7.11.2. Methods

Apps should be designed only to download content that has changed since last visiting site. Use of HTTP caching (described earlier in Section 7.8) for HTML based apps is one method to achieve this.

Non HTML based apps should be designed to use delta downloads wherever possible. For example if an app relies on a database that it downloads periodically to the device from a server, it should ideally only download the changes rather than the entire database.

7.11.3. References

HTTP Caching http://www.w3.org/Protocols/rfc2616/rfc2616-sec13.html

7.12. Data Batching

7.12.1. Description

It is inefficient to send small blocks of data on the network, and it can reduce device battery life and reduce network performance for all device users.

When a device switches from an idle to dedicated channel to send data it consumes 60-100 times the amount of power it does in the idle state (refer Section 3.3 of reference [1]). The state change also involves network signalling.

By batching data, we save on these state changes, reduce battery drain and network signalling (which is beneficial for other network users).

Batching applies on both the uplink (device to network) and downlink (network to device) sides.

7.12.2. Methods

It takes time, power and network signalling to switch between device RRC (Radio Resource Control) states and so it makes sense to batch up data and send it in one go rather than in small blocks of data – reducing these state changes.

Also, in the uplink, every transmission from the device that is separated by more than 10ms requires additional network scheduling resources and can therefore increase the latency seen by the device, and other devices in the network.

In the downlink data should also be batched where possible. This will help reduce device idle/connected mode transactions that consume device power.

By batching a bunch of uplink transactions into a single TCP request, it can make a huge difference.

Batching benefits both the network (fewer overheads) and the device /application (less latency and lower power consumption).

7.12.3. References

[1] GSMA Smarter apps for Smarter phones (Section 3.3 Efficient Network Connection Usage)
http://gsmaterminals.github.io/Developer-Guidelines-Public/

[2] AT&T Best Practice Deep Dive – Periodic Transfers (Application Resource Optimization)
http://developer.att.com/application-resource-optimizer/docs/best-practices/periodic-transfers

7.13. Use Buffering / Data Prefetching

7.13.1. Description

Buffering /Data Prefetching is the preloading of data from the network into the device's memory. Buffering can help an application look and feel more responsive thus improving user experience.

However prefetching data presents a danger of downloading unnecessary data if the user doesn't end up needing it. Buffering/data prefetching should only be used if there exists a high certainty that the data is definitely going to be needed/used by the user.

Buffering differs from caching, in that caching only occurs once the data has been accessed and consumed previously and thus helps when the data is needed again.

7.13.2. Methods

Developer needs to carefully analyse their apps expected user behaviour and balance the need for prefetching data to give a good user experience/latency with the costs of doing so, if the user doesn't end up needing the data.

7.14.Use of aggregators / hubs

7.14.1. Description

Aggregator/hubs batch up data from a group of devices to the send to the network.

7.14.2. Methods

In an M2M network, aggregators provide a common hub to disperse collected data rather than have every single device talk to the network individually. This technique would reduce the number of devices on the cellular network saving the developer money and reducing unnecessary network data traffic and signalling hotspots.

7.15. Firmware over the air updates

7.15.1. Description

Firmware over the Air (FOTA) upgrade is the ability of a device to have its firmware/operating system and RF chipset firmware upgraded using the cellular network ("over the air").

Note FOTA can be achieved by proprietary methods or standardized methods such as described by OMA (Open Mobile Alliance) Specifications body (http://openmobilealliance.org/) in which case it is using the OMA-DM (OMA Device Management) standard on FUMO (Firmware Update Management Object). Telstra has no preference for method used. Telstra expects the vendor/OEM or integrator to host the FOTA server for their devices.

Note that Telstra reserves the right to insist on a firmware upgrade using this capability at any time should we find device issues causing network harm/other user harmful impacts.

For M2M devices and applications

FOTA capability is extremely important for M2M devices given the:

- Large number of devices in remote locations
- Longer lifecycle / lifespan of these devices compared with smartphones. FOTA is important because if any bugs are found in the device while in the field, or any future incompatibilities

are found between the device and our radio network, these can be remotely rectified by the vendor/manufacture.

For M2M developers seeking Telstra endorsement of their M2M device or solution, note that it is now mandatory that there is a mechanism to update the firmware of both the cellular modem and the software of the integrated device remotely. In some rare cases exceptions may be made and other mechanisms for upgrade may be allowed e.g. fixed internet connectivity, Wi-Fi, cabled connection.

For Smartphone Apps

In an analogous fashion the smartphone app developer should make use of the operating system provided app store mechanism to provide app software updates.

This has the advantage that the developer can -

- Continuously improve and update the functionality of the app over time – making it more attractive to users.
- Rectify any bugs discovered by users in the field.
- Progressively apply principles outlined in this document (not already or initially implemented).

7.15.2. Methods

FOTA guidelines:

- Developer must work with module vendor / distributor to understand module FOTA capability and how to integrate in their device (may need to subscribe to vendors device management cloud service for instance)
- Developer must ensure device software is developed so that it interworks with vendor's module FOTA solution
- Delta software upgrades to be used if possible – rather than downloading entire software or firmware version, just download the delta component.

7.16. Careful use of Wakelocks

7.16.1. Description

In order to save battery life, some mobile OS platforms such as Android will dim the screen after a period of time, and then reduce the CPU power.

Wakelocks are (in Android) a power manager system service that app developers can use to control the power state of the phone.

A developer for instance might use a wakelock to prevent the screen turning off whilst a video or song is playing.

7.16.2. Methods

Developers should avoid use of wakelocks except where absolutely necessary and if they are used they must be released immediately after they have been used. If not released, the phone won't be able to power down the screen/ CPU and will drain the battery rapidly.

7.16.3. References

http://developer.android.com/reference/android/os/PowerManager.WakeLock.html

http://software.intel.com/en-us/articles/wakelocks-for-android

7.17. Defer some activities until charging

7.17.1. Description

Device battery life can be extended by deferring non time critical uploads/downloads until charging.

7.17.2. Methods

Consider settings options for your app to only upload / download large files such as captured photos and videos when charging. Google's "Google+" app has such an option in its settings to only upload photos on network when connected to a charger.

Common operating systems have a mechanism to determine device charge status. Refer references below.

7.17.3. References

Android

http://developer.android.com/training/monitoring-device-state/battery-monitoring.html

iOS:

http://developer.apple.com/library/ios/#documentation/uikit/reference/UIDevice_Class/Reference/UIDevice.html

Windows Phone Device:

http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff941122(v=vs.105).aspx
http://msdn.microsoft.com/en-us/library/windowsphone/develop/microsoft.phone.info.devicestatus.powersourcechanged(v=vs.105).aspx

7.18. Scale App activity with battery charge

7.18.1. Description

In order to extend device battery life, app activity should be ratcheted down as battery charge declines.

7.18.2. Methods

Some possible app activities that could be scaled as battery charge declines are:

- Reduce periodicity / frequency of app updates or polls as battery declines.
- Reduce retry algorithms in low battery situations so as to not hasten a flat battery
- Do not allow certain activities without user warning and acceptance
    - E.g. once battery reaches a certain limit, do not allow certain activities such as uploads / downloads of large files, streaming, GPS activation etc. Inform user that battery is low and connection to a charger is recommended to continue activity (allow user to override however).

Most common operating systems have a mechanism to determine battery charge status. Refer references below.

7.18.3. References

Google Android Battery Manager

http://developer.android.com/reference/android/os/BatteryManager.html

Apple IOS UIDevice Class Reference

http://developer.apple.com/library/ios/#documentation/uikit/reference/UIDevice_Class/Reference/UIDevice.html

Windows Phone Battery Class

http://msdn.microsoft.com/en-us/library/windowsphone/develop/jj207231(v=vs.105).aspx

For M2M developers refer to vendor's module documentation.

7.19. Provide user controls for app activities

7.19.1. Description

By allowing user to have control over an apps behaviour they can have some direct control over battery life, performance and usage costs.

7.19.2. Methods

Provide app settings to limit/control apps activity. Examples of possible app settings that could be implemented to control app activity are:

- Restrict data transfers dependant on connected network e.g. Don't allow on cellular, only on Wi-Fi

- Restrict large data transfers to particular networks e.g. only on Wi-Fi or only on fast networks i.e. only on 4G and not on 3G
- Restrict some activities dependent on battery charge status e.g. Only when above certain charge or when charging

7.20. Manage activity of backgrounded apps

7.20.1. Description

Backgrounded refers to when the user interface of the application is not visible to the user and the app is not actively being used, in a multitasking environment.

Once an app is placed in the background, the user might reasonably expect the app is not doing any data transfers. This however is not always the case.

Carefully managing the activity of backgrounded apps is important to ensure user satisfaction.

7.20.2. Methods

Avoid network chattiness for your app in background mode – unless it is very clear to the user that the app will still work in background mode.

Cease relevant activity of app when it is placed in background e.g. stop video/ audio streaming , network connection and so on until app is brought into the foreground but continue other activity that might be required e.g. keep track of videos user has watched etc.

Give user an option in the app settings to stop app data transfer in background use. Some apps continue to run when in the background including transferring data which might not be what the user wants.

7.21. Use best practice for location services

7.21.1. Description

- Using GPS for location services is one of the fastest ways to drain your battery.
- User privacy with respect to their location needs to be respected by the application and the relevant legislation adhered to.

7.21.2. Methods

- GPS should only be activated when required and turned off otherwise
- Ensure the app has appropriate user permissions to perform locating
- Use last known location to indicate to user a rough position whilst calculating actual position as likely it will at least be in the right part of the world
- Consider using any operating system provided network course locating services where the accuracy of GPS is not required (e.g. Based on Cell-Id or Wi-Fi servers)
- Use methods to assist GPS acquisition and obtaining a quicker fix such as AGPS (Assisted GPS) services. AGPS allows downloading of GPS satellite almanac location by the cellular network to help the time to a first location fix.

7.21.3. References

The android developer site has some good guidelines on location services for Mobile. While specific to Android code, it has some general principles that are applicable to other operating systems as well.

Refer: http://developer.android.com/guide/topics/location/strategies.html

7.22.Use asynchronous logic for app coding

7.22.1. Description

To maximize user satisfaction, apps should be designed to be responsive which can be achieved using asynchronous logic for the main code block of an app.

If an app is coded in a serial manner, any missing network or server responses could cause the app to halt whilst it waits for the response whereas asynchronous is not totally dependent on any network responses for its flow or progress.

This will prevent the device hanging if it doesn't get an expected response from the network for instance if it is busy at the time.

7.22.2. Methods

- Apps main code thread should not be designed in a sequential or serial flow where the next step is dependent on say a server response that may be lost or delayed due to network connectivity issues.
- App will ideally be able to cope with lost or delayed responses from its server/cloud. The app needs to appear to the user to be working even if it hasn't received all data it needs – so it should be doing things in parallel, such that some data can be presented while waiting for other data.
- Making the app code asynchronous will help decouple app progress from network responses.

7.22.3. References

Refer Section 3.2.1 of GSMA guidelines

https://gsmaterminals.github.io/Developer-Guidelines-Public/

7.23.Careful App UI design for responsiveness

7.23.1. Description

Careful app UI (user interface) design can assist in making application appear responsive.

7.23.2. Methods

Some UI design techniques and tricks to make the app appear responsive are:

Design user interface to have no hangs

Similar to the above item, the user interface of the app should be designed such that there are no dead ends / roadblocks or hanging menu items or UI elements if a user enters unexpected data or doesn't do as expected.

Thorough testing of the UI – especially by "real" users should assist with debugging these issues.

Error and status messages should be able to be dismissed by the user – rather than block the user from further progress, so that the user can continue using other elements of the app.

Sensible display of content

Sensible display of content can mask any temporary network connectivity issues and can make the app look more smooth and responsive. Techniques such as

- Displaying content on a page as it becomes available, even if out of order will make the app look fast
- Use content placeholders for content such as images, so the user knows something is happening, even if it hasn't happened yet!
- Pre-fetch commonly accessed content (there is a trade-off here between making app responsive and possibly downloading unnecessary information and increasing user downloads and network activity. This should only be done if there is a high certainty that data is going to be needed by the user)
- Don't display error messages to the user if they are not relevant – e.g. don't display temporary offline error messages if the user isn't using /needing the network or is using offline functionality.

Progress and Status indicators

If the UI doesn't provide appropriate feedback to the user, the user can be unsure about the app or the network working correctly.

To avoid this, appropriate progress and status messages or UI graphical displays should be given to the user, including methods to cancel operations at appropriate times to remove uncertainty and give the user some control should the app be slow or hanging.

Progressively save session info and data

App should be designed to progressively save session info and working data on device, to allow the app to continue working almost seamlessly where possible if network connectivity is lost and when it returns. This will work hand in hand with offline functionality that has been built into the app.

7.24. Provide Offline Functionality

7.24.1. Description

Offline functionalities are important to improve user experience when network connectivity is lost or slowed.

7.24.2. Methods

When network connectivity is lost, or when network throughput is slow; an app needs to have some functionality that can still be used by the user.

The app should be designed with offline usage in mind. Important data could be cached and in the event of network loss, data could still be processed on the device and returned to server once network connection resumed.

When connectivity is lost the app should in the background seek to re-establish network connectivity in a non-aggressive and finite retry method.

7.25. Scale app behaviour to network speed

7.25.1. Description

Scaling app behaviour based on network speed will make for a robust and more responsive appearing app.

7.25.2. Methods

App should be developed to scale functionality according to the capabilities of the network it is connected to. For instance:

- App full functionality provided when connected to very fast networks such as 4G, and reduced for slower networks
- More data intensive functions of the app should be stripped back as the connected network slows. For instance, when on 3G, consider restricting video streaming functionality as the network speeds will not be sufficient to support at high quality. Scale back the codec rate of video delivered when connected to lower speeds, but offer higher quality video streaming and image resolution when on the faster 4G network or Wi-Fi.

7.25.3. References

Section 7.6.3 details how the major operating systems make connectivity details available for developers.

7.26. Ensure application has a diagnostic capability

7.26.1. Description

Diagnostic capability for M2M devices and applications is important to allow customer complaints and faults to be investigated by operator or device vendor. Telstra recommends that any diagnostic functionality should be able to be accessed remotely.

7.26.2. Methods

A diagnostic capability will assist operator support staff in determining whether a complaint or issue is related to a network or device fault.

If it is a device fault, remote diagnostic capability will help device vendor/developer support staff isolate the issue.

Any device integration must not impact on the ability of the embedded module also to be diagnosed. The integrator should ensure that diagnostic information can be obtained for both the embedded module as well as the device as an integrated whole.

Diagnostic information ideally should be available over the air in real time, or alternatively by collection of error or session log files from the device and should support the following features (the more features supported the more readily any fault could be determined):

<u>Device Details</u>

- Device identification details – Serial Number, Model, IMEI (International Mobile-Station Equipment Identity)
- Device status details
- Device IP Address

- Time stamp of events

Radio Network Details

- Network Code (MCC, MNC – Mobile Country Code and Mobile Network Code respectively)
- Preferred PLMN (Public Land Mobile Network) List (read from UICC (Universal Integrated Circuit Card i.e. SIM card)
- Radio Technology being used (3G, 4G or Wi-Fi)
- Information on network attach type and registration status
- Packet registration area
    - For 3G RA (routing area)
    - For 4G TA (tracking area)
- For the radio technology being used the appropriate signal strength/quality indicators.
    - For 3G RSCP and Ec/Io
    - For 4G RSRP and RSRQ
- Radio Bearer details (for uplink and downlink bearers)
    - E.g. For 3G R99, HSDPA, HSDPA+, EUL
- Information on radio cell connected to –
    - For 3G PSC (Primary Scrambling Code)
    - For 4G PCI (Primary Cell Identity)
- Any additional / relevant info e.g. neighbour cells, session call history

Software/Firmware/Driver Details

- Enumeration of hardware/software/firmware versions of the embedded module and application hardware/software
- RNDIS/NDIS/MBIM Driver details if relevant (relevant to embedded modules within Windows Laptops)

Error / Diagnostic Messages

- Application software diagnostic codes
- Embedded module error codes
- Radio Network Error codes (per 3GPP specification 24.008)

Other considerations for the diagnostic capability would be:

- Alternate methods to transfer diagnostic info – e.g. GUI, or transfer log file off device – as coverage or device fault issues may impact on its ability to transfer data over air.

7.27.Wi-Fi Usage

7.27.1. Description

In some instances it may be preferred for a device to be connected via Wi-Fi where available. Benefits include less power consumption and generally cheaper data rates. Applications could be configured to only perform certain actions when a Wi-Fi connection is active such as large firmware and application updates, cloud syncing, and other large file transfers.

7.27.2. Methods

Mobile OSs are typically designed to default to using a Wi-Fi connection rather than a cellular connection providing Wi-Fi functionality is enabled and a successful W-Fi connection can be made.

App developers can also take advantage of Wi-Fi by providing options within apps to selectively do some actions only when connected by Wi-Fi.

Developers can use OS provided methods to check whether Wi-Fi is turned on, scan for Wi-Fi and switch off cellular if allowed. Different operating systems vary in what they allow to be controlled. For the main Mobile OS the relevant functions are:

- Android – WifiManager
- IOS – UIRequiresPersistentWiFi

7.27.3. References

Android
http://developer.android.com/reference/android/net/wifi/WifiManager.html

Apple
https://developer.apple.com/library/ios/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/PerformanceTips/PerformanceTips.html (Using Wi-Fi section)

7.28. Appropriate App Development Choices

7.28.1. Description

By careful and appropriate app development choices, battery life can be extended and network signalling minimized.

7.28.2. Methods

Reduce complexity of web sites the app uses

Developers should consider carefully before utilizing resource intensive web technologies for applications.

Complex websites will require more work of the device's CPU, and more network transactions.

Choose 3rd Party components carefully

Many app developers use 3rd party services to add features to their app such as advertising (banner ads, in app advertising), location services and usage analytics.

While they save considerable development effort, and provide services the developer may not be able to themselves, they need to be chosen carefully so that the benefits of including them aren't outweighed by the negative side effects like increased signalling and power consumption.

Utilize OS platform provided APIs to assist with development

APIs (Application Programming Interface) are software functions and methods that implement low level functionality that can be called upon by higher level software.

APIs help abstract the detail for instance of which network is being used for the connectivity so that the developer doesn't have to worry about these details when coding their apps.

Platform APIs exist to help manage and monitor a wider variety of functionality including authentication, compression/decompression, media transcoding, support of HTTP caching, threads, support security functions and so on.

Advantages of using APIs for developers are:

- Faster development times
- APIs can expose to the developer platform network capabilities and connectivity information without the programmer having to know the details of the network

7.29. Appropriate Technology Choices

7.29.1. Description

When selecting an embedded module for a M2M application or an integrated M2M device, a developer should take into the consideration a variety of factors:

- performance
- longevity
- reliability
- ease of development
- coverage
- throughput

7.29.2. Methods

Appropriate Cellular Technology Choice

Telstra recommends LTE as the preferred technology. LTE only modules are available as well as multimode LTE + 3G modules. 3G only modules are not preferred as their longevity is less than that of LTE.

Coverage / Radio Network Technology Support / Frequency Band Support

If developing for the preferred LTE network it is important to ensure that LTE coverage is available over the entirety of the expected usage areas.

Telstra's LTE coverage is constantly expanding so developers should refer to our coverage maps for the latest information at: https://www.telstra.com.au/coverage-networks/our-coverage

Refer to the table in section 5.1 of this document for information regarding Telstra's current and future frequency band support for both 3G and 4G technologies.

Choose devices /modules certified for use on Telstra's network

Certified modules have been tested by Telstra for compatibility with Telstra's network. Refer to section 5.6 for more information on the advantages of choosing a Telstra Certified device.

Data only or Voice and Data

Note when selecting an embedded module - some modules only support data and others support both voice and data.

If the application doesn't require voice, then a data only module is recommend as it will be cheaper and less complex.

FOTA (Firmware over the air)

FOTA (refer Section 7.15) is now a requirement for new modules to be certified for use on our network. Older modules do not have this functionality.

Having the ability to fix bugs and update devices remotely saves developers a lot of time and energy down the track. Particularly for an IoT solution that may feature hundreds or even thousands of devices spread out over a large area it would not be practical to physically attend to each device to individually to install a software patch or update.

Module Radio Network Features

Choose modules that support important radio network features such as –

- Advanced receiver types for 3G. Type 3i – gives best receiver performance resulting in better throughput performance and better receiver sensitivity and interference tolerance.
- Modules that are Telstra certified must support Receiver Diversity (2 receive antennas). Integrated device shall also support receive diversity i.e. have two external antennas or ports for best receive performance.
- Support power saving and network signalling reducing features such as
    - **CDRX** (Connected Discontinuous Reception) for 4G – this is a device feature that allows the device to have micro sleeps. This feature allows the device to reduce

battery consumption while minimizing impact to latency. CDRX is only invoked after 100ms of inactivity.

- **FD** (Fast Dormancy) for 3G - this is a 3G feature that allows the network to instruct the device to go to lower power state. This feature conserves power and reduces network signalling
- **PSM** (Power Save Mode) for 4G – this is an IoT device feature that requires network support. The device will inform the network that it will be going into power save mode (using close to no power in this state) along with information about when it will 'wake up' for a short period to receive any messages that may be waiting.

- Note Telstra no longer certifies 3G only modules for use on our network
- Telstra would prefer modules to use the highest speeds possible for non IOT applications (bearing in mind the economics for the developer/integrator) as this makes for a more efficiently operating network that benefits all users. Telstra would prefer that if an integrator chooses to use a 3G only module that it supports at least 7.2Mbps (Category 8) downlink speeds.

MTU (Maximum Transmit Unit Setting)

- Current recommendations for MTU Settings for IPv4 applications are that the link MTU size in the MS should be set to the value provided by the network as a part of the IP configuration (based on 3GPP recommendations – Refer 3GGP TS23.060 V10.5 Section 9.3).
- For IPv6 MTU size should ideally be discoverable using PMTUD feature (Path MTU Discovery) but if the device is not able to perform PMTUD then MTU size should be set to 1358 bytes as defined in 3GPP TS 23.060 v10.5 Appendix C. If 1358 bytes cannot be implemented or causes issues then the minimum MTU size shall be 1280 bytes.

Other Module Features to consider

Ensure module chosen has other required technologies that application requires (or that these can be built into the rest of the embedded device), such as

- Has appropriate hardware interfaces
- Has appropriate powering capabilities
- Has a useful development environment platform and support resources
- Has Wi-Fi capability if required
- Internal GPS/External GPS/No GPS as needed
- Supports IPv6 (refer Section 7.33)

Consider desired lifecycle of M2M device when making technology choices

It is extremely important to consider the life cycle of an M2M solution/application/device.

M2M applications often require devices to be remotely located, in large numbers and for long periods of time. It is not trivial to replace these devices so it is important to consider future proofing and longevity issues when selecting a device.

Even though newer technology modules may be more expensive they will have the greatest longevity.

Choice of a module that supports more of our bands (and future bands) for the desired technology (3G or 4G) will extend lifecycle of the device and improve performance (additional bands useful for congestion avoidance). Telstra prefers 4G modules as 4G/LTE supports a greater number of connected devices in any given area than does 3G.

Other lifecycle impacting aspects that need to be consider when choosing a module for M2M device:

- Does it use a Telstra Approved Module?
    - Approved modules and devices will typically have better longevity due to greater compatibility with our networks - features, technology and frequency bands
    - Approved Modules have been tested for compatibility with our network
    - Non-approved modules and any devices integrating them, have no guarantee that they will work with our network currently or into the future.
    - For M2M Integrated Device approval, there is a streamlined process for devices integrating already approved Modules

- Vendor Support?
    - EOS (end of support) date of module from the manufacturer
    - EOL (end of life) date of module from the manufacturer
    - Vendor FOTA support

- FOTA Capability
    - FOTA capability will help extend lifecycle of device by allowing software/firmware patches/upgrades and maintenance releases to be applied. Refer section 7.15.

7.29.3. References

- Refer 3GGP TS23.060 V10.5.0 Section 9.3 and Annex C http://www.etsi.org/deliver/etsi_ts/123000_123099/123060/10.05.00_60/ts_123060v100500p.pdf
- IPv6 Specification RFC2460 http://tools.ietf.org/html/rfc2460

7.30. Deactivate Unnecessary Features

7.30.1. Description

Deactivating unnecessary device features will help extend device battery life and may reduce unnecessary data usage.

7.30.2. Methods

Don't use or enable services such as GPS locating, Bluetooth, camera, accelerometer, other sensors if not needed for app. Turn these features on only when required and turn off promptly after use.

Consider using any network provided course locating where the accuracy of GPS is not required.

7.30.3. References

The following references all provide information on battery life impact of smartphone features.

http://arxiv.org/pdf/1212.1896v2.pdf

http://www.intel.com/content/dam/doc/best-practices/technology-tips-boost_battery_life_handhelds.pdf

http://static.usenix.org/event/usenix10/tech/full_papers/Carroll.pdf

7.31. Follow security guidelines

7.31.1. Description

Developers need to consider security and privacy aspects of their application in order to protect their users and their data.

Developers need to consider the following when developing applications

- Security of users sensitive information
- Fraud Prevention
- Provide an OTA (over the air) software/firmware update mechanism – so any identified security issues can be quickly patched (refer 7.15).

7.31.2. Methods

Some methods developers can employ to ensure the security of their app and its data:

General Guidelines:

- Use the respective OS platform's app store update mechanism to address any app security issues ASAP
- Do not store or send user passwords or any other sensitive information in unencrypted text
- Use secure protocols such as SSL/TLS for transmitting any sensitive information over the network
- Enforce higher security password requirements on the user. e.g. A mixture of upper & lower case, alpha numeric & special symbols, and lengths > 6 characters
- Ensure that no sensitive information is stored in the app log files
- Test the app to ensure that passwords / authentication cannot be bypassed
- Minimize app platform permissions to only the absolute minimum necessary so as to minimize vulnerabilities and increase user confidence
- Developers should use well known standardised security libraries / third party software APIs that provide security/encryption functions that have been well tested in the market (and hardened/patched against known vulnerabilities)
- Note many of the platform OS security protections can be circumvented by 'jail breaking' or 'rooting' the device – so ensure that app code uses its own security mechanisms beyond those provided by the OS platform.
- Developers should only distribute their app via the official OS platform's app store and not make the app package available for distribution elsewhere. Malicious code can be inserted into standalone versions and redistributed versions of the app can leave users vulnerable to identity theft and various forms of malware.

M2M Application Specific Guidelines

Security cannot be trivialised – especially considering some of the main applications of M2M. The impact of security and hacking breaches can be extremely serious.

M2M devices are key to emerging industries such as smart grids and health monitoring. Needless to say security (and privacy) breaches in these cases could have life threatening and wide spread community impact.

Security is also needed for Fraud prevention – given that these devices and their SIMS may be relatively accessible in high numbers.

Some specific security measures for M2M devices include:

- Firmware update capability (OTA) to allow device to be quickly patched should any security issues / vulnerabilities come to light. Refer section 7.15.
- Ensure the physical security of the SIM in the device

  For instance to avoid the oft-cited scenario where utility meters sim cards are stolen and used for data/call theft.
- Use IPv6 - due to its enhanced security features
- Utilize vendors FOTA to ensure you have the latest firmware for your device
- Review module and OS development platform security guidelines
- Ensure device has sufficient password protection / user authentication procedures to prevent against hacker access
- The physical security of devices when installing
- Tamper proofing the device
- Consider alarming device back to central server e.g. alarm if enclosure is opened
- Utilize external consulting/testing expertise against hacking/intrusion for critical M2M applications in utility and health monitoring areas.
- Consider hiding SSID for Wi-Fi connected devices. No need to broadcast.

7.31.3. References

OS Platform Security Guidelines

Each of the major mobile OS platforms has its own security guidelines for developers. These are a very good reference for developers.

Google Android:	http://developer.android.com/training/articles/security-tips.html
Apple iOS:	https://developer.apple.com/library/mac/documentation/Security/Conceptual/SecureCodingGuide/Introduction.html
Windows Phone:	http://www.microsoft.com/en-us/download/details.aspx?id=42509

General Security Guidelines

Android Developer Security Tips (whilst specific to Android contains principals that are applicable to all platforms): http://developer.android.com/training/articles/security-tips.html

GSMA IoT Security Guidelines - Covers IoT endpoints (devices), network elements and services ecosystem infrastrcture: https://www.gsma.com/iot/iot-security/iot-security-guidelines/

SSL (Secure Sockets Layer) – RFC 6101: http://tools.ietf.org/html/rfc6101

TLS (Transport Layer Security) – RFC 5246: http://tools.ietf.org/html/rfc5246

7.32. Follow privacy guidelines

7.32.1. Description

Telstra respects user's privacy and it is important that developers ensure users privacy. There are legal penalties for not following the applicable laws.

7.32.2. Methods

- Ensure that no user information is sent to third parties without the user being clearly informed and opting in to the service (they must be clearly informed of where the data is going and how it will be used)
- GPS should not be used without the users consent to track / record user location
- Ensure security guidelines above are adhered to as these will help safeguard the users privacy
- Ensure your app complies with Australian Privacy Laws.

7.32.3. References

Australian Privacy Laws: http://www.oaic.gov.au/privacy/privacy-news

http://www.futureofprivacy.org/wp-content/uploads/Best-Practices-for-Mobile-App-Developers_Final.pdf

7.33. Design apps with IPv6 transition in mind

7.33.1. IPv6 Requirements

Refer Section 7.33.2 below for an explanation of IPv6 terminology.

Telstra is committed to implementing the IP version 6 protocol (IPv6) for communication with mobile devices connected to its network. Telstra is progressively enabling APNs for IPv6 use.

Telstra.wap and Telstra.Internet APNs have been enabled for IPv6 Dual Stack Usage.

Telstra plans to enable single stack IPv6 on Telstra.wap APN in H2 2017.

Once IPv6 single stack capability is enabled in the network, new consumer mobile devices (e.g. handsets, smartphones and tablets) that support 464XLAT will be connected using IPv6 only (or "IPv6 single stack", with the "IPv6" Packet Data Protocol or PDP type).

Devices without 464XLAT support will be connected either using simultaneous IPv4 and IPv6 connections ("dual stack") with the IPv4v6 PDP type, or else with IPv4 only (as now).

Telstra-homed devices roaming on other networks will only use IPv4 for connections, until such time that there is general support for IPv6 amongst global mobile carriers.

Wireless broadband devices (USB Dongles, Wi-Fi hotspots, Gateways) will be required to support dual stack IPv4/IPv6 connections using the IPv4v6 PDP type.

M2M devices using a custom APN may be enabled to use IPv6 only (single stack) on a case by case basis. Telstra can be contacted by email at mobilitypartner@team.telstra.com to discuss this if required. Telstra expects future large-scale deployments of M2M devices will be configured to use IPv6 only (single stack).

IPv6 support is being progressively rolled out in Telstra Network on an APN (access point name) basis. Currently (as of Jan 2017) telstra.internet, telstra.wap and ims APNs have been enabled. For other APNs please check with Telstra.

7.33.2. Description and Terminology

The drivers for the use of IPv6 and the description of IPv4 exhaustion have been well documented on the Internet, and will be assumed to be understood by device application developers – see the Wikipedia article http://en.wikipedia.org/wiki/IPv6 for an introduction to IPv6 if required.

IPv4 and IPv6 are distinct protocols that do not natively interoperate, but it can be assumed that both protocols will need to coexist in the Internet for many years to come. This means that any device will need to communicate with other devices that may themselves either speak only IPv6 or IPv4, either by itself natively speaking both protocols ("dual stack" configuration) or through a protocol translator or tunnel in the network or the device (e.g. NAT64, 464XLAT, etc.).

Single Stack

A single stack network or device as the name implies only supports a single IP protocol type. Single stack is often denoted simply as SS.

There are two possibilities:

1. IPv4 SS network, supporting only IPv4 traffic as most existing wireless networks are today
2. IPv6 SS network, supporting only IPv6 traffic

Dual Stack

A dual stack network is a network whose nodes are capable of processing IPv4 and IPv6 traffic simultaneously. It thus facilitates the transition to IPv6 while many devices and internet sites are still IPv4 by letting the two protocols co-exist.

A dual stack UE (user equipment aka mobile device), supports the following PDP types IPv4 (single stack), IPv6 (single stack) and IPv4v6 (dual stack or "DS" – that is an IPv4 and an IPv6 connection simultaneously).

A dual stack UE will request a dual stack bearer (IPv4v6) and the network will allocate the appropriate bearer which may be either IPv4 single stack, IPv6 single stack, or dual stack (both an IPv4 and IPv6 connection).

464XLAT

For handsets & tablets that are IPv6 single stack capable there will still be a need for many years to co-exist with the IPv4 ecosystem. E.g. to reach IPv4 sites, or for apps that are not yet IPv6 compatible (i.e. contain IPv4 literal addresses within their code).

RFC6877 464XLAT provides a solution to allow IPV4 services & applications to work over an IPV6 single stack network. The 464XLAT solution requires at a minimum a NAT64 in the network along with 464XLAT daemon/code running on the device.

464XLAT refers to architecture for both network and device to allow this to work and this is described in RFC6877. Refer http://tools.ietf.org/html/rfc6877. 464XLAT was first proposed by T-Mobile (Cameron Byrne) in partnership with NEC and JPIX in this RFC.

The 464XLAT code required by the device is open source code and is available to be used in any operating system.

IPv6 Support – Smartphones & Tablets

Applications should be designed to use IPv6 as soon as it is available in the network.

Google has incorporated the XLAT code into the Android operating system since version 4.3 (October 2013). Version 4.4 has an improved version (e.g. handles MTU better) and is the minimum required version for use in Telstra's network. Other operating systems may include it in the future.

Telstra expects handsets and tablets to support IPv6 with 464XLAT capability.

IPv6 Support – M2M

M2M devices have the least IPv6 support as they tend to be the most cost sensitive devices and therefore are often produced using older chipsets with less feature support.

Importantly M2M devices have quite long life cycles compared to Smartphones. So it is even more vital that IPv6 is considered when developing M2M applications.

Developers should choose devices supporting IPv6 and ideally develop IPv6 compatible applications.

Telstra will be requiring module manufacturers to certify IPv6 capable devices going forward.

7.33.3. Methods

Some basic app guidelines for IPv6 are:

- Don't hardwire IPv4 addresses in to you app / app code – use variables to represent IP addresses.
- Ensure when coding to use a variable for IP addresses that can hold an IPv6 address
- Ensure apps are both IPv4 and IPv6 compliant
- For M2M solution developers - use IPv6 capable modules in your application and particularly Dual Stack IPv4v6 capable embedded modules in your device once they are readily available (not widely available currently – but at some point in the future). IPv6 capability will help future proof your device/application – increase its longevity and increase its security (if new security features of IPv6 are utilized).

- Developers using servers as part of their application should ensure that their servers are dual-stacked or IPv6 enabled. The application should be designed to provide IPv4 End to End or IPv6 End to End, with IPv6 E2E being preferred.

7.33.4. References

IPv6 General References
https://www.internetsociety.org/deploy360/ipv6/faq/
http://test-ipv6.com/

XLAT464 References
http://tools.ietf.org/html/rfc6877
https://sites.google.com/site/tmoipv6/464xlat

7.34.Off Peak transactions

7.34.1. Description

Cellular networks can only support so many simultaneous data users in a given cell. Therefore if users / applications can shift their data transactions to less popular times then the chances of heavily loading the network will be less and performance better.
Networks busy hours tend to be weekdays in the evening around 7-10pm.

7.34.2. Methods

Telstra recommends shifting transactions that don't need to be executed during popular usage times into the "quiet" hours between midnight and 6am by using batch and send techniques i.e. accumulating measurements made during the day and then sending that data at night.

For data that must be sent at regular intervals throughout the day, it is critical that that all the devices in the one geographic area (for example, say all those that are within a 5km radius of any given device), are not all trying to do it so at the same time as this will cause congestion at the local base station.

7.35. Conclusion

In conclusion, developers should consider the guidelines mentioned throughout this section.

One thing to bear in mind is that the general principles apply to both data transfers and signalling that occurs due to network accesses and detaches.

There should be randomization, sensible back-off algorithms and non-infinite retries for all types of network use - both data transfer and network accesses.

Appendix B is highly relevant to this and details some common Network Cause codes and how devices should behave upon receiving them.

Section 8 that follows provides some tips for Application testing, Section 9 has some relevant information on Wireless Network technology and Section 10 has some M2M application specific information.

8. APPLICATION TESTING

8.1. Importance of Testing:

One of the most important aspects of app development is testing.

Testing prior to deployment has obvious advantages -

- Less chance of significant errors being found by users
- Ability to tune app performance
- Ability to observe network performance
- Ability to tune device battery performance
- Ability to observe any unintended behaviour and rectify

8.2. General Testing Guidelines

- Field testing of the app should be performed rather than just using a simulation tool
- App should be tested on all networks it could end up on i.e. 3G, 4G and Wi-Fi. Performance differences should be noted and code optimization done as a result
- App should be tested in areas with poor coverage / network connectivity to see how robust / resilient the app is to poor network conditions and connectivity and whether further app optimization is required to account for these issues.
- App should be tested in peak times (for data apps typically around 9pm on a weekday) to see how it performs under busier network conditions where there may be additional delays in responses from network
- App should be tested against any user configurable settings that are possible i.e. if user can set times for push to be disabled – check that these actually are disabled at the set time, or if uploading of photos is only permitted on Wi-Fi, make sure that this occurs.
- Monitor battery usage with the app (most operating systems provide battery monitoring tools and app stores have battery monitoring apps)
- Monitor data usage of the app. Check that it is consistent with expectations
- Ensure app server responds as expected in all the different test cases and network conditions as possible
- Try as many different combinations of usage cases as possible to detect any unintended UI behaviour
- Try to test using a variety of device models to ensure your app caters properly for different screen types, resolutions, device types (tablets vs. handsets)
- Consider using an external test house that has a large selection of devices, or use crowdsourcing web sites to get beta testers using a variety of devices
- Use User Agent switchers on desktop browsers to see how mobile web sites will be rendered on different devices.
- Check app behaviour with no network connectivity (offline). Does it provide access to functionality that doesn't require network connection or does it hang?
- Check that app performs as expected when in airplane mode or without a sim (i.e. no network connectivity)
- Test performance after network connectivity is restored after losing it

- Check that app performs as expected when there is a cabled connection to computer (tethered)
- Check interaction with peripherals such as SD card, Bluetooth , device camera, and GPS
- Check that app performs as expected with above peripherals on/off/connected/not connected as appropriate.
- Check that app gives user appropriate indication of network connectivity, error conditions and ensure that there is a non-blocking UI. e.g. If no network connectivity the device doesn't flash up an error message and get stuck but rather allows the user to continue with activities within the app that don't require network activity.
- Ensure Security is tested. e.g. user authentication works as expected
- Does the app correctly handle interruptions e.g. from incoming calls.
- CPU usage of app is not excessive (most operating systems provide CPU monitoring tools)
- That the app performs as expected when the device is multitasking with other apps or when the app is background mode.

8.3. M2M Specific Testing Guidelines

Given the remote locations and rugged environments that M2M solutions can exist in testing must be all the more rigorous.

In addition to the testing described above the following additional testing is required for M2M:

- Test that FOTA solution works – test that both the application software can be updated and verify that firmware upgrade solution will work.
- Test that device is sufficiently rugged for planned deployment/usage. Is the device sufficiently hardened for the expected environment?
    - Test for heat, vibration, moisture, UV exposure, and generally adverse weather
- Test for failure conditions e.g. for power metering application, what happens if there is a power failure?
- Test that device over the air diagnostics work
- Verify that network reacquisition and retry algorithms function in a finite and non-aggressive way
- Test IPv6 functionality

8.4. References

http://www.appqualityalliance.org/online_testing_tool_and_best_practice_update

http://www.appqualityalliance.org/resources

9. WIRELESS TECHNOLOGY INFORMATION

9.1. RRC State Diagram

Network signalling is related largely to the RRC (Radio Resource Control) state transitions of the wireless device.

It is important not to have unnecessary network signalling as this decreases the networks performance in terms of responsiveness and the amount of traffic it can handle for all users.

It can therefore make an app appear slow or unresponsive and excessive state changes can quickly drain a devices battery.

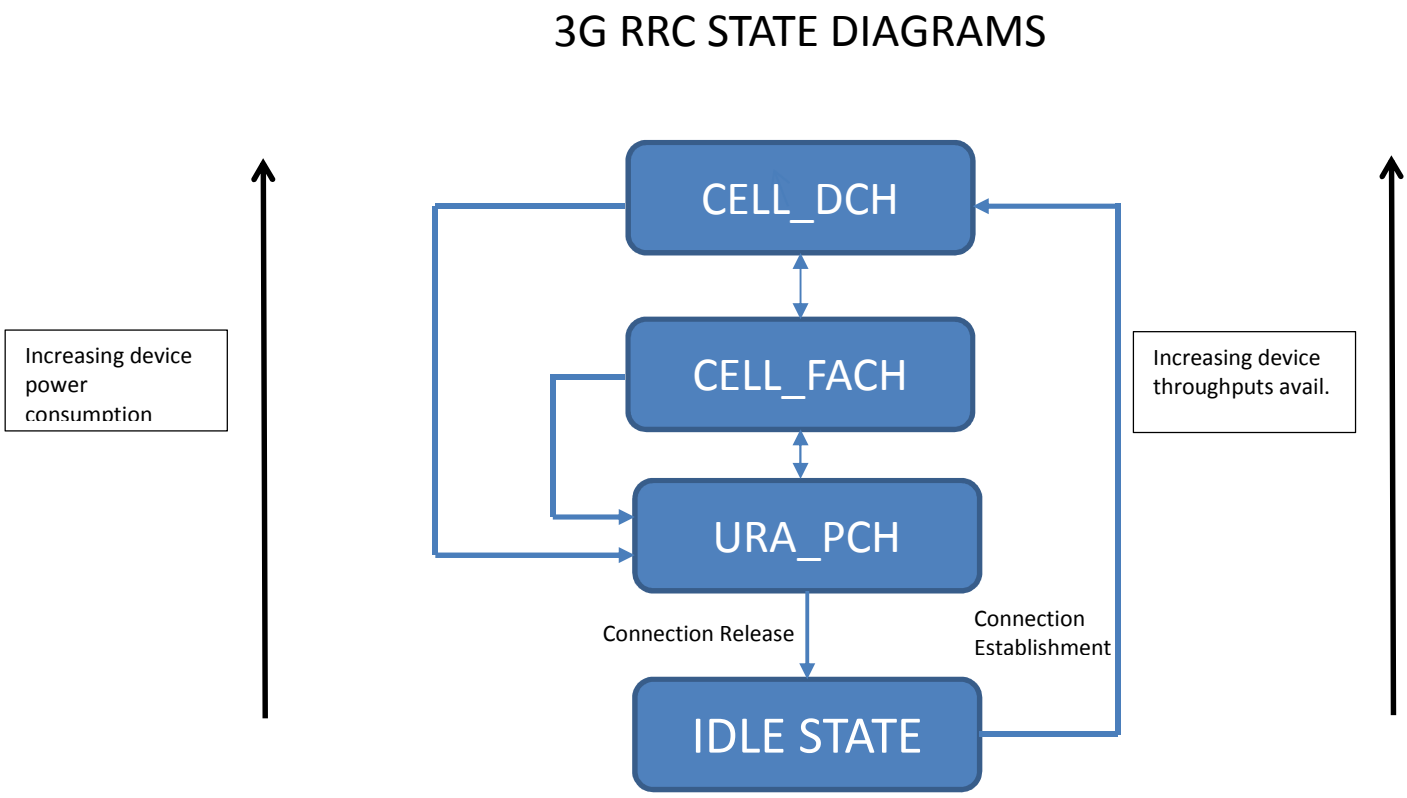
There are some app behaviours that cause unnecessary RRC state changes leading to too much signalling, which should be avoided, including –

- Having heart beats/pings to maintain always on activity
- Poor Network reattachment algorithms.
- Constant polling applications

This section of the document aims to give developers strategies and techniques to reduce unnecessary network signalling – and make their app "network friendly".

RRC State Machine

The RRC state machine describes how a wireless device (3G/4G) connects to the radio network in a variety of different states – which have different levels of connectivity, power consumption and throughputs.



This discussion will concentrate on 3G and 4G networks as these networks provide a better user experience – particularly 4G as this network has many advantages for users, developers and Telstra as an operator, over the older 3G technology.

3G

As the device moves up the RRC states stack it consumes increasing amounts of energy, and the throughput available from the device increases. Note that these state changes require network signalling. There are also network controlled timers that control the minimum time before a state change can occur. It takes time to change state, and for small blocks of data the time to change state may actually exceed the time to transmit the required data. Therefore it is sensible to batch up small amounts of data into larger blocks.

In 3G there were originally three states - IDLE, CELL_FACH and CELL_DCH. In later networks URA_PCH was added. In the IDLE state the mobile device can receive system information from the network but not send or receive any user data. There is no internet connectivity for the user. Power consumption is lowest in IDLE state compared to the other states.

In the CELL_FACH state, the mobile device can send low speed data on what is essentially a shared channel for all mobiles in the same cell area. Internet connectivity is available. This state consumes about 20 times the power of the IDLE state. When a larger amount of data is needed to be sent the mobile device will move to the CELL_DCH state where it now has a dedicated channel and can use higher speeds. Cell_DCH state uses approximately twice as much power as the CELL_FACH state.

In URA_PCH state the device is effectively in sleep mode but it is still being tracked by the network at URA level (UMTS routing area - group of cells). The logical connection between device and network is still up, so that device app is still effectively connected to its app server.

Not all networks support the URA state but this state is active in Telstra's network.

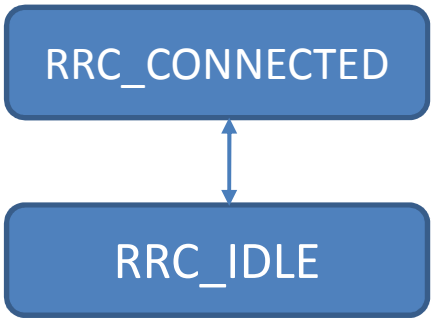
The advantages of the URA state is that power consumption is almost as low as the IDLE state but the logical connection between the UE and the network, or the app and its server is still up. If data communication is required then the device will move to CELL_FACH or CELL_DCH state depending on the amount of data required to be sent or received. The network will automatically switch the device down to the URA state after a period of inactivity (dependant on network settings).

To move from IDLE to CELL_DCH takes many times longer than from URA to FACH or FACH to DCH so the URA state provides improvements in latency.

4G/LTE

LTE or 4G on the other hand has only two main states – RRC_CONNECTED and RRC_IDLE. An app that is in the connected state might typically consume a few watts whilst in IDLE state it will only consume tens of milli-Watts. The RRC_CONNECTED state is the state for data transfer between the network and mobile device. LTE is designed to allow the device to move between these two states very quickly especially when moving from IDLE to CONNECTED. Once in CONNECTED mode a network timer will again determine how quickly the device moves back to IDLE. So again to maximise device battery life it is important to reduce the amount of small data transfers to the network. The recommendation to buffer, collect and forward intermittently is still relevant for LTE.

LTE RRC STATE DIAGRAM



The point of all the above discussion is that it takes time, power and network signalling to switch between these RRC states and so it makes sense to minimize unnecessary state change between IDLE and DCH.

Refer 3GPP TS 25.331 for more information on RRC states for 3G/4G devices.
https://www.etsi.org/deliver/etsi_ts/125300_125399/125331/12.03.00_60/ts_125331v120300p.pdf

3G Fast Dormancy

Note that in 3G networks, there is a feature called Network Fast Dormancy that aims to help reduce battery consumption and reduce network signalling in devices that support this feature. Telstra discourages the use of devices that implement their own proprietary fast dormancy (FD) algorithms as these are not ideal.

4G CDRX

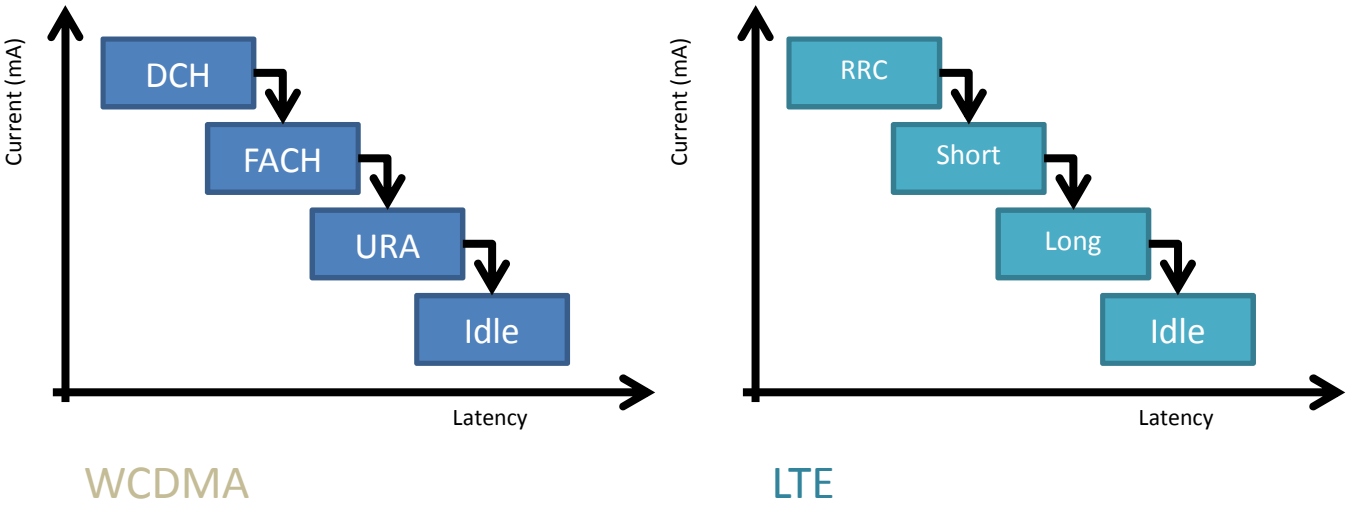
CDRX stands for Connected Discontinuous Reception. It is an LTE device feature that allows the device to have micro sleeps and is controlled by the network. It helps to reduce network signalling load which benefits all users of the network.

It is a feature that reduces device battery consumption while minimizing impact on latency. In Telstra's network CDRX is only invoked after 100ms of inactivity.

It can be considered analogous to FACH/URA in 3G in some ways.

The point to note is that the UE is still RRC connected while in CDRX sleep states.

The diagram below compares 3G and 4G RRC states, and shows CDRX states for 4G



9.2. Data Session Setup

Some brief information regarding data session setup differences between 3G and 4G.

In 3G to establish a data session the device must first attach to the network and then activate a Packet Data Protocol (PDP) Context. PDP context establishes the data connection between device and network and allocates the device an IP address. An initial PDP is known as Primary PDP context and it has a nominal QOS (Quality of Service) info.

M2M developers should consult with their vendor's documentation on relevant high level software commands or low level AT commands to perform these procedures as needed.

Devices can have multiple primary PDP contexts going to different data networks i.e. to different Access Point Names (APNs) i.e. different IP addresses. A mobile device needs an APN to identify the PDN (Packet data network) that it wishes to connect on. It is an address of an internet network to connect to.

There is also the concept of secondary PDP context. A secondary PDP context is associated with a primary PDP context i.e. has the same APN and IP address of the primary context. It is used to have a particular set of QOS settings.

LTE is different to 3G in that it has always on IP connectivity which is established by a **Default** Evolved Packet System (**EPS) bearer** by the Network Attachment procedure. This procedure attaches to the network and sets up a Default EPS bearer – and allocates IP address to the device.

An EPS bearer is analogous to PDP context in 3G.

When a device attaches to the LTE network for the first time it will be assigned a default bearer, which stays all the time and provides the always on IP connectivity. It has a nominal QOS.

If a specific QOS is required then this can be achieved by the network setting up a **Dedicated** Bearer. Currently Telstra does not yet have a specific QoS implementation for different applications / M2M so dedicated bearers should not be used for any purpose. UE initiated QoS is not supported and should not be attempted.

If an application requires a non-standard APN, then a new default bearer to the APN is established.

Developers should discuss any custom APN needs with Telstra via email address mobilitypartner@team.telstra.com

A default bearer remains as long as the UE is attached to the LTE network. A UE can have additional default bearers. Each default bearer has its own IP address.

LTE dedicated bearers must be paired with a default bearer and they use the same IP address as the default bearer.

An LTE device can have up to 18 bearers in total – but one must be a default bearer. A default bearer is needed per APN (and another IP address).

M2M developers need to consider the differences in network connection setup between 3G and 4G when developing their solution e.g. in the case of prompt for password applications on 4G where the device may connect to the network and fail if the correct password has not been previously entered.

9.2.1. References:

Specification Links on the topic:

GPRS Service Description Stage 2 TS 23.060

http://www.3gpp.org/ftp/Specs/html-info/23060.htm

GPRS Enhancements for E-UTRAN access TS 23.401

http://www.3gpp.org/ftp/Specs/html-info/23401.htm

3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3 TS 29.274

http://www.3gpp.org/ftp/Specs/html-info/29274.htm

Blog links on the topic:

http://lteuniversity.com/get_trained/expert_opinion1/b/chrisreece/archive/2008/12/11/pdp-context-vs-eps-bearer-a-battle-of-the-data-session-setups.aspx.

http://www.lteandbeyond.com/2012/01/lte-attach-procedure.html

http://4g-lte-world.blogspot.com.au/2012/05/default-bearer-dedicated-bearer-what.html

http://www.eventhelix.com/lte/attach/lte-attach.pdf

http://wired-n-wireless.blogspot.com.au/2009/03/dedicated-bearers-in-lte.html

http://wired-n-wireless.blogspot.com.au/2009/04/lte-ue-initial-attach.html

10. M2M SPECIFIC DEVICE CONSIDERATIONS

Developers need to carefully consider certain aspects of integration issues of embedded wireless modules into end user device/ application.

10.1. Regulatory Issues

When integrating a module into a device, certain regulatory requirements need to be met.

These include:

- Compliance with ACMA Regulatory safety requirements Radio communications (Electromagnetic radiation – Human exposure) Standard 2014.

  Refer: https://www.legislation.gov.au/Details/F2014L00960

- RCM - Regulatory Compliance Mark for the product (and embedded module). The RCM indicates a device's compliance with applicable ACMA technical standards — that is, for telecommunications, radio communications, EMC and EME. The RCM mark is replacing the previous C-Tick, A-Tick and RCM marks.

  Refer: http://www.acma.gov.au/theACMA/device-compliance-levels

- Provision of SAR (Specific Absorption Rate) reports or reason for exemption

  Refer: http://www.amta.org.au/sar

10.2. Antennas

- 3GPP defines how many antennas each LTE category shall support, and devices shall comply to these requirements
- Reducing the number of antennas has an negative impact on the received signal which impacts customer experience
- Antennas should support all frequencies bands supported by both the module and network
- Antenna should be optimized to suit the frequency bands to be used by the device.
  - For 3G devices they should be optimized for band 5 (850 MHz)
  - For 4G devices they should be optimized for all the bands they support and particularly bands 3 (1800 MHz) & 28 (700MHz)
- When installing remote equipment, directional antennas should be oriented toward the strongest received signal (in most cases)
- Antennas should be configured and placed to optimise radio performance within the physical constraints of the end product.
- Since the antenna is a key component for enabling wireless technology, the critical design parameters such as mounting location and space allocation should be considered in the early phases of the product development process in order to maximise its performance.
- Grounding and metal clearance near the antenna feed points shall provide consistent radiated performance in production. Pattern shape is another antenna performance parameter that should not be overlooked. Omni directional type pattern is much more desirable than the

directional one. Parasitic coupling to metal structures near the antenna can alter its pattern shape and operational bandwidth.

- To minimise interference, it is preferred that the positions of the antennas are as far as possible from any digital circuitry that generates high frequency noise (that is, high speed clocks).

10.3.RF Shielding / Interference Mitigation

Integration of wireless modules into end products shall minimise any possible interference with other components.

Radio Interference with Device Components:

- Interference in the end device is typically sourced from circuits such as CPU, memory chips, video circuits and other components generating high frequency noise which has the potential to couple into the radio through the antenna or other conducted paths. Such interference affects the overall wireless performance and User experience.

- The Embedded Device design must consider the integration of a 3G/4G radio module and minimise interference between the host system components and the 3G/4G embedded module and associated antenna subsystem.

Coexistence with other wireless technologies

- Attention needs to be paid to coexistence with other wireless technologies likely to be resident in the same unit. There should also be no interference between the 3G/4G radio interface and other radio interfaces present in the product.

10.4. Device Identification Codes

- The IMEI ranges of the wireless modules embedded into the end product must to be submitted to Telstra on a regular basis (for Telstra Certified Devices).
- IMEI – International Mobile Station Equipment Identity – is a unique identity code used by network operators to distinguish between devices on our network.
- Note Telstra prefers that Routers integrating modules to have a different TAC code to the embedded module - this enables us to separately identify devices using the same module should any device issues arise in the future. If this is not done then Telstra prefers a separate IMEI series within the TAC range to allow devices to be readily identified.
- TAC code refers to the subset of the IMEI that details the manufacturer and model of a device.
- The TAC code is the first eight digits of the 15 digit IMEI code. TAC stands for Type Allocation Code.

10.5. Ruggedness

- Ensure device is appropriate hardened / rugged against the elements for remote field deployment as appropriate.
- Ensure device has sufficient protection to prevent easy theft of UICC (SIM)
- Device should have a sealable, tamper proof enclosure

10.6. Algorithmic Robustness

- Similar to the issues discussed in section 7.22. M2M application should be coded in such a way that it shall not be totally dependent on any network responses for its flow or progress.
- Control loops should be robust all control loops should be designed for robustness appropriate to a wireless network.
- In coding the M2M application, the design should assume that each and every control function can fail to be received without warning and consequently the system should be designed such that the system should fail safe if a critical control function is not received
- As previously discussed the system should have graceful retry mechanisms.
- The system should only panic after a length of time reasonable for the application before deciding to fail where a control function is not received.

The benefits for taking this extra care in system design are:

- Greater system level availability
- Greater reach in terms of coverage
- Less stress on the network and
- Cheaper implementation.

10.7. References

Compliance with ACMA Regulatory safety requirements Radio communications (Electromagnetic radiation – Human exposure) Standard 2014.

https://www.legislation.gov.au/Details/F2014L00960

Provision of SAR (Specific Absorption Rate) reports or reason for exemption

http://www.amta.org.au/sar

11. APPENDIX A – TELSTRA WIRELESS AND M2M RELATED PRODUCT INFORMATION

11.1. Telstra Mobility Partners

Telstra has dedicated M2M product and solution teams to assist developers with integrating M2M solutions within our network. Email mobilitypartner@team.telstra.com for assistance.

11.2. Telstra IoT Offerings

https://www.telstra.com.au/business-enterprise/solutions/internet-of-things

11.3. Telstra Wireless M2M Control Centre information

https://www.telstra.com.au/business-enterprise/solutions/internet-of-things/platforms-and-services/sim-subscription-management

11.4. Telstra Mobile Assets and Workforce Enterprise Solutions

https://www.telstra.com.au/business-enterprise/solutions/mobility-solutions

11.5. Telstra Wireless Managed Data Networks – Wireless WAN

https://www.telstra.com.au/business-enterprise/download/document/business-mdn-wireless-wan.pdf

11.6. Telstra IP VPN information

https://www.telstra.com.au/business-enterprise/solutions/network-services/connectivity/vpn-service

11.7. Telstra Enterprise Support Contacts

https://enterprise-support.telstra.com.au/t5/tkb/communitypage

11.8. Telstra Mobile Phones

https://telstra.com.au/mobile-phones

11.9.Telstra Mobile Coverage

https://www.telstra.com.au/coverage-networks/our-coverage

11.10. AT Command Reference

3GPP Specification TS 27.007 specifies AT commands for controlling cellular modems.

Vendors often add their own proprietary extensions to these. For these extensions refer to the manufacturer's documentation.

http://www.3gpp.org/ftp/Specs/archive/27_series/27.007/

12. APPENDIX B – NETWORK CAUSE CODES AND DEVICE BEHAVIOUR

The network has a variety of cause codes that it can send to the device in response to device requests that indicate a reason for failure of the request.

The device should pay attention to these cause codes and behave accordingly.

In a few cases the industry (3GPP) specifications and network timers specify the retry algorithm's behaviour but not always completely. Thus in these cases and where not specified at all, the behaviour is left to the device manufacturer/integrator. As discussed earlier, if retries are required, they should not be aggressive and infinite in nature.

Some cause codes indicate trouble with a user's service subscription and retries are pointless (once issue is confirmed as not a one off) – in this case the user/developer needs to confirm their service subscription status with Telstra. For 3G these cause codes are described in 3GPP specification 24.008 annexes.

Refer http://www.3gpp.org/DynaReport/24008.htm
There are three main categories of error codes. Those related to Mobility Management (MM), Call Control (mainly applicable to voice calling) and Data Session Management (SM).

Mobility Management (MM) includes causes related to:

- MS Identification,
- Subscription Options
- Network Failures/Congestion/Authentication Failures
- The nature of request
- Invalid messages
- GMM = related specifically to data

Call Control (CC) causes are grouped into:

- Normal Class
- Resource unavailable class
- Service / Option not available
- Service Not implemented class
- Invalid Message
- Protocol error
- Interworking issues

GPRS/Data Session Management (SM) [ESM used for LTE] are divided in to subgroups of causes related to:

- Nature of request
- Invalid Messages

Similarly for LTE, the cause codes are described in 3GPP Specification TS 24.301. Refer http://www.3gpp.org/DynaReport/24301.htm .
There are some additional codes specific for LTE, but the main encountered issues are the same for both technologies.

The table below summarizes some commonly seen codes, their meaning and suggested device behaviour.

Cause Code	Meaning	Scenario where it may occur	Proposed Device Behaviour or Developer Action Required
8	Operator Determined Barring	Service is barred	Don't retry - contact Telstra Support to ascertain why service is barred.
26	Insufficient resource	Network has insufficient resources e.g. Congested	Since network is congested wait and try again. The device shall not enter an endless retry mechanism. After each rejection, the device shall introduce a backoff timer (recommended 12 minutes). We suggest doubling this backoff timer after each rejected request and ultimately stop the requests after a period of time (recommended 2 days). Device must still comply with relevant 3GPP standards and obey applicable network timers Consider sending data during off peak times (midnight - 6am)
27	Missing or unknown APN	Incorrectly configured device settings for APN profile	Confirm with Telstra that you have correctly configured and are using the correct APN for the application After each rejection, the device shall introduce a backoff timer (recommended 12 minutes). We suggest doubling this backoff timer after each rejected request and ultimately stop the requests after a period of time (recommended 2 days).
28	Unknown PDP address or PDP type	3G specific analogous to 54 for LTE. Possibly due to incorrect internet destination configured in device	The device shall not enter an endless retry mechanism. After each rejection, the device shall introduce a backoff timer (recommended 12 minutes). We suggest doubling this backoff timer after each rejected request and ultimately stop the requests after a period of time (recommended 2 days). Device must still comply with relevant 3GPP standards and obey applicable network timers. Developer should check device configuration and internet settings Seek Telstra technical support if error persists

29	User Authentication Failed	Service or SIM error	If this occurs more than once, then contact Telstra Support to confirm service subscription is correct and to investigate the issue.
30	Activation rejected by GGSN, Serving GW or PDN GW	This error occurs if the device/application is requesting a service that is not supported by the network.	Again device shall not endlessly try - but rather stop and provide meaningful error to the user. Developer to investigate what service is being requested and confirm with Telstra whether the service is supported and if not what an equivalent alternative service would be suitable. Again device shall not endlessly try - but rather stop and provide meaningful error to the user
31	Activation rejected, unspecified	Possibly due to the requested service option not being subscribed to or other reason	The device shall not enter an endless retry mechanism. After each rejection, the device shall introduce a backoff timer (recommended 12 minutes). We suggest doubling this backoff timer after each rejected request and ultimately stop the requests after a period of time (recommended 2 days). Device must still comply with relevant 3GPP standards and obey applicable network timers. Contact Telstra for support
32	Service option not supported	Occurs when the network doesn't support the service option.	Again device shall not endlessly try - but rather stop and provide meaningful error to the user. Developer to investigate what service is being requested and confirm with Telstra whether the service is supported and if not what an equivalent alternative service would be suitable. Do not automatically retry. May occur in a roaming network.
33	Requested service option not subscribed	Occurs due to the requested service option not being subscribed to.	The device shall not enter an endless retry mechanism. The device will not retry unless power cycled or a device setting is altered. Contact Telstra for support

34	Service option temporarily out of order	Likely due to network fault	Given that this is due to a network issue that is temporary trying again is reasonable. However the device shall not enter an endless retry mechanism. After each rejection, the device shall introduce a backoff timer (recommended 12 minutes). We suggest doubling this backoff timer after each rejected request and ultimately stop the requests after a period of time (recommended 2 days). Device must still comply with relevant 3GPP standards and obey applicable network timers If problem persists contact Telstra for support.
38	Network Failure	Likely due to network outage	Given that this is due to a network failure it is important not to repeatedly retry as this will make it difficult for the network to recover. Suggest backing of for tens of minutes before retrying. As always the device shall not enter an endless retry mechanism. After each rejection, the device shall introduce a backoff timer (recommended 12 minutes). We suggest doubling this backoff timer after each rejected request and ultimately stop the requests after a period of time (recommended 2 days). Device must still comply with relevant 3GPP standards and obey applicable network timers If problem persists contact Telstra for support.
50	PDP type IPv4 only allowed	Will occur if device requests an IP protocol type (e.g. IPv4v6) that is not allowed by the network or user subscription e.g. if requests IPv6 bearer when they aren't supported	Device shall set up an IPv4 bearer and not request IPv6
51	PDP type IPv6 only allowed	Will occur if device requests an IP protocol type (e.g. IPv4v6) that is not allowed by the network or user subscription e.g. if requests IPv4 bearer on IPv6 Single Stack network	Device shall set up a IPv6 bearer and not request IPv4 bearer
52	Single address bearer allowed		If device requests an IPv4v6 PDP and network sets the PDP type to IPv6 (or IPv4) with cause code #52 (single address bearer allowed), the device shall use the allocated IP address from the network. The device can subsequently request another PDP context activate for the other bearer if it requires dual stack connectivity if the network does not support IPv4v6 on one bearer.

53	ESM Information not received		
54	PDN connection does not exist	LTE specific analogous to 28 for 3G. Possibly due to incorrect internet destination configured in device	Developer should check device configuration and internet settings

13. APPENDIX C – APN TIMEOUTS

The following timeout info applies to all APNs that undergo NAT (Network Address Translation).

Inactivity timeout for general traffic:

TCP 30 min
UDP 2 min
ICMP 4 sec
DNS 5 sec

These timeouts are subject to change.

Extranet services (that do not undergo NAT) have the same timeouts applied TCP/UDP/ICMP/DNS on the Stateful firewall.

14. CONTACT FOR ENQUIRIES AND PROPOSED CHANGES

If you have any questions regarding this document or would like to suggest an improvement, contact:

Name	Devices & New Technology Team, Technology Strategy Commercial Engineering Networks & IT
Email	WADG-feedback@team.telstra.com

15. GLOSSARY

Abbreviation / Term	**Definition**
2G	2nd Generation Wireless Network based on GSM technology
3G	3rd Generation Wireless Network based on WCDMA technology
3GPP	3rd Generation Partnership Project
4G	4th Generation Wireless Network based on LTE technology standards
API	Application Program Interface
APN	Access Point Name
APP	Application
BSIC	Base Station identity Code
CA	Carrier Aggregation
EAS	Exchange Active Sync
Ec/Io	Measure of interference in 3G WCMA/UMTS systems
FOTA	Firmware Over The Air
GSM	Global System for Mobiles
IMEI	International Mobile Equipment Identity
IoT	Internet of Things
LTE	Long Term Evolution
MB	Megabyte
M2M	Machine to Machine

MCC	Mobile Country Code
MNC	Mobile Network Code
OS	Operating System
OMA	Open Mobile Alliance
OTA	Over The Air
OCT	Our Customer Terms
PCI	Physical Channel Indicator
PSM	Power Saving Mode
RRC	Radio Resource Control
RSCP	Received Signal Code Power
RSRP	Reference Signal Received Power
RSRQ	Reference Signal Received Quality
RSSI	Received Signal Strength Indicator
SDK	Software Development Kits
SMS	Short Message Service
SSL	Secure Sockets Layer
T&Cs	Terms and Conditions
TLS	Transport Layer Security
UMTS	Universal Mobile Telecommunications System
WBB	Wireless Broadband
WCDMA	Wideband Code Division Multiple Access
WDA	Wireless Devices & Applications
WDT	Wireless Device Technology
UI	User Interface
UX	User Experience

16. DOCUMENT CONTROL SHEET

The purpose of this section is to capture all changes made to the content of document.

Issue No	Issue Date	Nature of Amendment
Version 1	Circa 2005	Initial Document
…	…	…
Version 5	1st Dec 2006	New, rewritten, reformatted version
Version 5.3.3	12th February 2010	Updated network performance information & specifications. Minor corrections & clarifications to network information sections. Repositioned section 10. TELSTRA'S 3G UMTS & GPRS NETWORK ARCHITECTURE for more logical document flow. Updated Telstra web site URLs and removal of obsolete references, particularly sections 18 & 19
Version 6 Draft 1.0	Jan 2013	Completely new version of guidelines – to address both smartphone and M2M applications. Simplification of document
Version 6 Draft 2.0	Mar 2013	Updated based on feedback from Draft 1 reviewers
Version 6 Draft 3.0	April 2013	Updated based on feedback from Draft 2 reviewers
Version 6 Issue 1.0	11th June 2013	Includes changes required for Policy 61 Approval and reformatting to suit new branding template
Version 7 Issue 1.0	30th June 2014	Miscellaneous typo corrections Updated dead/changed URL links Updated references Updated IPv6 sections Updated Telstra LTE spectrum information Added appendix – Network Cause Codes and Device Behaviour
Version 8 Issue 1.0	23rd Mar 2017	Miscellaneous typo corrections Updated dead/changed URL links Updated LTE bands Updated Carrier Aggregation info Removed irrelevant 2G references Added IOT information Updated cause codes to align with MSRs
Version 9 Issue 1.0	21st December 2018	Changed dead URL links Updated network technology and features table Updated LTE device category tables Added GSMA IoT security guidelines reference