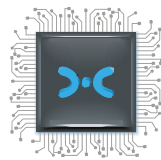


AUTO DISCOVERY REMOTE CONTROL

# ADRC TECHNICAL OVERVIEW



THE INTERNET OF THINGS



## DRAFT 1

Re-named to ADRC Technical Overview

Converted from InDesign to Word.

Updates based on ADRC Developer Guide

VERSION 1.5 // JUNE 2017

xped

# Contents

1	About this document .....	1	3.3.2	Foreign device on-boarding overview .....	21
1.1	Document purpose .....	1	3.4	Updating a Thing .....	21
1.2	Audience .....	1	3.5	Controlling and monitoring a Thing locally .....	23
1.3	Approach .....	1	3.6	Controlling and monitoring a Thing remotely .....	25
1.4	Terms and definitions .....	1	3.6.1	Setting up remote access .....	25
1.5	Reference documents .....	2	3.6.1.1	Certificate Authority .....	25
1.6	Document change history .....	2	3.6.1.2	Activate an IoTG account .....	25
1.7	Contact information .....	2	3.6.1.3	Authorise remote access for a user .....	26
2	Introduction .....	3	3.6.2	Accessing a Thing remotely .....	28
2.1	What is ADRC? .....	3	3.6.2.1	Remote access summary .....	28
2.2	ADRC user experience .....	7	3.6.2.2	Remote access detailed steps .....	29
2.3	ADRC key concepts .....	12	3.7	Security summary .....	31
3	How ADRC works .....	13	3.7.1	On-boarding .....	31
3.1	ADRC components and functions .....	13	3.7.2	Internet of Things Gateway (IoTG) .....	31
3.1.1	ADRC server software .....	13	3.7.3	Thing .....	31
3.1.2	The ADRC IoT stack .....	13	3.7.4	PAN key .....	32
3.1.3	Resource Modelling Language (RML) .....	14	3.7.5	Device browser (DeB) .....	32
3.1.4	Resource Control Protocol (RCP) .....	15	3.7.6	Remote access .....	32
3.1.5	The device browser app (DeB) .....	15	4	Additional services .....	33
3.2	Implementing ADRC .....	17	4.1	Ontology .....	33
3.2.1	Internet of Things Gateway (IoTG) .....	17	4.2	Cloud .....	33
3.2.2	Smartphone with DeB .....	18	4.3	Foreign protocol integration .....	33
3.2.3	ADRC-enabled Thing .....	19	4.3.1	Virtual gateway .....	33
3.3	On-boarding a Thing .....	20	4.3.2	Protocol adapters .....	34
3.3.1	ADRC on-boarding overview .....	20			

# Figures

Figure 1: User perspective .....	4
Figure 2: ADRC Components .....	5
Figure 3: Internet of Things Gateway (IoTG) .....	17
Figure 4: Device browser (DeB).....	18
Figure 5: ADRC-enabled Thing.....	19
Figure 6: On-boarding overview.....	20
Figure 8: Using a Thing locally.....	23
Figure 9: Setting up remote access for a user .....	26
Figure 10: Remote access summary .....	28
Figure 11: Remote access detailed steps.....	29

---

*Disclaimer: Whilst every reasonable effort has been made to ensure the accuracy of the information provided by Xped. Xped shall not be held liable for any inaccurate information of any nature, however communicated by Xped. The contents of this document are subject to change without notice.*

---

# 1 About this document

## 1.1 Document purpose

This document provides an end-to-end overview of Auto Discovery Remote Control (ADRC), which is the Xped platform for the Internet of Things.

This document covers the concepts behind ADRC, the user experience, how to implement ADRC, the ADRC system design and an explanation of how the individual ADRC components interact with each other.

## 1.2 Audience

This document is for all parties interested in understanding the ADRC platform, from hobbyists and students to developers and chip manufacturers.

To cater for different audiences who require varying levels of detail on how ADRC works, the following ADRC functions are presented in both summary format and as detailed steps:

- Accessing a Thing remotely – section 3.6.2.1 provides a summary of the steps involved and section 3.6.2.2 documents the detailed steps

## 1.3 Approach

The examples given in this document show a user with a smartphone connecting and controlling electronic devices in the home environment.

[The smartphone could be replaced by any smart device that has the required device browser installed.](#) The smart mobile device could be a tablet, laptop, smart watch and so on. Note: The smart device requires either an NFC or a BLE proximity connection capability to on-board a Thing. The examples given in this document are for Things on-boarded by an NFC tap.

At time of writing, Apple has locked the NFC functionality on the iPhone for exclusive use with mobile payment services. However, iPhone users can still participate in ADRC because iPhone users will have an additional menu option in the device browser to initiate any ADRC functionality that would normally be initiated automatically via NFC.

The smartphone could also be replaced by an automated client in a machine-to-machine environment.

The approach shown for implementing and using electronic devices in the home is the same approach that can be used in any environment.

The examples given are for electronic devices using the 802.15.4 wireless personal area network connectivity standard. Manufacturers could choose to offer other connectivity options.

## 1.4 Terms and definitions

This document introduces a number of terms that are defined in a separate document entitled *ADRC Glossary*.

## 1.5 Reference documents

Document name	Description
ADRC Glossary	Defines the terms used across all ADRC documents.
ADRC Developer Guide	A how-to guide for developing RML applications.
Resource Modelling Language Reference Manual	A language reference for developers who intend to write files in the Resource Modelling Language
Resource Control Protocol Interface Design Documents	<p>The reference specification that defines the Resource Control Protocol.</p> <p>Part 1: Host format used between the Internet of Things Gateway and the device browser or ADRC client</p> <p>Part 2: Wire format used between the Internet of Things Gateway and the Thing</p>

## 1.6 Document change history

Document version	Publication date	Changes
Version 1.5	30 June 2017	<p>Updated to reflect that both BLE and NFC are supported for proximity connections</p> <p>Updated to reflect that both the ADRC server software in the IoTG and the ADRC IoT stack in the Thing are licensed Xped technologies</p>

## 1.7 Contact information

For additional information and queries, email [info@xped.com](mailto:info@xped.com).

## 2 Introduction

The Internet of Things (IoT) is about connecting electronic devices (Things) to each other so these Things can communicate and exchange data. It is also about the interaction between people and Things. Things could be devices and appliances around the home, such as lights, a television, a sprinkler system, a security system and so on. Things could be equipment and infrastructure in an office, hospital or factory. Things could be anything and everything that has an on/off switch.

Currently, a user can control some Things via hardware remote controls or via apps. However, each Thing has its own remote control or app, each with a different user interface and this makes it both cumbersome and challenging for the user to connect and use each Thing.

Xped's Auto Discovery Remote Control (ADRC) platform solves this problem by simplifying the way a user interacts with the multitude of electronic Things that are in a user's everyday environment.


ADRC allows a user to connect all their electronic Things into a personal area network (PAN) within the home by simply tapping each Thing with the user's smartphone. Once the Things are connected, the user can control and monitor all of these Things from a single app on their smartphone.

The Xped ADRC solution 'makes technology easy again'.

### 2.1 What is ADRC?

Xped's ADRC platform is end-to-end technology that enables all kinds of Things from different manufacturers to work together. It allows a user to interact with any ADRC-enabled Thing in a familiar way, irrespective of the Thing's type, brand or location.

A Thing is ADRC-enabled by the manufacturer loading the ADRC functionality into the Thing's microchip at manufacturing time. Each Thing, no matter how complex, is connected and then controlled in exactly the same way as every other Thing. ADRC has been developed specifically for the unique requirements of Things rather than trying to retro-fit web technology into Things. This provides a solution that works for the simplest Thing, such as a small sensor, right up to a complex system, such as a car.

A user connects a Thing into the personal area network (PAN) by tapping their NFC/BLE-enabled smartphone on the touchpoint logo  on the ADRC-enabled Thing and giving the Thing a nickname. This is known as on-boarding a Thing. Many users are already familiar with tapping a contactless payment card on the contactless logo on a payment terminal to pay for goods.

There are a great many Things available using other technologies such as Z-wave, BLE, ZigBee and others that are popular especially in the smart home market. These *foreign things* can also be brought under the control of ADRC. They can be added to the device browser app and controlled just like a native ADRC device. This is done by developing a protocol adapter (software) to bridge to the foreign technology and make its devices look like an ADRC device.

Once a Thing has been on-boarded, the user controls the Thing using an app on their smartphone. This app is called the device browser app (DeB).

DeB communicates via Wi-Fi with a small hardware device called the Internet of Things Gateway (IoTG) and the IoTG communicates with a Thing, typically over an 802.15.4 low-data-rate personal area network (PAN) channel.

An IoTG may be connected to the Internet to allow the Things attached to it to be controlled from anywhere in the world.

The user perspective is shown in

*Figure 1.*

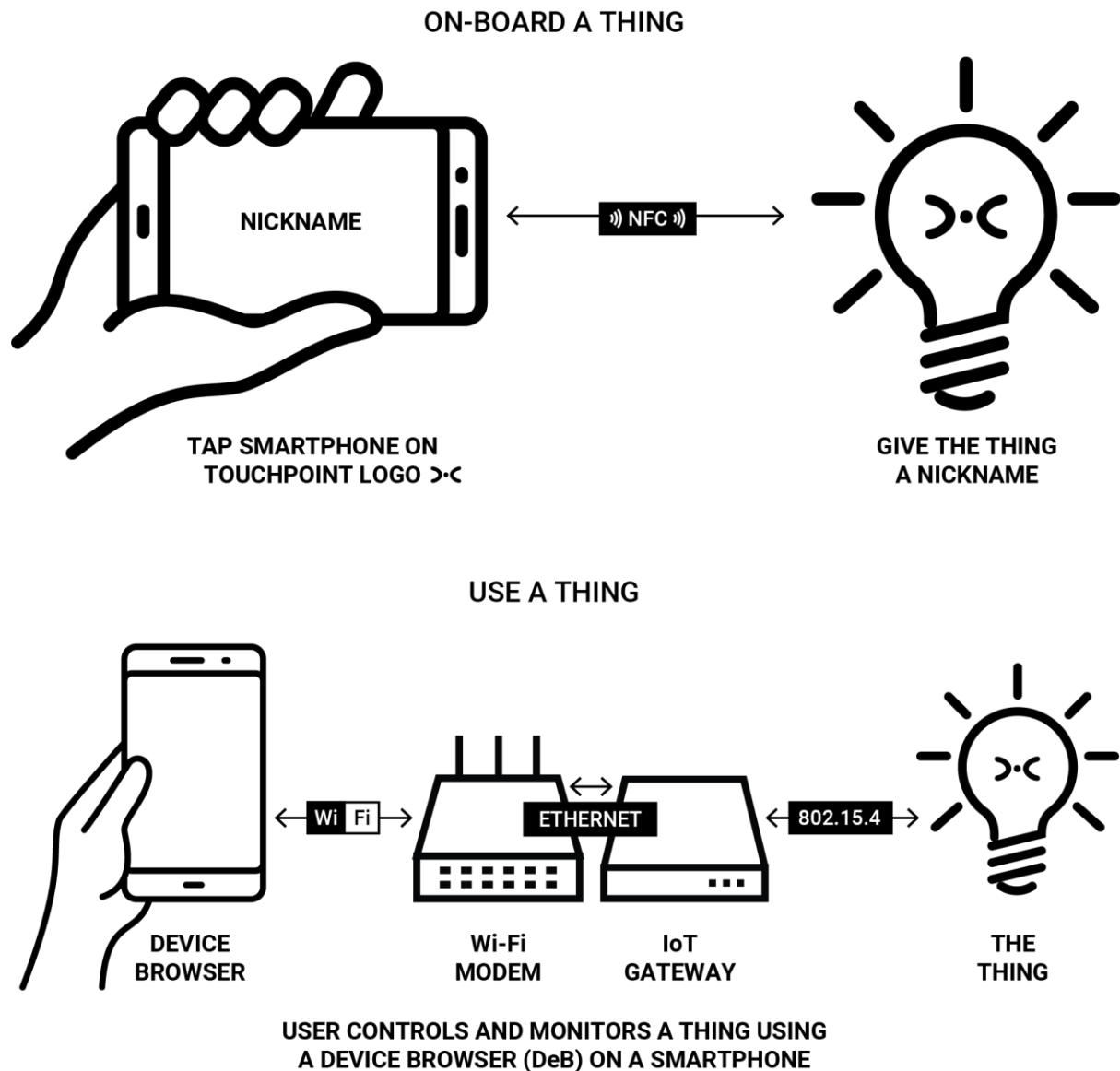


Figure 1: User perspective

**Note:** Wireless LED lighting is used as the main example throughout this document.

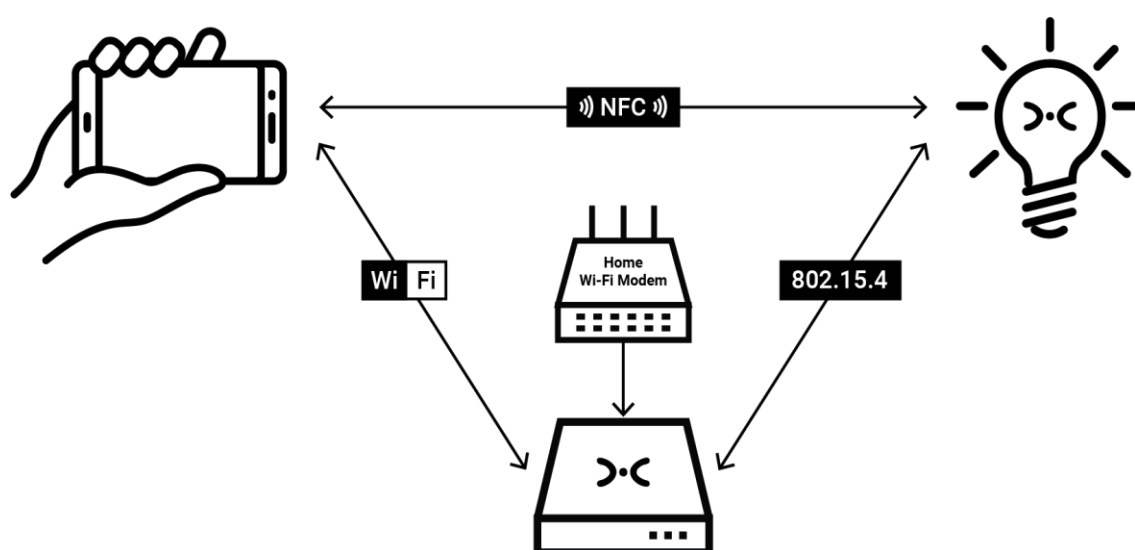
*IMPORTANT: The smartphone communicates directly with the Thing via NFC or BLE to securely on-board the Thing into the personal area network (PAN).*

*Once the Thing is on-boarded, the device browser app (DeB) on the smartphone controls and monitors the Thing by communicating with the Internet of Things Gateway (IoTG) via Wi-Fi. The IoTG then communicates with the Thing via an 802.15.4 (or similar wireless radio technology) personal area network (PAN) channel.*

The ADRC solution is made up of the following software and firmware components:

- The **ADRC IoT stack** that the manufacturer loads onto the microchip in a Thing
- The **Resource Modelling Language (RML)** that the manufacturer uses to describe a Thing and loads onto the microchip in a Thing
- The **device browser app (DeB)** that runs on a user's smartphone and provides the user interface
- The **ADRC server software** that runs on a small hardware device called an Internet of Things Gateway (IoTG)
- The **Resource Control Protocol (RCP)** that is used to transfer commands and RML files between a Thing, an Internet of Things Gateway and DeB

For ease of visualisation from a user perspective, [Figure 2](#) depicts these software and firmware components in relation to the physical components they run on. The physical components are the Thing, the user's smartphone and the Internet of Things Gateway (IoTG).



### SMARTPHONE

- Device Browser (DeB)
  - user interface to on-board Things
  - user interface to control and monitor Things
  - holds MAC address and wireless channel of the IoTG to send to a Thing during on-boarding
  - generates security key to be used by a Thing and the IoTG
- NFC capability to communicate with a Thing
- Wi-Fi capability to communicate with the IoTG locally
- Internet connection to communicate with the IoTG remotely

### IoT GATEWAY

- ADRC server software
  - system services
  - file system for RML files
  - pairing table with the following information for each Thing:
    - : MAC address
    - : wireless channel
    - : PAN id
    - : security key
- 802.15.4 wireless connectivity to communicate with the Thing
- Ethernet connection to Wi-Fi internet modem to communicate with DeB
  - locally via Wi-Fi and
  - remotely via the internet

### THE THING

- IoT stack in a Thing's microchip
  - NFC connectivity to communicate with the smartphone with DeB installed
  - 802.15.4 wireless connectivity to communicate with the IoTG
  - device management
  - communications with the manufacturer's application
  - storage for RML file
- RML file (or files if a Thing has multiple capabilities eg. TV with inbuilt DVD)
  - describes the capabilities and attributes of a Thing
  - is passed to the IoTG during on-boarding of a Thing
- MAC address and wireless channel of the IoTG received from DeB
- Security key received from DeB for secure communications with the IoTG
- Manufacturer's application for the Thing

Figure 2: ADRC Components



The physical components in [Figure 2](#) are:

- An **ADRC-enabled Thing** with:
  - > A microchip loaded with the licensed ADRC IoT stack. The IoT stack provides:
    - NFC/BLE connectivity to the smartphone
    - 802.15.4 wireless connectivity to the IoTG
    - device management
    - communications with the manufacturer's application in the Thing
  - > The Resource Modelling Language (RML) file(s) describing the capabilities and attributes of the Thing
  - > Addressing information (MAC, PAN id and wireless channel) to communicate with the IoTG
  - > The manufacturer's application that provides the functionality for the Thing
- A **smartphone** with:
  - > NFC/BLE to on-board a Thing into the personal area network (PAN)
  - > DeB installed, which is the user interface for controlling and monitoring Things
  - > Wi-Fi for DeB to connect locally to the IoTG to control and monitor Things
  - > An internet connection, if the user wishes to have remote access to the IoTG to control and monitor Things
- An **Internet of Things Gateway (IoTG)**, which:
  - > Connects via an Ethernet connection to the Wi-Fi modem in the home. This allows the user to use DeB to control Things locally via Wi-Fi and remotely via the Internet
  - > Communicates with Things over an 802.15.4 or similar wireless channel
  - > Is pre-loaded with the licensed ADRC server software that:
    - provides the required ADRC system services
    - has a pairing table to store the addressing information and security key for each Thing
    - has a file system to store the RML that describes the capabilities and attributes of each Thing
- A standard home **Wi-Fi modem**, which allows DeB and the IoTG to:
  - > Communicate locally via Wi-Fi to control and monitor Things
  - > Communicate remotely via the Internet to control and monitor Things

**Note:** It is envisaged that an all-in-one Wi-Fi modem with built-in IoTG functionality will become available. An all-in-one industrial version of the IoTG is already available.

These components and their functions are described in more detail in section [3 How ADRC works](#).

## 2.2 ADRC user experience

As shown in

*Figure 1*, the ADRC solution is very simple from an end user perspective. The user simply taps an NFC/BLE-enabled smartphone running DeB on an ADRC-enabled Thing, gives the Thing a nickname and the Thing is ready for the user to interact with.

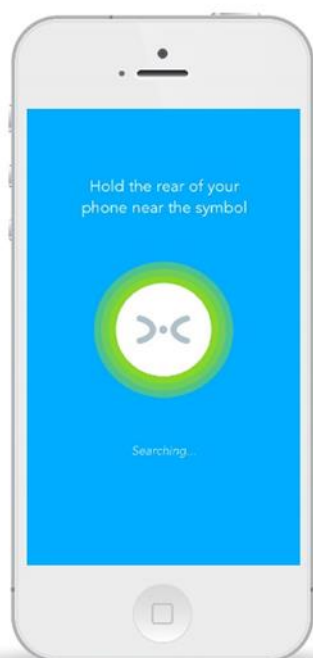
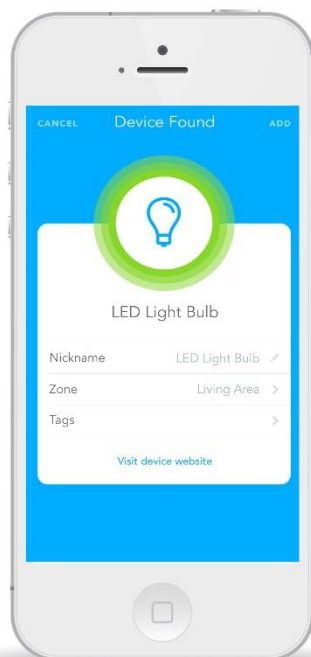
The smartphone running DeB automatically discovers the Thing, sets up a secure wireless network, gathers any resources needed and creates the user interface for the Thing. This is all from one simple tap.

The user steps for setting up the ADRC environment are:

1. Enable NFC or BLE, and Wi-Fi on the user's smartphone
  2. Download DeB to the user's smartphone
    - > DeB can be downloaded from the Apple App Store or from Google Play before the user starts on- boarding Things, or
    - > If the user taps an ADRC-enabled Thing or the IoTG with their smartphone before installing DeB, the user will be prompted to download DeB from the Apple App Store or from Google Play
  3. Plug in an IoTG and connect it to the Wi-Fi network in the home with an Ethernet cable
    - > The IoTG is a small device, generally smaller than a Wi-Fi modem
    - > Optionally, the user can set a PIN on the IoTG by tapping their smartphone on the IoTG. The user will be presented with a screen on their smartphone to set a PIN to restrict access to the IoTG
  4. On-board an ADRC-enabled Thing by tapping the smartphone on the touchpoint logo >< on the Thing
    - > This uses the NFC or BLE capability of the smartphone
- Note:** For iPhone users, the user activates on-boarding mode by selecting a menu item in DeB before tapping on the Thing.
5. When asked by DeB, give the Thing a nickname (optional) to complete the on-boarding process
  6. Once the Thing is on-boarded, control and monitor the Thing using DeB on the smartphone
    - > This uses the smartphone's Wi-Fi connection to the IoTG

The following screens show the interaction between the user and the device browser (DeB) during the on- boarding process.

When the user downloads DeB for the first time, an initial 'blank' screen is displayed to the user.



The user can touch the “+ ADD DEVICE” icon for prompts or simply tap the smartphone on a Thing to on-board it.

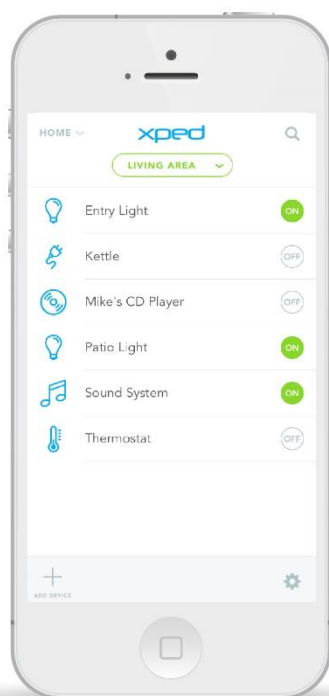
DeB searches for the Thing.  
In this example, it is a wireless LED light.

DeB displays the manufacturer-defined nickname for the Thing to the user. The user can override this nickname. DeB also asks the user to add a zone and tags for the Thing.

The Thing is now on-boarded.

In the following example, the user has changed the nickname to Entry Light

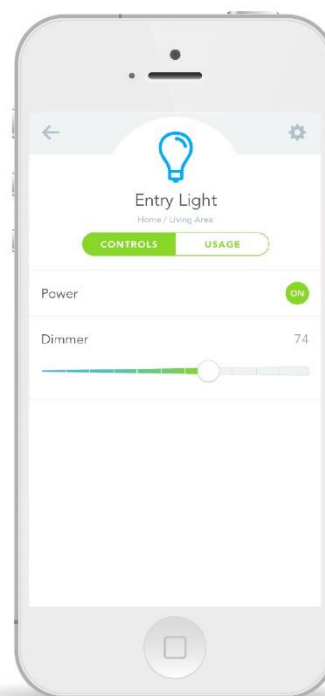
Once the Thing has been on-boarded, the user can control and monitor the Thing.



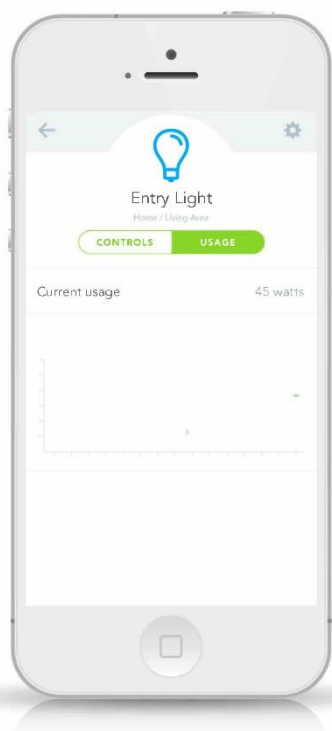
To access the complete set of controls for the Thing, the user selects the icon for the Thing. In this example, the user selects the Entry Light icon.

For ease of use, one of the controls for a Thing can be displayed on this screen. In this example, the Entry Light can be turned on/off by using the “ON” control to the right of the icon.

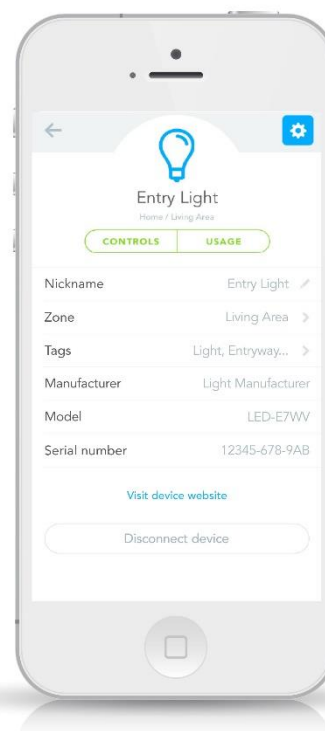
The manufacturer can choose which control is displayed on this screen. For example, in the screen shown above, the current temperature reading could be displayed for the thermostat instead of the on/off switch.




DeB displays the controls for the Entry Light and the user selects the required control.



The manufacturer has also defined a usage page to display some monitoring information for the Thing.



Each Thing also has a device settings page in DeB, which provides information about the Thing.

The device settings page is accessed via the cog icon  in the top right corner of the control page for the Thing. The device settings page shows information about the Thing and allows some of this information to be modified.

**Note:** A Thing may have more than one user interface that can be displayed to the user. For example, a television may offer the user a choice of remote control interfaces, such as a simple interface for basic users, a standard interface for most users and a full interface for technical users. Within DeB, the user selects the interface they prefer to use.

Each user in a household can have DeB running on their smartphone and each user has access to all the Things that have been on-boarded, unless a PIN has been set to restrict access to the IoTG or to a particular Thing. See section [3.4 Updating a Thing](#).

## 2.3 ADRC key concepts

ADRC is based on some key concepts that provide both simplicity for users and flexibility for product developers. These concepts are:

- The user only needs one app to on-board, control and monitor all their Things and this is the device browser app (DeB)
  - > Just as there are multiple web browsers for a user to choose from, it is envisaged that in the future there will be multiple device browser apps from different suppliers and the user will be able to choose their preferred DeB app
- The licensed components are the:
  - > IoT stack embedded in a Thing's microchip
  - > ADRC server software for an Internet of Things Gateway (IoTG)
  - > versions of the device browser app that are customized for a specific vendor
- The open source components are the:
  - > Generic Device browser app (DeB)
  - > Resource Modelling Language (RML) that is used to describe a Thing and its functionality
  - > Resource Control Protocol (RCP) that is the protocol used to transfer RML files and commands between the Thing, the IoTG and DeB
- Things have self-describing profiles, unlike current systems that have fixed device profiles or APIs. The self-describing profile is an RML file that is loaded onto the microchip in a Thing and:
  - > Describes the functions and capabilities of the Thing
  - > Is sent from the Thing to the IoTG and DeB during on-boarding. The ADRC software in the IoTG uses the RML file to control and monitor the Thing, whilst DeB uses the RML file to add the Thing and its functions into the user interface
  - > Means that the user only needs one app (DeB) to interact with any Thing rather than one app for each Thing or brand
  - > Gives product developers the flexibility to introduce new functionality without the constraints imposed by fixed profiles
- ADRC does not need to be an internet-based solution
  - > ADRC-enabled Things can operate purely over a Wi-Fi network and without an Internet or cloud connection. This means that ADRC-enabled Things do not have to contain an IP protocol stack
  - > This ability to operate offline, if desired, suits users concerned about privacy. These users can feel secure by opting out of the internet-connected world, while still being part of a smart home
- ADRC can be an internet-based solution
  - > The IoTG provides the connection to the internet via the household broadband modem. Users can control Things remotely via an internet connection and data can be transferred between Things and external parties, if required

## 3 How ADRC works

The ADRC solution is outlined in section [2.1 What is ADRC?](#). This current section explains ADRC in more detail.

### 3.1 ADRC components and functions

As described in section [2.1 What is ADRC?](#), the software and firmware components that make up the ADRC solution are the:

- **ADRC server software** that runs on an Internet of Things Gateway (IoTG)
- **ADRC IoT stack** that is loaded onto the microchip in a Thing.
- **Resource Modelling Language (RML)** that describes a Thing
- **Resource Control Protocol (RCP)** that is used to transfer RML files and commands
- **Device browser app (DeB)** that is the user interface on a smartphone or tablet

#### 3.1.1 ADRC server software

The ADRC server software is software developed by Xped.

The ADRC server software runs on an Internet of Things Gateway (IoTG), which is a small hardware device that can be provided by multiple suppliers. The IoTG provides low data rate personal area network (PAN) protocols to communicate with ADRC-enabled Things and an Ethernet connection to communicate with a Wi-Fi modem.

**Note:** The examples given in this document are for an IoTG communicating with a Thing using the 802.15.4 standard. However, a manufacturer could build an IoTG to support Things that use other connectivity protocols. In addition, the ADRC server functionality can be built into a Wi-Fi modem so that a separate IoTG is not required.

The IoTG manufacturer loads the ADRC server software onto the IoTG hardware during the manufacturing process.

The ADRC server software provides the following system services:

- Device management to enable the IoTG to manage and coordinate a personal area network (PAN) that can include any number of ADRC-enabled Things and any number of smartphones with DeB installed
- A file system to store the RML files that describe the Things and their capabilities
- A pairing table with the following information for each Thing:
  - > manufacturer and model
  - > MAC address
  - > wireless channel
  - > PAN id
  - > security key for secure communications between the IoTG and a Thing
- Certificate and key generation to enable remote access to Things. See section [3.6.1 Setting up remote access](#)

#### Reference information:

The IoTG software is available for licensing from Xped and can be supplied in a format compatible with the Yocto project.

#### 3.1.2 The ADRC IoT stack

The manufacturer of a Thing licenses the IoT stack firmware from Xped. At time of writing, the IoT stack has been developed for a Cortex M3 MCU and the Telink TLSR8269F512 SOC.



**Note:** To confirm which MCUs currently incorporate the ADRC IoT stack, refer to [xped.com](http://xped.com) or email [info@xped.com](mailto:info@xped.com).

The IoT stack can be loaded into a Thing's microchip during the manufacturing process or it can be supplied preloaded in an Xped designed module. The IoT stack provides all the functionality that a Thing needs to connect into an ADRC solution. This means that the manufacturer of a Thing only needs to create the Resource Modelling Language (RML) file(s) to describe the capabilities of a Thing.

The IoT stack provides:

- NFC and BLE connectivity for a Thing to communicate with a smartphone during the on-boarding process
- 802.15.4 personal area network (PAN) connectivity for a Thing to connect with an IoTG
- A device proxy to manage communications with the manufacturer's application in the Thing
- Security management
  - > Holds the security key the Thing receives from DeB during the on-boarding process
  - > Uses the security key to secure the communications between the Thing and an IoTG
- File storage for the RML that describes the capabilities of the Thing

#### **Reference information:**

For evaluation purposes, the stack is available as an Arduino shield and Raspberry Pi module from Xped.

### **3.1.3 Resource Modelling Language (RML)**

The Resource Modelling Language (RML) is an open source XML-based language developed by Xped.

RML is the ADRC component that removes the need for fixed profiles and enables self-describing profiles. An RML file describes a Thing and provides the information needed by DeB to display the user interface for the Thing and tells the IoTG about the functionality of the Thing so the Thing can be controlled and monitored.

The manufacturer writes the RML needed for the Thing and loads the RML file(s) to the Thing during the manufacturing process.

**Note:** Normally the full RML for a Thing is stored in the Thing itself. However, if space is limited, the RML file in the Thing can simply contain a URL to where the full RML file is located.

A Thing could be a multi-unit device, such as a television with an inbuilt DVD player. These multi-function devices can contain multiple RML files, one RML file for each distinct hardware unit within the Thing. In this example, there would be one RML file for the television unit and one RML file for the DVD unit.

The RML file describes a Thing (or unit of a Thing) in terms of:

- Identity (manufacturer, model, version, nickname, logo)
- User interface and icons
- Functionality
- Settings
- Event handling
- The commands required to interact with the manufacturer's application in the Thing

RML is what makes it possible to have one device browser app (DeB) that can interact with all Things. Conceptually, RML is similar to HTML. HTML describes a webpage so that a web browser can draw it on screen, whereas RML describes a Thing so that DeB and an IoTG can interact with it.

RML standardizes the way all Things are described, just as HTML standardizes the way all websites are described.

During the on-boarding of a Thing (the smartphone tap), the Thing's RML file (or files) is sent to the IoTG and to DeB. See section [3.3 On-boarding a Thing](#).

#### Reference information:

Detailed information on RML is provided in the following documents:

- ADRC Developer Guide
- Resource Modelling Language Reference Manual

### 3.1.4 Resource Control Protocol (RCP)

Resource Control Protocol (RCP) is the protocol developed by Xped to transfer the RML files and commands between a Thing and an IoTG, and between an IoTG and DeB.

RCP has an RCP.host format (XML) for an IoTG to communicate with DeB (or other ADRC clients) and an RCP.wire format (like JSON) for an IoTG to communicate with a Thing. The IoTG translates between RCP.host and RCP.wire formats.

RCP will be familiar to developers who know HTTP. RCP is a RESTful protocol that has been augmented to allow servers to send unsolicited events to clients and has been designed to keep message size small to cater for machine-to-machine (M2M) applications.

#### Reference information:

Detailed information on RCP is provided in the following documents:

- Resource Control Protocol Interface Design Documents

### 3.1.5 The device browser app (DeB)

The device browser app (DeB) is software designed by Xped.

Xped has released an generic free version of DeB to the Apple App Store and to Google Play.

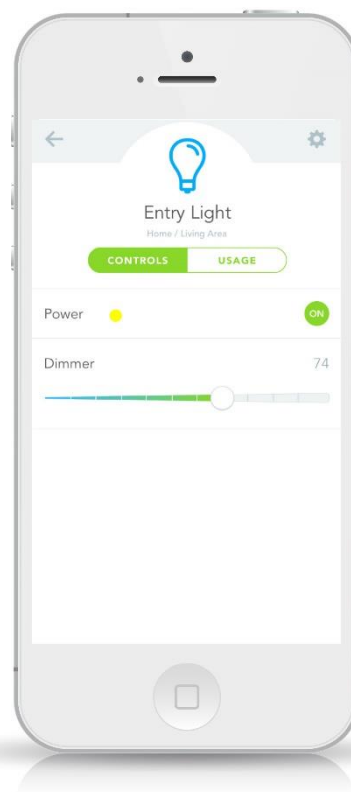
Just as there are multiple web browsers for a user to choose from, it is envisaged that in the future there will be multiple device browser apps from different providers and a user will be able to choose their preferred DeB app.

DeB is a generic device browser that uses the RML files from Things to create the user interface for a Thing. DeB allows a user to have a single app to control all the ADRC-enabled Things in the user's personal area network.

#### DeB:

- Discovers the addressing details of the IoTG and sends this data to a Thing during the on-boarding process so that the Thing can communicate with the IoTG
- Receives the addressing details of a Thing during the on-boarding process and sends this data to the IoTG so that the IoTG can communicate with a Thing
- Provides a virtual gateway capability to enable WiFi or BLE communications between a smartphone and a Thing depending upon the connection capability of the smartphone
- Provides the user interface to control, manage and monitor Things
  - > DeB renders the RML received from a Thing to create the specific icons and control widgets for that Thing
- Provides a unique meta-state display mechanism that indicates the status of a requested control action to the user. This mechanism displays a coloured fault indicator on the DeB screen to inform the user of the status of the action the user has requested, for example, a request to switch a light bulb off. If a Thing does not respond to a user's request, the user sees a yellow indicator. This shows the user that a Thing is not responding, but DeB is still trying to communicate with that Thing. If DeB doesn't receive a response from a Thing after multiple re-tries, the fault indicator turns to red to indicate that a Thing is not responding and the user should physically check the Thing, for example, to check that it is plugged in.

An example of the fault status indicator (the yellow dot) is shown in the following screen.



## 3.2 Implementing ADRC

As described in section [2.2 ADRC user experience](#), setting up an ADRC environment is a very simple process from a user perspective.

This current section describes the background activities that take place automatically between the various ADRC components during set-up.

Once the required ADRC components have been implemented, a Thing can be on-boarded into the user's personal area network (PAN) by a simple tap from a smartphone and the Thing can then be controlled by the DeB app on that smartphone.

### 3.2.1 Internet of Things Gateway (IoTG)

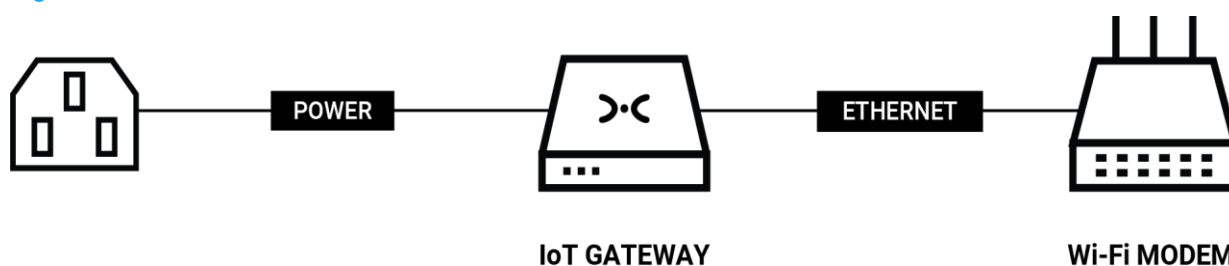
The user sets up their IoTG by simply powering it on and connecting it to the home Wi-Fi modem via an Ethernet cable.

The IoTG then automatically:

1. Generates a personal area network (PAN) id, which represents the IoTG's personal area network.  
The PAN id:
    - > is a random number
    - > uses the IEEE 802.15.4 protocol
    - > is similar to an SSID (Service Set Identifier) for a wireless local area network
  2. Looks for the least busy wireless channel that is available to use to communicate with Things:
    - > ADRC uses 4 wireless channels (channels 26, 25, 20, 15)
- Note:** These channels are in between the channels used for Wi-Fi to avoid data collisions
3. Advertises its address on the Wi-Fi network using Zero Configuration Networking so that a DeB app can discover it without any user intervention required:
    - > the IP address and TCP port of the IoTG are resolved by a Zeroconf service on the smartphone
    - > the RCP protocol name is \_rcp.\_tcp

These steps are shown in

Figure 3.



- When the IoTG is connected to power and to the Wi-Fi modem, it automatically:
  - creates a PAN id
  - selects a wireless channel to communicate with Things
  - advertises the IoTG address over the Wi-Fi network for DeB to find

Figure 3: Internet of Things Gateway (IoTG)

### 3.2.2 Smartphone with DeB

The user downloads DeB from the Apple App Store or Google Play. As soon as DeB is installed on the smartphone, DeB:

1. Looks for an IoTG on the Wi-Fi network that the smartphone is connected to
2. Uses Zero Configuration Networking (Zeroconf) to get the addressing information for the IoTG
  - > IP address
  - > TCP port
  - > Service name
3. Displays the initial DeB screen to the user, see section [2.2 ADRC user experience](#).

---

*IMPORTANT: DeB does not store the addressing information for the IoTG permanently in the smartphone, DeB looks for the address of the IoTG each time DeB starts.*

*Zeroconf is only used between DeB and the IoTG. ADRC pairing is used between a Thing and an IoTG, see section 3.3 On-boarding a Thing.*

---

These steps are shown in [Figure 4](#).

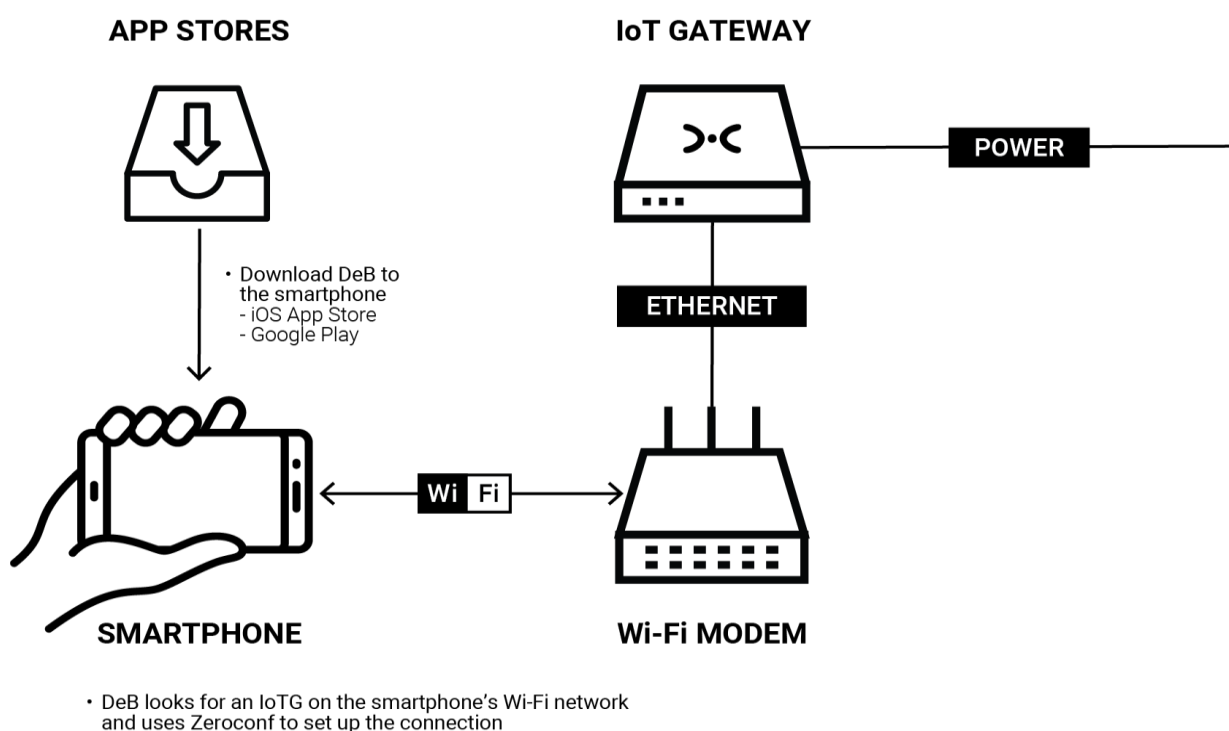


Figure 4: Device browser (DeB)

### 3.2.3 ADRC-enabled Thing

A manufacturer enables a Thing for ADRC by loading the following to a Thing's microchip during the manufacturing process:

1. The ADRC IoT stack, which provides:
  - > NFC/BLE connectivity to the smartphone
  - > 802.15.4 wireless connectivity to the IoTG
  - > the RCP protocol for transferring RML files and commands
  - > the file system for storing RML files and other resources
  - > a device proxy to manage communications with the manufacturer's application in the Thing
  - > security management
2. The Resource Modelling Language (RML) file(s) that the manufacturer has written to describe the capabilities and attributes of the Thing
3. The addressing information (MAC and default wireless channel) for the Thing

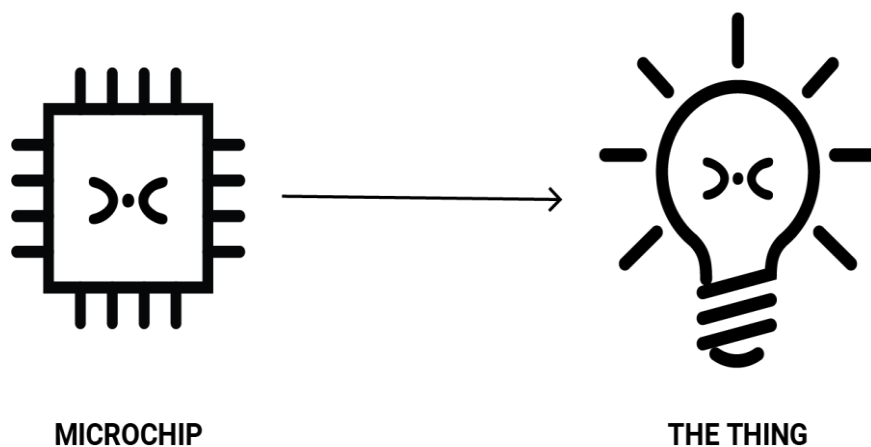
---

*IMPORTANT: The default wireless channel of the Thing will be overwritten by the wireless channel of the IoTG during the on-boarding of the Thing.*

---

A manufacturer includes a touchpoint logo >◁ on an ADRC-enabled Thing to indicate where the user taps their smartphone to on-board a Thing.

These steps are shown in [Figure 5](#).



- Manufacturer includes a microchip in the Thing containing:
  - > the licensed IoT stack
  - > RML file(s) describing the Thing and its functions
  - > the Thing's addressing information
  - > the manufacturer's application for the Thing

**Figure 5: ADRC-enabled Thing**

### 3.3 On-boarding a Thing

Once the components described in section [3.2 Implementing ADRC](#) are in place, a user simply taps their smartphone on a Thing to on-board it.

#### 3.3.1 ADRC on-boarding overview

As shown in

[Figure 1](#), on-boarding is a very simple and quick process from a user perspective:

- the user simply taps their smartphone on a Thing
- DeB presents the user with a screen to enter a nickname for the Thing and accept the pairing
- the user optionally enters a nickname for the Thing (a default nickname is suggested)
- the Thing is now on-boarded and the user can start controlling the Thing via the controls that are now automatically visible in DeB

This section describes the background communications that take place between DeB, the Thing and the IoTG to achieve the on-boarding of the Thing into the personal area network (PAN).

The background communications between DeB, the Thing and the IoTG during the on-boarding process are presented in [Figure 6](#).

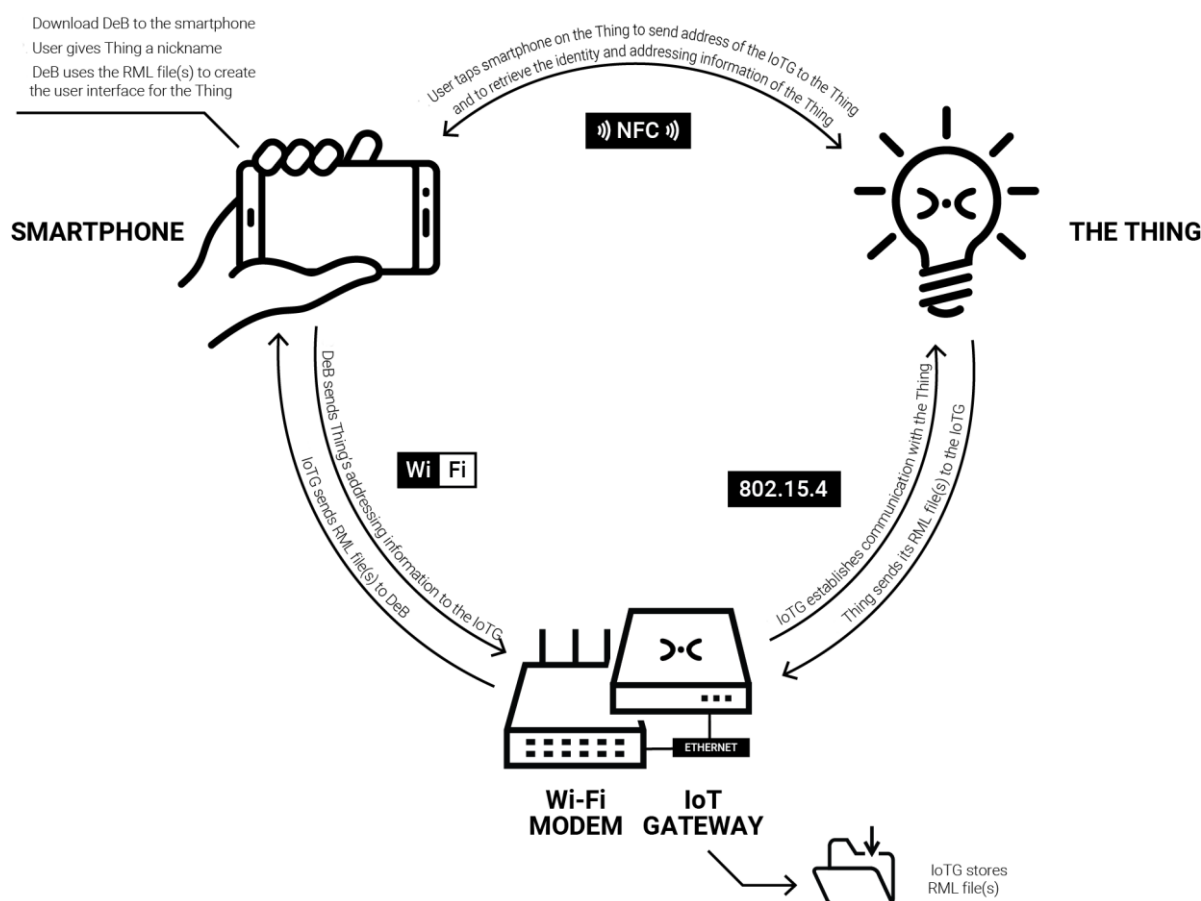
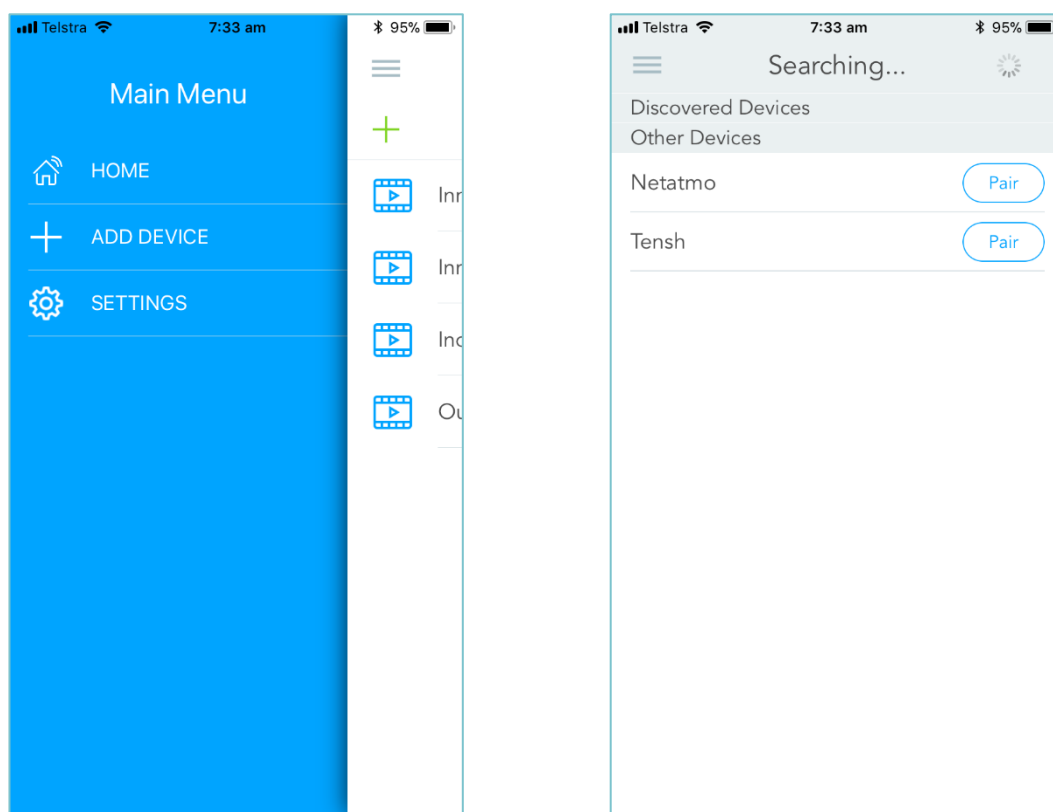


Figure 6: On-boarding overview

### 3.3.2 Foreign device on-boarding overview

Foreign devices are those devices that use another protocol other than ADRC. These devices cannot be on-boarded using the NFC/BLE tap method. Examples of such devices are IP video cameras, Z-wave devices, some BLE devices and some WiFi devices. Instead they are on-boarded by using the *ADD DEVICE* item from the DeB's main menu.



When the user selects the *ADD DEVICE* menu item, the searching screen appears. The system then begins searching for foreign devices that are able to be on-boarded. Depending of the technology of the foreign device, the user may have to manually set the device into on-boarding mode. This is true of Z-wave and ZigBee devices which require the user to press a button or perform some other action. The searching screen displays the foreign devices that have been detected and can be paired with. The user selects the *Pair* button next to the device they want to on-board and it is added.

### 3.4 Updating a Thing

Once a Thing has been on-boarded, the user uses DeB to communicate over Wi-Fi with the IoTG to control and monitor the Thing in the user's environment.

However, if the user wishes to perform the following administrative functions for a Thing, the user taps their smartphone on the Thing to communicate with the Thing via NFC or BLE:

- unpair the Thing
- set or change a PIN for the Thing
- change the nickname for the Thing

DeB then presents the user with screens to perform the required administrative function.

The message flow for updating and synchronising the administrative information between DeB, the Thing and the IoTG is similar to the message flow for on-boarding a Thing, see section 3.3.



---

*IMPORTANT: The Deb app uses NFC or BLE to communicate directly with the Thing to on-board or unpair a Thing, to set or change a PIN, and to change the nickname of the Thing.*

*The Deb app uses Wi-Fi to communicate with the IoTG to control and monitor a Thing in the everyday environment.*

---

### 3.5 Controlling and monitoring a Thing locally

During the on-boarding process, DeB uses the RML file(s) for a Thing to create the required user interface for a Thing, see step 19 in section 3.3 *On-boarding a Thing*.

As shown in section 2.2 *ADRC user experience*, it is now a very simple process for the user to control and monitor the Things in the personal area network (PAN) by selecting the icons and controls displayed in DeB.

This section describes what happens between DeB, the Thing and the IoTG when the user selects an icon and control action in DeB.

These steps are summarised in Figure 7 and then described in more detail in the rest of this section.

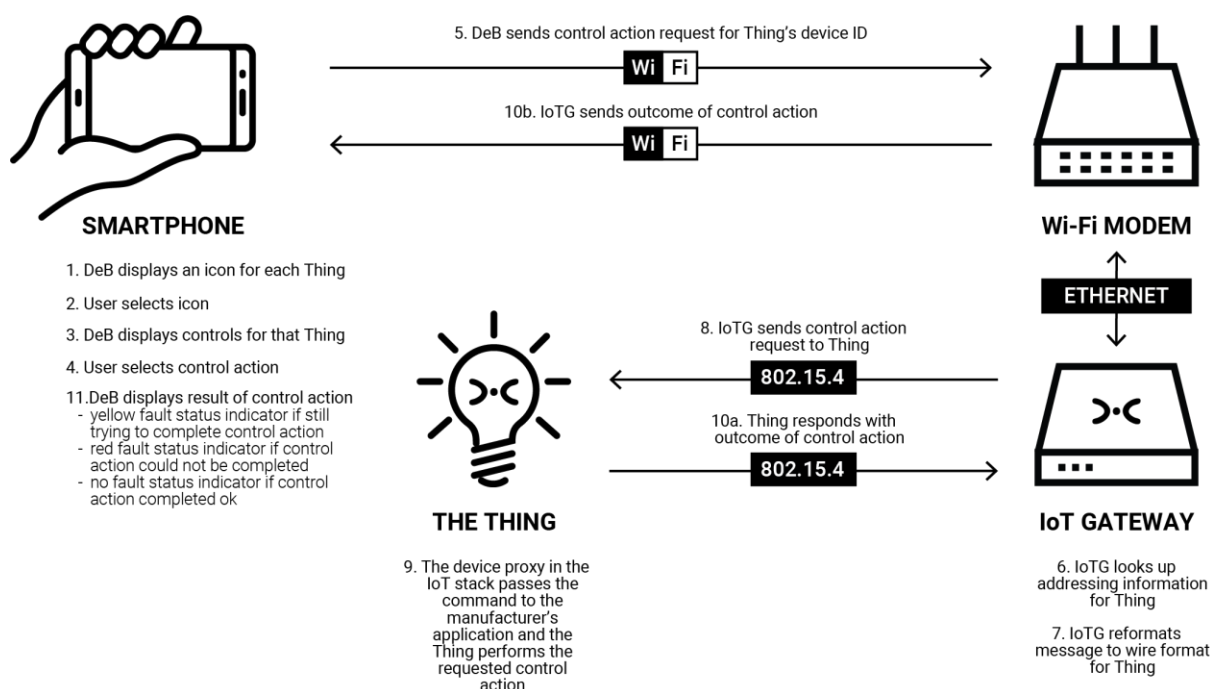


Figure 7: Using a Thing locally

Step 1	The user sees an icon in the DeB app for each on-boarded Thing	
Step 2	The user selects the icon for the Thing they wish to use	For example, a television, a light
Step 3	DeB displays the control widgets for that Thing	For example, on/off switch, channel changer, dimmer switch, and so on
Step 4	The user selects the required control	For example, switch off a television, dim a light

Step 5	DeB sends the control message to the IoTG over Wi-Fi for the IoTG to communicate with the Thing	In this example, a message to switch off a light bulb DeB uses the device id from the Active Device List to identify the Thing to the IoTG
Step 6	The IoTG uses the device id from the DeB message to look up the Active Device List to identify the addressing information for the Thing	
Step 7	The IoTG converts the host format message received from DeB to the wire format message required by the Thing	
Step 8	The IoTG sends the control message to the Thing via the 802.15.4 wireless channel	In this example, a message to the light bulb to switch itself off
Step 9	The manufacturer's application in the Thing receives the correctly formatted command in RCP wire format from the device proxy in the IoT stack, generally via a UART interface. The manufacturer's application in the Thing performs the requested function	In this example, the light bulb switches itself off
Step 10	Acknowledgment (ACK) messages are sent back from the Thing to the IoTG and from the IoTG to DeB indicating the outcome of the control action	
Step 11	If the requested control action cannot be performed, DeB will indicate this to the user by displaying a yellow fault indicator on the screen. The IoTG continues to retry a failed command. After a number of retries, the fault indicator will turn red to indicate that the outcome of the command is unknown. Once the status of the device can be verified, for example when communication is re-established or the device is turned back on, the fault status indicator will be removed.	See section <a href="#">3.1.5 The device browser app (DeB)</a>

## 3.6 Controlling and monitoring a Thing remotely

A user may also want to control and monitor their Things when they are not at home. ADRC enables a user to control and monitor their Things over the Internet.

Setting up remote access is done via a simple tap on the IoTG. Things are controlled remotely via DeB in exactly the same way as Things are controlled locally via DeB.

A Thing can only be on-boarded locally, which means that the smartphone with DeB must be accessing the IoTG using the local Wi-Fi connection and not using an Internet connection.

Once a user has on-boarded Things locally, a user can set up the ability to control and monitor Things remotely. This remote access capability also needs to be set up while the user is in the local environment, as the user sets up this capability by tapping their smartphone on the IoTG.

---

*IMPORTANT: A user on-boards a Thing by tapping the user's smartphone on a Thing.*

*A user sets up remote access to Things by tapping the user's smartphone on the IoTG. The user is setting up remote access to all the Things in that IoTG's personal area network, not just remote access to an individual Thing.*

*Access to an individual Thing, both locally and remotely, can be restricted by setting a PIN for a Thing, see section 3.4 Updating a Thing.*

---

### 3.6.1 Setting up remote access

Remote access to an IoTG is controlled via certificates and mutual authentication, using Extensible Authentication Protocol-Transport Layer Security (EAP-TLS) and Extensible Messaging and Presence Protocol (XMPP).

#### 3.6.1.1 Certificate Authority

Xped is the Certificate Authority (CA) for ADRC. Xped runs the XMPP server that issues, manages and controls certificates for all IoTGs. The Xped certificate is the highest-level certificate in the ADRC certificate hierarchy. The Xped certificate and private/public key pair were generated and installed by Xped during the initial configuration of the XMPP server.

Xped generates a private/public key pair and certificate for each IoTG and these are loaded into the IoTG as part of the manufacturing process of the IoTG. The security management processes follow the certificate and key management processes used for chips and SIMs.

The IoTG certificates are signed by the Xped certificate to create the hierarchy of trust.

#### 3.6.1.2 Activate an IoTG account

Xped creates an account for each IoTG on the XMPP server and each IoTG account is identified by a Universally Unique Identifier (UUID).

Initially, the IoTG account is set to inactive.

To activate the IoTG account on the XMPP server, the user must first set up an account on xped.com. Once the user logs on to xped.com, the user follows instructions on how to activate/deactivate an IoTG.

### 3.6.1.3 Authorise remote access for a user

An IoTG issues the certificates and keys to enable remote access for the smartphones that have access to that IoTG's personal area network. This process is very simple from a user perspective. The user simply taps their smartphone on the IoTG.

This section describes what happens between the smartphone, the IoTG and the XMPP server when the user taps the IoTG.

The certificate, keys and the XMPP account are issued to the smartphone.

These steps are summarised in [Figure 8](#) and then described in more detail in the rest of this section.

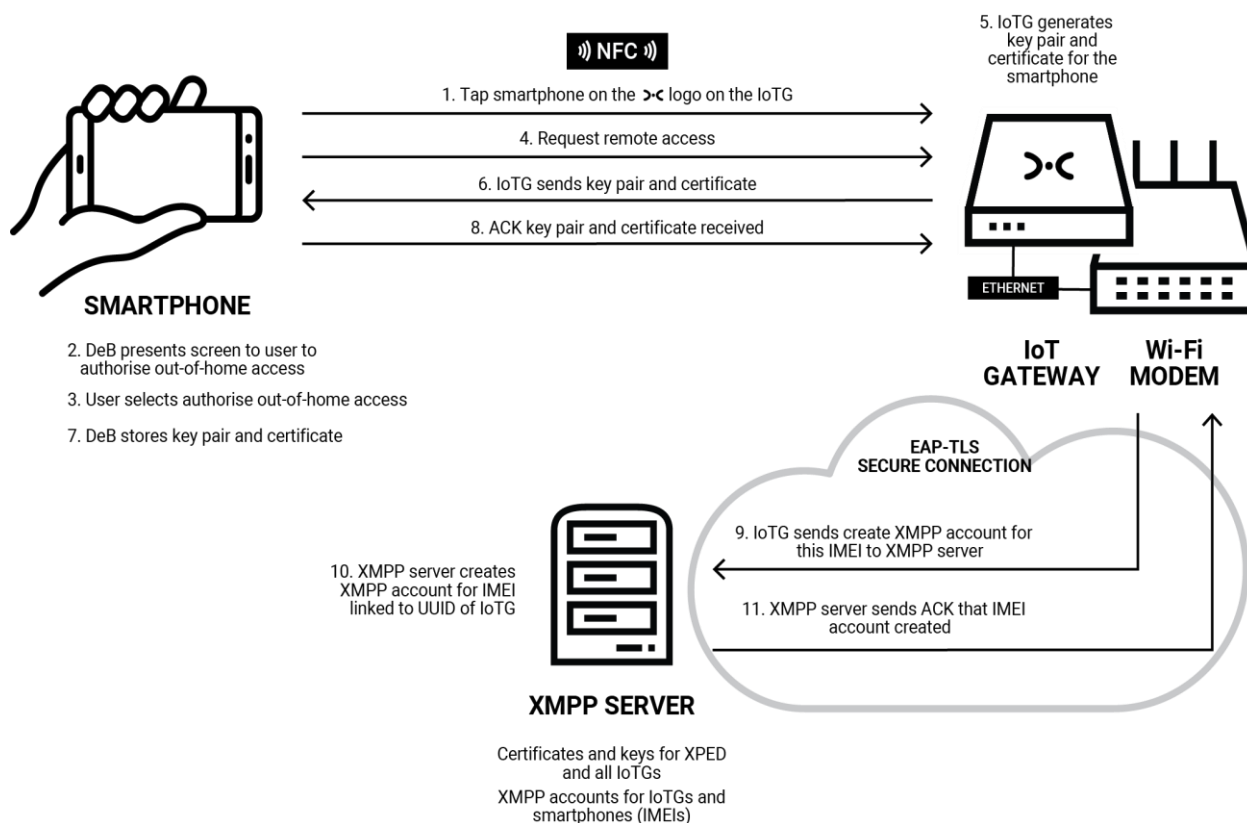


Figure 8: Setting up remote access for a user

Step 1	A user taps their smartphone on the IoTG, which sends an NFC/BLE message to the IoTG with the smartphone's identifier	
Step 2	DeB presents the user with a screen to authorise out-of-home access	<p>If a PIN has not already been set for the IoTG, the user is given the option to set a PIN</p> <p>If an IoTG PIN was set previously, the user must enter this PIN</p>
Step 3	User selects "authorise out-of-home access"	
Step 4	DeB sends a message to the IoTG over the Wi-Fi connection to set up out-of-home access	
Step 5	The IoTG generates a private/public key pair and certificate for the IMEI	
Step 6	The IoTG sends the certificate and key pair to the IMEI via Wi-Fi	The certificate for the IMEI is signed by the IoTG so the XMPP can verify and trust the IMEI's certificate when it is used for authentication during remote access
Step 7	DeB stores the certificate and key pair securely in the smartphone	
Step 8	DeB sends an acknowledgement (ACK) message back to the IoTG	
Step 9	The IoTG uses the home's Internet connection to send a message to the XMPP server to create an account for that IMEI	The IoTG and the XMPP server use each other's certificates to mutually authenticate and to set up a secure EAP/TLS connection
Step 10	The XMPP server creates the XMPP account for that IMEI allowing access to the requesting IoTG's domain	<p>The account is identified by the IMEI and is linked to the UUID of the IoTG that has requested the IMEI account</p> <p>The IoTG is added to the 'buddy' list of the IMEI account</p>
Step 11	The XMPP server sends an acknowledgement (ACK) message back to the IoTG to indicate the IMEI account has been set up successfully	

The smartphone (IMEI) is now enabled to control and monitor Things on that IoTG over the internet.

*IMPORTANT: If the user has access to multiple IoTGs (for example, home, holiday home, office), the user's smartphone must be authorised for remote access for each IoTG. There will only be one XMPP account for that IMEI and this account will indicate which IoTGs (domains) the user is authorised to access remotely.*

### 3.6.2 Accessing a Thing remotely

Once the smartphone has been authorised for remote access to an IoTG, as described in section 3.6.1 *Setting up remote access*, DeB and that IoTG become XMPP clients (buddies) and can 'chat' with each other over XMPP. This allows the user to remotely control Things on that IoTG.

DeB displays to a user whether the user is on a local Wi-Fi connection to the IoTG or a remote connection over the internet. DeB also shows which domain (group of Things) the user is connected to, if the user does have access to multiple domains. For example, a domain could be home, holiday home, office and so on.

The default DeB settings are local/all, which is local access and all groups.

#### 3.6.2.1 Remote access summary

From a user perspective, accessing a Thing remotely is just as simple as accessing a Thing locally. The user selects remote access in DeB and then uses the same icons and controls that are used to interact with a Thing locally.

When the user selects remote access in DeB, DeB communicates with the XMPP server, which then communicates with the IoTG.

The background communications between DeB, the XMPP server, the IoTG and the Thing during remote access are presented in summary form in Figure 9 and then described in detail in section 3.6.2.2 *Remote access detailed steps*.

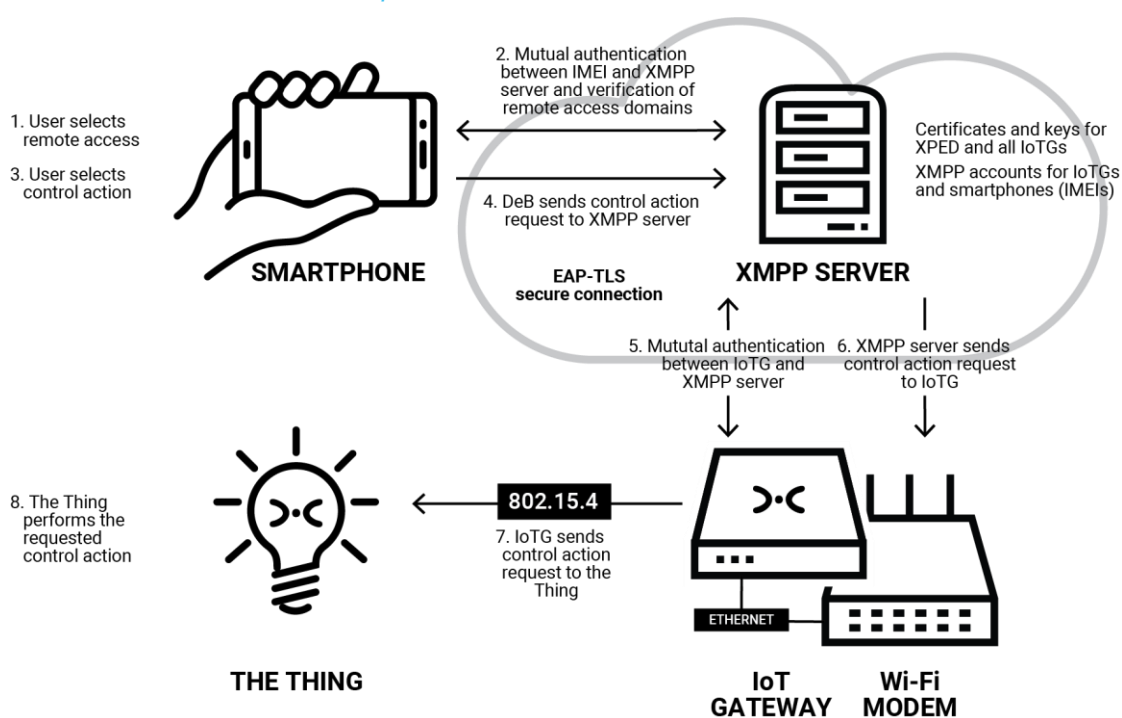


Figure 9: Remote access summary

### 3.6.2.2 Remote access detailed steps

The detail of the communications between DeB, the XMPP server, the IoTG and the Thing during remote access is shown in

[Figure 10](#) and then described in step-by-step detail in the rest of this section.

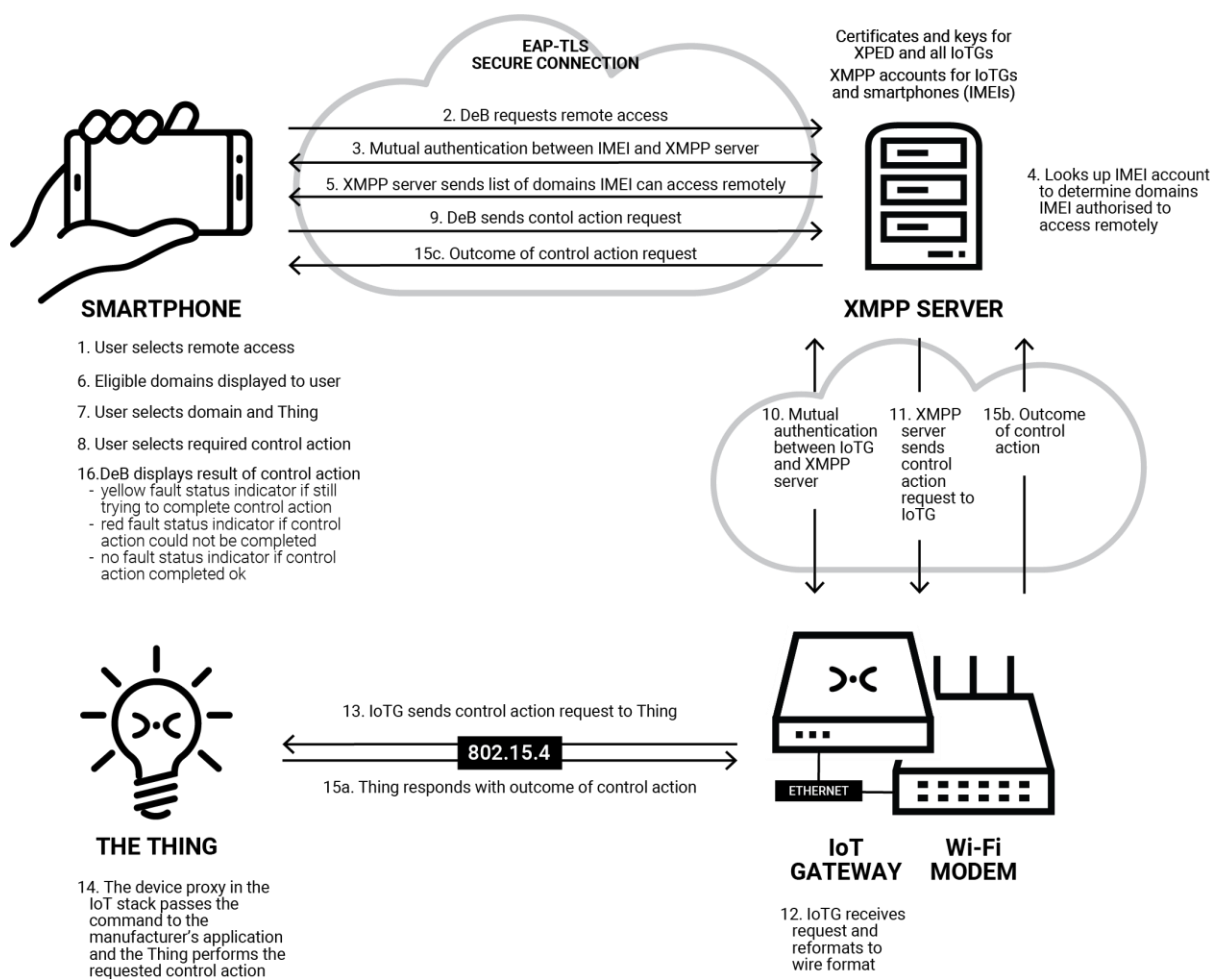


Figure 10: Remote access detailed steps



Step 1	The user selects remote access in DeB	
Step 2	DeB sends a message over the internet to the URL of the XMPP server to request remote access	
Step 3	The XMPP server and the smartphone (IMEI) mutually authenticate using their certificates and set up a secure EAP-TLS channel to communicate over the internet	
Step 4	The XMPP server accesses the XMPP account for that IMEI to determine which domain(s) that IMEI has access to	
Step 5	The XMPP server sends the list of domains that the IMEI has remote access to back to DeB	
Step 6	DeB displays the list of authorised domains to the user	
Step 7	The user selects the domain and the Thing within the domain	
Step 8	The user selects the required action, for example, to switch off a light bulb	
Step 9	DeB sends the control request message to the XMPP server	
Step 10	The XMPP server:	<p>Determines which IoTG that the message should be sent to</p> <p>Mutually authenticates with that IoTG and sets up a secure EAP-TLS connection</p>
Step 11	The XMPP sends the control request message to the IoTG	
Step 12	The IoTG:	<p>Receives the message from the XMPP server</p> <p>Re-formats the message into wire format for the Thing</p>

Step 13	Sends the message to the Thing via the 802.15.4 wireless channel	
Step 14	The manufacturer's application in the Thing receives the correctly formatted command in RCP wire format from the device proxy in the IoT stack, generally via a UART interface. The manufacturer's application in the Thing performs the requested function	In this example, the light bulb switches itself off
Step 15	Acknowledgement (ACK) messages are sent back from the Thing to the IoTG, from the IoTG to the XMPP server, and from the XMPP server to DeB	—
Step 16	If the requested control action cannot be performed, DeB will indicate this to the user by a fault status indicator on the screen	See section <a href="#">3.1.5 The device browser app (DeB)</a>

## 3.7 Security summary

The use of ADRC IoT technology does not introduce any new risks above any risks that may already be present in other IP-enabled or chip products.

ADRC has built security into the local connections and uses XMPP certificates and standards-based secure transport methods for remote access.

### 3.7.1 On-boarding

The on-boarding of Things is done over the local secured Wi-Fi network in the home and not via the internet.

### 3.7.2 Internet of Things Gateway (IoTG)

Similar to the WPA for a Wi-Fi modem, the IoTG has a PIN to secure access to the IoTG and prevent any non-authorised user accessing the personal area network (PAN) of the IoTG.

### 3.7.3 Thing

A PIN can be set for a Thing to prevent unauthorised access to a particular Thing.

The microchip in a Thing has standard hardware security features, which means the microchip is read- blocked and the licensed IoT stack cannot be read. The microchip will erase itself if it is tampered with. The microchip can only be re-written.

The Reference Design for the Thing defines a secure element chip, which is not currently used, but could be used for authentication, encryption and key exchange.

### 3.7.4 PAN key

During the on-boarding of each Thing, DeB generates a 128-bit AES security key for each Thing. This key is sent to the Thing and the IoTG and this key encrypts communication over the 802.15.4 wireless connection. DeB does not store a copy of the security keys as DeB does not use these keys.

### 3.7.5 Device browser (DeB)

DeB communicates with the IoTG locally over the secured home Wi-Fi network and remotely via an authenticated and secure connection with the XMPP server.

### 3.7.6 Remote access

A user must be in the local environment to set up remote access. The user's smartphone uses NFC or BLE (a tap) to communicate with the IoTG to initiate the generation of a private/public key pair and certificate for the IMEI. Access to the IoTG can be protected by a PIN.

The ADRC solution uses XMPP certificate-based mutual authentication to authorise a smartphone to access an IoTG remotely and to set up a secure connection.

## 4 Additional services

### 4.1 Ontology

All Things produce data, such as measurements, status information, warnings and information on any malfunctions that may have occurred. Data is only valuable if it can be interpreted and understood.

For the past few years, the World Wide Web Consortium has been working on Semantic Web technologies, such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL). An ontology is a formal definition of all the Things that exist in an application domain and how they relate to one another

The Resource Modelling Language (RML) standardises the way Things are described, which enables a smart home/workplace and other ontologies to be created. RML supports linkages into OWL ontologies so that DeB and other clients can automatically understand and interact with Things without any human intervention required. RML enables Things to have self-describing profiles. This feature will enable device crawlers to find Things and their RML. It will also allow big data engines and Artificial Intelligence agents to understand and interact with any Thing without any customisation and without any human intervention.

### 4.2 Cloud

A benefit of the ADRC IoTG is that the data can be collected locally. This allows users to decide how to protect, use or export their IoT data.

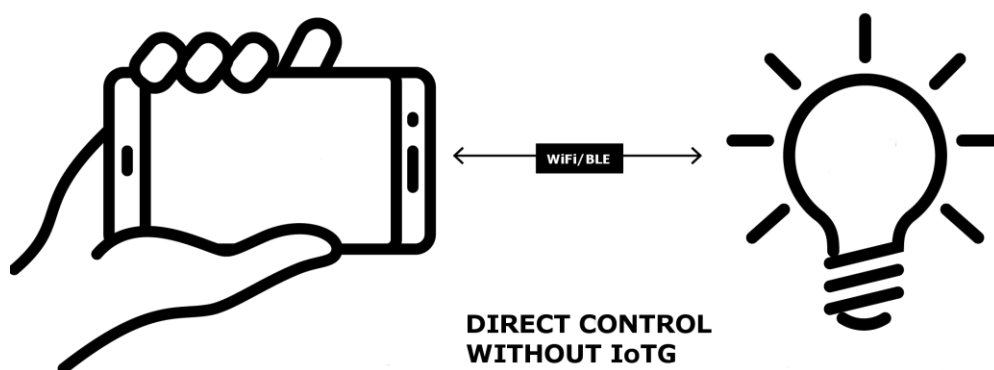
However, the IoTG can also connect to the Internet and when it is connected, it will send user details and device usage data to the ADRC cloud. The ADRC cloud provides a basic analytics dashboard that allows the data to be visualized by region, device type and command type.

### 4.3 Foreign protocol integration

#### 4.3.1 Virtual gateway

Using the *virtual gateway* concept, DeB can connect directly with devices that use WiFi or BLE without the need for an IoTG. This is a convenient feature for very low cost installations or situations where the user would like to try the system before investing in a gateway.

DeB supports a number of virtual gateway devices including an IR Blaster for controlling audio visual equipment and air conditioners and several IP video cameras. The list of supported devices will increase over time.

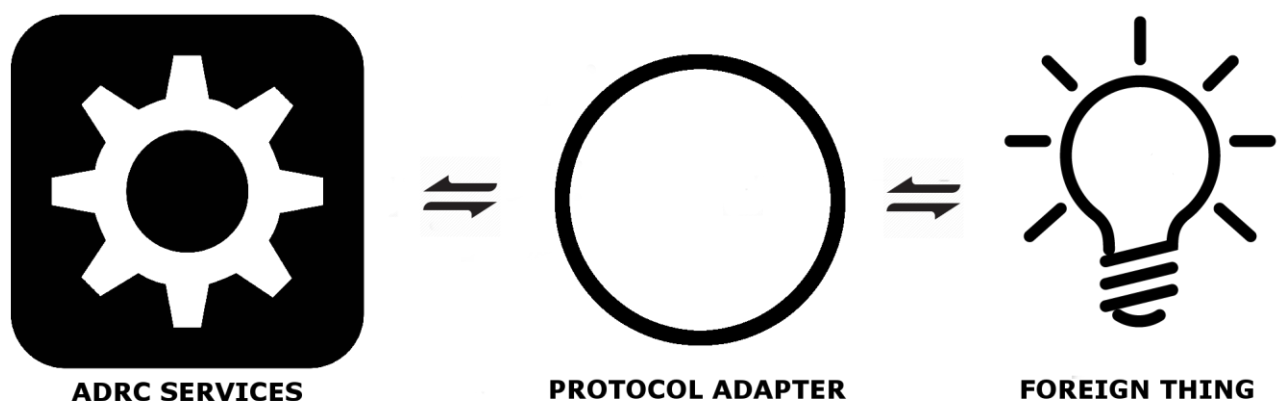


### 4.3.2 Protocol adapters

A protocol adapter is software that is installed on the IoTG that allows the ADRC services to communicate with devices that use a foreign protocol. A protocol adapter is required to be developed for each foreign protocol and makes those devices appear to behave like an ADRC device.

Protocol adapters can be developed by Xped or by a manufacturer and then installed on an IoTG via its remote update mechanism. This means that adapters are a flexible way of expanding the breadth of devices available to be controlled by DeB.

For each device class that is managed by a protocol adapter, an RML file is developed that describes this device to the ADRC system and to DeB. In this way the handling of foreign devices by DeB is exactly that same as for native ADRC devices.



# Contact details

## **Xped Registered & Corporate Office**

Level 6,  
412 Collins Street  
Melbourne VIC 3000  
AUSTRALIA

PO Box 16059  
Collins St West  
Melbourne VIC 8007  
AUSTRALIA

## **Head Office**

Suite 11, 2 Portrush Road  
Payneham SA 5070 AUSTRALIA

Email: [info@xped.com](mailto:info@xped.com)

