Arrow Connect Boot Camp

Tam Nguyen

Brandon Hall

Jim Lindblom

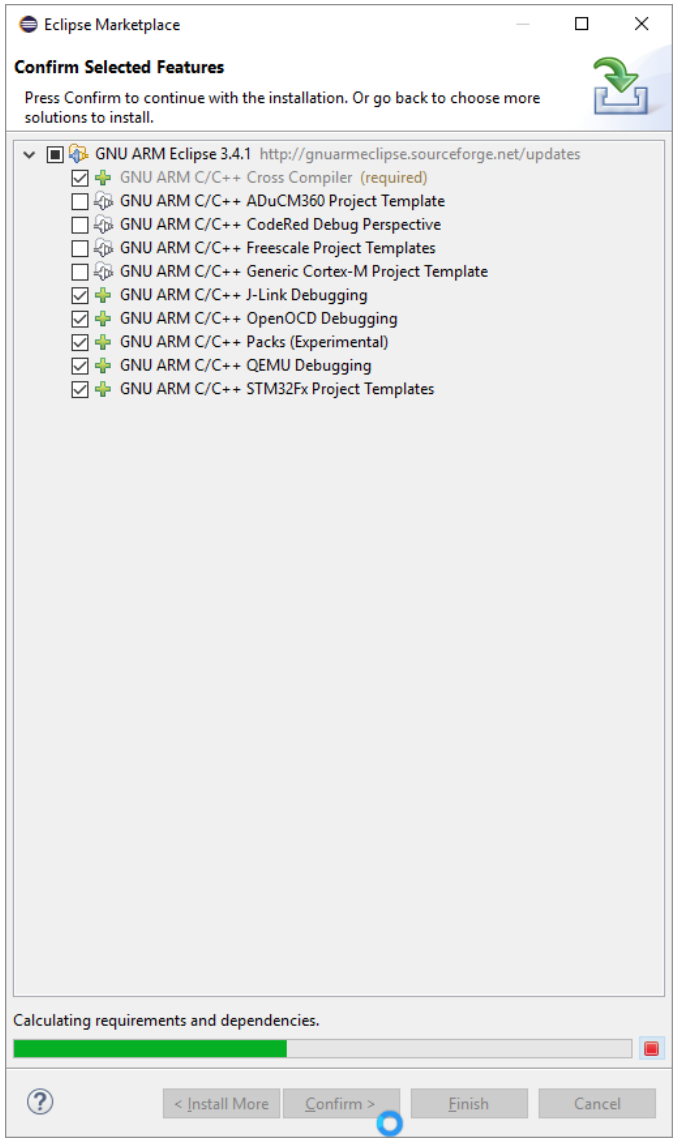
Denver - May 2017

Lab Prerequisites

1. Windows Laptop
2. Install Node-RED - https://developer.arrowconnect.io/docs/using-node-red-with-arrow-connect (password is ArrowConnect)
3. Install 7-Zip - http://www.7-zip.org/download.html
4. Install Oracle Java 8 - https://java.com/en/download/
5. Prepare training directory - `mkdir c:\acn`
6. Install Tera Term under `c:\acn\teraterm-4.94` - https://ttssh2.osdn.jp/index.html.en
7. Install Android platform tools under `c:\acn\platform-tools`
   https://developer.android.com/studio/releases/platform-tools.html
8. Download boot image to `c:\acn\platform-tools`
   https://content.arrowconnect.io/public/training/05-2017/boot-linaro-jessie-qcom-snapdragon-arm64-20161006-144.img
9. Download FS image to `c:\acn\platform-tools`
   https://content.arrowconnect.io/public/training/05-2017/linaro-jessie-developer-qcom-snapdragon-arm64-20161006-144.img
10. Download JCE 8 and extract to `c:\acn\jce`
    http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html
11. Download and install ST-Link V1/V2 Driver - https://content.arrowconnect.io/public/training/05-2017/st-link-driver.zip

Lab Prerequisites (Optional)

1. Download/Extract **Eclipse for C/C++ Developers**
    1. http://www.eclipse.org/downloads/eclipse-packages/
    2. 3 entries down
    3. Must match your system (32-bit or 64-bit)
    4. Extract to C:\acn
2. Download/Install GNU ARM Embedded Toolchain
    1. https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads
    2. Accept and use default install location
    3. **Do not install to C:/Program Files (x86)** – Can't have parenthesis in name
    4. **Do not "Add path to environment variable" at end**
3. Download/Install GNU ARM Eclipse Windows Build Tools
    1. https://github.com/gnuarmeclipse/windows-build-tools/releases
    2. 32-bit: gnuarmeclipse-build-tools-win32-2.8-201611221915-setup.exe
    3. 64-bit: gnuarmeclipse-build-tools-win64-2.8-201611221915-setup.exe
4. Open Eclipse.exe (C:\acn\eclipse.ex)
    1. Choose any folder as your workspace
5. Install GNU ARM Eclipse
    1. Go to **Help > Eclipse Marketplace**
    2. Search for **GNU ARM Eclipse**
    3. Install **GNU ARM Eclipse 3.4.1**
        1. See image to right for feature selections
    4. Restart Eclipse



Introduction

- How long have you been with Arrow?
- What's your current role and responsibility?
- How long have you been working with IoT technology?
- What are you hoping to learn from the training? Any specific area that you're interested or most relevant to your job?
  - Cloud
  - Gateway
  - Embedded
  - API & SDK
  - Developing demos
  - Custom app development

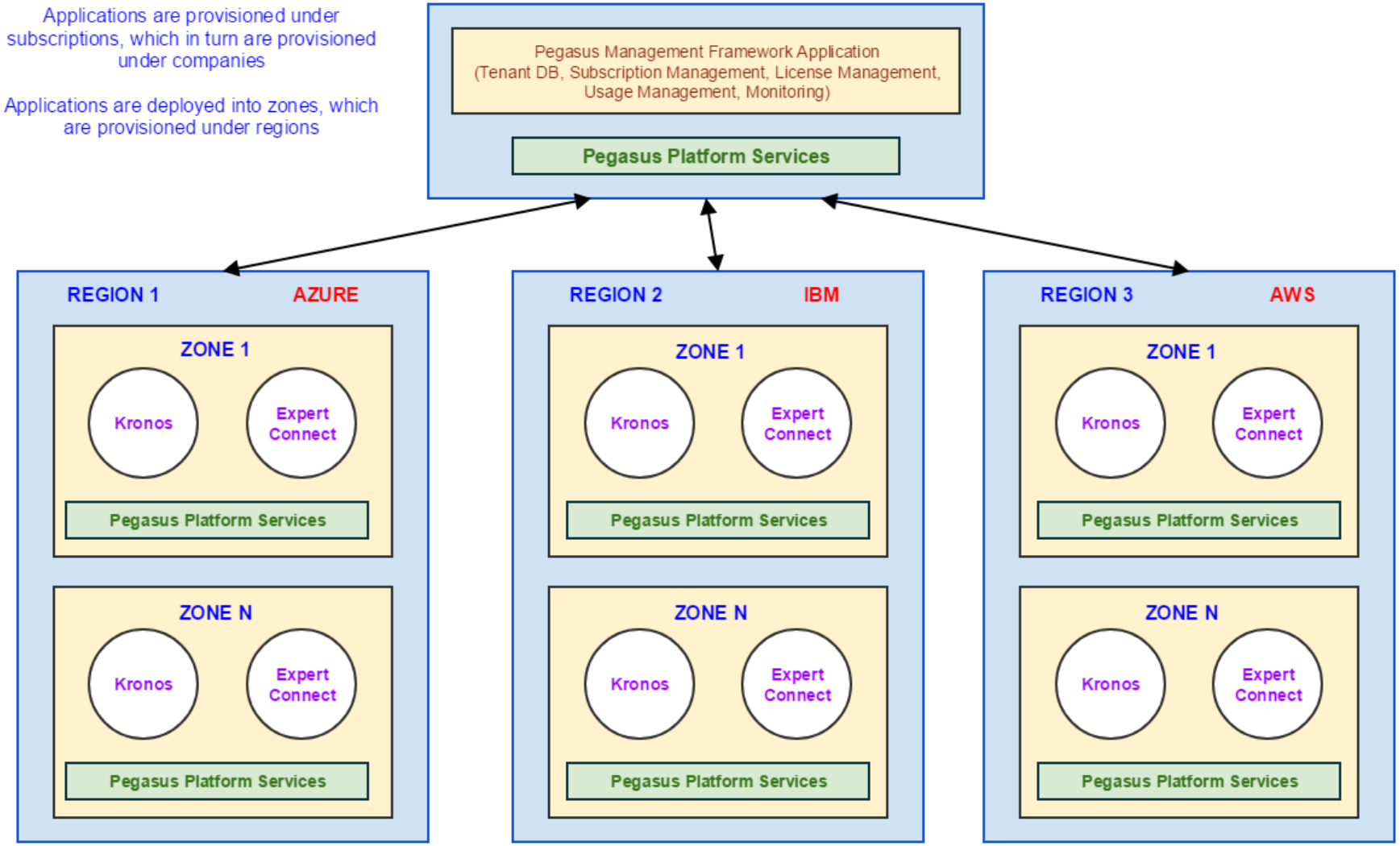
Arrow Cloud Services (PaaS) – Pegasus Platform

- Infrastructure-as-a-Service (IaaS) / Platform-as-a-Service (PaaS) / Software-as-a-Service (SaaS)
- Arrow's private cloud is divided into many network regions and zones and hosted by global public cloud providers such as MS Azure, IBM Bluemix, AWS, etc.
- Software built on this platform inherits the following features
  - Multi-tenancy (SaaS) – unlimited nested level
  - Multiple security schemes (SSO, AD, custom, RBAC, etc.)
  - Encryption at rest and in transit
  - Data encapsulation
  - Distributed Services
  - Highly Available
  - Resilient

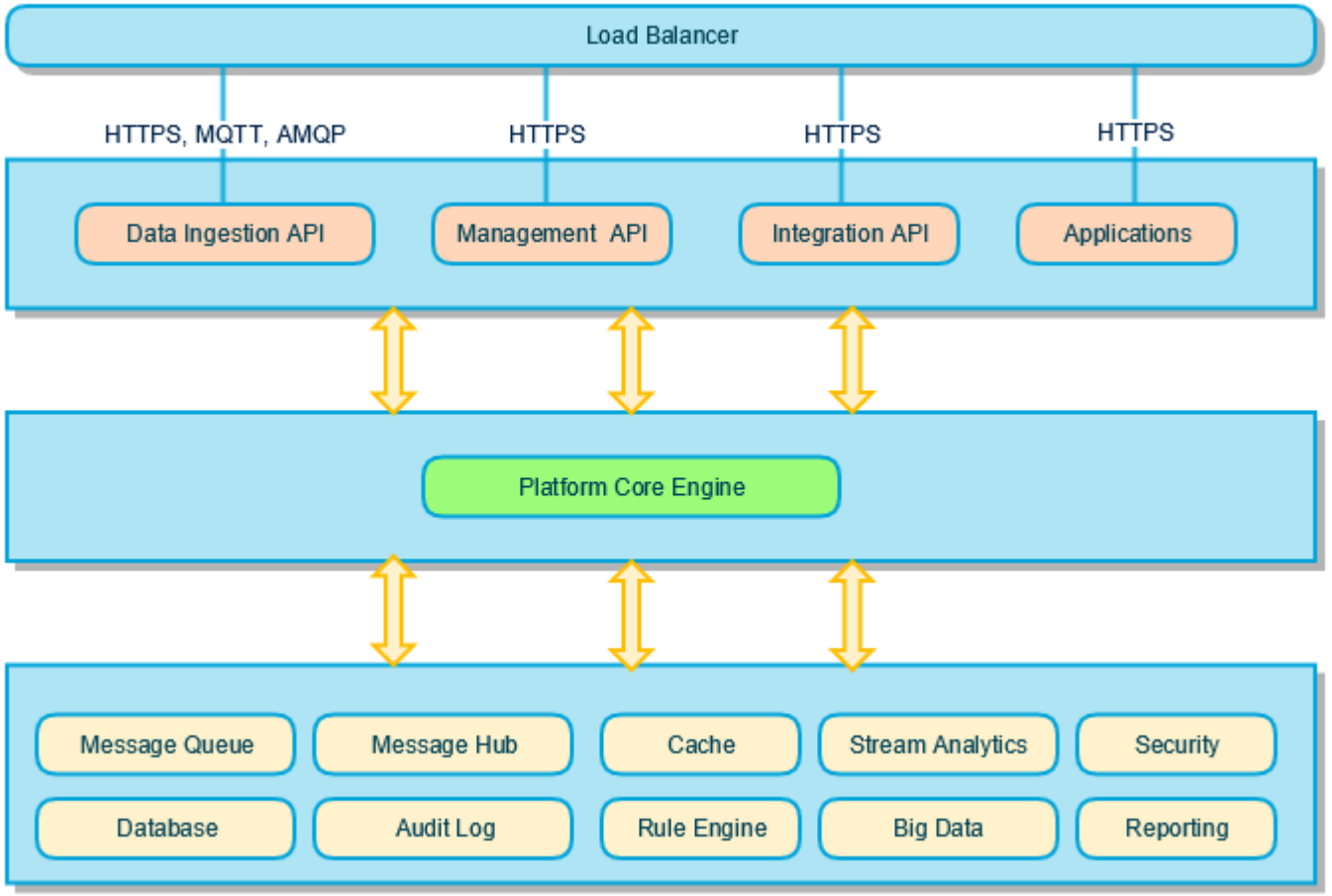
Pegasus Platform Deployment Diagram

Applications are provisioned under subscriptions, which in turn are provisioned under companies

Applications are deployed into zones, which are provisioned under regions



Pegasus Platform Component Diagram



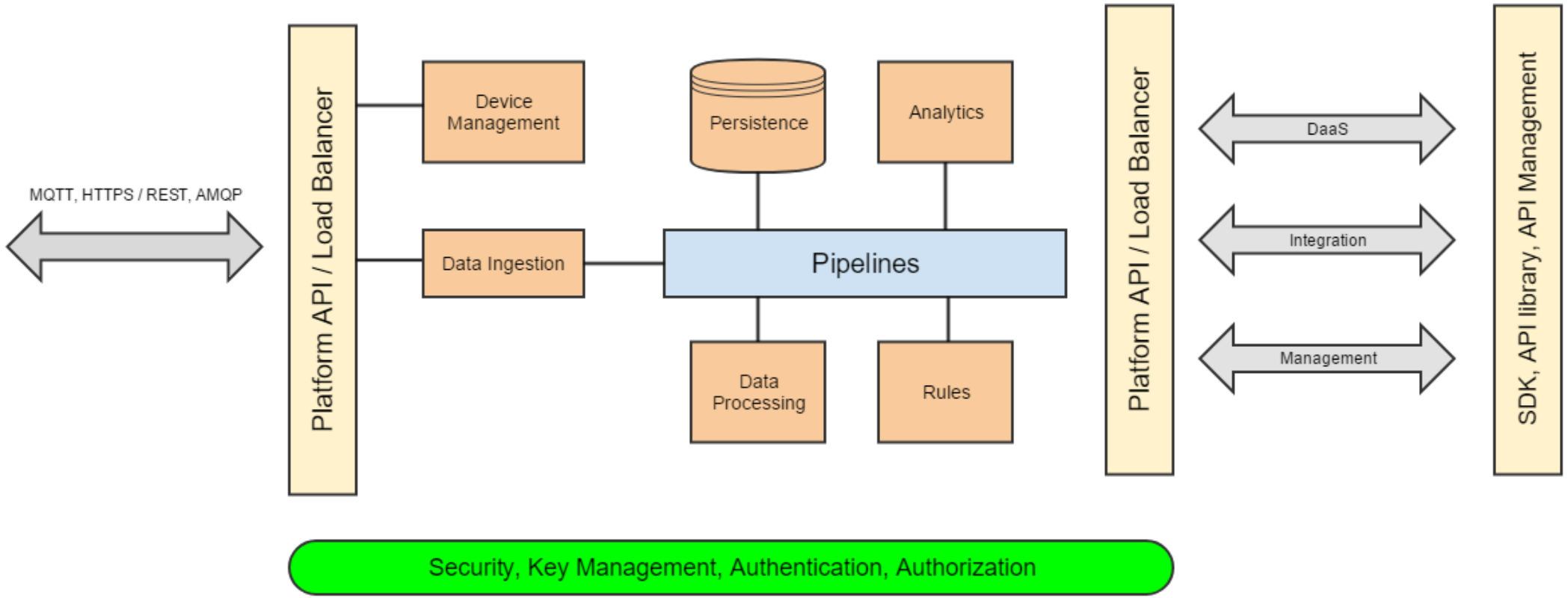
Arrow Connect Cloud (1/2) – Kronos Cloud

- Built on top of Pegasus Platform Services
- Complete IoT cloud solution
  - Device Management
  - Software/Firmware Management (July 2017)
  - Data Ingestion
  - Data Analytics
  - Stream Processing / Rule Engine
  - API Integration
  - SDKs in many programming languages
  - Management Portal
  - Dynamic Dashboards (Q3 2017)
  - Reporting (Q4 2017)

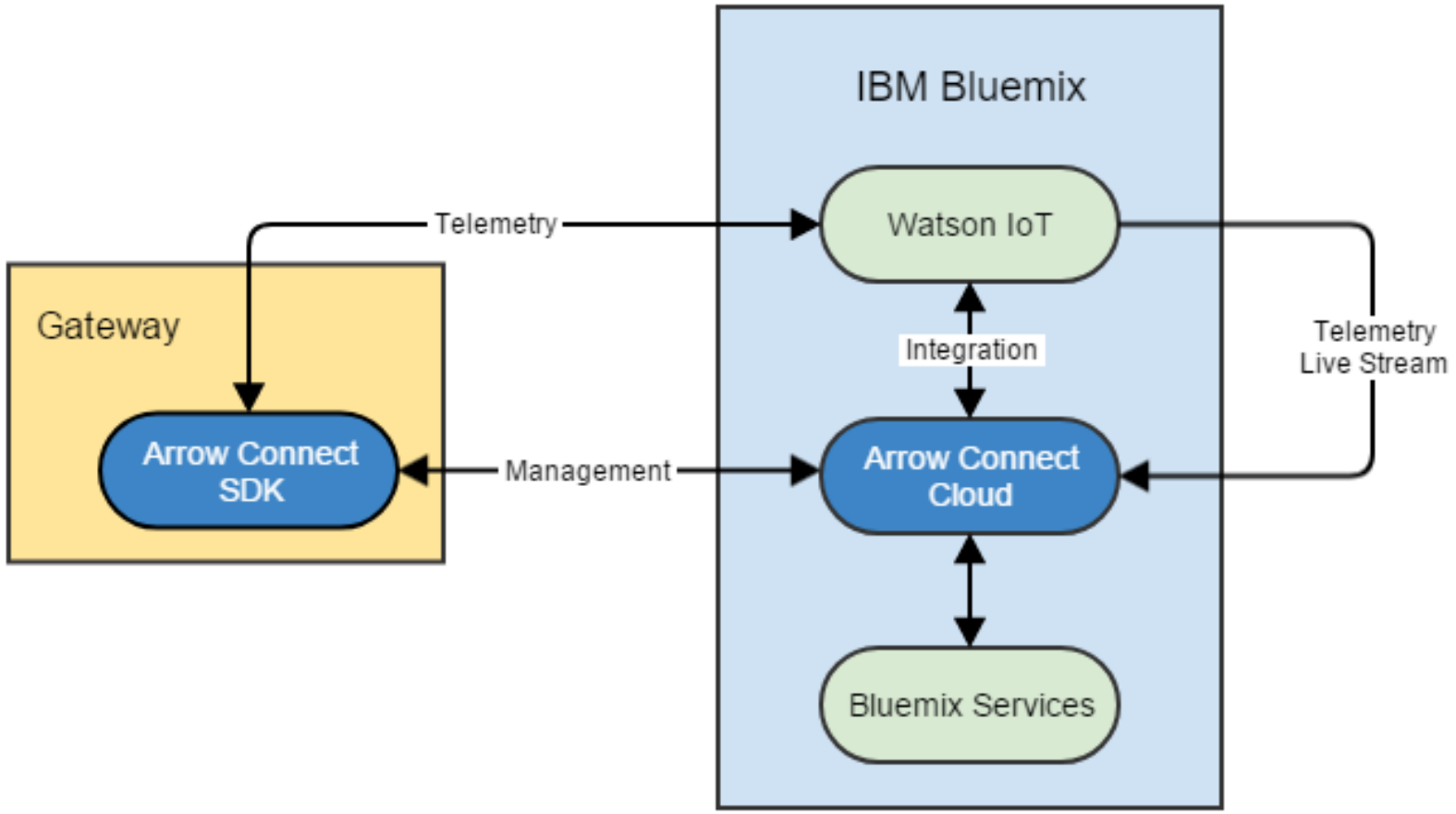
Arrow Connect Cloud (2/2)

- Flexible deployment options
  - Cloud-based Arrow solution (SaaS)
  - On-premise Arrow solution
  - Arrow Connect + IBM Watson IoT
  - Arrow Connect + MS Azure IoT
  - Arrow Connect + AWS IoT
- Product Extensions
  - Add-on third-party products and services
- Out-of-the-box Integration solutions
- Custom Application Development
  - Customer's Intellectual Property (IP)
  - Various hosting options

Arrow Connect Cloud – Architecture Diagram



Arrow Connect – IBM Watson IoT Integration



Pegasus Platform – API and SDK

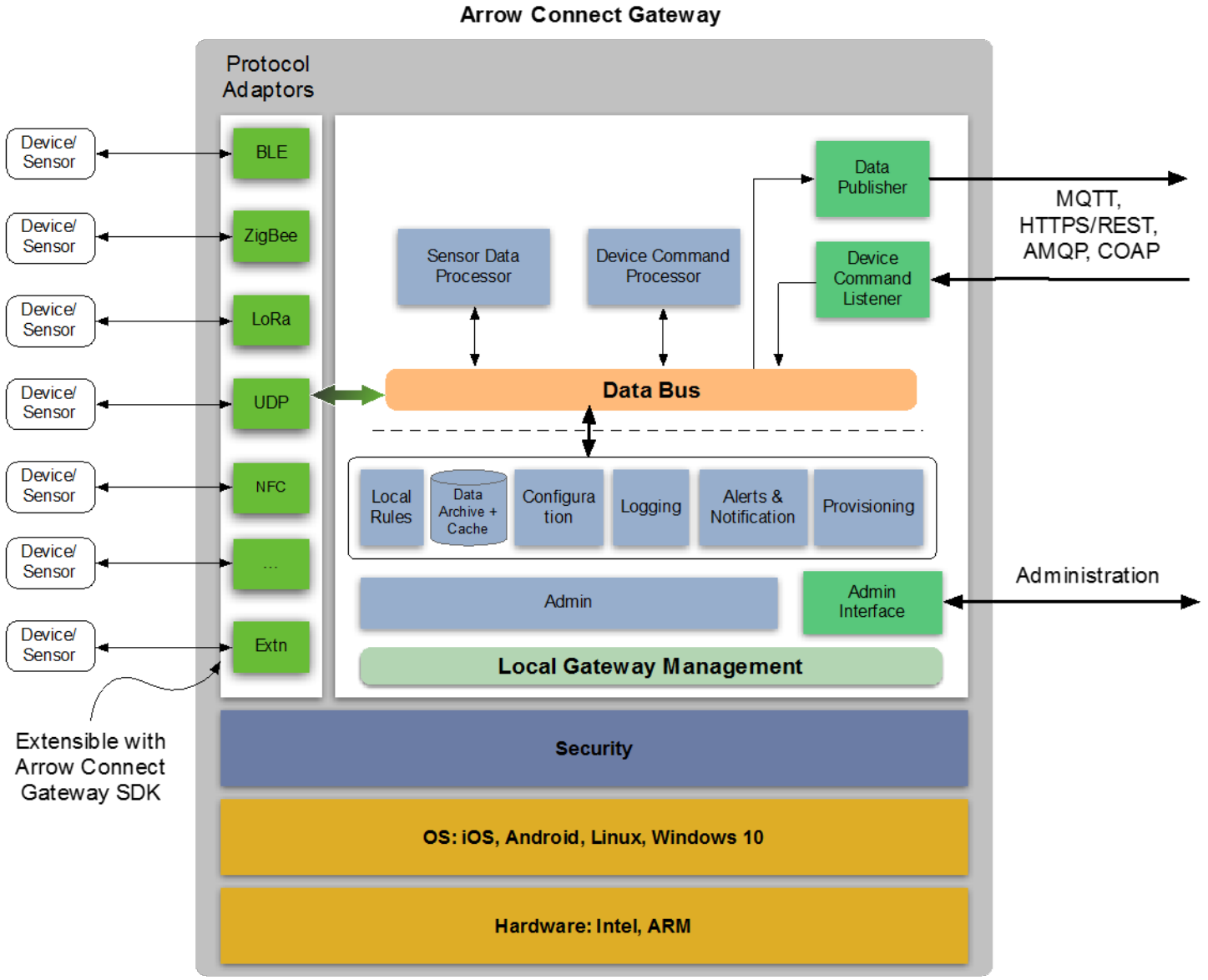
- A rich API set is available via REST API, but only for internal use at this time
- A subset of this API will be available publicly for external integration by Q4 2017
- A small SDK in Java is already available on GitHub because it is a dependency for Arrow Connect SDK - https://github.com/arrow-acs/acs-sdk-java
- Full SDKs in Java and other programming languages will be provided after the REST API is publicly available

Arrow Connect Gateway

- Full-featured gateway software – Production Ready
  - Linux
  - Windows
  - iOS
  - Android
- Modular design for Plug-and-Play (device adapters, databuses, cloud connectors, databases)
- Many out-of-the-box protocol support
- Extensible by Arrow and third-party software vendors
- Local management via web portal - Optional
- Edge analytics
- Pre-certified for many HW and SW design
- Multiple configurations from small footprint to full scale gateway solutions
- Easy integration with third-party gateway architecture
- SDK available for full custom gateway development
- Support of Windows 10 IoT Core is TBD

Arrow Connect Embedded

- Complete C-SDK for REST API methods
  - ANSI C standard
  - Portable to different compilers
- Reference Firmware
  - STM32 Nucleo, IDW01M1, IKS01A1, ARM mbed
  - ARIS board, Renesas S7, ThreadX RTOS
  - SX-ULPGN (QCA-4010), Silex Xtensa, ThreadX RTOS
  - ST SensorTile, STM32Cube
  - SenseAbility, Cypress PSoC
  - STM32L475 IoT Node Kit
  - OnSemi BB-GEVK, ARM mbed


Arrow Connect Gateway
Protocol Adaptors
Device/ Sensor
Device/ Sensor
Device/ Sensor
Device/ Sensor
Device/ Sensor
Device/ Sensor
Device/ Sensor
BLE
ZigBee
LoRa
UDP
NFC
...
Extn
Extensible with Arrow Connect Gateway SDK
Sensor Data Processor
Device Command Processor
Data Publisher
Device Command Listener
MQTT, HTTPS/REST, AMQP, COAP
Data Bus
Local Rules
Data Archive + Cache
Configura tion
Logging
Alerts & Notification
Provisioning
Admin
Admin Interface
Administration
Local Gateway Management
Security
OS: iOS, Android, Linux, Windows 10
Hardware: Intel, ARM

Arrow Connect – REST API (1/2)

- Arrow Connect REST API provides full integration into the platform. Our gateway software (Selene and Mobile Apps) use the same API and SDKs that we publish on GitHub
- Swagger UI is available to help with development and testing https://api.arrowconnect.io/swagger-ui.html
- When an application instance is configured as in development mode, REST API only requires API key. In production mode, it requires API Signing Request. That means Swagger cannot be used to test application instance running in production mode.
- Access Key determines which API methods you can use and what data you can access
- API categories
  - Gateway API – gateway to cloud integration
  - Admin API – cloud to cloud integration
- API URL depends on which region and zone the application is deployed.
  - For region A, zone 01, the URL is https://api-a01.arrowconnect.io
  - Default zone (for demo) is a01 and can be referenced with the suffix – https://api.arrowconnect.io

Arrow Connect – REST API (2/2)

- API method groups
  - account-api — User related methods
  - aws-integration-api — AWS integration methods
  - device-action-api — Device Actions methods
  - device-api — Device methods
  - device-state-api — Device State methods
  - device-type-api — Device Type methods
  - gateway-api — Gateway methods
  - ibm-integration-api — IBM integration methods
  - kronos-event-api — Platform event methods
  - node-api — Node (group) methods
  - node-type-api — Node Type methods
  - telemetry-api — Telemetry methods

Arrow Connect – Telemetry API

- Recommended protocol is MQTT for sending telemetry to cloud. This also applies to other IoT providers – IBM, Microsoft, and AWS
- Other protocols are supported – REST API and AMQP
- For streaming telemetry out of the platform, the following methods are supported
  - WebSocket
  - AMQP
  - MQTT (Q4 2017)
  - HTTP POST (Q4 2017)
- Standard telemetry query via REST API is also supported

Arrow Connect - SDK

- SDKs are provided and maintained by Arrow for faster and better integration development. SDK communicates with the platform via REST API https://github.com/arrow-acs
- Support many programming languages
  - Java
  - C/C++
  - iOS
  - Android
  - C# (TBD)
  - Node.js (TBD)
  - Python (TBD)
  - PHP (TBD)
  - Perl (TBD)
  - Ruby (TBD)
  - Go (TBD)

Application Lifecycle Management

- ❑ Agile development shop
- ❑ 3-week Scrum, 3-Scrum per release
- ❑ Internal and Offshore consulting team
- ❑ JIRA for issue and project tracking
- ❑ Confluence for documentation
- ❑ Internal GIT repository for all development projects
- ❑ External GitHub account for SDKs, reference firmware and sample codes. This is public repository – https://github.com/arrow-acs

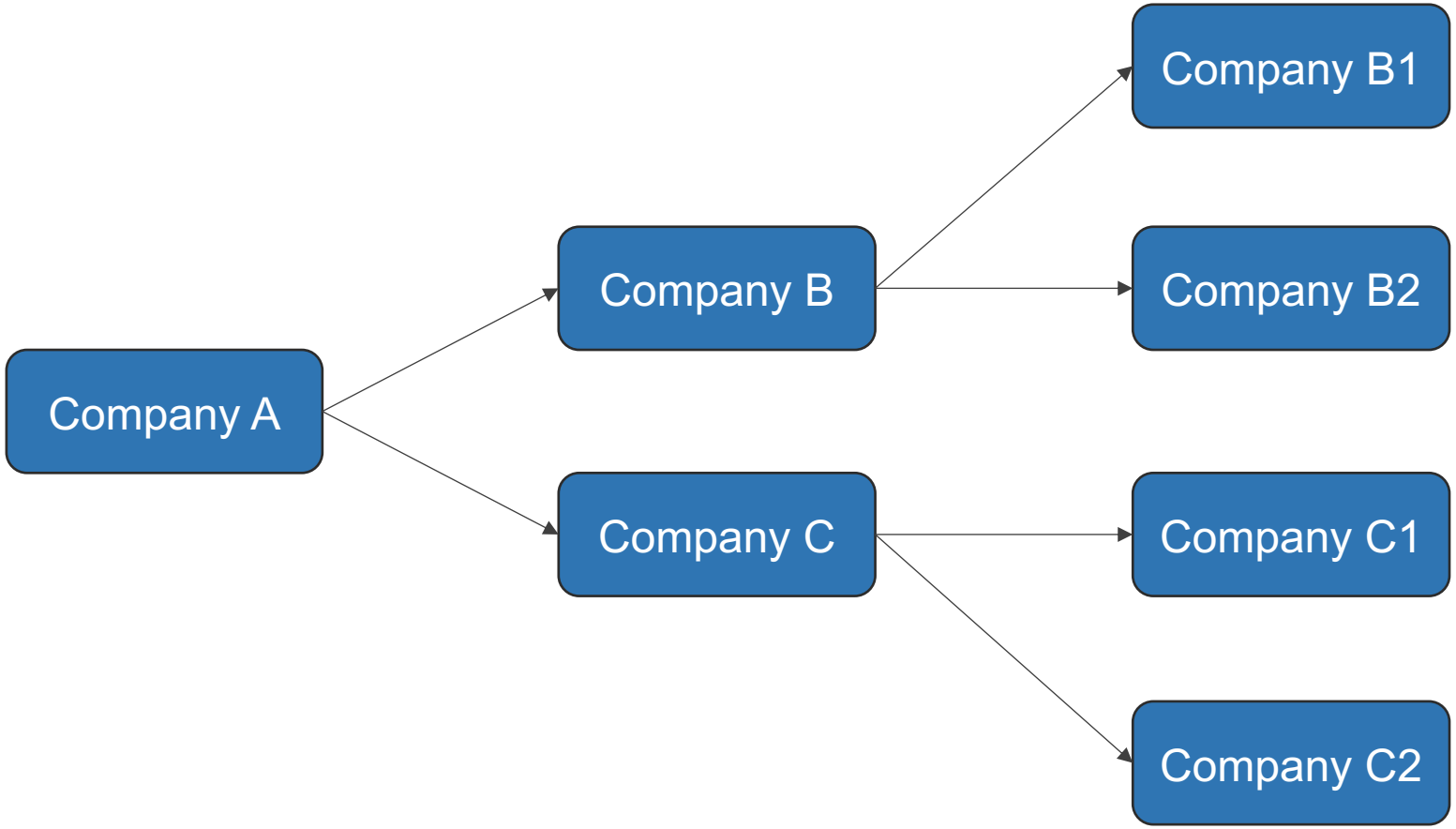
Developer’s Hub

- https://developer.arrowconnect.io
- Brandon Hall and Jim Lindblom manage the content of the site

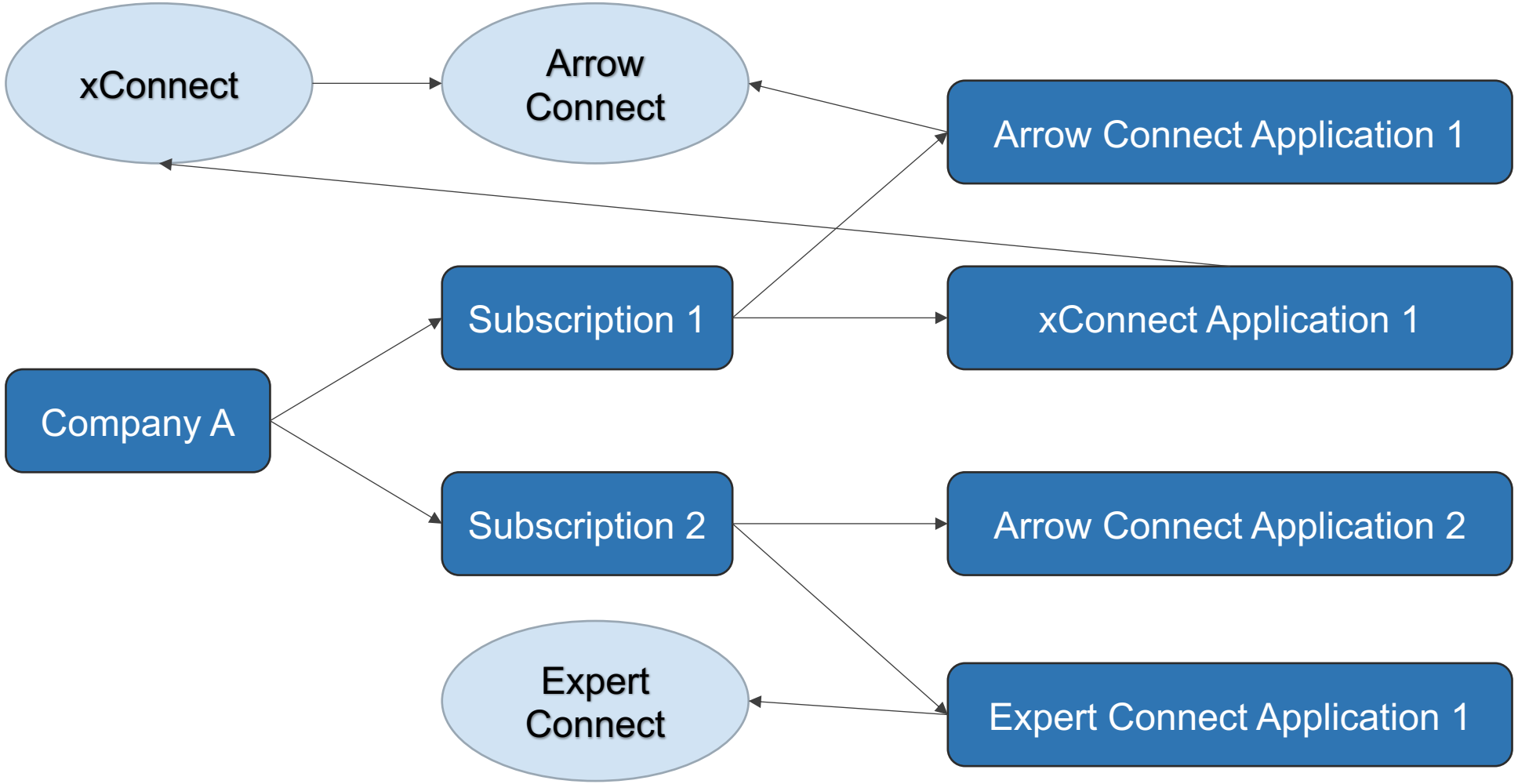
Pegasus Platform – Data Models

- ❑ Company (Tenant)
- ❑ Subscription
- ❑ Product
- ❑ Product Extension
- ❑ Application
- ❑ User
- ❑ Role
- ❑ Privilege
- ❑ Access Key

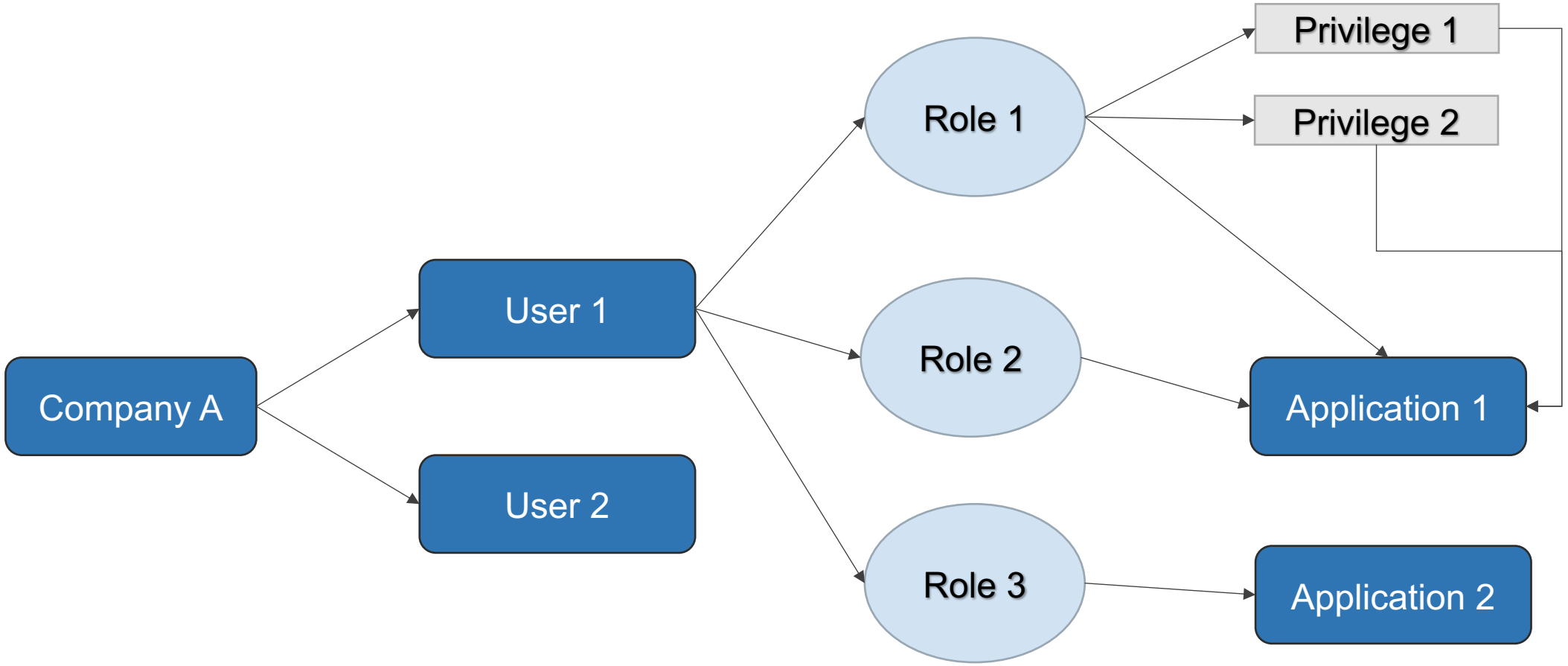
Pegasus – Company Model



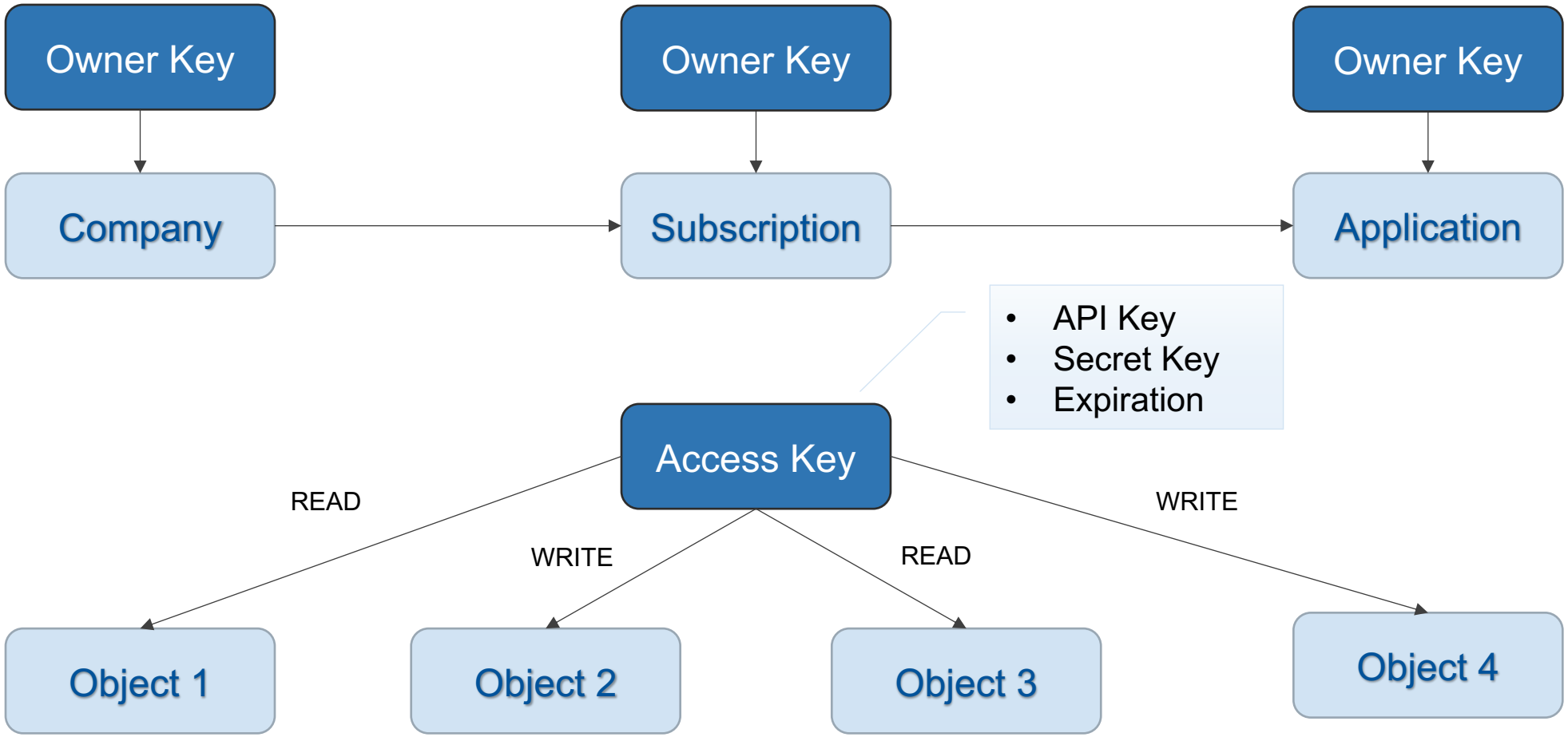
Pegasus – Subscription / Application Models



Pegasus – User RBAC Models



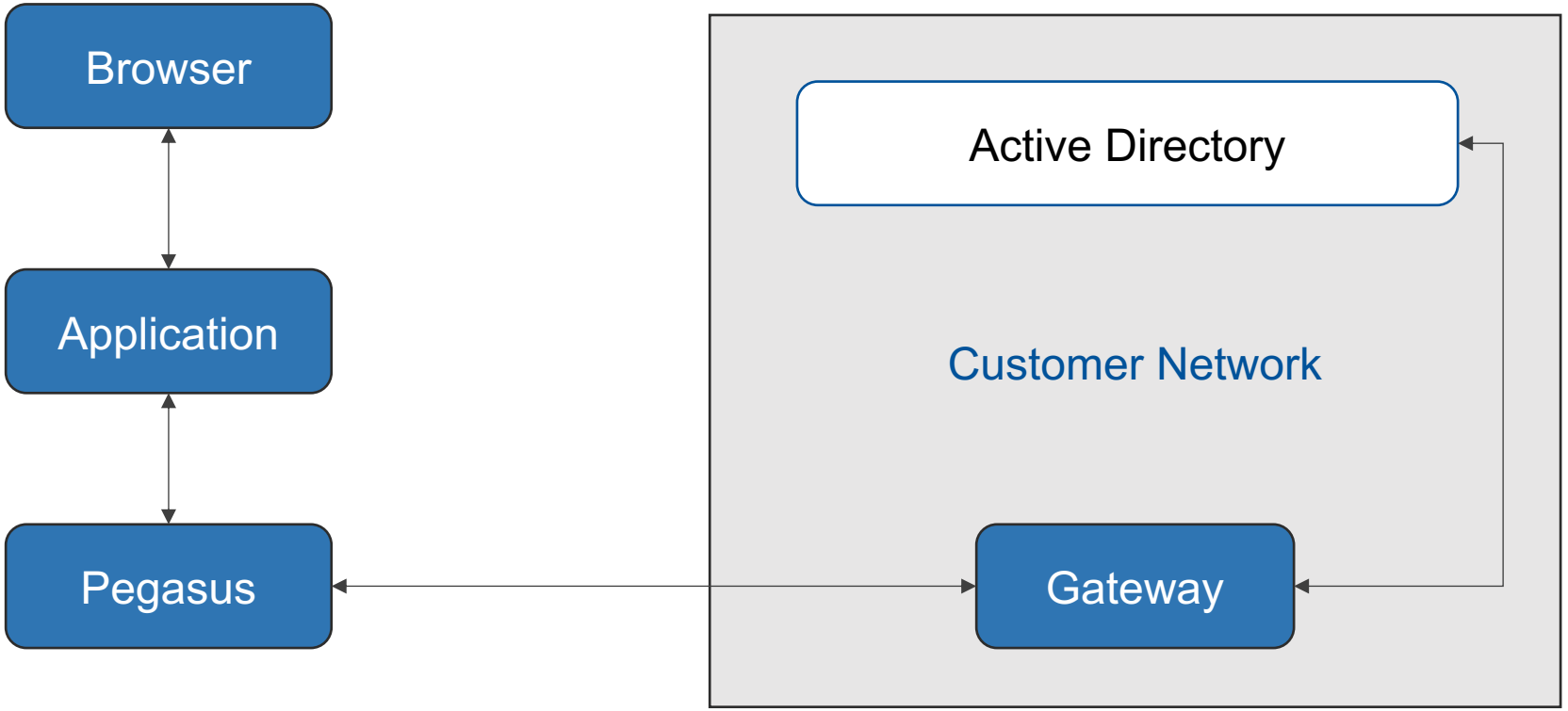
Pegasus – Access Key Model



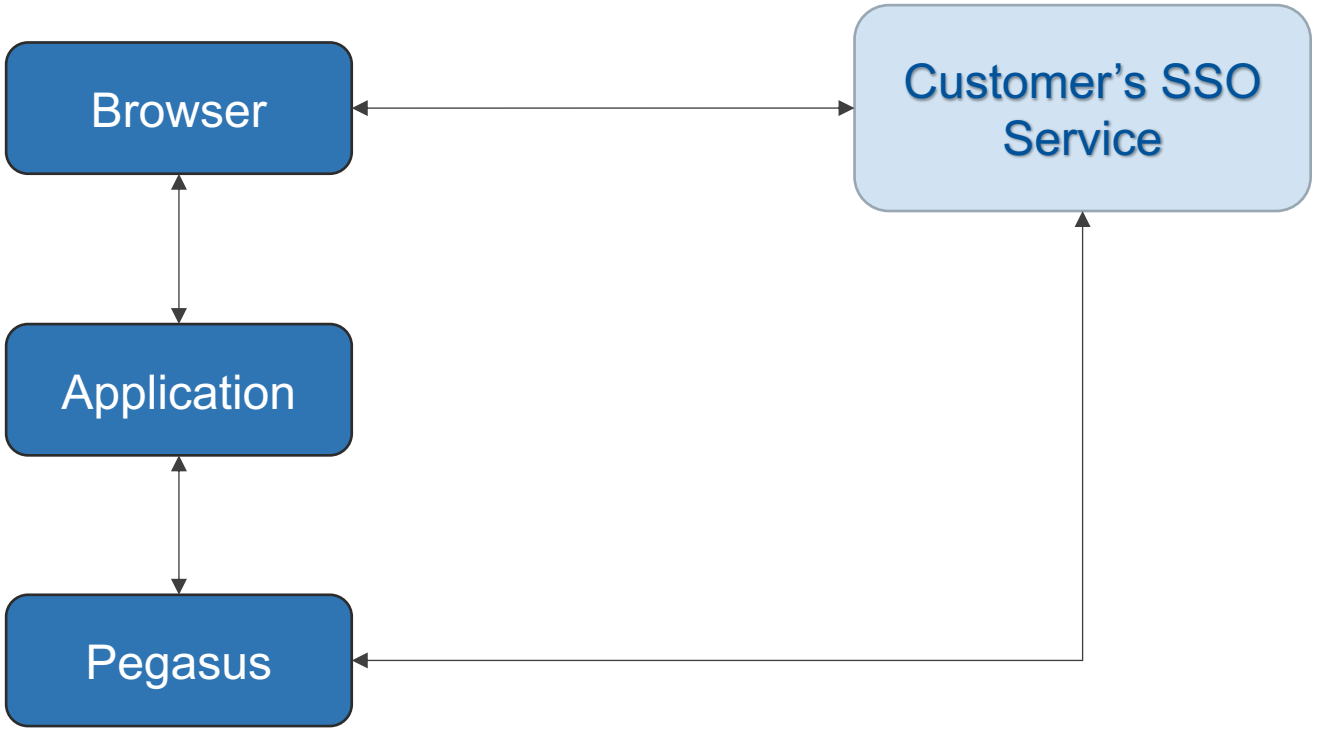
Pegasus – Internal Authentication

- ❑ Secured Internal Database
- ❑ Encrypted at rest
- ❑ Custom password hashing algorithm
- ❑ Strong password policy
- ❑ Account lock-out policy

Pegasus – Active Directory Authentication



Pegasus – Single Sign On Authentication (SSO)



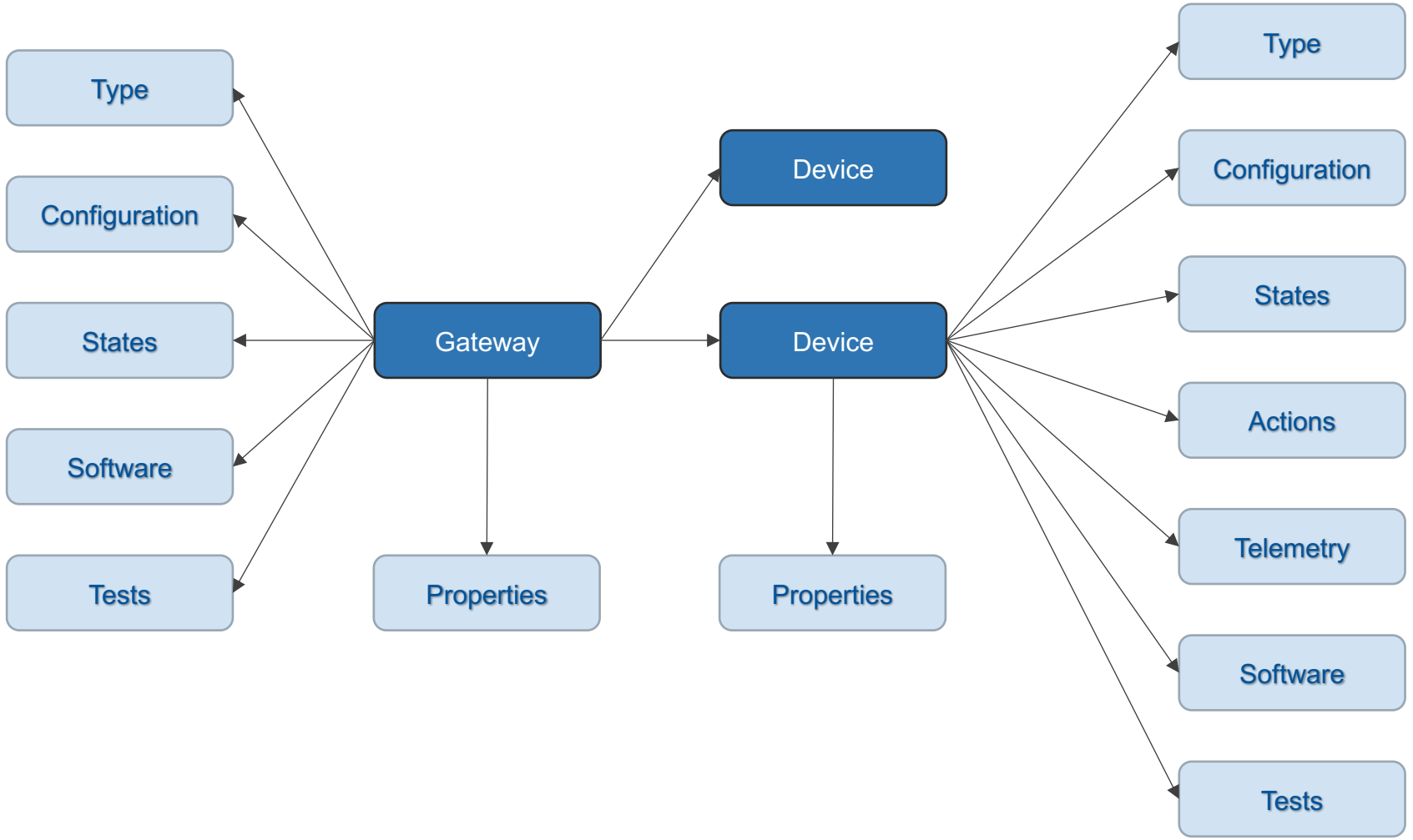
Pegasus Management Portal

- ❑ Ready for developer access by Q3 2017

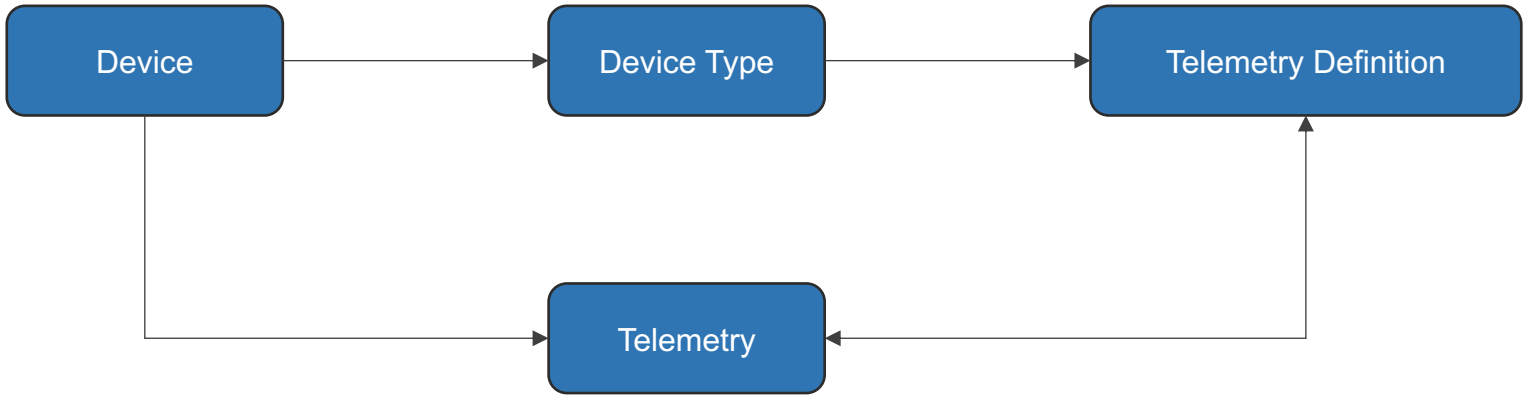
Arrow Connect Platform – Data Models

- ❑ Gateway
- ❑ Device
- ❑ Device Type
- ❑ Device State
- ❑ Device Action
- ❑ Node (Group)
- ❑ Telemetry
- ❑ Software Release
- ❑ Software Release Schedule
- ❑ Test Procedure
- ❑ Test Result

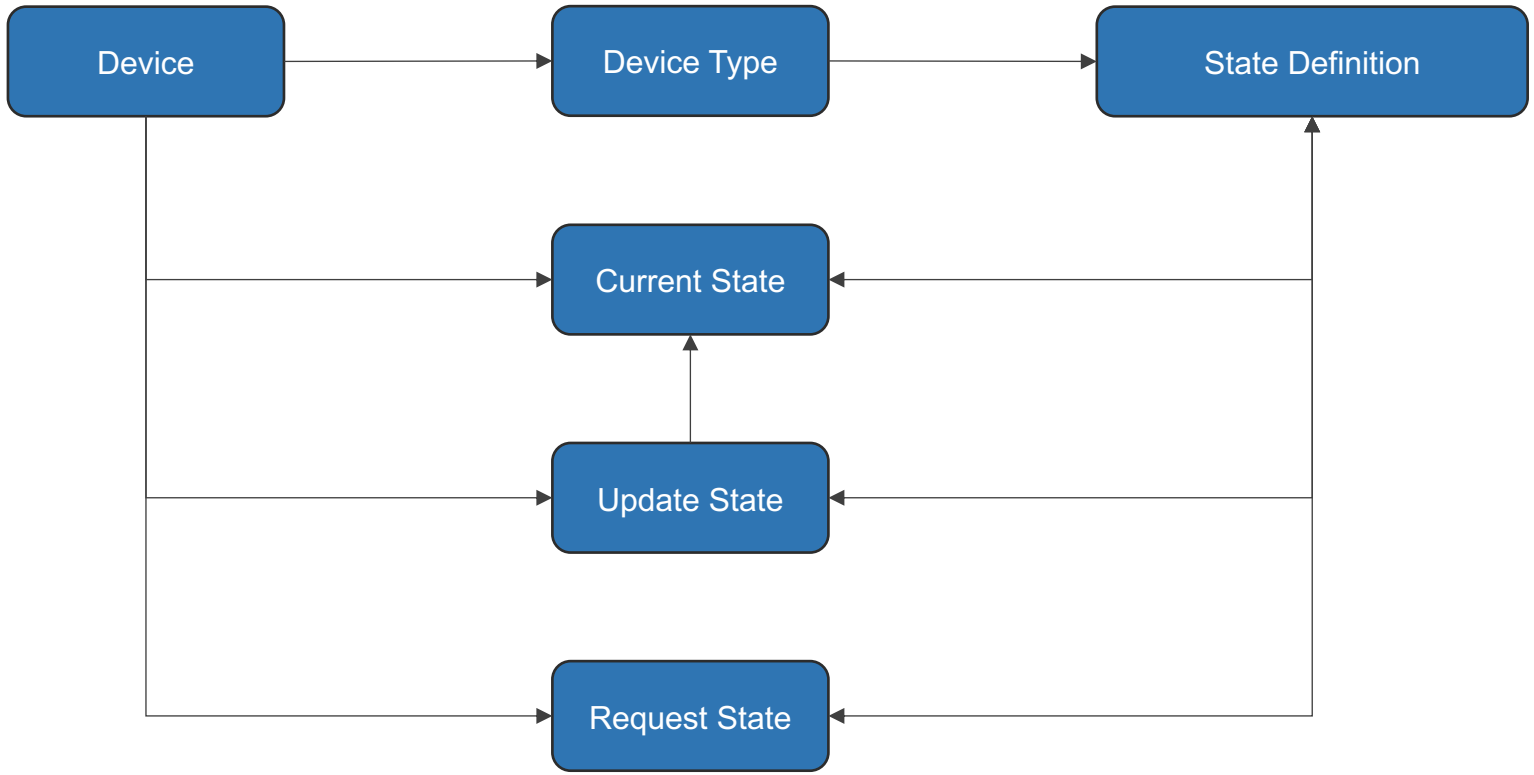
Arrow Connect – Gateway, Device Models



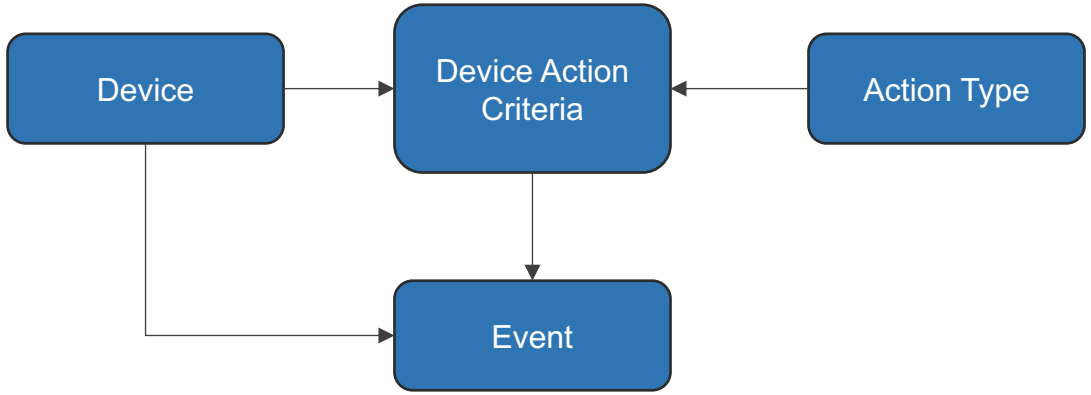
Arrow Connect – Device Type, Telemetry Models

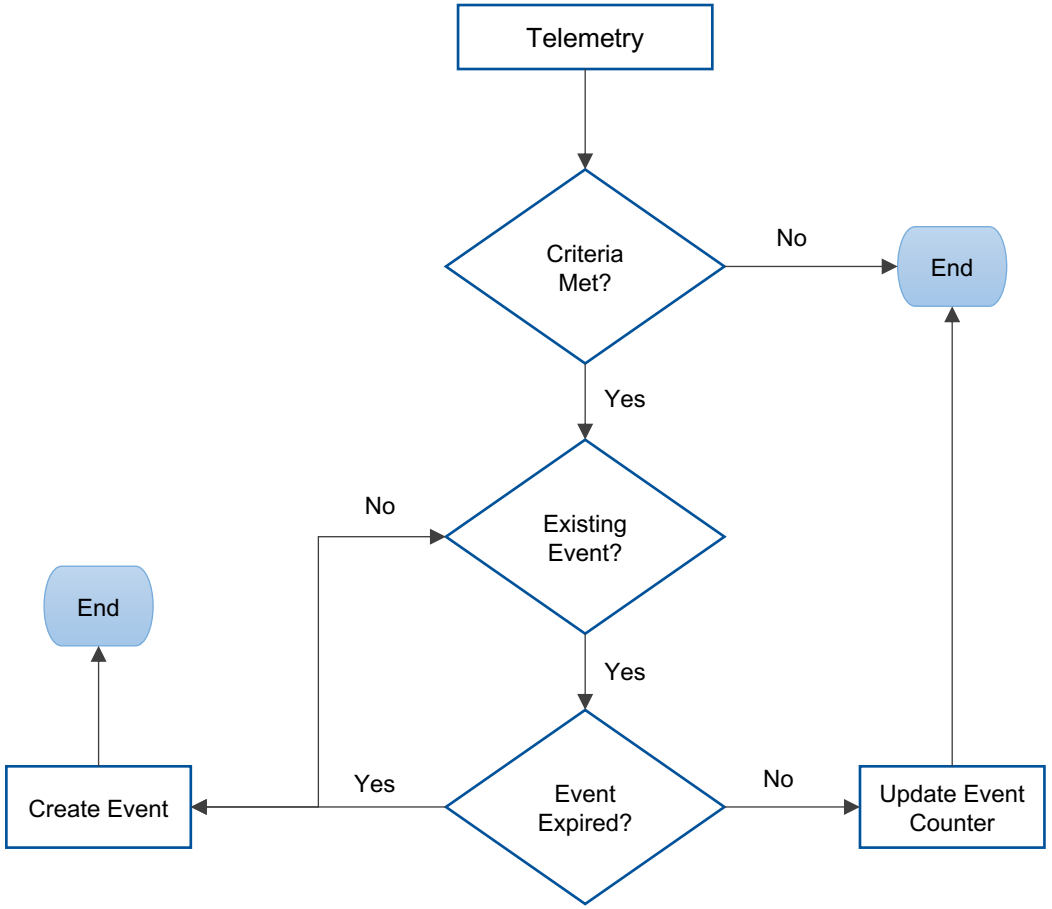


Arrow Connect – Device State Models

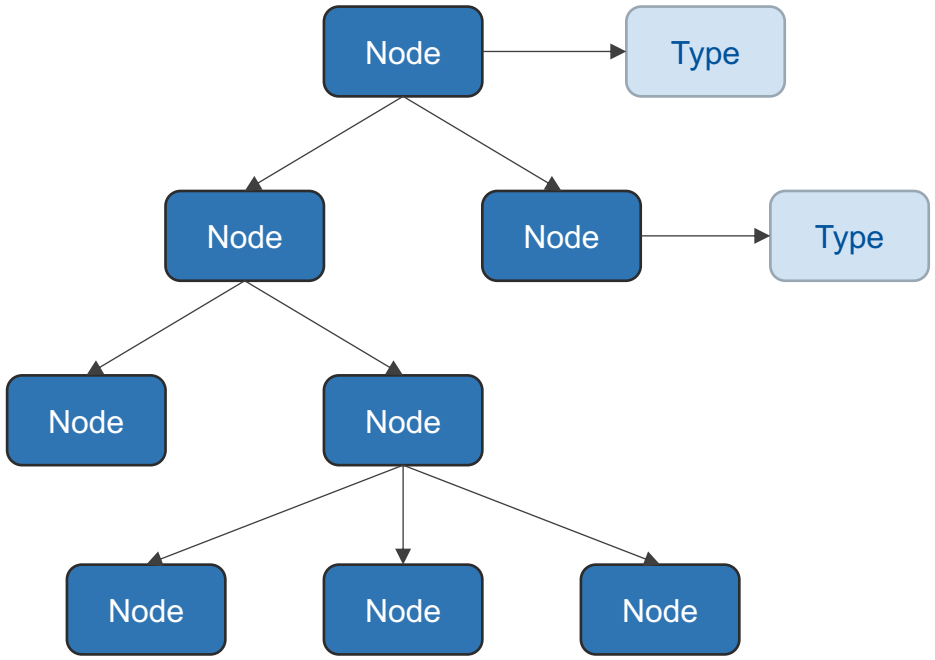


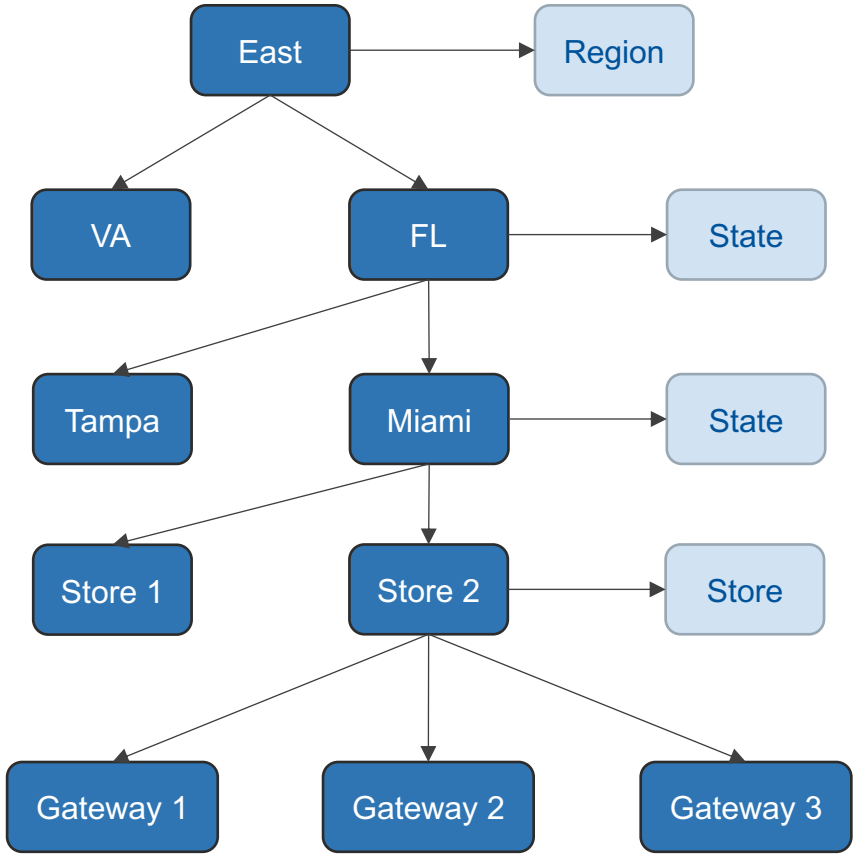
Arrow Connect – Device Action, Type Models



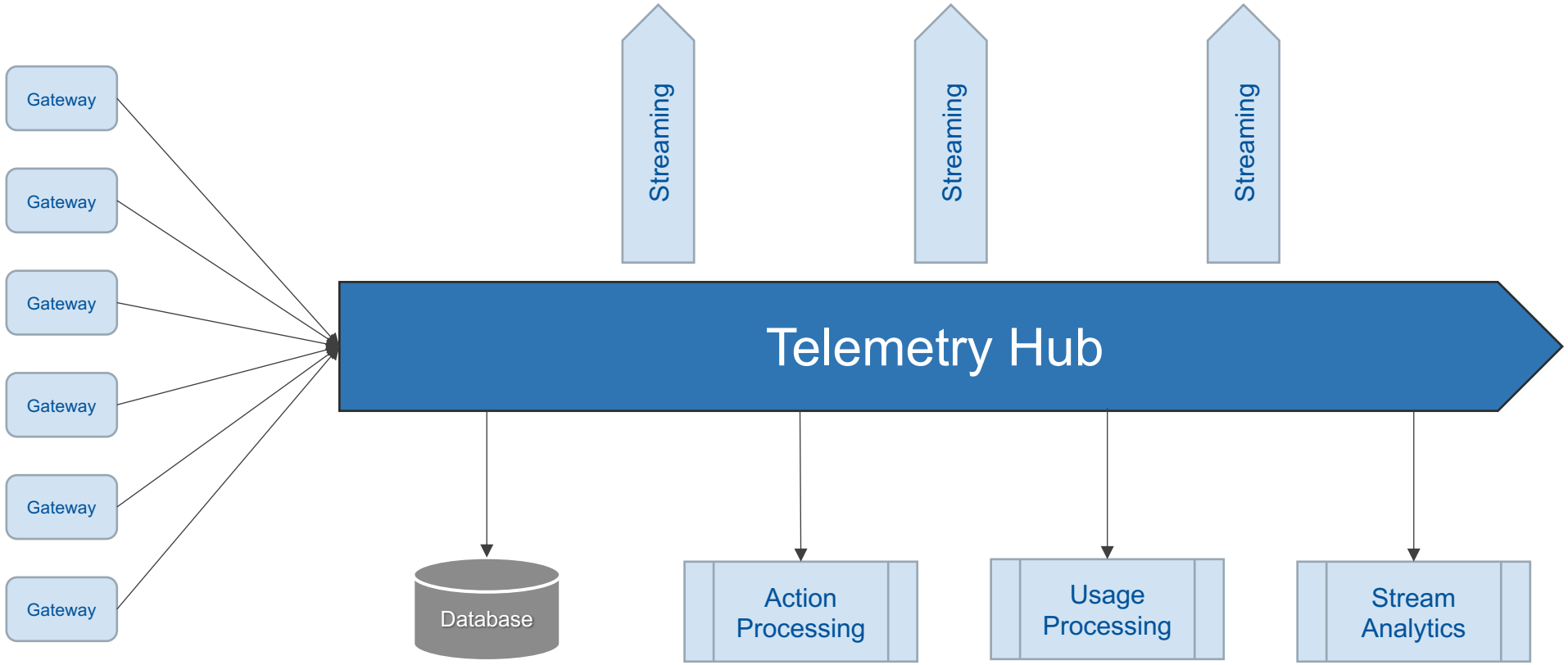


Arrow Connect – Node (Group) Model





Arrow Connect – Telemetry Processing



Arrow Connect – Firmware Management

- ❑ Centralized Software / Firmware repository
- ❑ Track both hardware and software versions of gateway and device
- ❑ Track historical upgrades
- ❑ Track upgrade path and compatibilities
- ❑ Perform upgrade by gateway, device, or group (location, type, etc.)
- ❑ Track upgrade progress
- ❑ Schedule upgrade by gateway, device, or group
- ❑ Upgrade status notification
- ❑ Reports
- ❑ Available July 2017

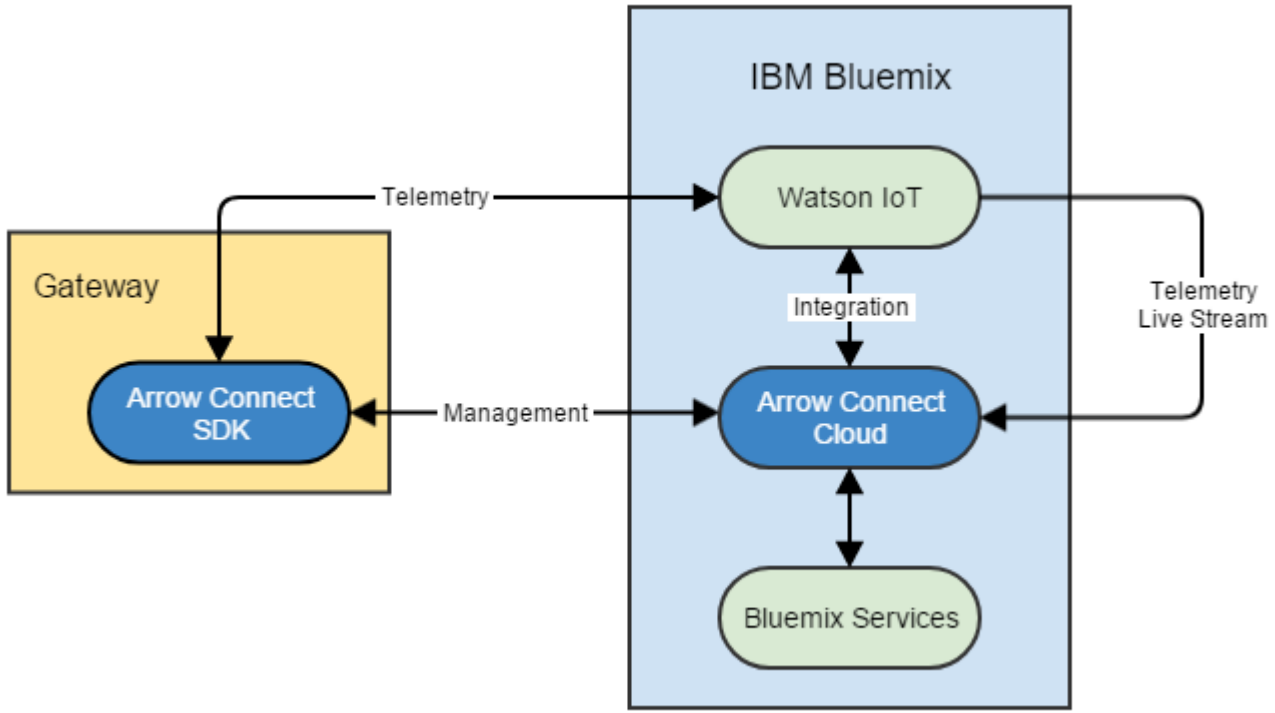
Arrow Connect – Firmware Test Management

- ❑ Test Procedure Respository
- ❑ Historical test results for gateways and devices
- ❑ Full API for integrated manufacturing testing
- ❑ Manual and Automated Testing
- ❑ Reports
- ❑ Available July 2017

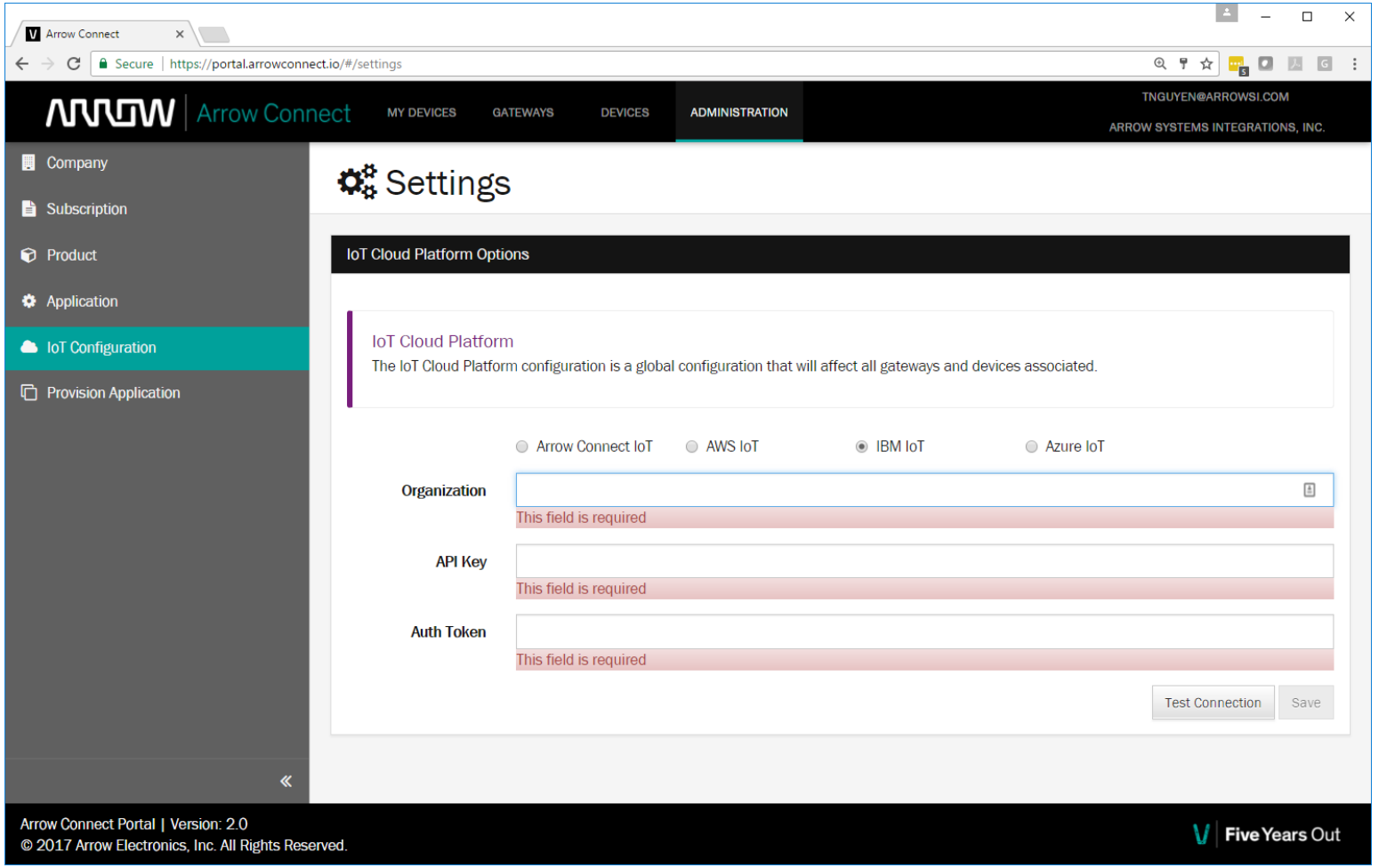
Arrow Connect – IoT Provider Integration

- Arrow Connect Cloud Platform provides a complete IoT solution. Integration with other IoT Provider such as IBM, Azure, and AWS is an option, not a requirement. Other IoT Provider usually charges a much higher cost for data ingestion in comparison to Arrow only solution
- Currently supports 3 IoT providers, more to come in the future.
  - IBM Watson IoT
  - Azure IoT Hub
  - AWS IoT
- Arrow Connect Cloud uses IoT Provider's SDK to integrate directly into their cloud platform in the back-end. This provide seamless integration of gateway/device respository and security requirement of the provider's platform
- Arrow Connect Gateway and SDK use IoT Provider's SDK to integrate directly into their cloud. Telemetry is sent from the gateway to the provider's cloud without having to be routed through Arrow Connect first
- IoT Provider configuration is done in Arrow Connect Portal. All it needs is an API account provided by the IoT Provider with enough privileges to execute the integration process
- Arrow Connect Gateway and SDK includes an internal process to download the credentials and configuration assigned by the IoT Provider from Arrow Connect

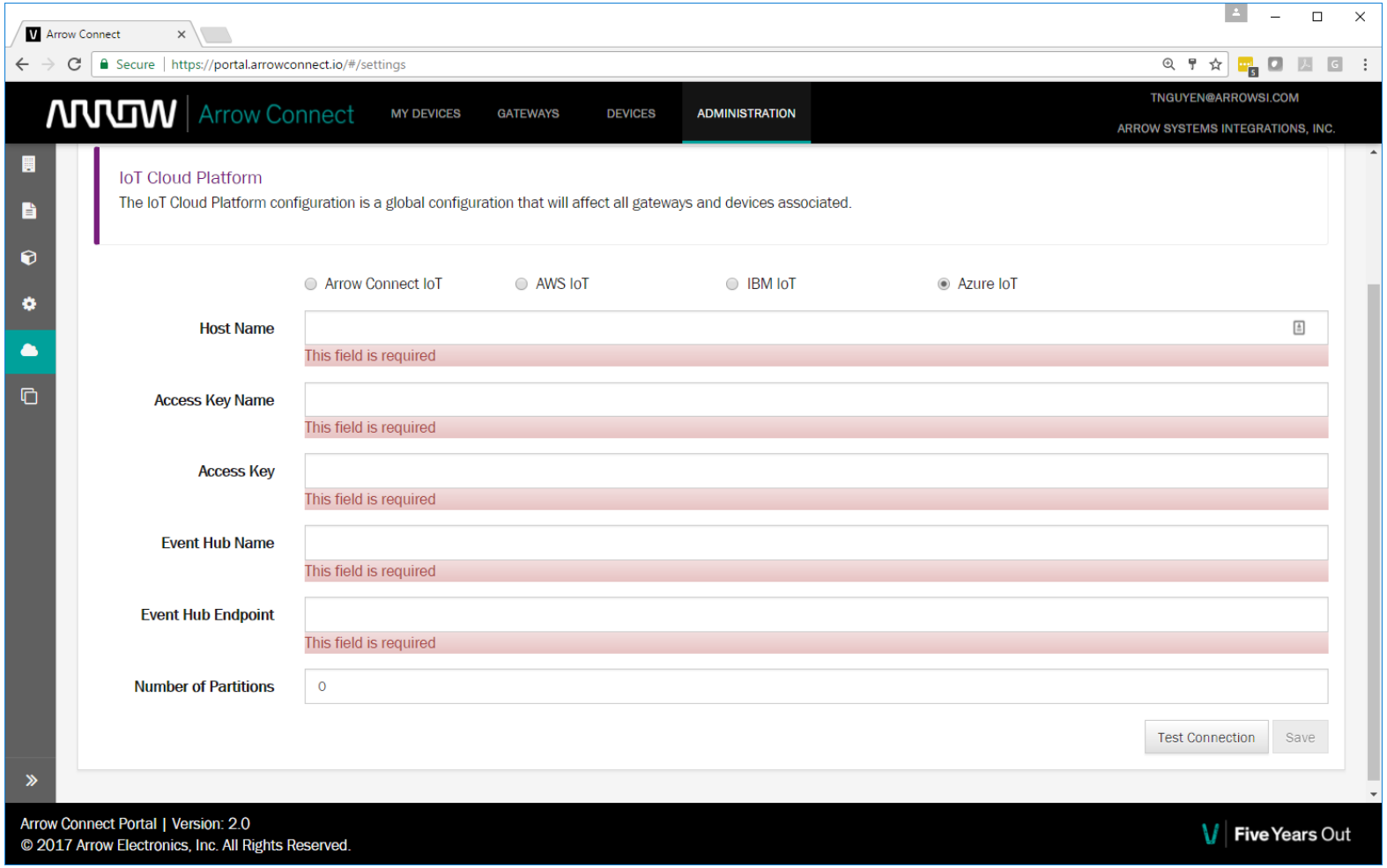
Arrow Connect – IBM Watson IoT Integration



Arrow Connect – IBM Watson IoT Configuration



Arrow Connect – Azure IoT Configuration



Arrow Connect – Portal

- ❑ Navigation, interaction with data models (CRUD)
- ❑ RBAC – user, admin, developer (defined in Pegasus portal)
- ❑ New features in upcoming releases
  - Firmware Management (July 2017)
  - Firmware Test Management (July 2017)
  - Missing UI components for existing functionalities (Q3/Q4 2017)
  - UX enhancements (Q3/Q4 2017)
  - User profiles, preferences (Q3/Q4 2017)
  - Dynamic Dashboards (Q3/Q4 2017)
  - Dynamic Widgets (Q3/Q4 2017)
  - Dynamic Demos with RBAC (Q3/Q4 2017)
  - Custom Reports (Q3/Q4 2017)
  - Customer's branding (Q3/Q4 2017)

Arrow Connect – REST API Lab (1/4)

- ❑ Register for a developer account – https://portal.arrowconnect.io
- ❑ Acquire Application Level API key from Arrow Connect Portal
  - Adminstration → Access Keys → Key's Owner field starts with `arw:pgs:app:xxx` → Click Hyperlink → Copy `Raw Api Key`
- ❑ Use Swagger UI for development and test - https://api.arrowconnect.io/swagger-ui.html
- ❑ Use `Raw API Key` for `x-auth-token` header
- ❑ Date format is ISO 8601 `yyyy-MM-ddTHH:mm:ssZ`
  - `Example: 2017-05-20T14:30:28Z`
- ❑ Gateway
  - Search gateways
  - Query details
  - Register new gateway
  - Control devices
  - Update device configuration (cloud to gateway to device)
  - Send command to device

Arrow Connect – REST API Lab (2/4)

- ❑ Device
  - Search devices
  - Query details
  - Create new device
  - Update device information (cloud only)
- ❑ Device State
  - Query states
  - Request state update
- ❑ Telemetry
  - Query telemetries
  - Query telemetry statistics
  - Send telemetry (see next slide for payload format)

Arrow Connect – REST API Lab (3/4)

- Payload is a JSON document which contains a map of keys to values. All values are in STRING format

```
{
    "key1"    : "value1",
    "key2"    : "value2"
}
```

- The key field has the following format. The delimiter is the | (pipe) character

```
<data-type>|<telemetry-name>
```

- Currently the API supports the following data type values
  - s - string
  - i - integer
  - f - floating point
  - b - boolean
  - d - date
  - dt - date and time
  - i2 - 2-tuple of integer values delimited by | (pipe) character
  - i3 - 3-tuple of integer values delimited by | (pipe) character
  - f2 - 2-tuple of floating point values delimited by | (pipe) character
  - f3 - 3-tuple of floating point values delimited by | (pipe) character
  - bi - binary value

Arrow Connect – REST API Lab (4/4)

- Sample payload

```
{
    " _|deviceHid"      : "abcdefg1234567890",
    "s|color"          : "red",
    "i|counter"        : "1246",
    "f|temperature"    : "60.5",
    "b|valid"          : "true",
    "d|expiredDate"    : "09/01/2016",
    "i2|dimension"     : "1920|1080",
    "f3|3dCoords"      : "23.3|24.2|83.1",
    ....
}
```

- Since the current REST API for sending telemetry does not include the deviceHid in the URL, you need to add it to the payload manually. New method will be added in the next release

Mobile Applications

- Download iOS and Android application (Enterprise Application Deployment)
  https://prism.arrowconnect.io/ipa/download.html
- Applications will be published in App Store and Google Play in Q3 2017
- Support multiple user profiles even from different companies and application instances
- Support many BLE devices out-of-the-box (more to come)
- Connect to multiple devices simultaneously
- Gateway and device configuration
- Manage device actions
- Telemetry summary and historical
- Support GPS

Mobile Applications - Lab

- ❑ Acquire the Application Code of your developer's account on Arrow Connect Portal
  - Administration → Settings → Application → Application Code
- ❑ Register new Profile or add Profile to existing installation with Application Code
- ❑ Internal phone device
  - Send telemetry to cloud
  - Control device from cloud
  - Discover other functionalities – dashboard, historical data, configurations, actions
- ❑ Thunderboard React
- ❑ Sensor Puck

Prepare Dragonboard 410c

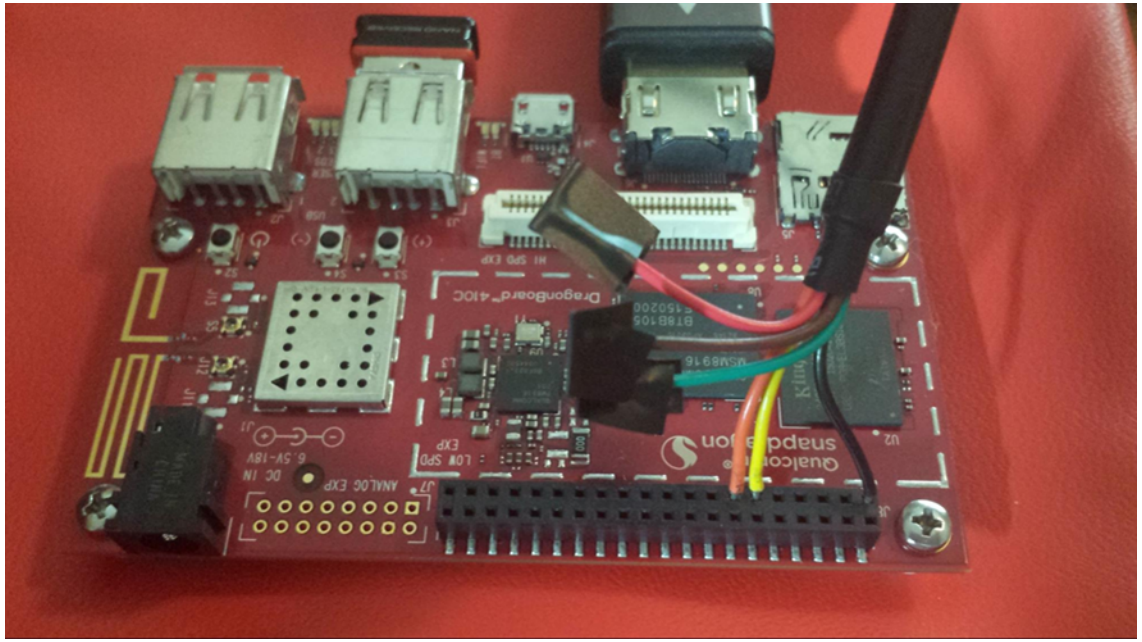
- ❑ Set S6 switches in the back of the board to 0-0-0-0-0
- ❑ Connect a micro USB-USB cable from the micro USB port on the DB410c to a USB port on the laptop
- ❑ Press and hold the Vol (-) button on the DB410c (S4)
- ❑ While pressing S4 button, simultaneously power up the DB410c. It will come up in fastboot mode

Install Debian OS

- ❑ Open terminal window on laptop
  - `cd c:\acn\platform-tools`
  - `fastboot devices`
  - `fastboot flash boot boot-linaro-jessie-qcom-snapdragon-arm64-20161006-144.img`
  - `fastboot flash rootfs linaro-jessie-developer-qcom-snapdragon-arm64-20161006-144.img`
- ❑ Power off board
- ❑ Unplug USB cable

Prepare System Console

- ❑ Connect FTDI cable to board as shown
- ❑ Connect FTDI cable to USB port on laptop
- ❑ Power on board
- ❑ Launch Tera Term application and connect to Serial Port using settings 115200-8-n-1-n
- ❑ Run command to set up Wi-Fi, input SSID and Passphrase
  - `nmtui`
- ❑ Check network access
  - `ifconfig`
  - `ping google.com`



Software Dependencies (1/3)

- Update OS
  - `apt update`
  - `apt –y upgrade`
- Check Java Runtime Environment (requires version 8+)
  - `dpkg-query -l | grep openjdk`
  - `java –version`
- Support Serial Devices
  - `apt install -y librxtx-java`
- Support BLE Devices
  - Requires BlueZ
    `dpkg-query -l | grep bluez`
  - Use Kura
    `apt install –y bluez-hcidump`
  - Use DBus (experimental mode)
    `apt install –y libdbus-java`
    `apt purge –y openjdk-7-jre-headless`

Software Dependencies (2/3)

- Support GPIO / I2C Devices (requires source compilation)
  - MRAA https://iotdk.intel.com/docs/master/mraa/
  - UPM https://iotdk.intel.com/docs/master/upm/
- Prepare dependencies
  - `apt install -y arduino-mk arduino git build-essential autoconf libtool swig3.0 python-dev cmake pkg-config libpcre3-dev`
  - `apt install –y openjdk-8-jdk`
  - `export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-arm64`
- Compile MRAA
  - `mkdir ~/dev; cd ~/dev`
  - `git clone https://github.com/intel-iot-devkit/mraa`
  - `mkdir mraa/build; cd mraa/build`
  - `cmake -DBUILDSWIGNODE=OFF -DBUILDSWIGJAVA=ON ..`
  - `make`
  - `make install`
  - `ldconfig /usr/local/lib/`

Software Dependencies (3/3)

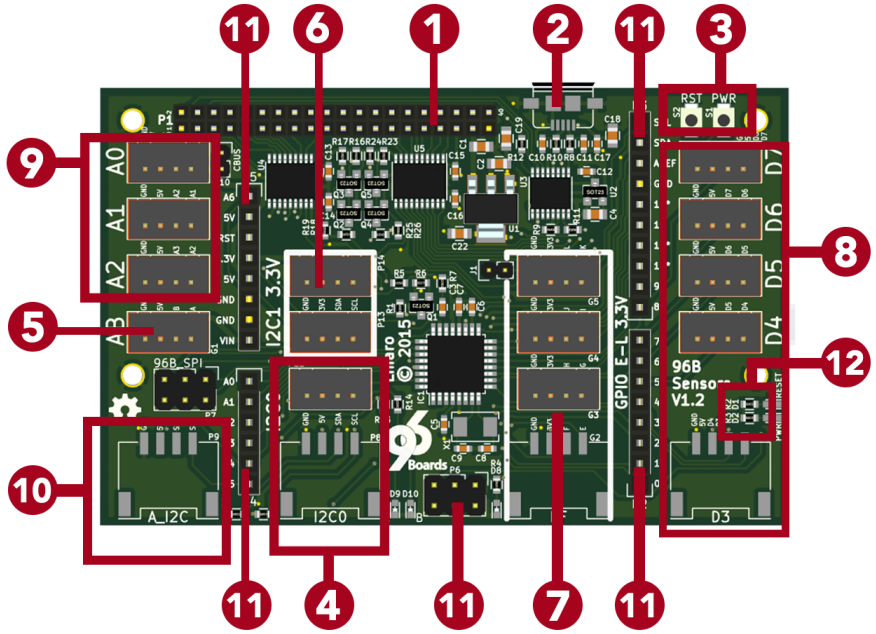
- Compile UPM - This step takes about an hour
  - `cd ~/dev`
  - `git clone https://github.com/intel-iot-devkit/upm`
  - `mkdir upm/build; cd upm/build`
  - `cmake -DBUILDSWIGNODE=OFF -DBUILDSWIGJAVA=ON ..`
  - `make`
  - `make install`
  - `ldconfig /usr/local/lib/libupm-*`
- Fix an issue with include files
  - `apt install –y libmraa-dev libupm-dev`
  - `mv /usr/local/include/upm /usr/local/include/upm-backup`

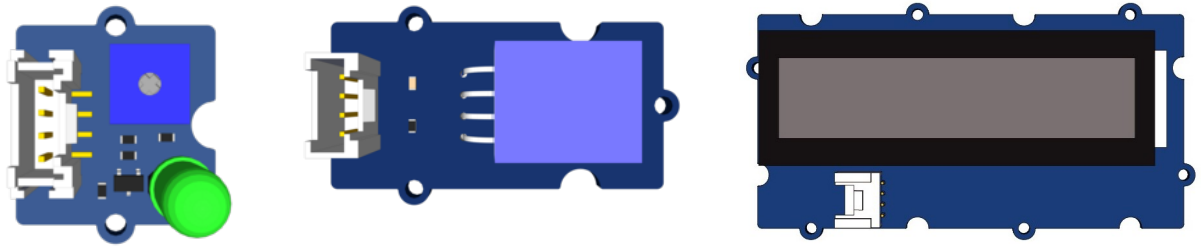
GPIO and I²C

- General-purpose input/output (GPIO) is a generic pin on an integrated circuit or computer board whose behavior—including whether it is an input or output pin—is controllable by the user at run time
  - GPIO pins can be configured to be input or output
  - GPIO pins can be enabled/disabled
  - Input values are readable (typically high or low)
  - Output values are writable/readable
  - Input values can often be used as IRQs (typically for wakeup events)
- I²C (Inter-Integrated Circuit) is a multi-master, multi-slave, packet switched, single-ended, serial computer bus, typically used for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication

Grove Sensor Board (1/2)

- ❑ Connect LED module to GPIO_E (#7)
- ❑ Connect LCD module to I2C-0 (#4)
- ❑ Connect DHT module to A0 (#9)





1. Low Speed Expansion connector
2. USB UART console connector
3. Reset and Power buttons
4. 5V I2C Grove connectors
5. 5V GPIO Grove connector
6. 3.3V I2C Grove connectors
7. 3.3V GPIO Grove connectors
8. ATMEGA D3-D7 Grove connectors
9. ATMEGA A0-A2 Grove connectors
10. ATMEGA I2C Grove connector
11. ATMEGA Arduino compatible socket
12. ATMEGA Reset and Power LEDs

Grove Sensor Board (2/2)

- Run examples
  - `cd ~/dev`
  - `wget https://content.arrowconnect.io/public/training/05-2017/grove-examples.tar.gz`
  - `tar xfvz grove-examples.tar.gz; rm grove-examples.tar.gz`
  - `cd acn`
  - `g++ led.cpp -o led -g -Wall -lmraa`
  - `./led`
  - `g++ lcd.cpp -o lcd -g -Wall -lupm-i2clcd`
  - `./lcd`
  - `cd dht; make upload reset_sty`
  - `python dht.py`

led.cpp

```
#include "mraa.hpp"

#define GPIO_E 27
#define SLEEP_TIME 1

using namespace std;

int main(int argc, char* argv[])
{
    mraa::Gpio* gpio;
    gpio = new mraa::Gpio(GPIO_E);
    gpio->dir(mraa::DIR_OUT);

    while (true) {
        gpio->write(0);
        sleep(SLEEP_TIME);
        gpio->write(1);
        sleep(SLEEP_TIME);
    }

    delete gpio;
    return 0;
}
```

lcd.cpp

```
#include <unistd.h>
#include "upm/jhd1313m1.hpp"

#define I2C_BUS  0
#define SLEEP_TIME 2

void display(upm::Jhd1313m1* lcd, string str1, string str2, int red, int green, int blue)
{
           lcd->clear();
           lcd->setColor(red, green, blue);
           lcd->setCursor(0,0); /* first row */
           lcd->write(str1);
           lcd->setCursor(1,2); /* second row */
           lcd->write(str2);
           sleep(SLEEP_TIME);
}
```

```
   lcd = new upm::Jhd1313m1(I2C_BUS, 0x3e, 0x62);
   while (true)
   {
          display(lcd, str1, red, RGB_RED);
          display(lcd, str2, grn, RGB_GRN);
          display(lcd, str3, blu, RGB_BLU);
   }
```

dht.ino

```
#include "DHT.h“

DHT dht(A0, DHT11);

void setup()
{
          Serial.begin(9600);
          dht.begin();
}

void loop()
{
          float h = dht.readHumidity();
          float t = dht.readTemperature();
          if (isnan(t) || isnan(h))
          {
                    Serial.println("Failed to read from DHT");
                    return;
          }
          Serial.print(h);
          Serial.print(",");
          Serial.println(t);
          delay(2000);
}
```

dht.py

```
import serial
dht = serial.Serial('/dev/tty96B0', 9600)
print(dht.readline())
```

Linux Gateway (Selene)

- ❑ Fully loaded, production-ready Arrow Connect gateway developed in Java
- ❑ Contains Main Engine and optional Web Portal
- ❑ Embedded database – H2 or SQLite
- ❑ Data Bus option – File, Redis, MQTT or RabbitMQ
- ❑ More than a dozen out-of-the-box PnP device adapters
- ❑ Many protocols – BLE, ZigBee, DUST, LoRa, GPIO, I2C, Modbus, etc.

Install Selene (Dragonboard 410c)

- Download selene
  - `cd /opt`
  - `wget https://content.arrowconnect.io/public/training/05-2017/selene-db410c.tar.gz`
  - `tar xfvz selene-db410c.tar.gz`
  - `rm selene-db410c.tar.gz`
- Modify file /opt/selene/config/devices/self.properties
  - `name = <gateway-name>`
  - `uid = <gateway-uid> (if omitted, default uid is gateway-<mac-address>)`
  - `apiKey = <your-application-api-key>`
  - `secretKey = <your-application-secret-key>`
- Start selene
  - `cd /opt/selene; bin/start.sh`
  - `tail –f log/app.log`
- Stop selene
  - `cd /opt/selene; bin/stop.sh`

Selene Directory Structure

- Default installation directory – `/opt/selene`
- `bin/*` – start/stop, deamon script
- `config/devices/*` – configuration files of all bootstrap devices
- `config/devices/self.properties` – main configuration file
- `config/log4j2.xml` – log file settings
- `config/selene.properties` – global configuration file
- `config/samples/*` – sample configuration files of supported devices
- `db/*` – database files
- `databus/*` – file-based message queue
- `lib/*` – selene binaries
- `log/*` – log files
- `update/*` – software upgrade files

Selene - Grove Examples (1/3)

- Copy the following files from directory `/opt/selene/config/samples` to directory `/opt/selene/config/devices`
  - `/opt/selene/config/samples/training-led.properties`
  - `/opt/selene/config/samples/training-lcd.properties`
- Modify the files to set your own device “name” and “uid”
  - `name = db410c-your-name-led`
  - `uid  = db410c-your-name-led`
- Start selene
- Use Swagger UI and Device State API to communicate with these devices
- LED and LCD devices use PeripheralModule which interacts with MRAA and UPM using Java Binding
- Exercise – Configure a new device read light sensor, see next slide

Selene - Grove Examples (2/3)

- Use Swagger UI and Device State API to communicate with LED device
  - `{ "states" : { "led" : "on" } }`
  - `{ "states" : { "led" : "off" } }`
- Use Swagger UI and Device State API to communicate with LCD device
  - `{ "states" : { "backlight" : "on" } }`
  - `{ "states" : { "display" : "off" } }`
  - `{ "states" : { "display" : "on" } }`
  - `{ "states" : { "backlight": "on", "color" : "FF0000" } }`
  - `{ "states" : { "row1" : "Hello", "row2" : "Arrow Connect" } }`

Selene - Grove Examples (3/3)

- Grove Sensor classes
  - com.arrow.device.peripheral.devices.AngleSensor
    - absoluteAngle, relativeAngle
  - com.arrow.device.peripheral.devices.Button
    - pressed
  - com.arrow.device.peripheral.devices.Led
    - led
  - com.arrow.device.peripheral.devices.LightSensor
    - light
  - com.arrow.device.peripheral.devices.RangerSensor
    - distance, working
  - com.arrow.device.peripheral.devices.Relay
    - relay
  - com.arrow.device.peripheral.devices.Slide
    - rawValue, refVoltage, voltage
  - com.arrow.device.peripheral.devices.TemperatureSensor
    - temperature
  - com.arrow.device.peripheral.devices.jhd1313m1.Jhd1313m1
    - backlight, color, display, row1, row2

BLE and Thunderboard React

- Bluetooth Low Energy (BLE, Bluetooth Smart) is a wireless personal area network technology designed and marketed by the Bluetooth Special Interest Group aimed at novel applications in the healthcare, fitness, beacons, security, and home entertainment industries. Compared to Classic Bluetooth, Bluetooth Smart is intended to provide considerably reduced power consumption and cost while maintaining a similar communication range
- AES-128 encryption
- Low Power Design - Keeping radio off
- Advertising Channels vs Data Channels
- Generic Attribute Profile (GATT)
  - Services - https://www.bluetooth.com/specifications/gatt/services
  - Characteristics - https://www.bluetooth.com/specifications/gatt/characteristics
  - Descriptors - https://www.bluetooth.com/specifications/gatt/descriptors
- Data is exposed as Attributes – handle / type / value
- Selene supports Thunderboard React demo out-of-the-box

Working with BlueZ Stack (1/2)

- Check bluetoothd process
  - `systemctl status bluetooth`
- Check Bluetooth module
  - `hciconfig`
  - `hciconfig hci0 up`
  - `hciconfig hci0 down` (may interfere with wi-fi, do not run from ssh session)
- Scan for BLE devices
  - `hcitool lescan`

    `00:0B:57:0C:28:C8 Thunder React #10440`
- Connect to device using gatttool
  - `gatttool –b <ble-address> -I`
  - `connect`

    `Connection successful`

Working with BlueZ Stack (2/2)

- List services
  - `primary`

```
attr handle: 0x0013, end grp handle: 0x0016 uuid: 0000180f-0000-1000-8000-00805f9b34fb
attr handle: 0x0020, end grp handle: 0x0029 uuid: 00001815-0000-1000-8000-00805f9b34fb
```

- List characteristics
  - `char-desc`

```
handle: 0x0015, uuid: 00002a19-0000-1000-8000-00805f9b34fb
handle: 0x0027, uuid: 00002a56-0000-1000-8000-00805f9b34fb
```

- Read / Write characteristic
  - `char-read-hnd 0x0015`

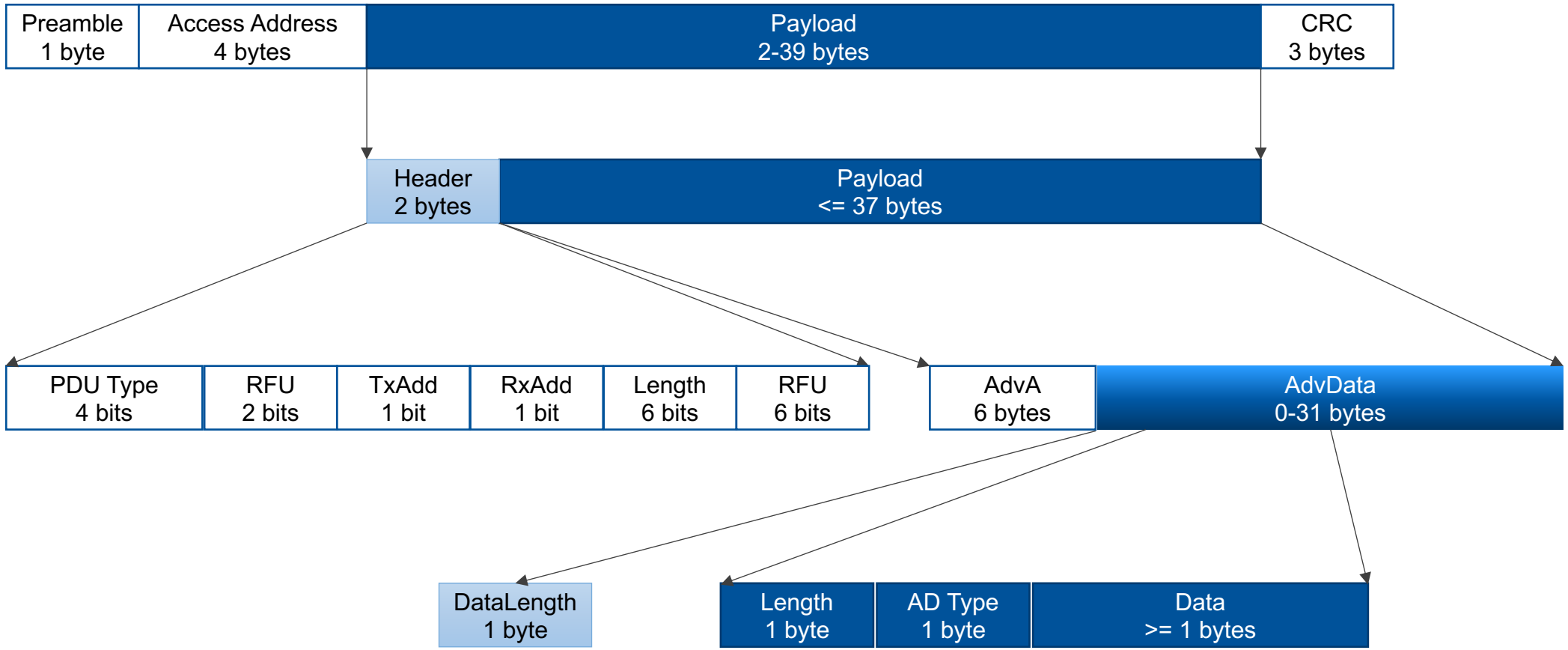
```
Characteristic value/descriptor: 61
```

  - `char-write-req 0x0027 [00, 01, 04, 05]`

```
both off, blue, green, both on
```

- Disconnect
  - `disconnect`
- Exercise
  - `Read humidity and temperature`

Sensor Puck – BLE Beacon (1/2)



Sensor Puck – BLE Beacon (2/2)

- On the TTY console window, type this command
  - `hcidump -x -R`
- Open an SSH window (linaro / linaro), type this command
  - `sudo hcitool lescan --duplicates`
- Turn on Sensor Puck

```
04 3E 1F 02 01 03 00 A8 75 E0 CA 81 D4 13 02 01 06 0F FF 35
12 00 0A A8 75 01 02 0F 01 2A 00 00 1E C1
```

  - MAC address = D4 81 CA E0 75 A8
  - Total length of next report = 0x13
  - AD Type 01 (flags), AD Type FF (manufacture specific data)
  - Company ID = 0x1235
  - Mode 00 (environment), Mode 01 (biometric)
  - Sequence = 0x0A, Address = 0x75A8
  - Humidity = 0x0201/0x0A, Temperature = 0x010F/0x0A, Light = 0x002A*2, UV = 0x00, Voltage = 0x1E

CLI Device (1/2)

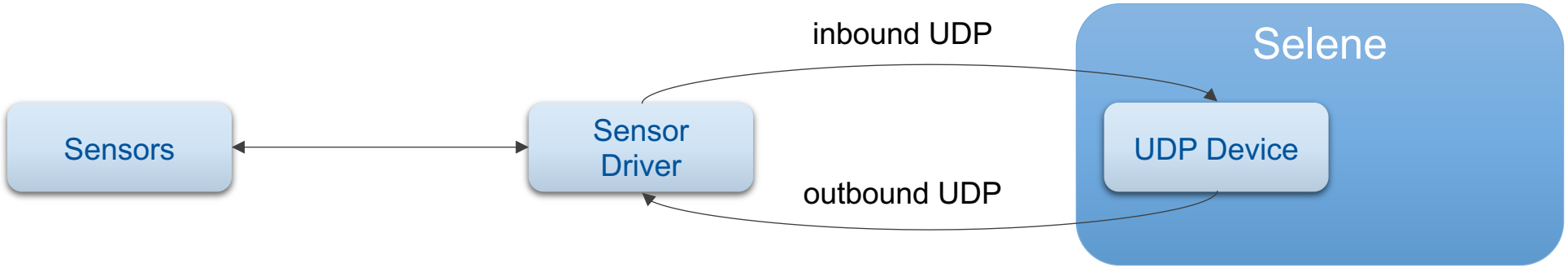
- Available configuration parameters
  - `type=<your-device-type>`
  - `command=ping -n 1 google.com`
  - `commandIntervalInSecs=2`
  - `dataType=string`
  - `arrayDelimiter=,`
- There are 7 options for `dataType`
  - `string`
  - `long`
  - `double`
  - `string-array`
  - `long-array`
  - `double-array`
  - `raw`
- The device executes the command at defined interval. Value is in seconds
- `arrayDelimiter` is required when `dataType` is of type `array`
- For `raw dataType`, command's output must follow Arrow Connect JSON format

CLI Device (2/2)

- Exercise
  - Configure a CLI device to send humidity and temperature from Grove DHT sensor to Arrow Connect every 5 seconds
  - Hint: use full file path

UDP Device (1/2)

- Available configurations parameters
  - `type=<your-device-type>`
  - `address=127.0.0.1` — `inbound UDP (device to selene)`
  - `port=41000` — `inbound UDP port`
  - `deviceAddress=127.0.0.1` — `outbound UDP (selene to device)`
  - `devicePort=42000` — `outbound UDP port`
- Inbound payload must following Arrow Connect JSON format.



UDP Device (2/2)

- Configure UDP device
  - `cd /opt/selene`
  - `cd config/samples/training-udp.properties config/devices/`
- Modify configuration paramaters
- Start selene with this new device
- Open an SSH terminal
  - `cd ~dev; mkdir udp; cd udp`
  - `wget https://content.arrowconnect.io/public/training/05-2017/udp-client.jar`
  - `java –jar udp-client.jar <outbound-port> <inbound-port>`
- Exercise
  - Send telemetries from UDP client to Cloud
    - `{ "f|temperature" : "25.6" }`
  - Send commands from Cloud to UDP client
    - `greetings`
    - `{ "msg" : "hello" }`

Arrow Connect Embedded

JIM LINDBLOM

Install Selene on Windows

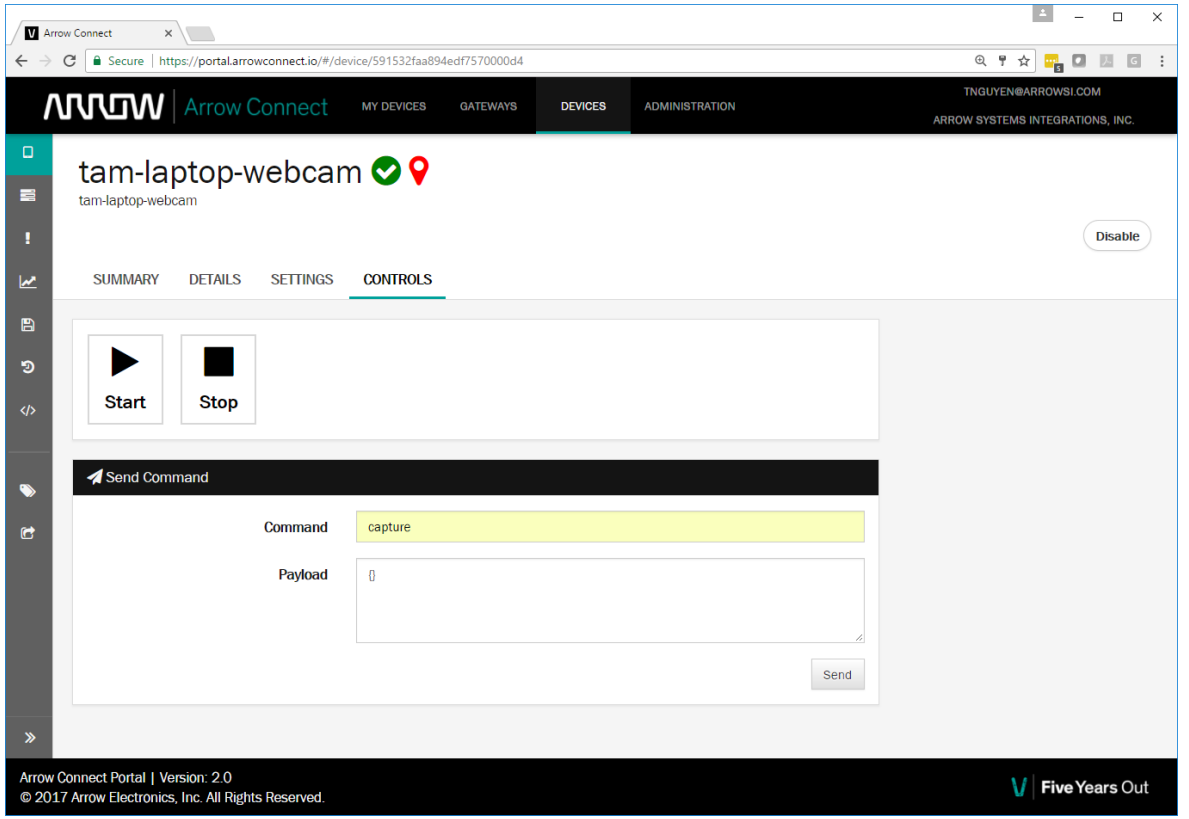
- Download and extract to c:\acn
  https://content.arrowconnect.io/public/training/05-2017/selene-windows.zip
- Modify file c:\acn\selene\config\devices\self.properties
  - `name = <your-gateway-name>`
  - `uid = <your-gateway-uid> (if omitted, default uid is gateway-<mac-address>)`
  - `apiKey = <your-application-api-key>`
  - `secretKey = <your-application-secret-key>`
- Modify file c:\acn\selene\config\devices\camera.properties
  - `name = <your-camera-name>`
  - `uid = <your-camera-uid>`
  - `enabled = true`
- Start Selene – open a windows terminal
  - `cd c:\acn\selene`
  - `bin\start.bat`
- Stop Selene
  - `Press Control-C`

Camera Device Configuration

index=0	This setting supports multiple cameras
width=640	Image width
height=480	Image height
frequencyInSecs=10	Automatically taking picture at this interval, set to -1 to turn it off
saveLocal=true	Save the image locally in addition to sending to the cloud
directory=c:/acn/selene/images	Local directory for images
detectFace=true	Run algorithm to detect face
markDetectedFace=true	If set to true, a box will be drawn around the detected face
faceDetectionFile=	Points to face detection algorithm file
faceDetectionScaleFactor=1.1	Face detection settings
faceDetectionMinNeighbors=3	Face detection settings
faceDetectionFlags=4	Face detection settings
faceDetectionMinSizeX=30	Face detection settings
faceDetectionMinSizeY=30	Face detection settings
overlayEnabled=true	If set to true, overlay the image with some text
overlayUrl=http://iot.arrow.com	Overlay text on the image
ipCamera=false	IP or USB camera
streamUrl=http://	Required when ipCamera is true

Camera Device Command

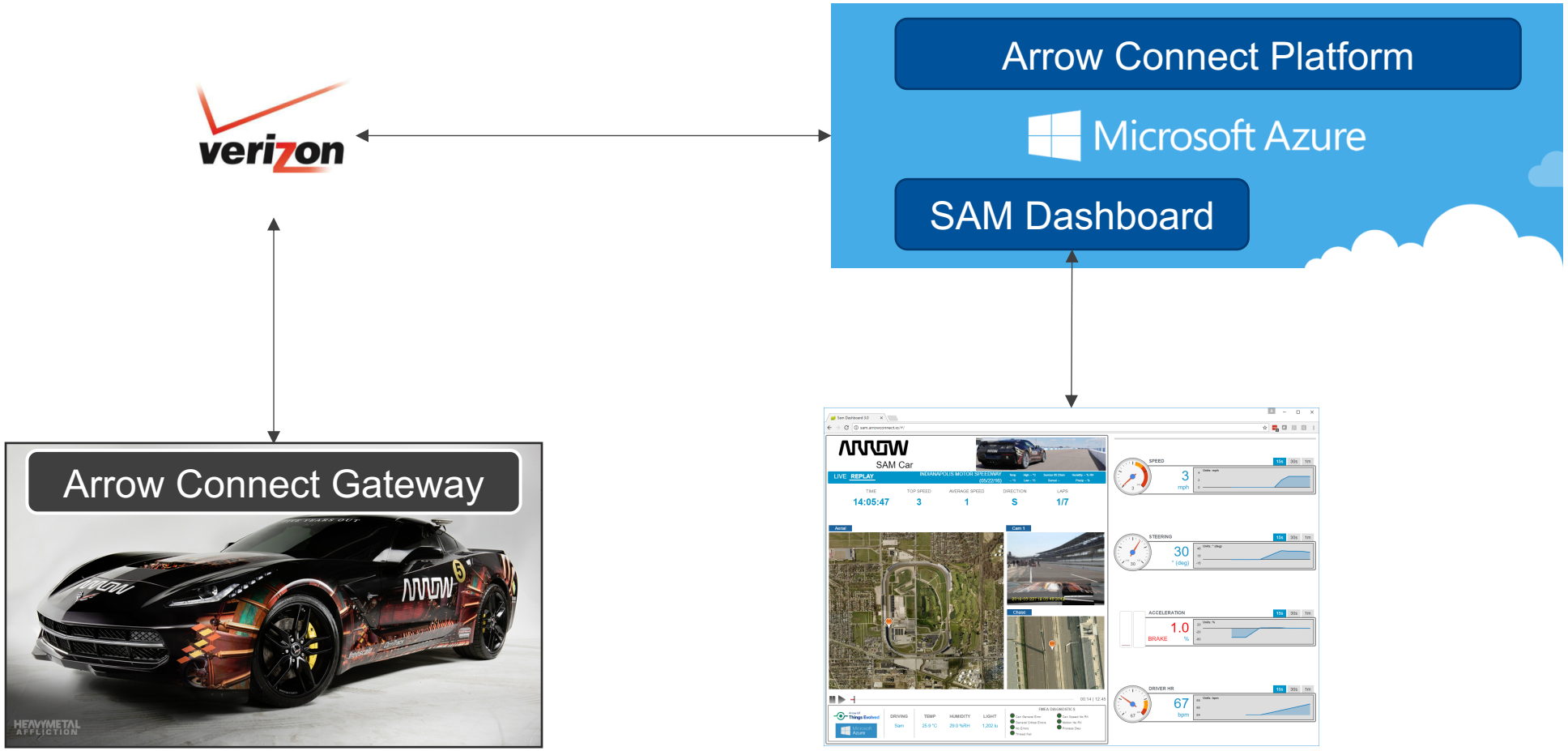
- To send command to trigger image capture – enter command `capture` and an empty payload `{}`



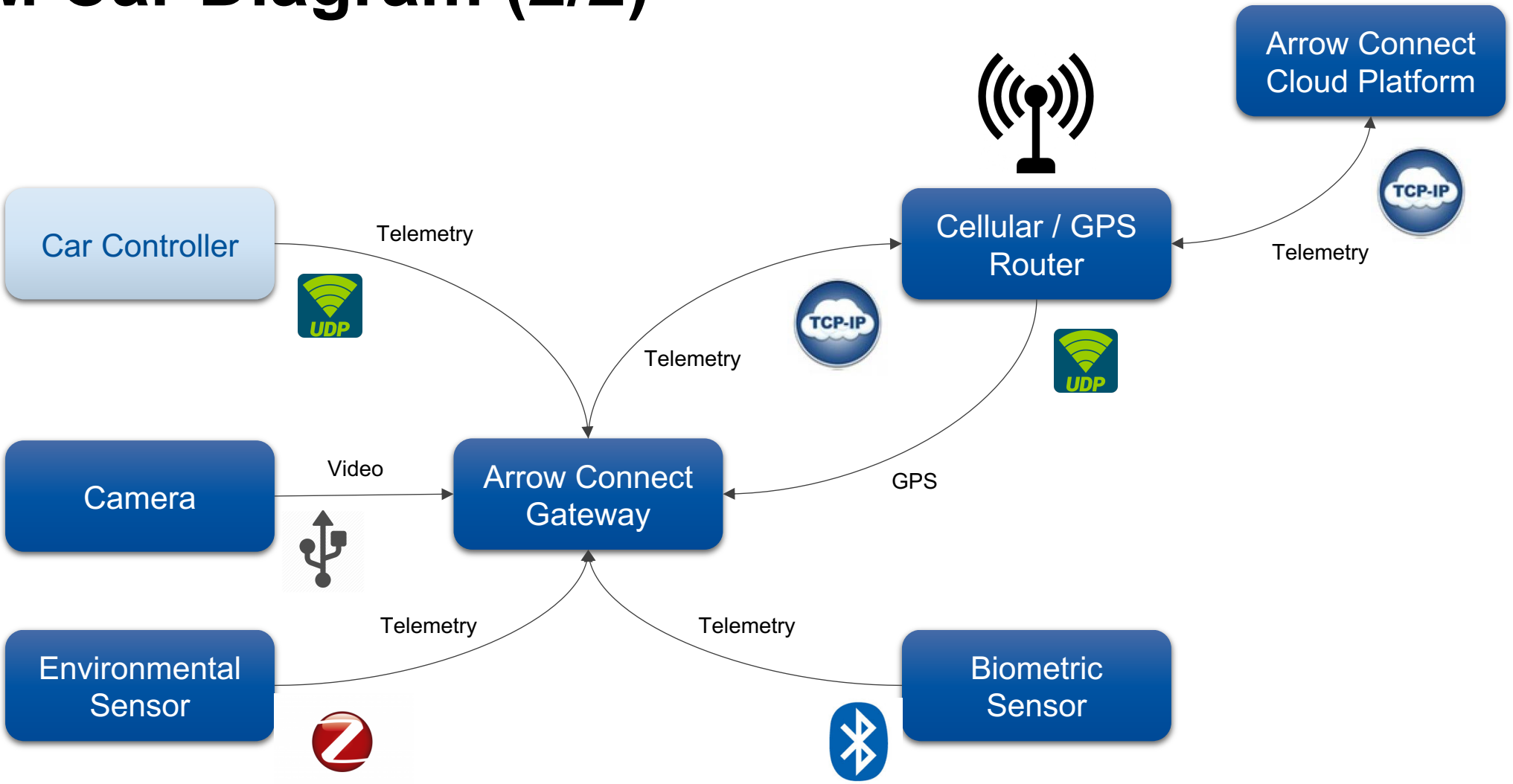
SAM Car Story

- Sam Schmidt is a former Indy Car Racing Driver who was diagnosed as a quadriplegic after an accident 15 years ago
- Arrow designed a Semi-Autonomous Motocar (SAM) for Sam to drive using his head to control the steering, acceleration, and braking
- In August 2015, the IoT team was tasked to IoT-Enable SAM car within one month, which also included developing a customized dashboard for live demonstration
- Arrow Connect Cloud Platform was still in its early stage at the time
- SAM car gateway software is version 0.1 of Selene today
- Story - http://community.arrow.com/sam
- SAM IoT Dashboard – http://sam.arrowconnect.io

SAM Car Diagram (1/2)



SAM Car Diagram (2/2)



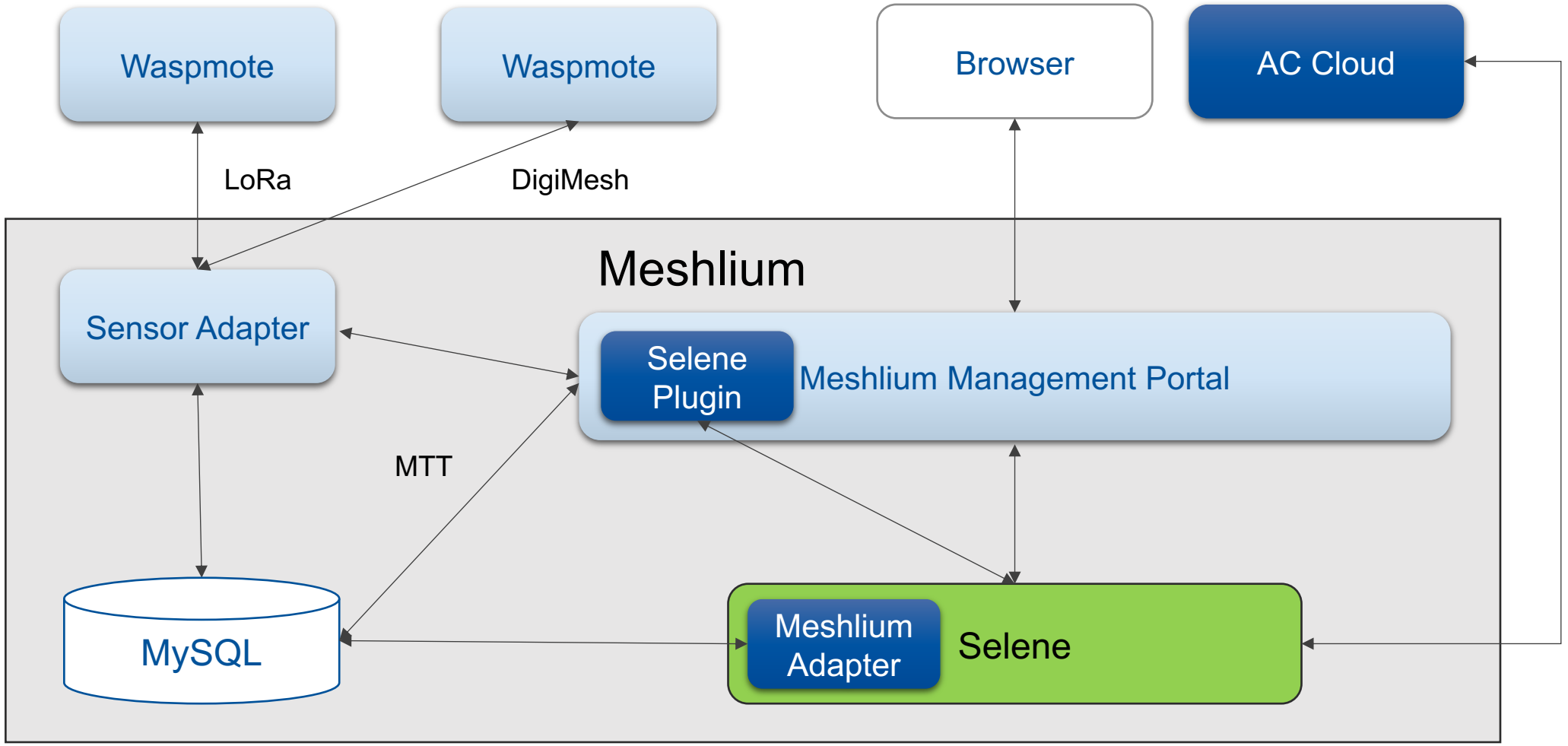
SAM Car Hardware

- ❑ Cellular Router – Digi Transport WR44 R
- ❑ Gateway – Advantech ARK-2121V
- ❑ Camera – Logitech HD Pro C920
- ❑ Environmental Sensors – XBee Sensors
- ❑ Biometric Sensor - Scosche Rhythm+ Armband

Libelium Meshlium and Waspmote

- Libelium Meshlium is fully certified for Arrow Connect Gateway. Our software is bundled with the Meshlium Gateway out-of-the-box http://www.libelium.com/arrow-electronics-extends-iot-offering-with-libelium-agreement/
- Waspmote development with Arrow Connect C-SDK is slated for Q3/Q4. For his scenario, the Waspmote device has Wi-Fi or Cellular capability and can connect to Arrow Connect Cloud directly instead of going through the Meshlium via some data protocol such as LoRa or Zigbee

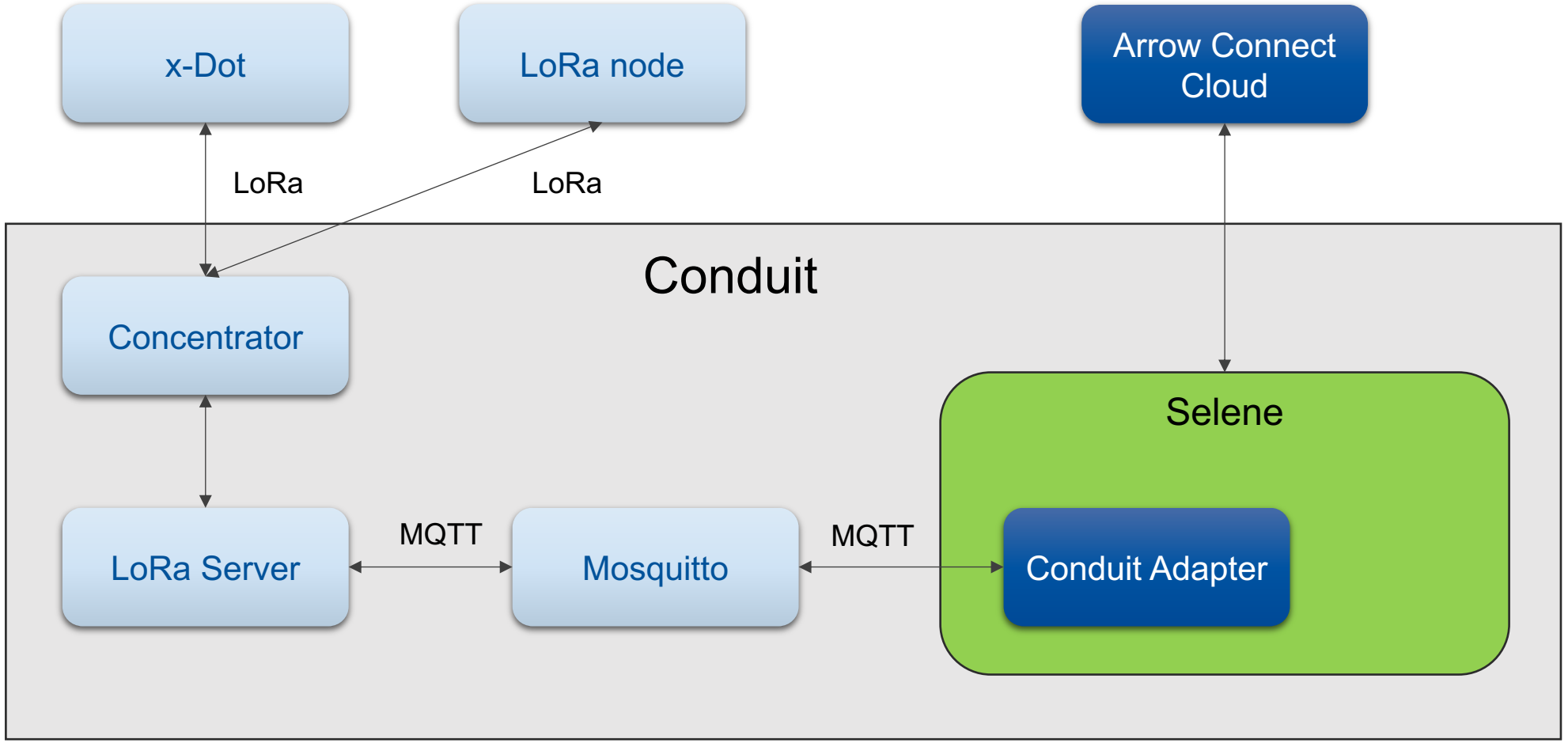
Meshilum – Selene Integration



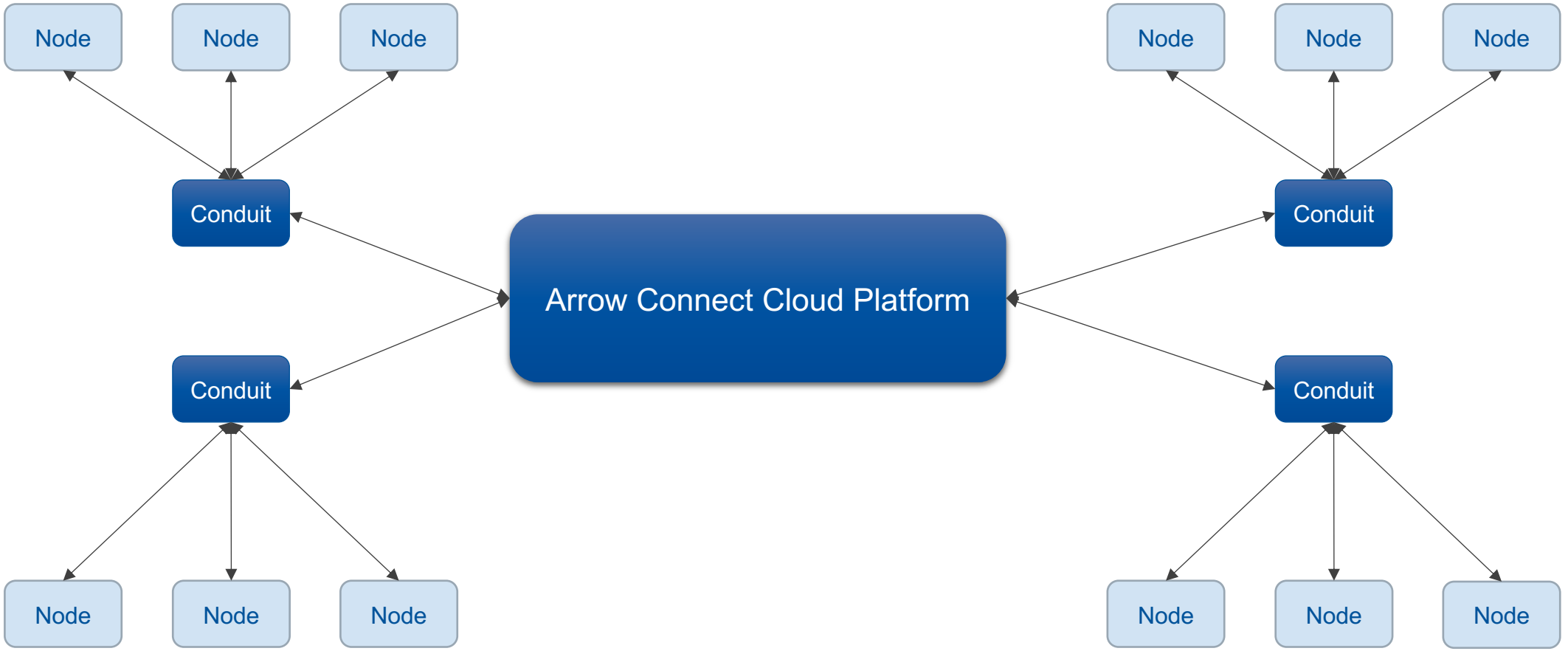
Multitech Conduit – LoRa and LoRaWAN

- Certification of Arrow Connect Gateway on Multitech Conduit and LoRa is currently in progress. ETA is July 2017. Implementation uses Selene's generic MQTT device adapter to communicate with the local gateway software stack
- Arrow Connect already supports Multitech Conduit with LoRaWAN implementation. The solution leverages LORIOT.io as a middleware LoRaWAN server. We developed a back-end integration that integrates LORIOT.io into Arrow Connect Cloud Platform

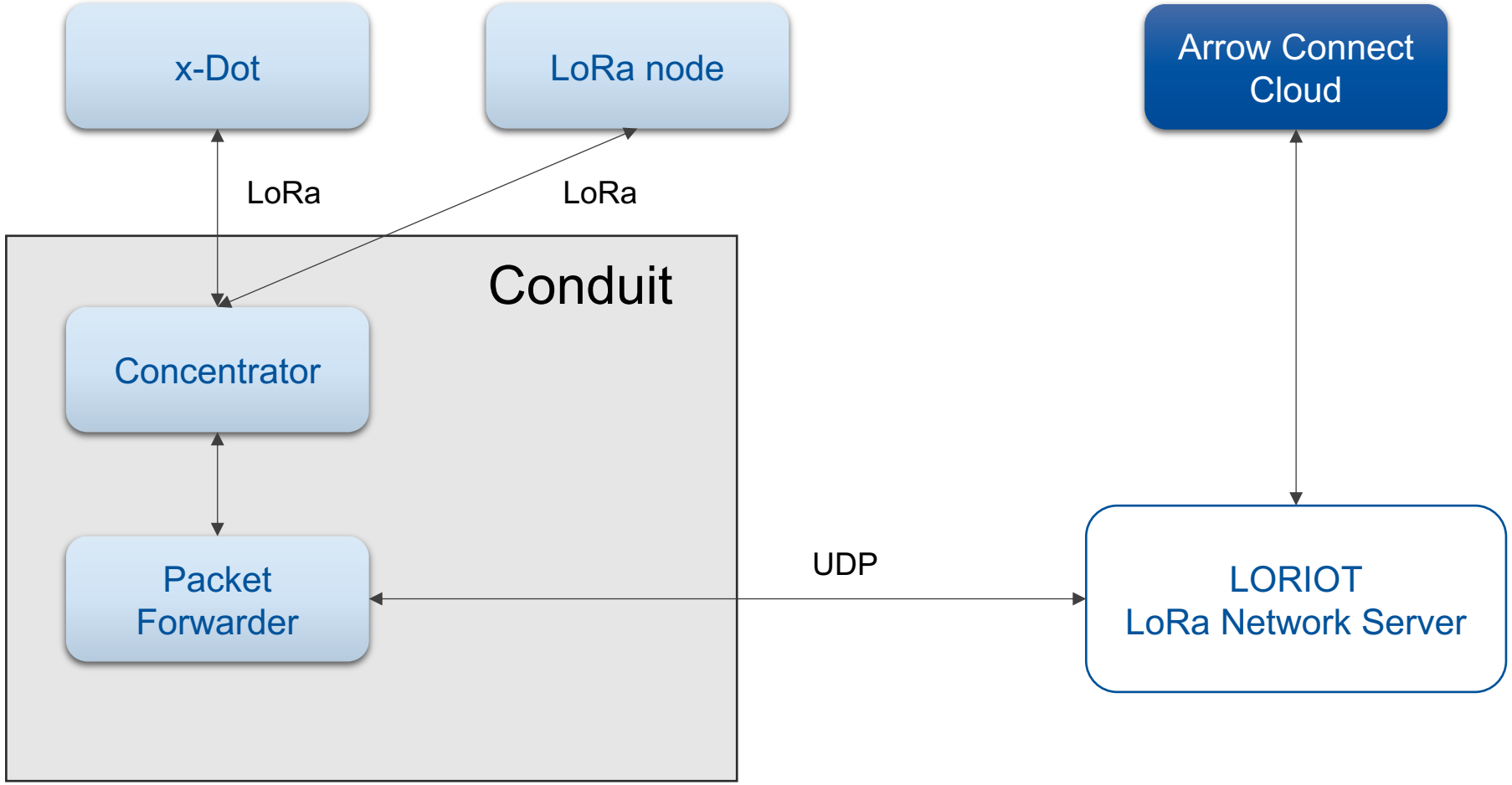
Conduit – Selene Integration (LoRa) (1/2)



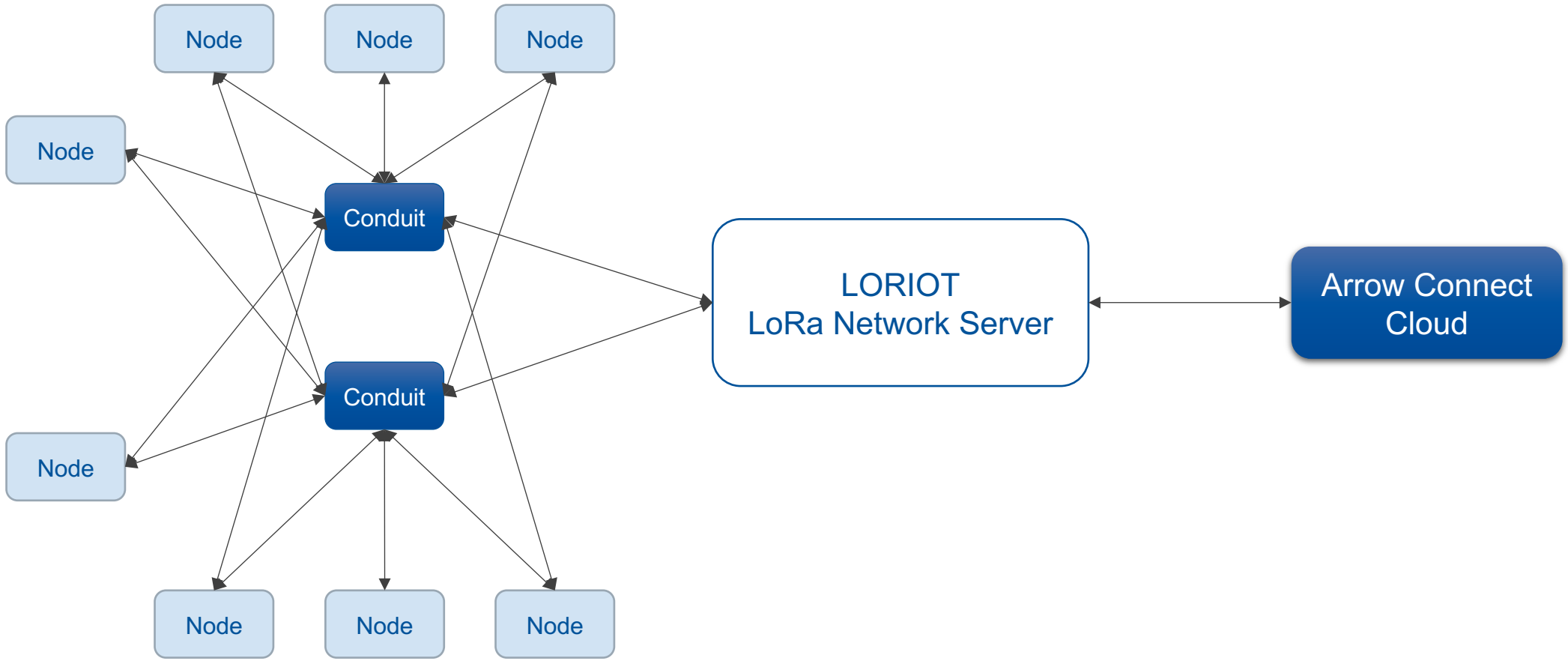
Conduit – Selene Integration (LoRa) (2/2)



Conduit – Arrow Connect (LoRaWAN) (1/2)



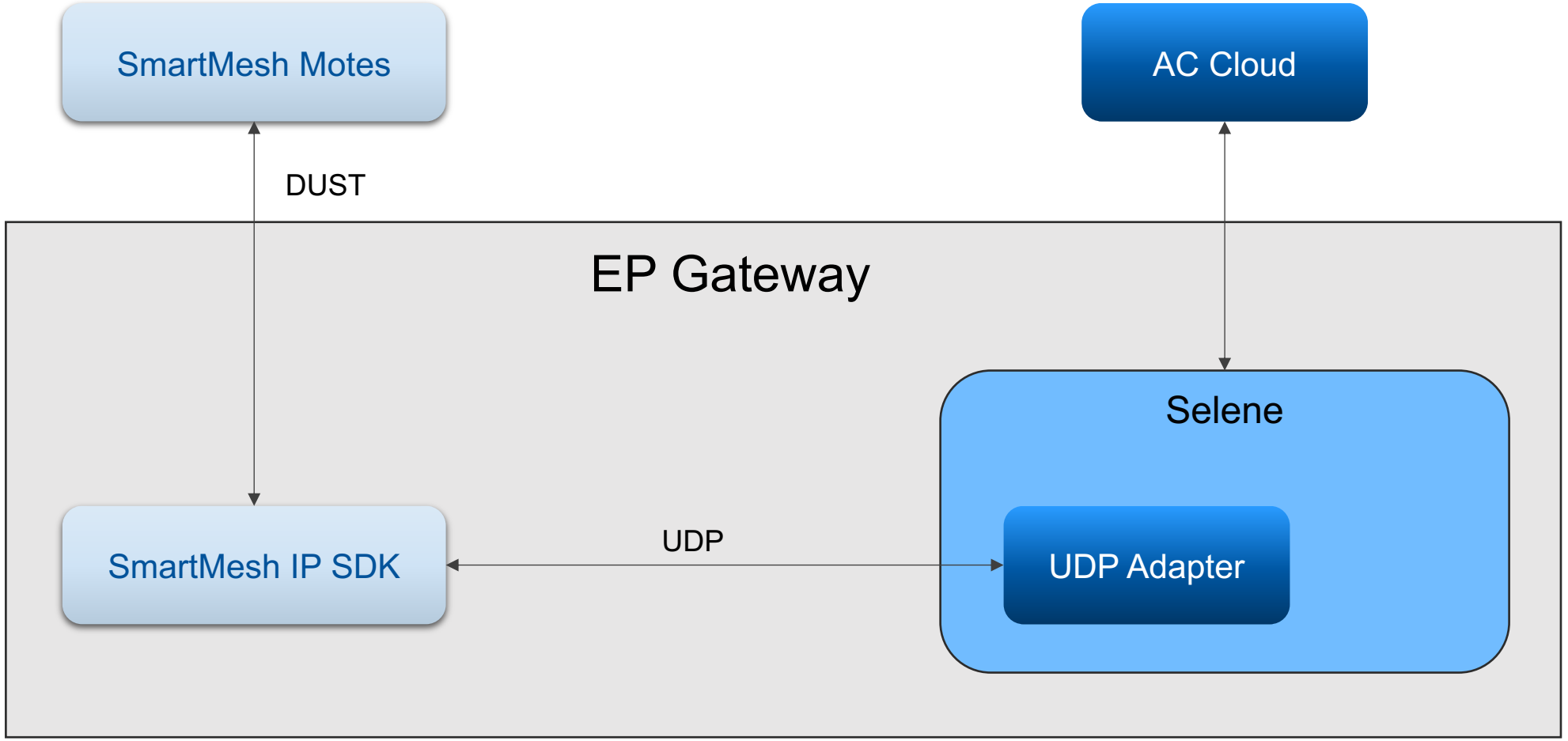
Conduit – Arrow Connect (LoRaWAN) (2/2)



Embedded Planet – SmartMesh IP

- Embedded Planet Gateway with SmartMesh-IP sensor nodes are fully certified with Arrow Connect Gateway
- Implementation uses Selene's generic UDP device adapter to communicate with the local gateway software stack

SmartMesh IP – Selene Integration



Zigbee – Selene Integration (1/2)

- Support both Zigbee Mesh and DigiMesh
  - IEEE 802.15.4 Radio
  - Reliable, Interoperable and Low Power
  - 3 types of nodes – coordinator, router, and end devices
  - DigiMesh is Digi's proprietary mesh network
- Use Digi XBee module running as coordinator via UART
  - Silicon Labs EM35xx SoC with EmberZNET PRO stack
  - Default firmware is running DigiMesh
  - Ability to run in bypass mode to support Zigbee Mesh
- Support Zigbee Home Automation 1.2
  - Custom Zigbee adapter to fully support HA 1.2 specification http://www.zigbee.org/zigbee-for-developers/applicationstandards/zigbeehomeautomation/
  - Rich API to interact with Zigbee network from the cloud

Zigbee – Selene Integration (2/2)