IoT Big Data Framework Architecture

Version 1.0

20 October 2016

This is a Non-binding Permanent Reference Document of the GSMA

Table of Contents

1 Introduction

1.1 Overview

This document has been developed to define a generalised architecture for delivery of "Internet of Things" "Big Data" services to support an ecosystem of third party application developers.

The architecture has been developed to enable many industry participants to work together. Mobile operators are seen as key participants in the delivery of an IoT Big Data ecosystem, though it is expected much of the IoT data that is collected will come from a range of data provider partners.

This document concentrates on defining a framework for delivery of IoT Big Data services and it is recognised in practice there will be many different approaches towards the services that are offered and the technology choices that are made. The proposed architecture offers a degree of flexibility which allows IoT Big Data services to be offered in more than one way.

Together with the accompanying documents “IoT Big Data Harmonised Entity Definitions” [1] and “IoT Big Data NGSIv2 Profile” [2], this document aims to define a framework of how mobile operators can approach the delivery of IoT Big Data services.

1.2 Scope

This document specifies a generalised architectural framework for the delivery of Big Data services based on the Internet of Things. It identifies the key functions and interfaces that enable IoT Big Data services to be delivered, and makes selections and recommendations particularly in the area of interfaces that support the creation of the IoT Big Data ecosystem. The framework outlines a logical architecture and it should be noted that operators may make different implementation decisions. In addition, not all mobile operators will implement exactly the same IoT Big Data services and this framework provides flexibility for them to approach the market according to their own strategy.

1.3 Definitions

Term	Description
LTE-M	LTE-M refers to LTE Category M, an evolution of the LTE standard and chipsets optimised for IoT applications. The “M” initially stood for “machines.”
Context Data	Contextual data is data that gives context to a person, entity or event. Examples of context data might include geographic/ mapping information, weather forecasts, schedules e.g. for transportation, or information generated from mobile networks/ users.
EC-GSM-IoT	Extended coverage GSM IoT (EC-GSM-IoT) is a standard-based Low Power Wide Area technology. It is designed as a high capacity, long range, low energy and low complexity cellular system for IoT communications.

Term	Description
GSM	Global System for Mobile communication[1] (GSM), is a standard developed by the European Telecommunications Standards Institute to describe the protocols for second-generation digital cellular networks used by mobile phones, first deployed in Finland in July 1991.
Internet of Things	The Internet of Things (IoT) describes the coordination of multiple machines, devices and appliances connected to the Internet through multiple networks. These devices include everyday objects such as tablets and consumer electronics, and other machines such as vehicles, monitors and sensors equipped with communication capabilities that allow them to send and receive data.
JSON	JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999
Long-Term Evolution	Long-Term Evolution (LTE) is a standard for high-speed wireless communication for mobile phones and data terminals. It is based on the GSM/EDGE and UMTS/HSPA network technologies, increasing the capacity and speed using a different radio interface together with core network improvements.
LTE-M	LTE for Machine to Machine (M2M) applications.
MQTT	MQTT (formerly MQ Telemetry Transport) is an ISO standard (ISO/IEC PRF 20922) publish-subscribe-based "lightweight" messaging protocol for use on top of the TCP/IP protocol. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited
NB-IoT	Narrow-Band IoT (NB-IoT) is a narrowband radio technology specially designed for the Internet of Things (IoT), hence its name. Special focus of this standard are on indoor coverage, low cost, long battery life and large number of devices. This technology can be deployed in GSM and LTE spectrum.

1.4 Abbreviations

Term	Description
API	Application Program Interface
CoAP	Constrained Application Protocol
FTP	File Transfer Protocol
GPU	Graphics Processing Unit
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
JSON	JavaScript Object Notation
LTE	Long-Term Evolution
MNO	Mobile Network Operator
NGSI	Next Generation Services Interface
RTSP	Real Time Streaming Protocol
SSD	Solid State Drive

[1] Originally Groupe Speciale Mobile

Term	Description
XML	Extensible Markup Language

1.5 References

Ref	Doc Number	Title
[1]	PRD CLP.26	IoT Big Data Harmonised Entity Definitions
[2]	PRD CLP.24	IoT Big Data NGSIv2 Profile
[3]	oneM2M	oneM2M - Standards for M2M and the Internet of Things - http://www.onem2m.org/
[4]	Hypercat	Hypercat is a consortium and standard driving secure and interoperable Internet of Things (IoT) for Industry - www.hypercat.io/
[5]	FIWARE NGSIv2	FIWARE-NGSIv2 Specification available at http://fiware.github.io/specifications/ngsiv2/stable/
[6]	MapReduce Tutorial	https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html

1.6 Conventions

"The key words "must", "must not", "required", "shall", "shall not", "should", "should not", "recommended", "may", and "optional" in this document are to be interpreted as described in RFC2119 **Error! Reference source not found.**."

2 IoT Big Data

Explosive growth is expected within the area of the "Internet of Things" (IoT) i.e. Internet connected devices, appliances, systems and sensors. Where once the Internet was confined to computing appliances this has transitioned through mobile devices including phones, smartphones and tablets to the point that ordinary "things" such as cars, TVs, fridges and thermostats are now "Internet Enabled".

Internet connectivity will continue to expand into further areas and as it does there is the opportunity to better understand the relationships between disparate data and derive new insights and inform decision making. Key to this is the application of Big Data technologies and techniques to the massive amounts of data that will be generated from this mass of Internet connected things.

2.1 What is "Big Data"?

"Big Data" is often defined in terms of **"3V's"** i.e.

- **Volume** - the amount of data generated, stored and analysed. The amount of data stored determines the level of insight that can be obtained from that data;
- **Variety** - type and nature of data. Historically data was structured and from a single source - in which case it would fit readily into 'columns' and 'rows'. Increasingly data is sourced from a variety of sources with many different formats;

- **Velocity** - the speed at which data is generated and processed. Where historically data could reasonably be expected to be uploaded via a daily 'batch' process now data is measured in thousands or even millions of transactions per minute.

In addition, other **"V's"** may be added including:

- **Variability** - Variations in the data sets. For example is a temperature measured in degrees Celsius, Fahrenheit or Kelvin;
- **Veracity** - Quality of the captured data. Where decisions are being made on data you need to be sure that the data is correct.

"Big Data" can be broadly defined as being associated with:

- Data sets that are so large or complex that traditional data processing applications/ methods are inadequate;
- Challenges include analysis, capture, data curation, search, sharing, storage, transfer, visualisation, querying, updating and information privacy;
- Use of predictive analytics or other advanced methods to extract value from data;
- Any size of data set (though likely to be large);
- Use in decision making for new applications, greater operational efficiency, cost reduction and reduced risk.

In the context of the "Internet of Things" this will allow new applications to be developed which mine either the real-time data coming from the many Internet connected devices and/or recorded historical data and insights.

2.2 What roles are mobile operators expected to adopt for IoT Big Data?

It is anticipated that mobile operators will choose to work in different ways to address the IoT Big Data ecosystem. This may include one or more of the following:

Connectivity Provider - where the mobile operators will provide the Internet connectivity using technologies such as (but not limited to) conventional 2G/3G/4G (e.g. LTE) data, emerging IoT technologies such as NB-IoT / LTE-M / EC-GSM-IoT;

- **Service Provider** - where the mobile operators will provide a managed IoT service directly to consumers or businesses;
- **IoT Platform Operator** - where the mobile operators will provide a multi-sided platform allowing IoT data from multiple devices/ sources to be collected and made available to third parties in a controlled manner along with other related data ("context data") that allows further insights to be drawn from the IoT data. The platform shall also provide a control channel through which third party applications are able to control IoT devices;
- **IoT Big Data Cloud Provider** - where the mobile operators provides its own "private cloud" for the storage of IoT and related data along with a Big Data processing software "stack" which allows the running of analytics and processing of intelligence on the IoT data. Such a "private cloud" service can then be used by the MNO for its

own services or otherwise made available to third parties for use in external applications and services;

- **Analytics/ Intelligence Provider** - where the mobile operators provides a bespoke service to customers in the development of analytics and intelligence gained from the IoT data;
- **Application Provider** - where the mobile operators itself will be developing one or more applications that leverage the data from IoT devices.

In general, it is seen that the further the operator moves into the higher complexity areas, such as IoT Big Data Cloud Provider and Analytics/ Intelligence Provider, the greater will be the commercial opportunity. This requires investments into Big Data systems and expertise so not every mobile operator interested in supporting the IoT Big Data ecosystem may take this approach.

2.3 Challenges for IoT Big Data

Some of the key challenges for IoT Big Data, which have a bearing on the design of architectures suitable for service delivery include

1. **The number of IoT devices**: With forecasted growth in the number of connected "things" expected into the billions world-wide there will be masses of devices which may be a data source, and which may be subject to third party control;
2. **The variety of IoT device**s: There will be enormous variety in the devices which may provide data, even in the case of similar devices e.g. an electronic thermostat. Data from any individual device manufacturer or model may be quite dissimilar from that of nominally identical devices in such areas as field names, units, and data structures;
3. **Intelligence of IoT devices:** IoT devices have more and more compute resources and integrate several technologies like Graphics Processing Unit (GPU) and Solid State Drive (SSD) storage. Simple sensors are evolving to autonomous systems which will be able to manage their own analytics and be part of large analytics networks;
4. **Risk of IoT device malfunction:** With a great number of IoT devices and manufacturers it is reasonable to assume there will be many occasions where IoT devices malfunction in various ways. In the most drastic situations devices will fail completely but it should be expected that more subtle malfunctions will occur which might result in aberrations of data coming from those devices, or a failure of the device to perform a required control function;
5. **Update frequency**: Though some devices (e.g. remote sensors) will produce data reports at a low frequency there may be substantial quantities of data streaming from more sophisticated Internet connected things such as cars;
6. **Historical data**: It is expected that many Big Data insights will derive from historical data recorded from IoT devices. This may be processed alone to derive analytics/ intelligence or considered alongside current data particularly to enable smart monitoring and control;
7. **Context data**: Much IoT data will make more sense when put in context with other data. Context data might be generally "static" (or at least with a slow update period) such as geographical data, or could be more dynamic e.g. weather forecast data. Another

important source of context data can be information gathered from the mobile networks themselves e.g. mobile user location or usage dynamics;

8. **Privacy issues:** With so many expected IoT devices acquiring data there could be a substantial risk relating to the disclosure of data which is considered personal to end users. When IoT data is stored in a Big Data system and made available to third parties there is a need to implement strong safeguards to ensure end users remain in control of their personal information. Mobile Network Operators (MNOs) are in a strong position to help users remain in control of their data and to make data available in the best way via consent, aggregation or anonymisation.

3 Key enablers for IoT Big Data services

The following enablers are identified for the delivery of Big Data services on top of IoT.

1. **Harmonised Entity Definitions**: A commonly agreed formal entity definition that models real world entities and addresses the variation in manufacturers and models of IoT devices and sensors. Using a Harmonised Entity Definition for the publishing of harmonised data entities and support for controlling devices will make it easier for third party applications to deliver IoT services and analytics/ intelligence to be drawn more easily and reliably from IoT data;
2. **Developer Application Program Interface (API) Access Management:** Access management enabling harmonised data entities to be queried or delivered as IoT device data/ Context Data or a "As Near Real Time"[2] stream to third party applications, services and systems. This enables basic querying of short to medium term historical data and provides support for the control of IoT devices or subsystems. It provides access control to the APIs used by application developers, and implements authentication and authorisation using industry standards to ensure harmonised data is exposed only to authorised applications and according to consent provided by users;
3. **Data and Control Broker:** Responsible for publishing harmonised data entities through a query and subscribe API, allowing applications to gain access to such data in a standard way. It may store data in the short to medium term, coming from multiple devices and data sources via the Data and Protocol Mediator.
4. **Data and Protocol Mediator:** Responsible for ingesting data from IoT devices as well as other external sources ("context data"). It ensures that data is transformed to the Harmonised Entity Definition before being passed to the Data & Control Broker. It may also translate protocols between the external world and a standardised protocol (NGSIv2) [5], which is used by the Data and Control Broker. This layer also transforms control requests coming from the Data and Control Broker to the lower level IoT control protocols;

[2] "Near Real Time" is expected to be within a few seconds of a device/ sensor recording the data rather than the sub-second Real Time that might be used within the device itself for control purposes

5. **IoT Big Data Store:** Typically based on off-the-shelf computing hardware and relevant database technologies this allows a massive collection of IoT data to be stored and made available for processing for analytics/ intelligence purposes
6. **IoT Big Data Processin**g: Also running on off-the-shelf computing hardware the analytics/ intelligence functions process the IoT data held within the Big Data Store to produce the more sophisticated analytics/ intelligence output.

4 General Architecture for IoT Big Data

The diagram below shows the general architecture for delivery of IoT Big Data services. This is explained in the following narrative.

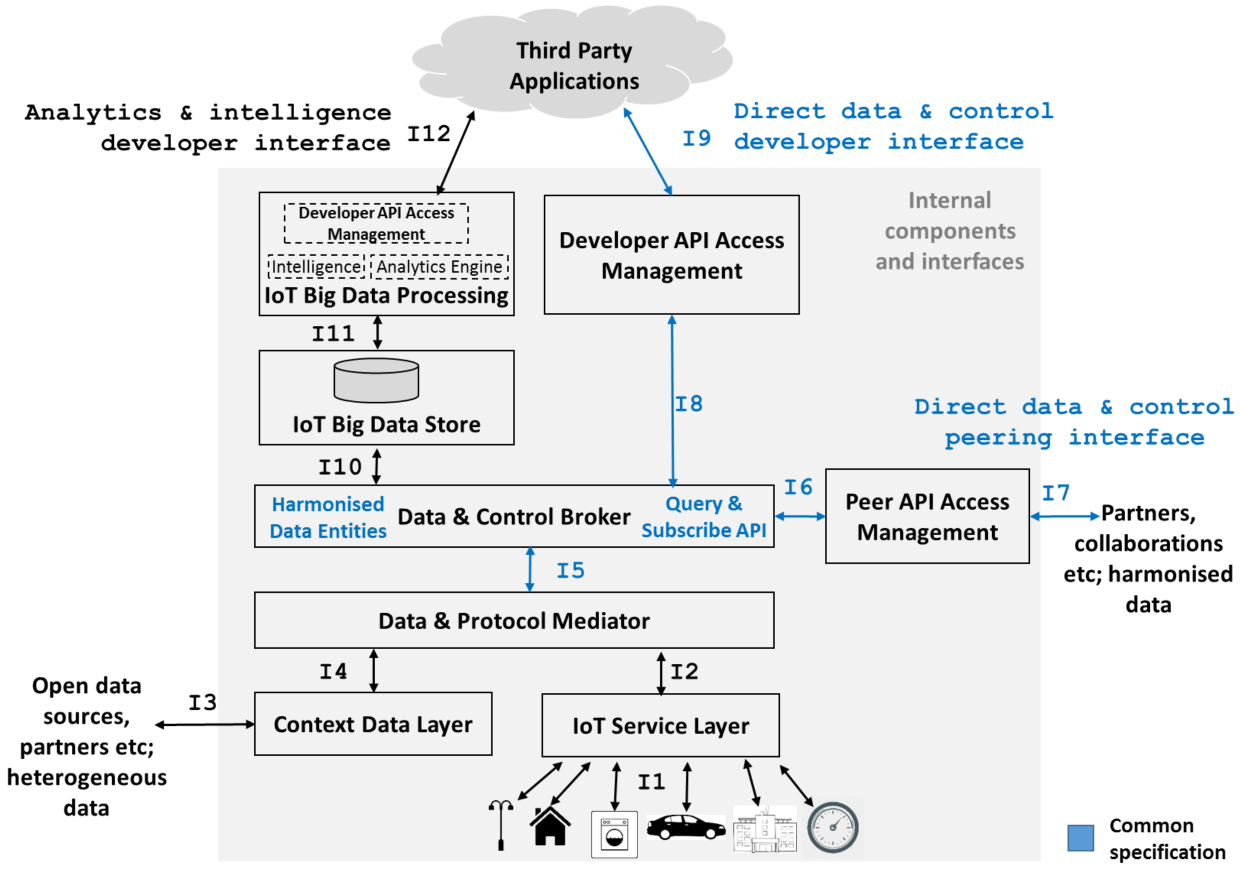
Third Party Applications
Analytics & intelligence developer interface
I12
I9
Direct data & control developer interface
Developer API Access Management
Intelligence
Analytics Engine
IoT Big Data Processing
Developer API Access Management
Internal components and interfaces
I11
IoT Big Data Store
I8
Direct data & control peering interface
I10
Harmonised Data Entities
Data & Control Broker
Query & Subscribe API
I6
Peer API Access Management
I7
Partners, collaborations etc; harmonised data
I5
Data & Protocol Mediator
I4
I2
Open data sources, partners etc; heterogeneous data
I3
Context Data Layer
IoT Service Layer
I1
🟦 Common specification

Figure 1: General Architecture for IoT Big Data

4.1 Functional Units

A number of functional units are identified in the IoT Big Data architecture as shown in the above diagram. Together these support the delivery of a wide range of IoT / Big Data services.

These functional units do not need to be implemented by all organisations delivering IoT Big Data services, for example:

- Those operators who want to concentrate on delivering analytics and intelligence services do not have to support the exposure of harmonised data or control over the ‘Direct data & control’ developer or peering interface, so the related parts of the architecture can be omitted;
- Those operators who only want to provide the delivery of IoT / context data to third party applications can do this by implementing the Developer API Access Management function rather than implementing the IoT Big Data Store and IoT Big Data Processing functions;
- The ingestion of external data through the Context Data Layer interface is similarly optional (though it is considered likely this is required for many use cases);
- The peering connection with other IoT Big Data platforms can be utilised when organisations wish to deliver analytics or intelligence services across a consolidated set of harmonised data (where some of this is supplied by other organisations for example in Smart Cities)

It should also be noted that the division above is purely for the sake of describing the functionality in terms of logically related services. This does not necessarily have a bearing on how the overall IoT Big Data service is delivered using any physical or cloud based platforms. It is also noted that the identified logical functions may be combined in implementations or other logical separations made.

4.1.1 Context Data Layer

This functional unit is concerned with obtaining external non IoT data ("Context data") which is either available to the third party application or used during the processing of IoT data e.g. "mashing up" IoT data with context data. The Context Data Layer is also able to communicate with the external data sources, e.g. to start and stop data feeds.

Examples of context data might include geographic/ mapping information, weather forecasts, schedules e.g. for transportation, or information generated from mobile networks/ users. This allows IoT data to be associated with further context data e.g. a moisture sensor reports the current moisture level whilst a weather forecast for the same geographical area identifies whether rain is predicted - allowing a decision to be made as to whether to water a crop.

Context data might be received in different ways e.g. via Hypertext Transfer Protocol (HTTP) based APIs which request data from external servers, information received within an email, via batch file by making an outgoing File Transfer Protocol (FTP) request or by a batch file being deposited via FTP, or data received using removable media. This unit is principally concerned with implementing the relevant adapters in order to receive the various types of context data.

4.1.2 IoT Service Layer

The IoT service layer is concerned with handling the device specific interactions required for obtaining data from IoT devices and sending control commands (where relevant) to those IoT devices. Therefore this layer is required to handle bi-directional communications both to IoT devices and to the upper layers of the architecture.

The IoT Service Layer is expected to handle the lower level interactions with IoT devices. Those devices might be connected using a variety of protocols and low level communication technologies including (but not limited to) oneM2M [3], Hypercat [4], Constrained Application Protocol (CoAP), MQ Telemetry Transport (MQTT), Real Time Streaming Protocol (RTSP), or device specific interfaces such as JavaScript Object Notation (JSON)/Extensible Markup Language (XML) over HTTP.

The IoT Service Layer is expected to handle authentication and security aspects regarding the interfacing with IoT devices.

4.1.3 Data and Protocol Mediator

The Data and Protocol Mediator is responsible for ingesting information from IoT devices as well as other external sources ("context data"). It ensures that data is transformed to the Harmonised Entity Definition before being stored and published by the Data & Control Broker.

The harmonisation process itself may be partially implemented in the 'Context Data Layer' function or the 'IoT Service Layer' function but the Data & Protocol Mediator will ensure that harmonisation is complete before data is exposed to the higher level parts of the architecture.

The harmonisation process includes:

- Conversion of payload encoding e.g. converting between an XML format payload of the IoT or context data and the JSON based structures defined in the Harmonised Entity Definitions;
- Mapping of the data structures and data fields between the lower level IoT device structures and fields and the Harmonised Entity Definitions e.g. the IoT device might store a temperature value in a field named 'temp' whereas the Harmonised Entity Definition in a field named 'currentTemperature';
- Unit conversion from the values and ranges of the lower level IoT devices to the Harmonised Entity Definition e.g.
    - The Harmonised Entity Definition might represent a switch as the Boolean true or false value whereas the IoT device could represent as the integer 1 or 0 respectively;
    - The Harmonised Entity Definition might represent a temperature in degrees Centigrade as a double precision float whereas the IoT device might record in degrees Fahrenheit.
- Data quality verification e.g. identifying a situation where an IoT sensor is apparently faulty such as delivering a sensor reading which is outside of the expected range. For example an outdoor temperature sensor transmitting a value which is significantly outside of the normal weather temperature range;

- Combining ("mash up") or linking relevant context data with IoT data e.g. associating a specific IoT sensor with its geographic location or weather forecast data to form a richer entity definition;
- Cross referencing related entity data e.g. different sensors with say the car to which they belong.

The Data & Protocol Mediator will also enable control requests to be processed - performing broadly a 'reverse' process compared with data harmonisation:

- Verifying the control request to make sure that the request is valid e.g.
    - Refers to a valid IoT device;
    - The control action is relevant to that IoT device e.g. a fixed position sensor cannot be asked to move to a different position;
    - The control action is valid according to the current state of the device/ system (which should be maintained by the control broker);
    - The parameter values supplied in the request are valid both in terms of individual parameter range and in combination.
- Transforming the high level control request into the equivalent device specific request payload e.g. generating an XML format payload if that is required by the IoT device;
- Mapping of the data structures and data fields in the control request between the high level structures and fields of the Harmonised Entity Definitions and the lower level IoT device structures and fields;
- Unit conversion from the values and ranges of the Harmonised Entity Definitions to the lower level IoT device parameters e.g.
    - Converting a switch state in the harmonised data entities from a Boolean true or false value to an IoT device state representation having an integer value of 1 or 0 respectively;
    - Converting a target temperature in the harmonised data entities from a request in degrees centigrade (as a double precision value) to an integer value in degrees Fahrenheit required by the IoT device;

It should be noted that the Data & Protocol Mediator may be implemented as a set of standalone “Data & Protocol Mediator Gateways” that each perform specific data transformation and protocol translation functions, e.g. for external data services or for IOT devices. The Data & Protocol Mediator gateways would communicate with the data sources/ devices on the southbound side and the Data & Control Broker on the northbound side. There are many options for the interface protocols to be used on these interfaces but NGSIv2 would fit well on the southbound side of the Data & Control Broker as the Data & Control Broker already supports NGSIv2. The southbound side of these Data & Protocol Mediator Gateways could also use oneM2M or even proprietary protocols.

4.1.4 Data & Control Broker

The Data & Control Broker is responsible for enabling third party application access to harmonised data entities through a query and subscribe API, allowing applications to gain access to such data in a standard way. The broker may store data in the short to medium term, coming from multiple devices and data sources via the Data and Protocol Mediator. This function also transforms control requests coming from the application layer to be passed onwards to the Data & Protocol Mediator.

The control process itself may be partially implemented in the 'IoT Service Layer' function but the Data & Control Broker in collaboration with the Data & Protocol Mediator will ensure responsibility for providing third party application access to control services in a consistent (harmonised) and controlled way. Control brokering will perform broadly a 'reverse' process compared with data harmonisation, receiving high level control requests from the third party application - normally formatted as a JSON based request communicated over HTTPS, and adapting this through the Data & Protocol Mediator and IoT Service Layer.

The Data & Control Broker is expected to have access to a data store which may act as a short to medium term buffer space for control actions or a short to medium term store for harmonised data entities. The expected use of this is:

- Retention of current instances of harmonised data entities processed from IoT devices and external sources (context data);
- Storage of control requests and any status information relevant to the request;
- Storage of a window of historical harmonised data entities that may be queried directly via the third party application. Note that it is expected that such a data store would be for short to medium term harmonised data entities, whereas longer term storage of harmonised data entities would be provided in the "IoT Big Data Store”;
- Storage of any results of Analytics and Intelligence results which become additional context data that can be queried or mashed up with other IoT data or external data sources.

It should be noted that the Data & Control Broker has the option of using its own internal database for data storage or the defined IoT Big Data Store function defined in this architecture i.e. some of the logically separate elements of the defined architecture may be physically implemented together.

It is assumed that this function does not have to be concerned with any issues around which application is consuming the service, or implementing any policies e.g. access/ throttling/ data anonymisation. Such concerns would instead be handled by the Developer API Access Management function detailed later.

4.1.5 Peer API Access Management

The Peer API Access Management function is responsible for interfacing with its peers in other organisations to receive and publish additional relevant harmonised IoT and context data. The policies applied to these trusted interfaces may be different to those applied to the main developer interface provided by the Developer API Access Management function. For example, organisations may be willing to share certain sensitive data with each other but require this sensitive data to be anonymised before being offered to third party developers. See sections 4.2.6 and 4.2.7 on I6 and I7 interfaces for more details.

4.1.6 Developer API Access Management

The Developer API Access Management function controls access to harmonised data entities, covering both IoT and context data, as well as control services to third party applications. It implements authentication, authorisation and access control using industry standards to ensure privacy and security around the harmonised data.

This function is mainly concerned with managing the privacy and security aspects of the data access by external parties. It is expected that this layer does not perform any actual IoT and context data processing or storage but is ensuring that data and control services from lower layers of the architecture are delivered in a properly managed way. It is assumed that any data processing/ complex queries/ analytics/ intelligence is the responsibility of the third party application.

The Developer API Access Management function access control for the harmonised data, it is expected to perform the following:

- Be responsible for presenting data & control services to third party applications via a RESTful based API over http[3]. This interface shall use JSON based encoding of data using the Harmonised Entity Definitions for both data and control and use the NGSIv2 interface to support entity retrieval/ simple queries;
- Implement API access control (i.e. application level security) to ensure only known/ approved applications have access to IoT and context data and control services. Access control should be provided on a per application basis allowing granularity over which application should be able to access what IoT and context data and control functions;
- Implement any usage policies against applications accessing the APIs e.g. applying IP address based access rules or throttling rules to ensure there is relevant fair usage of the platform;
- Apply user consent rules to IoT device and context data and for IoT device control as appropriate to the device. The obtaining of consent is outside of the scope of this document but it is assumed consent is granted to an application in the form of an OAuth2 access token which is then presented in API calls as an HTTP authorization bearer token;
- Perform anonymisation of data to ensure compliance with data protection legislation including the prevention of third party applications working together to correlate data about users e.g.

[3] Actually a secure (https) connection connected over TLS

    - By obscuring personal data belonging to individuals (e.g. mobile number, current location) except where there is specific consent for the release of such data;
    - By making certain information less precise e.g. transforming an accurate geographical location into a more general area;
    - By aggregating a number of related data and subjecting this to a minimum sample size to avoid inadvertently disclosing personal information;

- Provide access to a publish/ subscribe service so that IoT and context data can be pushed to the third party application server as new data is received;
- Log API usage information e.g. number of API calls made by an application, number and type of entity data retrieved, number and type of control requests received.

Usage information may be interrogated for use in analytics and potentially future service charging.

It should be noted that the description of the logical functions of the Developer API Access Management does not preclude any particular implementation. There are many API management related functions listed here and they may be implemented via a number of separate components.

4.1.7 IoT Big Data Store

The provision of Big Data Analytics and Intelligence is dependent on having access to the relevant mass of data from which the various insights can be obtained. This function provides data storage for this massive data and it may also provide short to medium term storage capabilities for use by the Data & Control Broker, depending on the specific implementation.

In a small scale or demonstration deployment, storage is a fairly simple problem of storing a database on a hard disk. As the data set expands, multiple disks are required and the number of processors required to process this data grow quickly. For IoT Big Data usage it is considered that the Data Store must be able to handle a data set greater than 50TB in size.

For small scale deployments/ prototypes a Relational Database such as MySQL may support IoT data storage. However realistically a NoSQL or 'graph' database is considered more suitable for commercial 'Big Data' deployment particularly because the graph data model is richer and more versatile.

"Big Data" databases address needs such as:

- The need to store vast amounts of data (orders of magnitude higher than Relational Databases reasonably work to);
- Insights are obtained when exploring ad-hoc relationships between data;
- Data is arriving at such a rate that it is impossible to maintain an indexing process;
- Data are not tightly constrained into fixed table/ column formats.

The "Big Data" database is expected to be used to store the harmonised data entities received from the IoT devices and/or the external data sources. As it is expected there could be many millions of IoT devices generating data frequently, the required storage space may be vast (i.e. of the order of many terabytes to many petabytes of data). It is expected the "Big Data" database could be implemented using products such as Apache Cassandra, Apache Hadoop, MongoDB, Neo4j, Titan or DynamoDB. To achieve high performance the database component may employ substantial quantities of memory to hold copies of data that is persistently stored on "hard disk".[4]

4.1.7.1 Cassandra

Cassandra[5] is a scalable database for large scale data storage from the Apache foundation and is used by many of the world's leading tech companies including github, Netflix, Apple and Instagram. The largest known deployment of Cassandra contains 75000 nodes (cloud servers) and stores over 10PB (Petabytes) of data.

Cassandra is a NoSQL data store, which provides a robust means of storing data which spans many nodes, however it does not provide a very powerful query interface; it's highly inefficient to query on anything other than Cassandra's equivalent of a 'primary key'. Several solutions can be combined with Cassandra to provide a more powerful query interface. Apache Spark is one of the most powerful of these.

4.1.7.2 Hadoop

Apache Hadoop[6] is a highly scalable storage platform designed to process very large data sets across hundreds to thousands of computing nodes that operate in parallel. It provides a very cost effective storage solution for large data volumes with no particular format requirements. MapReduce [6] is the programming paradigm that allows for this massive scalability, is at the heart of Hadoop. The term MapReduce refers to two separate and distinct tasks that Hadoop programs perform. Hadoop has two main components - HDFS and YARN.

HDFS[7] – the Hadoop Distributed File System is a distributed file system designed to run on commodity hardware. It differs from other distributed file systems in that HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

YARN[8] - YARN is a large-scale, distributed operating system for big data applications that runs on top of HDFS. It provides a framework for job scheduling and cluster resource management.

[4] SSD may also be used in a hybrid arrangement alongside high capacity magnetic storage hard disks
[5] http://cassandra.apache.org/
[6] http://hadoop.apache.org/
[7] https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
[8] https://hadoop.apache.org/docs/r2.7.2/hadoop-yarn/hadoop-yarn-site/YARN.html

4.1.7.3 MongoDB

MongoDB[9] is a hybrid open source and closed source database, where the core of the database is available freely on an open source license, although some features which may be required on larger commercial deployments are commercially supported add-ons. This model has made MongoDB arguably one of the most popular document oriented databases in use today.

A 'document' in MongoDB is a 'binary' representation of a JSON document. This allows arbitrary JSON encoded data to be stored in the database and then queried using a rich JSON based querying interface.

4.1.7.4 Graph Databases

Other databases such as Neo4J[10] or Titan[11] are a powerful way for structuring data which allows for easily traversing relationships as well as retrieving attributes about a particular node. It is worth clarifying that a Graph Database works efficiently where there are ad-hoc relationships between data whereas a Relational Database is efficient for more structured relationships between data.

Neo4J is a hybrid open source/commercial offering much like MongoDB, though widely regarded as the leading graph database.

Titan is an open source graph database which can be backed by several different back-ends - most notably Cassandra. It also has integration with Apache Lucene[12] to provide full text and geographic searching over hundreds of billions of triples.

Graph databases underpin graph query languages such as SPARQL[13] (as suggested in oneM2M) and Gremlin[14]. The key strength of these systems is that they're very well adapted for traversing different data types to perform ad-hoc mash-ups.

4.1.8 IoT Big Data Processing

The processing of stored IoT data to perform analytics and intelligence is identified as the responsibility of the IoT Big Data Processing function.

The IoT Big Data Processing function also provides related Developer API Access Management to control access to the intelligence and analytics by implementing authentication, authorisation and access control to ensure privacy and security.

A broad division is made between analytics and intelligence. In practice both analytics and intelligence will be processing subsets of the mass of IoT data retained in the IoT Big Data Store. The main difference is

[9] https://www.mongodb.com/
[10] https://neo4j.com/
[11] http://titan.thinkaurelius.com/
[12] https://lucene.apache.org/
[13] https://www.w3.org/TR/rdf-sparql-query/
[14] https://github.com/tinkerpop/gremlin/wiki

- Analytics - principally involves relatively conventional methods (by human analysts and normal programming techniques) of exploring links and statistical relationships between data and then the analytics engine will produce its output based on the execution of a defined process;
- Intelligence - encompassing machine learning / artificial intelligence, it would be expected that algorithms 'adapt' to the observed data and the match between predicted and desired outcomes.

The outputs from Analytics and Intelligence are expected to be in a wide range of different formats, many of which will not conform to a uniform 'API' based approach e.g. the generation of a PDF report or the generation of a data set to be FTP'd to the third party developer's platform.

Relevant products for Analytics & Intelligence provision include:

- **Apache Spark**

  Apache Spark[15] is a powerful data processing system based upon Cassandra or Hadoop[16] for the data storage component and provides several powerful tools for building applications around it such as an SQL interface, graph data library and a job server.

  Spark is not a complete solution out of the box, but does provide a powerful big data platform with great performance. Spark is considered the leading solution for high performance Big Data analytics.

  A Spark solution could be delivered over a RESTful interface or a websockets connection (for better notification and real time services). More usually however developers would use the standard programming interfaces available to Java, Python, Scala and R programming languages.

- **Apache TinkerPop3 + Titan + Elastic Search + Gremlin**

  Titan provides Casandra backends, integration with Elastic search[17], Apache Lucene[18] / Solr[19], Spark and others which allows it to support Geo searches, full text searches, graph traversals and regular 'SQLesque' queries making it ideal for the IoT Big Data project.

  Apache TinkerPop3[20] is a graph computing framework which is seeing a rapid adoption in data driven applications. Many projects are seeking to incorporate the TinkerPop specification into their interfaces for interoperability in graph databases

[15] http://spark.apache.org/
[16] http://hadoop.apache.org/
[17] https://www.elastic.co/
[18] https://lucene.apache.org/
[19] http://lucene.apache.org/solr/
[20] http://tinkerpop.apache.org/

and servers. There are several implementations of graph databases which expose a Gremlin[21] querying interface which makes it easier to query the graph database. Two such databases are Titan and Google Cayley.

- **Apache Mahout**

  Mahout[22] is designed for the development of high performance and scalable machine learning applications. It builds for example on top of Apache Spark / Hadoop and supports a range of machine learning algorithms. Uses include

  - Collaborative filtering – mines user behaviour and makes product recommendations (e.g. Amazon recommendations);
  - Clustering – takes items in a particular class (such as web pages or newspaper articles) and organizes them into naturally occurring groups, such that items belonging to the same group are similar to each other;
  - Classification – learns from existing categorizations and then assigns unclassified items to the best category;
  - Frequent itemset mining – analyses items in a group (e.g. items in a shopping cart or terms in a query session) and then identifies which items typically appear together.

  Apache Mahout can be used to make recommendations which are 'learned' from a data set - such as the readings of IoT sensors.

  As a simple example Mahout may be applied to a personal safety type application for the elderly where motion sensors around the home and/or on the person could detect anomalous behaviour e.g. a fall in the bathroom allowing an alert to be sent to a family member or the emergency services.

- **Tensorflow**

  Another open source set of tools for machine learning - developed originally by the Google Brain Team to support advances in search ranking algorithms as well as other Google research activities.

  Tensorflow[23] could be used for example in IoT applications such as developing high accuracy automated number plate recognition algorithms based on images captured from CCTV cameras. This can then be applied in the IoT Big Data system to applications such as security, congestion or traffic planning.

  Tensorflow can also be coupled with Apache Spark which is used to obtain the select the data from the IoT Big Data store to use with the tensorflow algorithms.

[21] Gremlin is maintained as part of Apache Tinkerpop (http://tinkerpop.incubator.apache.org)
[22] http://mahout.apache.org/
[23] https://www.tensorflow.org/

4.2 Interfaces

The following interfaces are identified in the IoT Big Data architecture.

4.2.1 Interface 1 (I1) - Providing IoT device connectivity between IoT devices and IoT Service Layer

This is expected to be a relatively low level interface which is generally IoT device specific. There are expected to be multiple technologies and protocols which are used simultaneously on this interface i.e. this will be a set of heterogeneous implementations rather than there being a single homogeneous standardised interface.

Standards such as oneM2M, MQTT and CoAP are expected to be adopted increasingly for this interface in the future. Devices may also expose proprietary interfaces such as XML/ JSON based interfaces over an http/ https connection.

Devices may be connected using a variety of technologies including (and not limited to) Wi-Fi, Bluetooth and Cellular technologies such as LTE-M/ NB-IoT/EC-GSM.

4.2.1.1 CoAP

CoAP[24] is similar to HTTP in many respects, though highly optimised for lower power devices. The unique feature of CoAP is that unlike HTTP it is not a strictly client-server model and clients are free to push/pull documents in an ad-hoc fashion.

4.2.1.2 MQTT

A publisher/subscriber model of communication which allows messaging between devices who subscribe to 'topics'. Underpins Facebook messenger as well as the Amazon IoT service. MQTT[25] allows for one-to-many messaging or one-to-one messaging on a massive scale, including various levels of quality of service (QoS). QoS allows a sender to ask for confirmation of message delivery.

Amazon IoT is an example of a platform which exposes an MQTT interface for IoT devices at scale.

4.2.1.3 AMQP

Another publisher/subscriber model protocol. Used as a queuing protocol for communicating with databases such as Reddis or RabitMQ as well as communication between IoT devices and the cloud. AMQP[26] is the preferred (though not only) protocol used for Microsoft Azure's IoT hub service. AMQP is predominantly a client-server model of communication, though links between devices can be proxied through a server to allow clients to communicate.

[24] http://coap.technology/
[25] http://mqtt.org/
[26] https://www.amqp.org/

4.2.1.4 HTTP

HTTP is still relevant for client-server models of IoT deployment when integrating with older devices. The main advantage of HTTP is that it has one of the broadest adoptions of any protocol and has a great deal of developer knowledge.

Often HTTP will be used over SSL (or more usually TLS) for higher connection level security to guard against Man In The Middle attacks.

4.2.1.5 oneM2M

oneM2M[27] is a specification generally aimed at harmonising the access to IoT device data via a common set of service functions and communications protocols. oneM2M has several implementations available for the first release of the specification.

4.2.2 Interface 2 (I2) – Allowing data and control to be transferred between the IoT Service Layer and the Data & Protocol Mediator

A uniform technology approach is recommended for this interface to provide an abstracted method for transmitting data and control between layers for example using a simple JSON based RESTful API over https.

There are two proposed alternatives for this interface

1. Use of the FIWARE NGSIv2 protocol provides a developer friendly API design between the layers
2. Use of the oneM2M protocol (JSON over HTTP)

It is recommended that individual operators should adopt one or other of FIWARE NGSIv2 or oneM2M. As this interface is a purely 'internal' interface there is no issue of compatibility between providers and therefore it is for individual operators to make their own choice of FIWARE NGSIv2, or oneM2M, or any other equivalent protocol.

4.2.3 Interface 3 (I3) – Connecting to external sources to receive context data

This interface supports the interaction between the Context Data Layer and external parties. It is envisaged that context data will be supplied in many different ways and with data encoded in a wide variety of formats.

Various kinds of context data can be envisaged such as

- Information gathered from the mobile network e.g. mobile location information (GPS and/or cell id); network/ cell site traffic loading etc.;
- Mapping information e.g. supplied from one or more GIS (Geographic Information System) files;
- Weather (forecast) information which might be obtained using an online (web) service;

[27] http://www.onem2m.org/

- Postcode/ zipcode area files;
- Information from third party suppliers e.g. taxi companies providing information on the availability of cars available for hire.

It is expected that a variety of data delivery options would be supported e.g.

- Both push and pull based RESTful / web services over http based connections;
- Fetching of data (by the IoT Big Data platforms) using FTP or SSH based FTP (SFTP);
- Receipt of data (initiated by the external system) into a 'dropbox' (which might be implemented using FTP);
- Receipt of data via email;
- System administrator loading data directly from files on removable storage.

The data itself might be supplied using a wide range of encoding formats e.g.

- XML encoded data;
- JSON encoded data;
- Excel sheet;
- CSV (Comma Separated Values);
- Proprietary data formats (depending on source).

The data supplier may also implement an authentication mechanism such as

- HTTP authorization (username+password);
- FTP authorization (username+password);
- SSH public key based access;
- VPN based connection.

4.2.4 Interface 4 (I4) – Passing context data between the Context Data Layer and the Data & Protocol Mediator

This interface supports context data being passed between the Context Data Layer and the Data & Protocol Mediator so it can be converted to harmonised data entities and utilised by the higher levels of the architecture. Conversely, control instructions such as to start or stop the receipt of data, can also be passed through this interface.

It is expected this interface will normally be implemented using a uniform technology approach such as using a simple JSON based RESTful API over https.

FIWARE NGSIv2 may be adopted for this interface as it is also an option on the I2 interface.

4.2.5 Interface 5 (I5) – Passing harmonised IoT and context data from the Data & Protocol Mediator to the Data & Control Broker

This interface supports the passing of data between the Data & Protocol Mediator and the Data & Control broker. For data ingestion, harmonised data will be passed from the Data & Protocol Mediator to the Data & Control Broker so it can be published via query and subscribe APIs. For control scenarios, the data will be passed from the Data & Control Broker to the Data & Protocol Mediator so it can be translated and instructions passed back down to the relevant IoT Service Layer or Context Data Layer functions.

It is expected this interface will normally be implemented using a uniform technology approach such as using a simple JSON based RESTful API over https.

The IoT Big Data architecture recommends FIWARE NGSIv2 is adopted for this interface as this allows easy use and wide flexibility over the entity model data.

It should be noted that some organisations may implement the Data & Protocol Mediator and the Data & Control broker functions as a single component, in which case the I5 interface will be internal and could utilise any appropriate technological approach.

4.2.6 Interface 6 (I6) – Sharing data & control with peers over the Peer API Access Management function

This is the API interface to allow the sharing of the harmonised data entities between peer organisations. The is the raw API and requires access management to ensure that relevant data privacy and security requirements are met. Please see the next section on I7 Direct data & control peering interface.

The recommended design for this interface is:

- Harmonised Entity Definitions are used for the sharing of data and control;
- Personal data may be shared to a peer provider operating in the role of a 'Data Processor' subject to necessary system and operational process safeguards being in place when using that data;
- Data is shared encoded using JSON over an https connection and using the FIWARE NGSIv2 API for requesting entities/ subscribing to entity updates.

4.2.7 Interface 7 (I7) – Direct data & control peering interface

This interface is offered to peers for sharing harmonised data entities (IoT and/or context data). It is expected that peers are trusted partners and therefore more sensitive data may be shared over this interface compared to the northbound interface with third party developers.

Aside from the technical aspects of the API definition there will be a need in the respective platforms to track the source of data and the license terms under which data access is provided. This may affect the form in which data may be shared to other parties.

Though practically there are alternative approaches that could be adopted e.g. by connecting the northbound interface from the Developer API Access Management functional unit to the Context Data Layer or IoT Service Layer units of another provider, it is considered more efficient to standardise the Data & Control Peering interface as a better way to achieve the same benefits to the IoT Big Data ecosystem[28] for the following reasons:

- The policies implemented between IoT Big Data peers do not have to be the same as the policies offered towards third party developers. E.g. if one of the peer organisations is government the data being shared may be based on 'legal intercept' rather than end user consent;
- IoT Big Data peers can potentially share a superset of the data that would be offered to third party developers in the knowledge (and as supported by legal contract) that such information is used in a properly controlled manner. Reasonably this could be the interface between Data Acquirers/ Data Controllers and Data Processors;
- The Context Data Layer and IoT Service Layer are not designed for working with harmonised data entities and therefore ingesting harmonised data would add more complexity to these functional units than required at the Data & Context Broker layer;
- Harmonised data entities are exposed by the Data & Control Broker and as these are already defined as a common format are easy to share between multiple Data & Control Broker instances.

FIWARE NGSIv2 protocol/ API is recommended and considered to be the optimal interface due to the following reasons

- As a RESTful API it is easily consumed from third party applications over the Internet;
- The same interface design and API supports any number of (harmonised) entity types;
- It is easy to add application level authorization and/or user consent indication using HTTP based standards for authorization e.g. OAuth2;
- The API provides a consistent model for reading IoT device data, context data or 'mashed-up' data combining IoT device data with context data;
- There is support for controlling IoT devices by updating relevant fields in the IoT device;
- Simple filters can be set allowing the querying of IoT device data/ context data matching the required filter criteria;
- The API supports a 'publish/ subscribe' model allowing the third party application to register for updates to IoT device data/ context data;
- Filters can also be added to subscriptions to reduce the data sent to the third party application;
- There is inclusion of support for geographically based queries for entities and in subscriptions.

Harmonised Entity Definitions are used with the NGSIv2 interface for

- Sending IoT device data to the third party application;
- Sending context data to the third party application;

[28] i.e. offering harmonisation of IoT Big Data for analytics and intelligence purposes across multiple providers

- Sending mashed up IoT device data plus related context data to the third party application;
- Sending mashed up IoT device data plus related pre-processed intelligence/ analytics context data to the third party application[29];
- Receiving IoT device control updates from the third party application.

In the first release NGSIv2 does not support the ability to access historical data sets or 'linked data entities' using JSON-LD but FIWARE expect to add these capabilities into the NGSIv2 specification in the near future.

4.2.8 Interface 8 (I8) – Supporting the transfer of harmonised data & control from the Data & Control Broker to the Developer API Access Management function

This interface is primarily concerned with transferring the following

- IoT data in the form of harmonised data entities from the Data & Control Broker function to the Developer API Access Management function
- Context data also in the form of harmonised data entities from the Context Data Layer function to the Data & Control Broker function
- IoT device control requests from the Developer API Access Management function to the Data & Control Broker function

In all cases the principle payload will be in the form of harmonised data entities, and it is recommended that this interface adopts use of FIWARE NGSIv2 allowing the entities to be transferred using JSON encoding over an https connection.

4.2.9 Interface 9 (I9) – Direct data & control developer interface

This interface is offered to the third party application for reading harmonised data (IoT and/or context data) and supporting control services. This interface is specified as using the FIWARE NGSIv2 protocol/ API.

This interface is subject to all aspect of access control including the implementation of access policies, user management, API key management, throttling and API analytics. Industry standard solutions should be used where possible to maximise the appeal and ease of use of this API for applications developers.

This is considered to be the optimal interface due to the following reasons

- As a RESTful API it is easily consumed from third party applications over the Internet;
- The same interface design and API supports any number of (harmonised) entity types;
- It is easy to add application level authorization and/or user consent indication using HTTP based standards for authorization e.g. OAuth2;

[29] This is dependent on the implementation of analytics/ intelligence 'feedback' as shown on the I11 interface

- The API provides a consistent model for reading IoT device data, context data or 'mashed-up' data combining IoT device data with context data;
- There is support for controlling IoT devices by updating relevant fields in the IoT device;
- Simple filters can be set allowing the querying of IoT device data/ context data matching the required filter criteria;
- The API supports a 'publish/ subscribe' model allowing the third party application to register for updates to IoT device data/ context data;
- Filters can also be added to subscriptions to reduce the data sent to the third party application;
- There is inclusion of support for geographically based queries for entities and in subscriptions.

Harmonised Entity Definitions are used with the NGSIv2 protocol for this interface to enable:

- Sending IoT device data to the third party application;
- Sending context data to the third party application;
- Sending mashed up IoT device data plus related context data to the third party application;
- Sending mashed up IoT device data plus related pre-processed intelligence/ analytics context data to the third party application[30];
- Receiving IoT device control updates from the third party application.

In the first release NGSIv2 does not support the ability to access historical data sets or 'linked data entities' using JSON-LD but FIWARE expect to add these capabilities into the NGSIv2 specification during the second half of 2016.

4.2.10 Interface 10 (I10) – Supporting transfer of harmonised data to the IoT Big Data Store

The IoT Big Data Store is expected to be have a large body of data holding temporal IoT data which is held in a 'data lake' which would be based on NoSQL or graph type databases. In some implementations, this interface will also support the storage and access of short to medium term data for use by the Data & Control Broker.

The interfacing technology used to connect to these databases will to a large extent depend on the product choices made for the respective databases. Operators are expected to freely choose products/ suppliers based on individual technology or commercial preferences. Typically there will be a number of options for the interface technology related to database choices and though there could be 'standards' which can be used to access certain databases It is also likely that proprietary interfaces may need to be used for NoSQL/ graph database operations.

[30] This is dependent on the implementation of analytics/ intelligence 'feedback' as shown on the I11 interface

Therefore, no specific restrictions are proposed for this interface as it is expected to follow operator database platform choices. It is noted that operations on this interface are generally expected to be simple and support for storing newly received IoT / context data in the NoSQL/ graph database.

Note: It is expected that harmonised data entities would be stored in the "Big Data" NoSQL/ graph database so that they can then be queried in more complex ways from the higher level Big Data Processing function.

4.2.11 Interface 11 (I11) – Enabling the IoT Big Data Processing platforms to interact with the IoT Big Data Store

The IoT Big Data Processing function is expected to make complex queries to retrieve data from the "Big Data" database. Analytics and Intelligence may involve the processing of substantial amounts of data records and therefore this interface is expected to be implemented over high speed network technologies.

As described above the interfacing technology used to connect to the IoT Big Data database(s) will be dependent on the supplier choices made for the respective platforms. Operators are expected to freely choose products/ suppliers based on technology or commercial preferences.

The querying interface from the IoT Big Data Processing platform is expected to require the support of more complicated queries than the simpler interface used by the Data & Control Broker when storing data into the IoT Big Data store.

It is expected that one typical technology choice for "Big Data" will be the NoSQL database Apache Cassandra, with Apache Spark used for large-scale data processing. Apache Spark offers an SQL-like query language that can be invoked from various programming languages normally using a network based 'driver'.

It should also be highlighted that there can be a closed loop for analytics and intelligence where there is processing of data sets to form results that may in turn be stored into the IoT Big Data store and then become inputs to future analytics and intelligence processing.

The results generated as part of the IoT Big Data Analytics and Intelligence function may be used as inputs to harmonised data entities either:

- Forming a 'snap shot' of Analytics/ Intelligence which can be accessed in turn as its own kind of stand-alone harmonised data entity, or,
- 'Mashed-up' with IoT device data and/or other context data.

The Analytics and Intelligence feedback is defined to support closing the loop between the running of such Analytics and Intelligence and the making of 'snap shots' or 'historical' results available to third party applications via the Direct Data & Control API Management.

It is expected that such Analytics and Intelligence is stored as harmonised data entities and therefore the preferred implementation of this interface is the use of the FIWARE NGSIv2

API. It is recommended that the Analytics and Intelligence functions implement the harmonisation processes to minimise the amount of work required by the Data & Control Broker.

4.2.12 Interface 12 (I12) – Analytics & intelligence developer interface

This is the interface identified for delivery of Analytics (Analytics as a Service) and/or Intelligence (also as a service) derived from harmonised entity data stored in the IoT Big Data Store. It is expected that Analytics and Intelligence services will support for example:

- Creation of reports which are the result of a substantial data processing activity;
- Running of complex queries across an extremely large data set resulting in for example a rich statistical analysis of the data set;
- Correlating data against a geographical and time based backdrop to allow visualisation of behaviours e.g. energy usage across a city throughout the year;
- Machine based learning (intelligence) which identifies correlation patterns and predictions between many apparently unrelated data;
- Processing of a very large data set to identify 'exceptions';
- Real time machine learning approaches based on streaming IoT data whereby patterns are identified and decisions can be made using machine learning techniques.

Analytics and Intelligence may be delivered using a broad number of methods including:

- Interactively generating a result file e.g. report as a PDF, Word or Excel file using a web based front end to deliver the service to end users;
- Lengthy processing of data which will produce a visualisation (e.g. a video/ animation) which is available for direct downloading to removable media when complete;
- Support for transferring a generated result from an FTP area or to a third party FTP server when complete;
- Generation of a set of results which can be imported to other systems e.g. as CSV (Comma Separated Values), SQL scripts (for creating new databases), XML or JSON encoded data.

Although there may be many methods of delivering Analytics and Intelligence services the following are recommended in the case that services are provided via a 'RESTful' API:

- Data transfer should use JSON encoding;
- JSON-LD[31] should be used to support linked data, and in the case this refers to either current IoT device data or context data the linked data should be retrievable through its URL via the "Direct Data & Control Interface" above;
- GEO-JSON[32] should be used to support geographical data/ queries;

[31] http://json-ld.org
[32] http://geojson.org

- The interface must be secured with end to end encryption using TLS;
- In the case results contain identified harmonised data entities there should be use of the defined Harmonised Entity Definitions;
- Proof of user consent (for the release of personal data) must be provided using OAuth2 bearer tokens (as per OpenID Connect/ Mobile Connect);
- The Analytics / Intelligence services must ensure compliance with data protection requirements (e.g. where necessary by anonymising data or aggregating result sets or ensuring compliance with the requirements of data providers).

4.3 Support for monetisation of IoT/ Big Data

The IoT Big Data framework architecture is expected to support monetisation of the various IoT/ Big Data services through the following technical enablers

- The 'source' of each data item is an attribute of each Harmonised Entity Definition and should be recorded in the data stores of the IoT Big Data infrastructure. This enables an accounting system (which is external to the IoT Big Data infrastructure) to

    - Calculate any charges which should be paid by any organization which is storing its data within the IoT Big Data infrastructure. Charges might include periodic subscriptions and/ or data volume related charges;
    - Calculate any payments which should be paid to any organization which is providing related data e.g. context data to the IoT Big Data infrastructure. Payments might include periodic subscriptions and/ or data volume related payments.

- Each application which is consuming API services through the I9 or I12 interface will be subject to authentication procedures which identify the application and associated consuming organization. This identification is applied to security, data privacy, traffic management (e.g. throttling) as well as accounting purposes.

    - It is expected that each externally initiated request for a resource (be this a single data record, a query or call to the analytics/ intelligence) will be recorded in an 'access log' or 'accounting log' which is provided by the 'Developer API Access Management' function;
    - Again it is expected that an external accounting system (external to the IoT Big Data infrastructure) shall process such 'access logs' or 'accounting logs';
    - The external accounting system is then expected to calculate any charges which should be paid by any organization which is obtaining data, or using the query functions/ analytics/ intelligence services provided by the IoT Big Data infrastructure. Charges might include periodic subscriptions and/ or data volume related charges.

- Any partner organization which is obtaining harmonised data from the IoT Big Data system through the I7 interface will also be subject to authentication procedures which identify the partner organization. This identification is applied to the 'peering interface' to support security, data privacy, traffic management (e.g. throttling) as well as accounting purposes.
    - It is expected that any partner obtaining resources from the IoT Big Data infrastructure (be this a single data record, a query or call to the analytics/ intelligence) will have the relevant details of the transaction recorded in an 'access log' or 'accounting log' which is provided by the 'Peer API Access Management' function;
    - Again it is expected that an external accounting system (external to the IoT Big Data infrastructure) shall process such 'access logs' or 'accounting logs';
    - It may be the case that a partner organization functions as both a consumer and supplier of data/ services so the external accounting system would be expected to calculate any net charges/ payments which should be paid to or are due from the partner organisation. Charges might include periodic subscriptions and/ or net data volume related charges.

5 Identified Deployment Scenarios

The generic architecture outlined above can be applied in a number of ways to support different commercial models. The following are examples of the most common deployment scenarios which are anticipated though data providers are able to create alternative solutions using the defined building blocks.

5.1 Data Provider offering Direct Data/ Control Services Only

In this first scenario the IoT Big Data provider is providing simple exposure of Direct Data or Control for the IoT devices they are connecting or managing.

In this case the data provider will not need to build a Big Data storage or processing platform and will not need to build expertise supporting the development of Big Data analytics or intelligence services. Therefore this scenario requires the lowest investment to set up, it also cannot support the provision of higher value analytics or intelligence services. Therefore any monetisation opportunity in this case is considered to be exclusively through basic data and control exposure.

Third party application developers or peers/partners in this scenario will be required to provide any of the services on top of the IoT device data/ control services exposed by the data provider. The application will receive IoT device data conforming with the Harmonised Entity Definitions over the Direct Data & Control Developer Interface and will (subject to support/ permissions by the data provider) be able to issue control command requests to IoT devices. This will mean that the third party application developer is responsible for selecting, building and operating any Big Data infrastructure required to support higher order Big Data analytics and intelligence.

Unlike the next scenario where only analytics/ intelligence is provided to third parties, this architecture will allow the development (but by the third party application developer or peer organisations) of services across an aggregated data set from multiple data providers. This is made easier for the third party to develop through the conformance of IoT device data and control to the defined Harmonised Entity Definitions, though there may be complexities or obstacles for data providers and third party application developers related to addressing privacy and data protection requirements.

This scenario is considered easiest to set up and maintain as it does not require the investment in the more complex part of the IoT Big Data infrastructure or expertise relating to analytics/ intelligence. However, the monetisation opportunity is expected to be restricted to 'basic data provision' only and complexity is pushed onto the third party application developer limiting the range of developers who will work with such an architecture.

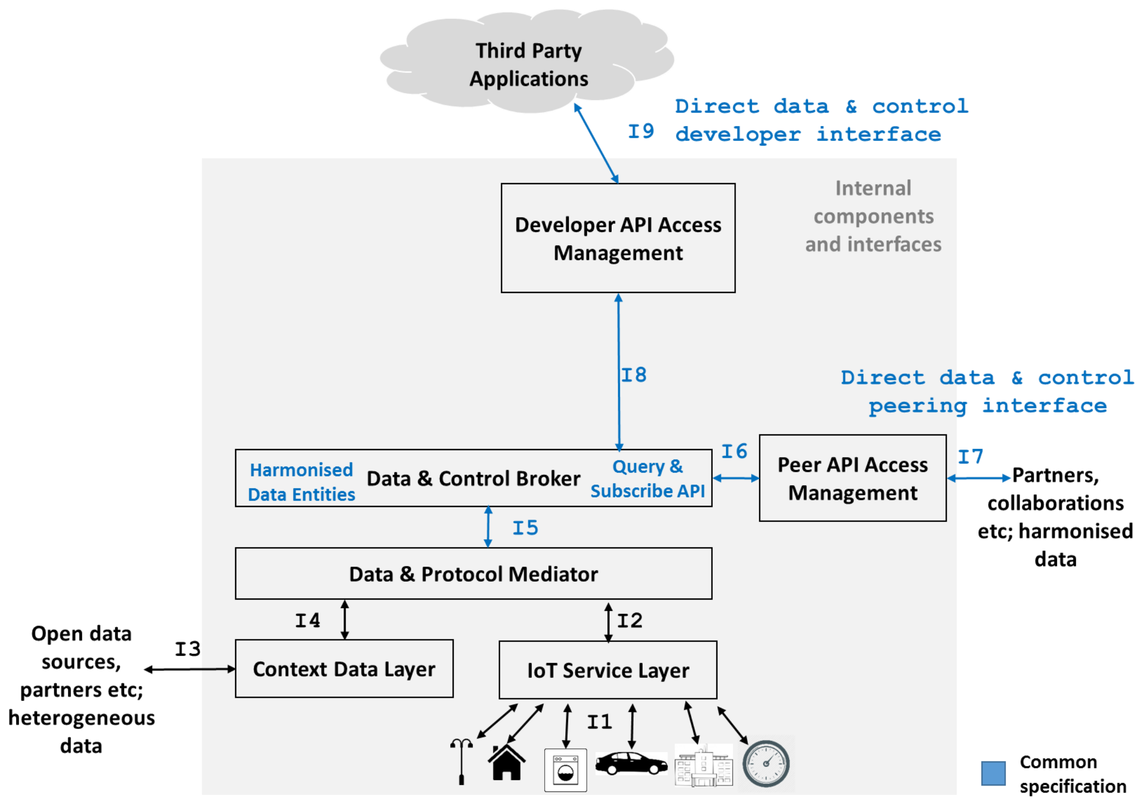
Third Party Applications
I9
Direct data & control developer interface
Internal components and interfaces
Developer API Access Management
I8
Direct data & control peering interface
Harmonised Data Entities
Data & Control Broker
Query & Subscribe API
I6
Peer API Access Management
I7
Partners, collaborations etc; harmonised data
I5
Data & Protocol Mediator
I4
I2
Open data sources, partners etc; heterogeneous data
I3
Context Data Layer
IoT Service Layer
I1
Common specification

Figure 2: Architecture for offering only Direct Data/ Control Services

5.2 Data Provider offering Analytics/ Intelligence Only

In this scenario the IoT Big Data provider is offering Big Data Analytics and Intelligence services exclusively covering IoT devices they are connecting to or managing.

In this case there is no provision for Direct Data or Control exposure for the IoT devices so this particularly suits data providers who want to provide the higher level analytics and intelligence services directly as opposed to providing "raw" entity data & control to third party application developers. Therefore the monetisation opportunity is considered to be through the offering of higher value analytics and intelligence.

Note: In this example there is no provision for the sharing (either incoming or outgoing) of IoT device data or control services to peer organisations though this could be enabled by implementing also the Data & Control Peering Interface as shown in the next scenario. This means that as shown any IoT Big Data analytics and intelligence services can only span the users and devices 'managed' by the data provider.

The restrictions of this scenario are therefore:

- It is not possible to run analytics/ intelligence across the whole of a user base/ IoT device base if the users/ devices are distributed across multiple data providers. This can be mitigated by implementing the Data & Control Peering Interface as shown in the next scenario;
- It is not possible to share "raw" IoT device data or control with other key partners e.g. government departments (though they could be provided with relevant analytics/ intelligence across the subset of users/ devices served). Again this can be mitigated by implementing the Data & Control Peering Interface as per the next scenario.

This scenario allows data providers to focus on the higher added value areas of IoT Big Data services i.e. Big Data analytics and intelligence but over a limited population of devices/ users. The data provider will have a corresponding investment in setting up their IoT Big Data platforms and range of services offered.

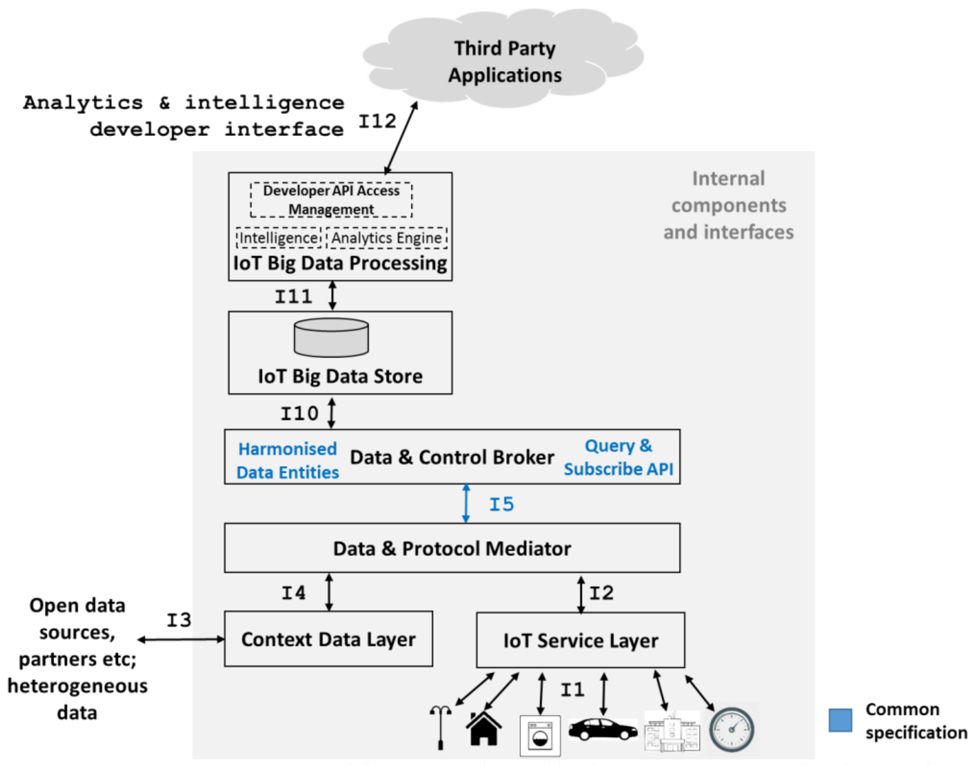

Figure 3: Architecture for offering only Analytics/ Intelligence

5.3 Data Provider offering Analytics & Intelligence over Harmonised Sources

This scenario enables more advanced analytics and intelligence across greater IoT device sets. Unlike the previous scenario where analytics and intelligence can only be supported across devices the data provider has a direct connection to this architecture supports a distributed or federated model of data providers.

A good example of this would be the deployment of multiple 'Smart City' systems across a country. It is likely that each city procures and operates their own infrastructure to support their Smart City applications - even if there is a single country level technology platform choice. In practice it is expected that cities might choose from a short-list of suitable suppliers.

A mobile operator in this case working as the aggregating data provider might connect to each Smart City system so that there is the opportunity of delivering Big Data analytics and intelligence at a country level e.g. analysis which identifies complex patterns across cities.

This scenario is supported by the Data & Control Peering interface which enables the indirect connection of other IoT devices using the platforms of partners to 'proxy' the supply of IoT device data and control services.

As described earlier the recommendation is to standardise on the sharing of data and/or control over this interface using the FIWARE NGSIv2 API and via Harmonised Entity Definitions. This enables a highly scalable approach to ecosystem growth (compared with the alternative where each data provider defines the exposure interface).

The same Data & Control Peering Interface can also support the exposure of data & control services for devices which the data provider directly connects to/ manages. This enables peer organisations such as

- Other data providers e.g. mobile operators who may be providing connectivity to other mobile subscribers/ IoT devices and may be focusing on different industry verticals;
- Major partners e.g. government departments who are collecting data across a range of data providers for bespoke applications e.g. infrastructure analysis/ planning or security applications.

This architecture allows data providers to focus on the highest added value areas of IoT Big Data services i.e. Big Data analytics and intelligence across a complete device population. The data provider can also work co-operatively with other organisations to enable growth of the ecosystem.

It is noted that this architecture involves investment in the more complex parts of the Big Data platform i.e. the Big Data store and processing infrastructure as well as the range of supporting services.

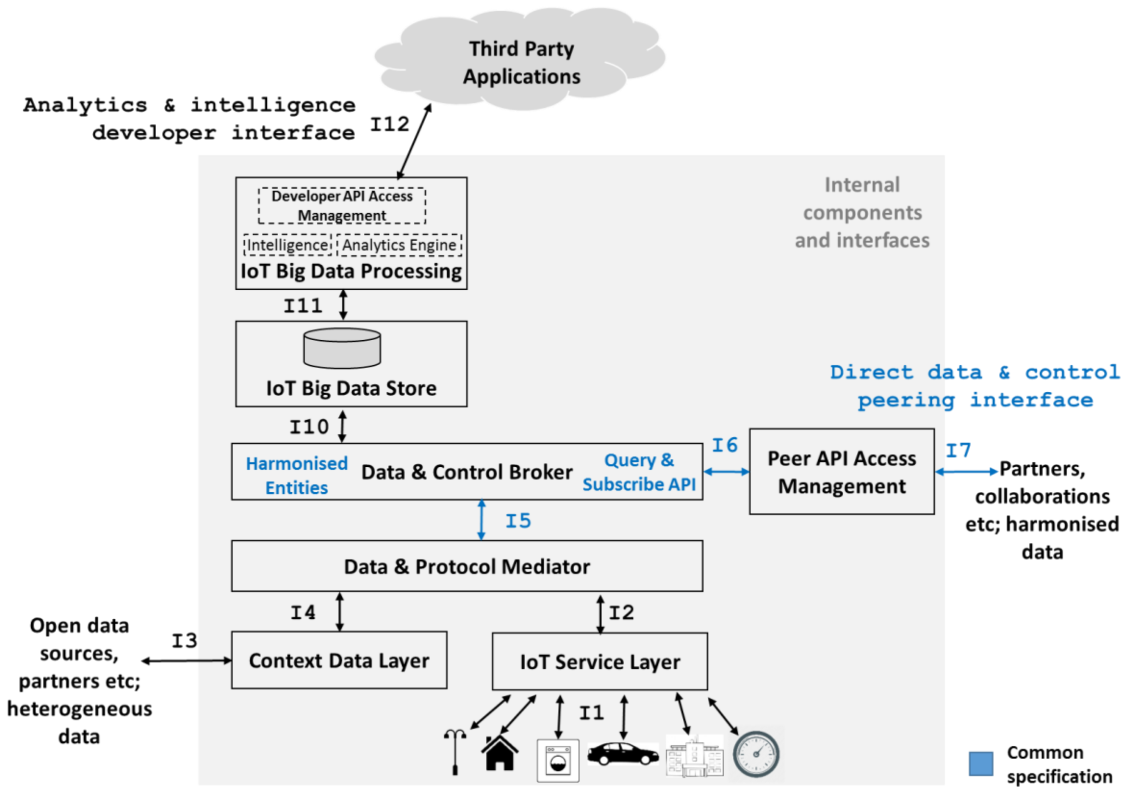
Third Party Applications
Analytics & intelligence developer interface
I12
Internal components and interfaces
Developer API Access Management
Intelligence
Analytics Engine
IoT Big Data Processing
I11
IoT Big Data Store
I10
Direct data & control peering interface
Harmonised Entities
Data & Control Broker
Query & Subscribe API
I6
Peer API Access Management
I7
Partners, collaborations etc; harmonised data
I5
Data & Protocol Mediator
I4
I2
Open data sources, partners etc; heterogeneous data
I3
Context Data Layer
IoT Service Layer
I1
Common specification

Figure 4: Architecture for offering Analytics & Intelligence over Harmonised Sources

5.4 Fully Scoped IoT Big Data Platform

In this final scenario the architecture supports the widest range of use cases through offering support for:

- The Direct Data & Control interfaces (developer and peering) allow access to current IoT device data (conforming to the Harmonised Entity Definitions) as well as short to medium term historical snapshots of the same;
- "Live" access to Analytics & Intelligence which in some use cases may also be used alongside Direct Data & Control;
- The option of aggregating IoT data / control across multiple providers.

This architecture builds upon all of the previous scenarios so that the most sophisticated use cases can be built using the IoT Big Data platforms.

As described previously it is expected certain data (and control) will be contained within islands of service e.g. the case there are multiple 'Smart City' systems across a country. This architecture continues the support for a distributed or federated model of data providers.

It is expected that in this case the mobile operator is typically working as the aggregating data provider. The Big Data analytics and intelligence capabilities can identify complex patterns across multiple cities and in this architecture this can be considered/ correlated with current IoT sensor data so that "real time" decisions can be made. As substantial processing may be involved "real time" in this situation is expected to be measured as a period of seconds rather than harder real time response which may need to be measured in milliseconds or less. Such decisions may be used for the generations of user alerts but could also be turned into control actions that are then used to command IoT devices in the field.

For example Big Data analytics and intelligence could be used to identify historical weather/ crop performance data which is processed against real time field sensors and this could be used to control watering systems to maintain optimal plant growth. Alternatively a decision could be made to harvest a crop if it is considered likely the peak condition/ yield has been achieved.

This architecture expects use of the Data & Control Peering Interface to support "region wide" or "country wide" analytics and intelligence service development.

As described earlier the recommendation is to enable sharing of data and/or control over this interface by standardising on use of the FIWARE NGSIv2 API and by transferring data and control via Harmonised Entity Definitions. This enables a highly scalable approach to ecosystem growth (compared with the alternative where each data provider defines the exposure interface).

The same Data & Control Peering Interface can also support the exposure of data & control services for devices which the data provider directly connects to/ manages. This enables peer organisations such as

- Other data providers e.g. mobile operators who may be providing connectivity to other mobile subscribers/ IoT devices and may be focusing on different industry verticals;
- Major partners e.g. government departments who are collecting data across a range of data providers for bespoke applications e.g. infrastructure analysis/ planning or security applications.

This architecture allows data providers to support the widest range of IoT Big Data services including the highest value Big Data analytics and intelligence services across a complete population. The data provider can also work co-operatively with other organisations to enable growth of the ecosystem.

This architecture comprises the most parts and therefore will involve the largest investment particularly in the more complex parts of the IoT Big Data platform i.e. the IoT Big Data store and processing infrastructure. As mentioned it is expected this architecture also supports the most complicated use cases as well as making it easy for developers to create applications which leverage IoT Big Data.

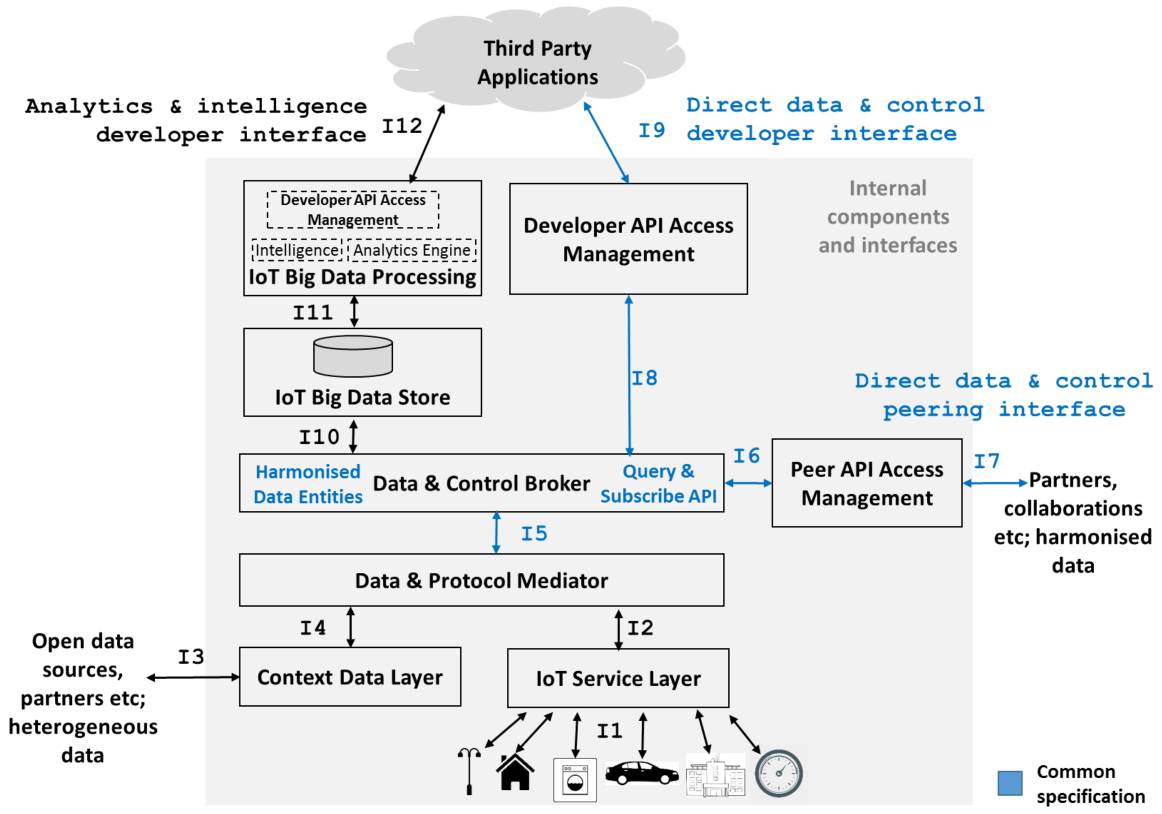
Third Party Applications
Analytics & intelligence developer interface
I12
Direct data & control developer interface
I9
Developer API Access Management
Intelligence
Analytics Engine
IoT Big Data Processing
Developer API Access Management
Internal components and interfaces
I11
IoT Big Data Store
I8
Direct data & control peering interface
I10
Harmonised Data Entities
Data & Control Broker
Query & Subscribe API
I6
Peer API Access Management
I7
Partners, collaborations etc; harmonised data
I5
Data & Protocol Mediator
I4
I2
Open data sources, partners etc; heterogeneous data
I3
Context Data Layer
IoT Service Layer
I1
Common specification

Figure 5: General architecture for fully scoped IoT Big Data

Annex A Application of FIWARE components to IoT Big Data architecture

Within this document the FIWARE NGSIv2 interface has been specified as the recommended standard for certain interfaces and proposed for others. FIWARE also provide several open source components that support various logical functions in the architecture framework, with the primary example being the 'Orion Context Broker[33]' which implements NGSIv2 and can perform the function of the Data & Control Broker.

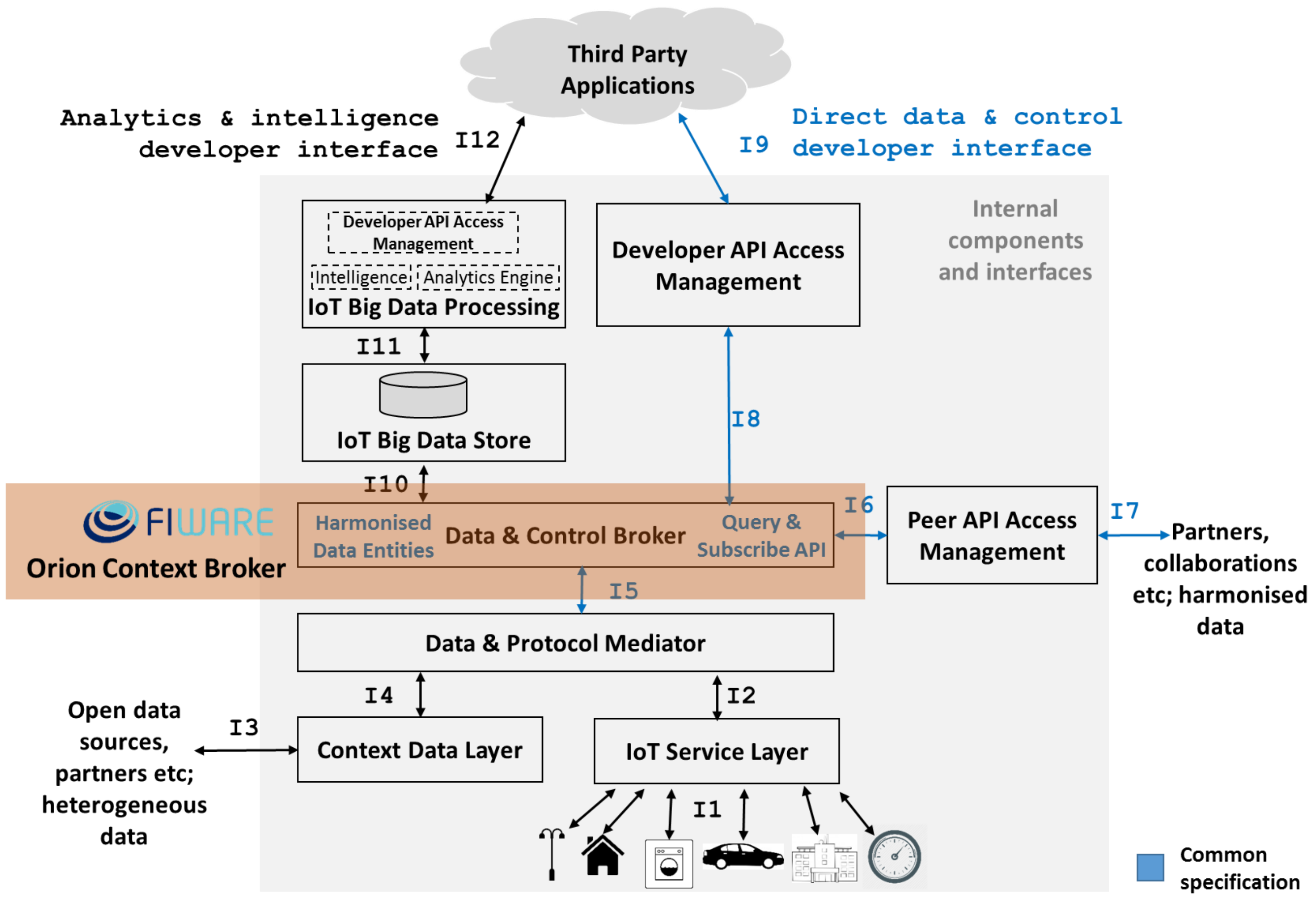
Third Party Applications
Analytics & intelligence developer interface
I12
I9
Direct data & control developer interface
Internal components and interfaces
Developer API Access Management
Intelligence
Analytics Engine
IoT Big Data Processing
Developer API Access Management
I11
I8
IoT Big Data Store
I10
FIWARE
Orion Context Broker
Harmonised Data Entities
Data & Control Broker
Query & Subscribe API
I6
Peer API Access Management
I7
Partners, collaborations etc; harmonised data
I5
Data & Protocol Mediator
I4
I2
Open data sources, partners etc; heterogeneous data
I3
Context Data Layer
IoT Service Layer
I1
Common specification

Figure 6: General architecture and FIWARE components

The FIWARE Orion Context Broker is an implementation of a generalised Publish/Subscribe Context Broker providing an NGSIv2 interface. Using its interfaces, clients can perform several operations:

- Update context information, e.g. send updates of temperature

[33] http://catalogue.fiware.org/enablers/publishsubscribe-context-broker-orion-context-broker

- Being notified when changes on context information take place (e.g. the temperature has changed)
- Query context information. The Orion Context Broker stores context information updated from applications, so queries are resolved based on that information.

Annex B Background information about FIWARE NGSI

The Next Generation Service Interfaces (NGSI) Enabler, defined by the Open Mobile Alliance (OMA), comprises a number of interfaces for Data Configuration and Management, Call Control and Configuration, Multimedia List Handling, Context Management, Identity Control, Service Registration and Discovery functions. In particular, the NGSI-9 and NGSI-10 interfaces define the Context Management Functions of the NGSI Enabler, and cover common operations and data structures enabling development of enhanced applications, exploiting distributed context data (location, preferences, network usage, etc.) provided by multiple parties, for example, mobile operators.

The NGSI Context Management specifications define abstract interfaces and data structures without prescribing a specific binding to any programming language, protocol or implementation technology. The NGSI-9 interface comprise operations for registering and discovering the availability of distributed context information sources (context providers), which actually are in charge of providing context data. The NGSI-10 interface comprise operations for querying, updating and subscribing to changes in context information.

In 2012, the FIWARE initiative (http://fiware.org) adopted NGSI-9 and NGSI-10 as the basis for the open specification of the API interface supported by the Context Broker Generic Enabler (GE), a key component of the FIWARE platform. This way, the FIWARE NGSI API specification was designed to work as a concrete API binding of the abstract NGSI-9 and NGSI-10 interfaces defined by OMA. As FIWARE intends to follow a driven-by implementation approach, parallel to the specifications, the FIWARE initiative also produced an open source reference implementation of the FIWARE Context Broker GE, named as Orion Context Broker. The first specification of the FIWARE NGSI API was based on an HTTP + XML binding "a la RPC", using POST requests and relatively complex payloads. With the increasing popularity of JSON, a HTTP+JSON binding was added one year later, transforming the payloads to a slightly less verbose, JSON format. Additionally, with the aim of providing simpler interfaces, some convenient REST (GET, DELETE) operations were included which simplified some common use cases. The FIWARE NGSI API available until May 2015 was given the name of FIWARE NGSI version 1 (FIWARE NGSIv1) and it is still maintained for backwards compatibility purposes.

Meanwhile, from 2013 to 2015 the feedback coming from developers building applications based on FIWARE was gathered. There were requests for a complete RESTful approach to the API, together with the simplification of payloads which were considered quite verbose and unfriendly for developers. Other developers were advocating the adoption of emerging standards such as GeoJSON for representing geospatial properties of entities. Last but not least, the absence of a query language for entity data resulted in workarounds for data filtering that were difficult to maintain. An important milestone was the release of a Node.js library[34] which made outstanding simplifications to the NGSI API, making it easier for developers to get started with the technology.

Based on the gathered feedback and wrapper libraries, a clean, fresh, brand new API version started to be developed in June 2015, being referred as FIWARE NGSI version 2 (NGSIv2). Over the course of one year the specification and implementation teams of the API have been working closely to create a new version of the API specifications and an accompanying open source reference implementation. In June 2016 the first release candidate of FIWARE NGSIv2 (RC-2016.05) was unveiled[35], together with the latest version of the Orion Context Broker product, the open source reference implementation of FIWARE NGSIv2. At the time of writing FIWARE NGSIv2 is being polished and completed (see roadmap) with new features, with the aim of promoting its wider adoption among the IoT Big Data, telecoms, industry or standards organizations and developer communities.

[34] https://www.npmjs.com/package/fiware-orion-client

[35] https://www.fiware.org/2016/06/08/fiware-ngsi-version-2-release-candidate/

Annex C Document Management

C.1 Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
0.10	8 Sept 16	New PRD - first draft	PSMC	Allan Bartlett / GSMA
1.0	11 Oct 16	Approved	PSMC	Allan Bartlett / GSMA

Other Information

Type	Description
Document Owner	IoT Big Data Ecosystem Project
Editor / Company	Allan Bartlett / GSMA

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at prd@gsma.com

Your comments or suggestions & questions are always welcome.