IoT Reference Architecture

Introduction

In this chapter we describe the IoT Reference Architecture. As mentioned earlier, the Reference Architecture is a starting point for generating concrete architectures and actual systems. A concrete architecture addresses the concerns of multiple stakeholders of the actual system, and it is typically presented as a series of views that address different stakeholder concerns

A Reference Architecture, on the other hand, serves as a guide for one or more concrete system architects. However, the concept of views for the presentation of an architecture is also useful for the IoT Reference Architecture. Views are useful for reducing the complexity of the Reference Architecture blueprints by addressing groups of concerns one group at a time. However, since the IoT Reference Architecture does not contain details about the environment where the actual system is deployed,

The stakeholders for a concrete IoT system are the people who use the system (Human Users); the people who design, build, and test the Resources, Services, Active Digital Artifacts, and Applications; the people who deploy Devices and attach them to Physical Entities; the people who integrate IoT capabilities of functions with an existing ICT system (e.g. of an enterprise);the people who operate, maintain, and troubleshoot the Physical and Virtual Infrastructure; and the people who buy and own an IoT system or parts thereof (e.g. city authorities).

In order to address the concerns of mainly the concrete IoT architect,and secondly the concerns of most of the above stakeholders, we havechosen to present the Reference Architecture as a set of architectural views (Kruchten 1995; SEI CMU 2013; Rozanski & Woods 2005, 2011):

• **Functional View**: Description of what the system does, and its main functions.

• **Information View**: Description of the data and information that the system handles.

• **Deployment and Operational View**: Description of the main real world components of the system such as devices, network routers,servers, etc.

**Functional view**

The functional view for the IoT Reference Architecture is presented inFigure and is adapted from IoT-A (Carrez et al. 2013). It consists of the Functional Groups (FGs) presented earlier in the IoT Functional Model, each of which includes a set of Functional Components (FCs). It is important to note that not all the FCs are used in a concrete IoT architecture



**Device and Application functional group**

The Device and Application FGs are already covered in the IoT Functional Model. For convenience the Device FG contains the Sensing, Actuation, Tag, Processing, Storage FCs, or simply components. These components represent the resources of the device attached to the Physical Entities of interest. The Application FG contains either standalone applications (e.g.for iOS, Android, Windows phone), or Business Applications that connect the IoT system to an Enterprise system.

**communication functional group**

The Communication FG contains the End-to-End Communication, Network Communication, and Hop-by-Hop communication components:

• **The Hop-by-Hop Communication** is applicable in the case that devices are equipped with mesh radio networking technologies such as IEEE 802.15.4 for which messages have to traverse the mesh from node-to-node (hop-by-hop) until they reach a gateway node which forwards the message (if needed) further to the Internet. The hop-by-hop FC is responsible for transmission and reception ofphysical and MAC layer frames to/from other devices. This FC has two main interfaces: (a) one “southbound” to/from the actual radio on the device, and (b) one “northbound” to/from the Network FC in the Communication FG.

• **The Network FC** is responsible for message routing & forwarding and the necessary translations of various identifiers and addresses. The translations can be (a) between network layer identifiers to MAC and/or physical network identifiers, (b) between high-level human readable host/node identifiers to network layer addresses (e.g. Fully Qualified Domain Names (FQDN) to IP addresses, a function implemented by a Domain Name System (DNS) server), and (c) translation between node/service identifiers and network locators in case the higher layers above the networking layer use node or service identifiers that are decoupled from the node addresses in the network. Network FC is responsible for handling messages that cross different networking or MAC/PHY layer technologies, a function that is typically implemented on a network gateway type of device. The Network FC interfaces the End-to-End Communication FC on the “northbound” direction, and the Hop-by-Hop

**The End-to-End Communication FC :** is responsible for end-to-end transport of application layer messages through diverse MAC/PHY layers. In turn, this means that it may be responsible for end-to-end retransmissions of missing frames depending on the configuration of the FC. For example, if the End-to-End Communication FC is mapped in an actual system to a component implementing the Transmission Control Protocol (TCP) protocol, reliable transfer of frames dictates the retransmission of missing frames. Finally, this FC is responsible for hosting any necessary proxy/cache and any protocol translation between networks with different transport/applicationlayer technologies.

**IoT Service functional group**

The IoT Service FG consists of two FCs: The IoT Service FC and the IoTService Resolution FC:

• **The IoT Service FC** is a collection of service implementations, which interface the related and associated Resources. For a Sensor type of a Resource, the IoT Service FC includes Services that receive requests from a User and returns the Sensor Resource value in synchronous or asynchronous (e.g. subscription/notification) fashion. The services corresponding to Actuator Resources receive User requests for actuation ,control the Actuator Resource, and may return the status of the Actuator after the action. A Tag IoT Service can behave both as a Sensor (for reading the identifier of the Tag), or as an Actuator (for writing a new identifier or information on the Tag, if possible). As mentioned earlier, Resources can also perform processing and storage (Processing or Storage Resources), and therefore their corresponding Service exposes the corresponding interfaces, An IoT Service for a particular Resource could also expose as a service the historical values of sensor values or actuator commands or tag identifiers.

• **The IoT Service Resolution FC**: contains the necessary functions to realize a directory of IoT Services that allows dynamic management of IoT Service descriptions and discovery/lookup/resolution of IoT Services by other Active Digital Artifacts. The Service descriptions of IoT Services m contain a number of attributes as seen earlier in the IoT Functional Model section. Dynamic management includes methods such as creation/update/deletion (CUD) of Service description, and can be invoked by both the IoT Services themselves, or functions from the Management FG (e.g.bulk creation of IoT Service descriptions upon system start-up). The discovery/lookup and resolution functions allow other Services or Active Digital Artifacts to locate IoT Services by providing different types of information to the IoT Service Resolution FC.

By providing the Service identifier (attribute of the Service description) a lookup method invocation to the IoT Service Resolution returns the Service description, while the resolution method invocation returns the contact information (attribute of the service description) of a service for direct Service invocation (e.g. URL). The discovery method, on the other hand, assumes that the Service identifier is unknown, and the discovery request contains a set of desirable Service description attributes that matching Service descriptions should contain.

**Virtual Entity functional group**

The Virtual Entity FG contains functions that support the interactions between Users and Physical Things through Virtual Entity services The Virtual Entity interaction paradigm requires functionality such as discovery of IoT Services based on Virtual Entity descriptions, managing the Virtual Entity-IoT Service associations, and processing Virtual Entity-based queries. The following FCs are defined for realizing these functionalities:

1. **The Virtual Entity Service FC** enables the interaction between Users and Virtual Entities by means of reading and writing the Virtual Entity attributes (simple or complex), which can be read or written, of course. Some attributes (e.g. the GPS coordinates of a room) are static and non-writable by nature, and some other attributes are non-writable by access control rules. In general attributes that are associated with IoT Services, which in turn represent Sensor Resources, can only be read. There can be, of course, special Virtual Entities associated with the same Sensor Resource through another IoT Service that allow write operations. Virtual Entity attributes corresponding to Tags can be read in most of the cases by Users, and can be written by special cases by other types of Users (e.g. Management applications), if possible of course, as is the case of re-writable RFID tags. Apart from the function to operate on the Virtual Entity attributes, a Virtual Entity Service can also expose the historical variations of the attributes of a Virtual Entity
2. **The Virtual Entity Registry FC m**aintains the Virtual Entities of interest for the specific IoT system and their associations. The component offers services such as creating/reading/updating/deleting Virtual Entity descriptions and associations. Certain associations can be static for example, the entity “Room #123” is contained in the entity “Floor #7”by construction, while other associations are dynamic, e.g. entity “Dog” and entity “Living Room” due to at least Entity mobility. Update and Deletion operations take the Virtual Entity identifier as a parameter.
3. **The Virtual Entity Resolution FC** maintains the associations between Virtual Entities and IoT Services, and offers services such as creating/reading/updating/deleting associations as well as lookup an discovery of associations. The Virtual Entity Resolution FC also provides notification to Users about the status of the dynamic associations between a Virtual Entity and an IoT Service, and finally allows the discovery of IoT Services provided the certain Virtual Entity attributes.
4. **The Virtual Entity and IoT Service Monitoring FC** includes: (a)functionality to assert static Virtual EntityIoT Service associations, (b)functionality to discover new associations based on existing associations or Virtual Entity attributes such as location or proximity, and (c) continuous monitoring of the dynamic associations between Virtual Entities and IoT Services and updates of their status in case existing associations are not valid any more.

**IoT process management functional group**

The IoT Process Management FG aims at supporting the integration of business processes with IoT-related services. It consists of two FCs:

• **The Process Modeling FC**: provides that right tools for modeling abusiness process that utilizes IoT-related services.

• **The Process Execution FC:** contains the execution environment of the process models created by the Process Modelling FC and executes the created processes by utilizing the Service Organization FG in order to resolve high-level application requirements to specific IoT services

It is important to note the IoT services mentioned above are not on the services from the IoT Service FG, but also from the Virtual Entity and the Service Organization FG.

**Service Organization functional group**

The Service Organization FG acts as a coordinator between different Services offered by the system. It consists of the following FCs:

1 **The Service Composition FC** manages the descriptions and execution environment of complex services consisting of simpler dependent services. An example of a complex composed service is a service offering the average of the values coming from a number of simple Sensor Services. The complex composed service descriptions can be well specified or dynamic/flexible depending on whether the constituent services are well-defined and known at the execution time or discovered on-demand. The objective of a dynamic composed service can be the maximization of the quality of information achieved by the composition of simpler Services, as is the case with the example “average” service described earlier.

2 **The Service Orchestration FC** resolves the requests coming from IoT Process Execution FC or User into the concrete IoT services that fulfil the requirements.

3. **The Service Choreography** FC is a broker for facilitating communication among Services using the Publish/Subscribe pattern. Users and Services interested in specific IoT-related services subscribe to the Choreography FC, providing the desirable service attributes even if the desired services do not exist. The Choreography FC notifies the Users when services fulfilling the subscription criteria are found.

It is important to note that the IoT services mentioned above are not only the services from the IoT Service FG, but also from the Virtual Entity FG and the Service Composition FC.

**Security functional group**

The Security FG contains the necessary functions for ensuring the security and privacy of an IoT system. It consists of the following FCs:

1. **The Identity Management F**C manages the different identities of the involved Services or Users in an IoT system in order to achieve anonymity by the use of multiple pseudonyms. The component maintains a hierarchy of identities (an identity pool), as well as group identities
2. **The Authentication FC** verifies the identity of a User and creates an assertion upon successful verification. It also verifies the validity of a given assertion.
3. **The Authorization FC** manages and enforces access control policies. It provides services to manage policies (CUD), as well as taking decisions and enforcing them regarding access rights of restricted resources. The term “resource” here is used as a representation of any item in an IoT system that needs a restricted access. Such an item can be a database entry (Passive Digital Artifact), a Service interface, a Virtual Entity attribute (simple or complex), a Resource/Service/Virtual Entity description, etc.
4. **The Key Exchange & Management** is used for setting up the necessary security keys between two communicating entities in an IoT system. This involves a secure key distribution function between communicating entities.
5. • **The Trust & Reputation FC** manages reputation scores of different interacting entities in an IoT system and calculates the service trust levels. A more detailed description of this FC is contained in the Safety, Privacy, Trust, Security Model.

**Management functional group**

The Management FG contains system-wide management functions that may use individual FC management interfaces. It is not responsible for the management of each component, rather for the management of the system as a whole. It consists of the following FCs:

1. **The Configuration FC** maintains the configuration of the FCs and the Devices in an IoT system (a subset of the ones included in the Functional View). The component collects the current configuration of all the FCs and devices, stores it in a historical database, and compares current and historical configurations. The component can also set the system-wide configuration (e.g. upon initialization), which in turn translates to configuration changes to individual FCs and devices.
2. **The Fault FC detects**, logs, isolates, and corrects system-wide faults if possible. This means that individual component fault reporting trigger fault diagnosis and fault recovery procedures in the Fault FC.
3. **The Member FC** manages membership information about the relevant entities in an IoT system. Example relevant entities are the FGs, FCs, Services, Resources, Devices, Users, and Applications. Membership information is typically stored in a database along with other useful information such as capabilities, ownership, and access rules & rights which are used by the Identity Management and Authorization FCs.
4. **The State FC** is similar to the Configuration FC, and collects and logs state information from the current FCs, which can be used for fault diagnosis, performance analysis and prediction, as well as billing purposes. This component can also set the state of the other FCs based on system-wise state information.
5. **The Reporting FC** is responsible for producing compressed reports about the system state based on input from FCs.

Real-World Design Constraints

**Introduction :**

This chapter outlines the technical design constraints to illustrate the questions that need to be taken into account when developing and implementingM2M and IoT solutions in the real world.

**Technical design constraints -hardware is popular again**

The IoT will see additional circuitry built into a number of existing products and machines, from washing machines to meters. Giving these things an identity, and the ability to represent themselves online and communicate with applications and other things, represents a significant, widely recognized opportunity.

For manufacturers of products that typically contain electronic components this process will be relatively straightforward. Selection of appropriate communications technologies that can be integrated with legacy designs (e.g. motherboards) will be relatively painless. The operational environments and the criticality of the information transmitted to and from these products, however, will present some unconventional challenges and design considerations

The IoT will, on the other hand, allow for the development of novel applications in all imaginable scenarios. Emerging applications of M2M and wireless sensor and actuator networks have seen deployment of sensing capabilities in the wild that allow stakeholders to optimize their businesses, glean new insight into relevant physical and environmental processes, and understand and control situations that would have previously been inaccessible.

The technical design of any M2M or IoT solution requires a fundamental understanding of the specificity of the intended application and business proposition, in addition to heterogeneity of existing solutions. Developing an end-to-end instance of an M2M or IoT solution requiresthe careful selection, and in most cases, development of a number of complementary technologies. This can be both a difficult conceptual problem and integration challenge, and requires the involvement of the key stakeholder(s) on a number of conceptual and technological levels. Typically, it can be considered to be a combinatorial optimization problem , where the optimal solution is the one that satisfies all functional and non-functional requirements, whilst simultaneously delivering a satisfactory cost-benefit ratio. This is particularly relevant for organizations wishing to compete with existing offerings, or for start-up ventures in novel application areas. Typically, capital costs in terms of “commissioning” and operational costs in “maintenance” must be considered. These may be balanced by resultant optimizations

**Devices and networks**

,Devices that form networks in the M2M Area Network domain must be selected, or designed, with certain functionality in mind. At a minimum, they must have an energy source (e.g. batteries, increasingly EH), computational capability (e.g. an MCU), appropriate communications interface (e.g. a Radio Frequency Integrated Circuit(RFIC) and front end RF circuitry), memory (program and data), and sensing

(and/or actuation) capability. These must be integrated in such a way that the functional requirements of the desired application can be satisfied, in addition to a number of non-functional requirements that will exist in all cases.

**Functional requirements**

Specific sensing and actuating capabilities are basic functional requirements .In every case with the exception of devices that might be deployed as a routing device in the and/or actuating devices the device must be capable of sensing or perceiving something interesting from the environment. This is the basis of the application. Sensors, broadly speaking, are difficult to categorize effectively. Selecting a sensor that is capable of detecting a particular phenomenon of interest is essential. The sensor may directly measure the phenomenon of interest (e.g. temperature), or may be used to derive data or information about the phenomenon of interest, based on additional knowledge (e.g. a level of comfort). Sensors may sense a phenomenon that is local (i.e. a meter detecting total electricity consumption of a space) or distributed (e.g. the weather).

Sensing principle and data requirements are also of essence when considering the real-world application. Consider a continuously sampling sensor, such as an accelerometer, versus a displacement transducer. Displacement can be sampled intermittently, whereas if an accelerometer is duty-cycled, it is likely that data points of interest (i.e. real-world events) may be missed. Furthermore, the data requirements of the stakeholder must be taken into account. If all data points are required to be transmitted (which is the case in many scenarios, irrespective of the ability to reason locally within an M2M Area Network or WSN), this implies higher network throughput, data loss, energy use, etc. These requirements tend to change on a case-by-case basis.

**Sensing and communications field**

The sensing field is of importance when considering both the phenomenon to be sensed (i.e. Is it local or distributed?) and the distance between sensing points. The physical environment has an implication on the communications technologies selected and the reliability of the system in operation thereafter. Devices must be placed in close enough proximity to communicate. Where the distance is too great, routing devices may be necessary. Devices may become intermittently disconnected due to the time varying, stochastic nature of the wireless medium. Certain environments may be fundamentally more suited to wireless propagation than others. For example, studies have shown that tunnels are excellent environments for wireless propagation, whereas, where RF shielding can occur (e.g. in a heavy construction environment), communication range of devices can be significantly reduced.

**Programming and embedded intelligence**

Devices in the IoT are fundamentally heterogeneous. There are, and will continue to be, various computational architectures, including MCUs (8-, 16-,32-bit, ARM, 8051, RISC, Intel, etc.), signal conditioning (e.g. ADC), and memory (ROM, (S/F/D)RAM, etc.), in addition to communications media, peripheral components (sensors, actuators, buttons, screens, LEDs), etc. In some applications, where it would previously have been typical to have homogeneous devices, a variety of sensors and actuators can actually exist, working collaboratively, but constituting a heterogeneous network in reality.

In every case, an application programmer must consider the hardware selected or designed, and its capabilities. Typically, applications may be thought of cyclically and logically. Application-level logic dictates the sampling rate of the sensor, the local processing performed on sensor readings, the transmission schedule (or reporting rate), and the management of the communications protocol stack, among other things.

The ability to reconfigure and reprogram devices is still an unresolved issue for the research community in sensor networks, M2M, and the IoT .It relates both to the physical composition of devices, logical construction of the embedded software, and addressability of individual devices and security, to name a few. Operating systems are typically used to make programming simpler and modular for embedded systems designers, but each comes with conceptual and implementation differences that impact the ability to handle certain desirable features

**Power**

Power is essential for any embedded or IoT device. Depending on the application, power may be provided by the mains, batteries, or conversion from energy scavengers (often implemented as hybrid power sources). The power source has a significant implication on the design of the entire system .If a finite power supply is used, such as a battery, then the hardware selected, in addition to the application level logic and communications technology, collectively have a major impact on the longevity of the application. This results in short-lived applications or increased maintenance costs. In most cases, it should be possible to analytically model the power requirements of the application prior to deployment. This allows the designer to estimate the cost of maintenance over time.

**Gateway**

The Gateway, is typically more straightforward to design if it usually acts as a proxy; however, there are very few effectiveM2M or IoT Gateway devices available on the market today. Depending on the application, power considerations must be taken into account. It is also thought that the Gateway device can be exploited for performing some level of analytics on data transitioning to and from capillary networks.

**Non-functional requirements**

There are a number of non-functional requirements that need to be satisfied

for every application. These are technical and non-technical:

• **Regulations:**

1. For applications that require placing nodes in public places, planning permission often becomes an issue.
2. **Radio Frequency (RF**) regulations limit the power with which transmitters can broadcast. This varies by region and frequency band.
3. **Ease of use, installation, maintenance, accessibility:** Simplification of installation and configuration of IoT applications is as yet unresolved beyond well-known, off-the-shelf systems. It is difficult to conceive a general solution to this problem. This relates to positioning, placement, site survey in programming, and physical accessibility of devices for maintenance purposes.
4. **Physical constraints (from several perspectives):**

* Can the additional electronics be easily integrated into the existing system?
* Are there physical size limitations on the device as a result of the deployment scenario?
* What kind of packaging is most suitable (e.g. IP-rated enclosures or outdoor deployment)?
* What kind and size of antenna can I use?
* What kind of power supply can I use given size restrictions (relates to harvesting, batteries, and alternative storage, e.g. super capacitors?

**Financial cost**

Financial cost considerations are as follows:

• ***Component Selection:*** Typically, the use of these devices in theM2M Area Network domain is seen to reduce the overall cost burden by using non-leased communications infrastructure. However, there are research and development costs likely to be incurred for each individual application in the IoT that requires devic e development or integration. Developing devices in small quantities is expensive.

**• Integrated Device Design**: Once the energy, sensors, actuators, computation, memory, power, connectivity, physical, and other functional and non-functional requirements are considered, it is likely that an integrated device must be produced. This is essentially going to be an exercise in Printed Circuit Board (PCB) design, but will in many cases require some consideration to be paid to the RF front-end design. This means that the PCB design will require specific attention to be paid to the reference designs of the RFIC manufacturer during development, or potentially the integration of an additional Integrated Circuit IC) that deals with the balun and matching network required.

**Data representation and visualization**

Each IoT application has an optimal visual representation of the data and the system. Data that is generated from heterogeneous systems has heterogeneous visualization requirements. There are currently no satisfactory standard data representation and storage methods that satisfy all of the potential IoT applications. Data-derivative products will have further ad hoc visualization requirements.

A derivative in these terms exists once a function has been performed on an initial data set which may or may not be raw data. These can be further integrated at various levels of abstraction, depending on the logic of the integrator. New information sources, such as those derived from integrated data streams from various logically correlated IoT applications, will present interesting representation and visualization challenges.

**Interaction and remote control:**

To exploit remote interaction and control over IoT applications, connectivity that spans the traditional Internet (i.e. from anywhere) on the side of the application manager, or other authorized entity, to the end-point (i.e.an embedded device), continues to be a challenging problem. Aside from authentication and availability challenges, for most constrained devices, heterogeneous software architectures, such as event-based operating systems running on devices with significantly varying concurrency models,

continue to pose significant challenges from a remote management perspective. Elements of Device Management, specifically reprogramming and reconfiguration of deeply embedded devices, will be required, particularly for devices deployed in inaccessible locations. This requires, among others, reliability, availability, security, energy efficiency, and latency performance, to be satisfactory whilst communicating across complex distributed systems.

Another significantly under-researched topic is the definition and delivery of end-to-end quality of service (QoS) metrics and mechanisms in IoT type applications. These will be necessary if Service Agreements (SA) or Service Level Agreements (SLA) are to be defined in the case of service provisions for IoT applications which may or may not be desirable to the application owner. This will be situation-specific. End-to-end latency, security, reliability, availability, times between failure and repair, responsibility,etc., are all likely to feature in such agreements