IoT Architectural Reference Model

Internet of Things

Rahul Shandilya

IoT A Reference Model

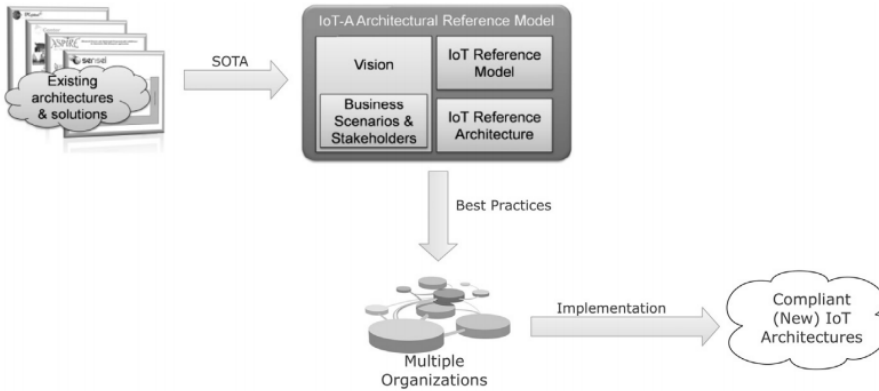
In the IoT domain, anyone can propose an own architecture and an own communication protocol, due to a wide variety of requirements to which each architecture should be compliant.
A common reference model for the IoT domain and the identification of reference architectures can help to a faster, more focused development and an exponential increase of IoT-related solutions.
The European Lighthouse Integrated Project has addressed for three years the Internet-of-Things Architecture (IoT-A) and created an architectural reference model together with the definition of an initial set of key building blocks. It wants to promote a common ground between architectures so that they can interoperate even a more levels. IoT-A has achieved that thanks to two steps:

- Establishing a Reference Model
- Providing a Reference Architecture

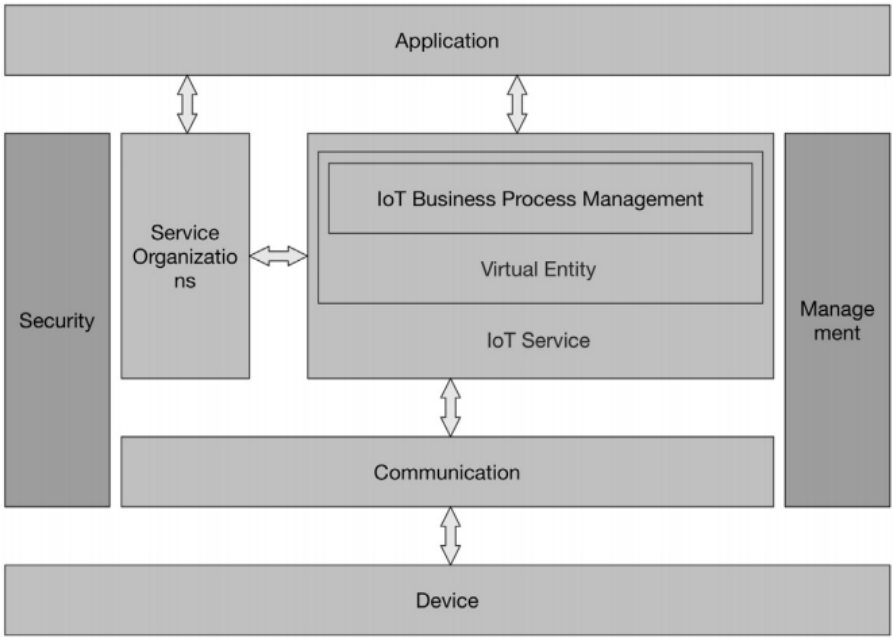
IoT-A ARM

The IoT-A ARM (Architecture Reference Model) consists of four parts:

- **Vision**: summarizes the rationale for providing an architectural reference model for the IoT;
- **Business scenarios**: define requirements provided by stakeholders that are the drivers of the architecture. They allow the architecture to be validated;
- **The IoT Reference Model**, which provides the highest abstraction level for the definition of the IoT-A Architectural Reference Model. It promotes a common understanding of the IoT domain. It includes the description of domain, communication and information model;
- **The IoT Reference Architecture**, which is the reference for building compliant IoT architectures. It provides views and perspectives on different architectural aspects that are of concern to stakeholders of the IoT.


Existing architectures & solutions
SOTA
IoT-A Architectural Reference Model
Vision
IoT Reference Model
Business Scenarios & Stakeholders
IoT Reference Architecture
Best Practices
Multiple Organizations
Implementation
Compliant (New) IoT Architectures

IoT-Information Model



Uses of IoT ARM

Cognitive Aid

When it comes to product development and other activities, an architectural reference model is of fourfold use.

- Firstly, it helps to guide discussions, since it provides a language everyone involved can use, and which is intimately linked to the architecture, the system, the usage domain
- Secondly, the high-level view provided in such a model is of high educational value, since it provides an abstract but also rich view of the domain. Such a view can help people new to the field to "find their way" and to understand the special features and intricacies of IoT.
- Thirdly, the IoT ARM can assist IoT project leaders in planning the work at hand and the teams needed.
- Fourthly, the IoT ARM helps to identify independent building blocks for IoT systems. This constitutes very valuable information when dealing with questions such as system modularity, processor architectures, third-vendor options, re-use of components already developed, etc.

Reference Model as a Common Ground

Establishing the common ground for the IoT encompasses defining IoT entities and describing their basic interactions and relationships with each other. The IoT ARM provides exactly such a common ground for the IoT field.

Generating Architectures

One of the main benefits is the use of the IoT ARM for generating compliant architectures for specific systems. This is done by providing best practices and guidance for translating the IoT ARM into concrete architectures. The benefit of this type of generation scheme for IoT architectures is not only a certain degree of automation in this process, and thus lower R&D efforts, but also that the decisions made follow a clear, documented pattern.

Identifying Differences in Derived Architectures

the IoT ARM defines a set of tactics and design choices for meeting qualitative system requirements. All of these facts can be used to predict whether two derived architectures will differ and where they will do so. The IoT ARM can also be used for reverse mapping. System architectures can be cast in the "IoT ARM" language and the resulting "translation" of the system architectures is then stripped of incompatible language and system partitions and mappings. The differences that remain are then true differences in architecture.

Achieving Interoperability

By comparing the design choices made when deriving two architectures, one can readily identify where in the architecture measures are necessary to achieve interoperability. Interoperability may be achieved a posteriori by integrating one IoT system as subsystem in another system, or by building a bridge through which key functionalities of the respective other IoT system can be used.

System Roadmaps and Product Life Cycles

the IoT ARM can be used to devise system roadmaps that lead to minimum changes between two product generations while still guaranteeing a noticeable enhancement in system capability and features. This approach also helps the designer to formulate clear and standardised, requirements-based rationales for the system roadmap chosen and the product life cycles that result from the system roadmap.

Benchmarking

While the reference model prescribed the language to be used in the systems/architectures to be assessed, the reference architecture stated the minimum (functional) requirements for the systems/ architectures. By standardising the description and also the ordering and delineation of system components and aspects, this approach also provided the benchmarking process with a high level of transparency and inherent comparability

IoT ARM

An ARM consists of two main parts: a **Reference model** and a **Reference Architecture**.

- A reference model describes the domain using a number of sub-models . The *domain model* of an architecture model captures the main concepts or entities in the domain in question, in this case IoT.

IoT ARM

An ARM consists of two main parts: a **Reference model** and a **Reference Architecture**.

- A reference model describes the domain using a number of sub-models . The *domain model* of an architecture model captures the main concepts or entities in the domain in question, in this case IoT.
- When these common language references are established, the domain model adds descriptions about the relationship between the concepts. These concepts and relationships serve the basis for the development of an *information model* because a working system needs to capture and process information about its main entities and their interactions.

IoT ARM

An ARM consists of two main parts: a **Reference model** and a **Reference Architecture**.

- A reference model describes the domain using a number of sub-models . The *domain model* of an architecture model captures the main concepts or entities in the domain in question, in this case IoT.
- When these common language references are established, the domain model adds descriptions about the relationship between the concepts. These concepts and relationships serve the basis for the development of an *information model* because a working system needs to capture and process information about its main entities and their interactions.
- A working system that captures and operates on the domain and information model contains concepts and entities of its own, and these need to be described in a separate model, the *functional model*.

IoT ARM

An ARM consists of two main parts: a **Reference model** and a **Reference Architecture**.

- A reference model describes the domain using a number of sub-models . The *domain model* of an architecture model captures the main concepts or entities in the domain in question, in this case IoT.
- When these common language references are established, the domain model adds descriptions about the relationship between the concepts. These concepts and relationships serve the basis for the development of an *information model* because a working system needs to capture and process information about its main entities and their interactions.
- A working system that captures and operates on the domain and information model contains concepts and entities of its own, and these need to be described in a separate model, the *functional model*.
- IoT system contain communicating entities, and therefore the corresponding *communication model* needs to capture the communication interactions of these entities.

Apart from the reference model, the other main component of an ARM is the Reference Architecture. A System Architecture is a communication tool for different stakeholders of the system.

- A Reference Architecture captures the essential parts of an architecture, such as design principles, guidelines, and required parts (such as entities), to monitor and interact with the physical world for the case of an IoT Reference Architecture.

Apart from the reference model, the other main component of an ARM is the Reference Architecture. A System Architecture is a communication tool for different stakeholders of the system.

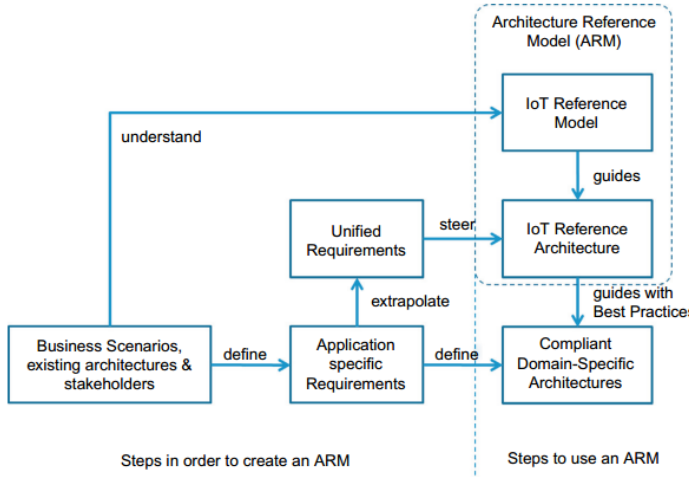
- A Reference Architecture captures the essential parts of an architecture, such as design principles, guidelines, and required parts (such as entities), to monitor and interact with the physical world for the case of an IoT Reference Architecture.
- Concrete architectures are instantiations of rather abstract and high-level Reference Architectures.

Apart from the reference model, the other main component of an ARM is the Reference Architecture. A System Architecture is a communication tool for different stakeholders of the system.

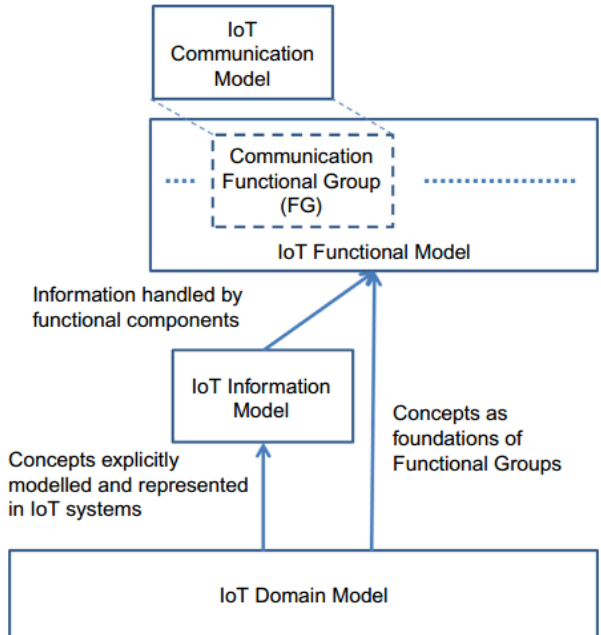
- A Reference Architecture captures the essential parts of an architecture, such as design principles, guidelines, and required parts (such as entities), to monitor and interact with the physical world for the case of an IoT Reference Architecture.
- Concrete architectures are instantiations of rather abstract and high-level Reference Architectures.
- A concrete architecture can be further elaborated and mapped into real world components by designing, building, engineering, and testing the different components of the actual system.

Apart from the reference model, the other main component of an ARM is the Reference Architecture. A System Architecture is a communication tool for different stakeholders of the system.

- A Reference Architecture captures the essential parts of an architecture, such as design principles, guidelines, and required parts (such as entities), to monitor and interact with the physical world for the case of an IoT Reference Architecture.
- Concrete architectures are instantiations of rather abstract and high-level Reference Architectures.
- A concrete architecture can be further elaborated and mapped into real world components by designing, building, engineering, and testing the different components of the actual system.
- The whole process is iterative, which means that the actual deployed system in the field provides invaluable feedback with respect to the design and engineering choices, current constraints of the system, and potential future opportunities that are fed back to the concrete architectures. The general essentials out of multiple concrete architectures can then be aggregated, and contribute to the evolution of the Reference Architecture.


Architecture Reference Model (ARM)
IoT Reference Model
understand
guides
Unified Requirements
steer
IoT Reference Architecture
extrapolate
guides with Best Practices
Business Scenarios, existing architectures & stakeholders
define
Application specific Requirements
define
Compliant Domain-Specific Architectures
Steps in order to create an ARM
Steps to use an ARM

IoT reference model



IoT domain model

A domain model defines the main concepts of a specific area of interest, in this case the IoT. These concepts are expected to remain unchanged over the course of time, even if the details of an ARM may undergo continuous transformation or evolution over time. The domain model captures the basic attributes of the main concepts and the relationship between these concepts. A domain model also serves as a tool for human communication between people working in the domain in question and between people who work across different domains.

IoT domain model

A domain model defines the main concepts of a specific area of interest, in this case the IoT. These concepts are expected to remain unchanged over the course of time, even if the details of an ARM may undergo continuous transformation or evolution over time. The domain model captures the basic attributes of the main concepts and the relationship between these concepts. A domain model also serves as a tool for human communication between people working in the domain in question and between people who work across different domains. The main concepts of ARM domain model are:

- *User and Physical Entity*: As interaction with the physical world is the key for the IoT. The first most fundamental interaction is between a human or an application with the physical world object or place. The physical interaction is the result of the intention of the human to achieve a certain goal (e.g. park the car).A Physical Entity, as the model shows, can potentially contain other physical entities (e.g building).

IoT domain model

A domain model defines the main concepts of a specific area of interest, in this case the IoT. These concepts are expected to remain unchanged over the course of time, even if the details of an ARM may undergo continuous transformation or evolution over time. The domain model captures the basic attributes of the main concepts and the relationship between these concepts. A domain model also serves as a tool for human communication between people working in the domain in question and between people who work across different domains. The main concepts of ARM domain model are:

- *User and Physical Entity*: As interaction with the physical world is the key for the IoT. The first most fundamental interaction is between a human or an application with the physical world object or place. The physical interaction is the result of the intention of the human to achieve a certain goal (e.g. park the car).A Physical Entity, as the model shows, can potentially contain other physical entities (e.g building).
- *Virtual Entity*: A Physical Entity is represented in the digital world as a Virtual Entity. A Virtual Entity can be a database entry, a geographical model (mainly for places), an image or avatar, or any other *Digital Artifact*. Each Virtual Entity also has a unique identifier for making it addressable among other Digital Artifacts.

- A Virtual Entity representation contains several attributes that correspond to the Physical Entity current state (e.g. the parking spot availability). The Virtual Entity representation and the Physical Entity actual state should be synchronized whenever a User operates on one or the other, if of course that is physically possible. There are cases that state synchronization can occur only one way.

- A Virtual Entity representation contains several attributes that correspond to the Physical Entity current state (e.g. the parking spot availability). The Virtual Entity representation and the Physical Entity actual state should be synchronized whenever a User operates on one or the other, if of course that is physically possible. There are cases that state synchronization can occur only one way.
- *Physical Artifact*:In order to monitor and interact with the Physical Entities through their corresponding virtual entities, the Physical Entities or their surrounding environment needs to be instrumented with certain kinds of Devices, or certain Devices need to be embedded/attached to the environment. The Devices are physical artifacts with which the physical and virtual worlds interact. For the IoT Domain Model, three kinds of Device types are the most important - Sensors, actuators and Tags.

- A Virtual Entity representation contains several attributes that correspond to the Physical Entity current state (e.g. the parking spot availability). The Virtual Entity representation and the Physical Entity actual state should be synchronized whenever a User operates on one or the other, if of course that is physically possible. There are cases that state synchronization can occur only one way.
- *Physical Artifact*:In order to monitor and interact with the Physical Entities through their corresponding virtual entities, the Physical Entities or their surrounding environment needs to be instrumented with certain kinds of Devices, or certain Devices need to be embedded/attached to the environment. The Devices are physical artifacts with which the physical and virtual worlds interact. For the IoT Domain Model, three kinds of Device types are the most important - Sensors, actuators and Tags.
- *Resources*:Resources are software components that provide data for, or are endpoints for, controlling Physical Entities. Resources can be of two types, on-Device resources and Network Resources.

- A Virtual Entity representation contains several attributes that correspond to the Physical Entity current state (e.g. the parking spot availability). The Virtual Entity representation and the Physical Entity actual state should be synchronized whenever a User operates on one or the other, if of course that is physically possible. There are cases that state synchronization can occur only one way.
- *Physical Artifact*:In order to monitor and interact with the Physical Entities through their corresponding virtual entities, the Physical Entities or their surrounding environment needs to be instrumented with certain kinds of Devices, or certain Devices need to be embedded/attached to the environment. The Devices are physical artifacts with which the physical and virtual worlds interact. For the IoT Domain Model, three kinds of Device types are the most important - Sensors, actuators and Tags.
- *Resources*:Resources are software components that provide data for, or are endpoints for, controlling Physical Entities. Resources can be of two types, on-Device resources and Network Resources.
- An on-Device Resource is typically hosted on the Device itself and provides information, or is the control point for the Physical Entities that the Device itself is attached to. The Network Resources are software components hosted somewhere in the network or cloud.

- A Virtual Entity is associated with potentially several Resources that provide information or control of the Physical Entity represented by this Virtual Entity.

- A Virtual Entity is associated with potentially several Resources that provide information or control of the Physical Entity represented by this Virtual Entity.
- Resources expose (monitor or control) functionality as Services with open and standardized interfaces, thus abstracting the potentially low-level implementation details of the resources.

- A Virtual Entity is associated with potentially several Resources that provide information or control of the Physical Entity represented by this Virtual Entity.
- Resources expose (monitor or control) functionality as Services with open and standardized interfaces, thus abstracting the potentially low-level implementation details of the resources.
- *Services* are therefore digital artifacts with which Users interact with Physical Entities through the Virtual Entities. Therefore, the Virtual Entities that are associated with Physical Entities instrumented with Devices that expose Resources are also associated with the corresponding resource Services. These associations are important to be maintained for lookup or discovery by the interested Users.

- A Virtual Entity is associated with potentially several Resources that provide information or control of the Physical Entity represented by this Virtual Entity.
- Resources expose (monitor or control) functionality as Services with open and standardized interfaces, thus abstracting the potentially low-level implementation details of the resources.
- *Services* are therefore digital artifacts with which Users interact with Physical Entities through the Virtual Entities. Therefore, the Virtual Entities that are associated with Physical Entities instrumented with Devices that expose Resources are also associated with the corresponding resource Services. These associations are important to be maintained for lookup or discovery by the interested Users.
- IoT Services can be classified into three main classes according to their level of abstraction: (i)Resource-Level Services (ii) Virtual Entity-Level Services (iii) Integrated Services .

Information Model

Information is defined as the enrichment of data (raw values without relevant or usable context) with the right context, so that queries about who, what, where, and when can be answered.

- On a high-level, the IoT Information Model maintains the necessary information about Virtual Entities and their properties or attributes.

Information Model

Information is defined as the enrichment of data (raw values without relevant or usable context) with the right context, so that queries about who, what, where, and when can be answered.

- On a high-level, the IoT Information Model maintains the necessary information about Virtual Entities and their properties or attributes.
- These properties/attributes can be static or dynamic and enter into the system in various forms, e.g. by manual data entry or reading a sensor attached to the Virtual Entity. Virtual Entity attributes can also be digital synchronized copies of the state of an actuator.

Information Model

Information is defined as the enrichment of data (raw values without relevant or usable context) with the right context, so that queries about who, what, where, and when can be answered.

- On a high-level, the IoT Information Model maintains the necessary information about Virtual Entities and their properties or attributes.
- These properties/attributes can be static or dynamic and enter into the system in various forms, e.g. by manual data entry or reading a sensor attached to the Virtual Entity. Virtual Entity attributes can also be digital synchronized copies of the state of an actuator.
- In the presentation of the high-level IoT information model, we omit the attributes that are not updated by an IoT Device (sensor, tag) or the attributes that do not affect any IoT Device (actuator, tag), with the exception of essential attributes such as names and identifiers

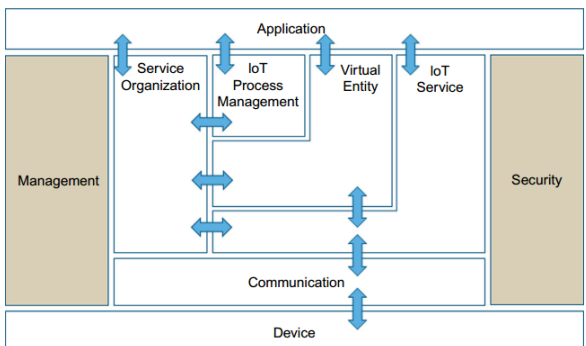
Information Model

Information is defined as the enrichment of data (raw values without relevant or usable context) with the right context, so that queries about who, what, where, and when can be answered.

- On a high-level, the IoT Information Model maintains the necessary information about Virtual Entities and their properties or attributes.
- These properties/attributes can be static or dynamic and enter into the system in various forms, e.g. by manual data entry or reading a sensor attached to the Virtual Entity. Virtual Entity attributes can also be digital synchronized copies of the state of an actuator.
- In the presentation of the high-level IoT information model, we omit the attributes that are not updated by an IoT Device (sensor, tag) or the attributes that do not affect any IoT Device (actuator, tag), with the exception of essential attributes such as names and identifiers
- A Virtual Entity object contains simple attributes/properties: (a) entityType to denote the type of entity, such as a human, car, or room (the entity type can be a reference to concepts of a domain ontology, e.g. a car ontology); (b) a unique identifier; and (c) zero or more complex attributes of the class Attributes.

Functional model

In the IoT-ARM project, *Functional Decomposition* (FD) refers to the process by which the different *Functional Components* (FC) that make up the IoT ARM are identified and related to one another. The main purpose of Functional Decomposition is, on the one hand, to break up the complexity of a system compliant to the IoT ARM in smaller and more manageable parts, and to understand and illustrate their relationship on the other hand.



The most important functional group are:

- *Device functional group:* The Device FG contains all the possible functionality hosted by the physical Devices that are used for instrumenting the Physical Entities.

The most important functional group are:

- *Device functional group:* The Device FG contains all the possible functionality hosted by the physical Devices that are used for instrumenting the Physical Entities.
- *Communication functional group* The Communication FG abstracts all the possible communication mechanisms used by the relevant Devices in an actual system in order to transfer information to the digital world components or other Devices.

The most important functional group are:

- *Device functional group:* The Device FG contains all the possible functionality hosted by the physical Devices that are used for instrumenting the Physical Entities.
- *Communication functional group* The Communication FG abstracts all the possible communication mechanisms used by the relevant Devices in an actual system in order to transfer information to the digital world components or other Devices.
- *IoT Service functional group*: The IoT Service FG corresponds mainly to the Service class from the IoT Domain Model, and contains single IoT Services exposed by Resources hosted on Devices or in the Network (e.g. processing or storage Resources).

The most important functional group are:

- *Device functional group:* The Device FG contains all the possible functionality hosted by the physical Devices that are used for instrumenting the Physical Entities.
- *Communication functional group* The Communication FG abstracts all the possible communication mechanisms used by the relevant Devices in an actual system in order to transfer information to the digital world components or other Devices.
- *IoT Service functional group*: The IoT Service FG corresponds mainly to the Service class from the IoT Domain Model, and contains single IoT Services exposed by Resources hosted on Devices or in the Network (e.g. processing or storage Resources).
- *Virtual Entity functional group* The Virtual Entity FG corresponds to the Virtual Entity class in the IoT Domain Model, and contains the necessary functionality to manage associations between Virtual Entities with themselves as well as associations between Virtual Entities and related IoT Services, i.e. the Association objects for the IoT Information Model.

- *IoT Service Organization functional group* The purpose of the IoT Service Organization FG is to host all functional components that support the composition and orchestration of IoT and Virtual Entity services.

- *IoT Service Organization functional group* The purpose of the IoT Service Organization FG is to host all functional components that support the composition and orchestration of IoT and Virtual Entity services.
- *IoT Process Management functional group* The IoT Process Management FG is a collection of functionalities that allows smooth integration of IoT-related services (IoT Services, Virtual Entity Services, Composed Services) with the Enterprise (Business) Processes

- *IoT Service Organization functional group* The purpose of the IoT Service Organization FG is to host all functional components that support the composition and orchestration of IoT and Virtual Entity services.
- *IoT Process Management functional group* The IoT Process Management FG is a collection of functionalities that allows smooth integration of IoT-related services (IoT Services, Virtual Entity Services, Composed Services) with the Enterprise (Business) Processes
- *Management functional group* The Management FG includes the necessary functions for enabling fault and performance monitoring of the system, configuration for enabling the system to be flexible to changing User demands, and accounting for enabling subsequent billing for the usage of the system.

- *IoT Service Organization functional group* The purpose of the IoT Service Organization FG is to host all functional components that support the composition and orchestration of IoT and Virtual Entity services.
- *IoT Process Management functional group* The IoT Process Management FG is a collection of functionalities that allows smooth integration of IoT-related services (IoT Services, Virtual Entity Services, Composed Services) with the Enterprise (Business) Processes
- *Management functional group* The Management FG includes the necessary functions for enabling fault and performance monitoring of the system, configuration for enabling the system to be flexible to changing User demands, and accounting for enabling subsequent billing for the usage of the system.
- *Security functional group* The SecurityFG contains components for Authentication of Users (Applications,Humans), Authorization of access to Services by Users, secure communication (ensuring integrity and confidentiality of messages) between entities of the system such as Devices, Services, Applications, and last but not least,assurance of privacy of sensitive information relating to Human Users

Communication Model

The communication model for an IoT Reference Model consists of the identification of the endpoints of interactions, traffic patterns (e.g. unicast vs. multicast), and general properties of the underlying technologies used for enabling such interactions.

Communication Model

The communication model for an IoT Reference Model consists of the identification of the endpoints of interactions, traffic patterns (e.g. unicast vs. multicast), and general properties of the underlying technologies used for enabling such interactions.

- The potential communicating endpoints or entities are the Users, Resources, and Devices from the IoT Domain Model. Users include Human Users and Active Digital Artifacts.

Communication Model

The communication model for an IoT Reference Model consists of the identification of the endpoints of interactions, traffic patterns (e.g. unicast vs. multicast), and general properties of the underlying technologies used for enabling such interactions.

- The potential communicating endpoints or entities are the Users, Resources, and Devices from the IoT Domain Model. Users include Human Users and Active Digital Artifacts.
- *User-Service / Service-Service Interactions*: , the IoT Domain Model entities involved in this interaction are mainly two: User and Service. If a Service is constrained, or if it needs to provide access to constrained Users, it must be accessible with constrained protocols (e.g., 6LoWPAN, UDP, CoAP, etc.). Finally, when the two elements belong to different sub-networks, gateway(s) and/or proxy(ies) must be deployed for ensuring successful end-to-end communication.

Communication Model

The communication model for an IoT Reference Model consists of the identification of the endpoints of interactions, traffic patterns (e.g. unicast vs. multicast), and general properties of the underlying technologies used for enabling such interactions.

- The potential communicating endpoints or entities are the Users, Resources, and Devices from the IoT Domain Model. Users include Human Users and Active Digital Artifacts.
- *User-Service / Service-Service Interactions*: , the IoT Domain Model entities involved in this interaction are mainly two: User and Service. If a Service is constrained, or if it needs to provide access to constrained Users, it must be accessible with constrained protocols (e.g., 6LoWPAN, UDP, CoAP, etc.). Finally, when the two elements belong to different sub-networks, gateway(s) and/or proxy(ies) must be deployed for ensuring successful end-to-end communication.
- *Service / Resource / Device Interactions*: The complexity of this interaction is due to variety of different properties that a Device can have; in particular, a Device can be as simple and limited as a Tag and as complex and powerful as a server.

Safety, privacy, trust, security model

The fact that Human Users are part of the system that could potentially harm humans if malfunctioning, or expose private information, motivates the Safety and Privacy needs for the IoT Reference Model and Architecture.

Safety, privacy, trust, security model

The fact that Human Users are part of the system that could potentially harm humans if malfunctioning, or expose private information, motivates the Safety and Privacy needs for the IoT Reference Model and Architecture.

- **Safety**: The IoT Reference Model can only provide IoT-related guidelines for ensuring a safe system to the extent possible and controllable by a system designer. A system designer of such critical systems typically follows an iterative process with two steps: (a) identification of hazards followed by, (b) the mitigation plan.

Safety, privacy, trust, security model

The fact that Human Users are part of the system that could potentially harm humans if malfunctioning, or expose private information, motivates the Safety and Privacy needs for the IoT Reference Model and Architecture.

- **Safety**: The IoT Reference Model can only provide IoT-related guidelines for ensuring a safe system to the extent possible and controllable by a system designer. A system designer of such critical systems typically follows an iterative process with two steps: (a) identification of hazards followed by, (b) the mitigation plan.
- **Privacy**: he IoT-A Privacy Model depends on the following functional components: Identity Management, Authentication, Authorization, and Trust & Reputation. The Trust and Reputation functional component maintain the static or dynamic trust relationships between interacting entities.

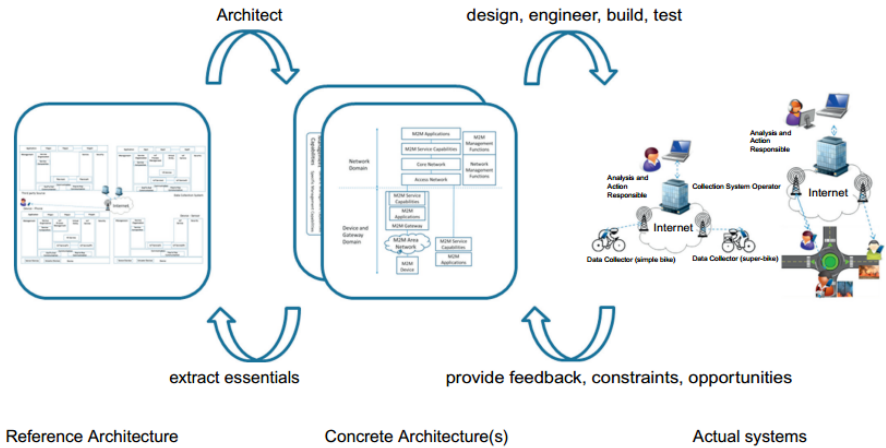
Safety, privacy, trust, security model

The fact that Human Users are part of the system that could potentially harm humans if malfunctioning, or expose private information, motivates the Safety and Privacy needs for the IoT Reference Model and Architecture.

- **Safety**: The IoT Reference Model can only provide IoT-related guidelines for ensuring a safe system to the extent possible and controllable by a system designer. A system designer of such critical systems typically follows an iterative process with two steps: (a) identification of hazards followed by, (b) the mitigation plan.
- **Privacy**: he IoT-A Privacy Model depends on the following functional components: Identity Management, Authentication, Authorization, and Trust & Reputation. The Trust and Reputation functional component maintain the static or dynamic trust relationships between interacting entities.
- **Security** The Security Model for IoT consists of communication security that focuses mostly on the confidentiality and integrity protection of interacting entities and functional components such as Identity Management, Authentication, Authorization, and Trust & Reputation.

IoT Reference Architecture

The Reference Architecture is a starting point for generating concrete architectures and actual systems. A Reference Architecture, serves as a guide for one or more concrete system architects.



The Reference Architecture is presented as set of architectural views. Views are useful for reducing the complexity of the Reference Architecture blueprints by addressing groups of concerns one group at a time.

The Reference Architecture is presented as set of architectural views. Views are useful for reducing the complexity of the Reference Architecture blueprints by addressing groups of concerns one group at a time. In order to address the concerns of mainly the concrete IoT architect, and secondly the concerns of most of the above stakeholders following views are chosen.

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

The Reference Architecture is presented as set of architectural views. Views are useful for reducing the complexity of the Reference Architecture blueprints by addressing groups of concerns one group at a time. In order to address the concerns of mainly the concrete IoT architect, and secondly the concerns of most of the above stakeholders following views are chosen.

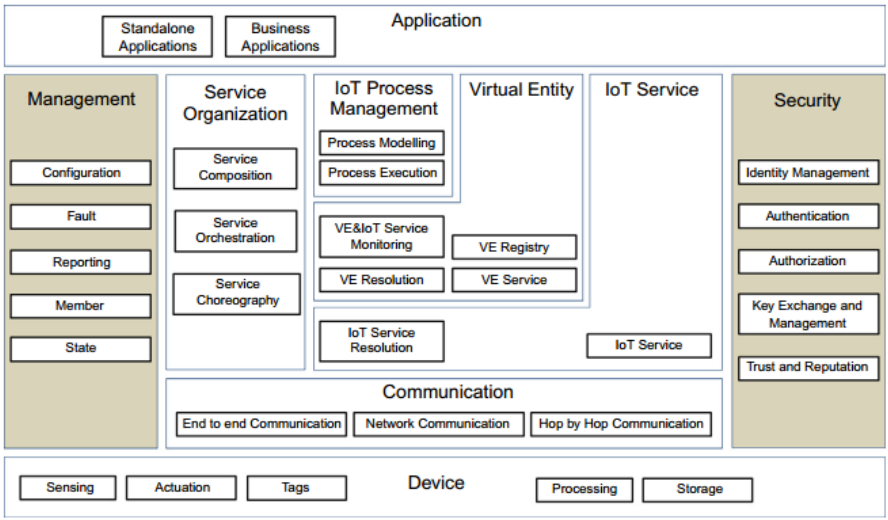
- **Functional View**: Description of what the system does, and its main functions.
- **Information View**: Description of the data and information that the system handles.

The Reference Architecture is presented as set of architectural views. Views are useful for reducing the complexity of the Reference Architecture blueprints by addressing groups of concerns one group at a time. In order to address the concerns of mainly the concrete IoT architect, and secondly the concerns of most of the above stakeholders following views are chosen.

- **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

It consists of the Functional Groups (FGs) presented earlier in the IoT Functional Model, each of which includes a set of Functional Components (FCs).



Information View

The information view consists of (a) the description of the information handled in the IoT System, and (b) the way this information is handled in the system;

Information View

The information view consists of (a) the description of the information handled in the IoT System, and (b) the way this information is handled in the system; The pieces of information handled by an IoT system complying to an ARM such as the IoT-A are the following:

- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model (attributes that have values and metadata such as the temperature of a room).

Information View

The information view consists of (a) the description of the information handled in the IoT System, and (b) the way this information is handled in the system; The pieces of information handled by an IoT system complying to an ARM such as the IoT-A are the following:

- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model (attributes that have values and metadata such as the temperature of a room).
- IoT Service Descriptions, which contain associated Resources, interface descriptions, etc. IoT Service output itself is another important part of information generated by an IoT system.

Information View

The information view consists of (a) the description of the information handled in the IoT System, and (b) the way this information is handled in the system; The pieces of information handled by an IoT system complying to an ARM such as the IoT-A are the following:

- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model (attributes that have values and metadata such as the temperature of a room).
- IoT Service Descriptions, which contain associated Resources, interface descriptions, etc. IoT Service output itself is another important part of information generated by an IoT system.
- Resource Descriptions, which contain the type of resource (e.g. sensor), identity, associated Services, and Devices. Device Descriptions such as device capabilities (e.g. sensors, radios)

Information View

The information view consists of (a) the description of the information handled in the IoT System, and (b) the way this information is handled in the system; The pieces of information handled by an IoT system complying to an ARM such as the IoT-A are the following:

- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model (attributes that have values and metadata such as the temperature of a room).
- IoT Service Descriptions, which contain associated Resources, interface descriptions, etc. IoT Service output itself is another important part of information generated by an IoT system.
- Resource Descriptions, which contain the type of resource (e.g. sensor), identity, associated Services, and Devices. Device Descriptions such as device capabilities (e.g. sensors, radios)
- IoT Business Process Model, which describes the steps of a business process utilizing other IoT-related services (IoT, Virtual Entity, Composed Services).

Information View

The information view consists of (a) the description of the information handled in the IoT System, and (b) the way this information is handled in the system; The pieces of information handled by an IoT system complying to an ARM such as the IoT-A are the following:

- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model (attributes that have values and metadata such as the temperature of a room).
- IoT Service Descriptions, which contain associated Resources, interface descriptions, etc. IoT Service output itself is another important part of information generated by an IoT system.
- Resource Descriptions, which contain the type of resource (e.g. sensor), identity, associated Services, and Devices. Device Descriptions such as device capabilities (e.g. sensors, radios)
- IoT Business Process Model, which describes the steps of a business process utilizing other IoT-related services (IoT, Virtual Entity, Composed Services).
- Management information such as state information from operational FCs used for fault/performance purposes, configuration snapshots, reports, membership information, etc.

Information Handling

The presentation of information handling in an IoT system assumes that FCs exchange and process information. The exchange of information between FCs follows the interaction patterns below

- **Push:** An FC A pushes the information to another FC B provided that the contact information of the component B is already configured in component A, and component B listens for such information pushes.

Information Handling

The presentation of information handling in an IoT system assumes that FCs exchange and process information. The exchange of information between FCs follows the interaction patterns below

- **Push:** An FC A pushes the information to another FC B provided that the contact information of the component B is already configured in component A, and component B listens for such information pushes.
- **Request/Response**: An FC A sends a request to another FC B and receives a response from B after A serves the request.

Information Handling

The presentation of information handling in an IoT system assumes that FCs exchange and process information. The exchange of information between FCs follows the interaction patterns below

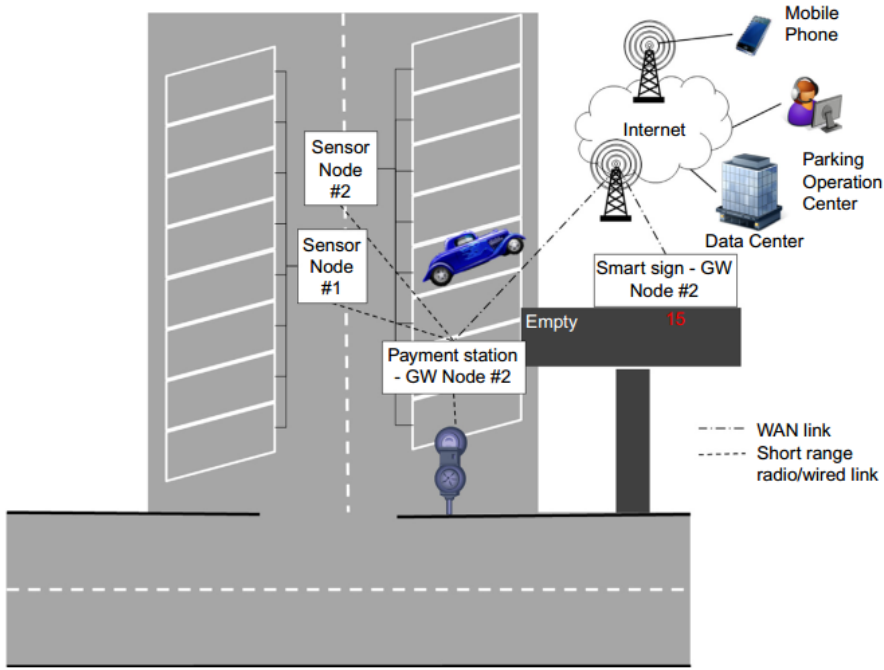
- **Push:** An FC A pushes the information to another FC B provided that the contact information of the component B is already configured in component A, and component B listens for such information pushes.
- **Request/Response**: An FC A sends a request to another FC B and receives a response from B after A serves the request.
- **Subscribe/Notify**: Multiple subscriber components (SA, SB) can subscribe for information to a component C, and C will notify the relevant subscribers when the requested information is ready. This is typically an asynchronous information request after which each subscriber can perform other tasks.

Information Handling

The presentation of information handling in an IoT system assumes that FCs exchange and process information. The exchange of information between FCs follows the interaction patterns below

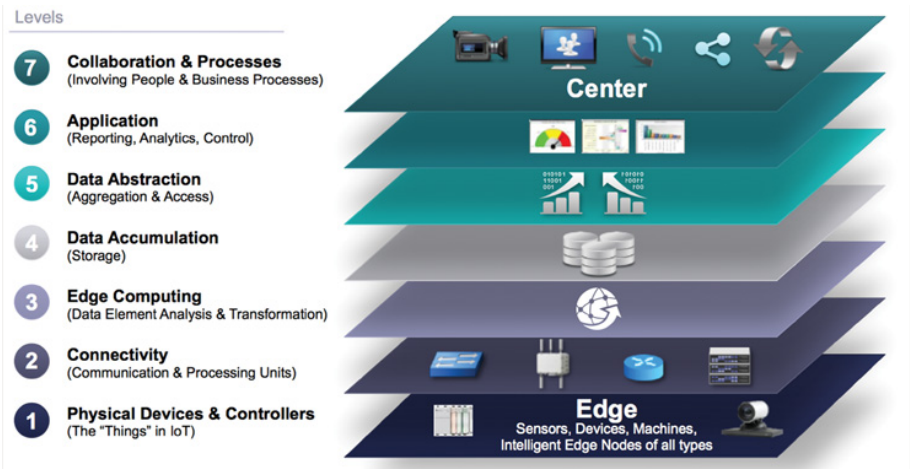
- **Push:** An FC A pushes the information to another FC B provided that the contact information of the component B is already configured in component A, and component B listens for such information pushes.
- **Request/Response**: An FC A sends a request to another FC B and receives a response from B after A serves the request.
- **Subscribe/Notify**: Multiple subscriber components (SA, SB) can subscribe for information to a component C, and C will notify the relevant subscribers when the requested information is ready. This is typically an asynchronous information request after which each subscriber can perform other tasks.
- **Publish/Subscribe**: In the Publish/Subscribe (also known as a Pub/Sub pattern), there is a third component called the broker B, which mediates subscription and publications between subscribers (information consumers) and publishers (or information producers)

Deployment and Operational View



The IoT World Forum (IoTWF) Standardized Architecture

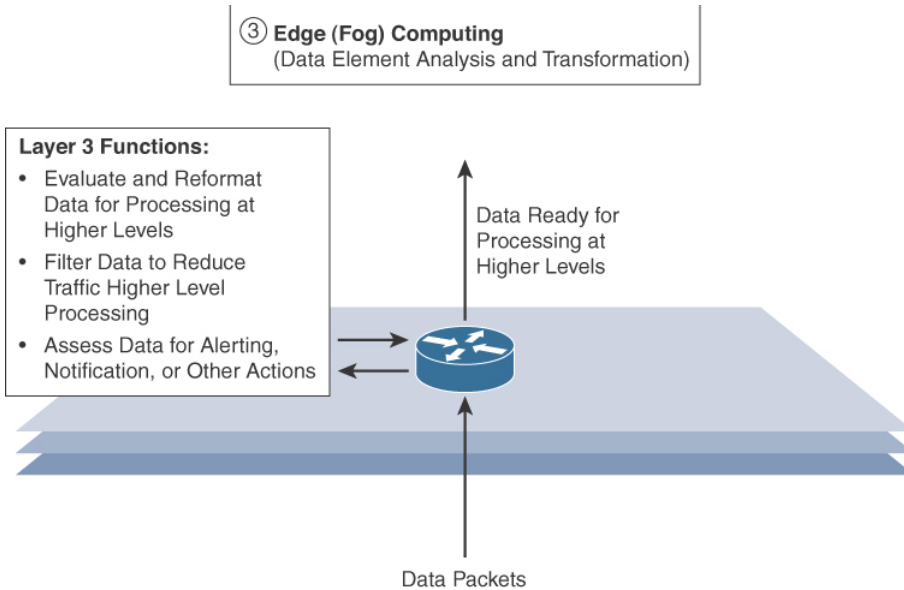
In 2014 the IoTWF architectural committee (led by Cisco, IBM, Rockwell Automation, and others) published a seven-layer IoT architectural reference model. While various IoT reference models exist, the one put forth by the IoT World Forum offers a clean, simplified perspective on IoT and includes edge computing, data storage, and access.



② **Connectivity**
(Communication and Processing Units)

Layer 2 Functions:

- Communications Between Layer 1 Devices
- Reliable Delivery of Information Across the Network
- Switching and Routing
- Translation Between Protocols
- Network Level Security


③ Edge (Fog) Computing
(Data Element Analysis and Transformation)
Layer 3 Functions:
• Evaluate and Reformat Data for Processing at Higher Levels
• Filter Data to Reduce Traffic Higher Level Processing
• Assess Data for Alerting, Notification, or Other Actions
Data Ready for Processing at Higher Levels
Data Packets

IoT Reference Model Layer	Functions
Layer 4: Data accumulation layer	Captures data and stores it so it is usable by applications when necessary. Converts event-based data to query-based processing.
Layer 5: Data abstraction layer	Reconciles multiple data formats and ensures consistent semantics from various sources. Confirms that the data set is complete and consolidates data into one place or multiple data stores using virtualization.
Layer 6: Applications layer	Interprets data using software applications. Applications may monitor, control, and provide reports based on the analysis of the data.
Layer 7: Collaboration and processes layer	Consumes and shares the application information. Collaborating on and communicating IoT information often requires multiple steps, and it is what makes IoT useful. This layer can change business processes and delivers the benefits of IoT.