Horizontal Architecture Approach for IoT Systems

A systemic implementation of IoT ecosystems is usually based on a layered architecture style, which can range from data acquisition layer (i.e., Perception layer) at the bottom, to application layer at the top as shown in Fig.2.4. In this kind of architecture, layers from the bottom usually contribute to devices integration and data capturing, while layers from the top are responsible for data distribution and utilization by IoT applications.

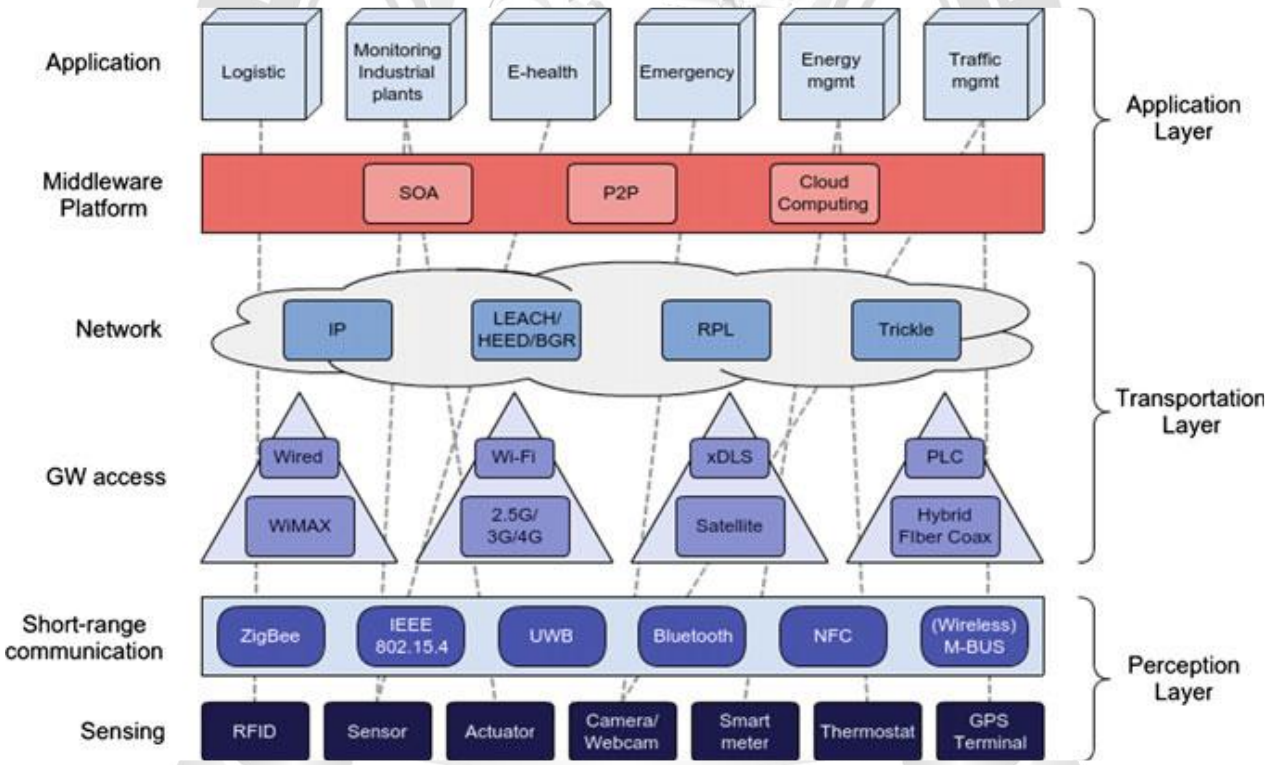

Fig.2.4 IoT systems architecture overiew

[Ref: L.A. Amaral et al. Middleware Technology for IoT Systems: Challenges and Perspectives Toward 5G,2016]

The common architectural approach largely used by current IoT systems is a vertical strategy where each application is built on its proprietary ICT infrastructure and dedicated physical devices. In this approach, similar applications do not share any feature of the IoT infrastructure (e.g., managing services and network), resulting in unnecessary redundancy and increase of costs (i.e., financial and computational costs). Totally vertical approach should be overtaken by a more flexible and horizontal approach, where a common operational platform is able to manage the network and the application services, and abstracts across a diverse range of data sources to enable applications to work properly.

As shown by Fig. 2.4, through a horizontal IoT middleware approach, applications no longer work in isolation, and share infrastructure, environment, and network elements by means of a common service platform (i.e., the IoT middleware platform) that arranges on behalf of them. Figure 2.4 also shows the three different layers (or interaction phases) in which the cyber-physical world interactions should take place. Specifically, they are: (i) perception layer, (ii) transportation layer, and (iii) application layer (i.e., process, management and utilization phases). Each layer is characterized by different interacting technologies and protocols and has different purposes and functions as discussed below:

Perception layer: it refers to procedures for sensing the physical environment, collecting real-time data, and reconstructing a general perception of it in the virtual world (i.e., in the system logical domain). Technologies such as RFID and sensors provide identification of physical objects and sensing of physical parameters. While technologies such as IEEE 802.15.4 and Bluetooth are responsible for data collecting.

Transportation layer: it includes mechanisms to deliver the collected data to applications and to different external servers. Methods are therefore required for accessing the network through gateways and heterogeneous technologies (e.g., wired, wireless, satellite), as well as for addressing and/or routing.

Applications layer: it deals with processing and analyzing information flows, forwarding data to applications and services, and providing feedbacks to control applications. In addition, it is responsible for critical functions such as device discovery, device management, data filtering, data aggregation, semantic analysis, and information utilization/distribution. Indeed, these

functions are essential for IoT ecosystems, and as such, must be handled by an IoT middleware platform.

The first step toward the Internet of Things is the collection of information about the physical environment (e.g., temperature, humidity, brightness) or about objects (e.g., identity, state, energy level). Data acquisition is encompassed by using different sensing technologies attached to sensors, cameras, GPS terminals, while data collection is generally accomplished by short range communications, which could be open source standard solutions (e.g., Bluetooth, ZigBee, Dash7, Wireless M-BUS) as well as proprietary solutions (e.g., Z-Wave, ANT).

Once data is gathered through sensing technologies, it needs to be transmitted across the network in order applications can be able to consume the data. Heterogeneous communication technologies form the backbone to access the network.

When data arrives in the application layer, information flows are processed and then forwarded to applications. The IoT middleware layer covers a fundamental role for managing the above operations. It is crucial for hiding the heterogeneity of hardware, software, data formats, technologies and communication protocols that characterize an IoT ecosystem. Besides, it is responsible for abstracting all the features of objects, network, and services, and for offering a loose coupling of components. Additional features of this layer are service discovery and service composition.

SOA based IoT Middleware

A service-oriented architecture (SOA) is a set of principles and methodologies for designing and developing software in the form of interoperable services, usually over the Internet. Services comprise unassociated, loosely coupled units of functionality that have no calls to each other embedded in them.

Service-oriented architecture consists of components which are implemented as independent services which can be dynamically bonded and orchestrated, and which possess loosely coupled configurations, while the communication between them uses messages. Orchestrating means a process which predefines an order of calling the services (in sequences and in parallel) and the data and message exchanges.

In IoT ecosystems, computation, storage, data management, and communication services are intended to be high ubiquitous and distributed. Furthermore, the entities of the environment (people, things/objects, platforms, and surrounding spaces) are intended to create to IoT applications a highly decentralized common pool of resources, which must be interconnected by a dynamic network of networks.

The smart integration of both intended services and entities represents the real ecosystem of the IoT. In this context, middleware for IoT is considered an important building block for the provision of IoT services, which are extremely desired to be highly pervasive and distributed. Indeed, middleware is an IoT platform intended to be a service of services to IoT ecosystems.

The notion of service-based IoT systems has been realized according to the principles of SOA and ROA (Resource-Oriented Architecture) architecture styles, which increasingly coexist in the IoT ecosystem since ROA allows the deployment of lightweight SOA-based communication mechanisms embedded into resource constrained IoT devices. SOA-based techniques provide to IoT applications a uniform and structured abstraction of services for communication with IoT devices. On the other hand, ROA-based approaches realize the necessary requirements to make the devices (things) addressable, searchable, controllable, and accessible to IoT applications through the Web.

IoT middleware is a software layer or a set of sub-layers interposed between technological (perception and transportation layers) and application layers as shown in fig.2.5. The middleware's ability to hide the details of different technologies is fundamental to exempt the programmer from issues that are not directly pertinent to her/his focus, which is the development of specific applications enabled by IoT infrastructures. In this way, IoT middleware has received much attention in the last years due to its major role of simplify the development of application and the integration of devices.

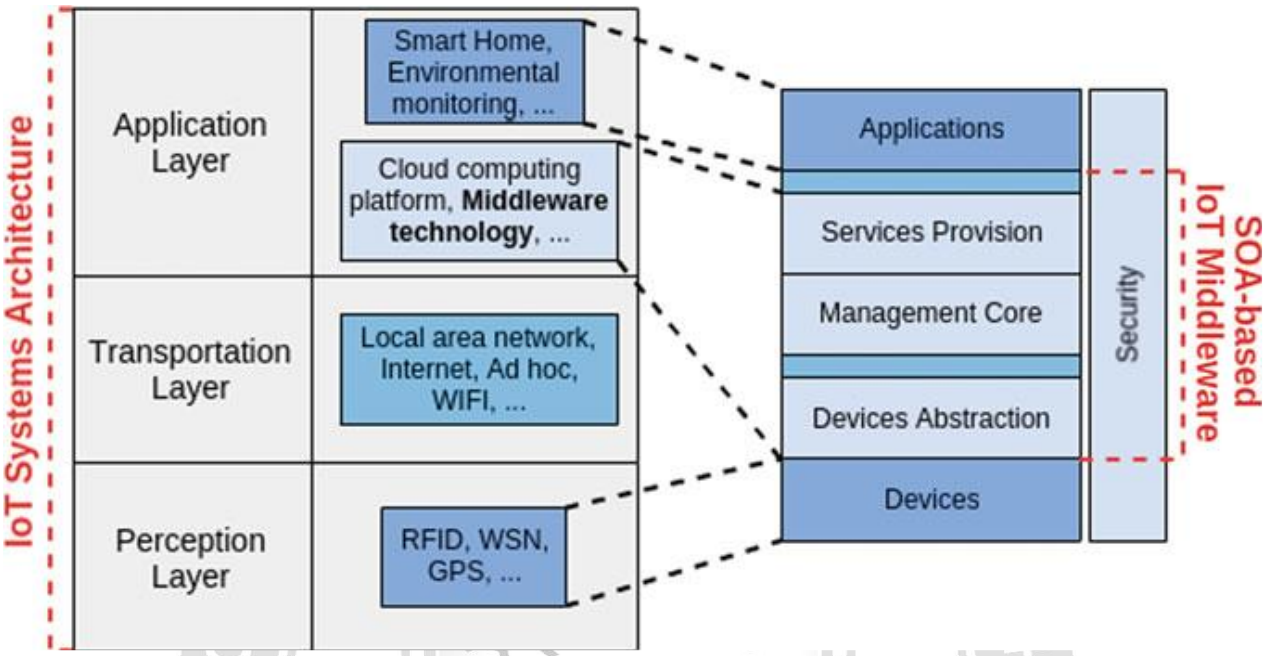

Fig.2.5 SOA-based IoT middleware architecture

[Ref: L.A. Amaral et al. Middleware Technology for IoT Systems: Challenges and Perspectives Toward 5G,2016]

Many of the system architectures proposed for IoT middleware comply with the SOA approach. A SOA structure for IoT middleware is illustrated in Fig. 2.5. According to this structure, the applications layer allows end users to request information services and to interact with the middleware.

The devices layer can be composed of any IoT device which can connect to the middleware to provide services based on its features/resources. The devices abstraction layer can be embedded into both devices and middleware. Each service in the services provision layer is composed of one or more services from the devices. The devices function is abstracted into services by the devices abstraction layer and provided by the middleware through the services provision layer.

The applications should use an API from the services provision layer to consume the provided services. All the processing activity is generated in the management core layer also

called middleware core. The security layer must ensure security in all exchanged and stored data, since the middleware architecture enables some vulnerability points that can be explored by security threats.