Towards a definition of the Internet of Things (IoT)

Issue 1 – Published 13 MAY 2015

Towards a Definition of the Internet of Things (IoT)

What the Internet of Things is

This document gives an all-inclusive definition of IoT that ranges from small localized systems constrained to a specific location to a large global system that is distributed and composed of complex systems. The document also provides an overview of the IoT's basic architectural requirements.

Telecom Italia S.p.A.
Authored by: Roberto Minerva, Abyi Biru, Domenico Rotondi

Many thanks to Daniel W. Engels, PhD.

This work was carried out under an internship program in Telecom Italia of the specializing master in Future Broadband Networks of Politecnico di Torino.

Table of Contents

List of Figures

List of Tables

1. Goals and Purpose of this Document

Internet of Things, IoT, is an application domain that integrates different technological and social fields, and these are summarized in **Figure 1.** Technological and social aspects related to IoT

. Despite the diversity of research on IoT, its definition remains fuzzy. We'd like to address this challenge, because having a sound definition that addresses all the IoT's features can facilitate a better understanding of the subject, lead to further research and advance our understanding of this emerging concept.

This document aims to give an all-inclusive definition of IoT that ranges from small localized systems to a large global system that is distributed and made of complex systems. The document also provides an overview of the IoT's basic architectural requirements.

This document directly refers to the sources and it extracts integral parts of original documents in order to preserve the ideas and results of original works. We believe that this work will be enhanced through contributions by people working in the area of IoT. Thus, we welcome comments on or contributions to any section of the document.

This document will be shared via the IEEE IoT Initiative Web portal as a living document, possibly as an IoT Wiki. We should point out that Chapter 5 will be the authors' major contribution to this work, as it offers a definition of IoT inferred from the preceding chapters. But it will also be the chapter most in need of future revision because IoT is morphing so quickly. We have provided a few, simple criteria to apply in order to verify if a specific system is an IoT related system. And we have introduced the notion of a definition that can be scaled to encompass small wireless sensor networks as well as large complex systems.

Generally speaking, the IoT covers many areas (see **Figure 1.** Technological and social aspects related to IoT

) ranging from enabling technologies and components to several mechanisms to effectively integrate these low-level components. Software is then a discriminant factor for IoT systems. IoT operating systems are designed to run on small-scale components in the most efficient way possible, while at the same time providing basic functionalities to simplify and support the global IoT system in its objectives and purposes. Middleware, programmability – in terms of application programming interfaces (APIs) – and data management seem to be key factors for building a successful system in the IoT realm. Management capabilities are needed in order to properly handle systems that can potentially grow up to millions of different components. In this context, self-management and self-optimization of each individual component and/or subsystem maybe strong requirements. In other words, autonomics behaviors could become the norm in large and complex IoT systems. Data security and privacy will play an important role in IoT deployments. Because IoT systems will produce and deal with personally identifiable information, data security and privacy will be critical from the very beginning. Services and applications will be built on top of this powerful and secure platform to satisfy business needs. So many applications are envisioned as well as generic and reusable services. This outcome will require new, viable business models for IoT and its related ecosystems of stakeholders. Finally, IoT can have an impact on people and the society they live in, and so it must be conceived and conducted within the constraints and regulations of each country.

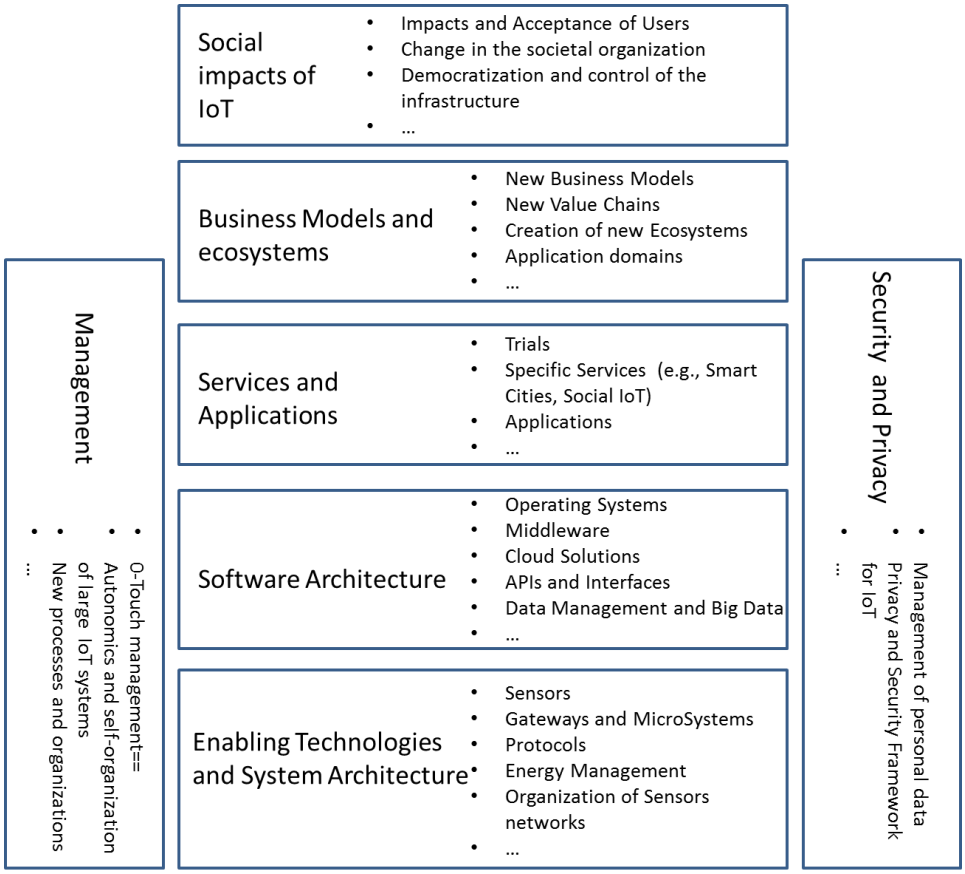


Figure 1. Technological and social aspects related to IoT

2. State of the Art

2.1 Introduction

This chapter will address state of the art definitions and architectural models for IoT offered by standardization organizations, IoT projects, academia, national initiatives, white papers, books and related industries. While we have tried to be thorough, our effort cannot be said to be exhaustive, given the proliferation of interest in the subject.

Different definitions and architectural models for IoT reflect different perspectives and support different business interests. Analyzing these different definitions and architectures can help illuminate their strengths and weaknesses. Still, as stated earlier, we see a need to have a common and non-biased definition that effectively encompasses the expansive nature of the subject. We believe the following review of different definitions and architectural models will serve us in composing that more universal definition.

2.2 Historical Background *[The authors thank Prof. Daniel Engels for this chapter.]*

Radio-frequency identification, or RFID, may be a crucial technology for IoT. The roots of RFID technology can be traced back to World War II. The Germans, Japanese, Americans and British all used radar—discovered in 1935 by Scottish physicist Sir Robert Alexander Watson-Watt—to warn of approaching enemy planes while they were still miles away. But there was no way to

identify which planes belonged to the enemy and which were a country's own pilots returning from a mission.

The Germans discovered that if pilots rolled their planes as they returned to base, it would change the radio signal reflected back to radar systems. This crude method alerted the radar crew on the ground that these were German planes and not allied aircraft. Essentially, this was the first passive RFID system.

Under Watson-Watt, who headed a secret project, the British developed the first active "identify friend or foe" (IFF) system. When a British plane received British radar signals, it would broadcast a signal back that identified the aircraft as friendly. RFID works on this same basic concept. A signal is sent to a transponder, which wakes up and either reflects back a signal (passive system) or broadcasts a signal (active system).

Advances in radar and radio-frequency (RF) communications systems continued through the 1950s and 1960s. Scientists and academics in the United States (U.S.), Europe and Japan explored how RF energy could be used to identify objects remotely. Companies began commercializing anti-theft systems that used radio waves to determine whether an item had been paid for or not. Electronic article surveillance tags, for instance, which are still used in packaging today, have a 1-bit tag. The bit is either on or off. If someone pays for the item, the bit is turned off, and a person can leave the store. But if the person doesn't pay and tries to walk out of the store, automated readers at the door detect the tag and sound an alarm.

Mario W. Cardullo claims to have received the first U.S. patent for an active RFID tag with rewritable memory on January 23, 1973. That same year, Charles Walton, a California entrepreneur, received a patent for a passive transponder used to unlock a door without a key. In the latter application, a card with an embedded transponder communicated a signal to a reader near the door. When the reader detected a valid identity number stored within the RFID tag, the reader unlocked the door. Walton licensed the technology to Schlage, a lock maker, and other companies.

The U.S. government was also working on RFID systems. In the 1970s, Los Alamos National Laboratory was asked by the U.S. Department of Energy (U.S. DOE) to develop a system for tracking nuclear materials. A group of scientists devised the concept of putting a transponder in a truck and readers at the gates of secure facilities. The gate antenna would wake up the transponder in the truck, which would respond with an ID and, potentially, other data, such as the driver's ID. This system was commercialized in the mid-1980s when the Los Alamos scientists who worked on the project left to form a company to develop automated toll payment systems. These systems have become widely used on roads, bridges and tunnels around the world.

At the request of the U.S. Department of Agriculture, Los Alamos also developed a passive RFID tag to track cows and doses of hormones and medicines they'd received. It was difficult to ensure that each cow got the right dosage and wasn't given two doses accidentally. Los Alamos came up with a passive RFID system that used UHF radio waves. The device drew energy from the reader and simply reflected back a modulated signal to the reader using a technique known as backscatter.

Later, companies developed a low-frequency (125 kHz) system, featuring smaller transponders. A transponder encapsulated in glass could be injected under a cow's skin. This system is still

used in cows around the world today. Low-frequency transponders were also put in cards and used to control access to buildings.

Over time, companies commercialized 125 kHz systems and then moved up the radio spectrum to a high frequency band (13.56 MHz), which was unregulated and unused in most parts of the world. High frequency RF offered greater range and faster data transfer rates. Companies, particularly those in Europe, began using it to track reusable containers and other assets. Today, 13.56 MHz RFID systems are used for access control, payment systems (e.g., Mobile Speedpass) and contactless smart cards. They're also used in anti-theft devices in cars. A reader in the steering column reads the passive RFID tag in the plastic housing around the key. If it doesn't get the ID number it is programmed to look for, the car won't start.

In the early 1990s, IBM engineers developed and patented an ultra-high frequency (UHF) RFID system. UHF offered longer read range (up to 20 feet under good conditions) and faster data transfer. IBM did some early pilots with Wal-Mart, but never commercialized this technology. When it ran into financial trouble in the mid-1990s, IBM sold its patents to Intermec, a bar code systems provider. Intermec RFID systems have been installed in numerous different applications, from warehouse tracking to farming. But the technology was expensive at the time due to the low volume of sales and the lack of open, international standards.

UHF RFID got a boost in 1999, when the Uniform Code Council, EAN International, Procter & Gamble and Gillette put up funding to establish the Auto-ID Center at the Massachusetts Institute of Technology (MIT). Two professors there, David Brock and Sanjay Sarma, had been researching the possibility of putting low-cost RFID tags on all products to track them through the supply chain. Their idea was to put only a serial number on the tag to keep the price down, as a simple microchip that stored very little information would be less expensive to produce than a more complex chip with more memory. Data associated with the serial number on the tag would be stored in a database that would be accessible over the Internet.

Sarma and Brock essentially changed the way people thought about RFID in the supply chain. Previously, tags were a mobile database that carried information about the product or container they were on with them as they traveled. Sarma and Brock turned RFID into a networking technology by linking objects to the Internet through the tag (Roberti, "History of RFID," 2005). For businesses, this was an important change, because now a manufacturer could automatically let a business partner know when a shipment was leaving the dock at a manufacturing facility or warehouse, and a retailer could automatically let the manufacturer know when the goods arrived.

Between 1999 and 2003, the Auto-ID Center gained the support of more than 100 large end-user companies, plus the U.S. Department of Defense and many key RFID vendors. It opened research labs in Australia, the United Kingdom, Switzerland, Japan and China. It developed two air interface protocols (Class 1 and Class 0), the Electronic Product Code (EPC) numbering scheme (Sarma et al., "RFID Systems," 2003), and a network architecture for looking up data associated on an RFID tag on the Internet (Brock, "Electronic Product Code," 2001). The technology was licensed to the Uniform Code Council in 2003, and the Uniform Code Council created EPCglobal, as a joint venture with EAN International, to commercialize EPC technology. The Auto-ID Center closed its doors in October 2003, and its research responsibilities were passed on to Auto-ID Labs.

The Auto-ID Center used the term "Internet of Things" beginning in about 2000 and heavily promoted the concepts and ideas of a connected world with the EPC system as the basis of how things are connected to the Internet. Though Kevin Ashton (then the executive director of the Auto-ID Center) claims to have coined the term "Internet of Things," according to Prof. Daniel Engels, the term was used in a 1997 publication by the International Telecommunication Union (ITU) (Thiesse et al., "Overview of EPC," 2006).

2.3 Standards

Though many organizations work on the standardization process, we focus here on those that work on IoT and provide a definition for it. Accordingly, we considered IoT definitions from the European Telecommunications Standards Institute (ETSI), ITU, IEEE, the Internet Engineering Task Force (IETF), the National Institute of Standards and Technology (NIST), the Organization for the Advancement of Structured Information Standards (OASIS) and the World Wide Web Consortium (W3C). This list may be expanded in the future.

2.3.1 IEEE

IEEE is a global, professional engineering organization whose mission is to foster technological innovation and excellence for the benefit of humanity.

In its special report on Internet of Things issued in March 2014 (IEEE, "Internet of Things," 2014), IEEE described the phrase "Internet of Things" as:

"A network of items—each embedded with sensors—which are connected to the Internet."

This statement is written as a description of the "Internet of Things," not as an official definition of the concept. But we can see that the description addresses just the physical aspect of IoT.

The IEEE Standards Association (IEEE-SA), a globally recognized standards-setting body within IEEE, develops consensus standards through an open process that engages industry and brings together a broad stakeholder community. IEEE standards set specifications and best practices based on current scientific and technological knowledge.

The IEEE-SA has a portfolio of over 900 active standards and more than 500 standards under development. In its research into IoT, it has identified over 140 existing standards and projects that are relevant to the IoT. (See http://standards.ieee.org/innovate/iot/ for the lists of standards and projects.)

One project that directly relates to IoT is IEEE P2413™ (http://standards.ieee.org/innovate/iot/). The scope of IEEE P2413 is to define an architectural framework, addressing descriptions of various IoT domains, definitions of IoT domain abstractions, and identification of commonalities between different IoT domains.

The goals for the IEEE P2413 working group that is developing this standard are to

- accelerate the growth of the IoT market by enabling cross-domain interaction and platform unification through increased system compatibility, interoperability and functional exchangeability.
- define an IoT architecture framework that covers the architectural needs of the various IoT application domains.

- increase the transparency of system architectures to support system benchmarking, safety and security assessments.
- reduce industry fragmentation and create a critical mass of multi-stakeholder activities around the world.
- leverage the existing body of work.

IEEE P2413 is currently considering the architecture of IoT as three-tiered, with the layers explained in Figure 2.

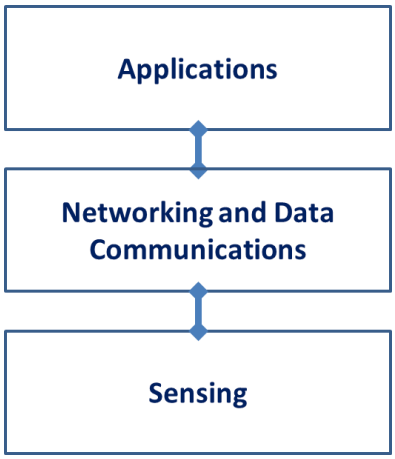


Figure 2. Three-tier architecture of IoT

IEEE P2413 also currently posits the extent of an IoT market and the stakeholders of IoT, as represented in Figure 3, below:

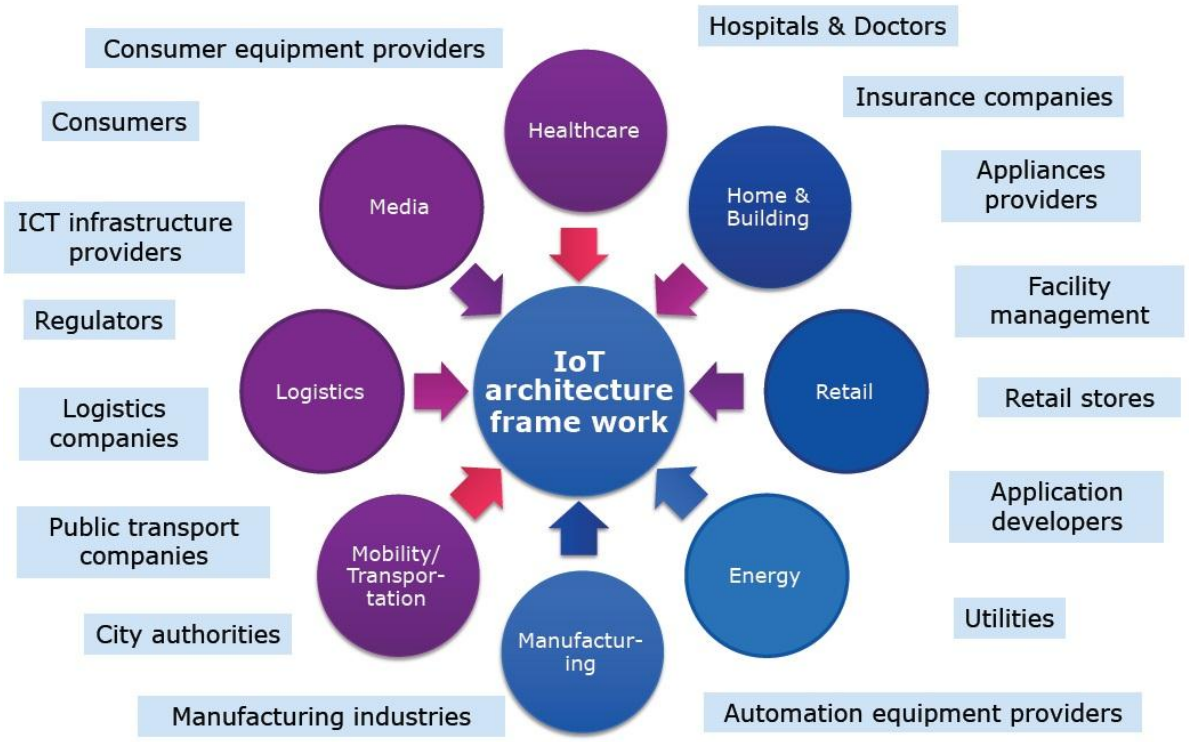


Figure 3. IoT markets and stakeholders

2.3.2 ETSI

ETSI produces globally applicable standards for information and communications technologies (ICT), including fixed, mobile, radio, converged, broadcast and Internet technologies. It is officially recognized by the European Union (EU) as a European Standards Organization (ESO).

Though ETSI doesn't mention the word "Internet of Things" in its document, it discusses a similar concept under the label of "machine to machine (M2M) communication" (ETSI, "Machine-to Machine," 2010). Accordingly, ETSI defines M2M communication as:

"Machine-to-Machine (M2M) communications is the communication between two or more entities that do not necessarily need any direct human intervention. M2M services intend to automate decision and communication processes."

ETSI also deals with the architectural view of the M2M communication, where the logical entities comprising the architecture are also represented in Figure 4 and defined as follows:

M2M Device: A device that runs M2M application(s) using M2M service capabilities. M2M devices connect to network domain in the following two ways:

Direct Connectivity: M2M devices connect to the network domain via the access network. The M2M device performs the procedures such as registration, authentication, authorization, management and provisioning with the network domain. The M2M device may provide service to other devices connected to it that are hidden from the network domain.

Gateway as a Network Proxy: The M2M device connects to the network domain via an M2M gateway. M2M devices connect to the M2M gateway using M2M area network. The M2M gateway acts as a proxy for the network domain towards the M2M devices that are connected to it. Examples of procedure that are proxied include: authentication, authorization, management and provisioning.M2M devices may be connected to the network domain via multiple M2M gateways.

M2M Area Network: Provides connectivity between M2M devices and M2M gateways.

M2M Gateway: A gateway that runs M2M application(s) using M2M service capabilities. The gateway acts as a proxy between M2M devices and the network domain. The M2M gateway may provide service to other devices connected to it that are hidden from the network domain.

Access Network: is a network, which allows the M2M device and gateway domain to communicate with the core network.

Core Network: provides:

- ✓ IP connectivity at a minimum and potentially other connectivity means,
- ✓ Service and network control functions,
- ✓ Interconnection (with other networks) and
- ✓ Roaming.

M2M Service Capabilities: Applications that run the service logic and use M2M service capabilities accessible via an open interface.

Network Management Functions: All the functions required to manage the access and core networks: these include provisioning, supervision, fault management, etc.

M2M Management Functions: All the functions required to manage M2M service capabilities in the network domain. The management of the M2M devices and gateways uses a specific M2M service capability (ETSI, “Machine-to Machine,” 2010).

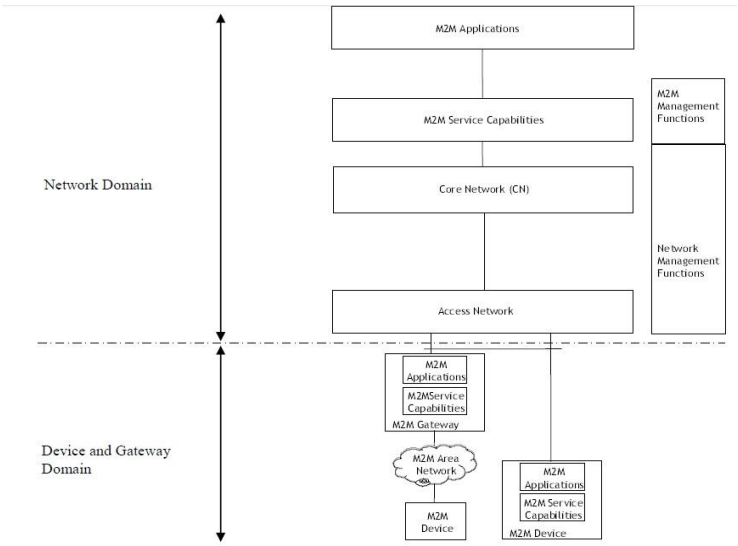


Figure 4. ETSI architectural model for M2M communication

2.3.3 OneM2M

OneM2M is a global partnership developing standards for machine-to-machine (M2M) communications enabling large-scale implementation of IoT. OneM2M works in partnership with various standardization organizations, vendors and service providers like ETSI, IEEE, Cisco, Telecom Italia and others.

OneM2M doesn't offer a precise definition of M2M/IoT systems; instead it provides an exhaustive list of requirements that a M2M/IoT system fulfills (ETSI, "oneM2M Requirements," 2013). But it does provide an illustration (see Figure 5, below) of the functional roles in an M2M ecosystem.

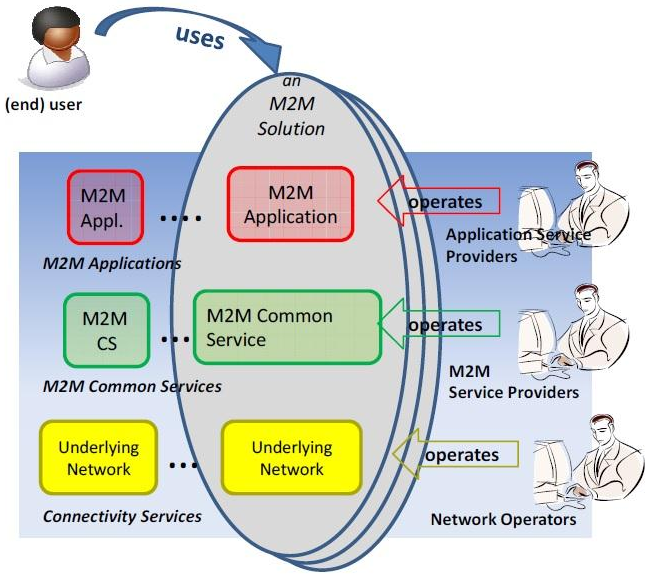


Figure 5. Functional roles in the M2M ecosystem

The functional entities and their requirements in this illustration are described below:

1. The User (individual or company – aka, end-user):
   - ✓ Uses an M2M solution
2. The Application Service Provider:
   - ✓ Provides an M2M application service
   - ✓ Operates M2M applications
3. The M2M Service Provider:
   - ✓ Provides M2M services to Application Service Providers
   - ✓ Operates M2M common services
4. The Network Operator:
   - ✓ Provides connectivity and related services for M2M Service Providers
   - ✓ Operates an underlying network. Such an underlying network could, e.g., be a telecom network.

Any of the above functional roles may coincide with any of the other roles. These functional roles do not imply business roles or reflect architectural assumptions.

OneM2M provides a detailed standard for M2M/IoT in relation to architecture, interfaces, security, communication protocols and the like. The oneM2M has a layered model, which is represented by the picture, Figure 6, below.

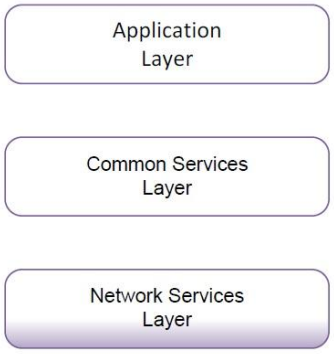


Figure 6. oneM2M layered model

Application layer: comprises oneM2M applications and related business and operational logic.

Common services layer: consists of oneM2M service functions that enable oneM2M applications (e.g., management, discovery and policy enforcement).

Network services layer: provides transport, connectivity and service functions.

ETSI, a contributor to the oneM2M Global Initiative, is working towards "horizontalizing" the pipes. As shown in Figure 7. Vertical and horizontal pipe standardization scenarios, below, a vertical pipe scenario is one in which there is one application, one network and one (or a few) type(s) of device(s). On the other hand, a "vertical pipe scenario" can also describe a model where applications share common infrastructure, environments and network elements (ETSI, "oneM2M Requirements," 2013).

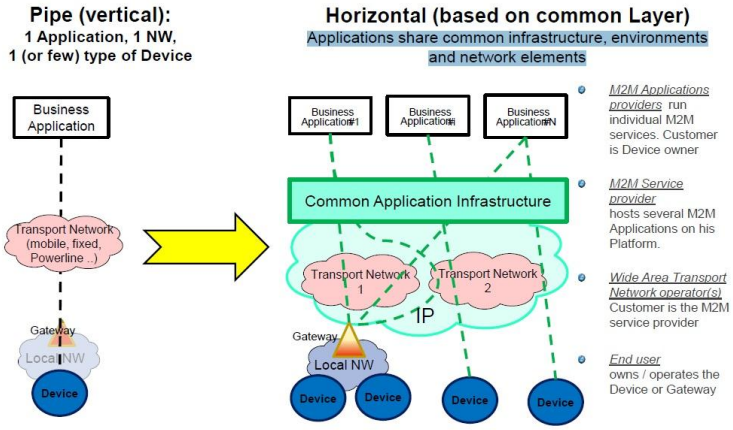


Figure 7. Vertical and horizontal pipe standardization scenarios

2.3.4 ITU

The ITU is the United Nations specialized agency for information and communication technologies (ICTs). It allocates global radio spectrum and satellite orbits, develops the technical standards that ensure networks and technologies seamlessly interconnect and strives to improve access to ICTs to underserved communities worldwide.

In its 2005 IoT report, ITU describes the IoT as a "ubiquitous network," in which the concept of ubiquitous networks is founded upon the all-inclusive use of networks and networked devices (ITU, SERIES Y, 2005). Literally, a ubiquitous networked environment is one in which networks and connectivity are available everywhere and anytime. Early forms of ubiquitous information and communication networks are evident in the widespread use of mobile phones.

The word "ubiquitous" comes from the Latin root of *ubique*, meaning everywhere. However, it is applied to the world of ICTs in at least two slightly different ways.

- In European usage, it tends to be interpreted geographically, meaning available from all parts of the globe, no matter how remote. Although possible, thanks to satellite technology, this may not be economically feasible.
- In Japan and the Republic of Korea, the word is used more often in a social rather than geographical context, meaning that a particular communication service may be universally available. The phrase "ubiquitous network society" is defined in Japan, for instance, as "available anywhere, anytime, by anything and anyone."

Accordingly, ITU endorses the definition of IoT as a network that is: *"Available anywhere, anytime, by anything and anyone."*

In this context, consumer products might be tracked using tiny radio transmitters or tagged embedded hyperlinks and sensors. As illustrated in **Figure 8.** ITU definition of IoT

, connectivity will take on an entirely new dimension. Today, users can connect at any time and at any location. Tomorrow's global network will not only consist of humans and electronic devices, but all sorts of inanimate things as well. These things will be able to communicate with other things, e.g., fridges with grocery stores, laundry machines with clothing, implanted tags with medical equipment and vehicles with stationary and moving objects.

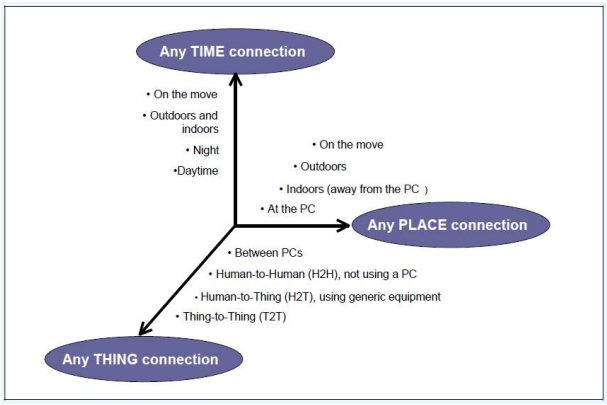


Figure 8. ITU definition of IoT

Additionally, ITU described the enabling technologies for the realization of the IoT. These technologies are: RFID for tagging things, sensor technologies for "feeling" things, smart technologies for making things "think" and nanotechnology for shrinking things. These enabling technologies are explained below.

Tagging things: RFID

In order to connect everyday objects and devices to large databases and networks – and indeed to the network of networks (the Internet) – a simple, unobtrusive and cost-effective system of item identification is indispensable. Only then can data about things be collected and processed. RFID offers a means to tag things.

RFID has an advantage over bar code because traditional bar codes identify only a category of product. For instance, all Gillette Mach 3 razor blades have the same bar code. However, with RFID tags, each pack of blades would have its own unique identifier that can be transmitted to suitably located readers for monitoring. The RFID tag can hold much more data than a bar code, and becomes in some sense a mini-database embedded in the item. Currently, the Electronic Product Code (EPC) is the dominant standard for data contained in RFID tags for the purpose of item-level tracking. RFID also allows data capture without the need for a line of sight between a sensor and a tag. Some applications limit the read range of RFID tags to between 0.15 – 0.20 meters, but the majority have a range of approximately one meter. Newer tags in the UHF RF bands could even have a range of 6.0-7.5 meters. This means that physical manipulation or access to individual items (often stacked or piled) is not needed for identification and tracking. This is not the case with the bar code, which must be "seen" at close range by scanners in order to be identified.

Eventually, it will be feasible to "tag and track" virtually every object on Earth. Anything from a medical instrument to a house key, from a cat to a human being, has the potential to become a node of the Internet.

Feeling things: Sensor technologies

Sensors are one of the key building blocks of IoT. As ubiquitous systems, they can be deployed everywhere – from military battlefields to vineyards and redwoods and on the Golden Gate Bridge. They can also be implanted under human skin, in a purse or on a t-shirt. Some can be as small as four millimeters in size, but the data they collect can be received hundreds of miles away. They complement human senses and have become indispensable in a large number of industries, from health care to construction. Sensors have a key advantage in that they can anticipate human needs based on information collected about their context. Their intelligence, "multiplied" by numerous networks, allows them not only to report about the external environment, but also to take action without human intervention.

Within an intelligent networked system, sensors perform the functions of input devices – they serve as "eyes," collecting information about their environment. In contrast, actuators serve as output units – they act as "hands," implementing decisions.

Thinking things: Smart technologies

Embedded intelligence in the things themselves can further enhance the power of the network by devolving information processing capabilities to the edges of the network.

Smart materials incorporate sensors and actuators, as they sense stimuli and respond accordingly. Currently, there are three main kinds of smart materials.

- "Passive" smart materials that respond directly and uniformly to stimuli without processing any of the signal;

- "Active" smart materials that can, with a remote controller, sense a signal and determine how to respond; and
- "Autonomous" smart materials that carry fully integrated controllers, sensors and actuators.

Shrinking things: nanotechnology

Nanotechnology focuses on the design, characterization, production and application of structures and devices through the manipulation and characterization of matter at the nanoscale. Potential benefits include increased speed and memory capacities, and a decrease in energy consumption and, of course, size.

ITU-T Study Group 13

ITU-T Study Group 13 leads the work of the ITU on standards for next-generation networks (NGN) and future networks (ITU, SERIES Y, 2005). It has defined IoT as:

"A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies."

NOTE 1 – Through the exploitation of identification, data capture, processing and communication capabilities, the IoT makes full use of things to offer services to all kinds of applications, while ensuring that security and privacy requirements are fulfilled.

NOTE 2 – From a broader perspective, the IoT can be perceived as a vision with technological and societal implications.

2.3.5 IETF

The Internet Engineering Task Force (IETF) is a large, open, international community of network designers, operators, vendors and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. It is open to any interested individual.

IETF provides its own description of IoT, along with definitions for "Internet" and "thing" (IETF, "Internet of Things," 2010):

"The basic idea is that IoT will connect objects around us (electronic, electrical, non-electrical) to provide seamless communication and contextual services provided by them. Development of RFID tags, sensors, actuators, mobile phones make it possible to materialize IoT which interact and co-operate each other to make the service better and accessible anytime, from anywhere."

IETF's definition of "Internet":

"The original 'Internet' is based on the TCP/IP protocol suite but any network based on the TCP/IP protocol suite cannot belong to the Internet because private networks and telecommunication networks are not part of the Internet even though they are based on the TCP/IP protocol suite. In the viewpoint of IoT, the 'Internet' considers the TCP/IP suite and non-TCP/IP suite at the same time."

IETF's definition of "things":

"In the vision of IoT, 'things' are very various such as computers, sensors, people, actuators, refrigerators, TVs, vehicles, mobile phones, clothes, food, medicines, books, etc. These things are classified as three scopes: people, machine (for example, sensor, actuator, etc.) and information (for example, clothes, food, medicine, books, etc.). These 'things' should be identified at least by one unique way of identification for the capability of addressing and communicating with each other and verifying their identities. In here, if the 'thing' is identified, we call it the 'object.'"

2.3.6 NIST

The National Institute of Standards and Technology (NIST) is part of the U.S. Department of Commerce and one of the U.S.'s oldest physical science laboratories. NIST measurements support the smallest of technologies—nanoscale devices so tiny that tens of thousands can fit on the end of a single human hair—to the largest and most complex of human-made creations, from earthquake-resistant skyscrapers to wide-body jetliners to global communication networks.

NIST mainly considers IoT under the umbrella of "cyber-physical systems" and it uses the two words interchangeably. NIST too gives a description of IoT rather than a formal definition. Two descriptions of IoT by NIST are presented below. One description is taken from a NIST team identified as the "Smart America/Global Cities Challenge" and the other is from the blogger Chris Greer, a NIST senior executive for cyber-physical systems.

The Smart America/Global Cities Challenge description of IoT:

"Cyber-physical systems (CPS) – sometimes referred to as the Internet of Things (IoT) – involves connecting smart devices and systems in diverse sectors like transportation, energy, manufacturing and healthcare in fundamentally new ways. Smart Cities/Communities are increasingly adopting CPS/IoT technologies to enhance the efficiency and sustainability of their operation and improve the quality of life. (NIST, "Global City Teams," 2014)"

Greer's description:

"Cyber-physical systems, also called the Internet of Things, are the next big advance for our use of the web. They allow complex systems of feedback and control that can help a robot coordinate with a dog or human in a search-and-rescue operation or help health care providers evaluate the recovery of patients after they leave the hospital" (Greer, "Internet's Next Big Idea," 2014).

In support of his description, Greer uses the following picture, Figure 9:

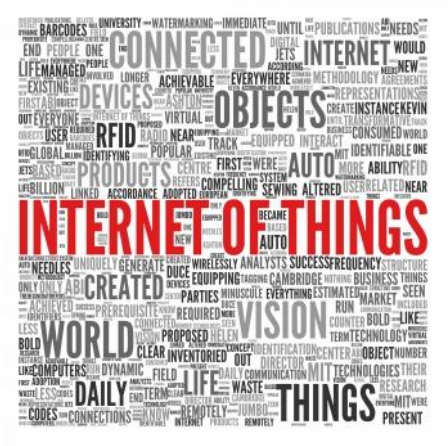


Figure 9. Chris Greer's pictorial representation of IoT

2.3.7 OASIS

OASIS is a non-profit consortium that drives the development, convergence and adoption of open standards for the global information society. OASIS promotes industry consensus and produces worldwide standards for security, IoT, cloud computing, energy, content technologies, emergency management and other areas.

OASIS describes IoT as (OASIS, "Open Protocols," 2014):

"System where the Internet is connected to the physical world via ubiquitous sensors."

OASIS describes the ubiquity of sensors as existing in "every mobile, every auto, every door, every room, every part, on every parts list, every sensor in every device in every bed, chair or bracelet in every home, office, building or hospital room in every city and village on Earth."

2.3.8 W3C

The W3C is an international community where member organizations, a full-time staff and the public work together to develop Web standards.

W3C addresses the IoT under the umbrella of "Web of Things" (see section 3.6). "Web of things" is related to IoT from the application and Web technologies perspective. W3C defines "Web of Things" as follows (W3C, "Web of Things," no date):

"The Web of Things is essentially about the role of Web technologies to facilitate the development of applications and services for the Internet of Things, i.e., physical objects and their virtual representation. This includes sensors and actuators, as well as physical objects tagged with a bar code or NFC. Some relevant Web technologies include HTTP for accessing RESTful services, and for naming objects as a basis for linked data and rich descriptions, and JavaScript APIs for virtual objects acting as proxies for real-world objects."

2.3.9 Recap

Among the standardization bodies, ETSI provides an architectural model for M2M communication centered on the usage of connectivity and related models. Data are also a relevant part of the architectural design. ITU describes enabler technologies that are required to bring IoT into reality. In addition to a structural model, ETSI gives a detailed design and

description of protocols, addressing security for M2M communication. IEEE 2413 provides an architectural framework including descriptions of various IoT domains, definitions of IoT domain abstractions, and identification of commonalities between different IoT domains. Most of the standardization bodies emphasize the network and communication aspect of IoT but W3C works on the standardization of the Web in a way that supports IoT applications and virtual representation of IoT components in the Internet. Accordingly, merging the communication-oriented works done by other standardization bodies, like ETSI, with the software-oriented work done by W3C will allow IoT to be practical.

As to the definitions given by these groups, most are general. They are intended to describe IoT rather than provide a formal definition that addresses all the features of the IoT concept. Among these definitions, we think the one given by ITU-T is the better one, as it tends to address the different facets of IoT. But ITU-T's definition still lacks features such as the sensing and actuation capability and ubiquity of the system. The effort with IEEE P2413 seems quite promising and interesting. An ecosystem for IoT and its impact on stakeholders will be identified. And some architectural principles are to be put forward.

From a general perspective, the segmentation of functions as proposed in the several definitions, however, seems to neglect a specific point that could be of paramount importance for the evolution of IoT: the platform or the infrastructure layer. It is at this layer that functions and services should be represented. The network layer just conveys information, while the platform decouples from the network the intricacies and the specificity of the applications. In our view, the real value of the IoT resides in the platforms and not in applications or communication capabilities.

Based on these definitions, it would appear that IoT will be characterized as a set of interworking networks of things that can be made smart if they can be identified, named and addressed (smart objects). "Things" can be physical objects or their descriptions or data related to them or even relationships between objects. For a majority of definitions a thing will be a node of a network. IoT systems show scaling capabilities, from small systems based on a few sensors up to large and complex systems. Under this perspective the differentiation between nodes is emerging: sensor, actuator, gateway, virtual object. All of them assume ubiquitous connectivity, while each entity performs different functions. Another emerging aspect is the possibility of using functions offered at things' interfaces.

2.4 Research Projects

2.4.1 CASAGRAS Project

CASAGRAS stands for "Coordination and support action for global RFID-related activities and standardization," a project financed by the EU focused on foundational studies on international questions about RFID, in support of IoT.

CASAGRAS' definition of IoT (CASAGRAS, "Final Report," 2009):

"A global network infrastructure, linking physical and virtual objects through the exploitation of data capture and communication capabilities. This infrastructure includes existing and evolving Internet and network developments. It will offer specific object-identification, sensor and connection capability as the basis for the development of independent cooperative services and

applications. These will be characterized by a high degree of autonomous data capture, event transfer, network connectivity and interoperability."

As with many definitions that seek to encapsulate a multi-faceted concept there is a need to qualify what is meant by particular words in order to minimize ambiguity. Where a definition has to serve disparate nationalities and language barriers the difficulty of achieving clarity is even more demanding, particularly where specific terms do not have analogues in other languages. Thus to clarify CASAGRAS' IoT definition, the following terms are explained.

"Global network infrastructure" describes what it is. It is a structure that is similar in many ways to that of the global or world-wide Internet itself. It allows messages from communicating devices to be communicated to other communicating devices via a network of computer connections, packets of data comprising the message being sent via routing devices to the final destination and in the right order. IoT will invariably exploit this Internet infrastructure, at least initially. But the computer nodes will increasingly be replaced by autonomous computer functionality facilitated by "smart devices" or embedded computer-based systems that avoid the need for human intervention yet serve to satisfy human-defined needs, be they personal, corporate or otherwise.

"Physical objects" refer to any tangible physical entity or thing, be it animate or inanimate, at any level of complexity and able to be characterized in some way for the purposes of unique identification.

"Virtual objects "are those objects that are represented in media space and may exhibit a proxy relationship with a physical object. Again, the need is seen to assign identity to the object if it is to be accommodated within the IoT.

"Data capture" and "autonomous data capture" refers to the process of obtaining data from a particular source and introducing the data into a communication to a computing or other data handling system. Increasingly, the data capture process will exploit the advantages of automatic identification and data capture (AIDC) systems with less and less human intervention when implementing applications or services within the IoT.

"Specific object-identification" refers to the way in which objects will be identified, either through natural features where this is appropriate or by codes in data carriers such as linear bar codes, two-dimensional codes or RFID tags.

"Sensor" or "sensors" refer to a particular category of devices that can sense or measure defined physical, chemical or biological quantities and generates associated quantitative data. This is in contrast to other sensor definitions that are encountered in relation to the IoT in which devices such as RFID readers are considered to sense the data they acquire.

"Actuation" and "sensor-actuation networks" (SANs) are often coupled with sensors and the notion of sensing, implying a coupling that features in most control systems. Actuation is therefore a further important aspect for the IoT, not only with respect to sensing but also with respect to particular human-to-object applications in which a device or system has to be activated or operated (such as an access barrier or door).

"Connection capability" and "connectivity" both refer to the ability to introduce or interface between a source of data and a device that can carry or handle it. The greater the capability or

connectivity the more effectively data can be transferred. Performance factors and criteria will be associated with such capabilities.

The CASAGRAS project also offers a high-level architectural model of IoT. The CASAGRAS architecture consists of three layers:

I. ***Physical layers***- in which the physical objects or things are identified and rendered functional components of the Internet of Things through the use of object-connected data carrier technologies, including RFID.

II. ***Interrogator-Gateway Layer*** - providing effectively the interfaces between the object-connected devices and between the interrogator and the information management systems.

III. ***Information Management, Application and Enterprise Layer***- Interfacing with the interrogator-gateway layer the information management layer provides the functional platform for supporting applications and services.

The CASAGRAS project architectural model is depicted in Figure 10.

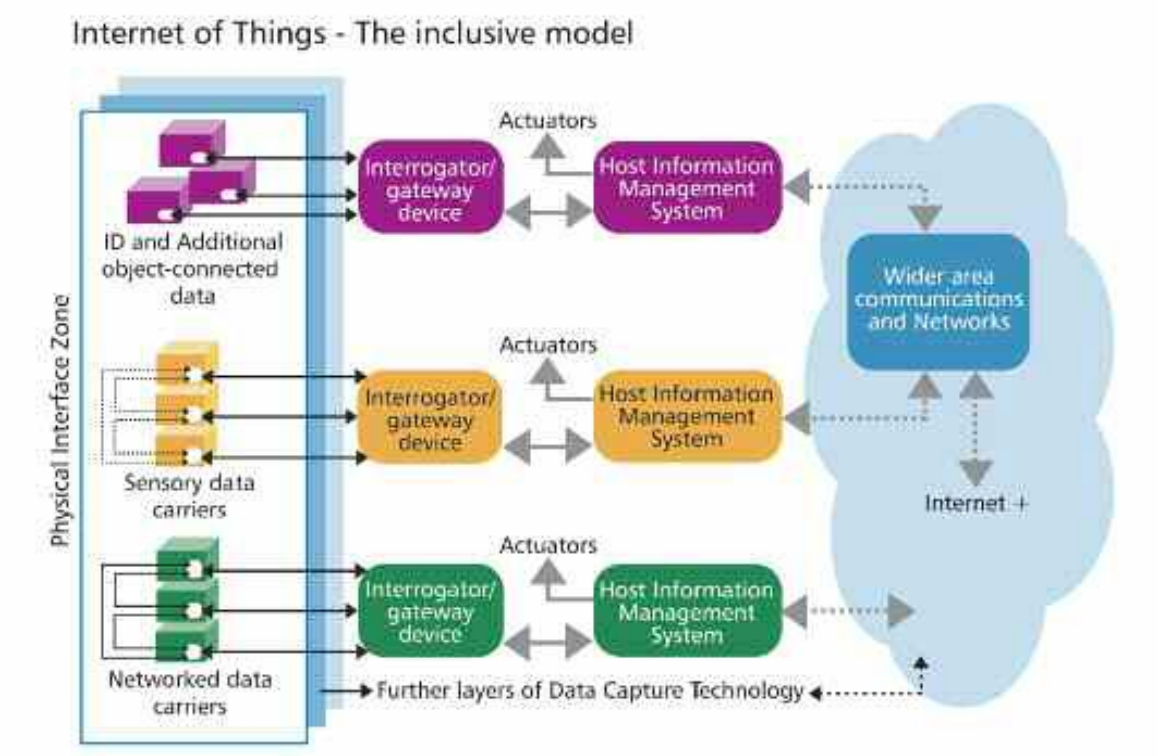


Figure 10. CASAGRAS project architectural model

2.4.2 Berkeley University (Cyber Physical Systems)

Most IoT activities in the U.S. are considered under the topic of cyber-physical systems (CPS). Though the two concepts of IoT and CPS are similar, there's a difference in the applications that the system is used for. This difference will be addressed in section 5.1 of this document. The

Berkeley University team defines cyber-physical systems as (Lee, “Cyber Physical Systems,” 2008):

“… integrations of computation, networking and physical processes. Embedded computers and networks monitor and control the physical processes, with feedback loops where physical processes affect computations and vice versa.”

2.4.3 IoT-A Project

IoT-A is a European project that aims to develop an architectural reference model for IoT.

IoT-A describes IoT in the following manner (Bassi, et al., “Enabling Things to Talk,” 2013; IoT-A, “Internet of Things Architecture,” 2011):

“It can be seen as an umbrella term for interconnected technologies, devices, objects and services.”

IoT-A project mainly focuses on developing an architectural reference model, along with security, addressing and management and protocol-level interaction of the various components of the architecture.

The IoT-A model has three sub-models: the Domain Model, Information Model and Functional Model. The IoT architecture is included in the Domain Model of the three sub-models. The IoT-A Domain Model and the interaction between the different components is represented in the following diagram, Figure 10:

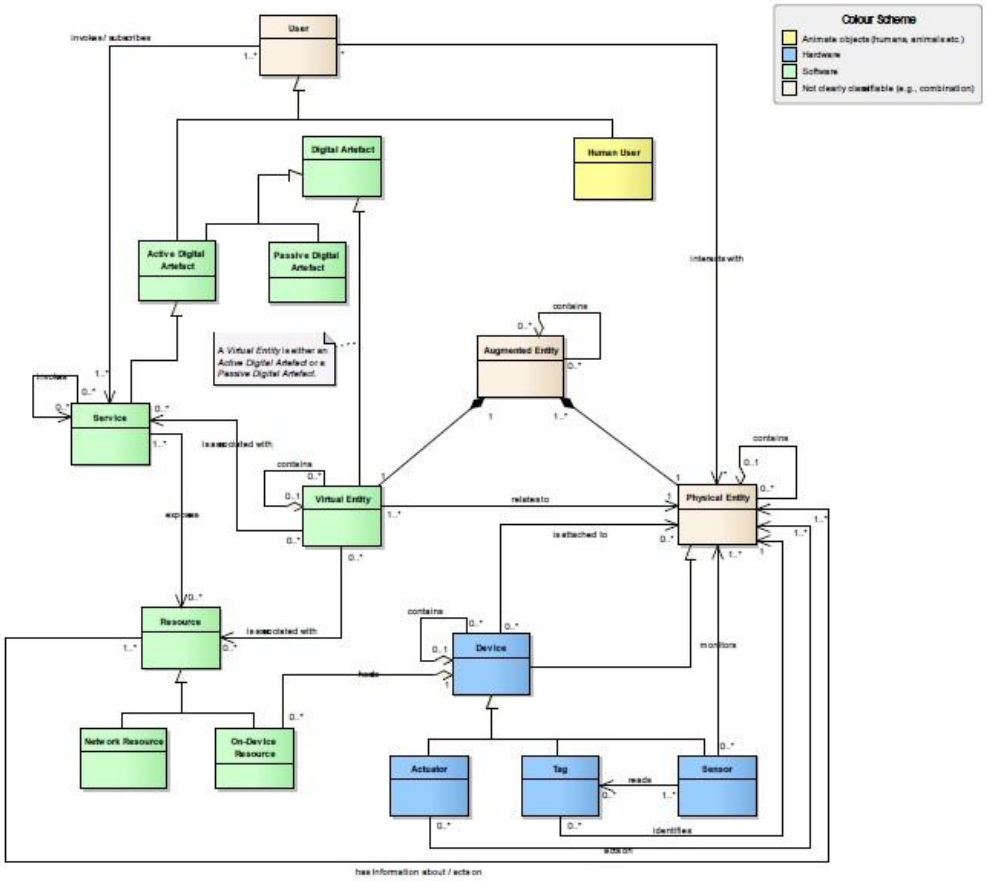


Figure 11. IoT-A architectural model components interaction

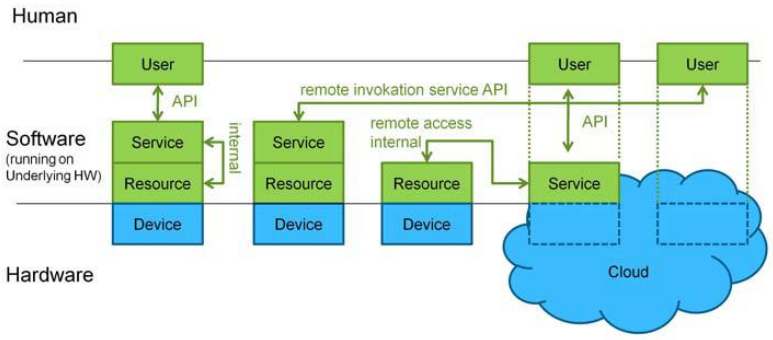


Figure 12. Devices, resources and services

Figure 12 depicts the relationship between services, resources and devices and shows several deployment options. Network-based resources are not shown, as they can be regarded as being hidden behind cloud-based services. In this document, we mainly consider the architectural

model of IoT-A. The components making up this particular architecture will be described in detail in a later chapter.

2.4.4 CERP-IoT Project

The IoT initiative (IoT-i), an EU Framework Programme 7 project, began in September 2010 and it brings together key actors from all relevant but currently fragmented IoT communities in Europe to work jointly towards a common vision of the IoT. The Cluster of European Research Projects on the Internet of Things, or CERP-IoT, completely adopts the architectural reference model of the IoT-A project.

The CERP-IoT definition of IoT (CERP-IoT, "Visions and Challenges," 2010) starts with the definition of "thing" in the context of "Internet of Things." In the CERP-IoT view, a "thing" could be defined as a real/physical or digital/virtual entity that exists and move in space and time and is capable of being identified. Things are commonly identified either by assigned identification numbers, names and/or location addresses.

As for an IoT definition, the CERP-IoT project states:

"Internet of Things (IoT) is an integrated part of Future Internet and could be defined as a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual "things" have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network. In the IoT, 'things' are expected to become active participants in business, information and social processes where they are enabled to interact and communicate among themselves and with the environment by exchanging data and information 'sensed' about the environment, while reacting autonomously to the 'real/physical world' events and influencing it by running processes that trigger actions and create services with or without direct human intervention. Interfaces in the form of services facilitate interactions with these 'smart things' over the Internet, query and change their state and any information associated with them, taking into account security and privacy issues."

This definition of IoT has three shortcomings, according to the authors of *Architecting the Internet of Things* (Ucklemann et al., 2011):

- First, it lists components that have been mentioned before in relation to other visions such as pervasive or ubiquitous computing and therefore it is difficult to distinguish from these concepts.
- Second, it misses wider consideration of current developments and user interactions in the Internet commonly referred to as Web 2.0. Similar to the relationship between the World Wide Web (WWW) and the Internet, the addition of Web 2.0 functionality may be seen as a user-centric extension to the Internet of Things rather than an integral part of it. However, whereas the development of the Internet began more than thirty years before the realization of the WWW in the early 1990s, the Internet of Things is already being influenced by Web 2.0 functionality right from the beginning. Both technology developments have been happening in parallel rather than consecutively.
- Third, it does not provide a reason why or how the Internet of Things will be a self-sustainable and successful concept for the future. Self-sustainability encompasses

viability, including a dynamic global network infrastructure with self-configuring capabilities based on standards and interoperable communication protocols as well as openness for future extensions, ideas and technologies. Economic success may never have been a part of a definition for the Internet or other technical network infrastructures. Nevertheless, we consider it a valid consideration within a holistic definition approach as economic success and adoption is just as important as technical sustainability in a forward-looking statement.

2.4.5 IERC Definition

IoT European Research Cluster, IERC, is a European Union-funded project aimed at addressing the large potential for IoT-based capabilities in Europe and to coordinate the convergence of ongoing activities.

The IERC definition states that IoT is (IERC, "Internet of Things," 2014):

"A dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual 'things' have identities, physical attributes and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network."

IERC has supported its definition using an explanatory figure, depicted below in Figure 13:

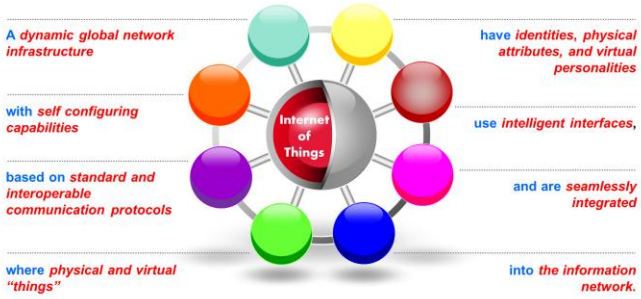


Figure 13. Pictorial representation of IoT by IERC project

2.4.6 ETP EPoSS Project

The European Technology Platform on Smart Systems Integration, ETP EPoSS is an industry-driven policy initiative, defining research and development (R&D) and innovation needs as well as policy requirements related to Smart Systems Integration and integrated Micro- and Nanosystems. EPoSS is contributing to EUROPE 2020, the EU's growth strategy for the coming decade, to become a smart, sustainable and inclusive economy.

ETP EPoSS provides three different definitions of IoT, which take into account the concepts of functionality and identity, seamless integration and semantic features of IoT (ETP EPoSS, Internet of Things in 2020," 2008).

"Things having identities and virtual personalities operating in smart spaces using intelligent interfaces to connect and communicate within social, environmental and user contexts."

A different definition, one that puts the focus on seamless integration, could be formulated as:

"Interconnected objects having an active role in what might be called the Future Internet."

Finally, ETP EPoSS defines IoT with a nod to semantics (ETP EPoSS, "Internet of Things in 2020," 2008):

"The semantic origin of the expression is composed by two words and concepts: 'Internet' and 'Thing,' where 'Internet' can be defined as 'the world-wide network of interconnected computer networks, based on a standard communication protocol, the Internet suite (TCP/IP),' while 'Thing' is 'an object not precisely identifiable.' Therefore, semantically, 'Internet of Things' means 'a world-wide network of interconnected objects uniquely addressable, based on standard communication protocols.'"

While the current Internet is a collection of rather uniform devices, however heterogeneous in some capabilities, it is expected that the IoT will exhibit a much higher level of heterogeneity, as totally different objects in terms of functionality, technology and application fields will belong to the same communication environment.

2.4.7 Internet Connected Objects for Reconfigurable Ecosystems (iCore)

iCore is an EU project which aims to empower the IoT through virtual objects and cognitive technologies.

iCore doesn't give a formal definition of IoT but we can grasp the view of iCore on IoT from the writings on the various deliverables released by the project. One such description of IoT by the iCore project (Berkers, "Vision of the Future," 2013):

"Our world is getting more and more connected. In the near future not only people will be connected through the Internet, but Internet connectivity will also be brought to billions of tangible objects, creating the Internet of Things (IoT)."

2.4.8 Other Internet of Things definitions

The Web site Postscapes (http://postscapes.com/internet-of-things-definition) offers a wide range of IoT definitions. Here we present a few of the interesting ones.

E-Flux: Internet of Things (Keller Easterling)
"An 'Internet of things' describes a world embedded with so many digital devices that the space between them consists not of dark circuitry but rather the space of the city itself. The computer has escaped the box, and ordinary objects in space are carriers of digital signals."

The Internet of People: Integrating IoT technologies is not a technical problem (Mike Kuniavsky)
"[The IoT] *...is the combination of distributed information processing, pervasive wireless networking and automatic identification, deployed inexpensively and widely. The underlying technologies and the applications that are traditionally discussed don't matter much, because it is this combination of factors that deeply affects people and industries, and it does it by connecting people's immediate experiences to the power of digitally aggregated and analyzed information. In other words, the Internet of Things turns physical actions into knowledge in the cloud and knowledge in the cloud into physical action in a way that's never existed before."*

LinkedIn IoT group discussion, April 2010 (Rick Bullotta)

"The Internet of Things represents the intersection of people (meatspace), systems (cyberspace) and the physical world (atomspace). It represents the 'connectedness' of these entities and the range of applications that it enables. It (will someday) represent a set of protocols for interacting with the information shadow (data, event streams) and capabilities (services) of the participants in the Internet of things. It (will someday soon) represent a semantic model for the connected entities. It is enabled by and intersects with the ubicompmacro trends. It affects and influences the development of the future internet (a topic we'll be discussing at the International Research Forum in a couple weeks) – the effect on IP addressing, security/packet validity, different types of QOS needs and higher level protocols could be substantial."

Between the Revolution of the Internet and the Metamorphosis of Objects **(Gérald Santucci, 2010)**

"The Internet of Things links the objects of the real world with the virtual world, thus enabling anytime, anyplace connectivity for anything and not only for anyone. It refers to a world where physical objects and beings, as well as virtual data and environments, all interact with each other in the same space and time."

Arduino, Sensors, and the Cloud **(Charalampos Doukas, 2012)**

"A global network infrastructure, linking physical and virtual objects using cloud computing, data capture and network communications. It allows devices to communicate with each other, access information on the Internet, store and retrieve data, and interact with users, creating smart, pervasive and always-connected environments."

2.4.9 Recap

Projects in the IoT space give better definitions and architectural models than the standardization bodies. Unfortunately, the acceptance of these definition and models is difficult outside of the community that works on a specific project. Among the projects mentioned above, CASAGRAS and IoT-A give an architectural model for IoT. The architectural model given by IoT-A appears to be the more complete one and it is endorsed by other European projects (like CERP-IoT, iCore) that try to create an acceptable framework for the development of IoT in Europe. This framework is based on the segmentation of functionalities and the identification of a number of basic components that provide APIs for programming functions, services and applications in the IoT field. The approach undertaken is the one of construction of a large horizontal platform to be used for a large number of IoT applications. For instance, the CASAGRAS architectural model is important especially from the layering point of view. The definition of IoT given by CERP-IoT is one of the better ones and it tries to address most features of IoT. In our view, integrating the different facets touched by the several projects and their integration into a standardization attempt will constitute an all-inclusive definition of IoT. There is still a long way to go in achieving this holistic view. The various projects are giving sufficient consideration to the current standardization framework and the input from projects and initiatives is much too fragmented in order to frame it into a standard. Clearly, a convergence of effort and ideas in the realm of IoT remains an unrealized goal.

2.5 National Initiatives

2.5.1 UK Future Internet Strategy Group

UK Future Internet Strategy Group is a business-led group established to help the United Kingdom play its part in shaping the future Internet by giving the UK a common voice and a

focus for stakeholder engagement, highlighting R&D priorities, advising the UK government and the EU on the future Internet.

The UK future Internet strategy group considers the IoT under the umbrella of "Future Internet," defined as (UK FISG, "Future Internet Report," 2011):

"An evolving convergent Internet of things and services that is available anywhere, anytime as part of an all-pervasive, omnipresent, socio–economic fabric, made up of converged services, shared data and an advanced wireless and fixed infrastructure linking people and machines to provide advanced services to business and citizens."

2.5.2 Digital Lifestyle Malaysia (DLM)

DLM is an initiative undertaken by The Malaysian Communications and Multimedia Commission (MCMC) to promote and accelerate the development and adoption of applications and services. It includes the adoption of intelligent IoT infrastructures in Internet-based communications transactions to promote growth and better quality of life. The commission describes IoT in the following way (Ramalingam, "Engage and Interact," 2013):

"The Internet of Things is a web in which gadgets, machines, everyday products, devices and inanimate objects share information about themselves in new ways, in real time. Using a range of technologies such as embedded radio frequency identification (RFID) chips linked with IP addresses (internet signatures), near-field communications, electronic product codes and GPS systems just about anything can be connected to a network. The connected objects can then be tracked and output information can be recorded, analyzed and shared in countless ways via the Internet."

2.5.3 Internet of Things Strategic Research Agenda (IoT-SRA)

IoT-SRA is a Finnish research initiative with the goal of directing the research efforts in Finland to focus on significant value creation.

Before providing a definition of IoT, IoT-SRA surveyed the definitions given by other entities and categorized the definitions into three perspectives as "Things-oriented," "Internet-oriented" and "Semantics-oriented" visions. The IoT-SRA presented the three visions and their respective definitions in the following ways (IoT-SRA, "Internet of Things Strategic Research," 2011):

1. The Things-oriented vision focuses on the things' identity and functionality, which is in line with the original idea presented by MIT Auto-ID Labs for using RFID tags to uniquely identify things. While the original idea was tied to the RFID and Electronic Product Code (EPC), other identification alternatives have emerged, and the concept of an identifiable object has been expanded to include virtual entities. From this perspective, IoT is defined as:

   "Things having identities and virtual personalities operating in smart spaces using intelligent interfaces to connect and communicate within social, environmental and user contexts."

Or

"A world-wide network of interconnected objects uniquely addressable based on standard communication protocols."

2. The Internet-oriented vision emphasizes the role of the network infrastructure and is concerned with the applicability of the available (and future) Internet infrastructure, including IP protocol stack and Web standards for the purpose of interconnecting smart objects. This perspective is promoted by, for example, the IPSO (IP for Smart Objects) Alliance, Internet architecture, and Web of Things community, suggesting that IoT shall be built upon the Internet architecture, by adopting and, when necessary, simplifying the existing protocols and standards. From this perspective, IoT can be defined (following the definition by the CASAGRAS project) as:

   "A global network infrastructure, linking physical and virtual objects through the exploitation of data capture and communication capabilities. This infrastructure includes existing and evolving Internet and network developments. It will offer specific object-identification, sensor and connection capability as the basis for the development of independent cooperative services and applications. These will be characterized by a high degree of autonomous data capture, event transfer, network connectivity and interoperability."

3. The Semantics-oriented vision focuses on systematic approaches towards representing, organizing and storing, searching and exchanging the things-generated information, by means of semantic technologies. According to this vision, the application of semantic technologies to IoT:

   "… promotes interoperability among IoT resources, information models, data providers and consumers, and facilitates effective data access and integration, resource discovery, semantic reasoning, and knowledge extraction" [through] "efficient methods and solutions that can structure, annotate, share and make sense of the IoT data and facilitate transforming it to actionable knowledge and intelligence in different application domains."

The IoT-SRA combines the above three approaches and devised their own definition as follows:

"[The IoT is] a global network and service infrastructure of variable density and connectivity with self-configuring capabilities based on standard and interoperable protocols and formats [which] consists of heterogeneous things that have identities, physical and virtual attributes, and are seamlessly and securely integrated into the Internet."

2.5.3 Recap

The importance of the national initiatives is very high. They show how to deploy and exploit current IoT solutions. The objective of the Initiatives is mainly to position a specific country at the forefront of this set of technologies and problem domains. They attempt to provide to a large extend the means and the perspective for a country to grow in this area. Having a clear

idea of what IoT is of paramount importance to achieve the expected results. The adoption of one IoT definition over another could have important consequences for how to reach global goals.

Most of the definitions given by the national initiatives are concise. Among the listed stakeholders, IoT-SRA made a survey of the definitions given by other bodies and categorized all the possible definitions into three classes as Internet-oriented, things-oriented and semantic-oriented. Finally, IoT-SRA provides a definition that encompasses all the three views mentioned above.

The national initiatives show clearly that approaching the IoT domain with a good understanding of what IoT is can guide a country's actions towards more concrete and appealing results in order to gain technological leadership in an important sector.

2.6 White Papers

2.6.1 "From the Internet of Computers to the Internet of Things" (Mattern et al., 2010)

"The Internet of Things represents a vision in which the Internet extends into the real world embracing everyday objects. Physical items are no longer disconnected from the virtual world, but can be controlled remotely and can act as physical access points to Internet services. An Internet of Things makes computing truly ubiquitous."

2.6.2 "Future Internet" (Society for Brain Integrity, Sweden, 2010)

The Society for Brain Integrity is a nonprofit organization established to create an awareness of cybernetic technology and electronic abuse, such as illegal data collection and manipulation of humans via a brain–machine interface.

The Society conceives of IoT as "Future Internet" and gives the following description of that concept:

"It means that any physical thing can become a computer that is connected to the Internet and to other things. IoT is formed by numerous different connections between PCs, human to human, human to thing and between things. This creates a self-configuring network that is much more complex and dynamic than the conventional Internet. Data about things is collected and processed with very small computers (mostly RFID tags) that are connected to more powerful computers through networks. Sensor technologies are used to detect changes in the physical environment of things, which further benefits data collection. The network becomes more powerful when intelligence can be embedded to things and processing power can be distributed more widely in the network."

The Society has defined the Future Internet as follows:

- ***Pervasiveness and ubiquity***: Digital content and services will be all around us in not only ICT devices but in any physical objects, too. Embedding computers to a physical environment creates a link between physical and digital worlds.
- ***Network of networks***: Internet of the Future connects networks of objects to the classic Internet. The result is a combination of different communication networks that are able to manage the complex communications of large amounts of information and enable

new kinds of services [82]. As the structure of the Internet becomes more complex and vulnerable to security threats, according to some views, governments and corporations are inclined to create separated spaces, "walled gardens" inside the Internet.

- ***Interoperability and Accessibility***: Devices and objects are networked and work seamlessly together. Interoperability is implemented also in the level of network architecture making the communication between services and applications also more fluent. Services and content can be accessed anywhere, anytime and with many different devices. Mobile devices will dominate globally as access points to the Internet. This is the case especially in developing countries where mobile devices are an affordable solution for the lack of built-in network infrastructure.
- ***Miniaturization with simplification***: In IoT the computers at the end nodes of the Internet are small to the point of being even invisible to the eye when embedded in the environment. The purpose of this kind of miniaturization is not necessarily to include the capacity of a full-blown computer in an ever smaller scale device but to include only those functionalities that are relevant and necessary in the particular environment and context of use. They are inexpensive and have low energy consumption and feature few functions like sensing, storing and communicating a limited amount of information and they normally need to be accessed with another device such as a mobile phone.
- ***Context-awareness***: Future Internet will be able to recognize different contexts by using different sensor technologies. On the physical level sensors gather information from the physical environment and on the digital level they gather information about the network and applications. When that information is combined with other input data the network and applications are able to dynamically adapt to optimal processes at any actual moment.
- ***Autonomy***: Input of information in the Future Internet does not have to be made by humans only. Machines will interact more and more with each other becoming more predominant than human-centric interaction. In IoT, sensors and actuators that are embedded in the environment can collect data autonomously and transmit it to each other and the network. In Semantic Web automatic processes can produce information combined from separate sources.
- ***Virtualization of resources***: Virtualization enables better exploitation of network resources with higher flexibility and security.
- ***Semantics***: Semantics are an important part of Future Internet. By the use of semantic annotations linked to the information in the Web locating information will become much easier, faster and more accurate.

2.6.3 "The Internet of Things: Networked objects and smart devices" (Hammersmith Group, 2010)

"The Internet of Things comprises a digital overlay of information over the physical world. Objects and locations become part of the Internet of Things in two ways. Information may become associated with a specific location using GPS coordinates or a street address. Alternatively, embedding sensors and transmitters into objects enables them to be addressed by

Internet protocols, and to sense and react to their environments, as well as communicate with users or with other objects."

2.6.4. "The Internet of Things" (Chui et al., 2010/McKinsey & Company)

"The physical world itself is becoming a type of information system. In what's called the Internet of Things, sensors and actuators embedded in physical objects—from roadways to pacemakers—are linked through wired and wireless networks, often using the same Internet Protocol (IP) that connects the Internet. These networks churn out huge volumes of data that flow to computers for analysis. When objects can both sense the environment and communicate, they become tools for understanding complexity and responding to it swiftly. What's revolutionary in all this is that these physical information systems are now beginning to be deployed, and some of them even work largely without human intervention."

2.6.5 "The Software Fabric for the Internet of Things" (Rellermeyer et al, 2008)

"The notion of an 'Internet of Things' refers to the possibility of endowing everyday objects with the ability to identify themselves, communicate with other objects, and possibly compute."

2.6.6 "The Internet of Things: In a Connected World of Smart Objects" (Accenture & Bankinter Foundation of Innovation, 2011)

"The Internet of Things (IoT) consists of things that are connected to the Internet, anytime, anywhere. In its most technical sense, it consists of integrating sensors and devices into everyday objects that are connected to the Internet over fixed and wireless networks. The fact that the Internet is present at the same time everywhere makes mass adoption of this technology more feasible. Given their size and cost, the sensors can easily be integrated into homes, workplaces and public places. In this way, any object can be connected and can 'manifest itself' over the Internet. Furthermore, in the IoT, any object can be a data source. This is beginning to transform the way we do business, the running of the public sector and the day-to-day life of millions of people."

2.6.7 "China' s Initiative for the Internet of Things and Opportunities for Japanese Business, " (Inoue et al., 2011/Normura Research Institute (NRI))

NRI is a Japanese research institute working mainly on areas related to the financial industry.

NRI indicated that IoT was considered to be "the network connecting things" in the late 1990s when RFID began to attract attention in the logistics and retail fields.

According to NRI, currently the concept has developed into a broader concept in which:

"a system automatically recognizes information about a thing such as 'unique attributes,' state at that 'time' and 'location' by using sensors and cameras connected to the Internet, and creates value-added information by comprehensively analyzing the state and location of two or more things. At the same time, the system uses such information to automatically control equipment and devices."

2.6.8 Recap

Once again, different perspectives produce different definitions in the white papers we've cited and excerpted. The definition given by the "Society for Brain Integrity in Sweden" provides defining features of IoT, in addition to attempting definitions. Overall, the white papers cover a

range of concepts, including ubiquity, unique identification, heterogeneous communication, service, the "smartness" of things and the connection of physical items with the virtual world. Like the previous definitions, no one definition here embraces all features of IoT. But the different definitions touch different aspects of IoT, and these can be merged to into an all-inclusive definition of IoT.

2.7 Books

2.7.1 *Architecting the Internet of Things* (Uckelmann et al. editors, 2011.)

Before attempting definition for IoT, the editors of this book considered what the IoT is *not* – or at least not exclusively. In support of this idea, they reference a related blog discussion started by Tomas Sánchez López, who argued that the IoT is not only:

- Ubiquitous / pervasive computing, which does not imply the usage of objects nor does it require a global Internet infrastructure,
- The Internet Protocol (IP), as many objects in the Internet of Things will not be able to run an Internet Protocol,
- A communication technology, as this represents only a partial functional requirement in the Internet of Things similar to the role of communication technology in the Internet,
- An embedded device, as RFID tags or wireless sensor networks (WSN) may be part of the Internet of Things, but as a stand-alone they miss the back-end information infrastructures and in the case of WSN the standards to relate to "things,"
- The application, just as Google or Facebook could not be used in the early 1990s to describe the possibilities offered by Internet or WWW.

In addition to the above negations, the authors added two more: "The Internet of Things is not the Internet of People (although we believe that the Internet of People will link to the Internet of Things) and it is not the Intranet or Extranet of Things."

Therefore, applications that provide only access to a small group of stakeholders (e.g., a few companies) should not be considered to represent the full scope of the IoT. However, all fields of research mentioned above overlap partially with the IoT, as depicted in Figure 14, below.

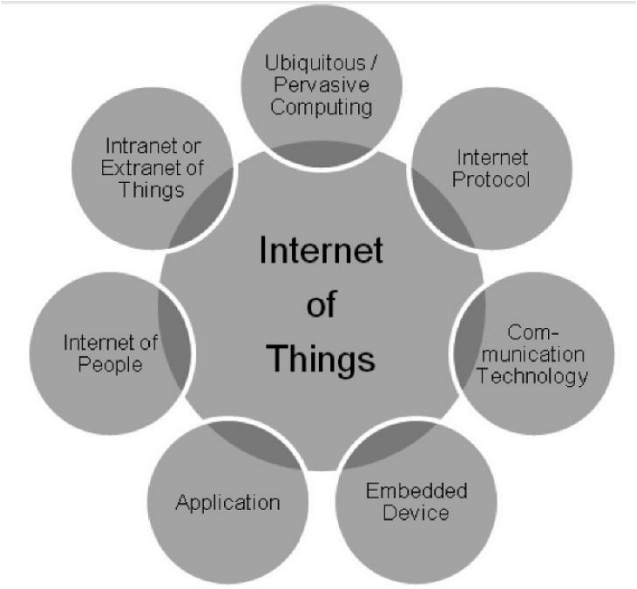


Figure 14. Overlaps of the Internet of Things with other fields of research

A minimalist approach towards a definition may include nothing more than things, the Internet and a connection in between. “Things” are any identifiable physical object independent of the technology that is used for identification or providing status information of the object and its surroundings. “Internet” in this case refers to everything that goes beyond an extranet, thus requiring access to information for more than a small group of people or businesses. A closed loop application consequently has to be regarded as an Extranet of Things. The Internet acts as a storage and communication infrastructure that holds a virtual representation of things linking relevant information with the object.

Eventually, the editors of *Architecting the Internet of Things* define the IoT as:

“*The future Internet of Things links uniquely identifiable things to their virtual representations in the Internet containing or linking to additional information on their identity, status, location or any other business, social or privately relevant information at a financial or non-financial pay-off that exceeds the efforts of information provisioning and offers information access to non-predefined participants. The provided accurate and appropriate information may be accessed in the right quantity and condition, at the right time and place at the right price. The Internet of Things is not synonymous with ubiquitous/pervasive computing, the Internet Protocol (IP), communication technology, embedded devices, its applications, the Internet of People or the Intranet/Extranet of Things, yet it combines aspects and technologies of all of these approaches.*”

This definition is built upon fundamental concepts that may be defined as follows:

Right quantity can be achieved through high granularity of information combined with filtering and intelligent processing.

Right time does not necessarily mean anytime, but more precisely “when needed.” It may be sufficient to receive information about an object only once a day or only in the case of a status change. Consequently, right time is not synonymous with real-time.

Right place does not imply any place. Rather, it implies the place where the information is needed or consumed (which may not be the same place it is generated). If information is not generated and consumed in the same place and if either of these places have unreliable or intermittent network connectivity, then effective data synchronization protocols and caching techniques may be necessary to ensure the availability of information at the right place.

Right information is a condition that may be met if it can be utilized with minimum effort. This includes human-readable information for human interaction as well as semantically and syntactically enriched machine-readable information, which may in turn require the transformation of low-level raw data (possibly from multiple sources) into meaningful information and may even require some pattern recognition and further analysis to identify correlations and trends in the generated data.

Right price is not automatically the lowest price, but instead it is a price between the costs for information provisioning and the achievable market price. Information provisioning costs include labor costs as well as infrastructure costs.

2.7.2 *The Internet of Things: 20th Tyrrhenian Workshop on Digital Communications* (Giusto et al., editors, 2010)

"The expression 'Internet of Things' is wider than a single concept or technology. It is rather a new paradigm that involves a wide set of technologies, applications and visions. Also, complete agreement on the definition is missing as it changes with relation to the point of view. It can focus on the virtual identity of the smart objects and their capabilities to interact intelligently with other objects, humans and environments or on the seamless integration between different kinds of objects and networks toward a service-oriented architecture of the future Internet."

2.7.3 *Internet of Things: Legal Perspectives* (Weber et al., 2010)

"A world where physical objects are seamlessly integrated into the information network, and where the physical objects can become active participants in business processes. Services are available to interact with these 'smart objects' over the Internet, query their state and any information associated with them, taking into account security and privacy issues."

2.7.4 *6LoWPAN: The Wireless Embedded Internet* (Shelby et al, 2011)

"Encompasses all the embedded devices and networks that are natively IP-enabled and Internet-connected, along with the Internet services monitoring and controlling those devices."

2.7.5 *Internet of Things: Global Technological and Societal Trends from Smart Environments and Spaces to Green ICT* (Vermesan et al, editors, 2011)

"The Internet of Things could be conceptually defined as a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual 'things' have identities, physical attributes and virtual personalities, use intelligent interfaces and are seamlessly integrated into the information network."

2.7.6 Recap

Like the other stakeholders we've cited, the definitions given by these books largely cover only some aspect of IoT and are aimed at describing an IoT is for their specific readers. But the definition given by the book, *Architecting the Internet of Things*, is helpful because it provides several different perspectives. First, it discusses what an IoT system is not, which provides

contrast to the topic. And by explaining the terms used in its definition helps comprehension by persons from different backgrounds. In our view, if amendments were made to the features of IoT addressed by this definition, it would qualify as an inclusive definition of IoT.

2.8 Industrial Activities

2.8.1 SAP Definition

SAP defines the IoT as (Haller, "Internet of Things," 2009):

"A world where physical objects are seamlessly integrated into the information network, and where the physical objects can become active participants in business processes. Services are available to interact with these 'smart objects' over the Internet, query and change their state and any information associated with them, taking into account security and privacy issues."

The SAP definition does not explain the key words it relies on. "Smart object," for instance, would benefit from further clarification. And the statement that "physical objects can become active participants in business processes" needs exegesis as well, as physical objects have always been associated with business processes. (Most businesses, irrespective of their size and function, are invariably involved with physical entities of one kind or another.) The implication here is the manner in which tagged or otherwise identified objects are integrated into business processes.

2.8.2 CISCO (Bradley, "Internet of Everything," 2013)

Cisco works on the IoT under the label, "the Internet of everything," which it defines as:

"Bringing together people, process, data and things to make networked connections more relevant and valuable than ever before, turning information into actions that create new capabilities, richer experiences and unprecedented economic opportunity for businesses, individuals and countries."

Yesterday, in this view, people, process, data and things functioned independently. Today, the Internet of Everything (IoE) brings them all together by combining machine-to-machine (M2M), person-to-machine (P2M), and person-to-person (P2P) connections.

Information extracted from these networked connections creates new capabilities, richer experiences and economic opportunity, as depicted in the overlapping domains in Figure 15.

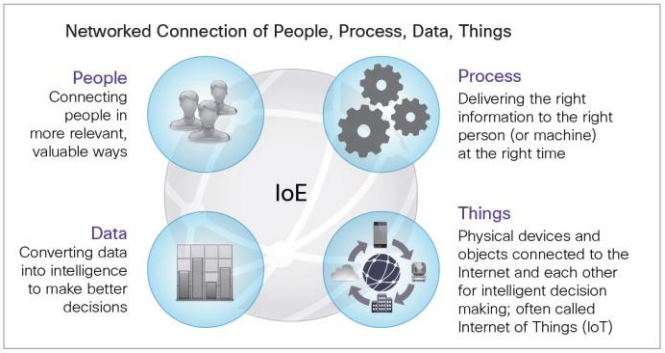


Figure 15. Cisco's pictorial representation of IoE

2.8.3 HP

HP provides the following definition (Miessler, "HP Security," 2014):

"The Internet of Things refers to the unique identification and 'Internetization' of everyday objects. This allows for human interaction and control of these 'things' from anywhere in the world, as well as device-to-device interaction without the need for human involvement."

In sum, the everyday things you grew up with – such as your toaster, your alarm clock, your car, your refrigerator and your television – are all going to be network/Internet connected. This means you (and hopefully just you) will be able to interact with them from wherever you are in the world – right from your mobile device.

A factory provides a good example of how the Internet of Things will bring significant advantage to how we conduct our daily business. Imagine a factory floor where the various components – such as the delivery trucks, the warehouse doors, the shipping containers, etc. – all are network aware and able to interact with each other in real time.

A forklift can configure itself to lift an inbound package, because the package told it that it was coming. And all the doors are open for it as it moves, because the doors know where the forklift is. And because the package is temperature sensitive, the storage area's thermostat made an automatic adjustment upon sensing it was on its way.

That's just a few devices interacting. Now think of billions of devices doing the same thing. An HP illustration puts this potentially daunting imagery into simplified form in Figure 16.

Figure 16. HP's pictorial representation of IoT

2.8.4 Recap

The definitions given by industry players are very precise, but address only a small aspect of IoT. Most of them emphasize the fact that "things" are made part of the Internet. Two of the three organizations cited above stated the benefit of IoT from a business process and economic development point of view, which reflects the profit motive as a potentially fundamental driver of IoT adoption/creation. Still, this approach reflects an enterprise point of view rather than striving for an all-inclusive, holistic definition of IoT.

2.9 Summary

In this chapter we surveyed the definitions and architectural models given by different stakeholders of IoT. As noted, each definition and/or model tends to reflect the viewpoint and motivations of the individual, organization or business providing it. Yet all can contribute to an attempt to devise an all-inclusive definition of IoT. From the approach taken by the book *Architecting the Internet of Things*, for instance, we see the value in defining what an IoT is and

what IoT is not so that we can define limits for a system to be considered as IoT. Having a demarcation line for an IoT system is helpful. The same book also contributes to our quest to define IoT by giving an explanation of the fundamental concepts in the definition. In fact, we endorse this specific approach as we provide our own definition of IoT in the next chapter. As for the architectural models provided by these stakeholders, we favor IoT-A's model and we will largely adopt this architecture as our own in a later chapter. The architectures given by ETSI and CASAGRAS may be adopted in approaches that rely on a layering point of view.

Thus, in the coming chapters, we combine these sometimes disparate efforts to provide an all-inclusive definition and a minimal set of architectural models for IoT.

3. Architectural View

3.1 Introduction

We emphasize that different architectures for IoT are offered by various stakeholders, which reflects that there is no standardized architecture approved by an authorized body. This lack of standardized architecture contributes to the fuzziness that obscures a clear definition of an IoT system.

In this chapter we will present the minimal architectural components that an IoT system must possess. These architectural components can be derived from the requirements that an IoT system must fulfill.

We have reviewed the suggested IoT architectures offered by various projects, academic and industrial bodies. For the purposes of this paper, we focused on the architectural model offered by the IoT-A (Bassi, et al., "Enabling Things to Talk," 2013; IoT-A, "Internet of Things Architecture," 2011) and CASAGRAS (CASAGRAS, "Final Report," 2009) projects. We present here the minimal set of architectural components identified by this project. In the future, other components can be added to the list based on the application scenario.

3.2 Description of Architectural Components

The hardware unit in an IoT system falls into one or more of the following categories:

- ✓ Sensors/actuators
- ✓ Processing units
- ✓ Storage units
- ✓ Communication units

Having identified categories of hardware, we must add the software, middleware components and associated protocols which provide the means of linking and driving the hardware and provide service discovery support to constitute a fully operational system or systems.

The generic IoT scenario can be identified with that of a generic user that needs to interact with a (possibly remote) physical entity. In this short description we have already introduced the two key actors of the IoT, the "user" and "physical entity" (CASAGRAS, "Final Report," 2009).

I. User

A person or some kind of active digital entity (e.g., a service, an application or a software agent) that has a goal. The attainment of the goal is achieved via interaction with the physical environment. This interaction is mediated by the IoT.

II. Physical entity

A "physical entity" may be defined as a discrete, identifiable part of the physical environment which is of interest to the user for the attainment of his/her goal. Physical entities can be almost any object or environment, from humans or animals to cars, from store or logistic chain items to computers, from electronic appliances to closed or open environments. Physical entities are represented in the digital world via a virtual entity. There are many kinds of digital representations of physical entities: 3D models, database entries, objects (or instances of a class in an object-oriented programming language), even a social network account could be viewed as such a representation. In the IoT context, virtual entities have two fundamental properties:

- They are digital entities that are associated with a single physical entity that they represent. While ideally there is only one physical entity for each virtual entity, it is possible that the same physical entity can be associated with several virtual entities, e.g., a different representation per application domain or per IT system. Each virtual entity must have one and only one ID that identifies the represented object. Digital entities can be either active elements (e.g., software code) or passive elements (e.g., a database entry).
- Ideally, digital entities are synchronized representations of a given set of aspects or properties of the physical entity. This means that relevant digital parameters representing the characteristics of the physical entity can be updated upon any change of the physical entity. Conversely, changes that affect the virtual entity could manifest themselves in the physical entity.

Augmented entity is defined as the composition of a physical entity and its associated virtual entity. Any changes in the properties of an augmented entity have to be represented in both the physical and digital world. This is what actually enables everyday objects to become part of digital processes.

III. Device

A "device" is used to achieve the association between virtual and physical entity. This is done by embedding, attaching or simply placing the device in close proximity to the physical entity. Devices provide the technological interface for interacting with or gaining information about the physical entity. By so doing the device actually enhances the physical entity and allows the latter to be part of the digital world. A device thus mediates the interactions between physical entities (that have no projections in the digital world) and virtual entities (which have no projections in the physical world), generating a paired couple that can be seen as an extension of either one. Devices are thus technical artifacts for bridging the real world of physical entities with the digital world of the Internet. This is done by providing monitoring, sensing, actuation, computation, storage and processing capabilities in the device.

From a functional point of view, devices can belong to any of the following types.

- **Tags –** One of the characteristics of IoT is ubiquity, which can be realized through unique identification of the "things" that are connected to the Internet. This unique identification is done by attaching tags on the "things." Tags are used by specialized sensors typically known as readers. Their sole purpose is to facilitate an identification process. RFID is a perfect solution for providing this unique identification of "things." The transponder or tag of an RFID is used to carry data, which is located on the object to be identified. This normally consists of a coupling element (such as a coil or microwave antenna) and an electronic microchip, less than one-third millimeter in size. Tags can be passive, semi-passive or active, based on their power source and the way they are used, and can be read-only, read/write or read/write/re-write, depending on how their data is encoded. Tags do not need a built-in power source, as they obtain the energy they require to function from the electro-magnetic field emitted by readers.
- An **interrogator or reader** reads the transmitted data (e.g., on a device that is handheld or embedded in a wall). Compared with tags, readers are larger, more expensive and power-hungry. In the most common type of system, the reader transmits a low-power radio signal to power the tag (which, like the reader, has its own antenna). The tag then selectively reflects energy and thus transmits some data back to the reader, communicating its identity, location and any other relevant information. Most tags are passive, and activated only when they are within the coverage area of the interrogator. While outside this area, they remain dormant. Information on the tag can be received and read by readers and then forwarded to a computer database. Frequencies currently used for data transmission by RFID typically include 125 kHz (low frequency), 13.56 MHz (high frequency) or 800-960 MHz (ultra-high frequency). RFID standards relate both to frequency protocols (for data communication) and data format (for data storage on the tag).
- **Sensors** provide information about the physical entity they monitor. Information in this context ranges from the identity of the physical entity to measures of the physical state of the physical entity. Like other devices, sensors can be attached or otherwise embedded in the physical structure of the physical entity or be placed in the environment and indirectly monitor entities. An example of the latter is a camera that recognizes people's faces. Information from sensors can be stored for later retrieval.
- **Actuators** can modify the physical state of a physical entity. Actuators can move (translate, rotate, etc.) simple physical entities or activate/deactivate functionalities of more complex ones.

IV. Sensor Operating Systems

Most operating systems (OS) that may be used for IoT were designed for wireless sensor networks (WSN) like TinyOS and Contiki. But, practically, it seems that most of the OSs that were designed for use in WSN fail to meet one or more of the requirements of IoT. The developers of RIOT claim that they've bridged this gap of OS requirements between WSN and IoT. We will discuss all three OSs.

OSs for sensor nodes follow either one of two different design concepts, *event-driven* and *multi-threaded*. In event-driven systems every action an OS has to perform is triggered by an event (e.g., a timer, an interruption indicating new sensor readings or an incoming radio packet). The multi-threaded OS multiplexes execution time between the different tasks, implemented as threads. While switching from one thread to another, the current context has to be saved and the new context must be restored.

In this section we will first present the features of TinyOS and Contiki. Then we will discuss the features of RIOT and what its developers believe to be the IoT system requirements missing from the WSN OS but included in RIOT.

TinyOS

TinyOS is composed of a scheduler and a series of modules (Levis et al., "TinyOS," 2005). The application programs and modules are compiled together as a system. TinyOS executes operations based on events, and the event module allows the subsequent operations to run in a lesser space. In TinyOS, when an event is triggered, all the tasks related to the event that send out the signal are executed rapidly. After the event transpires and all related tasks are accomplished, the untapped central processing unit (CPU) reverts to SLEEP mode rather than actively searching for the next dynamic event. The event-driven mode of TinyOS makes effective system use of CPU resources. TinyOS uses three associated properties to manage power consumption. First, every part of the equipment can stop itself by calling the command StdControl.stop. Secondly, TinyOS will check the I/O pin and the control register of the processor to identify the processor's state through the component HPL Power Management. Last, the timer of TinyOS can work in the lowest power-cost mode, which most processors run in their power-down mode. TinyOS tasks are deferred function calls and are placed in a simple first-in, first-out (FIFO) task-queue for execution. TinyOS tasks are taken sequentially from the queue and are run to completion. Once running, the TinyOS task cannot be interrupted (preempted) by another TinyOS task. Event handlers are triggered in response to a hardware interrupt and are able to preempt the execution of a currently running TinyOS task.

CONTIKI

Contiki is an open source, network-able, multi-tasking, real-time OS developed for portable and memory-constrained embedded systems. It was released on March 10, 2003, by Adam Dunkels, a researcher at the Swedish Institute of Computer Science (Dunkels, "Contiki," 2004). The OS has a very versatile base system that provides multitasking and TCP/IP networking along with additional libraries for extra functionality, which led to its adoption for many different uses. With its ability to reprogram and update a network, the OS is a common choice for networks of embedded sensors.

The main features that Contiki emphasizes are its minimalistic, event-driven kernel with optional preemptive multithreading, native TCP/IP stack support, dynamic program loading and unloading and small memory requirements. Contiki's kernel is event-based, making it completely responsive to real-time events. This feature classifies it as a real-time operating system. It provides only the basic functions of CPU multiplexing and message passing to programs. It is very similar to the operation of the TinyOS kernel in that a process will only execute when a corresponding event triggers it. If an event occurs, it will trigger an event handler which runs a process to completion, and finally returns control back to the kernel. This

design has the benefits of resulting in more compact code and requiring less memory than a thread-driven kernel, which must store and keep track of a stack for each thread.

Nonetheless, this design has many downsides. First, the code for event-driven kernels is designed like a state machine, which is written very differently from the more traditional ways of writing code. Most importantly, there are no wait() statements or preemption of processes in event-driven kernels, which becomes a major issue for long-running computations. For example, if the system were to compute private or public key encryptions, which is a considerable computation, no other events would be able to grab hold of the kernel and CPU resources, no matter how urgent, until the encryption task had been completed.

Contiki addresses this issue by implementing multi-threading as a library on top of the kernel. By implementing multithreading as an optional library, only programs that wish to incorporate them pay the extra memory and program costs. Dunkels calls the implementation he created protothreads. Protothreads are a stackless, small memory thread design comprised of a single C function that only requires 2 bytes of RAM per thread to record its state. What the library really does is provide a context of blocking and preemption on top of the event-based kernel. It is an abstraction of the event-based operation of the kernel that allows sequential program flow without having to write complex state machine code or a full-blown multithreading program. The real beauty of protothreads is that the library is pure C with no architecture-specific code. They can be implemented with or without an OS and have been widely used outside of the Contiki OS.

A Contiki system is partitioned into two parts: the core and the loaded programs as shown in Figure 17. The partitioning is made at compile time and is specific to the deployment in which Contiki is used. Typically, the core consists of the Contiki kernel, the program loader, the most commonly used parts of the language run-time and support libraries and a communication stack with device drivers for the communication hardware.

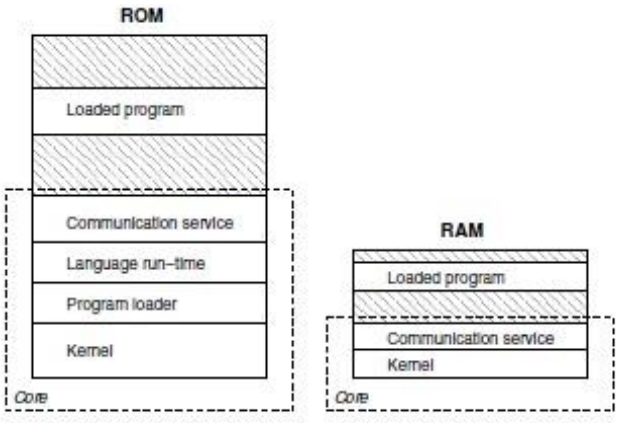


Figure 17. Contiki operating system partitioning

RIOT

The developers of the RIOT OS considered the requirement of an IoT system to be different from WSN (Baccelli et al., “RIOT OS,” 2013). They state that an IoT OS needs to fulfill the following requirements:

- ✓ Minimal requirements to memory (RAM and program memory) and computing power
- ✓ Ability to run on constrained hardware without more advanced components like a memory management unit (MMU) or a floating-point unit (FPU)
- ✓ Support to a variety of hardware platforms
- ✓ High degree of energy efficiency
- ✓ Standard programming interface
- ✓ Support for high-level programming languages
- ✓ An adaptive and modular network stack
- ✓ Reliability

Ideally, the capabilities of a full-fledged OS (e.g., Linux, Unix, BSD or Windows) are desirable on all IoT devices. The just-cited OSs are appealing because they are developer-friendly. They possess numerous, available system libraries, network protocols or algorithms and near-zero learning curve in the sense that developers can code in standard C or C++. However, their minimal requirements in terms of CPU and memory do not fit constrained IoT devices powered by small micro-controllers.

On the other hand, the trade-offs that enable a typical lightweight OS targeting WSNs to run on the most constrained IoT devices make it significantly less developer-friendly and/or inappropriate on IoT devices that are less constrained. These points are illustrated in Table 1, which compares attributes of the cited OSs, where, P means: Supports Partially, N means: Doesn't Support and Y means: Supports Fully.

Table 1. Comparison of different operating systems

OS	Min RAM	Min ROM	C Support	C++ Support	Multi-Threading	Modularity	Real-Time
Contiki	<2KB	<30KB	P	N	P	P	P
Tiny OS	<1kB	<4kB	N	N	P	N	N
Linux	~1MB	~1MB	Y	Y	Y	Y	P
RIOT	~1.5kB	~5kB	Y	Y	Y	Y	Y

OS	IPv6	TCP	6LoWPAN	RPL	CoAP
Contiki	Y	P	Y	Y	Y
Tiny OS	N	P	Y	Y	Y
Linux	Y	Y	Y	P	P
RIOT	Y	Y	Y	Y	N

Features of RIOT:

RIOT OS aims at bridging the gap we observed between operating systems for WSNs and traditional full-fledged OSs currently running on Internet hosts. The key design goals for RIOT OS were energy-efficiency, small memory footprint, modularity and a developer-friendly programming interface, which make RIOT the best choice to power the widest spectrum of IoT devices.

The system is based on a microkernel and offers real multi-threading. In order to provide maximum modularity, RIOT implements the micro-kernel architecture inherited from FireKernel, a kernel designed to fulfill strong real-time requirements for emergency scenarios. RIOT thus supports multi-threading and real-time in that it features zero-latency interrupt handlers, and minimum context-switching times combined with thread priorities. In order to achieve maximum energy efficiency, RIOT introduces a tickless scheduler able to function on constrained devices. Most schedulers use timers to wake up periodically and check if something needs to be done: this is the timer tick. However, if the processor is idle, it has to wake up from its power-saving sleep state every timer tick, even when there is nothing to do. This behavior is thus not desirable for energy-constrained systems. Moreover, with most microcontrollers, a timer interrupt cannot wake up the processor from deep-sleep mode (only external interrupt sources can). Hence, using a timer tick prevents using deep-sleep mode. By avoiding timer-tick dependency with its tickless scheduler, RIOT significantly decreases the energy consumption of the system.

V. Middleware

Middleware is software that resides between RFID interrogators and enterprise software (O'Connor, "Guide to RFID Middleware," 2010; see also, Bandyopadhyay et al., "Role of Middleware for IoT," 2011). Middleware serves a number of key functions: It configures and manages hardware, such as interrogators, so they operate optimally. And it processes tag data, filtering out duplicate tag reads and aggregating the data that's passed along to back-end applications. Middleware can run on a dedicated computer at each facility where RFID interrogators are deployed, on each interrogator or on a networking appliance where the technology is used. These solutions are often called "edgeware" because they're deployed close to the edge of the network—at, for instance, a manufacturing facility or distribution center. Middleware can also be deployed at a data center with a wide area network (WAN) to communicate with the readers.

VI. Resources

Resources are software components that provide information about physical entities or enable the controlling of devices (Kong et al., "Model of Resource Addressing in IoT," 2010). Resources can be of two types:

- **On-device resources** are hosted on devices, that is to say, software that is deployed locally on a device. They include executable code for accessing, processing and storing sensor information, as well as code for controlling actuators.

- **Network resources** are resources available somewhere in the network, e.g., back-end or cloud-based databases.

A virtual entity can also be associated with one or more resources that enable interaction with the physical entity that the virtual entity represents.

Storages are a special type of resource that stores information coming from resources and that thus provides information about physical entities. This may include location and state-tracking information (history), static data like product type information, and many other properties. Since storages are resources, they can be deployed either on-device or in the network.

- **On-device storages** typically store information about one or only a few physical entities, e.g., the physical entity they observe.
- **Network-based storages** aggregate information about a large number of physical entities.

Note that also human users can update the information in storage, since not all known information about an entity is, or even can be, provided by devices.

VII. Service

Service provides a well-defined and standardized interface, offering all necessary functionalities for interacting with physical entities and related processes. All this is done via the network. Services expose the functionality of a device by accessing its hosted resources.

The components of an IoT system can be categorized into the following three layers:

I. **Physical layers –** in which the physical objects or things are identified and rendered functional components of the IoT through the use of object-connected data carrier technologies, including RFID. The objects so identified may also be grouped or networked to fulfill particular application needs. Devices with additional functionality, in the form of sensory, actuation, global positioning and local communications capabilities, may be used to achieve network structures as well as single-device operation. The components that belong in this physical layer are tags, sensors, actuators and the physical entity.

II. **Interrogator-Gateway Layer –** providing effectively the interfaces between the object-connected devices and information management systems. Fixed, broadband and mobility communication technologies will yield the connectivity required for the IoT. Networking of interrogators and gateway devices may also be seen as an important infrastructural feature in this layer and an important contributory feature within the IoT. Interfacing with respect to actuation and control devices within real-world applications is a further important feature of this layer. This layer contains the interrogator or reader part of the architectural components mentioned above.

III. **Information Management, Application and Software Layer –** interfacing with the interrogator-gateway layer, the information management layer provides the functional platform for supporting applications and services. This layer contains the

application and the software parts of the architectural components. Among the architectural components mentioned above the ones that belong to this layer are Middleware, Resources and Service.

The IoT can have three models based on the level of complexity and intelligence of the system. These three models are:

- A model based specifically on read-only RFID data carriers
- Additional Object Connected data model based specifically on RFID (ostensibly with read-write functionality and added data-carrying capability)
- Additional Object Connected data model based on RFID and other Edge technologies (ostensibly covering sensory data capture, extended data-carrying capability and other attributes such as location or positioning facilities)

The most basic model for IoT has data carriers which are essentially passive RFID tags carrying unique identifiers, with each tag having the capability for interrogation and response via a wireless channel. There is no intrinsic processing capability within the tags and no facility for communications between tags.

Applications using these data carriers rely upon the identifier as the means of locating remotely stored information about the item to which it is attached. The tags are interrogated using reader or interrogator devices that have the facility to communicate wirelessly with the tags and further communicate with an application.

From these discussions we can demarcate a line for a system to be considered as an IoT system. The lowest level of complexity for an IoT system is a “thing” attached with a passive tag where the static data can be read from anywhere, and anytime through the use of an application hosted on the Internet.

Accordingly, we can set the following points for a system to be considered as an IoT system (Mahalle, “Identity Management Framework,” 2010):

- The physical entity (the “thing”) has to be uniquely identifiable. Through this unique identification, the physical identity has to be accessed from anywhere, anytime. The lowest level of information that can be obtained from this uniquely identified object is a statically stored data on the device that is associated with the physical entity.
- The uniquely identified physical entity has to be connected to the Internet. A thing that is uniquely identified and not connected to the Internet or connected to an intranet or extranet cannot be considered as part of the IoT system. A uniquely identified physical object that is connected to the intranet or extranet will have to be considered as an intranet or extranet of things. The connectivity of IoT goes beyond extranet.

3.3 Addressing

The first step towards the realization of IoT is the unique identification or addressing of the “things” connected to the Internet. Mainly, IPv6 and EPC are considered useful in uniquely identifying “Things” (Piispanen, “EPC and IPv6-based discovery services,” 2011). In this section we will review addressing schemes based on IPv6 and EPC.

3.3.1 IP for Things

If, in a future IoT, everyday objects are to be addressed and controlled via the Internet, then we should ideally not be resorting to special communications protocols as is currently the case with RFID. Instead, things should behave just like normal Internet nodes. In other words, they should have an IP address and use the Internet Protocol (IP) for communicating with other smart objects and network nodes. And due to the large number of addresses required, they should use the new IPv6 version with 128-bit addresses. The benefits of having IP-enabled things are obvious, even if the objects in question are not going to be made globally accessible but instead used in a controlled intranet environment. This approach enables us to build directly on existing functionality such as global interoperability, network-wide data packet delivery (forwarding and routing), data transport across different physical media, naming services (URL, DNS) and network management. The use of IP enables smart objects to use existing Internet services and applications and, conversely, these smart objects can be addressed from anywhere since they are proper Internet participants. Last but not least, it will be easy to use important application layer protocols such as HTTP. IPv6 also provides the desirable capability of automatic address configuration, enabling smart objects to assign their own addresses.

Until recently, however, the prospect of full IP support for simple things appeared illusory due to the resources required – such as processor capacity and energy – and thus the costs involved. Instead, it was suggested to connect smart objects to the Internet indirectly via proxies or gateways. But the disadvantage of such non-standardized solutions is that end-to-end functionality is lost because standardized Internet protocols would be converted to proprietary protocols over the last few meters. Gateways would also generate added complexity, making installation, operation and maintenance time-consuming and costly. However, there are now not only 16-bit microcontrollers with sufficient storage that require less than 400 µW/MIPS, but also TCP/IPv6 stacks that can operate with 4 kB RAM and 24 kB flash memory. Equally important are wireless communications standards such as IEEE 802.15.4 that cover the layers below IP and consume relatively little power – ZigBee implementations require approximately 20 to 60 mW(for 1 mW transmission power, a range of 10 to 100 meters and a data transmission rate of 250 kbit/s). Whenever possible, the wireless unit is being used for short periods of time only in order to save energy. This approach enables AA batteries to provide a modest level of computing power and wireless communication that is nevertheless sufficient for many purposes over many months.

The opportunities that this scenario opens up have recently led to companies and standards committees adopting various measures. At the end of 2008, Atmel, Cisco, Intel, SAP, Sun Microsystems and other companies founded the “IP for Smart Objects” (IPSO) corporate alliance to promote the implementation and use of IP for low-powered devices such as radio sensors, consumption meters and other smart objects. More specifically, the “IPv6 over Low Power Wireless Area Networks” (6LoWPAN) working group set up by the Internet Engineering Task Force (IETF) is addressing the problem of supporting IPv6 using the 802.15.4 wireless communication standard. This is a technical challenge because the maximum length of 802.15.4 data frames is only 127 bytes due to the lower data rate, higher susceptibility to failure and bit error rate of wireless communications. The IPv6 packet header alone is 40 bytes long (primarily due to the source and target addresses each being 16 bytes long), and unfragmented IPv6 packets can be up to 1280 bytes long.

To make IPv6 communications function efficiently in wireless networks, a protocol modification layer has been defined that essentially deals with four issues: embedding IPv6 packets in 802.15.4 frames, fragmenting long packets to fit these frames, statelessly compressing packet headers (typically to just 6 bytes), and forwardingIPv6 packets via multi-hop wireless routes. It is possible to compress the IPv6header so drastically because 802.15.4 nodes communicate mainly within their own wireless network, and therefore most of the information can be reconstructed from the general context or the surrounding 802.15.4 frames and considerably shorter local addresses can be used. The working group's proposal has now been published as proposed Internet standard RFC 4944.

3.3.2 Electronic Product Code (EPC)

Electronic Product Code (EPC) was developed by Auto-IDCenter (currently Auto-ID Labs) at MIT (Brock, "Electronic Product Code," 2001). In the supply chain business case the manufacturer of the product is responsible for issuing a unique EPC number for each RFID tag (Thiesse, "Real-world EPC Networks," 2009). The EPC number can be either a 64-bit or 94-bit identifier. EPC is currently managed by EPCglobal.

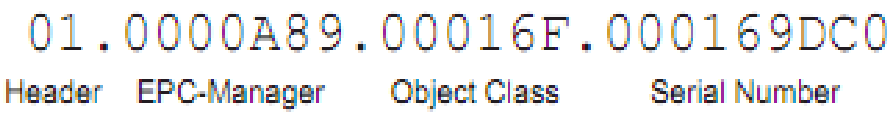


Figure 18. EPC number format

An EPC number consists of four parts. The first part is the header which defines the version of the EPC number used. The second part is the identifier of the EPC-Manager that assigns the EPC number to the object. The third part is the object class identifier that essentially defines the product type of the object. The last part is reserved for the unique serial number that identifies the product from other products among the same object class.

3.3.2.1 EPCGLOBAL NETWORK

EPCglobal Architecture Framework (EPC Network) is a concept that allows the storing and querying of data related to objects identified with EPC numbers. The structure of the EPC network is shown in **Figure 19.** EPC global network architecture

. The data that is stored in local databases consists of EPC read events that are created when an RFID tag is read. The read events typically contain the EPC number of the product with the time and location of the read event. Other data such as sensor measurement data can be included with the EPC read event as well.

The EPCglobal network consists of three different components:

- ✓ Object Name Service (ONS)
- ✓ EPC Information Services (EPCIS)
- ✓ EPC Discovery Service (EPCDS)

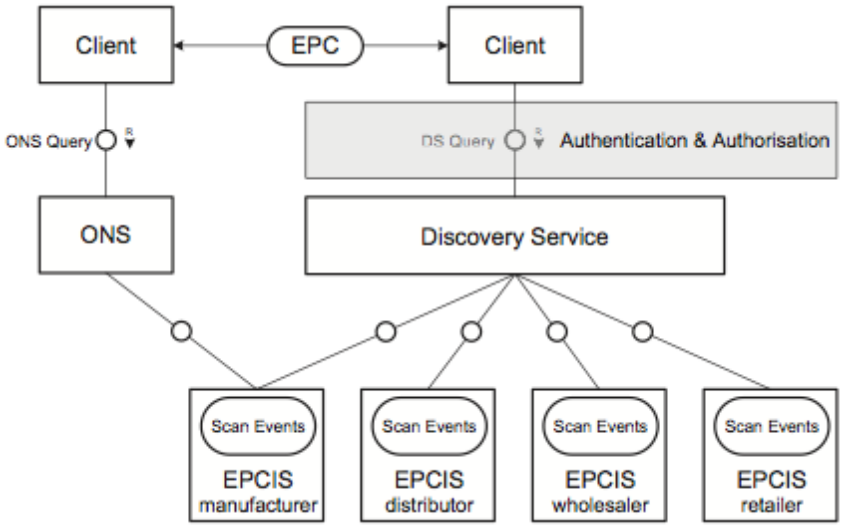


Figure 19. EPC global network architecture

Object Name Service

The Object Name Service (ONS) is based on Dynamic Name Service (DNS). ONS resolves information sources to an EPC number. Information sources can be websites, Web services or EPCIS repositories. The typical case is that the EPC number is resolved to the EPCIS repository of the manufacturer of the product. The ONS uses the first three parts of the EPC number to resolve the information source. This means it does not process the request at the serial number level, which means that the ONS cannot be used for retrieving information associated with a specific EPC number.

EPC Information Services (EPSIS)

EPC Information Services (EPCIS) is essentially a local database for holding the EPC read events for every company in the supply chain. EPCIS also contains a query interface so that EPC read events can be queried using the EPC number. EPCIS serves a simple repository and query interface but they don't offer any functionality to locate all the information available related to a specific EPC number.

The early specifications of the EPC network specification provided a mechanism for the OSN to resolve all the EPCIS repository addresses given an EPC number. This didn't take into account any of the privacy concerns related to EPC-related data. Because of this a different information system called EPC Discovery Service has been developed for the discovery and confidentiality of data.

When an RFID tag is read, the read event is stored in the local EPCIS repository. The first time a certain EPC number is read at a company, the EPCDS-service is also notified that the EPCIS repository holds data associated with the EPC number.

EPC Discovery Service (EPCDS)

EPC Discovery Service is the service that allows users to find all the data related to an EPC number (Lorenz et al., "Discovery Services in EPC Networks," 2011). The EPCDS follows a so

called Directory of Resources approach where EPCIS repositories register information about the availability of data corresponding to an EPC number at the EPCDS. The EPCDS informs the querying client which EPCIS repositories hold information related to the EPC number. The clients can then directly query the EPCIS repositories for data. In this scenario the EPCDS services also make sure that the querying client is authorized to request information about the given EPC number.

Bootstrapping

The sheer amount of data in IoT means that it is not viable to have one discovery service that covers all the EPC repositories in the world. A bootstrapping method must be developed that allows clients to find the correct EPCDS-service using just the EPC number of the object for querying. Several solutions have been suggested for this bootstrapping process.

The ONS-service could be used to resolve the EPCDS the product belongs to. The manufacturer of the product would essentially dictate to which EPCDS the product belongs to when issuing an EPC number for it. This is problematic since all companies in the supply chain may not be willing to distribute information to that EPCDS due to technical, political or economic reasons.

Another possible solution for the bootstrapping challenge is to build a peer-to-peer overlay network for collaboration of discovery services. This would, however, significantly complicate the EPCnetwork structure.

3.3.3 Choosing between EPC and IPv6

Both EPC and IPv6 can uniquely identify the "Things" in an IoT system. But according to researchers at the Silicon Valley World Internet Center, each of them cannot stand alone as an addressing mechanism for IoT due to each one's weakness (Vadhia et al., "IPv6 vs. EPC," 2004). They suggest that in order to have item-level identification and communication capability simultaneously, both EPC and IPv6 must be used together.

A comparison of EPC and IPv6 attributes is presented here in Table 2.

Table 2. Comparison of IPv6 and EPC

	IPv6	EPC
Objects to identify	Network interfaces	Physical objects
Primary application	Routing address	Pointer to information
Address allocated by	Network manager	Item manufacturer
Unique identifier	Yes	Yes
Identifier length (bits)	128	64, 96, other
Can identifier change?	Yes	No
Area of difficulty	Mobility	No location information

Mapping EPC TO IPV6

EPC codes can be used as unique identifiers, but they cannot be used as routing addresses. Similarly, an IPv6 address can be used as a routing address but not as an item identifier at the same time. One suggested solution is to use both EPC and IPv6 in the IoT system so as to be able to communicate and uniquely identify things by mapping EPC to IPv6.

An IPv6 address consists of a 64-bit network prefix and a 64-bit Extended Unique Identifier (EUI-64). EUI-64 is an extension of the MAC addresses in the physical layer of network communication. A simple way to construct an IPv6address is to use the network prefix in the readers' network and append the EPC number to it, replacing the EUI-64 part of the IPv6 address.

3.4 Programmability

A programmability feature enables devices to users' specific needs and demands (Littman, "IoT: Path to Programmable World," 2014; see also Wasik, "Welcome to Programmable World," 2013). At the simplest level, a programmable device is one that can take on a variety of behaviors at a user's command without requiring physical changes. For example, a programmable synthesizer can sound like a number of different instruments depending on the player's preference, while a traditional piano can sound only the way it was physically designed to sound.

This is the language of the future: tiny, intelligent things all around us, coordinating their activities. Coffeepots that talk to alarm clocks. Thermostats that talk to motion sensors. Factory machines that talk to the power grid and to boxes of raw material. A decade after Wi-Fi put all our computers on a wireless network and half a decade after the smartphone revolution put a series of pocket-size devices on that network we are seeing the dawn of an era when the most mundane items in our lives can talk wirelessly among themselves, perform tasks on command and provide us with data we've never had before.

Imagine a factory where every machine, every room, feeds back information to solve problems on the production line. Imagine a hotel room where the lights, the stereo and the window shade are not just controlled from a central station but adjust to your preferences before you even walk in. Think of a gym where the machines know your workout as soon as you arrive or a medical device that can point toward the closest defibrillator if you have a heart attack.

Devices are only meaningfully programmable in so far as end users, the customers who want to communicate their preferences to these devices, can program them. The transition to a programmable world only truly begins when control of devices becomes accessible to those with modest technical knowledge. Programmable devices are supposed to make our lives easier, but this is only the case if getting a device to perform a task is less challenging and time-consuming than completing the task ourselves. Well-designed devices must have accessible, intuitive interfaces that make it possible for users to communicate their intentions without having to learn any sophisticated programming languages.

This communication was once done through buttons and knobs, but nowadays it often happens via remote controls, touch-panel displays or smartphone apps. And while a good user interface

simplifies programming, it also carries with it the cost of learning how to use the interface itself. If users need to learn different interfaces for their vacuums, their locks, their sprinklers, their lights and their coffeemakers, it's difficult to persuasively argue that their lives have been made any easier. In that case, all that the gadgetry has done is replace a user's physical tasks with digital ones.

One promising solution is to combine the interfaces for several products in the same system. The advent of the smartphone has been a boon for programmable devices because it offers a standardized, portable platform for users to interact with their devices. Even something as seemingly simple as a light bulb can now be made to change colour and brightness through an iPhone app. However, the smartphone doesn't completely solve the problem if the interface for each device is different: end users still have too many remote controls, they're all just contained in a phone-sized drawer. Some manufacturers have moved beyond simply putting different apps on the same screen to building integrated platforms that can manage multiple devices. Belkin's "WeMo Home Automation" line, for example, allows users to control a range of WeMo products, from heaters to lights to electronics, wirelessly through one app.

Another promising step forward is "If-This-Then-That" (IFTTT), one of the most successful and advanced attempts at a simple, holistic end-user programming system. IFTTT lets users coordinate over 100 Web services, such as Facebook and *The New York Times*, through trigger-action programming. Users issue commands in the form of "if-then" statements called "recipes," and the program executes each action automatically when the trigger condition is met. For example, users can connect the Weather Channel with Gmail so that *if* there is rain in tomorrow's forecast, *then* they will receive an email with a weather update (Atzori et al., "Smart Things in the Social Loop," 2013).

3.5 Virtualization

In our connected world, and in the spirit of the IoT, more and more devices are becoming "always connected" and remotely controlled. Vending machines, power and water meters, and communication equipment – all of these small-to-medium size devices are becoming Internet-ready to reduce the cost of management, enable better repair and control and offer innovative consumer services like pay-as-you-go energy consumption. But when these systems require multiple functions or must be properly customized for different needs and markets, another CPU chip or virtualization is required to securely run multi-function software stacks with no cross-influence (Rave, "Virtualization's Impact," 2014).

While the IoT promises better living through connected devices and the data and insights they generate, it will also usher in a new era of privacy and security concerns. One area of increased security risk is the number and magnitude of new "attack surfaces" associated with IoT. IoT represents an interconnected ecosystem in which third-party information technology (IT) infrastructures of information providers, consumers and brokers are interwoven in a service-oriented manner with networks of devices/sensors and clouds providing computing, apps, storage and analytics. One weak link can expose the entire chain.

In order to foster the potential of IoT and minimize security risks, a new network paradigm centered on cloud-based networks is required (SiliconAngle, "Internet of Things Needs Network of Clouds," 2014). New technology enablers have progressively fostered virtualization at different levels and have allowed the various paradigms known as "Applications as a Service," "Platforms as a Service" and "Infrastructure and Networks as a Service."

The following are some of the benefits of a cloud-based IoT system:

- **Cloud-Based Virtual Network Overlays:** Private device networks are expensive, and securing the Internet outright is impossible. Cloud-based virtual network overlays leverage network virtualization and software-defined network (SDN) technologies to create private virtual device networks over the Internet.
- **Programmable Flows:** IoT is essentially a service-oriented architecture where data from connected devices is collected and can flow through series of real-time or demand-based computational, analytical or event-processing functions, many of which will be cloud-based. An SDN-enabled cloud network allows flows to be programmatically routed through the proper services.
- **Underlay Network and Cloud Agnostic:** IoT represents billions of connected devices running anywhere in the world across any type local connections. Virtual network overlays extend across any cloud datacentre and run over any local connection.
- **Borderless Admission Control:** Because IoT is really an ecosystem of interconnected organizations, people, processes and devices, there is no well-defined border. As a result, control of the devices, organizations, users and flows admitted onto the network must be an integral function.
- **Network Service Virtualization:** IoT requires network services to ensure security, visibility, compliance and control of connected devices, users, processes and data. Since such intelligence on every connected device is not viable, the IoT requires virtualized and distributed network services that can be deployed anywhere and "in-line" with data flows.
- **Security at Scale:** Enterprise IT organizations may deal with an attack surface of hundreds of thousands of devices. A single IoT network of connected devices can represent an attack surface of millions of devices. Cloud networks that incorporate deep packet inspection, Network Service Virtualization and policy can take advantage of the cloud's low-cost computing and big data infrastructure to provide a wide range of advanced security functions.

3.6 Web of Things

The Web of Things (WoT) is a concept and plan to fully incorporate every-day physical objects into the World Wide Web by giving them an Application Programming Interface (API), thus greatly facilitating the creation of their virtual profiles as well as their integration and reuse for various applications.

"The Web of Things is primarily an evolution of the Internet of Things where the primary concern has been how to connect objects together at the network layer: similar to the way the Internet addressed the lower-level connectivity of computers (layers 3-4 of the OSI model), the Internet of Things is primarily focusing on using various technologies such as RFID, Zigbee, Bluetooth or 6LoWPAN.2.5 Projects and Academic Activities." (Casteleyn et al., editors, *Web Engineering*, 2014).

Diverging from the traditional view of IoT, which gives everyday devices an IP address and makes them interconnected via the Internet, WoT enables them to speak the same language, so as to communicate and interoperate freely on the Web (Zeng, “Web of Things: A Survey,” 2011; see also W3C: “What is the Web of Things?” [no date]; “Web of Things,” Wikipedia, [no date]).

The WoT vision offers a scenario in which a collection of Web services can be discovered, composed and executed. This enriches the scope of traditional Web services by promoting the Web from only cyber-world services to both cyber-world and physical-world services. Furthermore, WoT actually is an ecosystem of services. It is not only about adding more services but more about orchestrating various kinds of services in a graceful manner, making the services more human-centric, intelligent and, ultimately, helpful.

The inclusion of services from smart things makes WoT different from the traditional Web. For example, the embedded tiny Web servers are not as powerful as traditional ones. The service may not be always as available as traditional ones due to intermittent connectivity. Furthermore, traditional Web protocols might not be suitable to provide services in the low-power and lossy networks consisting of resource-constrained smart things.

The smart things can be connected to the Web in two ways as direct integration and indirect integration, which are elaborated below.

Direct integration: To directly integrate things to the Web, it is first required that all the things must be addressable, i.e., everything must have an IP address, or must be IP-enabled when connected to the Internet. WoT also requires connectivity and interoperability at the application layer. Web servers shall be embedded such that things can understand each other through the Web language specified by Web standards. With the development in both communication and computation technologies, it is likely that more devices will become IP-enabled and can be embedded with a Web server. Those devices can be directly integrated into the Web and abstracted as Web services.

Indirect integration: Not all devices can be powerful enough to be embedded with a Web server. Some devices are too limited to allow them to be Web server embedded, such as RFID tags. On the other hand, sometimes there is no need to directly integrate all the smart things (e.g., sensor nodes in a sensor network) into the Web due to considerations of cost, energy use and security vulnerability. For both cases, a different pattern, indirect integration, can be adopted. In this pattern, an intermediate proxy would be located between the smart things and the Web. The proxy is usually called a “smart gateway.” To the smart things (inward), the smart gateway communicates with the smart things (e.g., reading from RFIDs) and therefore shall understand the proprietary protocols of the smart things. To the Web (outward), it abstracts the proprietary protocols or native APIs of smart things and offer uniform accessible Web APIs over the Web.

3.7 IoT-aware Process Modeling Concept (IAPMC)

Standard business process modeling notations provide graphical symbols that enable the specification and documentation of business processes and workflows. Some of the graphical notations also offer execution semantics for supporting the direct execution of specified process models through a process execution engine. So far, modeling notations are used for purely software-based environments such as specialized enterprise resource planning (ERP) systems. The lowest level, which represents the actual information in the form of data and native

interfaces, is not considered, as this is usually one single resource component (e.g., an SQL database). Thus, features brought through the individual devices and device types of this lowest level are neither included in current business process modeling approaches, nor in the business process execution process.

IoT envisions a world in which clearly-identifiable devices are connected with each other in a Web-like structure. With the help of services, the physical objects can interact with each other so that they can become active participants of future business processes. The direct integration of real-world technology promises entirely new and changing business processes that currently cannot be completely represented and executed using existing process modeling techniques and notations. For the area of business process modeling, new requirements are needed, which shall be covered by introducing the IAPMC.

Table 3 offers a summary of coverage of IoT characteristics by existing BPM notations.

Table 3. Coverage of IoT Characteristics by existing BPM

IoT characteristics	BPMN 2.0	eEPC	UML 2.3
Entity-based concept	Partly	No	Partly
Distributed execution	Partly	No	Partly
Interactions	No	No	No
Distributed data	Partly	No	Partly
Scalability	Partly	No	Partly
Abstraction	Yes	No	Partly
Availability / Mobility	No	No	No
Fault tolerance	Partly	No	Partly
Flexibility/Event based	Yes	Partly	Partly
Uncertainty of Information	No	No	No
Real-time	Yes	No	No

In order to describe business processes that include IoT-technology and to match the IoT domain model, BPMN 2.0 is the most suitable state-of-the-art approach and provides coverage for more IoT specific properties than other approaches.

IAPMC model addresses some of the weaknesses of the BPMN 2.0 and allows it to be suitably implemented to an IoT environment. But it is worth noting that IAPMC does not solve all the weaknesses of BPMN2.0 as this IoT-aware business process model is still under development.

An initial version of IAMC and its BPMN 2.0 implementation rests on the following seven pillars:

- ✓ IoT Activity
- ✓ Sensing Activity
- ✓ Process Resources
- ✓ Physical Entity
- ✓ Real World Data Object / Store
- ✓ Mobility Aspect
- ✓ IoT Process Ratios

The IAMC model introduces the above-mentioned IoT specific features to the existing BPMN 2.0 model so as to fill the gap that BPMN 2.0 has in modeling IoT-related processes.

3.8 Recap

In this chapter, some issues related to the construction of a software platform have been considered. The " IoT Platform" is a topic of particular relevance, in fact it is the platform that will induce the value of IoT solutions. Small IoT solutions have limited needs on software platforms, but the more we move towards larger systems the more we need an IoT platform that is capable of supporting many functionalities and offering a high level of programmability. Programming the world means that smart objects can be controlled by a highly distributed platform by means of APIs or protocols. The more objects to control the more functions and data will be available. This yields to interoperability and standardization problems as well as to a management problem. It will become impossible to use traditional ways of management when the entity to be managed will number in the billions. So the platform should also be able to support autonomics and self-organization capabilities. Virtualization will also a cause the replication of functionalities to be managed in non-traditional ways.

The current software infrastructure of IoT seems to be too immature to cope with these problems. It is understood that the definition of an IoT software infrastructure is a major research and development issue.

4. Interaction Paradigms

This document contains a description of some protocols commonly used in IoT applications. It also relates these protocols to different interaction paradigms in order to provide a comparison between the different capabilities and mechanisms to allow the interactions between (smart) objects.

Some major interaction paradigms (e.g., Client – Server, P2P, Message Passing and others) are considered in the context of IoT.

4.1 Some Major Interaction Paradigms

In this document the way in which parts and components of an IoT system interact and cooperate is seen as the "interaction paradigm." Actually, a large IoT system could be implementing several interaction paradigms between its components.

The client–server (C-S), and the peer to peer (P2P), interaction paradigms are represented in

Figure 20. Client–Server and Peer-to-Peer interaction paradigms**.**

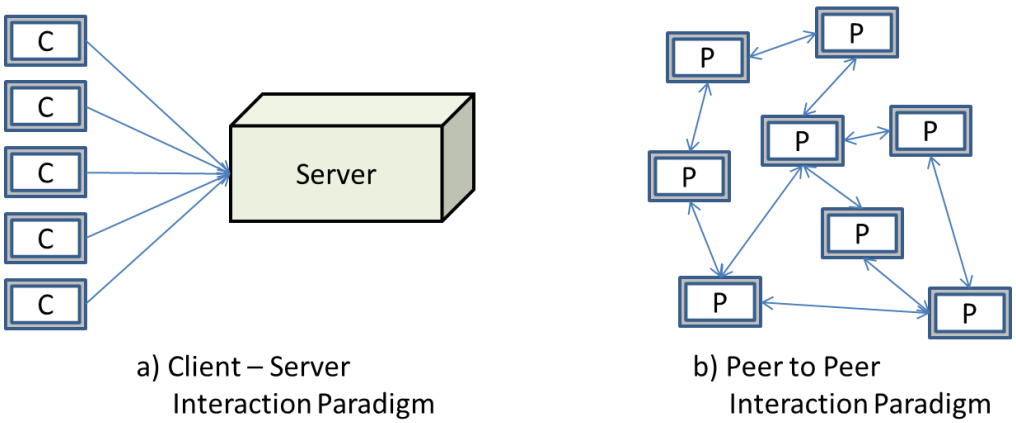


Figure 20. Client–Server and Peer-to-Peer interaction paradigms

The C-S paradigm is based on a very simple interaction between the clients and the server; a client sends a request (essentially a structured message) to a server and expects (non-blocking) a response from the server. **Figure 21.** The Client-Server interaction paradigm depicts the simple interaction.

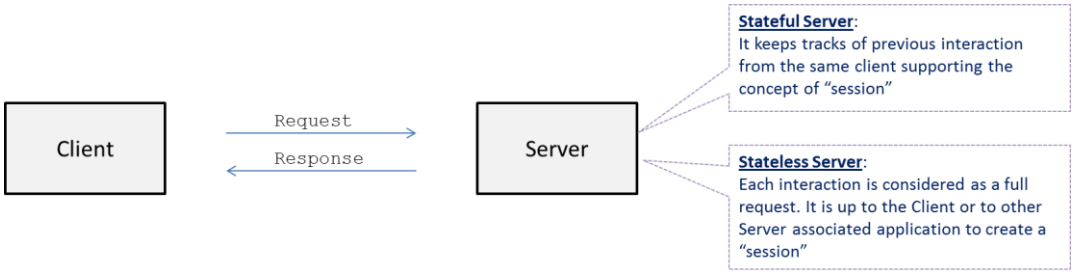


Figure 21. The Client-Server interaction paradigm

The Server could be Stateful or Stateless; the difference is whether the system keep track of the previous interactions with clients and has a finite state machine associated to the interactions going on. A stateful server is more complicated to manage especially if many clients are requesting in parallel the functions of the server. There is the REST architectural proposition related to Web services and architecture that is promoting with a lot of success the idea of having stateless server as a principle of the architectural design.

Some interaction paradigms in proper sense are:

- Tasks and channels.
- Message passing.
- Data Parallelism.
- Shared Memory.

The message passing model is represented in Figure 22. It is based on a very simple organization in which a sender forwards messages by means of a queue to a recipient.

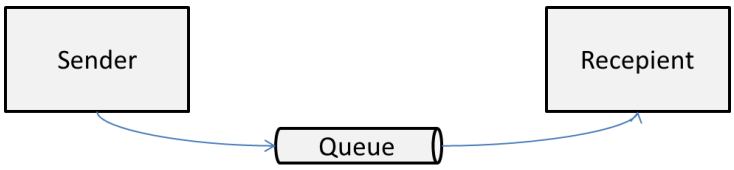


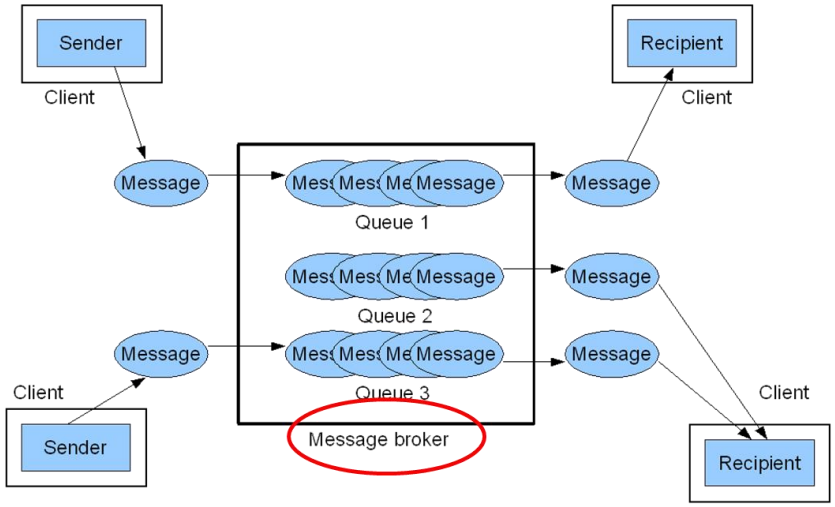
Figure 22. Message passing model

Obviously the different mechanisms have advantages and drawbacks. A deeper analysis is available in several sources (Foster, "Designing and Building Parallel Programs," 1995; and Kubiatowicz, "Integrated Shared-Memory and Message-Passing Communication in the Alewife Multiprocessor," 1998). A short analysis of pros and cons is provided here:

- Message Passing: MP paradigm can be interpreted as an "interrupt with data", i.e., events are queued for processing together with associated data (they could also be large bulk of data). Receiving tasks can then operate on data received when the message is taken from the queue. There is the need of an explicit treatment of communication (e.g., "send message to task n") and management of data (data are copied from data structures in the source of the message, transmitted to a queue and then copied to a data structure at the sink destination). Assembling and disassembling of messages can be cumbersome.
- Shared Memory: one of the major advantages is that the communication details between programs and the "shared memory" are hidden to the programmers, freeing them from the problem of dealing with communication and (in the shared memory) with the needs to deal with data management (replication, caching, data coding and encoding, optimization of distribution of data). The programmers can actually focus on the parallelism and the issues to solve. However, the programs have to adopt a sort of polling in order to synchronize with new data and new events (and this can be time and resource consuming), in addition data management is optimized having in mind a system optimization, specific applications needing other optimization policy will find difficult to implement specific application oriented policies. This could be an example of abstraction that eases programming, but sometimes it is detrimental for applications that need to do "different" things.
- Tasks and Channels: this paradigm can be seen as a variation of the message passing. According to one source (Foster, "Designing and building parallel programs,, 1995), the "task/channel" paradigm enforces the programmer to better conceptualize the communication of a parallel and distributed program. This could be provide advantages in design a large distributed system, but it also introduces more overhead in the need to control individual channels and in creating a relationship between channels and related tasks.
- Data Parallelism: while this paradigm can lead to significant advantages when the same instructions can be applied to different data, its general application is not easy, in fact not all the application domain offer problems that can be effectively solved with this paradigm. Actually the major drawback of this paradigm is its applicability to more restricted set of applications compared to the previous ones.

In large Message Passing systems, senders and recipients share a common infrastructure made out of queues, these can be organized as a sort of Message Broker in charge for receiving and dispatching (in an asynchronous manner) messages between sources and sinks

(**Figure 23.** A Message Passing MP System).



http://middleware.sovibox.fi/message-brokers/

Figure 23. A Message Passing MP System

There are at least three important features:

- The model can be synchronous or asynchronous, i.e., in the first case, the sender waits for the receiver to deal with the message, in the second case once a message has been sent, the sender can proceed with its computation. The choice of synchronicity and asynchronicity depends on the applications requirements. The introduction of queues in order to store messages for later processing is a case in which asynchronous mechanisms are implemented.
- Routing, the system (especially by means of the Broker Functions) can route the messages even if the sender has not provided a full path to the destination.
- Conversion, the Broker functions can also make compatible different messages format in such a way to support the interoperability between different messaging systems.

Communication between the Sender and Receivers is not “brokered,” because the event message manager enables an asynchronous but full communication between the two parties.

Other interaction mechanisms are emerging, especially for data-intensive processing. Two sources are useful for analyses of the relations between stream processing and the Complex Event Processing (Grelck et al., "Asynchronous Stream Processing with S-Net" (2010); and (Margara et al., “Processing Flows of Information,” 2011).

There is a trend in using http protocol to transport protocol primitives supporting different Interaction Models. As a broad classification, Web service protocols are categorized into two groups, Simple Object Access Protocol (SOAP) (Box et al., “Simple Object Access Protocol,” 2000) and REpresentational State Transfer (REST) (Mueller, “Understanding SOAP and REST Basics,” 2013). An IoT scenario favors the REST architecture due to its simplicity and convenience in a constrained environment. In the following section, we will briefly describe the two Web service architectures before going to the protocols that are directly related to interactions in the IoT environment.

SOAP

SOAP relies exclusively on XML to provide messaging services. Microsoft originally developed SOAP to take the place of older technologies that don't work well on the Internet such as the Distributed Component Object Model (DCOM) and Common Object Request Broker Architecture (CORBA). These technologies fail because they rely on binary messaging; the XML messaging that SOAP employs works better over the Internet.

SOAP is designed to support expansion, so it has all sorts of other acronyms and abbreviations associated with it, such as WS-Addressing, WS-Policy, WS-Security, WS-Federation, WS-ReliableMessaging, WS-Coordination, WS-AtomicTransaction and WS-RemotePortlets. The point is that SOAP is highly extensible, but one only uses the pieces one needs for a particular task. For example, when using a public Web service that's freely available to everyone, one really doesn't have much need for WS-Security.

The XML used to make requests and receive responses in SOAP can become extremely complex. In some programming languages, the user needs to build those requests manually, which becomes problematic because SOAP is intolerant of errors. However, other languages can use shortcuts that SOAP provides; that can help reduce the effort required to create the request and to parse the response. In fact, when working with .NET languages, one never even sees the XML.

Part of the specification is the Web Services Description Language (WSDL). This is another file that's associated with SOAP. It provides a definition of how the Web service works, so that when you create a reference to it, the integrated development environment (IDE) can completely automate the process. So, the difficulty of using SOAP depends to a large degree on the language used (Curbera, "Web Services Platform Architecture: SOAP, WSDL," 2005; see also Curbera et al., "Unraveling the Web Services Web," 2002).

One of the most important SOAP features is built-in error handling. If there's a problem with a request, the response contains error information that can be used to fix the problem. Given that you might not own the Web service, this particular feature is extremely important; otherwise you would be left guessing as to why things didn't work. The error reporting even provides standardized codes so that it's possible to automate some error handling tasks in your code.

An interesting SOAP feature is that you don't necessarily have to use it with the HyperText Transfer Protocol (HTTP) transport. There's an actual specification for using SOAP over Simple Mail Transfer Protocol (SMTP) and there isn't any reason you can't use it over other transports.

REST

REpresentational State Transfer (REST) is a type of software architecture for distributed hypermedia systems (Mueller, "Understanding SOAP and REST Basics," 2013). The term was introduced in 2000 by Roy Fielding in his doctoral dissertation. REST-style architectures conventionally consist of clients and servers. Clients create requests to servers; servers process requests and return the appropriate responses. Requests and responses are built around the transfer of resource representations. A resource can be any coherent and meaningful concept that may be addressed. A representation of a resource is a document that captures the current or intended state of that resource.

Many developers found SOAP cumbersome and hard to use. For example, working with SOAP in JavaScript means writing a lot of code to perform extremely simple tasks because one must create the required XML structure absolutely every time.

REST provides a lighter-weight alternative. Instead of using XML to make a request, REST relies on a simple URL in many cases. In some situations one must provide additional information in special ways, but most Web services using REST rely exclusively on obtaining the needed information using the URL approach. REST can use four different HTTP 1.1 verbs (GET, POST, PUT and DELETE) to perform tasks.

Unlike SOAP, REST doesn't have to use XML to provide the response. One can find REST-based Web services that output the data in Command Separated Value (CSV), JavaScript Object Notation (JSON) and Really Simple Syndication (RSS). The point is that you can obtain the output you need in a form that's easy to parse within the language you need for your application.

Comparison between SOAP and REST

SOAP is definitely the heavyweight choice for Web service access. It provides the following advantages when compared to REST:

- ✓ Language, platform and transport independent (REST requires use of HTTP)
- ✓ Works well in distributed enterprise environments (REST assumes direct point-to-point communication)
- ✓ Standardized
- ✓ Provides significant pre-build extensibility in the form of the WS* standards
- ✓ Built-in error handling
- ✓ Automation when used with certain language products

REST is easier to use for the most part and is more flexible. It has the following advantages when compared to SOAP:

- ✓ No expensive tools require to interact with the Web service
- ✓ Smaller learning curve
- ✓ Efficient (SOAP uses XML for all messages, REST can use smaller message formats)
- ✓ Fast (no extensive processing required)
- ✓ Closer to other Web technologies in design philosophy

4.2 Protocol Usage in the Context of IoT

The interaction between IoT endpoints may follow the M2M communication concept. The term M2M refers to systems that enable machines to communicate with back-end information systems and/or directly with other machines, in order to provide real-time data. M2M communication can be event-based, that is, triggered by the occurrence of a particular event, and/or polling based, which takes place in predefined time intervals. M2M applications that realize M2M communication include four basic stages.

- ✓ Data collection
- ✓ Transmission of specific data over a communication network
- ✓ Assessment of the data

- ✓ Response to the available information

Based on the previously mentioned stages, M2M applications can be used to orchestrate and deliver services in remote endpoints without the need for human intervention. The logic used to process the collected data can implement complex decision making, thus enabling the provision of services.

For devices used in the IoT world, it is difficult to implement the old HTTP protocol due to various reasons. The main reason is the fact that IoT is made up of unseen devices that require very little interaction. IoT devices consume very little power and frequently have poor network connectivity. Accordingly, HTTP is too heavy to be a good fit for these devices. An HTTP request requires a minimum of nine TCP packets, even more when one considers packet loss from poor connectivity, and plain text headers can get very verbose. And even with all this overhead HTTP doesn't guarantee message delivery (Kellogg, "Why HTTP Won't Work for IoT," 2014).

The HTTP overhead also adds to IoT operating expenses. Current costs of wireless connectivity are exorbitant. Bandwidth conservation is especially important with enterprise customers that often have hundreds of thousands or millions of devices deployed.

IoT network traffic falls into two categories: telemetry and telecommand. Telemetry is the act of gathering telemetrics, or sending data over long distances. Usually telemetry involves sending data from many dumb sensors to a smart hub of some sort. Telecommand is the act of sending commands across a network.

Most telemetry protocols are modeled as publish/subscribe architecture. Sensors connect to a broker and periodically publish their readings to a topic. A central cluster of servers (the cloud) will then subscribe to the topic and process sensor readings in real-time, as illustrated in Figure 24. A typical enterprise arrangement will have thousands or millions of sensors sending telemetrics to a handful of servers that split up the task of processing the data.

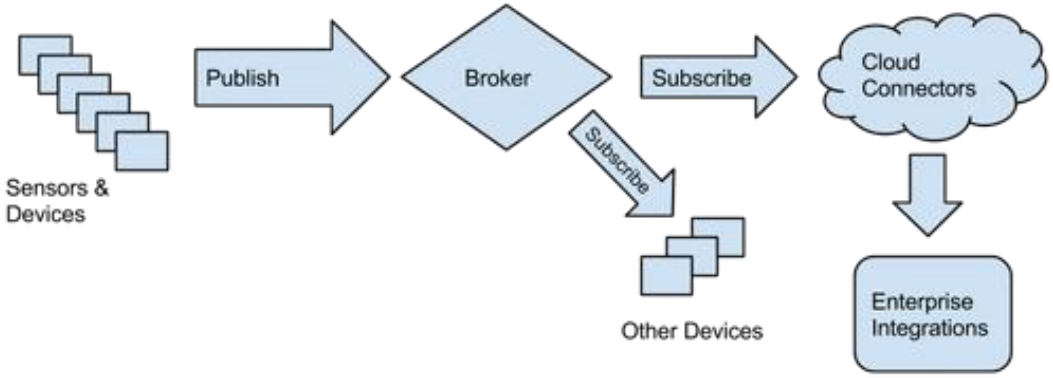


Figure 24. Architecture of telemetry delivery system

One of the well-known telemetry protocols is MQTT. In the following section we present a closer look at the existing telemetry and telecommand protocols.

4.3 MQ Telemetry Transport (MQTT)

From Wikipedia: "*MQTT (formerly Message Queue Telemetry Transport) is a publish-subscribe based 'light weight' 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 and/or network bandwidth is limited. The Publish-Subscribe messaging pattern requires a message broker. The broker is responsible for distributing messages to interested clients based on the topic of a message."

MQTT is an extremely simple and lightweight messaging protocol. Its publish/subscribe architecture is designed to be open and easy to implement, with up to thousands of remote clients capable of being supported by a single server. These characteristics make MQTT ideal for use in constrained environments where network bandwidth is low or where there is high latency and with remote devices that might have limited processing capabilities and memory (Hunkeler et al., "MQTT-S – A Publish/Subscribe Protocol," 2008).

The MQTT protocol includes the following benefits:

- ✓ Extends connectivity beyond enterprise boundaries to smart devices
- ✓ Offers connectivity options optimized for sensors and remote devices
- ✓ Delivers relevant data to any intelligent, decision-making asset that can use it
- ✓ Enables massive scalability of deployment and management of solutions

MQTT minimizes network bandwidth and device resource requirements while attempting to ensure reliability and delivery. This approach makes the MQTT protocol particularly well-suited for connecting M2M, which is a critical aspect of the emerging concept of an IoT.

The MQTT protocol includes the following attributes:

- ➢ Open and royalty-free for easy adoption
  MQTT is open to make it easy to adopt and adapt for the wide variety of devices, platforms and operating systems that are used at the edge of a network.
- ➢ A publish/subscribe messaging model that facilitates one-to-many distribution
  Sending applications or devices do not need to know anything about the receiver, not even its address.
- ➢ Ideal for constrained networks (low bandwidth, high latency, data limits and fragile connections)
  MQTT message headers are kept as small as possible. The fixed header is just two bytes, and its on demand, push-style message distribution keeps network utilization low.
- ➢ Multiple service levels allow flexibility in handling different types of messages
  Developers can designate that messages will be delivered at most once, at least once or exactly once.
- ➢ Designed specifically for remote devices with little memory or processing power
  Minimal headers, a small client footprint and limited reliance on libraries make MQTT ideal for constrained devices.
- ➢ Easy to use and implement with a simple set of command messages
  Many applications of MQTT can be accomplished using just CONNECT, PUBLISH, SUBSCRIBE and DISCONNECT.
- ➢ Built-in support for loss of contact between client and server

The server is informed when a client connection breaks abnormally, allowing the message to be re-sent or preserved for later delivery.

MQTT was designed and promoted as a public standard by IBM, so IBM WebSphere MQ Telemetry uses the MQTT protocol to extend the IBM universal messaging backbone to connect a wide range of remote sensors, actuators and telemetry devices.

Basic concepts of MQTT

The MQTT protocol is built upon several basic concepts, all aimed at assuring message delivery while keeping the messages themselves as lightweight as possible.

Publish/subscribe

The MQTT protocol is based on the principle of publishing messages and subscribing to topics, which is typically referred to as a publish/subscribe model. Clients can subscribe to topics that pertain to them and thereby receive whatever messages are published to those topics. Alternatively, clients can publish messages to topics, thus making them available to all subscribers to those topics.

Topics and subscriptions

Messages in MQTT are published to topics, which can be thought of as subject areas. Clients, in turn, sign up to receive particular messages by subscribing to a topic. Subscriptions can be explicit, which limits the messages that are received to the specific topic at hand or can use wildcard designators, such as a number sign (#) to receive messages for a variety of related topics.

Quality of service levels

MQTT defines three qualities of service (QoS) levels for message delivery, with each level designating a higher level of effort by the server to ensure that the message gets delivered. Higher QoS levels ensure more reliable message delivery but might consume more network bandwidth or subject the message to delays due to issues such as latency.

Retained messages

With MQTT, the server keeps a message even after sending it to all current subscribers. If a new subscription is submitted for the same topic, any retained messages are then sent to the new subscribing client.

Clean sessions and durable connections

When an MQTT client connects to the server, it sets the clean session flag. If the flag is set to true, all of the client's subscriptions are removed when it disconnects from the server. If the flag is set to false, the connection is treated as durable, and the client's subscriptions remain in effect after any disconnection. In this event, subsequent messages that arrive carrying a high QoS designation are stored for delivery after the connection is re-established. Using the clean session flag is optional.

Wills

When a client connects to a server, it can inform the server that it has a will, or a message, that should be published to a specific topic or topics in the event of an unexpected disconnection. A will is particularly useful in alarm or security settings where system managers must know immediately when a remote sensor has lost contact with the network.

Comparison between MQTT and HTTP

Although comparison is often made between MQTT and other common protocols, the most useful comparison is with HTTP, for the following reasons:

- HTTP is the most widely used and available protocol. Almost all computing devices with a TCP/IP stack have it. In addition, because HTTP and MQTT are both based on TCP/IP, developers need to choose between them.
- The HTTP protocol uses a request/response model, which is currently the most common message exchange protocol. MQTT uses a publish/subscribe pattern. Developers need to understand the relative advantages of each type of model, depicted in Table 4.

Table 4. Comparison between MQTT and HTTP

	MQTT	HTTP
Design orientation	Data centric	Document centric
Pattern	Publish/subscribe	Request/response
Complexity	Simple	More complex
Message size	Small, with a compact binary header just two bytes in size	Larger, partly because status detail is text-based
Service levels	Three quality of service settings	All messages get the same level of service
Extra libraries	Libraries for C (30 KB) and Java (100 KB)	Depends on the application (JSON, XML), but typically not small
Data distribution	Supports 1 to zero, 1 to 1, and 1 to n	1 to 1 only

4.4 Constrained Application Protocol (CoAP)

Frequently IoT and home automation projects come to the point where something must move, make a sound, flash an indicator light or the like. Maybe an occupant reads a thermocouple and has decided that the room is too hot and now wants it to cool it off; a message must be sent to the heating device to turn off. Maybe a Bluetooth Low Energy (BLE) device detects that the phone in the occupant's pocket is nearby, so it sends a message to open the garage door. These are examples of telecommand.

CoAP is one of those Telecommand protocols (Kellogg, “CoAp Telecommands Breathe Life into Sensor Networks,” 2014). Telecommand is the sibling of telemetry. Where telemetry is sending data from a sensor to a remote hub, telecommand is sending commands to a remote device. Together, the two categories account for all of the IoT protocols in one form or another.

Conceptually CoAP is compatible with HTTP except that it's designed for devices with constrained resources like sensors and microcontrollers that have restricted memory and bandwidth capabilities (see Table 6). It's designed to support RESTful architecture, so it supports the four main methods (GET, POST, PUT and DELETE) and can identify resources by a URL.

It should be noted that unlike HTTP, CoAP is a compact binary format. While conceptually it might seem like a subset of HTTP, in reality the wire protocol is very different. CoAP is designed for constrained platforms where every bit and CPU cycle matters. The entire header fits into less than four bytes, 30 bits to be exact, which includes a 16-bit message ID.

Even though CoAP runs over user datagram protocol (UDP), it has optional delivery guarantees much like MQTT. Additionally, CoAP can run over any datagram protocol, including SMS and SCTP on cellular networks. It uses a mechanism of detaching itself from TCP and instead implementing delivery guarantees on top of an unreliable protocol.

The major difference between CoAP and typical HTTP Web applications is that, instead of having many weak clients and a few powerful servers, CoAP usually has many weak servers and a few powerful clients. This inverted topology is an artifact of RESTful architecture. In the HTTP web, the servers controlled the resources. But in the CoAP Web, the sensors and actuators control the resources. For a thermocouple the temperature reading is the resource. For an actuator the resource is the ability to move something, like a door or a faucet.

Table 5. COAP methods and their description

Action	Description
GET /temp	Get a temperature reading in degrees Celsius
POST /door	Open or close a door
PUT /config	Set configuration values for a sensor or actuator
DELETE /sensor/42	Tell an actuator to remove a sensor from the list that it reads from and reacts to

4.5 SensorML

SensorML provides standard models and an XML encoding for describing any process, including the process of measurement by sensors and instructions for deriving higher-level information from observations (Botts, “SensorML and Processing,” 2009). Processes described in SensorML are discoverable and executable. All processes define their inputs, outputs, parameters and method, as well as provide relevant metadata. SensorML models detectors and sensors as processes that convert real phenomena to data.

In brief, SensorML:

- is the means by which sensors and processes make themselves and their capabilities known; it describes inputs, outputs and taskable parameters.
- provides the history of measurement and processing of observations; and it supports quality knowledge of observations.
- supports on-demand derivation of higher-level information (e.g., geolocation or products) without *a priori* knowledge of the sensor system.

SensorML supports IoT by providing the ability to describe a sensor (or other online processing component) and to provide a link to the real-time values coming from this component.

An example of a small section of a sensorML code is given below, where the whole code describes a sensor with a simple data stream consisting of temperature. We segmented out the part that describes the data so that it can be accessed using a given URL. Accessing this URL would return either the latest value(s) or open up an html stream of real-time values.

```
<swe:DataStream>
    <!-- describe output -->
    <swe:elementTypename="temperature">
         <swe:Quantity
            definition="http://mmisw.org/ont/cf/parameter/air_temperatu
re">
          <swe:uomcode="Cel"/>
       </swe:Quantity>
        </swe:elementType>
```

4.6 Recap

Different implementations of IoT (sub)systems could benefit from different interaction paradigms and related protocols. In this regard, it is worth noting the increased importance of the concept of Broker, i.e., an Entity capable of dispatching messages and information between producers and consumers (see Collina et al., "Introducing the QEST Broker: Scaling the IoT by Bridging MQTT and REST," 2012; and Blackstock et al., "MAGIC Broker 2: An open and extensible platform for the Internet of Things," 2010).
The ability to deal with continuous streaming of data is also gaining relevance for IoT applications.

5. A Definition of Internet of Things

As seen in previous chapters, looking for a comprehensive definition of Internet of Things is not easy. The definition often depends on the particular vision of the proponent entity with respect to the assets of IoT that are deemed more relevant. In other words, many definitions are biased towards the assets that a specific proponent wants to emphasize. In this chapter the authors attempt to propose a more neutral definition that encompasses the many facets of the Internet of Things. First a discussion about IoT and Cyber-Physical Systems is presented, then a few details on Wireless Sensor Networks (WSN) are provided, than these and other inputs are used for characterizing a new definition of IoT.

5.1 Internet of Things and Cyber-Physical Systems

In most academic and project activities, the difference between “Internet of Things” and “Cyber-Physical Systems (CPS)” is not made clear and it is difficult to find a source that draws a clear-cut distinction between the two terms. Most persons consider the two definitions as different explanations for the same idea and use the words interchangeably. The general trend appears to be that CPS is a U.S. characterization for IoT. However, actual differences exist and in this subsection we will address the commonalities and differences between these two concepts.

A cyber-physical system is a system of collaborating computational elements controlling physical entities. It is when the mechanical and electrical systems (e.g., sensors and communication tools) embedded in products and materials are networked using software components. They use shared knowledge and information from processes to independently control logistics and production systems. Accordingly, CPS tends to go beyond a mere unique identification and control of individual things to the level of networking between identified objects and sharing information about a specific condition so as to accomplish a certain goal with better efficiency. In contrast to traditional embedded systems, the CPS is a network of interacting appliances with physical inputs and outputs instead of standalone devices.

Common applications of CPS typically fall under sensor-based, communication-enabled autonomous systems. For example, many wireless sensor networks monitor some aspect of the environment and relay the processed information to a central node so that the central node can make decisions with more reliable data collected from numerous distributed sources.

In this context, a so-called “smart grid” can be considered a good example of a CPS. A smart grid is a modernized electrical grid that uses analog or digital information and communications technology to gather and act on information – such as the behaviors of suppliers and consumers – in an automated fashion to improve the efficiency, reliability, economics and sustainability of the production and distribution of electricity.

In contrast, an IoT system starts from the level where a single “thing” is identified using a unique global identifier and can be accessed from anywhere, anytime. The level of information obtained by accessing the “thing” can be as low as a static data that is stored on the RFID tags. Primarily, IoT is concerned with unique identification, connecting with the Internet and accessibility of “things.” Yet, identified objects in an IoT system can still be networked together so as to control a certain scenario in a coordinated way, in which case an IoT system can be considered to grow to the level of a CPS.

Generally we can say that CPS is mainly concerned about the collaborative activity of sensors or actuators to achieve a certain goal and to do this CPS uses an IoT system to achieve the collaborative work of the distributed systems.

In the last section of this chapter we emphasize that for a system to be considered as an IoT, the “things” so identified have to be connected to the Internet which is a network above an intranet or extranet. But a CPS doesn’t have this requirement as long as the collaborating objects are uniquely identified within the application context and collaborate to achieve the required sensing or actuation goal.

From the foregoing discussions, we can conclude that from a networking or communication point of view, a CPS starts from the interconnection and collaboration of objects in an intranet

scenario and can grow to the level of interconnecting objects over the Internet to achieve a collaborative sensing or actuation task. Whereas from an applications point of view, it is IoT that starts from the lowest level of identifying an object to read statically stored data on RFID tags and can grow to the level of networking between the identified objects to do collaborative work where, in this case, it grows to the level of CPS.

Overall, from a networking or communication point of view, IoT targets a broader view of connecting objects in a global aspect whereas from an application point of view CPS targets the coordination of networked objects to achieve a specific goal.

5.2 Internet of Things and Wireless Sensor Networks

In the study of an IoT system, there are still some who confuse an IoT system with wireless sensor networks (WSN). But there is a clear difference between the two.

A WSN is a spatially distributed network of autonomous sensors that monitor physical or environmental conditions, such as temperature, sound, pressure, etc. and cooperatively pass their data through the network to a central location. The WSN is built of "nodes" – from a few to several hundreds or even thousands, where each node is connected to one (or sometimes several) sensors. The scope of WSN is the coordinated collection of data.

On the other hand, an IoT system's scope goes beyond this, in a way, where smartness can be added to the objects so that they can do the work of actuation to achieve a certain goal without human intervention. On top of that unique identification of "Things" and their connection to the Internet is another necessary feature of IoT that doesn't pertain to WSN.

Generally, WSN can be one part of IoT in that sensors used in an IoT system can be networked to achieve a coordinated result.

5.3 Features and Definition of Internet of Things

In this chapter we provide our definition of an IoT system together with a set of features that a system must have in order to be considered as an IoT system.

In previous chapters we have discussed the various features of an IoT system. In devising a definition for a subject of interest, it is beneficial to list the defining features that pertain to the subject of interest so that we may build a clear mental picture of the subject we are going to define. Accordingly, we list here the features of IoT.

Interconnection of Things: The first feature of IoT is derived from the name that describes it. It is a system that deals with the interconnection of "Things." The word "Thing" refers to any physical object that is relevant from a user or application perspective.

Connection of Things to the Internet: From the name IoT, we can also learn that the "Things" are connected to the Internet. Accordingly, from the name we can deduce that the system is not an Intranet or Extranet of Things.

Uniquely Identifiable Things: An IoT system is composed of things that are uniquely identifiable.

Ubiquity: As per ITU's definition (ITU, SERIES Y, 2005), ubiquity is a major feature of an IoT system, indicating a network which is available anywhere and anytime. But in the context of IoT, the concept "anywhere" and "anytime" need not necessarily refer, respectively, to "globally"

and “always.” The “anywhere” mainly refers to the concept of where it is needed and the “anytime” similarly refers to when it is needed.

Sensing/Actuation capability: There is the involvement of sensors/actuators in the IoT system. The sensors/actuators are connected to the “Things” and perform the sensing/actuation which bring the smartness of the “Things.”

Embedded intelligence: Smart and dynamic objects, with emergent behavior, embed intelligence and knowledge functions as tools and become an (external) extension to the human body and mind.

Interoperable Communication Capability: The IoT system has a communication capability based on standard and interoperable communication protocols.

Self-configurability: The other important behavior that an IoT system has is self-configurability. Due to the heterogeneity of devices – including sensors, actuators, storage devices, utility monitoring devices, mobile phones, network elements and computers – and the sheer number of devices that are being connected to the Internet under the IoT umbrella, remote or cloud-based control appears to be a daunting task destined to suffer from limited scalability. Hence, the natural direction for IoT devices is to manage themselves, both in terms of their software/hardware configuration and their resource utilization (energy, communication bandwidth, medium access, etc.). Self-configuration primarily consists of the actions of neighbor and service discovery, network organization and resource provisioning (Chatzigiannakis et al, “True Self-Configuration for IoT,” 2012).

Programmability: The “Things” of an IoT system has a programmability feature. At the simplest level, a programmable device is one that can take on a variety of behaviors at a user’s command without requiring physical changes. For example, a programmable synthesizer can sound like a number of different instruments depending on the player’s preference, while a traditional piano can sound only the way it was physically designed to sound.

The scope of an IoT system varies from a small system which contains uniquely identifiable things and small sensors to a system that interconnects millions of things with a capacity to deliver complex services. Accordingly, it is better to provide separate definitions of IoT for small systems and complex systems.

Small environment scenario

The lowest complexity of an IoT system is a uniquely identifiable “Thing” connected to the Internet, perhaps with static data stored in RFID tags, so that this data can be accessed from anywhere, at anytime, by anything. Within this low complexity, the “Thing” can be connected with a sensor/actuator so that the current state of the thing can be sensed and action can be taken using actuation. It can also have a programmability feature. Accordingly, our low complexity view of IoT includes “Things” that are uniquely identifiable and have sensing/actuation and programmability capabilities.

With this in mind, we devised a definition of IoT for low complexity systems as follows:

“An IoT is a network that connects uniquely identifiable “Things” to the Internet. The “Things” have sensing/actuation and potential programmability capabilities. Through the exploitation of

unique identification and sensing, information about the "Thing" can be collected and the state of the 'Thing' can be changed from anywhere, anytime, by anything."

Large environment scenario:

The IoT system can grow to a level of complexity where a large amount of "Things" can be interconnected to deliver a complex service and support an execution of a complex process. In such large environments, depicted in Figure 25, we devised the following definition of IoT:

"Internet of Things envisions a self-configuring, adaptive, complex network that interconnects 'things' to the Internet through the use of standard communication protocols. The interconnected things have physical or virtual representation in the digital world, sensing/actuation capability, a programmability feature and are uniquely identifiable. The representation contains information including the thing's identity, status, location or any other business, social or privately relevant information. The things offer services, with or without human intervention, through the exploitation of unique identification, data capture and communication, and actuation capability. The service is exploited through the use of intelligent interfaces and is made available anywhere, anytime, and for anything taking security into consideration."

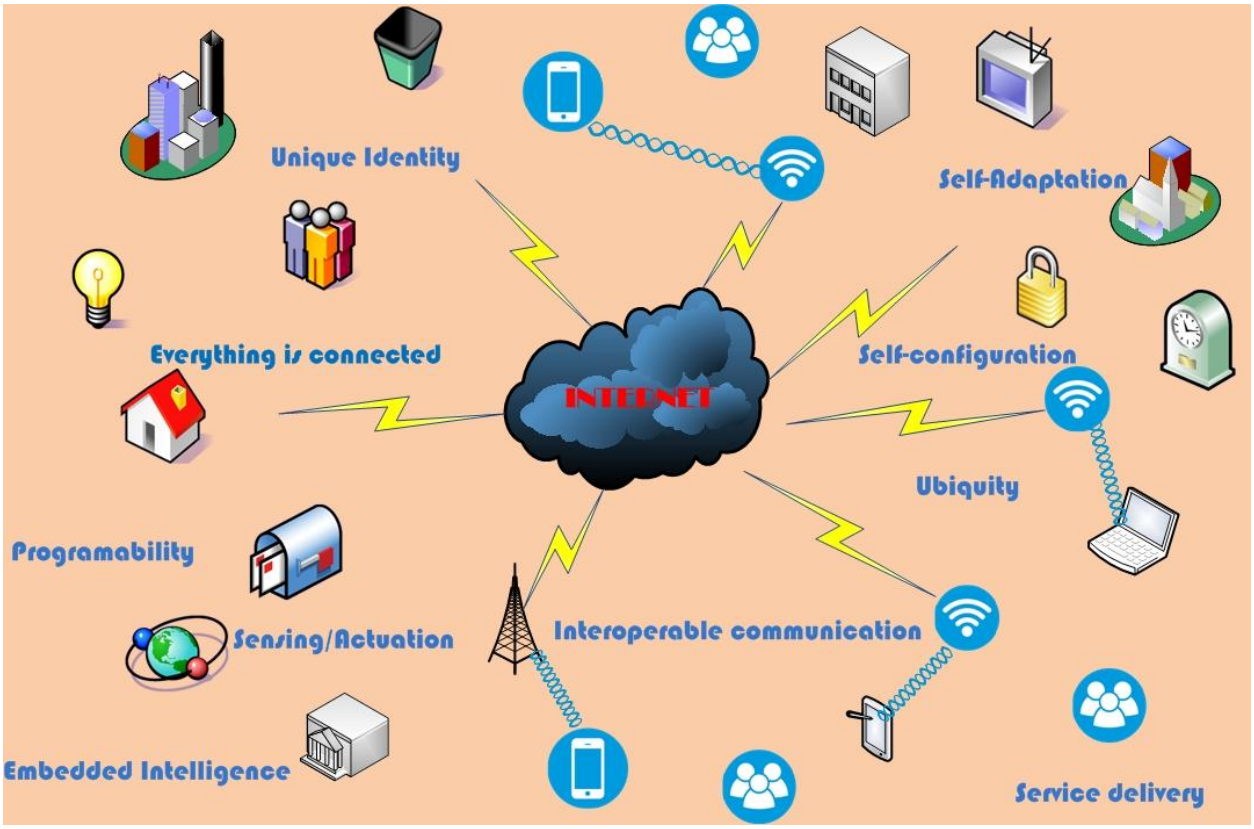


Figure 25. Features and scope of an IoT system

The actual distinguishing element between the *Small environment scenario* and the *Large environment scenario* is *complexity*, not only in terms of number of things, but also from the things ownership/management point of view. Indeed, IoT systems belonging to a single "administrative/management domain" (that can even pertain to a single person like for

devices/appliances in a smart home) do not constitute the real big problem, partly because, however great the system may be, will still be limited and framed by an overall logic.

Large environment scenario is characterized by things belonging to different "administrative/management domains", normally without explicit relationships among them, so that hundreds, or thousands, of millions can be easily reached. In this context, the complexity becomes dominant and elements like scalability, distributed logic, etc. become essential. All traditional approaches for managing trust, naming, discovery, etc. must be completely rethink to face the issues of the *Large environment scenario*.

Glossary

Active digital entity: Any type of active code or software program, usually acting according to business logic.

Actuator: Mechanical device for moving or controlling a mechanism or system. It takes energy, usually transported by air, electric current or liquid, and converts it into a state change, thus affecting one or more physical entities.

Address: Used for locating and accessing –"talking to"– a device, a resource or a service. In some cases, ID and address can be the same, but conceptually they are different.

Application: Software that implements business logic. Applications access resources that are needed to achieve the goal of the business logic through services. Applications can also provide services. Applications, for instance, can be implemented on a device, in an enterprise system or in the cloud. On-device applications are hardware dependent. In some cases, their implementation can be minimal, i.e., only an extension of the OS/firmware of the device.

Augmented entity: The composition of a physical entity and its associated virtual entity.

Business logic: The goal or behavior of a system involving things. Business logic serves a particular business purpose. Business logic can also define the behavior of a single or multiple physical entities or a complete business process.

"Connection capability" and "connectivity": Both refer to the ability to introduce or interface between a source of data and a device that can carry or handle it. The greater the capability or connectivity the more effectively data can be transferred.

Cyber-physical system (CPS): A system of collaborating computational elements controlling physical entities.

Device: A technical, physical component (hardware) with communication capabilities linking it to other IT systems. A device can be either attached to or embedded inside a physical entity or monitor a physical entity in its vicinity.

Electronic Product Code (EPC): An addressing mechanism designed as a universal identifier that provides a unique identity for every physical object anywhere in the world.

EPC Discovery Service (EPCDS): A service that allows users to find all the data related to an EPC number.

EPCglobal Network: A computer network used to share product data between trading partners, created by EPCglobal. Basis for the information flow in the network is the Electronic Product Code (EPC) of each product which is stored on an RFID tag.

EPC Information Services (EPCIS): An EPCglobal standard designed to enable EPC-related data sharing within and across enterprises.

FIFO: An acronym for First In, First Out, a method for organizing and manipulating a data buffer, where the oldest (first) entry, or "head" of the queue, is processed first.

Gateway: A device that provides protocol translation between peripheral trunks of the IoT that is provided with lower parts of the communication stacks. For efficiency purposes, gateways can act at different layers, depending on which is the lowest layer in a common protocol implementation. Gateways can also provide support for security, scalability, service discovery, geo-localization, billing, etc.

Identifier (ID): An artificially generated or natural feature used to disambiguate things from each other. There can be several IDs for the same Physical Entity. This set of IDs is an attribute of a physical entity.

Identity: The properties of an entity that makes it definable and recognizable.

Look-up: In contrast to discovery, look-up is a service that finds existing, known resources by using a key or identifier.

Machine to machine (M2M): A technology that enables networked devices to exchange information and perform actions without the manual assistance of humans.

Middleware: Software that resides between RFID interrogators and enterprise software.

MQTT: A machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium.

Nanotechnology: The manipulation of matter on an atomic, molecular and supramolecular scale.

Network resource: A resource hosted somewhere in the network, e.g., in the cloud.

Object Name Service (ONS): A mechanism that leverages Domain Name System (DNS) to discover information about a product and related services from the Electronic Product Code (EPC). It is a component of the EPCglobal Network.

On-Device resource: Are resources hosted inside a device and enabling access to the device, and thus to the related physical entity.

Passive digital entity: A digital representation of something stored in an IT-based system.

Physical entity: Any physical object that is relevant from a user or application perspective.

Programmability: The capability within hardware and software to change; to accept a new set of instructions that alter its behavior.

Publish–subscribe: A messaging pattern where senders of messages, called publishers, do not program the messages to be sent directly to specific receivers, called subscribers. Instead, published messages are characterized into classes, without knowledge of what, if any, subscribers there may be. Similarly, subscribers express interest in one or more classes, and only receive messages that are of interest, without knowledge of what, if any, publishers exist.

Radio-frequency identification (RFID): The wireless non-contact use of radio-frequency electromagnetic fields to transfer data, for the purposes of automatically identifying and tracking tags attached to objects.

Resource: A computational element that gives access to information about or actuation capabilities on a physical entity.

REST (**Re**presentational **S**tate **T**ransfer):An abstraction of the architecture of the World Wide Web. It relies on a stateless, client-server, cacheable communications protocol and in virtually all cases, the HTTP protocol is used.

Sensor: A device identifying or recording features of a given physical entity.

Service: A software component enabling interaction with resources through a well-defined interface, often via the Internet. It can be orchestrated together with non-IoT services.

SOAP: A protocol specification for exchanging structured information in the implementation of Web services in computer networks. It relies on XML Information Set for its message format, and usually relies on other application layer protocols, most notably Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission.

Storage: A special type of resource that stores information coming from other resources and provides information about physical entities. They may also include services to process the information stored by the resource. As storages are resources, they can be deployed either on a device or in the network.

Tag: A label or other physical object used to identify the physical entity to which it is attached.

Thing: Generally speaking, any physical object. In the term “Internet of Things,” however, it denotes the same concept as a physical entity.

User: A human or some active digital entity that is interested in interacting with a particular physical object.

Virtual entity: Computational or data element representing a physical entity. Virtual entities can be either active or passive digital entities.

Virtualization: In computing, the term refers to the act of creating a virtual (rather than actual) version of something, including but not limited to a virtual computer hardware platform, operating system (OS), storage device, or computer network resources.

Virtual objects: Objects that are represented in media space and may exhibit a proxy relationship with a physical object.

Web of Things (WoT): A computing concept that describes a future where everyday objects are fully integrated with the Web. The prerequisite for WoT is for the "things" to have embedded computer systems that enable communication with the Web. Such smart devices would then be able to communicate with each other using existing Web standards.

WebSphere MQ: IBM's Message Oriented Middleware offering. It allows independent and potentially non-concurrent applications on a distributed system to communicate with each other.

Wireless Sensor Network (WSN): A wireless network consisting of spatially distributed autonomous devices using sensors to monitor physical or environmental conditions.

References

1. Mark Roberti. Jan. 16, 2005, “The History of RFID Technology.” http://www.rfidjournal.com/articles/view?1338
2. Sarma, Sanjay E., Stephen A. Weis, and Daniel W. Engels. "RFID systems and security and privacy implications." In Cryptographic Hardware and Embedded Systems-CHES 2002, pp. 454-469. Springer Berlin Heidelberg, 2003.
3. Brock, David L. "The Electronic Product Code (EPC)." Auto-ID Center White Paper MIT-AUTOID-WH-002 (2001).
4. Thiesse, Frederic, and Florian Michahelles. "An Overview of EPC Technology." *Sensor Review* 26, no. 2 (2006): 101-105.
5. ITU, The Internet of Things Executive Summary, ITU Internet Reports 2005 available at http://www.itu.int/osg/spu/publications/internetofthings/InternetofThings_summary.pdf
6. IEEE, *The Institute*, “Special Report: The Internet of Things.” http://theinstitute.ieee.org/static/special-report-the-internet-of-things
7. Information about IEEE P2413 are available at http://standards.ieee.org/develop/project/2413.html
8. ETSI Technical Specification, “Machine-to-Machine Communications (M2M); M2M Service Requirements.” Technical Specification. ETSI TS 102 689 V1.1.1(2010-08)
9. ETSI document, “oneM2M Requirements Technical Specification.” oneM2M-TS-0002-V-0.6.2. 2013-10-17, http://www.ttc.or.jp/jp/document_list/pdf/j/TS/TS-M2M-0002v0.6.2.pdf
10. Joerg Swetina (NEC).”ETSI M2M / oneM2M and the need for semantics,” http://www.probe-it.eu/wp-content/uploads/2012/06/K1_ETSI-M2M-oneM2M-and-the-need-for-semantics-IoTWeek2012.pdf
11. Information about RFID technology available at RFID Journal, http://www.rfidjournal.com
12. ITU, SERIES Y: GLOBAL INFORMATION, INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS., AND NEXT-GENERATION NETWORKS, Next Generation Networks – Frameworks and functional architecture models: file:///C:/Users/11598004/Downloads/T-REC-Y.2060-201206-I!!PDF-E.pdf
13. IETF, “The Internet of Things - Concept and Problem Statement,” 2010, http://tools.ietf.org/id/draft-lee-iot-problem-statement-00.txt
14. NIST, “Global City Teams Challenge – SmartAmerica Round Two.” http://www.nist.gov/cps/sagc.cfm and http://www.nist.gov/cps/upload/20140723-SmartAmerica-Global-City-Teams-Challenge-Introduction-v1_6p.pdf
15. Chris Greer, June 11, 2014, ”The Internet's Next Big Idea: Connecting People, Information, and Things.” http://www.nist.gov/el/20140611_internets_next_big_idea.cfm
16. OASIS, “Open Protocols for an Open, Interoperable Internet of Things,” 2014, https://www.oasis-open.org/presentations/open-protocols-and-internet-of-things-oasis.ppt

17. “W3C: What is the Web of Things?” [no date] http://www.theinternetofthings.eu/w3c-what-web-things
18. Wikipedia item on “Web of Things”. http://en.wikipedia.org/wiki/Web_of_Things
19. CASAGRAS Project “Final Report, RFID and the Inclusive Model for the Internet of Things,” http://www.grifs-project.eu/data/File/CASAGRAS FinalReport (2).pdf
20. Edward A. Lee, ”Cyber Physical Systems: Design Challenges,” 2008, http://chess.eecs.berkeley.edu/pubs/427.html
21. Alessandro Bassi, Martin Bauer, Martin Fiedler, Thorsten Kramp, Rob van Kranenburg, Sebastian Lange, Stefan Meissner (editors),”Enabling Things to Talk: Designing IoT Solutions with the IoT Architectural Reference Model” (SpringerOpen, 2013)
22. IoT-A, “Internet-of-Things Architecture IoT-A, Project Deliverable D1.2 – Initial Architectural Reference Model for IoT,” 2011.
23. CERP-IoT. "Visions and Challenges for Realising the Internet of Things," European Commission (2010).
24. Vermesan, Ovidiu, Peter Friess, Patrick Guillemin, Sergio Gusmeroli, Harald Sundmaeker, Alessandro Bassi, Ignacio Soler Jubert et al. "Internet of things strategic research roadmap." Internet of Things-Global Technological and Societal Trends (2011): 9-52.
25. European Research Cluster on Internet of Things (IERC),“Internet of Things,” http://www.internet-of-things-research.eu/about_iot.htm
26. European Technology Platform on Smart Systems Integration web site http://www.smart-systems-integration.org/public/internet-of-things
27. INFSO D.4 Networked Enterprise & RFID and EPoSS, “Internet of Things in 2020, A Roadmap for the Future,” http://www.smart-systems-integration.org/public/documents/publications/Internet-of-Things_in_2020_EC-EPoSS_Workshop_Report_2008_v3.pdf
28. Frank Berkers, Wietske Koers, Katia Colucci, Oskar Kadlec, Dan Puiu, Marc Roelands, Stephane Menoret, iCore Deliverable D1.3, “Vision of the future business ecosystem, new roles and models of acceptance, 2013.
29. Postscapes, “Internet of things definitions: how the IoT has been used and defined over the years,” http://postscapes.com/internet-of-things-definition
30. UK Future Internet Strategy Group FUTURE INTERNET REPORT May 2011, https://connect.innovateuk.org/documents/3677566/3729595/Future+Internet+report.pdf
31. Digital Lifestyle Malasia available at http://dlm.skmm.gov.my/
32. DIGITAL LIFESTYLE MALAYSIA, Malini Ramalingam, “Engage and interact productively and responsibly to unlock the value of new media,” 2013, http://www.skmm.gov.my/skmmgovmy/media/General/pdf/Malini-Ramalingam-Digital-Lifestyle-Malaysia.pdf
33. Finnish Strategic Centre for Science, Technology, and Innovation: For Information and Communications (ICT) services, businesses, and technologies. “Internet of Things Strategic Research Agenda (IoT-SRA).” Version 1.0. 1st September 2011
34. IP for Smart Object Alliance, http://www.ipso-alliance.org/about/mission

35. Krikorian, R., and N. Gershenfeld. "Internet 0—inter-device internetworking." BT technology journal 22, no. 4 (2004): 278-284. Available at http://black.fri.uni-lj.si/iplight/files/research/Paper28Pages278-284.pdf
36. Barnaghi, Payam, Wei Wang, Cory Henson, and Kerry Taylor. "Semantics for the Internet of Things: early progress and back to the future." *International Journal on Semantic Web and Information Systems (IJSWIS)* 8, no. 1 (2012): 1-21.
37. Mattern, Friedemann, and Christian Floerkemeier. "From the internet of computers to the internet of things, From active data management to event-based systems and more: papers in honor of Alejandro Buchmann on the occasion of his 60th birthday." (2010) available in http://www.vs.inf.ethz.ch/publ/papers/Internet-of-things.pdf
38. Society for Brain Integrity – web site: http://www.svegritet.se/
39. Society for Brain Integrity, "Future Internet," 2010, http://www.svegritet.se/emergin-technologies/future-internet/
40. Future Internet 2020. (2009) Call for action by a high level visionary panel. May 2009. Retrieved on 28th May 2010, from http://www.future-internet.eu/fileadmin/documents/reports/Future_Internet_2020-Visionary_Panel.pdf
41. Quitney Anderson, J. & Rainie, L. (2008): The Future of the Internet III. Pew Internet & American Life Project.
42. Fleisch, E. (2010): What is the Internet of Things? An Economic Perspective. Auto-ID Labs White Paper WP-BIZAPP-053. Retrieved on 28th May 2010, from http://autoidlabs.org/uploads/media/AUTOIDLABS-WP-BIZAPP-53.pdf
43. Bauer, M.; Gluhak, A.; Johansson, M.; Montagut, F.; Presser, M.; Stirbu, V. & Vercher, J. (2009): Towards an Architecture for a Real World Internet. In Tselentis, G. et al. (Eds.) Towards the Future Internet – A European Research Perspective. IOS Press, Amsterdam.
44. Abramowicz, H.; Baucke , S.; Johnsson , M.; Kind , M.; Niebert, N.; Ohlman, B.; Quittek, J.; Woesner, H. & Wuenstel, K. (2009): A Future Internet Embracing the Wireless World. In Tselentis, G. et al. (Eds.) Towards the Future Internet – A European Research Perspective. IOS Press, Amsterdam.
45. Horrocks, I. (2007): Semantic Web: The Story So Far. W4A2007 Keynote, May 07–08, 2007, Banff, Canada.
46. The Hammersmith Group, "The Internet of things: Networked objects and smart devices," 2010, http://driverspack.org/download/the-internet-of-things-networked-objects-and-smart-devices/
47. Michael Chui, Markus Löffler, and Roger Roberts, "The Internet of Things," *McKinsey Quarterly*. http://www.mckinsey.com/insights/high_tech_telecoms_internet/the_internet_of_things
48. Rellermeyer, Jan S., Michael Duller, Ken Gilmer, Damianos Maragkos, Dimitrios Papageorgiou, and Gustavo Alonso. "The software fabric for the internet of things." In The Internet of Things, pp. 87-104. Springer Berlin Heidelberg, 2008. Available at http://www.duller.net/michael/fileadmin/pubs/Rellermeyer2008.pdf
49. Accenture and Bankinter, "The Internet of Things: In a Connected World of Smart Objects," 2011, http://www.fundacionbankinter.org/system/documents/8189/original/XV_FTF_Internet_of_things.pdf

50. Nomura Research Institute, Taiichi Inoue, Akihiro Hayakawa, Takuya Kamei, ”China’s Initiative for the Internet of Things and Opportunities for Japanese Business,” 2011, https://www.nri.com/global/opinion/papers/2011/pdf/np2011165.pdf
51. Uckelmann, Dieter, Mark Harrison, and Florian Michahelles. "Architecting the Internet of Things." Springer (2011) preview available at http://link.springer.com/book/10.1007/978-3-642-19157-2
52. Sánchez López T (2010) What the Internet of Things is NOT. Available at http://technicaltoplus.blogspot.com/2010/03/what-internet-of-things-is-not.html.
53. Giusto, Daniel, Antonio Lera, Giacomo Morabito, and Luigi Atzori. The Internet of Things. Berlin/Heidelberg, Germany: Springer, 2010.
54. Weber, Rolf H., and Romana Weber. *Internet of Things*. Springer, 2010.
55. Shelby, Zach, and Carsten Bormann. *6LoWPAN: The wireless embedded Internet*. Vol. 43. John Wiley & Sons, 2011.
56. Vermesan, Ovidiu, and Peter Friess. *Internet of Things: Global Technological and Societal Trends From Smart Environments and Spaces to Green ICT*, River Publishers, 2011.
57. Stephen Haller: “Internet of Things: An Integral Part of the Future Internet,” SAP presentation, http://services.future-internet.eu/images/1/16/A4_Things_Haller.pdf
58. CISCO, “Internet of Everything,” http://www.cisco.com/web/about/ac79/innov/IoE.html
59. Miessler, Daniel, ”HP Security and the Internet of Things,” 2014, http://h30499.www3.hp.com/t5/Fortify-Application-Security/HP-Security-and-The-Internet-of-Things/ba-p/6450208#.U9_M6dQsL2s
60. Martin Bauer (NEC), Nicola Bui (CFR), Pierpaolo Giacomin (HEU), Nils Gruschka (NEC), Stephan Haller (SAP), Edward Ho (HSG), Ralf Kernchen (UniS), Mario Lischka (NEC) ; Jourik De Loof (ALU BE), Carsten Magerkurth (SAP), Stefan Meissner (UniS), Sonja Meyer (SAP), Andreas Nettsträter (FHG IML), Francisco Oteiza Lacalle (TID), Alexander Salinas Segura (UniW), Alexandru Serbanati (CATTID), Martin Strohbach (NEC), Vincent Toubiana (ALBLF), Joachim W. Walewski (Siemens), ”Internet-of-Things Architecture IoT-A, Project Deliverable D1.2 – Initial Architectural Reference Model for IoT”
61. Baccelli, Emmanuel, Oliver Hahm, Matthias Wählisch, Mesut Günes, and Thomas Schmidt. "RIOT: One OS to rule them all in the IoT." (2012).
62. Saraswat, Lalit, and Pankaj Singh Yadav. "A comparative analysis of wireless sensor network operating systems." International Journal of Engineering and Technoscience 1, no. 1 (2010): 41-47.
63. Levis, Philip, Sam Madden, Joseph Polastre, Robert Szewczyk, Kamin Whitehouse, Alec Woo, David Gay et al. "TinyOS: An operating system for sensor networks." In Ambient intelligence, pp. 115-148. Springer Berlin Heidelberg, 2005.
64. Dunkels, Adam, Bjorn Gronvall, and Thiemo Voigt. "Contiki-a lightweight and flexible operating system for tiny networked sensors." In Local Computer Networks, 2004. 29th Annual IEEE International Conference on, pp. 455-462. IEEE, 2004.
65. “The Contiki Operating System” http://userpages.umbc.edu/~jnosek1/cmsc491e/contiki.doc
66. Mary Catherine O’Connor, ”A Guide to RFID Middleware,” *RFID Journal*, April 5, 2010, https://www.rfidjournal.com/purchase-access?type=Article&id=7490&r=%2Farticles%2Fview%3F7490%2F5

67. Emmanuel Baccelli, Oliver Hahm, Mesut G¨unes, Matthias W¨ahlisch, Thomas Schmidt, “RIOT OS: Towards an OS for the Internet of Things,” 32nd IEEE International Conference on Computer Communications (INFOCOM 2013), Apr 2013, Turin, Italy. <hal-00945122> https://hal.inria.fr/file/index/docid/945122/filename/2013-riot_os.pdf
68. Bandyopadhyay, Soma, Munmun Sengupta, Souvik Maiti, and Subhajit Dutta, "Role of Middleware for Internet of Things: A Study." *International Journal of Computer Science and Engineering*, Survey 2, no. 3 (2011): 94-105, http://airccse.org/journal/ijcses/papers/0811cses07.pdf
69. Kong, Ning, Xiao-Dong Li, Wan-Ming Luo, Bao-Ping Yan and Ruan Jian Xue Bao,, "Model of the Resource Addressing in the Internet of Things, " *Journal of Software*, vol. 21, no. 7 (2010): 1657-1666.
70. Mahalle, Parikshit, Sachin Babar, Neeli R. Prasad, and Ramjee Prasad. "Identity Management Framework Towards Internet of Things (IoT): Roadmap and Key Challenges." *Recent Trends in Network Security and Applications*, pp. 430-439. Springer, Berlin/Heidelberg, 2010.
71. Matias Piispanen, ”EPC and IPv6 -based discovery services,” Aalto University, 2011, https://wiki.aalto.fi/display/esgsem/2011s-iot-3
72. Lorenz, M., and Müller, J., and Schapranow, M., and Zeier, A. and Plattner, H., “Discovery Services in the EPC Network,” Hasso-Plattner-Institute, 2011. http://www.intechopen.com/source/pdfs/18103/InTech-Discovery_services_in_the_epc_network.pdf
73. Brock, David L. "The electronic product code (epc)." Auto-ID Center White Paper MIT-AUTOID-WH-002 (2001). Available at http://cocoa.ethz.ch/media/documents/2014/06/archive/MIT-AUTOID-WH-002.pdf
74. Thiesse, Frédéric, Christian Floerkemeier, Mark Harrison, Florian Michahelles, and Christof Roduner. "Technology, standards, and real-world deployments of the EPC network." Institute of Electrical and Electronics Engineers, *Internet Computing*, March-April 2009, http://dspace.mit.edu/openaccess-disseminate/1721.1/62248
75. Dinesh Vadhia, Rohit Gupta, PhD., “IPv6 vs. EPC,” Silicon Valley World Internet Center, Feb. 12, 2004. http://www.worldinternetcenter.com/Pubs/Pubs2004/feb05/IPv6vEPC.pdf
76. Littman, Michael, Samuel Kortchmar, “Internet of Things: The Path to a Programmable World," http://footnote1.com/the-path-to-a-programmable-world/
77. Wasik, B., "Welcome to the programmable world,” *Wired*," (2013). http://www.wired.com/2013/05/internet-of-things-2/
78. Atzori, Luigi, Davide Carboni, and Antonio Iera, "Smart things in the social loop: paradigms, technologies, and potentials," *Ad Hoc Networks* (2014)121-132. http://www.researchgate.net/publication/236108286_Smart_things_in_the_social_loop_Paradigms_technologies_and_potentials/file/3deec51a9c230d7e4e.pdf
79. Chatzigiannakis, Ioannis, Henning Hasemann, Marcel Karnstedt, Oliver Kleine, A. Kroller, Myriam Leggieri, Dennis Pfisterer, K. Romer, and Cuong Truong. "True self-configuration for the IoT," 2012 3rd International Conference on the Internet of Things (IOT), pp. 9-15. IEEE, 2012. https://www.deri.ie/sites/default/files/publications/iot12_0.pdf
80. Micha Rave, ”Virtualization’s Impact on Mobile Devices and the IoT,” *Embedded Computing Design*, Feb. 20, 2014, http://embedded-computing.com/articles/virtualizations-impact-mobile-devices-the-iot/

81. SiliconAngle, “The Internet of Things needs a network of clouds,” http://siliconangle.com/blog/2014/07/03/the-internet-of-things-needs-a-network-of-clouds/
82. “Web of Things”.http://en.wikipedia.org/wiki/Web_of_Things
83. McKeown, Nick. "Software-defined networking." INFOCOM keynote talk (2009). Available at http://www.cs.rutgers.edu/~badri/552dir/papers/intro/nick09.pdf
84. Lantz, Bob, Brandon Heller, and Nick McKeown. "A network in a laptop: rapid prototyping for software-defined networks." In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, p. 19. ACM, 2010. Available http://klamath.stanford.edu/~nickm/papers/a19-lantz.pdf
85. Deze Zeng, Song Guo, and Zixue Cheng, “The Web of Things: A Survey,” (Invited Paper) Journal of Communications, vol. 6, no. 6 (2011): 424-438. School of Computer Science and Engineering, University of Aizu, Japan, Available at http://www.ojs.academypublisher.com/index.php/jcm/article/viewFile/jcm0606424438/3601
86. 18. Sonja Meyer (SAP), Klaus Sperner (SAP), Carsten Magerkurth (SAP), Stefan Debortoli (SAP), Matthias Thoma (SAP).”Internet of Things Architecture, IoT-A Project Deliverable D2.2 – Concepts for Modelling IoT- Aware Processes”
87. Cachin, Christian, Rachid Guerraoui, and Luâis Rodrigues. “Introduction to reliable and secure distributed programming”. Berlin Heidelberg: Springer verlag, 2011.
88. Foster, Ian. “Designing and building parallel programs”. Boston: Addison-Wesley Reading, 1995.
89. Kubiatowicz, John David. “Integrated Shared-Memory and Message-Passing Communication in the Alewife Multiprocessor,” PhD dissertation, Boston: MIT, 1998.
90. Grelck, Clemens, Sven-Bodo Scholz, and Alex Shafarenko. "Asynchronous stream processing with S-Net." International Journal of Parallel Programming (Springer) 38, no. 1 (2010): 38-67.
91. Margara, Alessandro, and Gianpaolo Cugola. "Processing flows of information: from data stream to complex event processing." Proceedings of the 5th ACM international conference on Distributed event-based system. New York: ACM, 2011. 359-360
92. Box, Don, et al. Simple object access protocol (SOAP) 1.1. Standard, W3C, 2000
93. Fielding, Roy T., and Richard N, Taylor. "Principled design of the modern Web architecture." ACM Transactions on Internet Technology (TOIT) (ACM) 2, no. 2 (2002): 115-150.
94. John Mueller. January 8 2013.”Understanding SOAP and REST basics”. http://blog.smartbear.com/apis/understanding-soap-and-rest-basics/
95. Curbera, Francisco, Frank Leymann, Tony Storey, Donald Ferguson, and Sanjiva Weerawarana. “Web services platform architecture: SOAP, WSDL, WS-policy, WS-addressing, WS-BPEL, WS-reliable messaging and more”. Englewood Cliffs: Prentice Hall PTR, 2005.
96. Curbera, Francisco, Matthew Duftler, Rania Khalaf, William Nagy, Nirmal Mukhi, and Sanjiva Weerawarana. "Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI." IEEE Internet Computing (IEEE) 6, no. 2 (2002): 86-93

97. Dimitris Tsaimos (CSE), Norbert Vicari (Siemens), Werner Liekens (ALU BE), Alexis Olivereau (CEA), Andreas Nettsträter (FHG IML), Michele Rossi (CFR), Pierpaolo Giacomin (HEU).IoT-A."Internet-of-Things Architecture, IOT-A Project Deliverable D3.1 - Initial M2M API Analysis"
98. Lien, Shao-Yu, Kwang-Cheng Chen, and Yonghua Lin. "Toward ubiquitous massive accesses in 3GPP machine-to-machine communications." Communications Magazine, IEEE 49, no. 4 (2011): 66-74. Available at http://nfudee.nfu.edu.tw/ezfiles/43/1043/img/790/05741148.pdf
99. Tim Kellogg, "Why HTTP won't work for IoT," January 15, 2014. . http://www.iotworld.com/author.asp?section_id=3224&doc_id=562380
100. Valerie Lampkin, Weng Tat Leong, Leonardo Olivera, SwetaRawat, NageshSubrahmanyam, Rong Xiang. September 2012."Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry"
101. Hunkeler, Urs, Hong Linh Truong, and Andy Stanford-Clark. "MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks." In Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on, pp. 791-798. IEEE, 2008
102. Tim Kellogg, "CoApTelecommands Breathe Life into Sensor Networks," February 4, 2014. http://www.iotworld.com/author.asp?section_id=3224&doc_id=562510
103. "Introduction to SensorML". http://www.ogcnetwork.net/SensorML_Intro
104. Mike Botts. September 2009. "SensorML and Processing" available at http://www.docstoc.com/docs/160164488/SensorML-and-On-Demand-Processing-Botts-Innovative-Research
105. "Internet of Things – Simple Sensor," http://www.sensorml.com/sensorML-2.0/examples/iotSimple.html
106. Collina, Matteo, Giovanni Emanuele Corazza, and Alessandro Vanelli-Coralli, "Introducing the QEST Broker: Scaling the IoT by Bridging MQTT and REST," in *Personal Indoor and Mobile Radio Communications* (PIMRC), 2012 IEEE 23rd International Symposium on, pp. 36-41.
107. Blackstock, Michael, Nima Kaviani, Rodger Lea, and Adrian Friday, "MAGIC Broker 2: An Open and Extensible Platform for the Internet of Things," in *Internet of Things* (IOT), IEEE, 2010, pp. 1-8.
108. Dominique Guinard, Connected devices" real-time push and the Web of Things". https://evrythng.com/2014/10/connected-devices-real-time-push-web-things/
109. Sven Casteleyn, Gustavo Rossi, Marco Winckler, editors, *Web Engineering: 14th Annual Conference, ICWE, 2014*, Toulouse, France. https://books.google.com/books?id=epaLBAAAQBAJ&dq=%E2%80%9CThe+Web+of+Things+is+primarily+an+evolution+of+the+Internet+of+Things+where+the+primary+concern+has+been+how+to+connect+objects+together+at+the+network+layer&source=gbs_navlinks_s