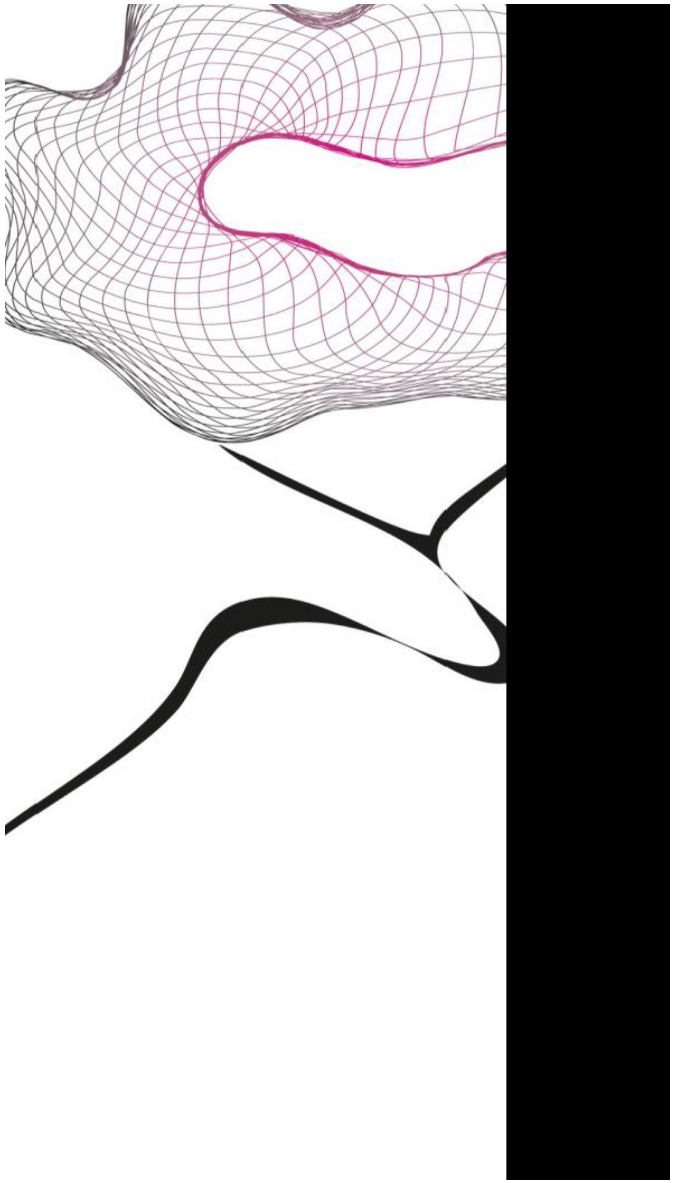
MIDDLEWARE FOR INTERNET OF THINGS



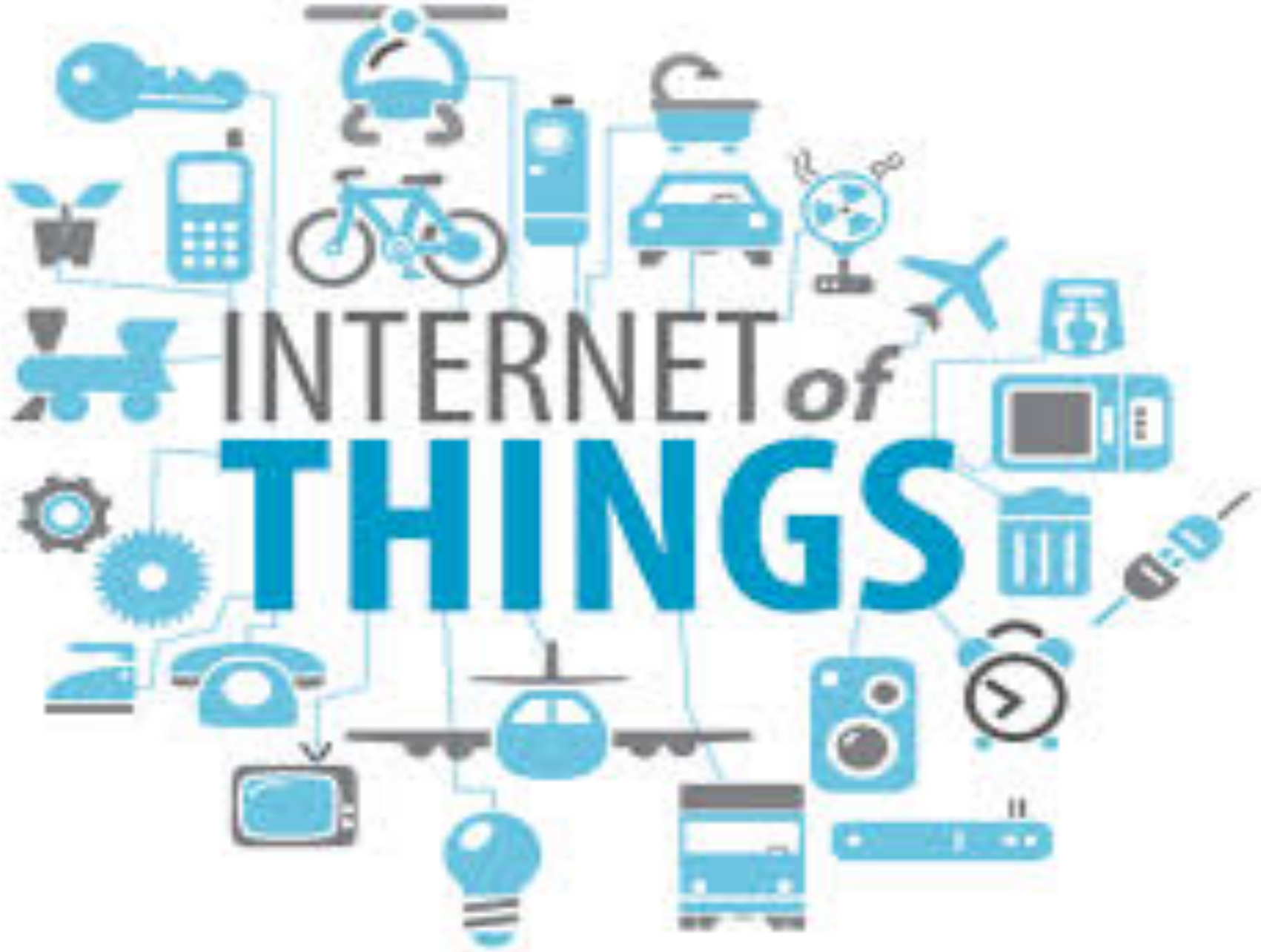
By

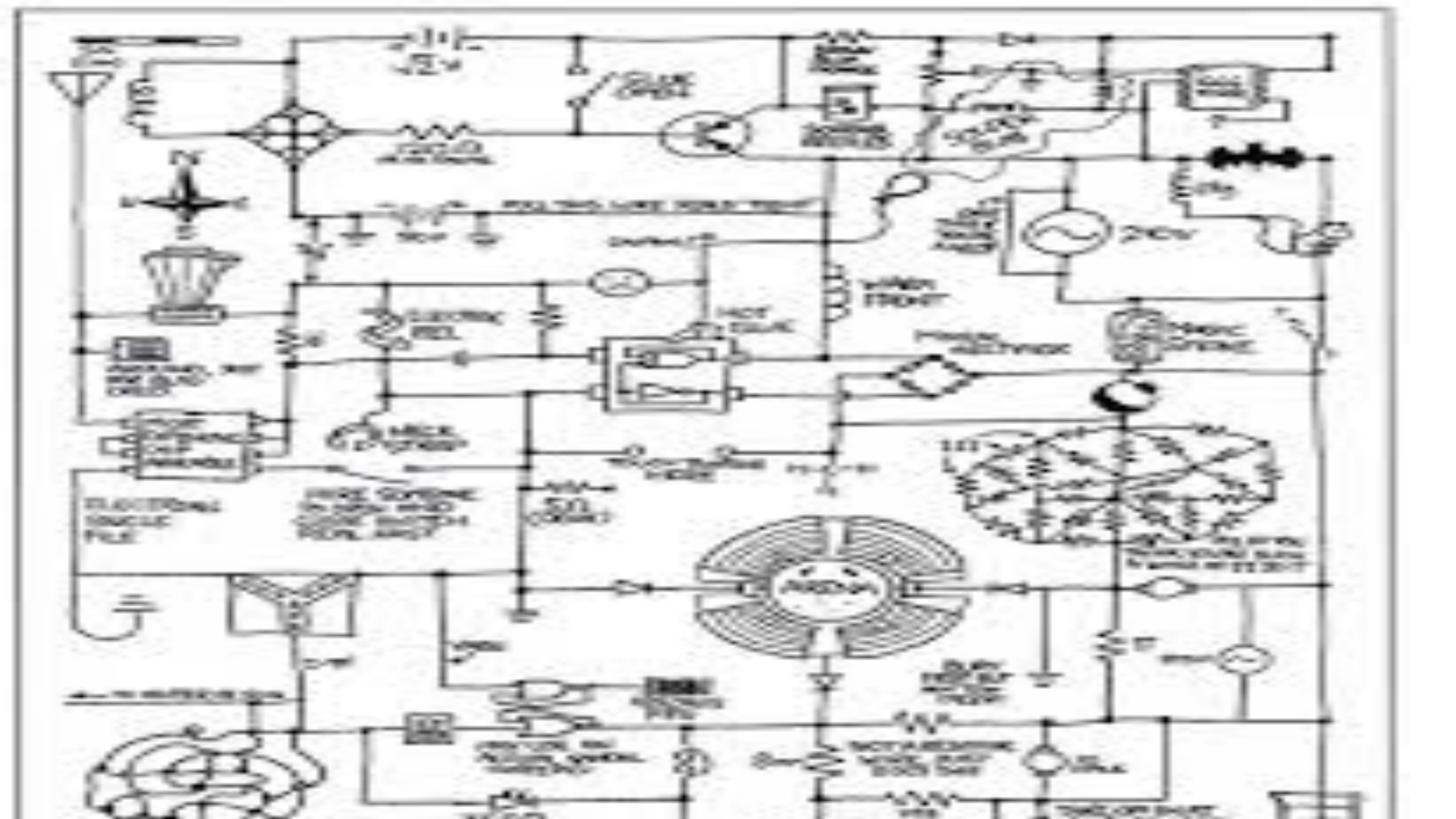
Dr Gaurav Bajpai

Head, Computer and Software Engineering

College of Science and Technology

University of Rwanda


INTERNET of
THINGS




SELF
CLEAN
CUTE
FETCH
SLEEP
MODE
On
Off
Funnunch.com


CAUTION
Internet
of
Things
NO Humans Beyond This Point


0819
0001
1154
0302

Definition of IoT

International Telecommunication Union (ITU) defines IoT as

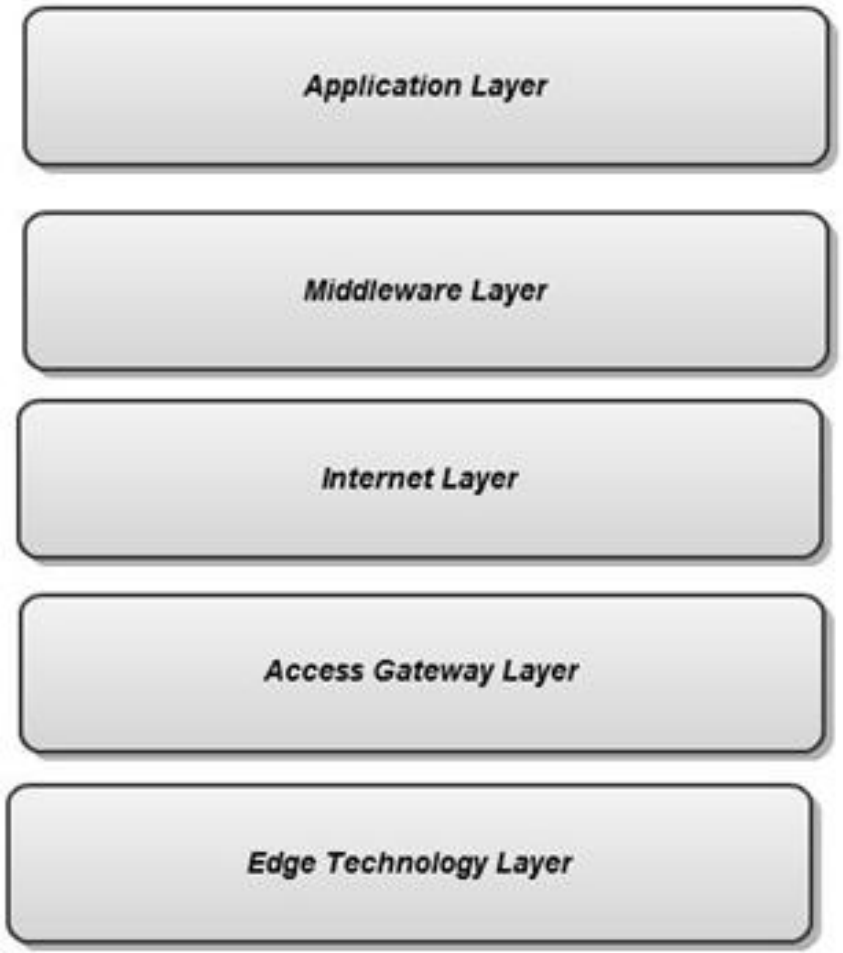
"A global infrastructure for the Information Society, enabling advanced services by interconnecting (physical and virtual) things based on, existing and evolving, interoperable information and communication technologies"


The Internet
of Things

Internet of Things-Architecture (IoT-A)

Defines it as "*The idea of a globally interconnected continuum of devices, objects and things in general emerged with the RFID technology, and this concept has considerably been extended to the current vision that envisages a plethora of heterogeneous objects interacting with the physical environment*."

Layered Architecture of the Internet of Things (IoT).



Edge Technology layer

This is a hardware layer that consists of embedded systems, RFID tags, sensor networks and all of the other sensors in different forms. This hardware layer can perform several functions, such as collecting information from a system or an environment, processing information and supporting communication.

Access Gateway layer

This layer is concerned with data handling, and is responsible for publishing and subscribing the services that are provided by the *Things*, message routing, and hovelling the communication between platforms.

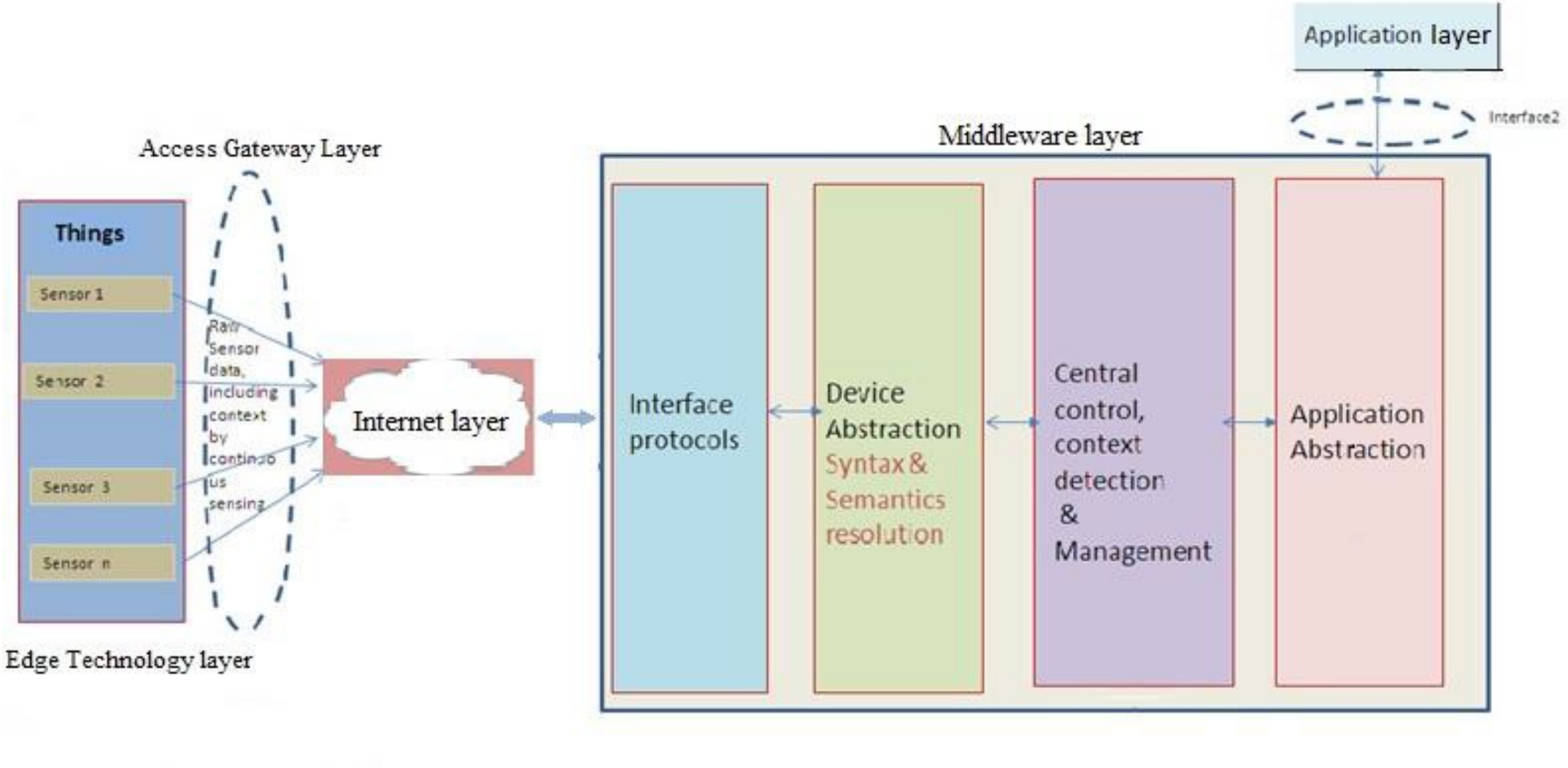
Middleware layer

This layer has some critical functionalities, such as aggregating and filtering the received data from the hardware devices, performing information discovery and providing access control to the devices for applications.

Application layer

This layer is responsible for delivering various application services. These services are provided through the middleware layer to different applications and users in IoT-based systems. The application services can be used in different industries such as, logistics, retail, healthcare, etc.

Functional components of a middleware for IoT-based systems



1. Interface protocols

This component is in charge of providing *technical interoperability*. Interoperability in the context of *Interface protocols* means: *the ability of two systems to interoperate by using the same communication protocols*.
According to ETSI (European Telecommunications Standards Institute) *technical interoperability* is defined as the *association of hardware or software components, systems and platforms that enable machine-to-machine communication to take place. This kind of interoperability is often centered on (communication) protocols and the infrastructure needed for those protocols to operate.*

2. Device Abstraction (DA)

This component is responsible for providing an abstract format to facilitate the interaction of the application components with devices. This abstraction provides *syntactic* and *semantic* interoperability,

- *Syntactic interoperability* is associated with data formats. The messages transferred by communication protocols must have a well-defined syntax and encoding format, which can be represented by using high-level transfer syntaxes such as, HTML and XML.
- *Semantic interoperability* is usually associated with the meaning of the content of message which is understandable for human. Thus, interoperability on this level means that there is a common understanding among people on the meaning of the content (information) being exchanged among them.

3. Central control, Context detection & Management (CCM)

- Context characterizes the situation of an entity, which can be a place, a person or an object that is relevant to the user, applications and their interactions.

- The CCM functional component is responsible to support context-aware computation that is a computational style that take to account the context of the entities that interact with the system. A middleware for IoT-based systems must be context-aware to work in smart environments.

Context-awareness includes two functionalities:

1) Context detection, which consists of collecting data from resources, and selecting the information that can have an impact on the computation.

2) Context processing, to use the gathered information to perform a task or make a decision.

4. Application Abstraction

- This functional component provides an interface for both high-level applications and end users to interact with devices.

- For instance, this interface can be a RESTful interface or can be implemented with some query-based language.

Review of IoT-based Middleware

The term *Internet of Things (IoT)* refers to a wide set of applications and research areas such as, distributed computing and knowledge management. Much of the IoT research is related to the field of *pervasive computing*.

World Wide Web Consortium (W3C) defines *pervasive computing* as a *vision about our future computing life style. In this vision, computer systems will be seamlessly integrated into our everyday life, anticipating our needs and providing relevant services and information for us in an anytime-and-any-where fashion. Pervasive Computing is all about making the human's life easier by exploiting the available computing technologies .*

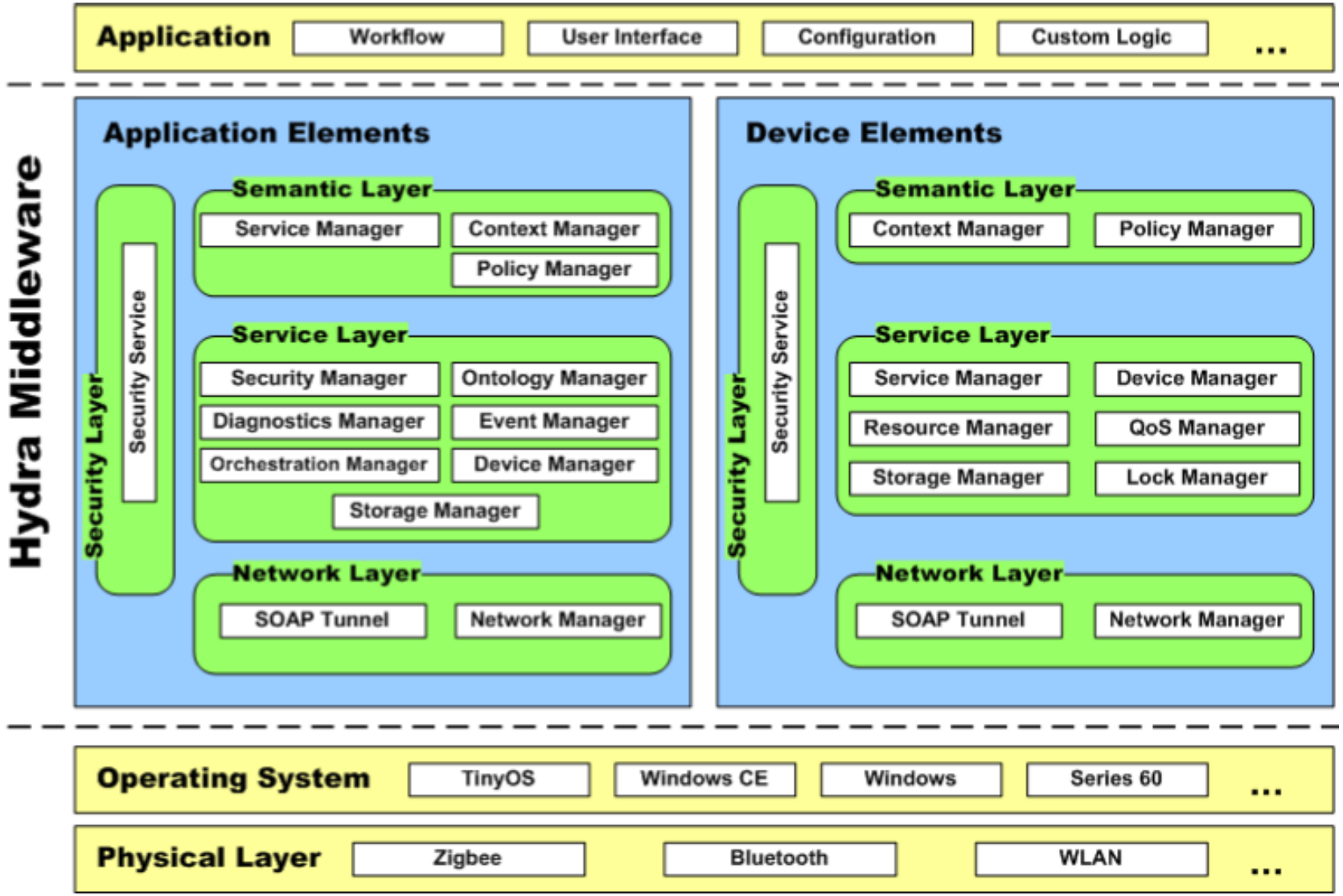
This motivates scientists to develop new technologies for the everyday people's lives.

Typical IoT Middleware

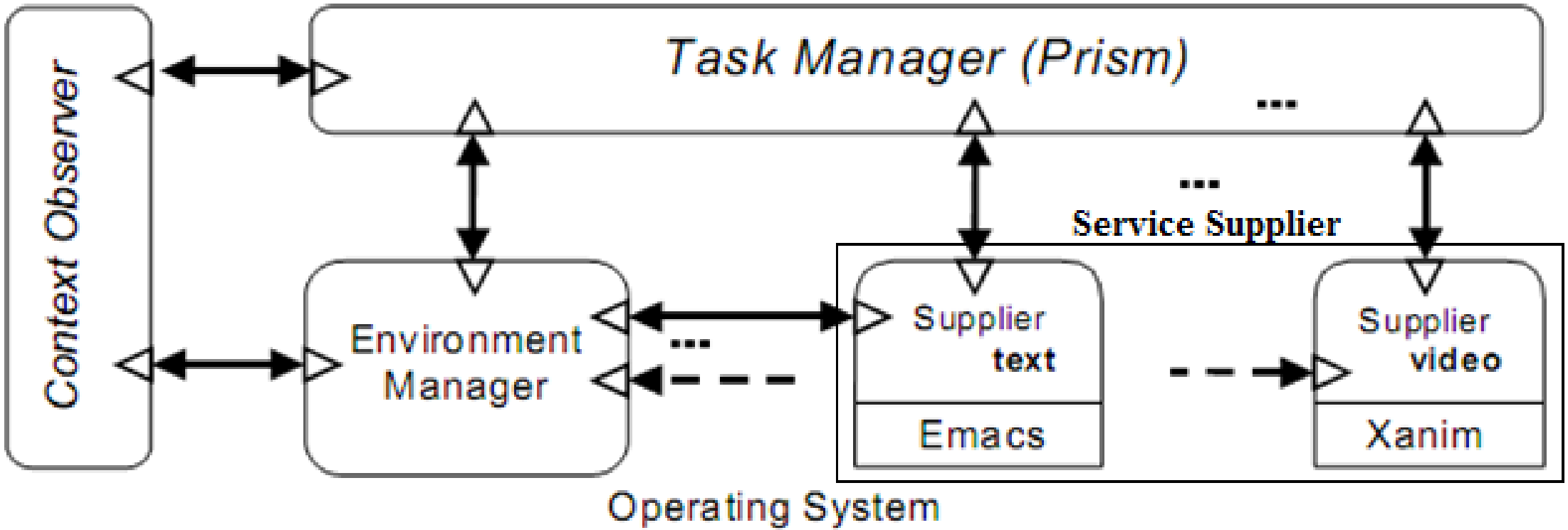
- HYDRA it is the most popular and well documented middleware in comparer to the mentioned middleware.
- AURA is middleware focus on elaboration and manipulation of the gathered data from devices. Its aim is to provide ease of configuration and deployment for end user and developer. Since to meet this goal, we require to facilitate the gathered data manipulation, AURA that was the only middleware among that considered the data manipulation.
- TinyDB focus on gathering data from devices. Since in an IoT-based system we need to gather data from environment through different devices, TinyDB that is a popular middleware.
- WiseMID is the only middleware among the reviewed middleware that is specific for energy saving purpose. As saving the energy of devices is important issue.

HYDRA

Components inside and outside of Hydra middleware



AURA



The components of the AURA framework architecture and their interactions

TinyDB

TinyDB middleware was the first project to propose the idea of abstracting from devices. TinyDB allows end-users to interact with devices without knowing about the details of the devices specification, such as the communication protocols that are supported by these devices.

Monitoring Queries

It asks the value of one or more attributes periodically and continuously, such as, e.g., reporting the temperature of a warehouse every hour.

Network Health Queries

It provides information about the network itself. For instance, selecting neighbouring nodes, with a battery lifetime larger than a threshold.

Exploratory Query

It shows the status of a specific node or a set of nodes at a specific time, such as, e.g., selecting the temperature of the sensor with same specific id.

Actuation Query

This kind of query can be used to ask for a physical action. For instance, an enduser of a system wants to turn off a fan in a room when the temperature of the room is lower than a threshold.

WISeMid

- WISeMid is an energy-aware middleware for integrating wireless sensor networks and Internet.
- In an IoT-based system, saving energy in interaction among devices is important, because they usually have limited power suppliers.
- Also, IoT-based system is IP-based communication.
- Therefore, as the WISeMid middleware considers both Ipbased communication and energy awareness factors.

Aggregation service

The goal of the data aggregation service is to aggregate correlated or redundant data, and reduce the overall size of transferred data in a network. In this way, we can decrease the network traffic and save energy by decreasing the interactions with devices. In this service, a user sends one request and receives an answer based on an aggregation of the last values of the requested devices. In this way the number of transaction decreases and energy can be saved.

Reply Storage Timeout

This service stops sending the same messages with the same parameters. For instance, if the sensed data of a device is fixed for a specific period of time, we can send only one request to the considered device, and then use the reply message to answer to all of the equivalent request messages arriving during that specific period. Therefore, WISeMid prevents the system from getting information from sensors when the data is still up-to-date.

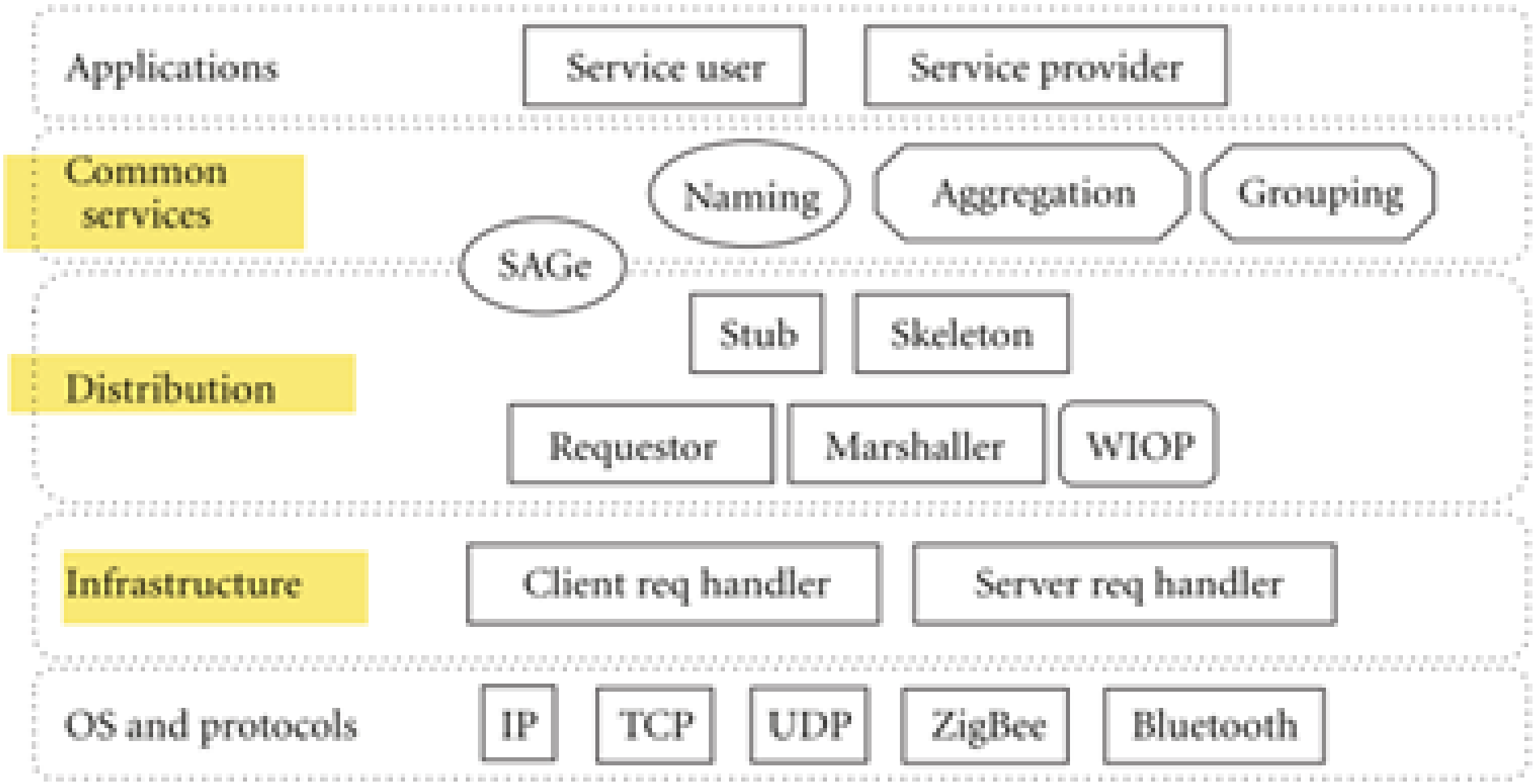
Atomic Type Conversion

This service decreases the size of messages in an IoT-based system. For instance, if we define an integer data type for a field that gets a numeric value like „1" in this case, we many bytes are unnecessarily used. To save bytes, the argument format can convert from Integer to Short. Thus, WISeMid removes unnecessary bytes from messages.

Invocation asynchrony patterns

This service provides four patterns to handle the end-user requests in an asynchronous way. These patterns prevent the system waste time with blocking, when requests can be handled in an asynchronous way.

WISeMid architecture



Comparative Analysis of the Middleware

Functional component	Aura	Hydra	TinyDB	WISeMid
Application Abstraction	✓	✓	×	✓
Central control, context detection & Management	✓	✓	×	✓
Device Abstraction	×	✓	✓	×
Interface Protocol	×	✓	✓	✓

IoT-middleware comparison.

IoT Middleware	Features of Middleware				
	Device Management	Interoperation	Platform Portability	Context Awareness	Security and Privacy
HYDRA	✓	✓	✓	✓	✓
ISMB	✓	×	✓	×	×
ASPIRE	✓	×	✓	×	×
UBIWARE	✓	×	✓	✓	×
UBISOAP	✓	✓	✓	×	×
UBIROAD	✓	✓	✓	✓	✓
GSN	✓	×	✓	×	✓
SMEPP	✓	×	✓	✓	✓
SOCRADES	✓	✓	✓	×	✓
SIRENA	✓	✓	✓	×	✓
WHEREX	✓	✓	✓	×	×

IoT-middleware Interfaces

IoT Middleware	Interface protocols				
	Zigbee	RFID	WiFi	Bluetooth	Sensor (others)
HYDRA	✓	✓	✓	✓	✓
ISMB	×	✓	×	×	✓
ASPIRE	×	✓	×	×	×
UBIWARE	×	✓	✓	×	✓
UBISOAP	×	✓	✓	×	✓
UBIROAD	×	✓	✓	✓	✓
GSN	×	✓	✓	×	✓ IEEE-1451
SMEPP	×	×	✓	✓	✓
SOCRADES	×	✓	×	×	✓
SIRENA	×	✓	×	✓	✓
WHEREX	✓	✓	✓	✓	✓

Thank You