# Interoperable IoT Architectures – Web of Things

The purpose of the WoT Working Group is to counter the fragmentation of the IoT through the specification of building blocks that enable easy integration of IoT device and services across IoT platforms and application domains.

**Terminology W3C**

● Thing Description (TD): Structured data describing a Thing, comprises general metadata, domain-specific metadata, Interaction Affordances ,and links to related Things.

● Binding Templates: A re-usable collection of blueprints for the communication with different IoT platforms. The blueprints provide information to map Interaction Affordances to platform-specific messages through WoT Thing Description as well as implementation notes for the required protocol stacks or dedicated communication drivers.

● Consumed Thing: A software abstraction that represents a remote Thing used by the local application. The abstraction might be created by a native WoT Runtime, or instantiated as an object through the WoT Scripting API.

● Consuming a Thing: To parse and process a TD document and from it create a Consumed Thing software abstraction as interface for the application in the local runtime environment.

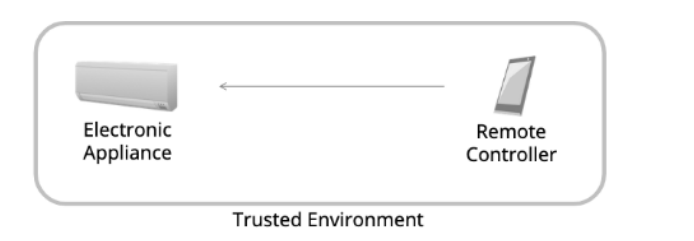
● Exposed Thing: A software abstraction that represents a locally hosted Thing that can be accessed over the network by remote Consumers. The abstraction might be created by a native WoT Runtime, or instantiated as an object through the WoT Scripting API.

● Thing Directory: A directory service for TDs that provides a Web interface to register TDs (similar to [CoRE-RD]) and look them up

**Common Patterns**

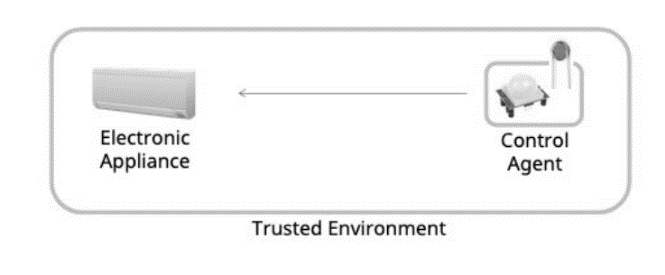
Have the role to illustrate how device/things interact with controllers, other devices, agents and servers

* Client role as the initiator of a transport protocol
* Server role as passive
* A device can be client and server simultaneously

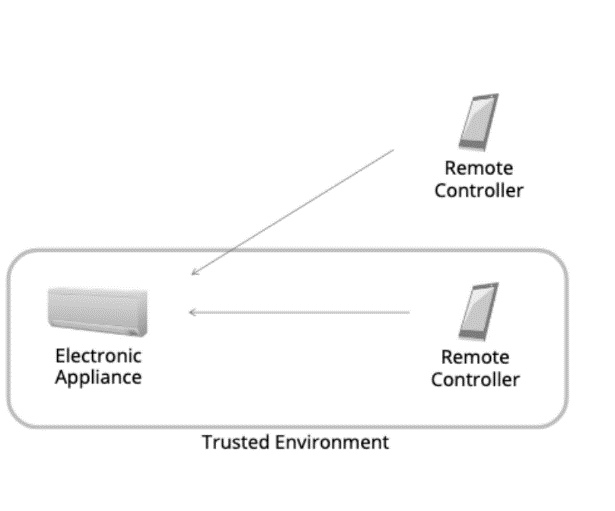
**Pattern 🡪 Device controllers**

Controlled by user-operated remote controller

* In this pattern there is an electronic appliance that has server role that can accept request from the other devices and responds to them.
* The other device is the remove controller that has client role (browser or native app), can send request like to read sensor value or turn on a device.

**Pattern 🡪 Thing to Thing**

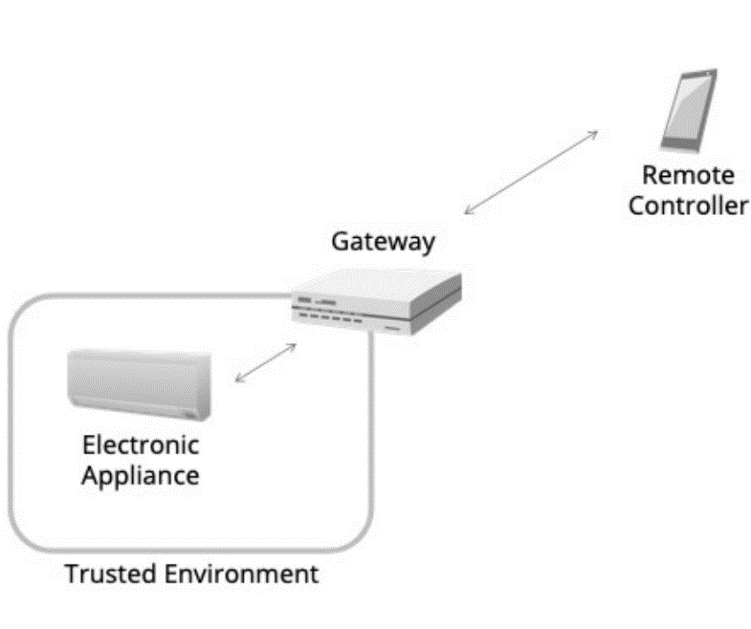
A sensor detect a change on the room condition and notify to turn on the electronic appliance

* The sensor can notify a change of state to other devices
* In this case two devices have the server role so at least one device must also have a client role.

**Pattern 🡪 Remote Access**

The remote controller can switch between different network and protocols.

* When the controller is in the local network it is a trusted device and **no additional**  networking **security** or **access control** is required
* Outside instead is not a trusted network
* In this case the remote controller and the electronic appliance have the a client and a server role.

**Pattern 🡪 Smart Home Gateway**

The gateway is placed between a local network and the internet.

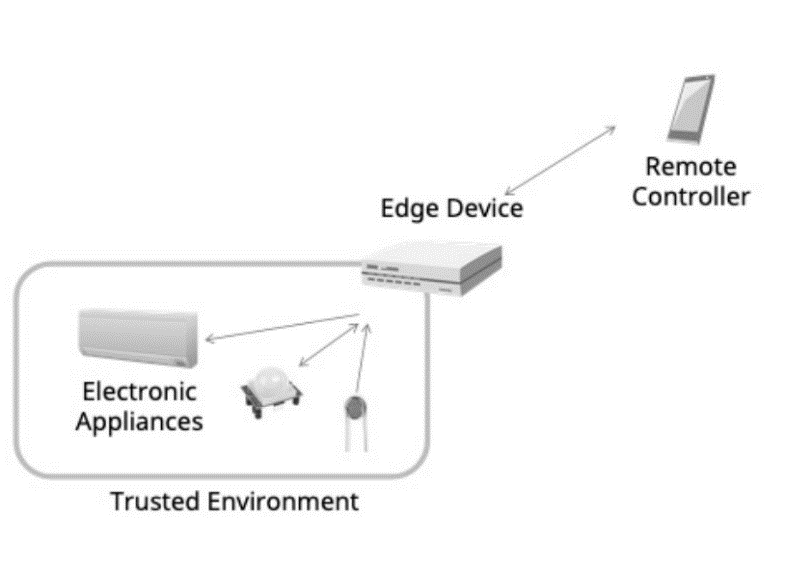
Can Receive commands

It is also a virtual representation of a device.

Typically offers **proxy**  and **firewall** functionality.

The gateway cover both client and server role.

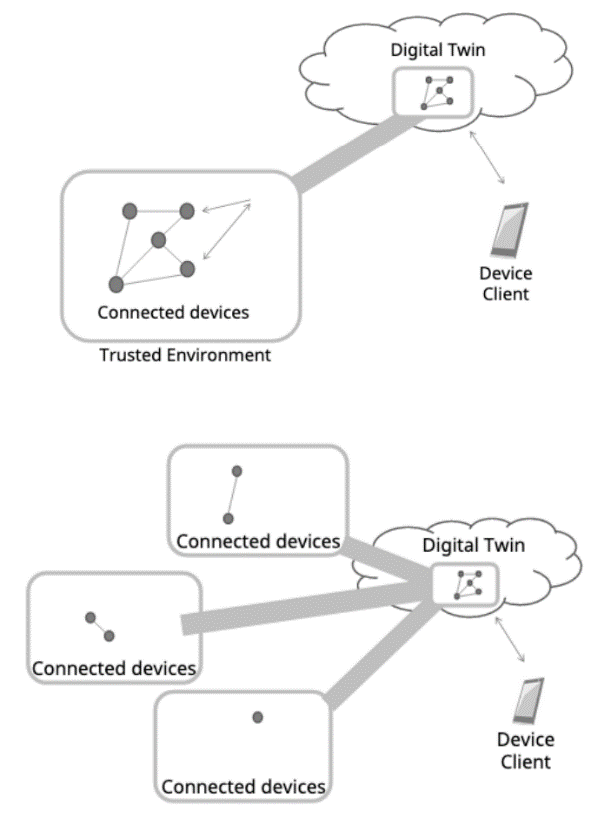
The gateway act as a server for the appliance and as a client for the remove controller.



**Pattern 🡪 Edge Devices**

While the gateway work as bridge, the edge device has local compute capabilities and typically bridges between different protocols

Edge device can provide **preprocessing**, **filtering**  and **aggregation** of data provided by connected devices and sensors

**Pattern 🡪 Digital Twins**

Is a virtual representation, model of a device or a group of devices that resides on a cloud server or edge device

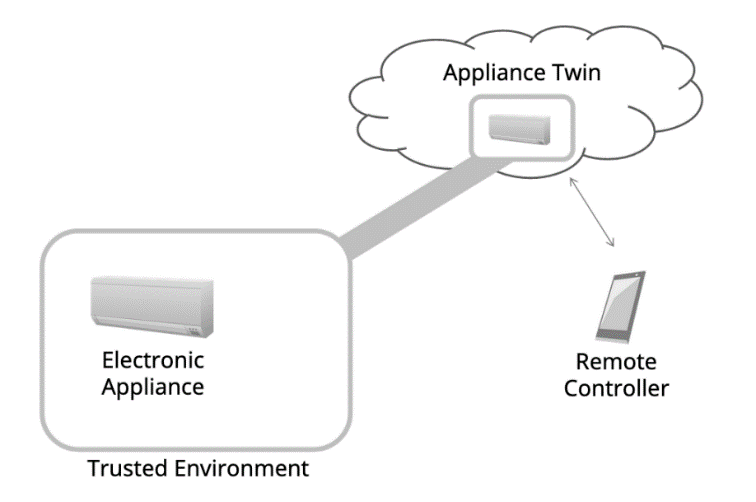
* Can be used to represent real-world devices which may not be always online.

**Cloud Ready devices**

In this patterns appliance are directly connected to the cloud.

The cloud mirrors the appliances and, acting as a digital twin.

Authorized controllers can be located anywhere.

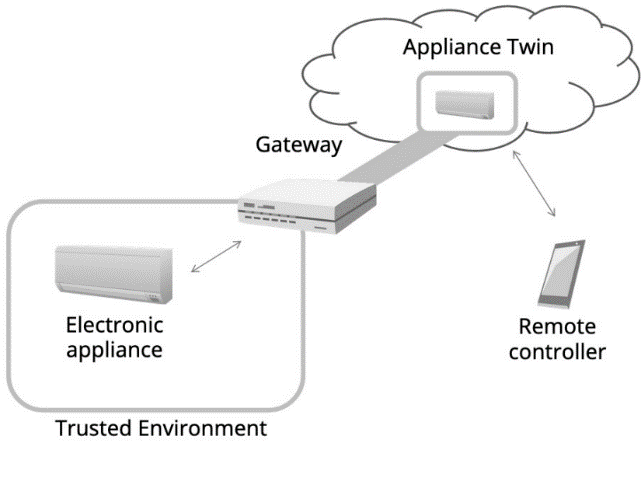


**Legacy Devices**

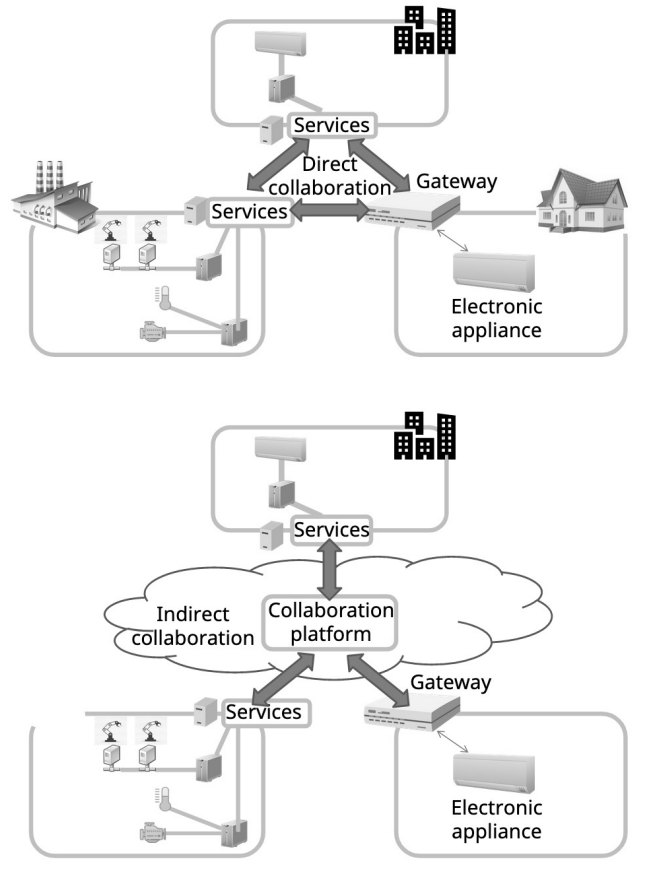
In this case cannot directly connect to the cloud.

The gateway is needed to relay the connection and works as:

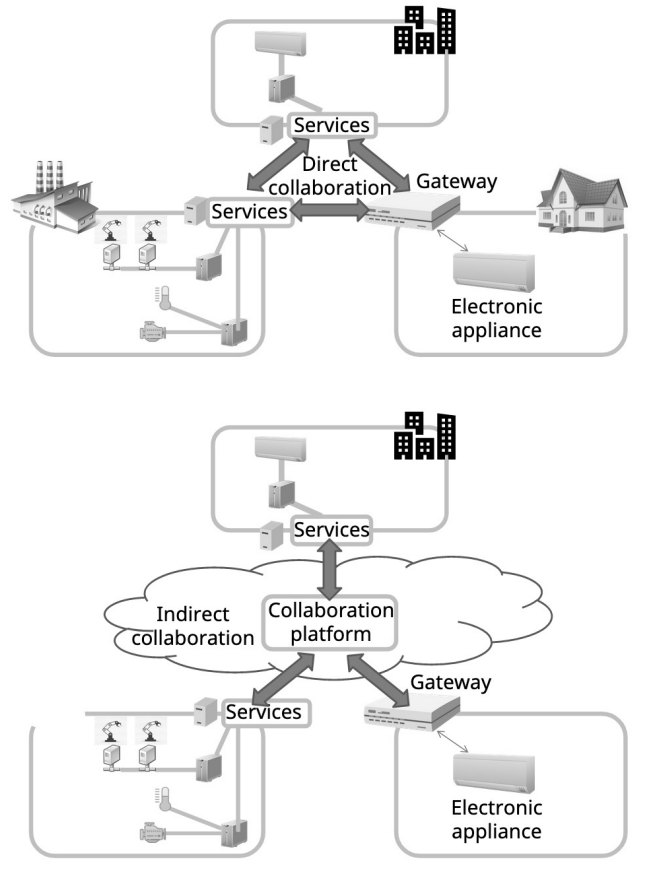
🡪Integrator of protocols

* Firewall
* Privacy
* Local agent (in case of network connection disruption)
* Emergency Service (Running locally)

The cloud mirrors the gateway with all connected appliance and act as a digital twin and also receive command from remote controllers

**Pattern 🡪 Multi Cloud (More than one cloud)**

**Pattern 🡪 Cross-domain Collaboration**

In this pattern each system involves and interact with other systems in other domain. This type of system is called **Symbiotic ecosystem**

* Direct Collaboration: model, systems exchange info in a peer-to-peer manner
* Indirect Collaboration: systems exchange info via some collaboration platform.

**W3C Wot Architectural Common Principles**

Flexibility, Compatibility, Scalability and Interoperability

**W3C Wot \_Things Functionalities**

Reading, updating, subscribing, invoking functions, subscribing to event notification

WoT Support also a mechanism which enables describing things and their functions:

* Should not be only human-readable, but also machine-readable.
* Should allow semantic annotation
* Should be able to be exchanged using multiple formats
* Should support also internalization

Attributes described such as name, explanation, version, format and description, link to other related things.

**WoT 🡪 Devices**

Access to devices is made using a description (TD) of their functions and interfaces

The **Thing Description** includes a general metadata about the deice, information models, representing functions, transport protocol description, and security info.

General metadata contains device identifiers (**URI),** device info such as **serial number, production date, location** and other human readable info.

Info models defines device attributes, and represent device’s internal settings, control functionality and notification functionality.

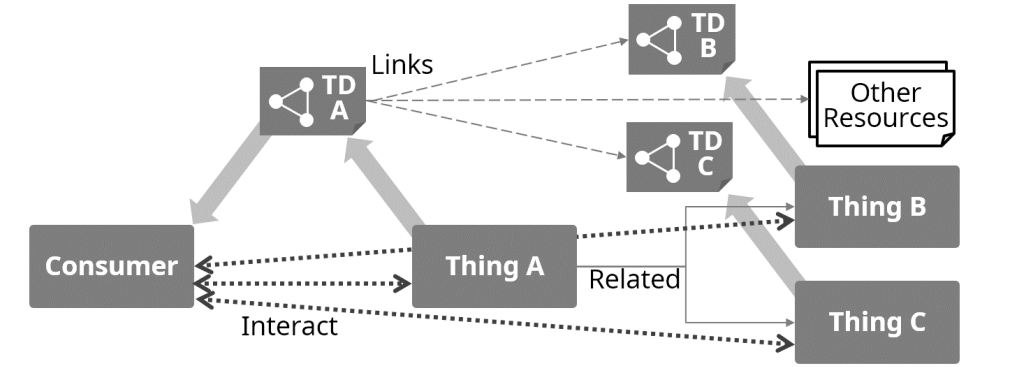
**Security** info includes description about **authentication, authorization** and secure **communications**.

TD must be accessible to other components.

**WoT 🡪 Applications**

Need to be able to generate and use network and program interfaces based on metadata and descriptions.

Must be able to obtain these descriptions, therefore need t e able to conduct search operations.



**WoT 🡪 Digital Twins**

Need to generate program interfaces internally based on metadata.

A Twin has to produce a description for the virtual device.

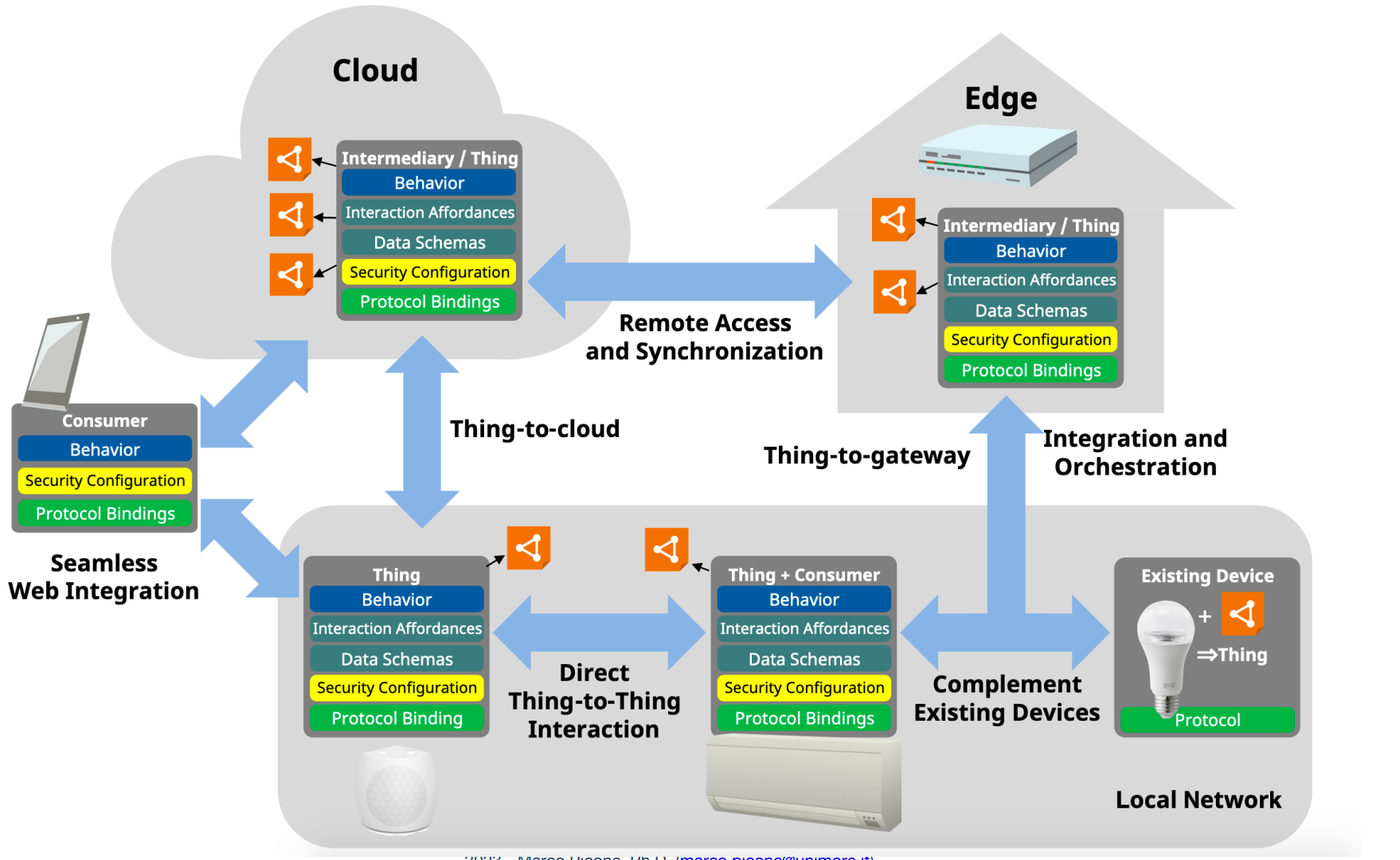
Id of vt devices must be newly assigned, therefore, are different from the original devices.

Transport and security mechanism and settings can be different from the original devices if necessary

**WoT 🡪 Discovery**

Directory can serve the requirement of sharing **TDs** so devices and twins can automatically or allow the users to register the TD.

**Descriptions** of the devices need to be **searchable.**



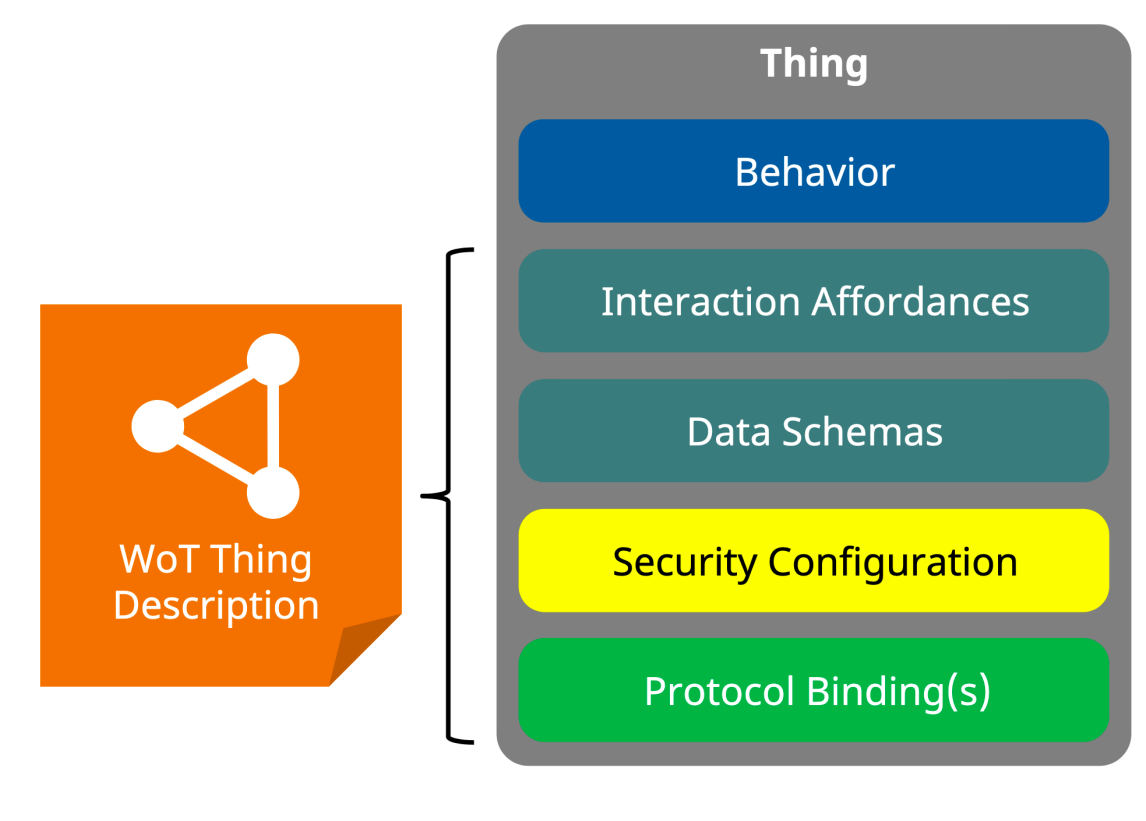
**WoT 🡪 Architecture - Affordances**

A central aspect is to provide machine-understandable metadata

A key to this self-descriptiveness lies in the concept of affordances

Affordance refers to the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly used.

* Descriptions: In the TD the affordances must be described.
* Interoperability
* Communication & Interaction: allow WoT devices to communicate and interact with each other
* Security & Authorization



**Behavior** aspect of a Thing includes both the autonomous behavior and the handlers for the Interaction Affordance.

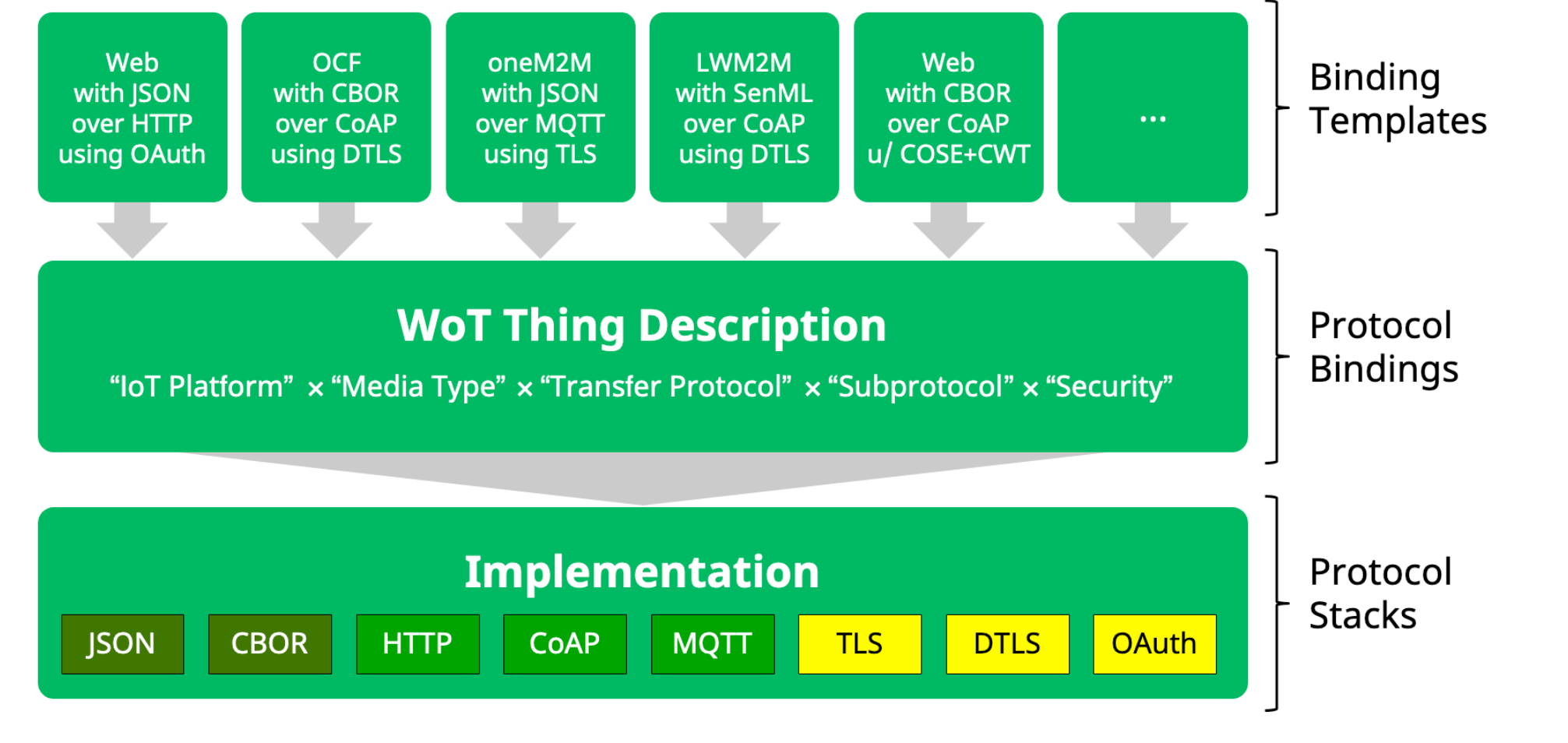
**The Interaction affordances** provide a model of how consumers can interact with the thing.

**Data schemas** describes the info model and the related payload structure and the items passed between the Things and Consumers during interactions.

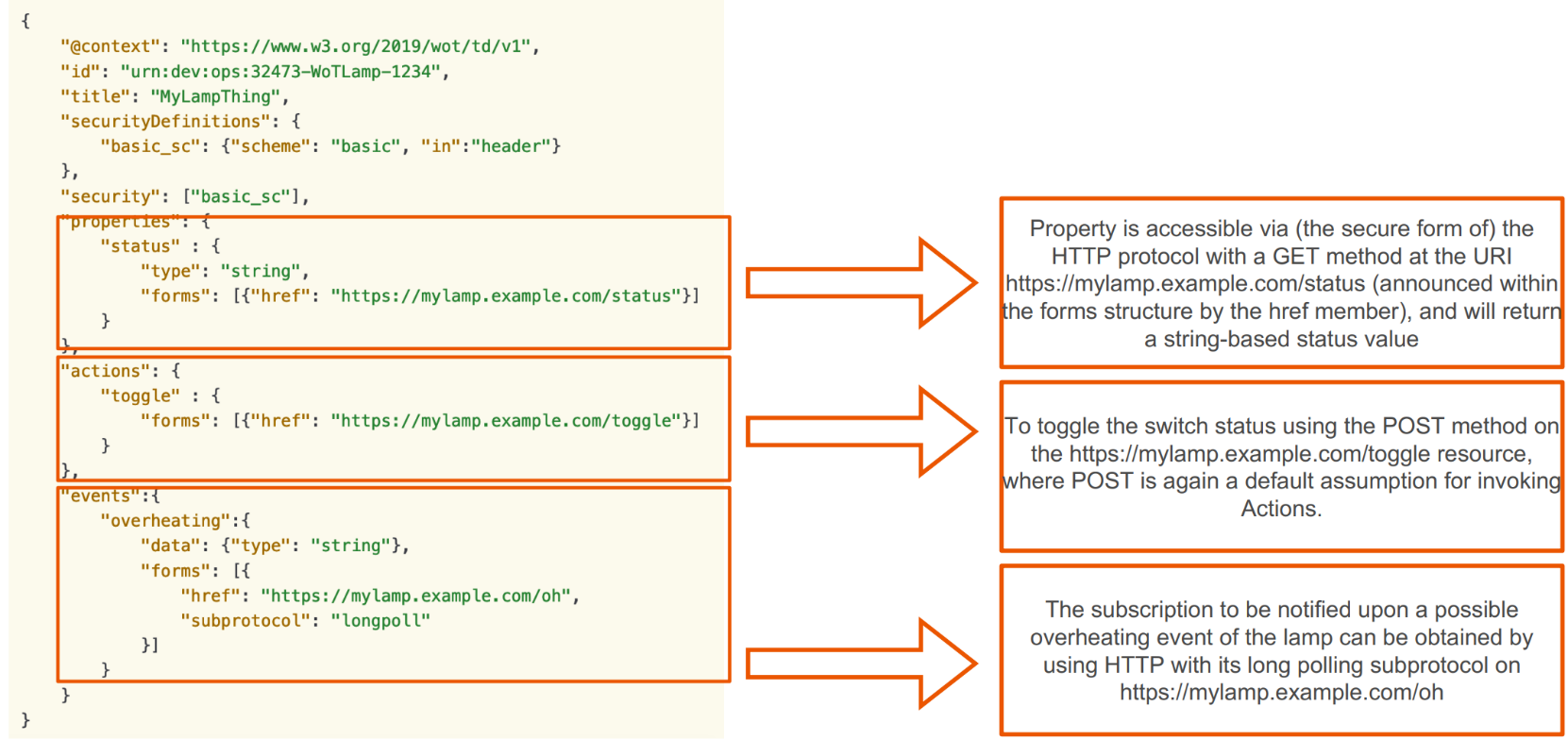
**The Protocol Binding** add the additional detail needed to map any interaction with a message of a determinate protocol

**The security configuration** aspect of a thing are the mechanism used to access the Interaction affordance.

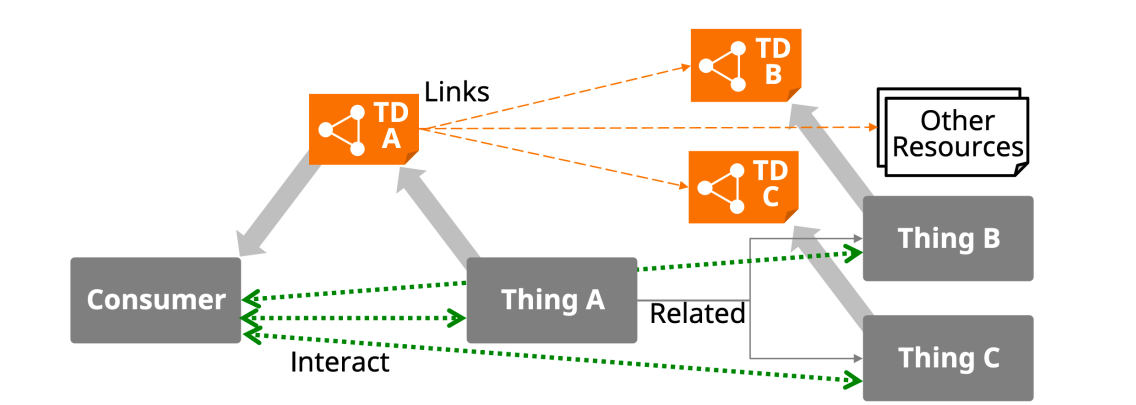
**Protocol binding:**



Thing Description is based on JSON

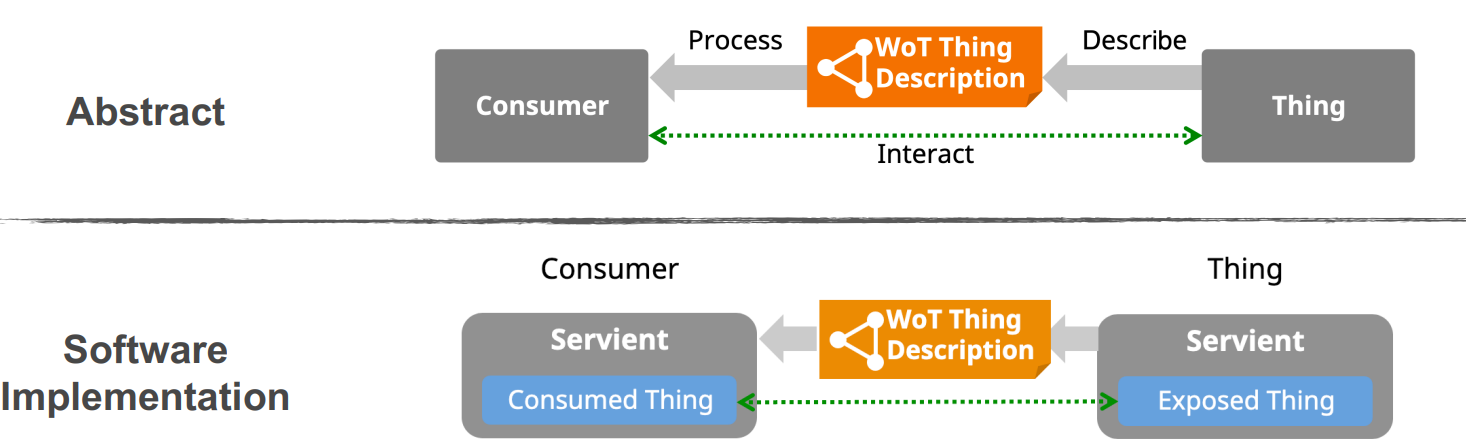
Example of a TD:

**Things Linking**

****

Linking do not apply hierarchical Things but relations between things and other resources.

System components and their Interconnectivity

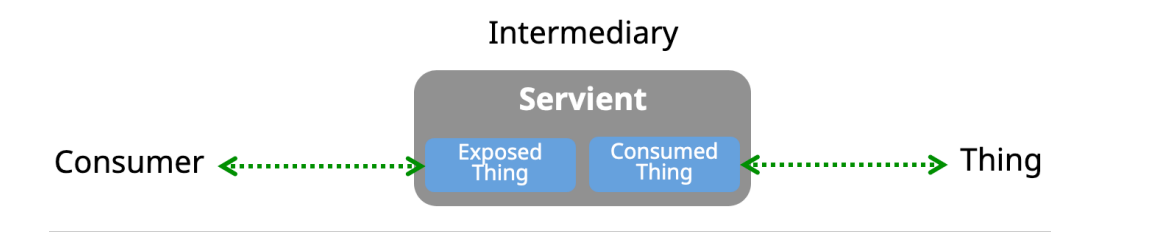


Direct communication applies when both Servients use the same network protocol(s) and are accessible to each other

In a Thing, a Servient software stack contains a representation of a Thing called Exposed Thing

It makes its WoT interface available t Consumers of the Thing

Intermediary is yes another WoT architecture implemented by a servient like the consumer



Is located between a Thing and its consumers.

Consumer and Intermediary can communicate in a different protocol than Intermediary and Thing.

The purpose can be relaying, optionally caching or transforming communication.

Take a look to discovery approach and use case od WoT last 20 pages.