14th Swedish National Computer Networking Workshop SNCW-2018

AVATAR CONCEPTION FOR "THING" REPRESENTATION IN INTERNET OF THINGS

Ievgeniia Kuzminykh

Post-doc at Computer Science Dept., Blekinge Institute of Technology, Karlskrona, Sweden

INTRODUCTION

The complexity of ensuring IoT security is that the system is heterogeneous, consists of many assets on each of the architecture layer. Many experts in IoT security focus on threat analysis [1] and risk assessments to estimate the impact if a security incident or a breach occurs.

In order to provide the general security requirements for the IoT system using threat risk modelling, the first thing to do is to identify the main security stakeholders, security assets, possible attacks, and, finally, threats for the IoT system. Using this general IoT threat model as a basis you can create a specific set of security objectives for a specific IoT application domain.

In this work, we will try to highlight the assets that necessary for further analysis of the treat model for Internet of Things. We will also specify the stakeholders who are the connecting link between IoT devices, services and customers, as well as link between transfer and displaying the client commands onto smart things.

For describing the model of component interaction in IoT system we will use **the avatar-oriented approach** since it allows us to merge objects into a system of objects. IoT Service has a more complex structure than a single entity. The application can use several services to display all information to end user, can aggregate data from several devices.

To manipulate data objects the avatar representation approach is most appropriate, then you can easily connect or disconnect microservices, data from things, visual representation of data.

CLASSICAL IOT ARCHITECTURE VS. AVATAR – ORIENTED

The classical architecture (Fig.1) displays a data flow from the end device to storage and data handler from where the interaction with user begins. But apart from the time characteristics of the assets and the physical interaction of components there are certain actions and events that occur in the system, as well as the reaction to these events.

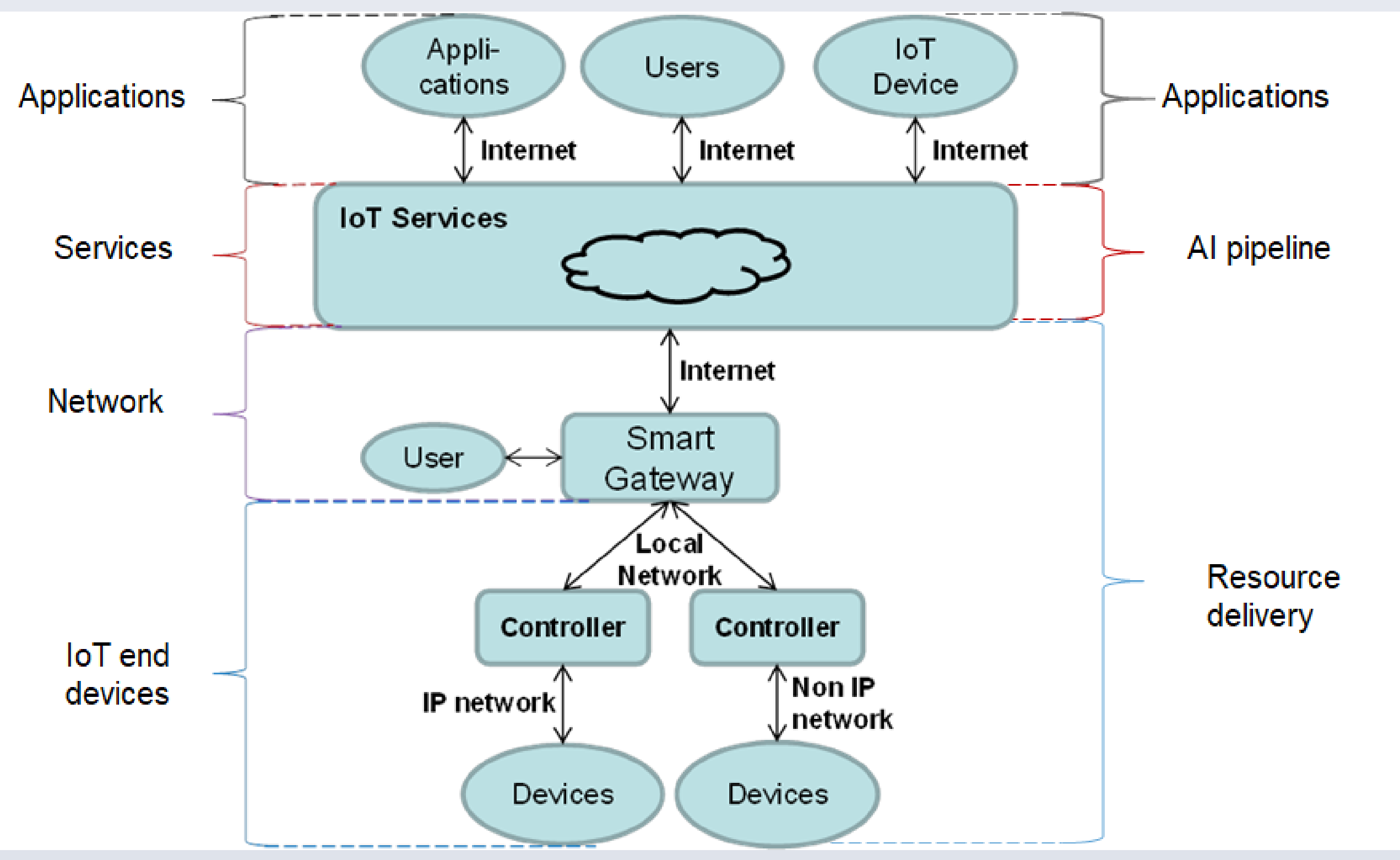


Fig.1 Classical IoT architecture

Such representation is closer to security since it contains assets that are more convenient for manipulating the security language such as an action (property), an event, a reaction, which can be interpreted in the language of risk theory as: for action is damage, exploit, risk, threat, vulnerability, for event is attack, for reaction - logging, countermeasure.

The presentation of avatar-oriented model of interaction of IoT components is shown on Fig.2 and does not in any way eliminates the use of classical architecture.

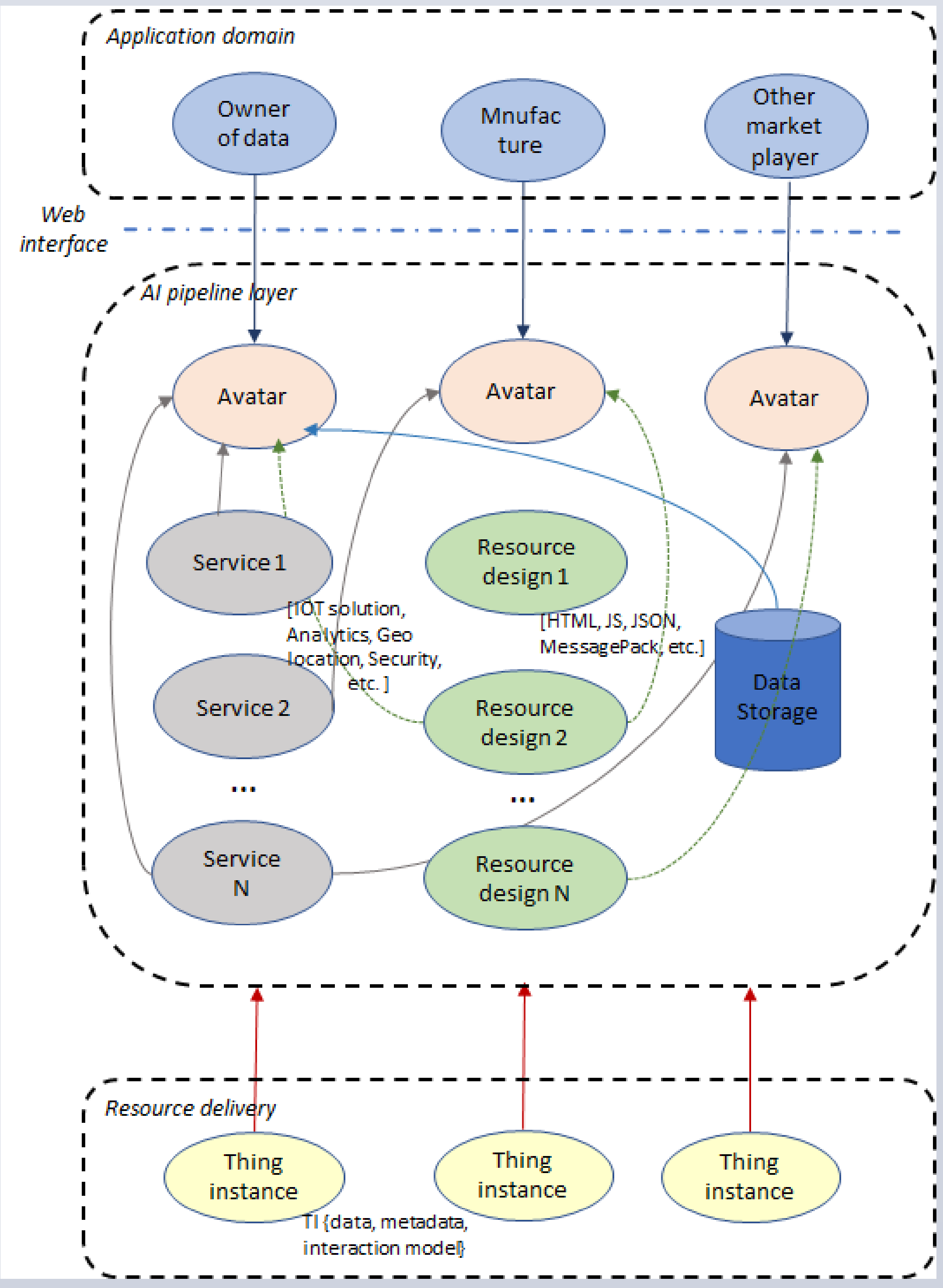
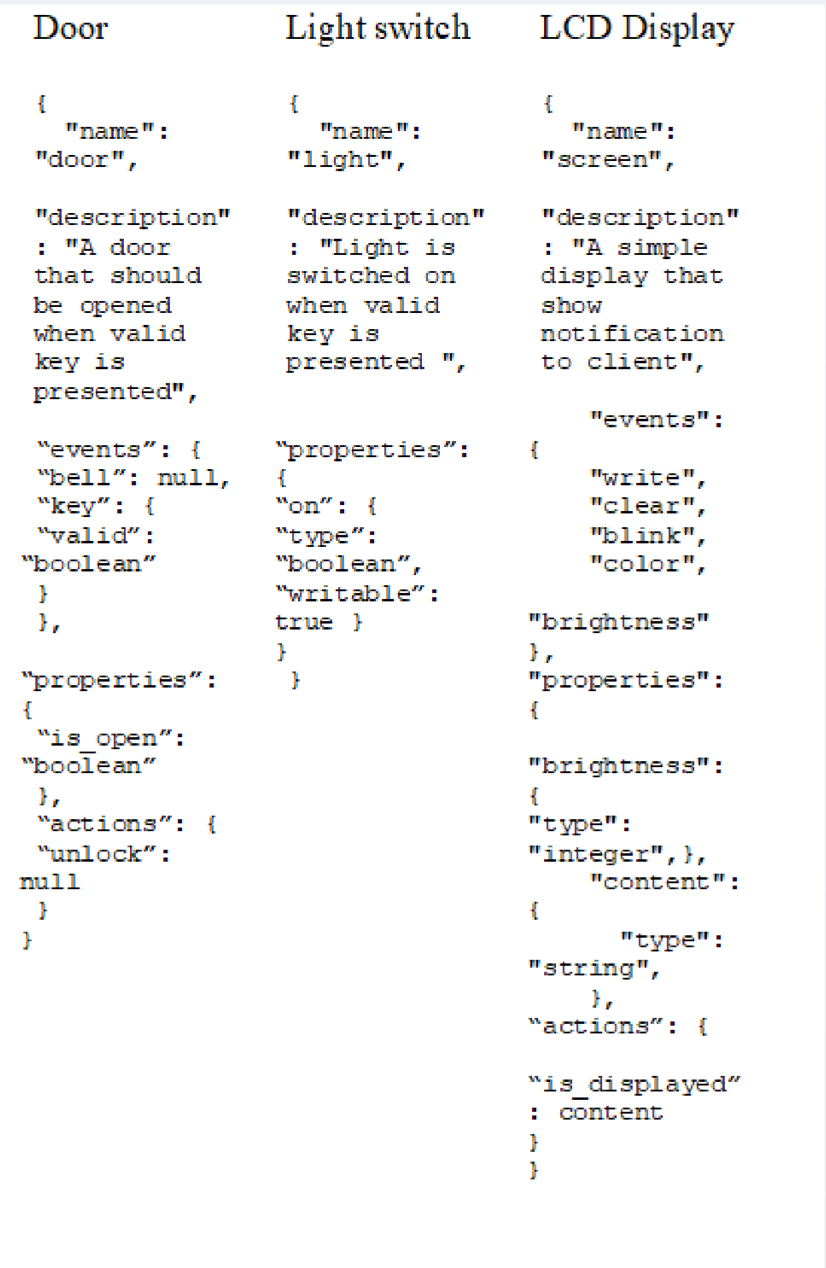


Fig.2 Avatar-oriented IOT architecture

WHAT IS "THING" INSTANCE?

Over IoT system the data and meta data circulate, last one describes the type of data and interaction models inherented for particular application platform or service [2]. .

Applications and services do not need primitive data from sensors, they need data of a higher level. At the level of the resource a "thing instance" is created.



Door

```
{
  "name":
"door",

"description"
: "A door
that should
be opened
when valid
key is
presented",

 "events": {
 "bell": null,
 "key": {
 "valid":
"boolean"
 }
 },

"properties":
{
 "is_open":
"boolean"
 },
 "actions": {
 "unlock":
null
 }
}
```

Light switch

```
{
  "name":
"light",

"description"
: "Light is
switched on
when valid
key is
presented ",

"properties":
{
"on": {
"type":
"boolean",
"writable":
true }
}
 }
```

LCD Display

```
{
  "name":
"screen",

"description"
: "A simple
display that
show
notification
to client",
   "events":
{
   "write",
   "clear",
   "blink",
   "color",

"brightness"
},
"properties":
{

"brightness":
{
"type":
"integer",},
   "content":
{
     "type":
"string",
   },
"actions": {

"is_displayed"
: content
}
}
```

Fig.3 Example of Thing Instance

Thing instance (TI) is an object representation of the merged data that contains the *data*, *metadata*, *interaction model* attributes, requirements for *communication* and *security*.

Data – information that thing provides to user.

Metadata – supporting information about Thing Instance. It includes Protocols and ports, Data formats&encodings, Multiplexing and buffering of data, Efficient use of protocols, ect.

Interaction model - link from a Thing to the interaction patterns it provides.

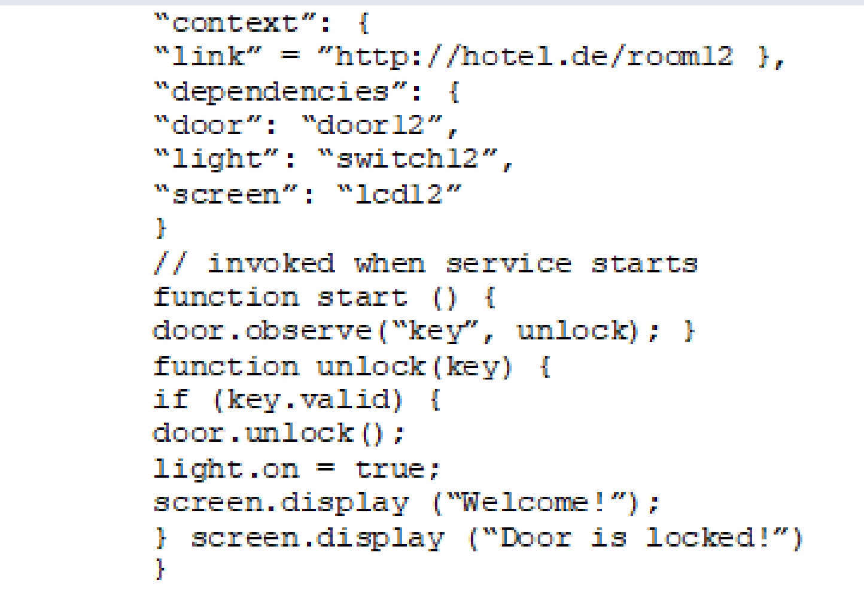
Security - links a given Thing to the security information that indicates the access metadata information for securely transmitting information via all the resources of the Thing.

Link - provides Web links to arbitrary resources that relate to the specified Thing Instance.

The *interaction model* should support multiple interaction patterns and messaging methods. By default, Interaction Patterns contains of such assets such as *Property*, *Action*, and *Event*.

WHAT IS AVATAR?

- Each Thing Instance can have one or more virtual representations of physical or abstract entities which are called *avatars*.
- *Avatars* have attributes such as history, patterns of interaction, description, services, identifiers, access control policy, data processing policy, security policy.
- *Avatars* have URIs and are accessible via the web interface.
- *Avatars* allow to simplify the collection of services and applications that can use information from different sources.



```
"context": {
"link" = "http://hotel.de/room12 },
"dependencies": {
"door": "door12",
"light": "switch12",
"screen": "lcd12"
}
// invoked when service starts
function start () {
door.observe("key", unlock); }
function unlock(key) {
if (key.valid) {
door.unlock();
light.on = true;
screen.display ("Welcome!");
} screen.display ("Door is locked!")
}
```

Fig.3 Example of avatar representation of "smart room"

STAKEHOLDERS AND ASSETS

At each layer of the avatar-oriented IoT model it is possible to identify specific stakeholders. At the resource delivery layer when instance object is creating the stakeholder is *Manufacturer of the Device*.

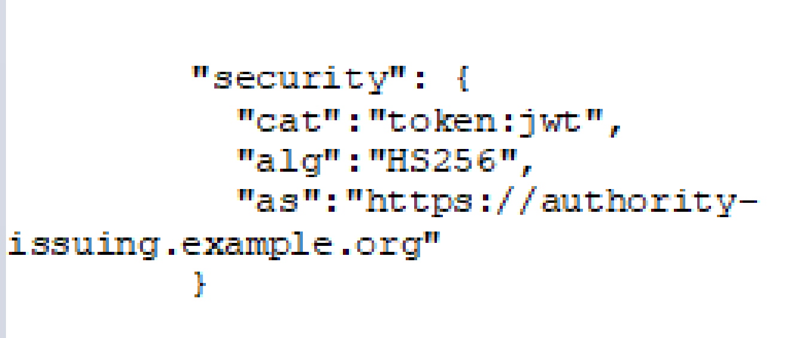
The next stakeholder is the *Thing Provider* which uses the thing instance to build various specific solutions for different IoT domains.

Stakeholder *Thing User* can be either a physical user or an abstract user. *Thing User* can differ in the functionality of using avatars and data itself, depending on the access rights.

Having information about the stakeholders and functional layers of IoT architecture, the assets for security can be allocated. We can specify assets such as *Thing user data*, *Thing provider data*, *Thing instance itself*, *Interface* (Administrative, Device Web Interface, Cloud Interface, Mobile Application) [3].

WHAT ABOUT SECURITY?

Web of Things [4] framework gives very limited approach for implementation security aspects in thing instance. Just couple of line presented in non-official draft of WoT standard. JSON Web Token (JWT) type is assigned (cat), the corresponding hashing algorithm "HS256" (alg), and issuing authority of the security token (as).



```
          "security": {
             "cat":"token:jwt",
             "alg":"HS256",
             "as":"https://authority-
issuing.example.org"
          }
```

Fig.4 Security requirements for TI

DISCUSSION

For now, security metadata is defined as optional! That is why it is big challenge for researchers and software developers to implement security methods and mechanism for IoT that is avatar-oriented. Among the tasks under development are ensuring privacy and protecting Thing and related Assets against attacks.

REFERENCES

[1] ETSI TR 103 167 V1.1.1 Machine-to-Machine Communications (M2M). Tech. recommendation, 2011

[2] Web of Things (WoT) Thing Description. W3C Draft, April 2018. https://w3c.github.io/wot-thing-description/#introduction

[3] OWASP Internet of Things Project. https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project

[4]Web of Things (WoT) Security and Privacy Considerations. W3C Draft, April 2018. https://rawgit.com/w3c/wot-security/master/index.html#introduction