



UNIVERSITÀ  
DEGLI STUDI  
FIRENZE

# FIWARE

## a new interoperable framework

ESAME: SOFTWARE ARCHITECTURES  
AND METHODOLOGIES

PROFESSORI: VICARIO ENRICO,  
PARRI JACOPO, SAMPIETRO SAMUELE

RELATORE: PIETRO BERNABEI

MATRICOLA: 7064862

# Obiettivi

- Scoprire il **framework** FIWARE e il suo ecosistema
- Mettere l'accento sulla capacità di facilitare **l'interoperabilità**:
  - Attraverso lo sviluppo di un **prototipo**



# Indice Argomenti

FIWARE

NGSI-LD

Smart  
DataModels

CoBrA

Progetto

Riflessioni

Conclusioni





# Digital Twin

## Digital Twins

Un gemello digitale è un modello virtuale di un oggetto fisico. Esegue il **ciclo di vita** dell'oggetto e utilizza i **dati in tempo reale** inviati dai **sensori** sull'oggetto per **simulare** il comportamento e **monitorare** le operazioni. I gemelli digitali possono replicare molti elementi del mondo reale, da singole apparecchiature in una fabbrica a installazioni complete, come turbine eoliche e persino intere città. La tecnologia dei gemelli digitali consente di supervisionare le prestazioni di una risorsa, identificare potenziali guasti e prendere decisioni più informate sulla manutenzione e sul ciclo di vita.



Internet dei Digital Twins

**Making cities inclusive, safe, resilient and sustainable**



# NGSI-LD Standard

ETSI GS CIM 009 V1.7.1 (2023-06)



**Context Information Management (CIM);  
NGSI-LD API**

# API

NGSI-LD ha lo scopo di definire un'API che consenta alle applicazioni di eseguire diverse operazioni sulle **context information** come:

- **interrogazioni e aggiornamenti** delle context information;
- **subscription/notification**;
- registrazione dei **context source** al sistema per interrogarli per possibili aggiornamenti.



# Modello Informativo

Il modello informativo NGSI-LD definisce anche:

- la **struttura** delle informazioni di contesto che devono essere supportate da un sistema NGSI-LD e
- specifica i **meccanismi di rappresentazione** dei dati che saranno utilizzati dall'API stessa.

# Modello Informativo

NGSI-LD  
Meta Model

Entity

Relationship

Property

NGSI-LD  
Cross-Domain  
Ontology

GeoProperty

Language  
Property

Temporal  
Property

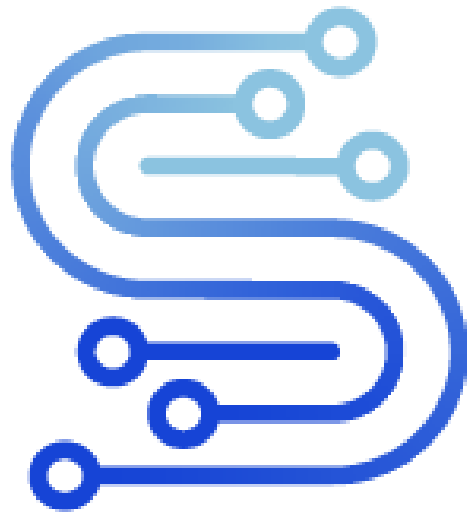
...

Domain  
Model

Machine

machine  
Component

machineModel



# Smart Data Models



# Scopo

- **Rappresentazione armonizzata** dei **formati** e della **semantica** per la pubblicazione e il consumo dei dati.
- Definire dei data model **comuni** e **compatibili**.
- Definire dei data model base da cui partire per una **personalizzazione**.
- La **standardizzazione dei formati** permette la creazione di un ambiente di **replicazione e interoperabilità** di soluzioni in settore diversi.
- Progetto Open Source



# Domain

SMART CITIES

SMART AGRIFOOD

SMART WATER

SMART ENERGY

SMART ENVIRONMENT

SMART SENSING

SMART AERONAUTICS

SMART DESTINATION

CROSS SECTOR

SMART ROBOTICS

SMART HEALTH

SMART MANUFACTURING



# Repository

- **Schema.json** - definisce gli attributi e i valori che possono assumere in un modello
- **examples\*** - esempi nei formati Normalized, KeyValues e JSON/JSON-LD
- **Notes.yaml** - note per la personalizzazione



# Schema.json

```
{
  "$schema": "http://json-schema.org/schema", 0
  "$schemaVersion": "0.0.1",
  "$id":
    "https://smart-data-models.github.io/dataModel.ManufacturingMachine/ManufacturingMachine/schema.json",
  "modelTags": "GSMA",
  "title": "Smart Data models - Manufacturing Machine dataModel schema,",
  "description": "Description of a generic machine",
  "type": "object",
  "required": [ "id", "type" ], 1
  "allOf": [ 2
    {
      "$ref":
        "https://smart-data-models.github.io/data-models/common-schema.json#/definitions/GSMA-Commons"
    },
    {
      "$ref": 3
        "https://smart-data-models.github.io/data-models/common-schema.json#/definitions/Location-Commons"
    }
  ],
  "properties": {
    "type": {
      "type": "string",
      "description": "Property. NGSI entity type. It has to be ManufacturingMachine",
```

```
    "enum": [
      "ManufacturingMachine" 4
    ],
    "machineModel": { 5
      "anyOf": [
        { 6
          "type": "string",
          "minLength": 1,
          "maxLength": 256,
          "pattern": "^[\\w\\-\\.\\{\\}\\$\\+\\*\\[\\]\\`~^@!,:\\\\\\\\]+$",
          "description": "Property. Identifier format of any NGSI entity"
        },
        {
          "type": "string",
          "format": "uri",
          "description": "Property. Identifier format of any NGSI entity"
        }
      ],
      "description": "Relationship. A reference to the associated Machine Model for this machine" 7
    },
    "countryOfManufacture": { 8
      "type": "string",
      "description": "Property. Model:'https://schema.org/Text'. The country where this machine was manufactured"
    },
    ...
  }
}
```





# JSON+LD

- JSON-LD estende il concetto di JavaScript Object Notation con i **Linked Data**.
  - link a **risorse esterne**.
  - Per condividere dati referenziati tra loro.
- Dato questo scambio di linked data presenti su domini diversi, NGSI-LD introduce il concetto di **@Context** all'interno del formato JSON.



## @Context

- Il @Context permette di **associare** a tutti i termini presenti nel JSON una **definizione semantica certa**.
  - Con il fine di **ridurre l'ambiguità** nella loro interpretazione.
- Scopo:
  - di garantire **l'interoperabilità** tra sistemi;
  - la riduzione degli errori causati da una **errata interpretazione** o da un **errato popolamento dei dati**.
- Definizione del contenuto:
  - Linked Data a risorse esterne;
  - Definizione diretta.



# @Context

```
"@context": [  
  {  
    "operationComposite": "urn:operationComposite.property"  
  },  
  "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"  
],
```

Definizione diretta

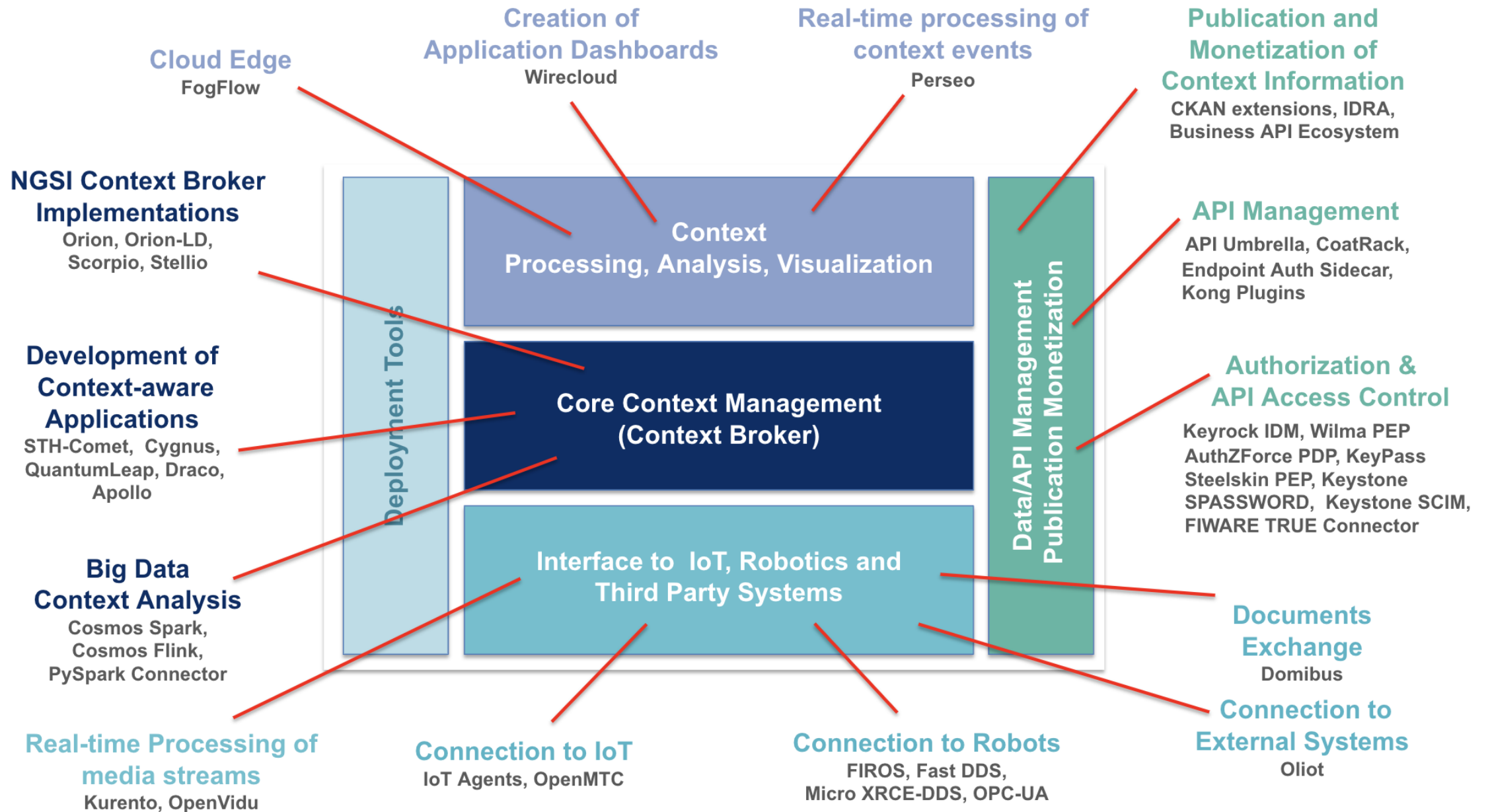


Linked Data

A breve un esempio di payload

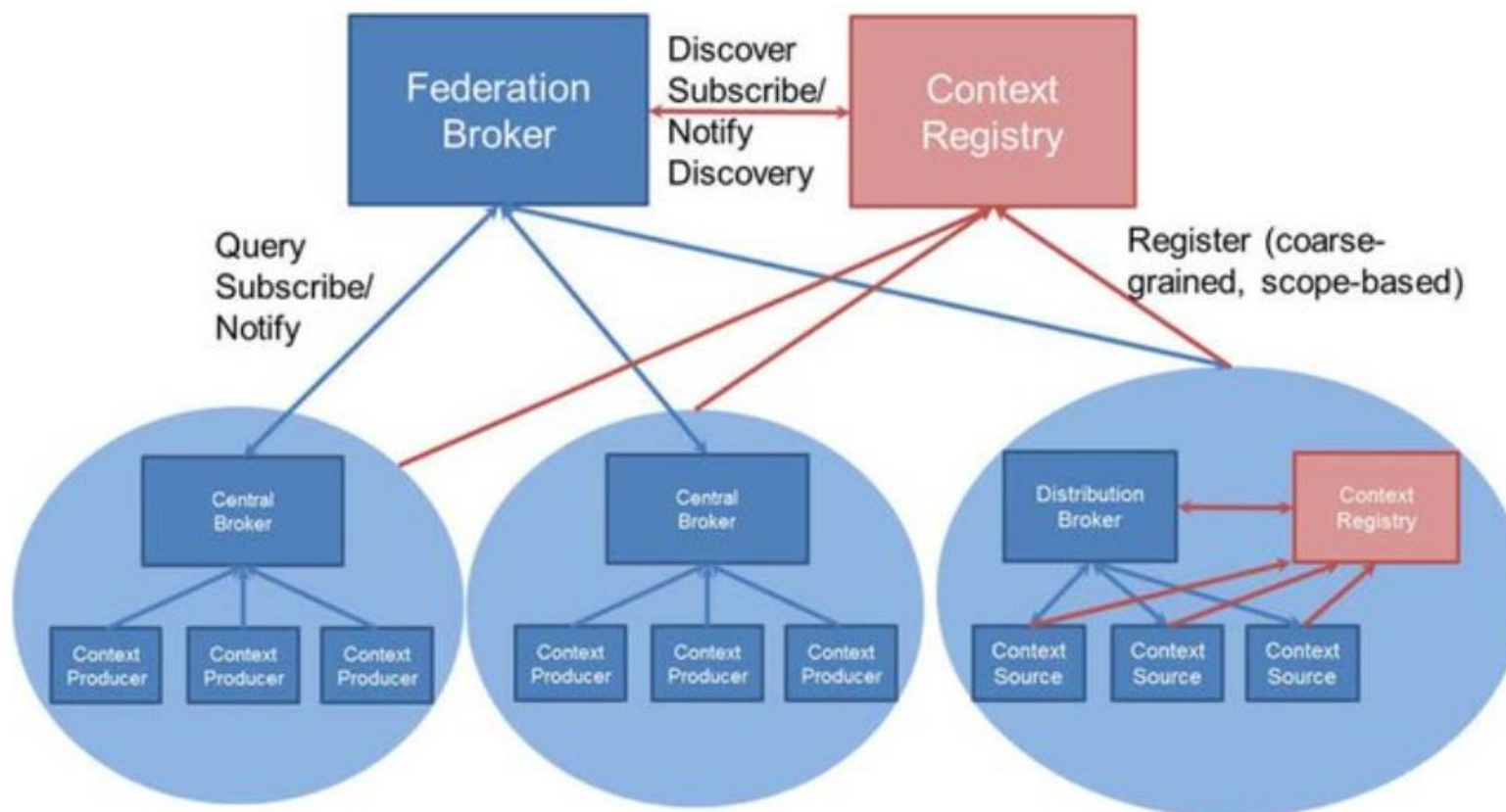


# CoBrA Context Broker Architecture





# Architettura Distribuita





# Core Component



ScorpioBroker



OrionBroker



SirioBroker

# Progetto

- Introduzione
- Use Case
- Model Domain
- Architettura
- Smart Data Models
- Codici

# Obiettivo sperimentale

---

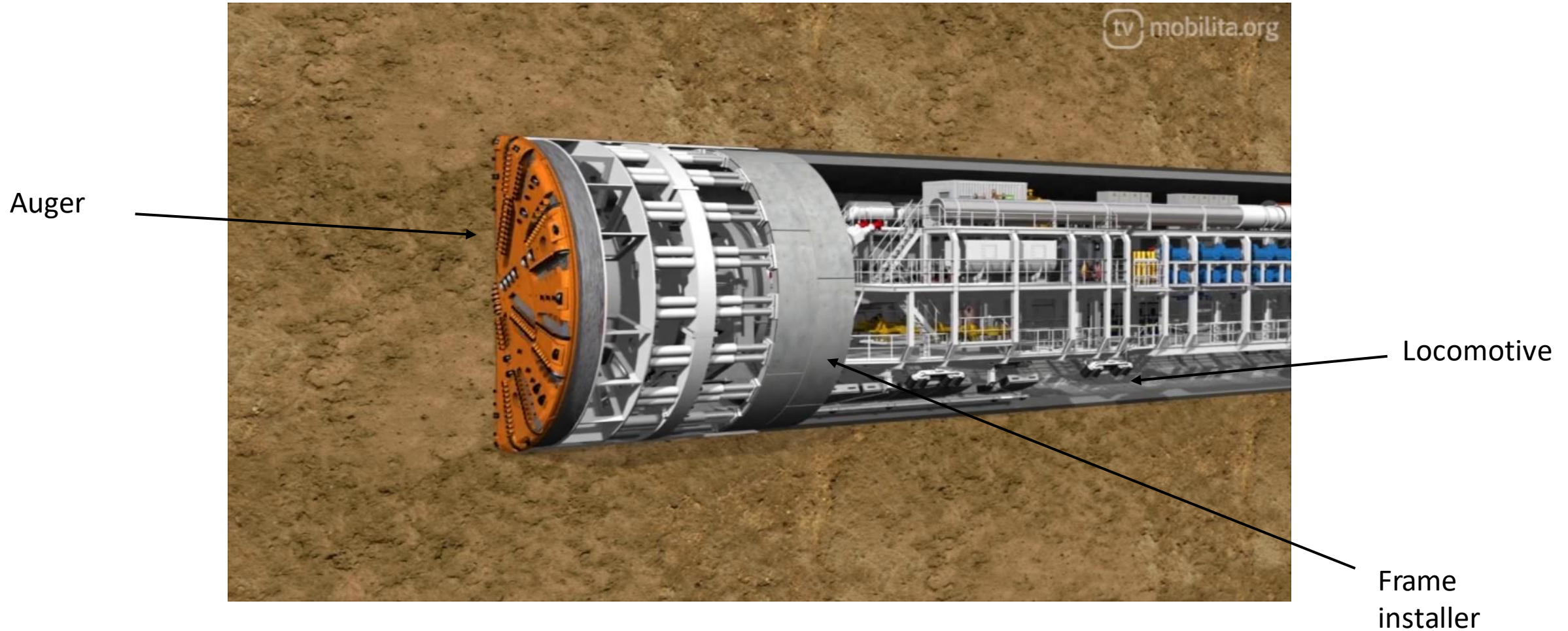
Dimostrare le capacità di **interoperabilità** del framework attraverso lo sviluppo di un prototipo in ambito di **Smart Manufacturing**



# Interoperabilità

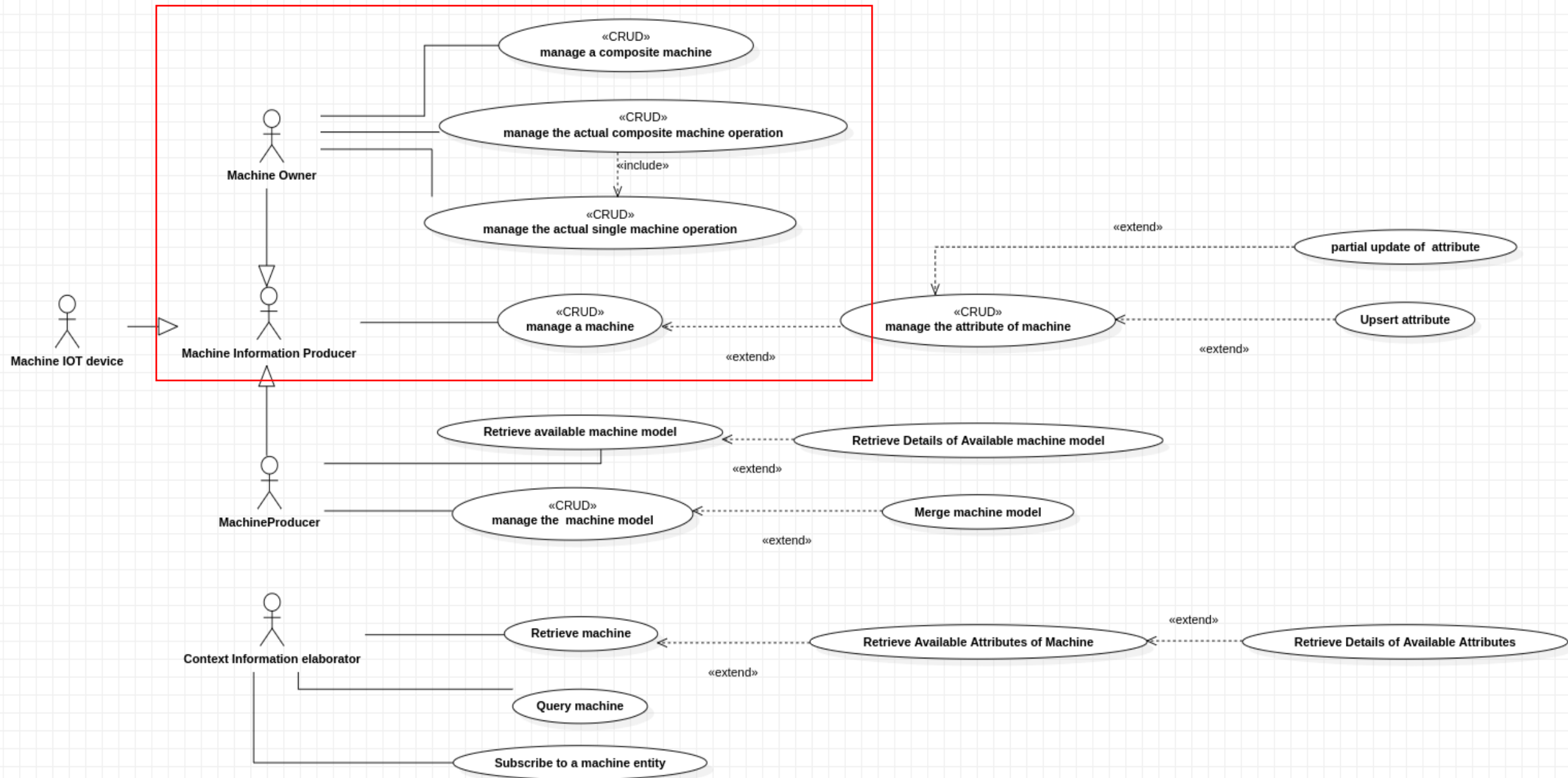
- **Treccani**: Capacità di **due o più sistemi**, reti, mezzi, applicazioni o componenti, di **scambiare informazioni** tra loro e di **essere poi in grado di utilizzarle**.
- **Sogei**: Il termine interoperabilità individua la capacità di un sistema informatico di **cooperare e scambiare informazioni o servizi** con altri sistemi/prodotti in maniera più o meno **completa e priva di errori**.  
Una operazione che risponde a criteri di affidabilità e ottimizzazione delle risorse.

# Ambiente Operativo



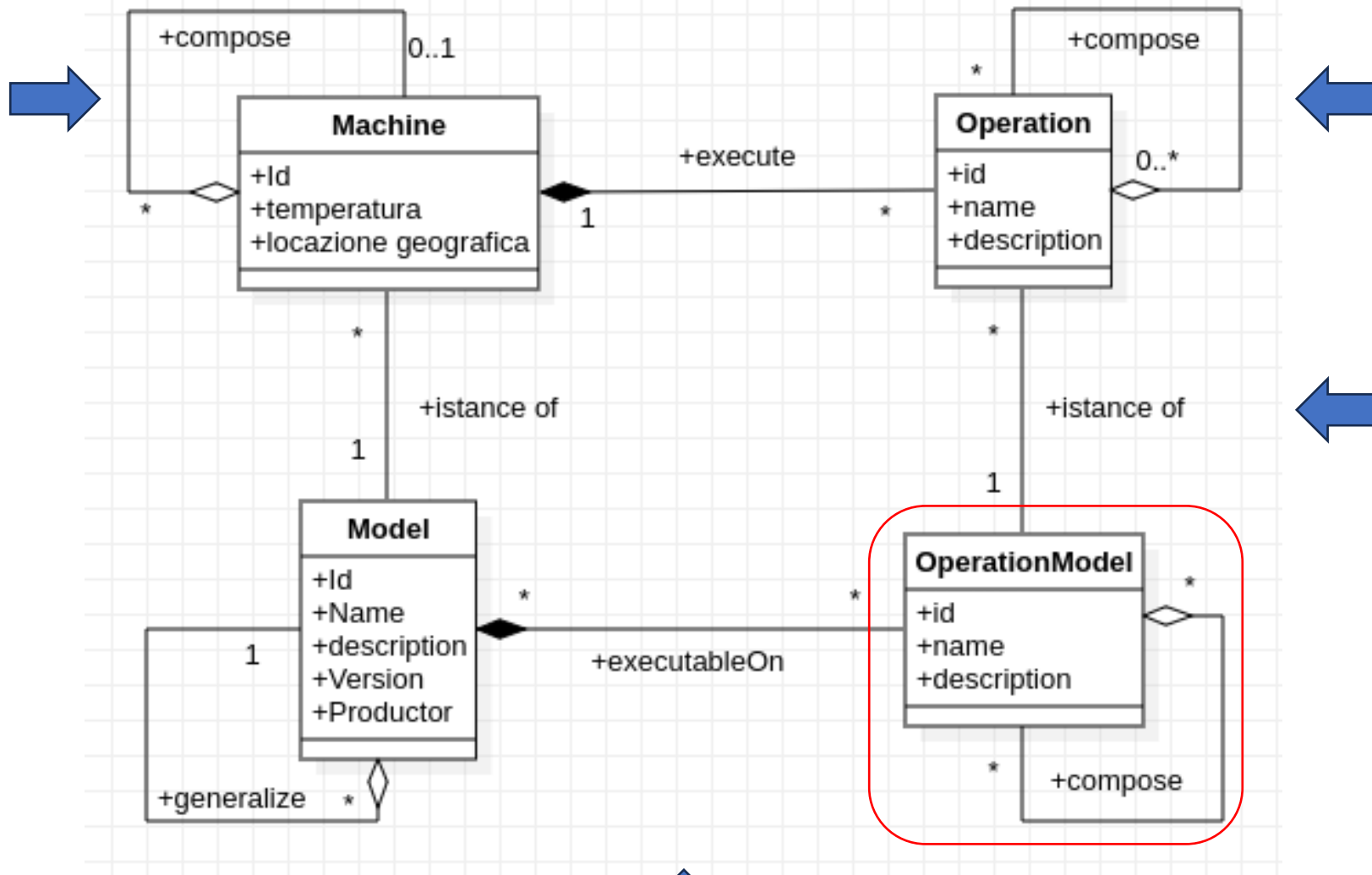




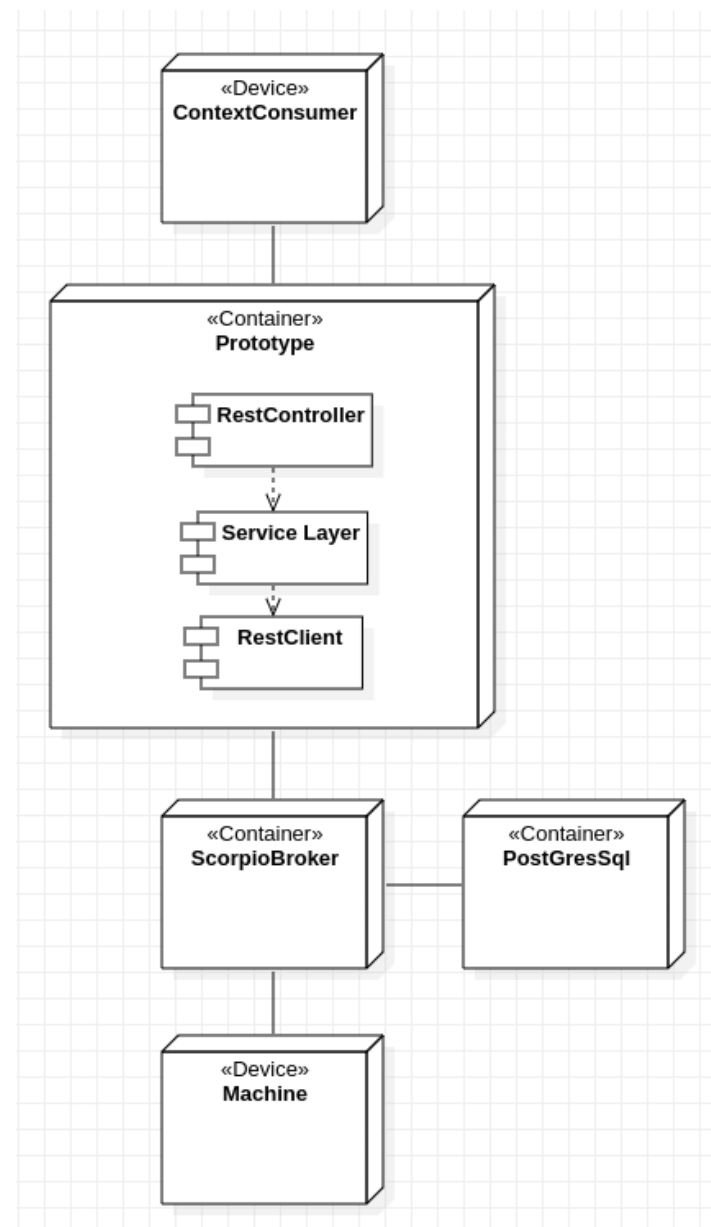




# Modello di dominio concettuale

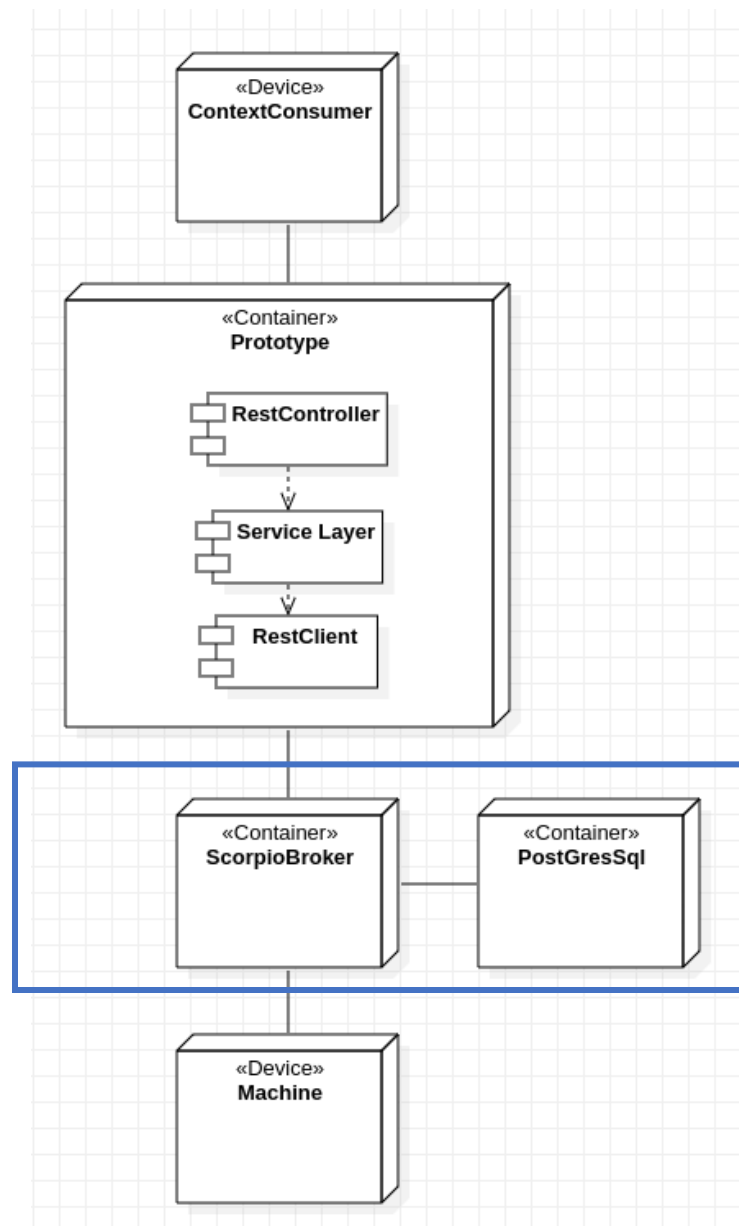


# Architettura

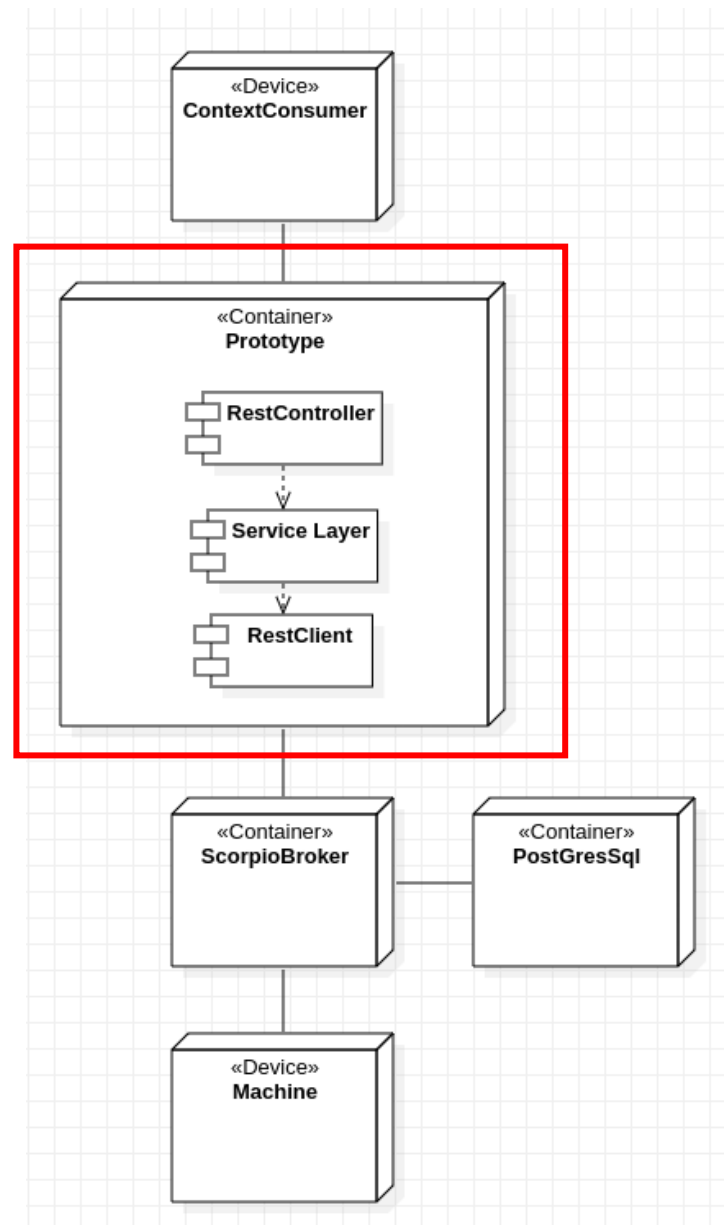


**Maven™**

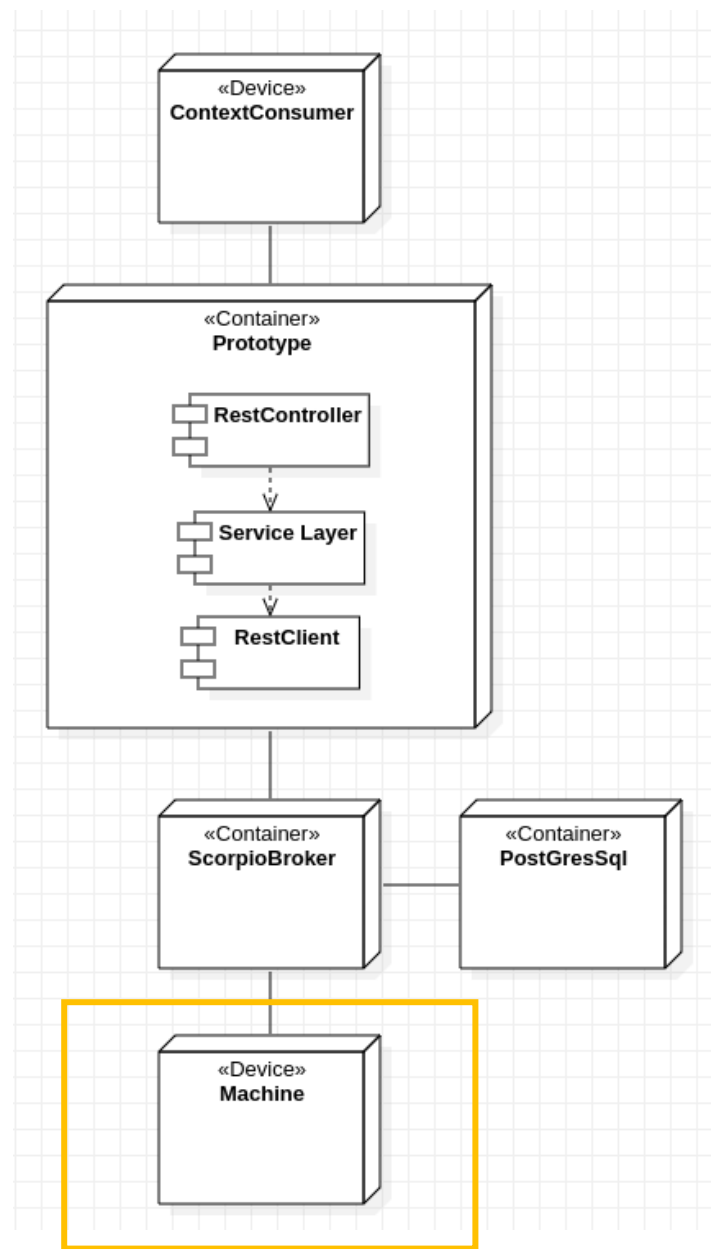
# ScorpioBroker



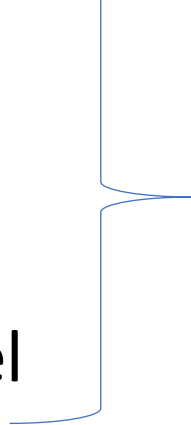
# Prototipo



# Simulatore Macchinari



# Context e Io Smart Data Model

- MachineManufacturing
  - MachineManufacturingModel
  - MachineManufacturingOperation
  - MachineManufacturingOperationModel
  - Subscription e Notification
- 
- Auger
  - Composite



# Machine Manufacturing




# Auger Machine

```
1 {
2   "@context": [
3     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld",
4   ],
5   "assetIdentifier": "ID12345",
6   "countryOfManufacture": "UK",
7   "dataProvider": "https://provider.example.com",
8   "description": "Auger - Trivella",
9   "firmwareVersion": "A.10",
10  "firstUsedAt": "2017-05-04T10:18:16Z",
11  "hardwareVersion": "2.1",
12  "id": "urn:ngsi-ld:Machine:9166c528-9c98-4579-a5d3-8068aea5d7k0",
13  "installedAt": "2017-05-04T10:18:16Z",
14  "location": {
15    "coordinates": [
16      -104.99404,
17      39.75621
18    ],
19    "type": "Point"
20  },
21  "machineModel": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd65k9",
22  "manufacturedAt": "2017-05-04T10:18:16Z",
23  "power": 4.4,
24  "rotationalSpeed": 100,
25  "supplierName": "ACME NorthEast Inc.",
26  "type": "ManufacturingMachine",
27  "voltage": 220
28 }
```

# Composite Machine

```
1 {
2   "@context": [
3     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"
4   ],
5   "assetIdentifier": "ID12350",
6   "description": "Composite Auger",
7   "firstUsedAt": "2017-05-04T10:18:16Z",
8   "id": "urn:ngsi-ld:Machine:9166c528-9c98-4579-a5d3-8068aea5d9b0",
9   "installedAt": "2017-05-04T10:18:16Z",
10  "location": {
11    "coordinates": [
12      -104.99404,
13      39.75621
14    ],
15    "type": "Point"
16  },
17  "machineModel": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd6490",
18  "manufacturedAt": "2017-05-04T10:18:16Z",
19  "power": 4.4,
20  "advancementSpeed": 10,
21  "supplierName": "ACME NorthEast Inc.",
22  "machineComponent": {
23    "value": {
24      "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd65k9": "urn:ngsi-ld:Machine:9166c528-9c98-4579-a5d3-8068aea5d7k0",
25      "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd64t9": "urn:ngsi-ld:Machine:9166c528-9c98-4579-a5d3-8068aea5d6c0",
26      "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd69c3": "urn:ngsi-ld:Machine:9166c528-9c98-4579-a5d3-8068aea5dq90"
27    },
28    "type": "Property"
29  },
30  "type": "ManufacturingMachine",
31  "voltage": 220
32 }
```



# Machine Manufacturing Model

# Composite Machine

```
1 {
2   "@context": [
3     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"
4   ],
5   "id": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd6490",
6   "type": "ManufacturingMachineModel",
7   "name": "Auger Composite Machine",
8   "description": "A composite machine",
9   "brandName": "Transformer",
10  "version": "v1",
11  "root": "false",
12  "machineModelParent": "urn:ngsi-ld:MachineModel:4146335f-839f-4ff9-a575-6b4e6232b734",
13  "processDescription": "A composite machine that drill to construct tunnel",
14  "componentMachineModel": {
15    "type": "Relationship",
16    "object":
17      [
18        "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd65k9",
19        "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd64t9",
20        "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd69c3"
21      ]
22  },
23  "operationModel": {
24    "type": "Property",
25    "value": [
26      { "Forward": "urn:ngsi-ld:MachineOperationModel:27577638-bd8a-4732-b418-fc8b949a0t90" }
27    ]
28  }
29 }
```

# Auger Machine

```
1 {
2   "@context": [
3     {"machineModelChildren": "https://smartdatamodels.org/dataModel.ManufacturingMachine/machineModelChildren"},
4     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"
5   ],
6   "id": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd65k9",
7   "type": "ManufacturingMachineModel",
8   "name": "Auger Machine",
9   "description": "Machine to digger",
10  "brandName": "QuickT",
11  "version": "v1",
12  "root": "false",
13  "machineModelParent": "urn:ngsi-ld:MachineModel:4146335f-839f-4ff9-a575-6b4e6232b734",
14  "machineModelChildren":
15    "urn:ngsi-ld:MachineModel:a74fcf24-58fa-11e8-ae3e-df1abd78f83f",
16
17  "processDescription": "Industrial Drilling",
18  "operationModel": {
19    "type": "Property",
20    "value":
21      {"Drill": "urn:ngsi-ld:MachineOperationModel:27577638-bd8a-4732-b418-fc8b949a0t90"}
22  }
23 }
```



# Machine Manufacturing Operation Model

---

# Composite Machine

```
1 {
2   "@context": [
3     {
4       "operationComposite": "urn:operationComposite.property"
5     },
6     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"
7   ],
8   "id": "urn:ngsi-ld:MachineOperationModel:27577638-bd8a-4732-b418-fc8b949a0t90",
9   "type": "ManufacturingMachineOperationModel",
10  "machineModel": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd6490",
11  "operationType": "process",
12  "description": "Forward",
13  "operation": "Forward",
14  "operationComposite": {
15    "type": "Property",
16    "value":
17      {
18        "Forward": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd69c3",
19        "Install": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd64t9",
20        "Drill": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd65k9"
21      }
22  }
23 }
24 }
```

# Auger Machine

```
1 {  
2   "@context": [  
3     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"  
4   ],  
5   "id": "urn:ngsi-ld:MachineOperationModel:27577638-bd8a-4732-b418-fc8b949a0c90",  
6   "type": "ManufacturingMachineOperationModel",  
7   "machineModel": "urn:ngsi-ld:MachineModel:00b42701-43e1-482d-aa7a-e2956cfd65k9",  
8   "operationType": "process",  
9   "description": "Drill",  
10  "operation": "Drill"  
11 }
```



# Machine Manufacturing Operation

---

# Composite Machine

```
1 {  
2   "@context": [  
3     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"  
4   ],  
5   "id": "urn:ngsi-ld:MachineOperation:27577638-bd8a-4732-b418-fc8b949a0t90",  
6   "type": "ManufacturingMachineOperation",  
7   "machine": "urn:ngsi-ld:Machine:9166c528-9c98-4579-a5d3-8068aea5d6c0",  
8   "operationType": "process",  
9   "description": "Forward",  
10  "result": "ok",  
11  "plannedStartAt": "2016-08-22T10:18:16Z",  
12  "plannedEndAt": "2016-08-28T10:18:16Z",  
13  "status": "finished",  
14  "startedAt": "2016-08-22T10:18:16Z",  
15  "endedAt": "2016-08-28T10:18:16Z",  
16  "commandSequence": "Forward",  
17  "operationComposite": {  
18    "type": "Relationship",  
19    "object": [  
20      "urn:ngsi-ld:MachineOperation:27577638-bd8a-4732-b418-fc8b949a0b0f",  
21      "urn:ngsi-ld:MachineOperation:27577638-bd8a-4732-b418-fc8b949a0c7c",  
22      "urn:ngsi-ld:MachineOperation:27577638-bd8a-4732-b418-fc8b949a0t90"  
23    ]  
24  }  
25 }
```

# Auger Machine

```
1 {
2   "@context": [
3     "https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld"
4   ],
5   "id": "urn:ngsi-ld:MachineOperation:27577638-bd8a-4732-b418-fc8b949a0b0f",
6   "type": "ManufacturingMachineOperation",
7   "machine": "urn:ngsi-ld:Machine:9166c528-9c98-4579-a5d3-8068aea5d7k0",
8   "operationType": "process",
9   "description": "Drill",
10  "result": "ok",
11  "plannedStartAt": "2016-08-22T10:18:16Z",
12  "plannedEndAt": "2016-08-28T10:18:16Z",
13  "status": "finished",
14  "startedAt": "2016-08-22T10:18:16Z",
15  "endedAt": "2016-08-28T10:18:16Z",
16  "commandSequence": {
17    "type": "property",
18    "object": [
19      "Drill",
20      "Drill",
21      "Continue_to_drill"
22    ]
23  }
24 }
```



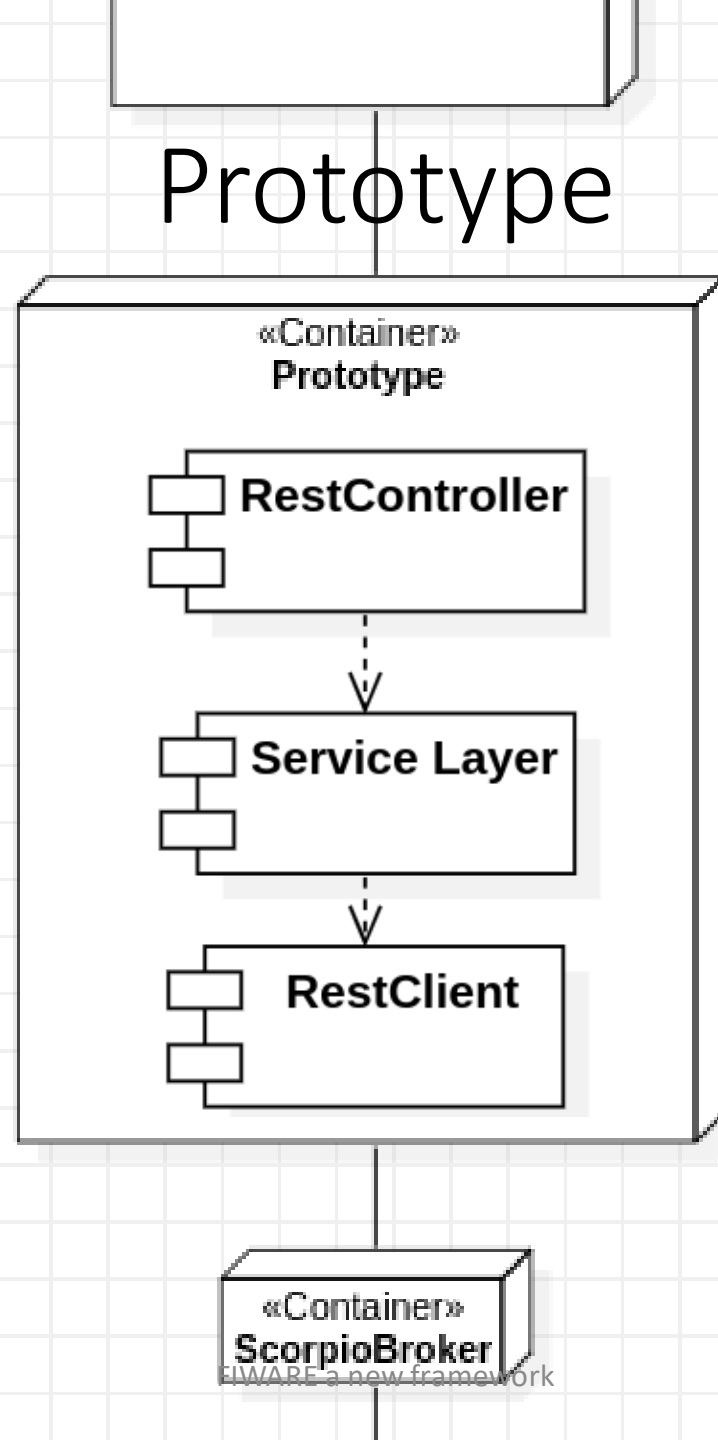
# Subscription and Notification

## Subscription

```
{
  "id": "urn:subscription:1",
  "type": "Subscription",
  "entities": [{
    "type": "ManufacturingMachineOperation"
  }],
  "notification": {
    "endpoint": {
      "uri": "http://172.17.0.1:8090/notification",
      "accept": "application/json"
    }
  },
  "@context": ["https://pastebin.com/raw/Mgxv2ykn"]
}
```

## Notification

```
{
  "id" : "notification:-5808910960552836161",
  "type" : "Notification",
  "subscriptionId" : "urn:subscription:1",
  "notifiedAt" : "2023-10-25T08:02:02.653000Z",
  "data" :
  [ {
    "id" : "urn:ngsi-Id:MachineOperation:317b26cd-ccdc-45cc-9002-5ee426316763",
    "type" : "ManufacturingMachineOperation",
    "description" : {
      "type" : "Property",
      "value" : "Drill"
    },
    "machine" : {
      "type" : "Property",
      "value" : "urn:ngsi-Id:Machine:9166c528-9c98-4579-a5d3-8068aea5d7k0"
    }
  }, { ... } ]
}
```



Prototype

«Container»  
**Prototype**

**RestController**

**Service Layer**

**RestClient**

«Container»  
**ScorpioBroker**

FIWARE a new framework



[POST] /machine/{id}/operation/{action}

```
}

@POST
public Response postOneSingleMachine(@PathParam("idMachine") String uri, Object> json) throws URISyntaxException {
    return machineService.saveMachine(json);
}

@POST
@Path("/{idMachine}/operation/{action}")

public Response postOperationOnAMachine(@PathParam("idMachine") String uri,
    @PathParam("action") String action, HashMap<String, Object> details) throws Exception
    if (details == null)
        details = new HashMap<>();
    return machineService.executeOperation(uri, action, details);
}

@DELETE
@Path("/{idMachine}")
http://localhost:8080/machine/{idMachine}
public Response deleteMachine(@PathParam("idMachine") String uri) {
    return machineService.deleteEntity(uri);
}
```

# RestClient – Configurazione delle richieste HTTP

```
@ClientHeaderParam(name = "Link", value = "${rest-client.linkheader}")  
rest-client.linkheader=<https://raw.githubusercontent.com/smart-data-models/dataModel.ManufacturingMachine/master/context.jsonld>;  
rel="http://www.w3.org/ns/json-ld#context";  
type="application/ld+json"
```

```
@ClientHeaderParam(name = "Content-Type", value = "${rest-client.contenttype}")  
rest-client.contenttype=application/ld+json
```



# Modalità di invio delle informazioni

- Content: application/json
- Link header: required
- @Context section: denied
- POST e GET

- Content: application/ld+json
- Link header: denied
- @Context section: required
- POST

E' possibile anche l'uso dell'applicativo **senza** la sezione **@Context**

```

23 @Path("/")
24 @ApplicationScoped
25 @RegisterRestClient
26 //baseUri ="http://localhost:9090/ngsi-ld/v1"
27 @Consumes(MediaType.APPLICATION_JSON)
28 @Produces(MediaType.APPLICATION_JSON)
29 @ClientHeaderParam(name = "Link", value = "${rest-client.linkheader}")
30 public interface ClientRest {
27
28     @GET
29     @Path("/entities/")
30     public List<Map<String, Object>> findAll(@QueryParam("type") String type);
27
28     @POST
29     @Path("/entities/")
30     public Response postEntity(HashMap<String, Object> entity);
28
29     @GET
30     @Path("/entities/{id}")
27     public HashMap<String, Object> findEntity(@PathParam("id") String id);
28
29     @GET
30     @Path("/entities/{id}")
27     public HashMap<String, Object> findAttributeOfEntity(@PathParam("id") String id,
28         @QueryParam("attrs") String attribute);
29
30
27     @GET
28     @Path("/entities/")
29
30     public List<HashMap<String, Object>> findEntityByFilter(@RestQuery Map<String, String> map, @QueryParam("q")
27
28     @POST

```

# Service Class

```
public List<Map<String, Object>> getAllMachine() {  
    return restClient.findAll("ManufacturingMachine");  
}  
  
public Response saveMachine(HashMap<String, Object> entity) {  
    if (entity.get("type").equals("ManufacturingMachine"))  
        return restClient.postEntity(entity);  
    else  
        return Response.status(422, "It is not a ManufacturingMachine").build();  
}  
  
public Response deleteEntity(String id) {  
    HashMap<String, Object> entity = restClient.findAttributeOfEntity(id, "type");  
    if (entity.get("type").toString().compareTo("ManufacturingMachine") == 0) {  
        return restClient.delete(id);  
    }  
    return Response.status(400, "It is not a ManufacturingMachine").build();  
}
```

# Service Class

```
public Response executeOperation(String machineUri, String action, HashMap<String, Object> details)
    throws Exception {
    HashMap<String, Object> machine = restClient.findEntity(machineUri);
    if (machine.get("type").toString().compareTo("ManufacturingMachine") != 0) {
        throw (new Exception("Entity retrieved is not a ManufacturingMachine"));
    }
    HashMap<String, Object> operationalModel = restClient.findEntityByFilter(
        Map.of("type", "ManufacturingMachineOperationModel"), createFilter("description", action),
        createFilter("machineModel", getMap(machine, "machineModel").get("value").toString())).get(0);
    if (operationalModel.containsKey("operationComposite")) {
        Map<String, Object> elements = getValueOfPropertyField(operationalModel, "operationComposite");
        Map<String, Object> machineComponent = getValueOfPropertyField(machine, "machineComponent");
        List<HashMap<String, Object>> operationList = elements.entrySet().stream()
            .map((Entry<String, Object> x) -> {
                return createOperation(machineComponent.get(x.getValue()).toString(), x.getKey(), details);
            }).collect(Collectors.toList());
        operationList.add(
            addListOfId(createOperation(machineUri, action, details), operationList, "operationComposite"));
        return restClient.postEntity(operationList);
    } else {
        HashMap<String, Object> operation = createOperation(machineUri, action, details);
        return restClient.postEntity(operation);
    }
}
```

# Context Broker (ScorpioBroker)

## Pro

- Molto versatile
- Centrale nello sviluppo
- Non indispensabile
- Rende più veloce lo sviluppo a livello infrastrutturale

## Contro

- Documentazione ridotta
- Mancanza della versione testing
  - Versione In-Memory
- Mancanza di una libreria ClientRest

# SmartDataModels

## **Pro**

- Buon punto di inizio per lo sviluppo dei modelli
- Strategia degli schemi

## **Contro**

- Non imprescindibile
- Non imposto a livello di NGSI-LD  
l'impiego degli schemi su  
nessuna componente

# Elementi di difficoltà

- Documentazione insufficiente o non aggiornata, per quanto concerne FIWARE e ScorpioBroker (NGSI-LD particolarmente esaustiva).
- La possibilità di avere vari formati per la condivisione delle context information risulta essere error prone nei primi momenti.
- Strategia nella definizione dei data models personalizzati (differenza proprietà e relazione, ed esempi)

# Libertà

- Del Context broker nella gestione delle context information
  - Indipendente dalla tipologia di context information
- Nell'uso della sezione @Context
  - Possibilità di inserire (o no) i dizionari personalizzati per termini specifici.
- Nella robustezza e nello sviluppo del codice
  - Verifica consistenza delle context information
  - Verifica della consistenza delle richieste di context information
  - Scelta nella rappresentazione delle context information nel codice
- Nell'uso del contenuto delle context information da parte degli attori
  - Parzialità nell'uso delle context information



# Conclusioni

- Diverse strategie per permettere l'interoperabilità.
- Un sistema backend (context broker) pre stabilito utile anche al di fuori del ecosistema dei digital twin.
- Gestione della fluidità delle informazioni.
- Una filosofia diversa per la gestione della consistenza dei dati persistiti.



*"That's all Folks!"*