Un Langage Dédié au Domaine du Web Sémantique des Objets Contraints

Fatma-Zohra Hannou
Maxime Lefrançois

PFIA –IoT et IA- 27 Juin 2022

OUTLINE

CoSWoT project

- Context
- Motivation
- Interoperability in IoT
- Research Problem

A DSL for constrained objets embedded applications

- Domain Specific Language
- Methodology
- Analysis
- Design
- Implementation

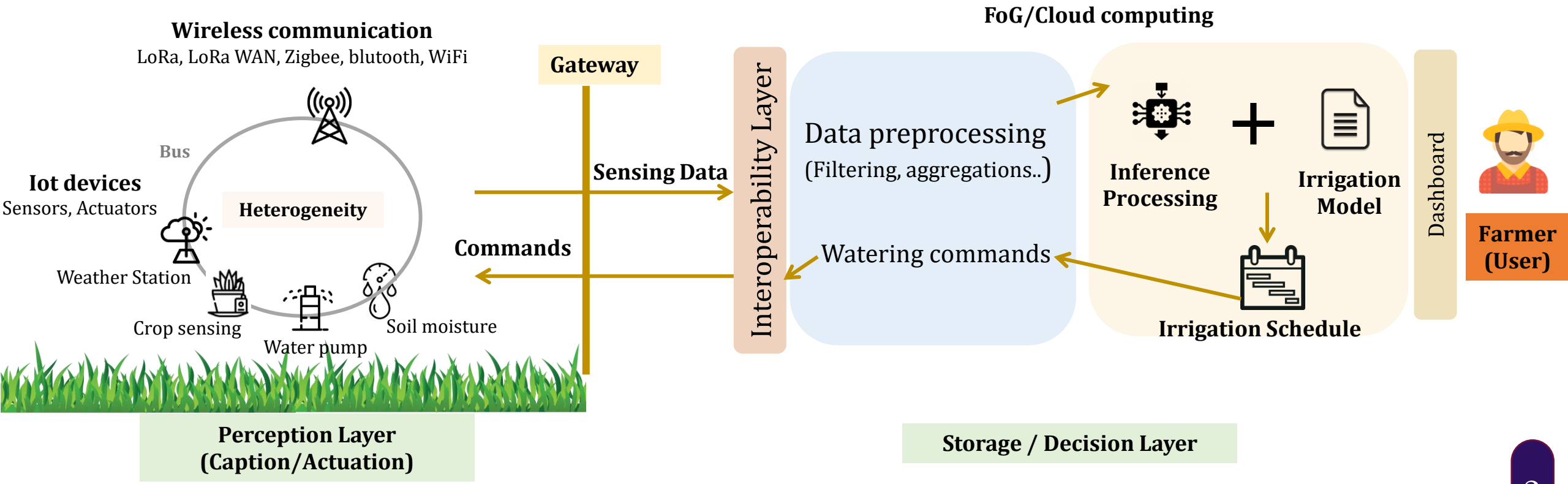
Summary

Constrained Semantic Web of Things Project

Context

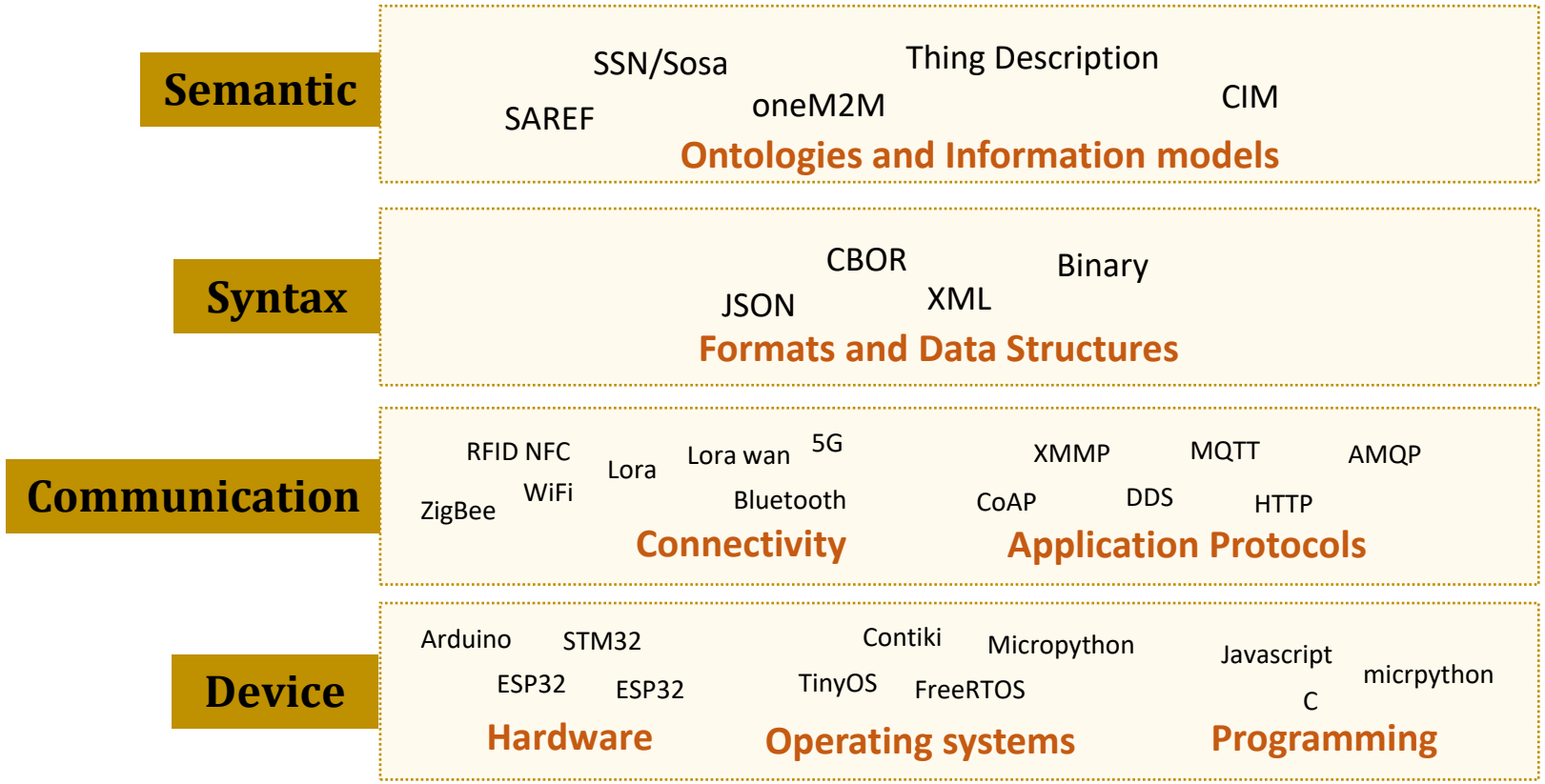
Goal : Build a Web of Things platform to enable the development and execution of intelligent and decentralized applications

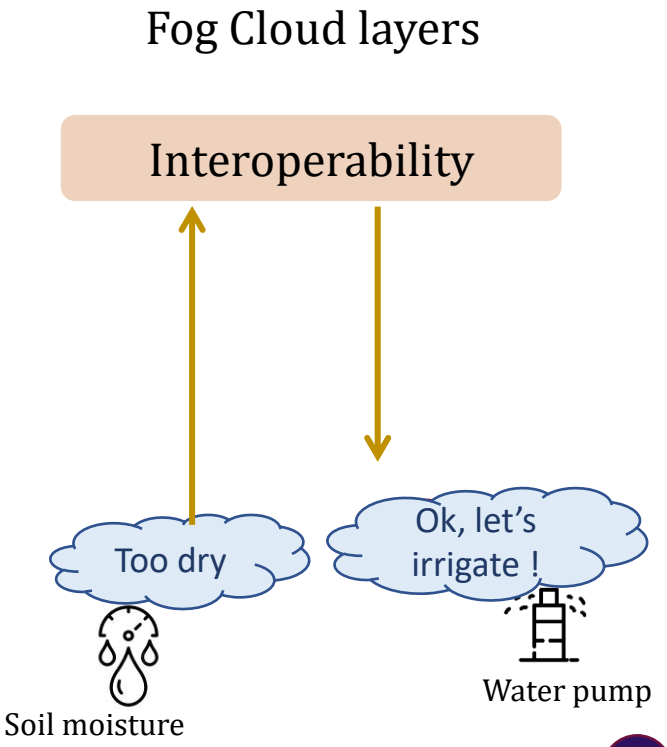
Use cases: Smart building (automatic heating, window opening, energy), Smart agriculture (irrigation, fire/freeze detection..)



Context

Heterogeneity dimensions are diverse: device level, communication protocols, syntax, semantics, cross-platforms and cross-domains.

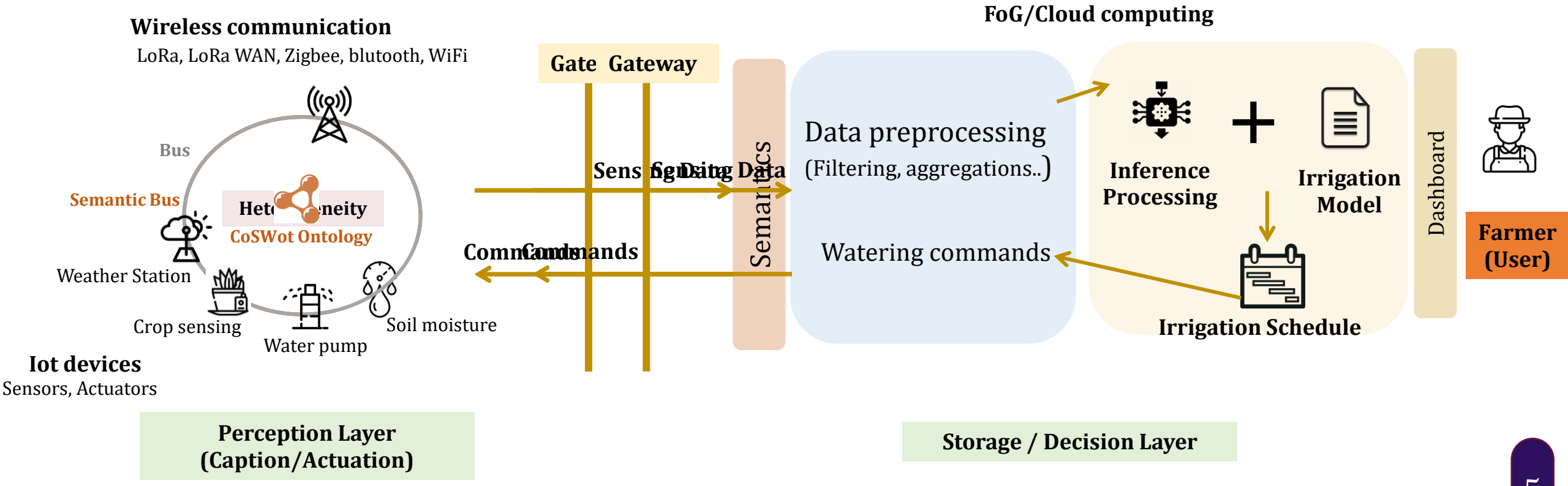




Interoperability

Introducing/pushing semantic interoperability within the perception layer

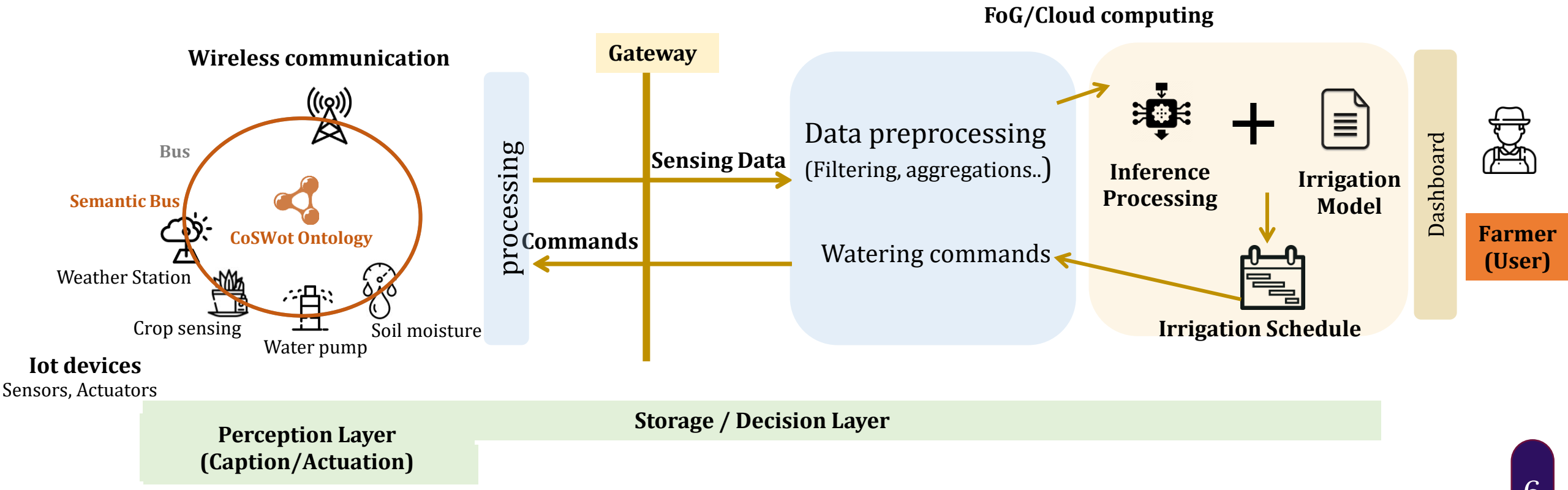
- Shared semantic model: CoSWoT ontology
- Semantic Bus exchange mechanism



Interoperability

Introducing semantic interoperability in the perception layer

- Shared semantic model: CoSWoT ontology
- Semantic Bus exchange mechanism
- Generate semantic data ? Processing ?



Motivation

Achieve semantic interoperability within the perception layer (constrained objects)

Optimize power consumption by reducing data transfer rate

Enhance response time, reducing information transfer latency

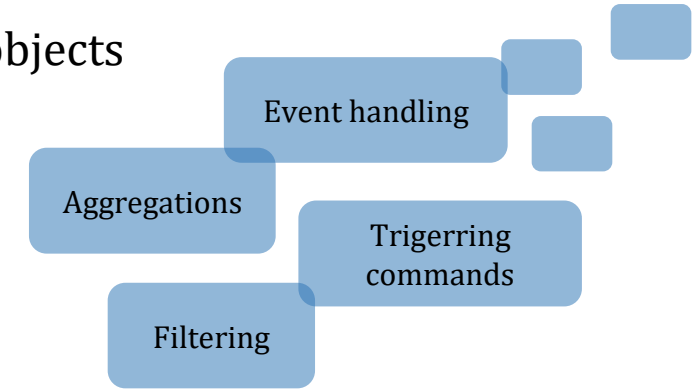
Ensure alternative degraded operating mode in case of communication issues

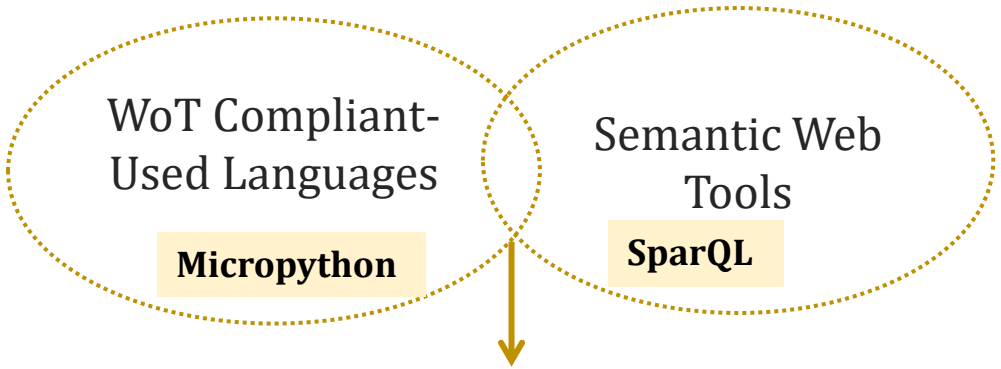
Privacy requirements, partial data disclosure

Research Direction

A programming language to write smart semantic web applications in constrained objects

- Write applications in constrained objects with respect to their resources limitations
- Process/transform RDF entities (triples, graphs..)
- Support for communication protocols (semantic bus)
- With readable and expressive syntax. Easy to use for developpers





New programmin language for constrained objects and a **lightweight** interpreter

CoSWoT
Domain Specific Language

Domain Specific Language

A Domain Specific language is a programming language created to support a particular problem space, related to a specific domain. They are designed to make users effective in the domain tasks.

Unlike General Purpose Languages, DSLs are not required to be computationally universal programming languages (Turing completeness)

SQL

Domain: Relational databases

Access, insert, modify or extract data from relational databases

```
SELECT MAX(TEMP),
MIN(TEMP), AVG(RAIN), ID
FROM STATS GROUP BY ID;
```

HTML

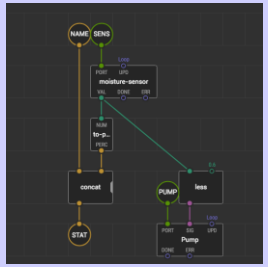
Domain: Web application

Define document structure and content

```
<html>
<head>
<title>My beautiful
page</title>
</head>
<body>
<div id="main">
```

XOD

Domain: visual microcontroller programming platform



Latex

Domain: Document markup

```
\newcommand
\fibonacci
[1]
 \counter=#1 \fone=1
\ftwo=1 \temp=0 \the\fone,
\the\ftwo \fibloop
```

Development process

Sloane patterns present a methodology for DSL construction following 6 phases. The development is not a sequential process, the phases should be applied iteratively

Decision

Review existing approaches addressing the problem space
Comparative study and opportunity definition
(**task automation, extend data structures, AVOPT**)

Analysis

Define the scope of the domain and its vocabulary (**CoSWoT ontology**)
Use case requirements specifications (**program patterns**)

Design

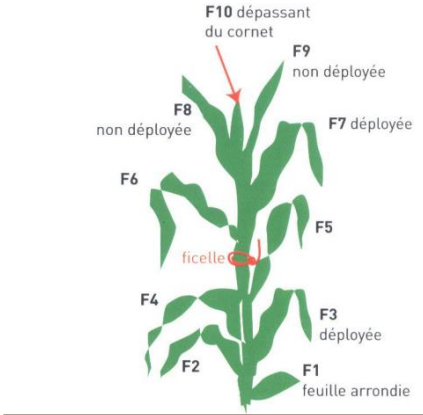
Implementation

Deployment

Maintainance

Smart Irrigation

The irrinov method outputs two decision patterns : start watering cycle / stop watering based on decision tables tailored per crop and soil characteristics.



Data required to apply decision tables:

- **CropGrowth:** A qualitative observed property of the crop
- **Probe30:** observtion, measures the soil pressure indicating the humidity rate at depth 30cm (valid values when reached by 2 out of 3 probes, tolerated difference up to 30cbar)
- **Prob60:** same as probe30, at 60cm depth.
- **RootZoneDailyMoisture** Computed as the aggregation of Probe30Daily and Probe60Daily
- **Rain Quantity** recorded by a wather station

Decision pattern: RainQuantity < 10mm

SoilType: profond
CropeType: maize

EDB: **CropGrowth** (crop, soil, stage), **SoilMoisture** (soil, depth, day, value), **RoundDuration** (soil, ndays)**, RainQuantity** (day, rain)

IDB: **irrigationState** (soil, crop, day, state), **totalSoilMisture** (soil, day, value)

Simple rule

If 6 <irrigationRoundDuration <9 **and** 10 < growthStage <14 **and** Moisture >40 **then** start(irrigation)

If rainQuantity <10mm **then** start/keep (irrigation)

Datalog:

TotalSoilMoisture(soil, day, w) :- SoilMoisture(soil, depth,day, value), w=sum(value): *{SoilMoisture(soil, depth,day, value)}*

irrigationState(soil, crop ,day , **True**) :- RoundDuration (soil, ndays), ndays <9, ndays >5, **TotalSoilMoisture**(soil, day, w), w>40, CropGrowth (crop, 10leafs), **RainQuantity** (day, rain), rain <10

Development process

Sloane patterns present a methodology for DSL construction following 6 phases. The development is not a sequential process, the phases should be applied iteratively

Decision

Review existing approaches addressing the problem space
Comparative study and opportunity definition
(**task automation, extend data structures, AVOPT**)

Analysis

Define the scope of the domain and its vocabulary (**CoSWoT ontology**)
Use case requirements specifications (**program patterns**)

Design

Relation between DSL and existing languages (**extension micropython**)
Formal design approach (Abstract syntax –metamodels-, grammar, semantics)

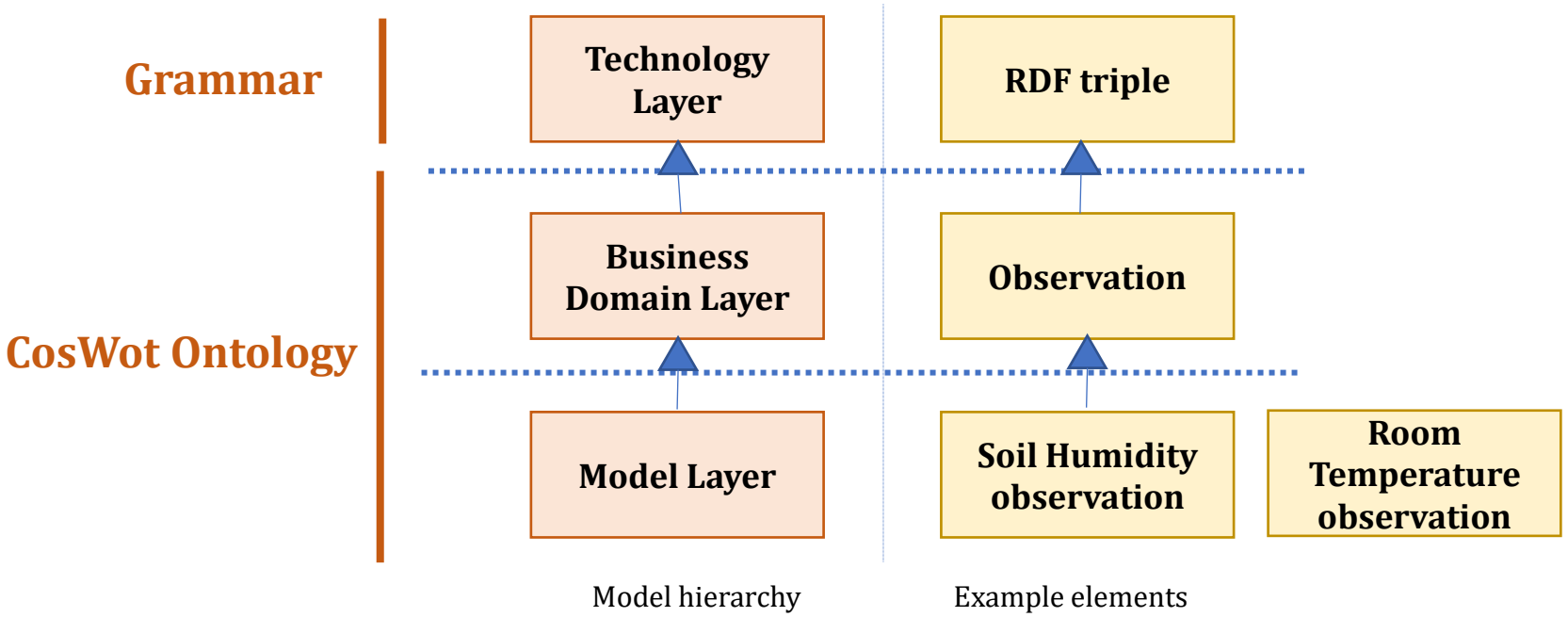
Implementation **Deployment** **Maintainance**

Design

A. Relation between DSL and existing languages (partial use of existing language, specialization of existing language, **extension** of existing language)

B. Informal design approach vs. **Formal** design approach (reg expressions, grammars…)

Abstract syntax (metamodels), concrete syntax, semantics.



Development process

Sloane patterns present a methodology for DSL construction following 6 phases. The development is not a sequential process, the phases should be applied iteratively

Decision

Review existing approaches addressing the problem space
Comparative study and opportunity definition
(**task automation, extend data structures, AVOPT**)

Analysis

Define the scope of the domain and its vocabulary (**CoSWoT ontology**)
Use case requirements specifications (**program patterns**)

Design

Relation between DSL and existing languages (**extension micropython**)
Formal design approach (Abstract syntax –metamodels-, grammar, semantics)

Implementation

First implementations of Linked Data Python

Deployment

Maintainance

Implementation

RDF graph creation,
Triples addings

Pythonic loops,
conditions…

```
@prefix sosa: <http://www.w3.org/ns/sosa/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.

from rdflib import Graph

g = g{
    [] a sosa:Observation ;
      sosa:hasSimpleResult 0 .
    [] a sosa:Observation ;
      sosa:hasSimpleResult 0.25 .
    [] a sosa:Observation ;
      sosa:hasSimpleResult 0.7 .
    [] a sosa:Observation ;
      sosa:hasSimpleResult 0.5 .
    }

for s,p,o in g:
    if p != rdf:type:
        continue
    value = g.value(predicate = sosa:hasSimpleResult , subject=s, any=False)
    print(value)

for s,p,o in g{ <s> <p> 1,2,3}:
    print(o)

if len(g{ <s> <p> <o> , <o2> , <o> }) == 2:
    print("ok")
else:
    print("nok")


l = [ o for s,p,o in g{ <s> <p> 1,2,3}]
print(l)
```

Prefix Declaration
(turtle syntax)

Conclusion

- CosWot project aims to Build a WoT platform to enable the development and execution of intelligent and decentralized applications
- Most interoperability proposals focus on unifying stndards for access, representation and semantics beyond gateways and middlewares. We aim to inject interoperability within constrained objects by defining a lightweigth **semantic domain specific language**.
- **Linked Data python** language analysis and design phase to determine end-users requirements

Thank you !

Questions ?