Technische Universität Berlin

Building a Semantic Swarm of Edge Intelligence

Anh Le-Tuan, Jiangtao Shuai

Open Distributed Sytems

ODS

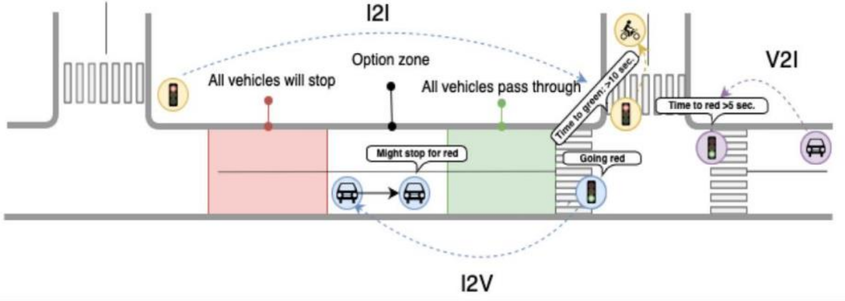
Smartedge Project - Semantic low-code programming tools for edge intelligence

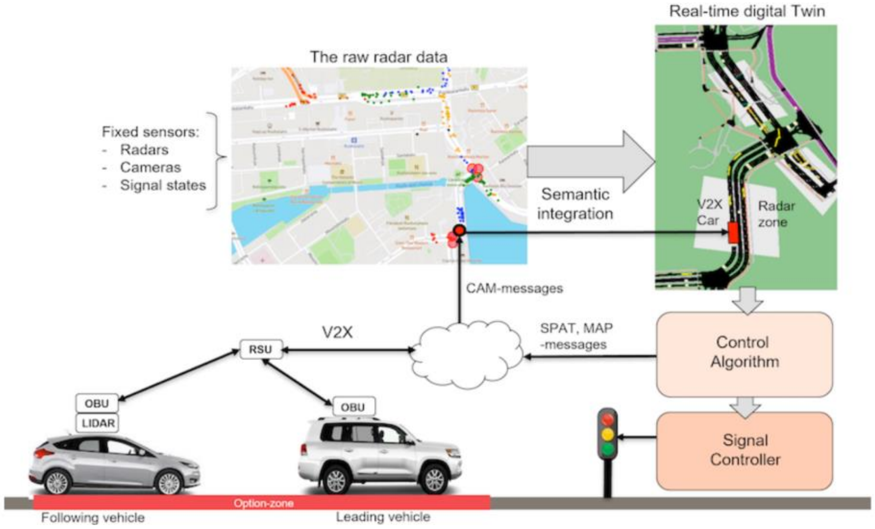
- Enable **dynamic** integration of **decentralized** edge intelligence at runtime
- Seamless, real-time **discoverability** and **composability** of **autonomous** intelligence swarms
- Achieved via **semantic-based interplay** of edge devices with cross-layer low-code toolchains

Use Case – Preventing rear-end collisions by enhancing road intersection safety

Problem:

- **Rear-end collisions** cause ~30% of accidents at traffic light intersections
- High risk occurs in the **option-zone** where vehicles decide to stop or pass when green is ending





Current challenges:

- Existing detection methods are passive, waiting for safe conditions
- Leads to sub-optimal traffic light performance and higher risks

Smartedge approach:

- Uses **swarm intelligence** for proactive traffic management
- Connected vehicles and infrastructure collaborate to prevent high-risk situations
- Enables safer, more efficient traffic flow

Use Cases – Using swarms of mobile robots in smart factories

Problem:

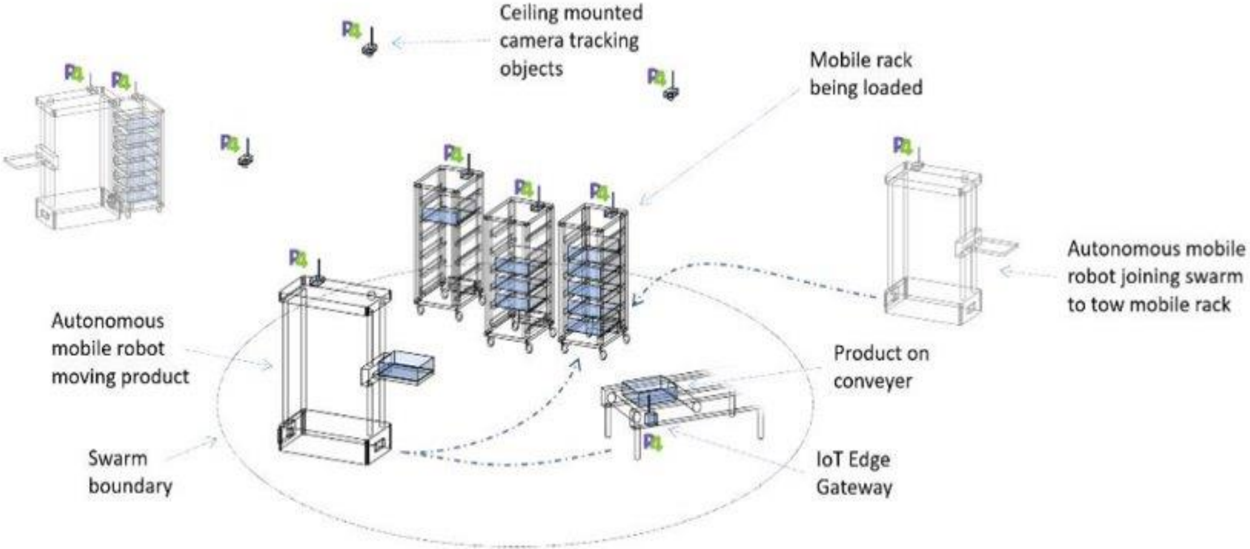
- Traditional factories have **static production lines**, limiting flexibility
- Conventional systems rely on **fixed paths** and **centralized control**, struggling with **dynamic environments**

Current Challenges:

- Factories **need to adapt quickly** to changing layouts and **unpredictable** obstacles
- Robots often **lack a shared understanding** of the environment, **limiting collaboration**
- Centralized control systems hinder scalability and real-time responsiveness
- Difficult to efficiently manage and coordinate multiple autonomous robots and devices

SmartEdge Approach:

- **Develop swarms of autonomous mobile robots** and smart edge devices with semantic understanding of their environment
- Robots and devices share semantic and physical models, enabling collective perception and reasoning
- **Dynamic swarm formation** guided by Smartedge recipes, with autonomous membership management


Ceiling mounted camera tracking objects
Mobile rack being loaded
Autonomous mobile robot joining swarm to tow mobile rack
Autonomous mobile robot moving product
Product on conveyer
Swarm boundary
IoT Edge Gateway

Smartedge Swarm - Collaborative and Autonomous Edge Intelligence

- A swarm is **a collective of edge devices**, vehicles, sensors, and robots working **together** to achieve **shared objectives**. Each **member acts autonomously**.
- Seamless operation across the **edge-cloud continuum** enables smooth integration between cloud and edge layers.
- The **cloud defines** the swarm’s goals and membership but does **not** control real-time actions.
- Real-time **collaboration** and autonomous coordination among swarm members are key, with members dynamically requesting services from each other.

Enabling Smartedge Swarms with **Semantic Technologies**

- Semantic models provide a **shared understanding** of devices, capabilities, and data across the swarm
- Use of standardized vocabularies and ontologies ensures **interoperability** among heterogeneous devices
- Semantic annotations enable **automatic discovery** of devices based on their capabilities
- Facilitate **declarative configuration** and orchestration of swarm applications through semantic descriptions

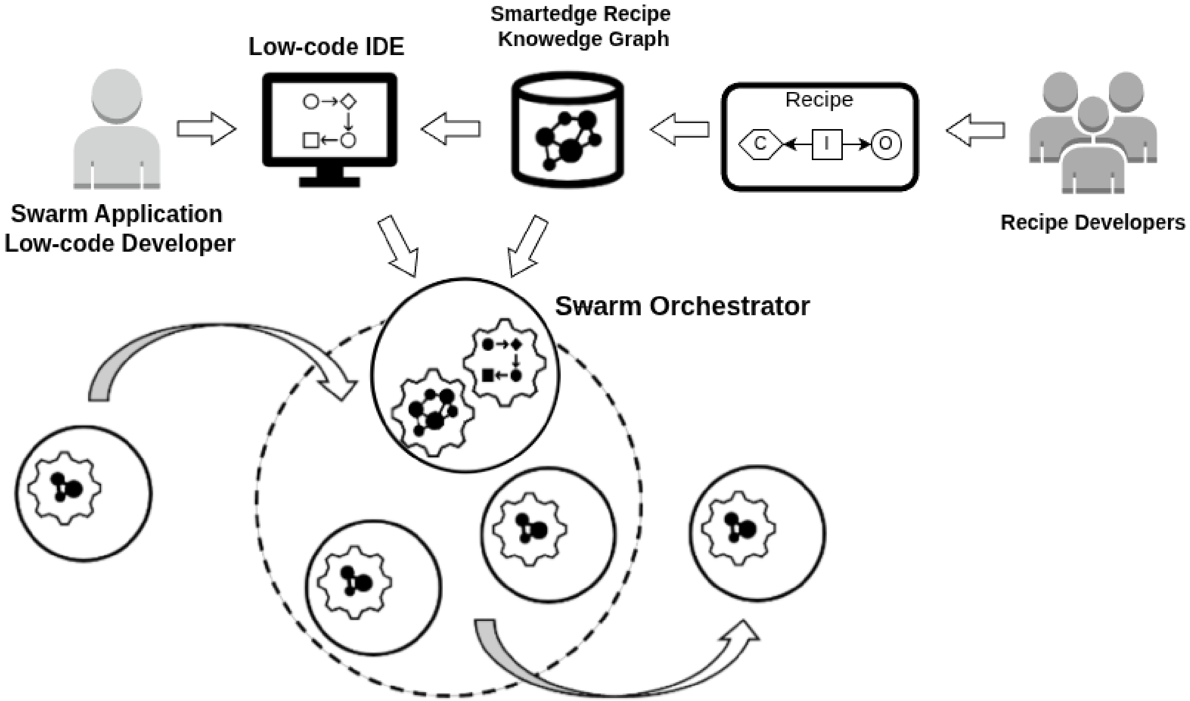
Building **Smartedge Applications**

- SmartEdge enables building distributed edge applications through **modular primitive components**
- **Applications are modeled as semantic workflows** that orchestrate collaboration among swarm members
- **The Recipe Model** defines application components, tasks, required semantic capabilities, and data flows
- **Recipes abstract what needs to be done**, enabling dynamic semantic matching and binding to available swarm members

Recipe Creation and Usage Workflow

- **Recipe Developers** design recipes specifying capabilities, interactions, and operations
- Recipes are stored in the **SmartEdge Recipe Knowledge Graph** for reuse and discovery
- **Swarm Application Low-Code Developers** use low-code IDEs to compose workflows based on stored recipes
- The **Swarm Orchestrator** deploys and manages swarm members executing recipe tasks collaboratively

Enables **modular**, **reusable**, and **dynamically adaptable** edge applications


Low-code IDE
Smartedge Recipe Knowedge Graph
Recipe
C
I
O
Swarm Application Low-code Developer
Recipe Developers
Swarm Orchestrator

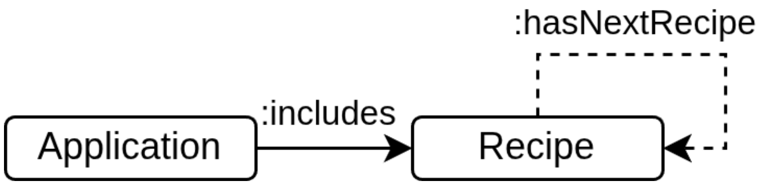
Detailed Recipe Model Structure

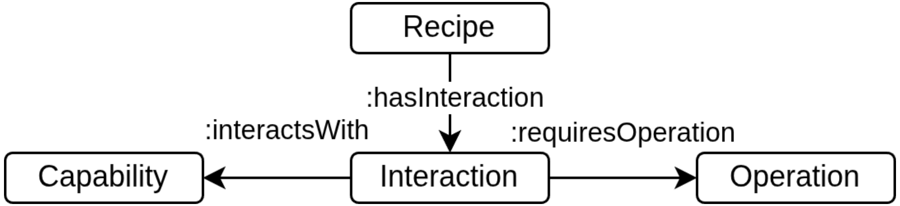
An **Application** includes one or more **Recipes**, sequenced to define the workflow (first diagram)

Each **Recipe** consists of:

- **Capabilities** - functions or features required from swarm members
- **Interactions** - coordination or communication between capabilities
- **Operations** - specific actions required by interactions

Capabilities **interact with** interactions, which in turn **require** operations





Semantic Discovery in SmartEdge

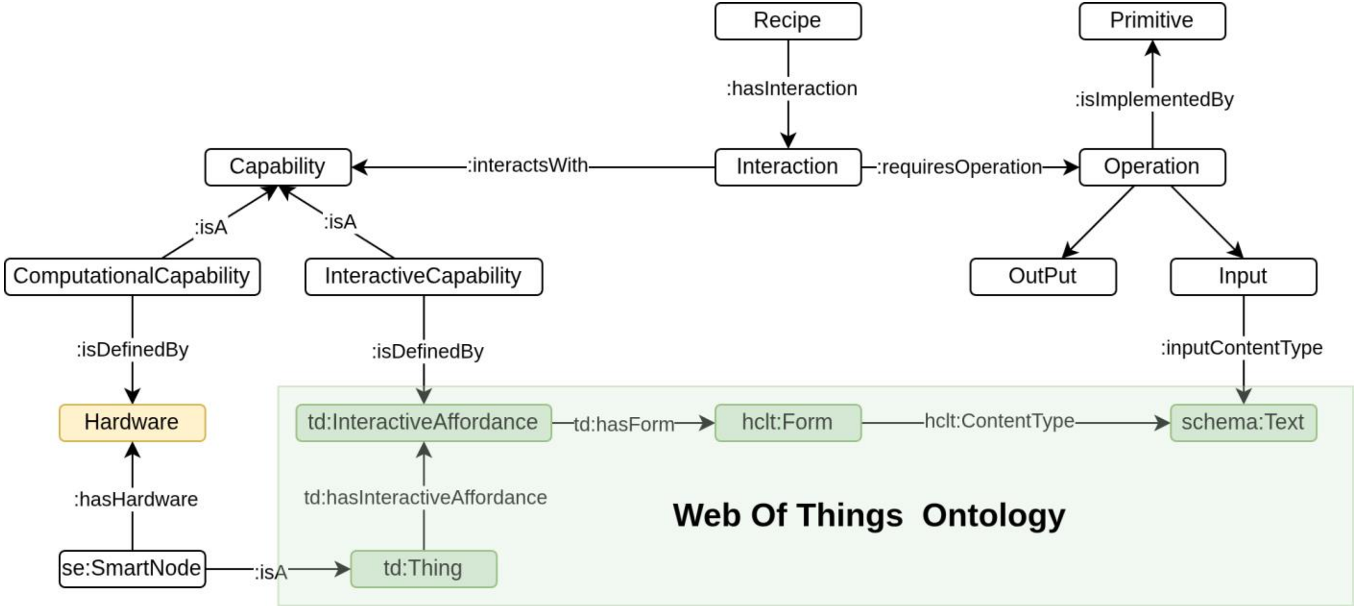
- Semantic descriptions enable automatic discovery of swarm members based on capabilities
- Runtime queries to Knowledge Graph find suitable nodes matching recipe needs
- Supports flexible and adaptive swarm membership based on dynamic conditions

Using Web of Things (WoT) to Describe Capabilities

Capabilities are categorized into:

- **Computational Capabilities** are defined by the hardware specifications of a Smartnode
- **Interactive Capabilities** are described using WoT concepts such as **Interactive Affordance**, **Form**, and **ContentType**

WoT provides standardized semantic vocabularies to precisely model device interfaces and behaviors



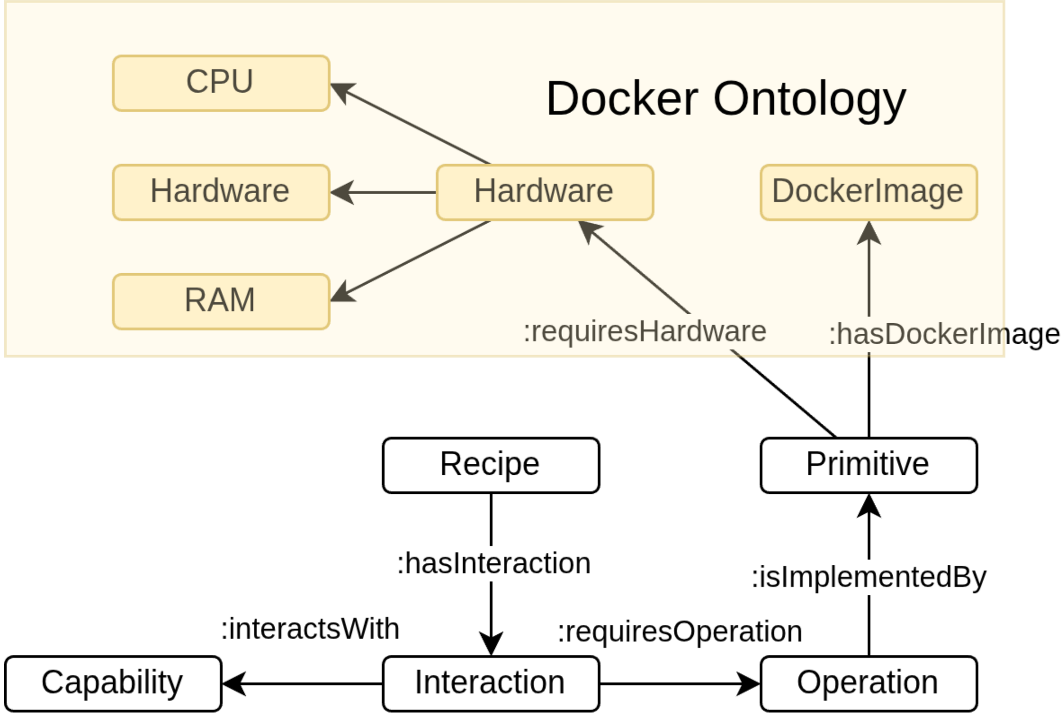
Open Distributed Sytems
ODS

Dynamic Binding of Swarm Members

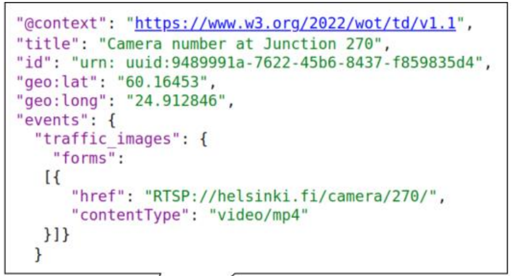
- Tasks dynamically assigned to devices matching semantic capabilities
- Binding adapts in real-time as devices join, leave, or change state
- Enables swarm elasticity and robust operation in dynamic edge environments

Operations and Primitives – Executable Actions

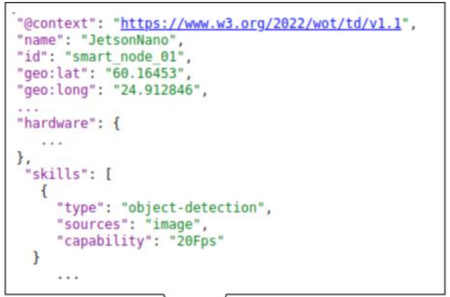
- An **Operation** defines a specific task to be performed in a recipe
- Each Operation is implemented by a **Primitive**, which is the **executable unit** (e.g., software container, script)
- Each Operation is implemented by a Primitive, which is the executable unit (e.g., software container, script)



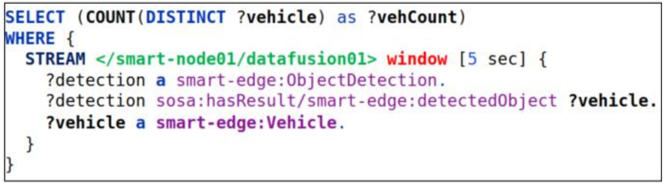
Running Example:



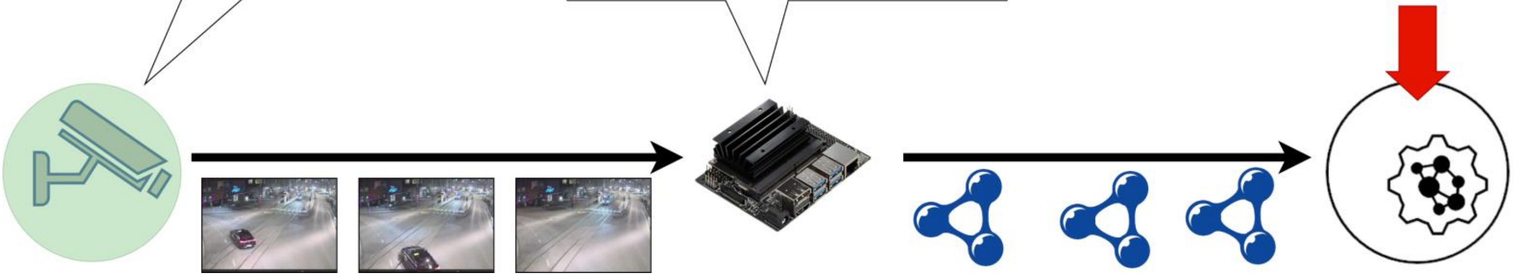
```
"@context": "https://www.w3.org/2022/wot/td/v1.1",
"title": "Camera number at Junction 270",
"id": "urn: uuid:9489991a-7622-45b6-8437-f859835d4",
"geo:lat": "60.16453",
"geo:long": "24.912846",
"events": {
  "traffic_images": {
    "forms":
   [{
      "href": "RTSP://helsinki.fi/camera/270/",
      "contentType": "video/mp4"
   }]}
  }
```



```
"@context": "https://www.w3.org/2022/wot/td/v1.1",
"name": "JetsonNano",
"id": "smart_node_01",
"geo:lat": "60.16453",
"geo:long": "24.912846",
...
"hardware": {
   ...
},
 "skills": [
   {
     "type": "object-detection",
     "sources": "image",
     "capability": "20Fps"
  }
     ...
```



```
SELECT (COUNT(DISTINCT ?vehicle) as ?vehCount)
WHERE {
  STREAM </smart-node01/datafusion01> window [5 sec] {
    ?detection a smart-edge:ObjectDetection.
    ?detection sosa:hasResult/smart-edge:detectedObject ?vehicle.
    ?vehicle a smart-edge:Vehicle.
  }
}
```



Hands - on

- Github: https://github.com/anhlt18vn/AIoT-Handons.git TDD Endpoint: **http://172.21.10.151:8080/rdf4j-workbench/repositories/1**

- pip install -e smartedge

- workspace.py