Open APIs for Open Minds

Letting your chatbot have a friendly talk with the Context Broker

Integrating FIWARE and AI Agents using LLMs

David Campo

Ph.D. in Computational Intelligence | Sr FIWARE Expert & evangelist

FIWARE

Agenda

1. Learning goals
2. Challenges and motivation
3. Essential concepts: LLMs, MCP, Agents
4. Proposed architecture and implementation
5. DEMO: Chatting with the CB
6. Thoughts on improvements and future work
7. Summary & QA

Agenda

1. **Learning goals**
2. Challenges and motivation
3. Essential concepts: LLMs, MCP, Agents
4. Proposed architecture and implementation
5. DEMO: Chatting with the CB
6. Thoughts on improvements and future work
7. Summary & QA

Learning Goals

- **Understand the importance of lowering tech barriers**
- **What a LLM is?**
  - Why do we need them?
  - What is RAG and fine-tuning? How do they work with NGSI?
- **Agents: MCP to the rescue**
  - Why do we need them?
  - How MCP protocol works
- **Wrapping everything up**

Agenda

1. Learning goals
2. **Challenges and motivation**
3. Essential concepts: LLMs, MCP, Agents
4. Proposed architecture and implementation
5. DEMO: Chatting with the CB
6. Thoughts on improvements and future work
7. Summary & QA

FIWARE

What if…

- We could interact with a FIWARE app like:

  Which building is closest to the current location of 'Buttercup'?

- Instead of:

```
curl -G -iX GET
'http://localhost:1026/ngsi-ld/v1/entities/?type=Building&geometry=Point
&coordinates=%5B13.261,52.257%5D&georel=near%3BmaxDistance=
=1000&limit=1&offset=3' -H 'Accept: application/json' -H 'Link:
<http://context/user-context.jsonld>;
rel=http://www.w3.org/ns/json-ld#context; type=application/ld+json'
```

Which animals are fed exclusively with Grass and are in an active reproductive state?

Notify me if there is any animal with abnormal heart rate

How many female adult cows in heat are there in the system?

FIWARE

What if…

- And having this answer:

Based on the data from the Context Broker, Buttercup is currently located at the Big Red Barn (urn:ngsi-ld:Building:002)
The closest buildings to Buttercup's current location are:

1. **Big Red Barn** (urn:ngsi-ld:Building:002) - 0 km away (this is Buttercup's current location)
2. **Triton Tower** (urn:ngsi-ld:Building) - 0.68 km away

- Instead of this one:

[{"id":"urn:ngsi-ld:Building:farm001","type":"Building","category":{"type":"VocabProperty","vocab":"farm"},"address":{"type":"Property","value":{"streetAddress":"Großer Stern 1","addressRegion":"Berlin","addressLocality":"Tiergarten","postalCode":"10557"},"verified":{"type":"Property","value":true}},"location":{"type":"GeoProperty","value":{"type":"Point","coordinates":[13.3505,52.5144]}},"name":{"type":"Property","value":"Victory Farm"},"owner":{"type":"Relationship","object":"urn:ngsi-ld:Person:person001"}},{"id":"urn:ngsi-ld:Building:farm002","type":"Building","category":{"type":"VocabProperty","vocab":"farm"},"address":{"type":"Property","value":{"streetAddress":"Hardenbergplatz 8","addressRegion":"Berlin","addressLocality":"Tiergarten","postalCode":"10787"},"verified":{"type":"Property","value":true}},"location":{"type":"GeoProperty","value":{"type":"Point","coordinates":[13.3127,52.4893]}},"name":{"type":"Property","value":"Animal Farm"},"owner":{"type":"Relationship","object":"urn:ngsi-ld:Person:person002"}},{"id":"urn:ngsi-ld:Building:tower003","type":"Building","category":{"type":"VocabProperty","vocab":"water_tower"},"address":{"type":"Property","value":{"streetAddress":"John-Foster-Dulles-Allee","addressRegion":"Berlin","addressLocality":"Tiergarten","postalCode":"10557"},"verified":{"type":"Property","value":true}},"location":{"type":"GeoProperty","value":{"type":"Point","coordinates":[13.3598,52.5165]}},"name":{"type":"Property","value":"**Triton Tower**"},"owner":{"type":"Relationship","object":"urn:ngsi-ld:Person:person001",**...**}}]%

Challenges and motivation

Why do we care about this?

- Democratizing technical domains
  - Lowering knowledge barriers
  - Facilitating onboarding and collaboration

- Boosting productivity and speeding up processes
  - Rapid prototyping and scoping
  - Task delegation and automation

- Enhancing experimentations
  - Accelerate iterative testing
  - Focus on domain-related queries instead of technicals

FIWARE

Agenda

1. Learning goals
2. Challenges and motivation
3. **Essential concepts: LLMs, MCP, Agents**
4. Proposed architecture and implementation
5. DEMO: Chatting with the CB
6. Thoughts on improvements and future work
7. Summary & QA

Essential concepts: Large Language Models (LLM)

What are LLMs?

- Advanced AI models trained to understand/generate human-like text.
- Examples: ChatGPT (OpenAI), Gemini (Google), Llama (Meta).
- Trained on massive text datasets (books, articles, code, etc.).

How are they created?

1. **Pre-training**:
   - Model learns patterns from vast text data (unsupervised learning).
   - Uses *transformer architecture* (self-attention for context).
2. **Fine-tuning**:
   - Adjusted for specific tasks (e.g., Q&A, summarization) via human feedback.

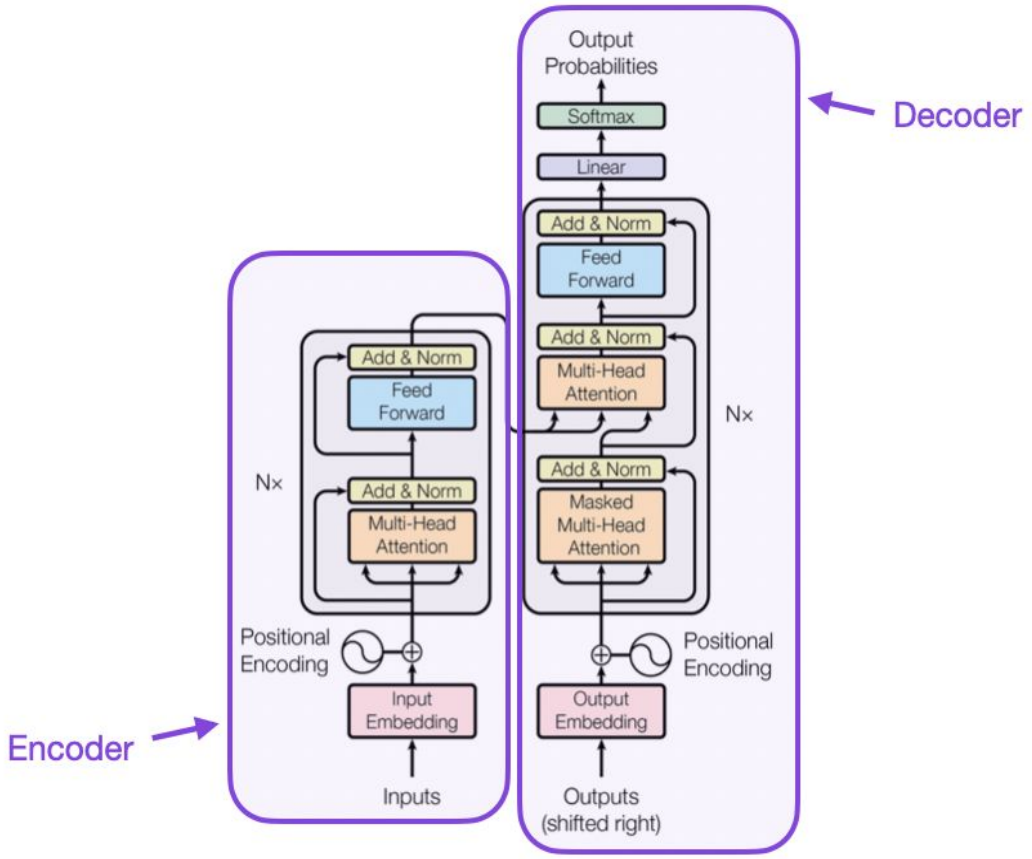


Figure 1: The Transformer - model architecture.

Essential concepts: Large Language Models (LLM)

How do they work?

- Predict next word/token based on input (statistical patterns).
- No "true understanding"—just sophisticated pattern recognition.
- Language models assume that the text can be predicted as a probabilistic system in which the next word depends on the previous ones

Large Language Models (LLM)

Once we have this model, we can complete sentences using the model to obtain each word

When the dog saw the thief barked …

When the dog saw the thief barked to …

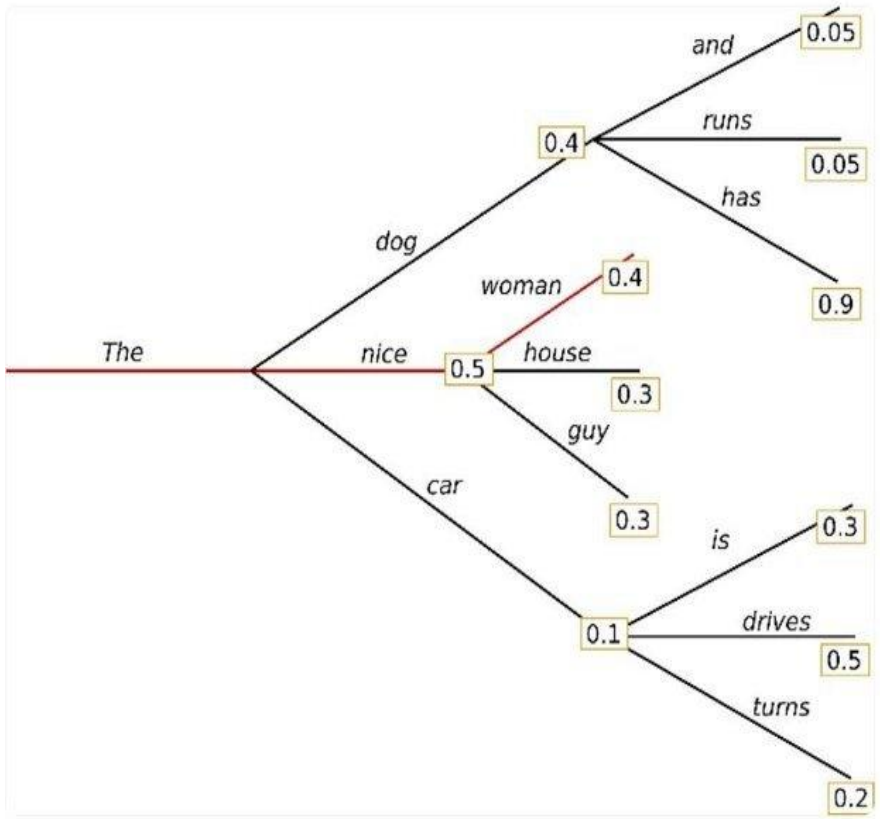
When the dog saw the thief barked to scare …

Large Language Models (LLM)

Once we have this model, we can complete sentences using the model to obtain each word

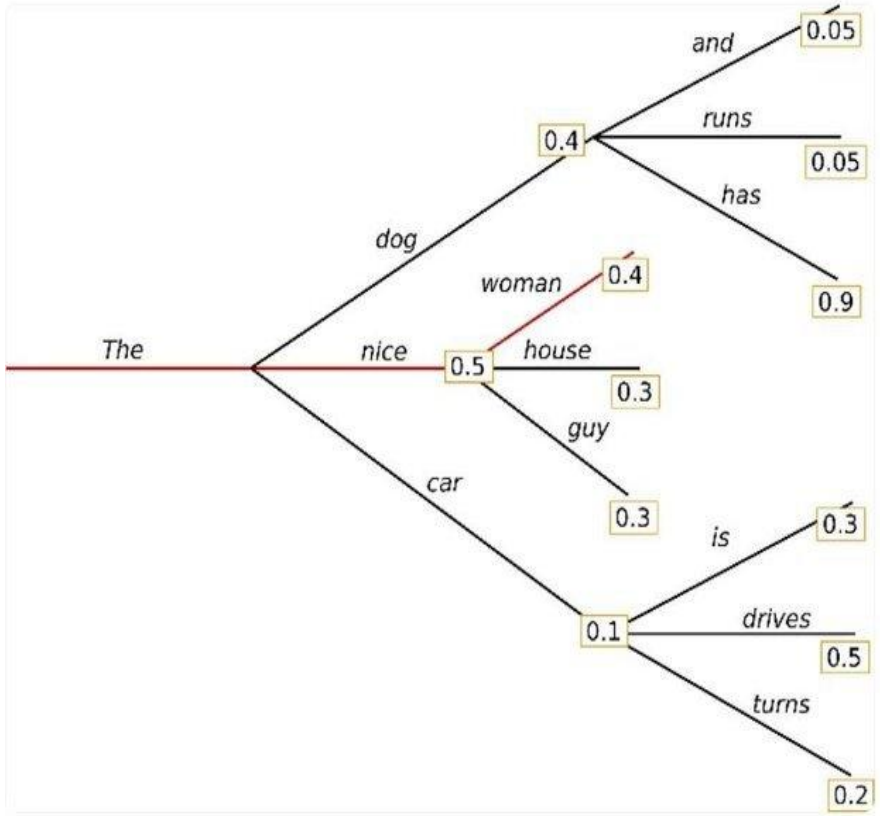
As it is based on the probabilities of the next Word (token), there are a lot of possible paths depending on how the model decides:

- Path more probable -> deterministic (always the same result)
- Roll a die to choose the next Word (token) based on their probabilities

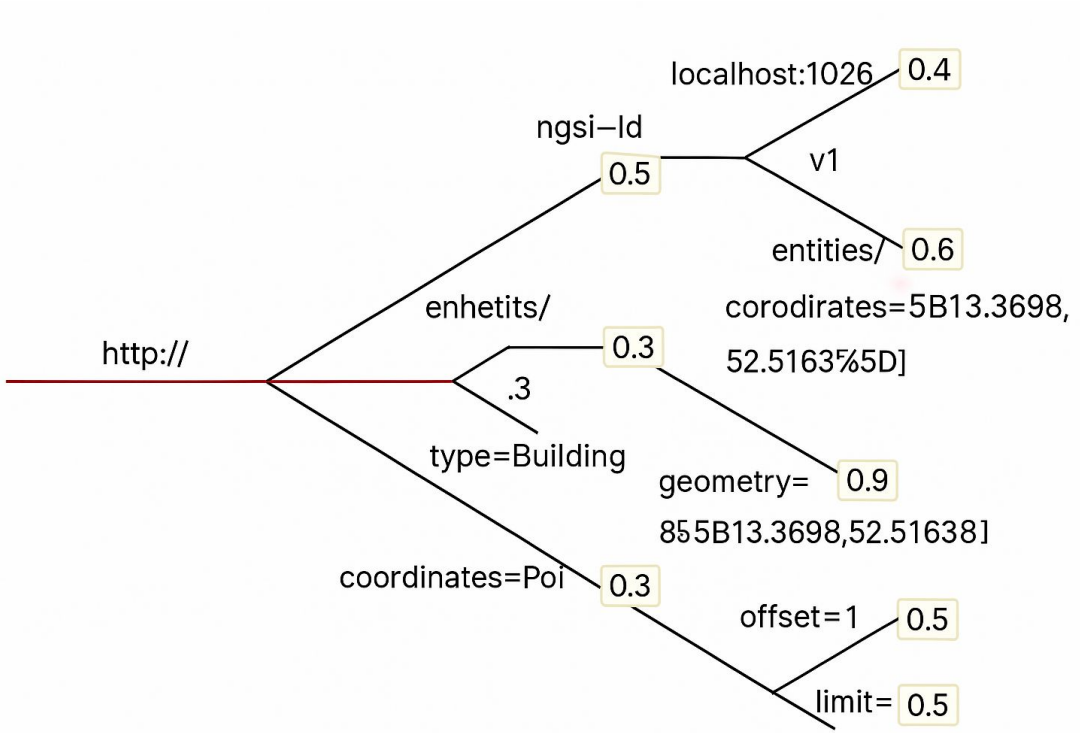


Large Language Models (LLM)

But, what is the difference between



and this?



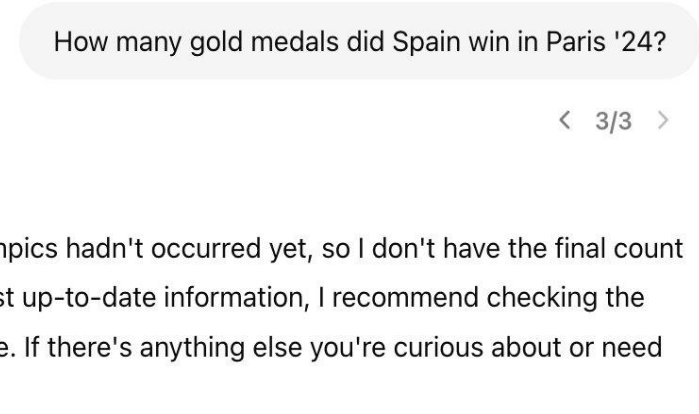
Essential concepts: LLM

Key Advantages

- Versatile: Can write, translate, debug code, etc.
- Context-aware: Handle long-range dependencies in text.
- Zero/few-shot learning: Perform tasks with minimal examp

Challenges

- **Hallucinations**: Generate plausible but false information.
- **Bias**: Reflect biases in training data.
- **Resource-heavy**: Require huge compute/data.
- **Black-box**: Hard to interpret decision-making.
- **Information obsolescence**
- **Lack of expertise in some areas**



Limitations: Context Window and Knowledge of the LLM

To solve these challenges there are alternatives

- Retrieval Augmented Generation (RAG) architectures: the model receives the information from an external sources and answers the question with that information received

- Fine-tuning: retrain-your model with additional data (not valid for real-time data, and not easy to obtain good results)

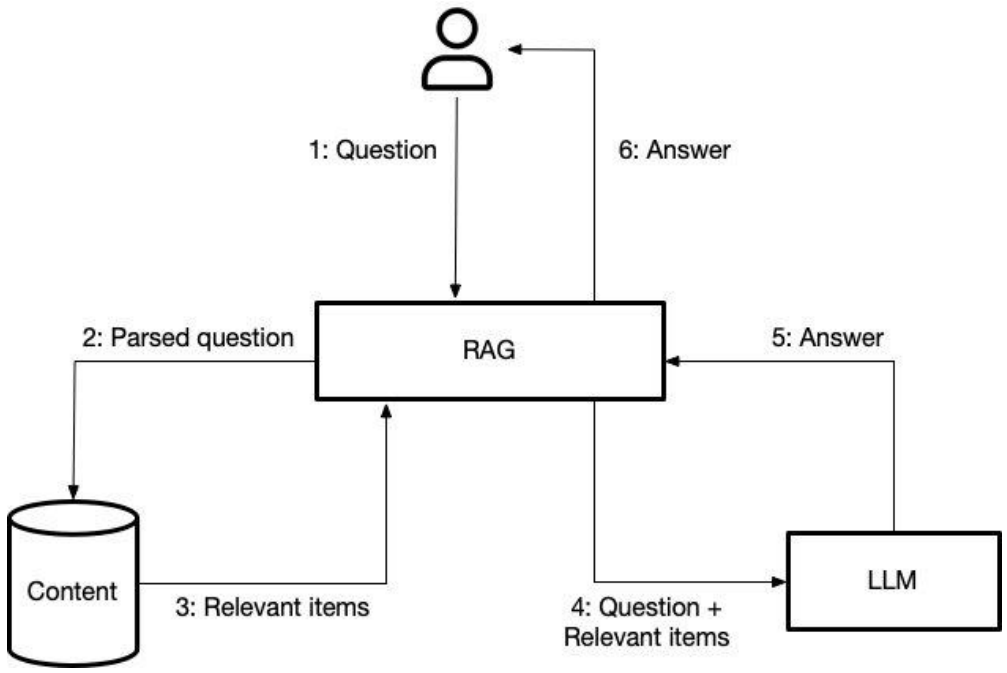
Essential concepts: Retrieval Augmented Generation (RAG)

What is RAG?

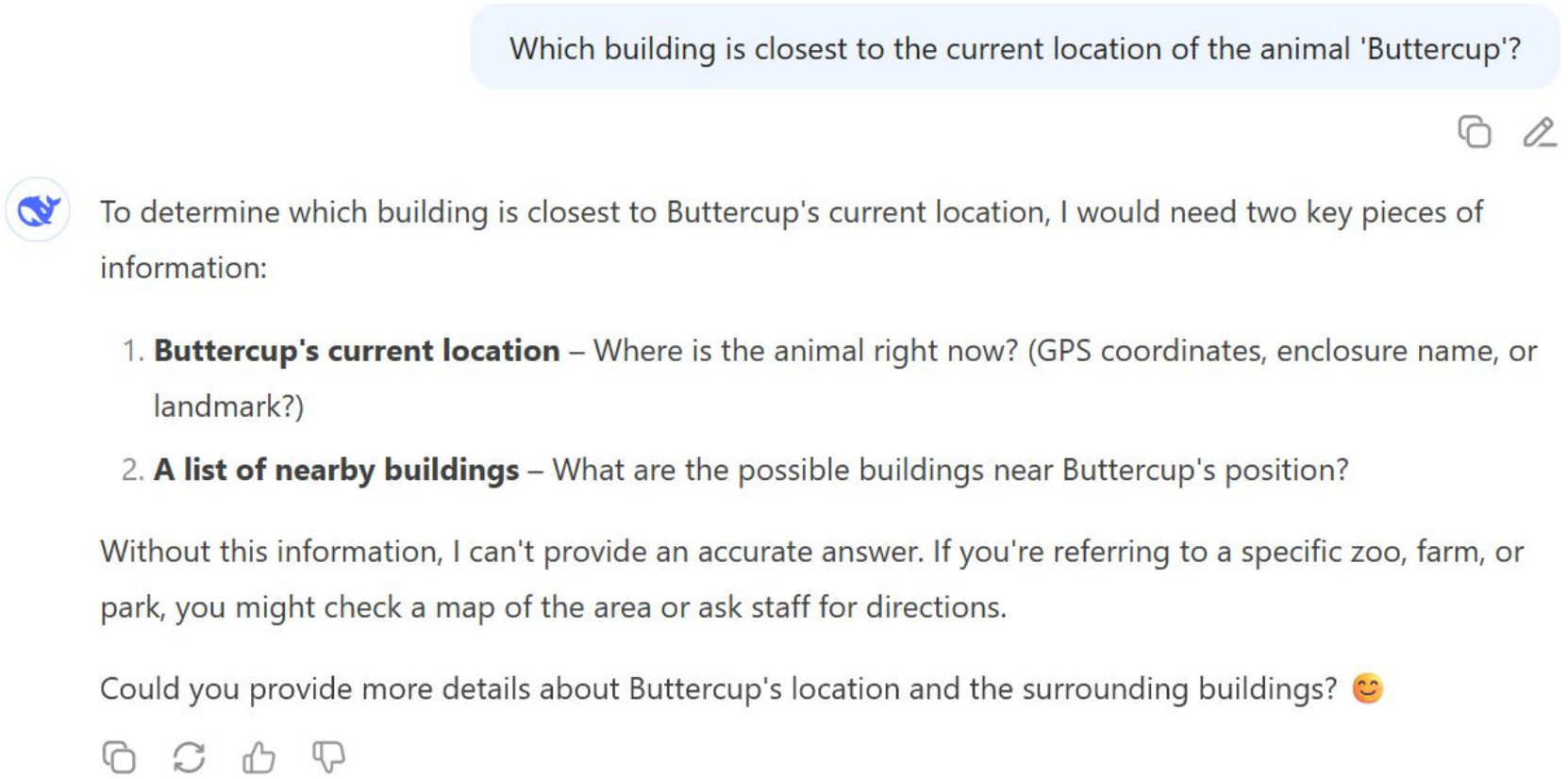
- Hybrid approach combining **retrieval** (search) + **generation** (LLM).
- Allows LLMs to fetch real-time/data-specific info *beyond* their training data.

How It Works

1. **Retrieve**:
   - User query → search external databases/documents (e.g., Wikipedia, company docs,).
   - Finds relevant snippets (*context,* Context Broker).
2. **Augment**:
   - Retrieved context + user query → fed to LLM.
3. **Generate**:
   - LLM produces an answer *grounded* in the retrieved data.



Essential concepts: Retrieval Augmented Generation (RAG)



Essential concepts: Retrieval Augmented Generation (RAG)

Which building is closest to the current location of the animal 'Buttercup'?

Now that Buttercup's location is provided (`[13.2611, 52.2571]`), we can calculate the closest building using the **Haversine formula** to measure distances between her coordinates and the three buildings in the dataset.

Buildings and Their Coordinates:

1. **Victory Farm**: `[13.3505, 52.5144]`
2. **Animal Farm**: `[13.3127, 52.4893]`
3. **Triton Tower**: `[13.3598, 52.5165]`

Calculations:

Using the Haversine formula (approximate distances in kilometers):

Building	Distance to Buttercup (km)
Victory Farm	~29.2 km
Animal Farm	~26.3 km
Triton Tower	~29.5 km

Result:

Animal Farm (coordinates `[13.3127, 52.4893]`) is the **closest building** to Buttercup's current location (`[13.2611, 52.2571]`), at approximately **26.3 km** away.

Additional Notes:

- Buttercup is linked to `barn002` via `locatedAt` , but since this building isn't listed in the dataset, we only compared her location to the three available buildings.
- If `barn002` 's coordinates were provided, it might be closer than the listed farms.

Let me know if you'd like further details! 🐐📍

Essential concepts: Retrieval Augmented Generation (RAG)

Considerations

- Typically the information is stored in a vector database
  - Information is divided in chunks
  - These chunks are stored in vector databases
  - When the user makes a question the most semantic similar chunks are retrieved
  - The question + the chunks are passed to the LLM
  - The LLM answers the question using only the information from the chunks

- But this is not the only option…

FIWARE and RAG

- We can use the Context Broker as the source of content
- ✅ Advantages:
  - Up-to date information
  - Filter by ids or other relevant attributes, etc.
  - Possibility to build NGSI queries using the LLM

Essential concepts: Fine-tuning LLMs

What is Fine-Tuning?

- Adapting a **pre-trained LLM** (e.g., GPT, Llama) to a specialized task/domain by training it further on smaller, task-specific data.

How It Works

1. **Start with Pre-Trained LLM**:
   - Leverages general language knowledge (from massive datasets).
2. **Train on Specialized Data**:
   - Domain-specific examples (e.g., medical reports, legal contracts, **Smart Farm**).
3. **Adjust Weights**:
   - Updates model parameters to excel at the target task.

Essential concepts: Fine-tuning LLMs

Why Use Fine-Tuning?

✅ **Task Specialization**: Improves performance on niche tasks (e.g., code generation, customer support).
✅ **Data Efficiency**: Requires less data than training from scratch.
✅ **Control**: Aligns outputs with specific styles/formats (e.g., formal tone).

Challenges

⚠️ **Compute Cost**: Requires GPUs/TPUs.
⚠️ **Overfitting**: Risk of losing generalization if data is too narrow.
⚠️ **Data Quality**: Needs clean, labeled examples.

Essential concepts: MCP & Agents

What Are AI Agents?

- **Definition**: Autonomous software entities that perceive their environment, make decisions, and take actions to achieve specific goals.
- **Capabilities**:
  - Reasoning and planning
  - Learning from interactions
  - Executing complex tasks with minimal human intervention
- **Examples**: Virtual assistants, automated customer service bots, and intelligent data analysis tools.

Essential concepts: MCP & Agents

What is MCP (Model Context Protocol)?

MCP is an open protocol that standardizes how applications provide context to LLMs. Think of MCP like a USB-C port for AI applications. Just as USB-C provides a standardized way to connect your devices to various peripherals and accessories, MCP provides a standardized way to connect AI models to different data sources and tools.

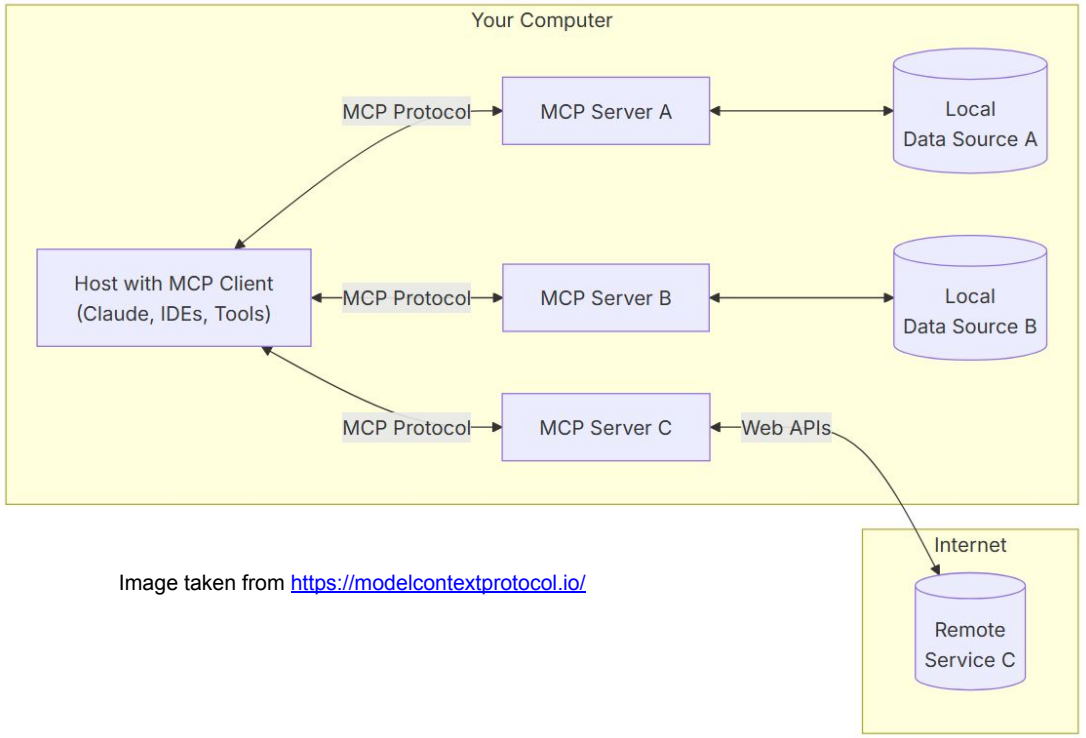


Image taken from https://modelcontextprotocol.io/

Essential concepts: MCP internal working

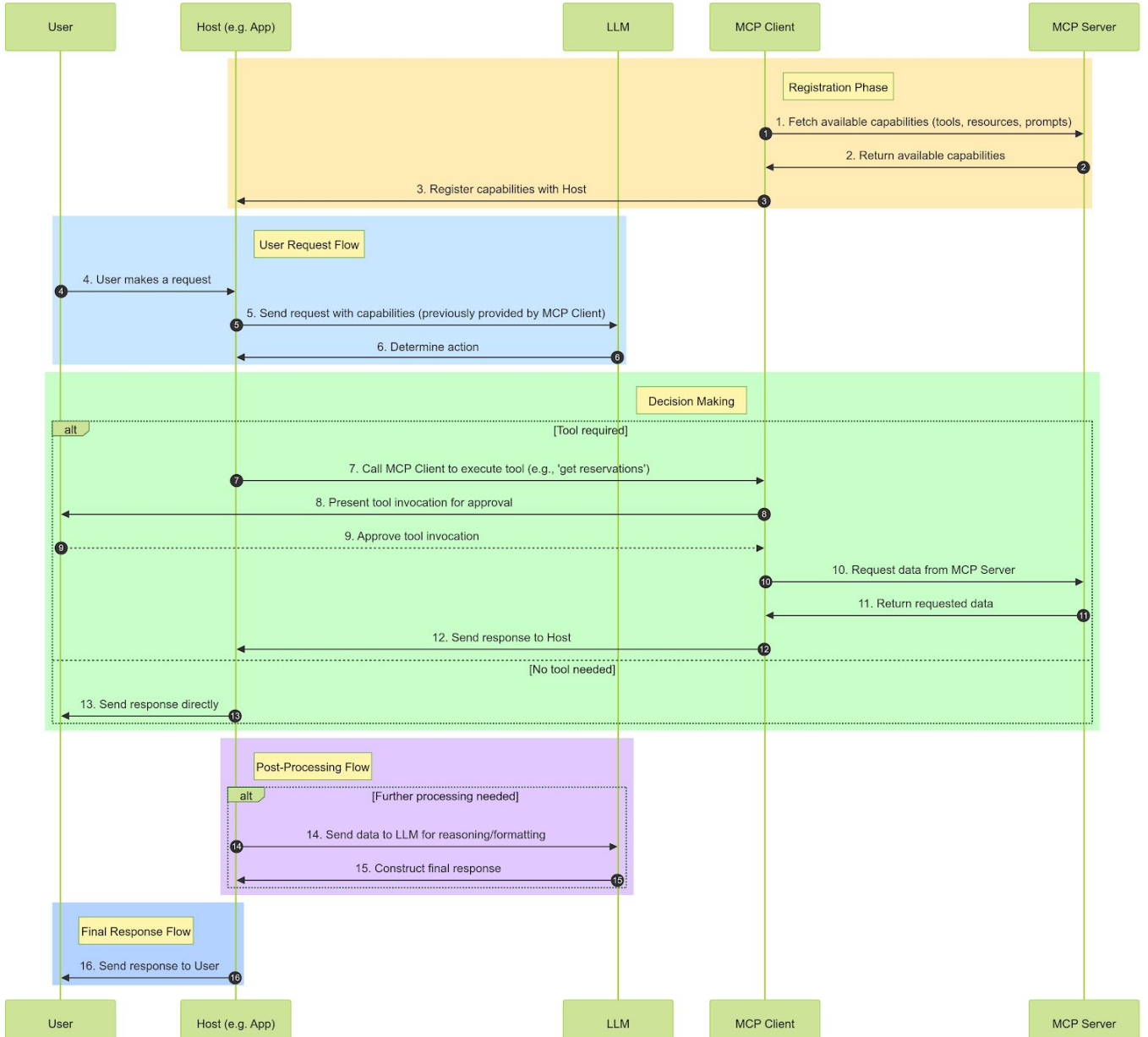


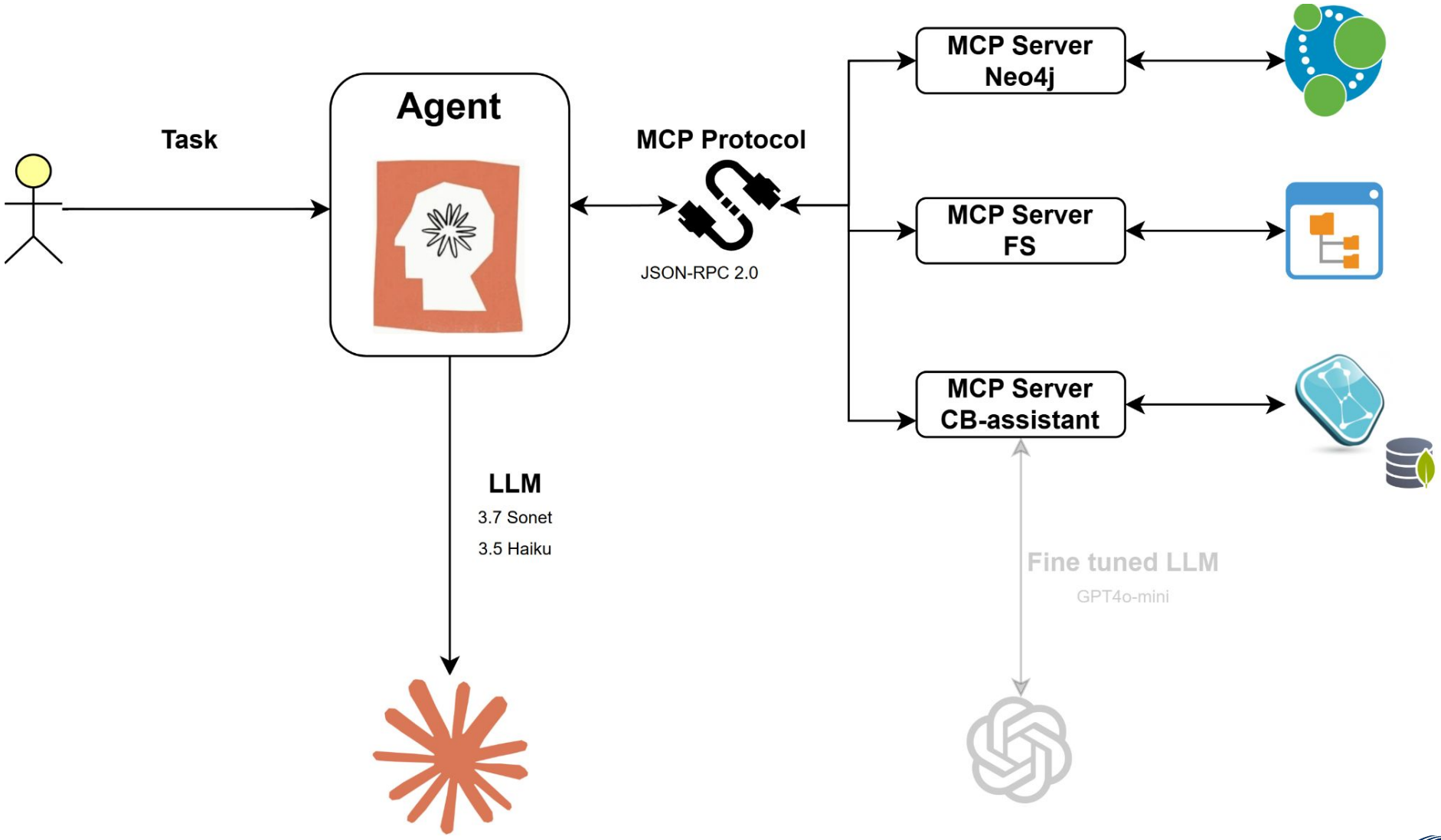
Diagram taken from [here]

FIWARE

Agenda

1. Learning concepts
2. Problem and motivation
3. LLMs + MCP = Agents
4. **Proposed architecture and implementation**
5. DEMO: Chatting with the CB
6. Thoughts on improvements and future work
7. Summary

Proposed architecture



Implementation - CB MCP Tools

```
# This tool gets an entity by ID from the Context Broker
@mcp.tool()
def get_entity_by_id(address: str="localhost", port: int=1026, entity_id: str="") -> str:
    """Get an entity by ID from the Context Broker."""
    try:
        url = f"http://{address}:{port}/ngsi-ld/v1/entities/{entity_id}"
        response = requests.get(url)
        response.raise_for_status()
        return json.dumps(response.json())
    except requests.exceptions.RequestException as e:
        return json.dumps({"error": str(e)})
```

Other MCP Tools are:

- CB_version
- get_entity_types
- get_all_entities
- get_entity_by_id
- publish_to_CB
- magic_query_context_broker →

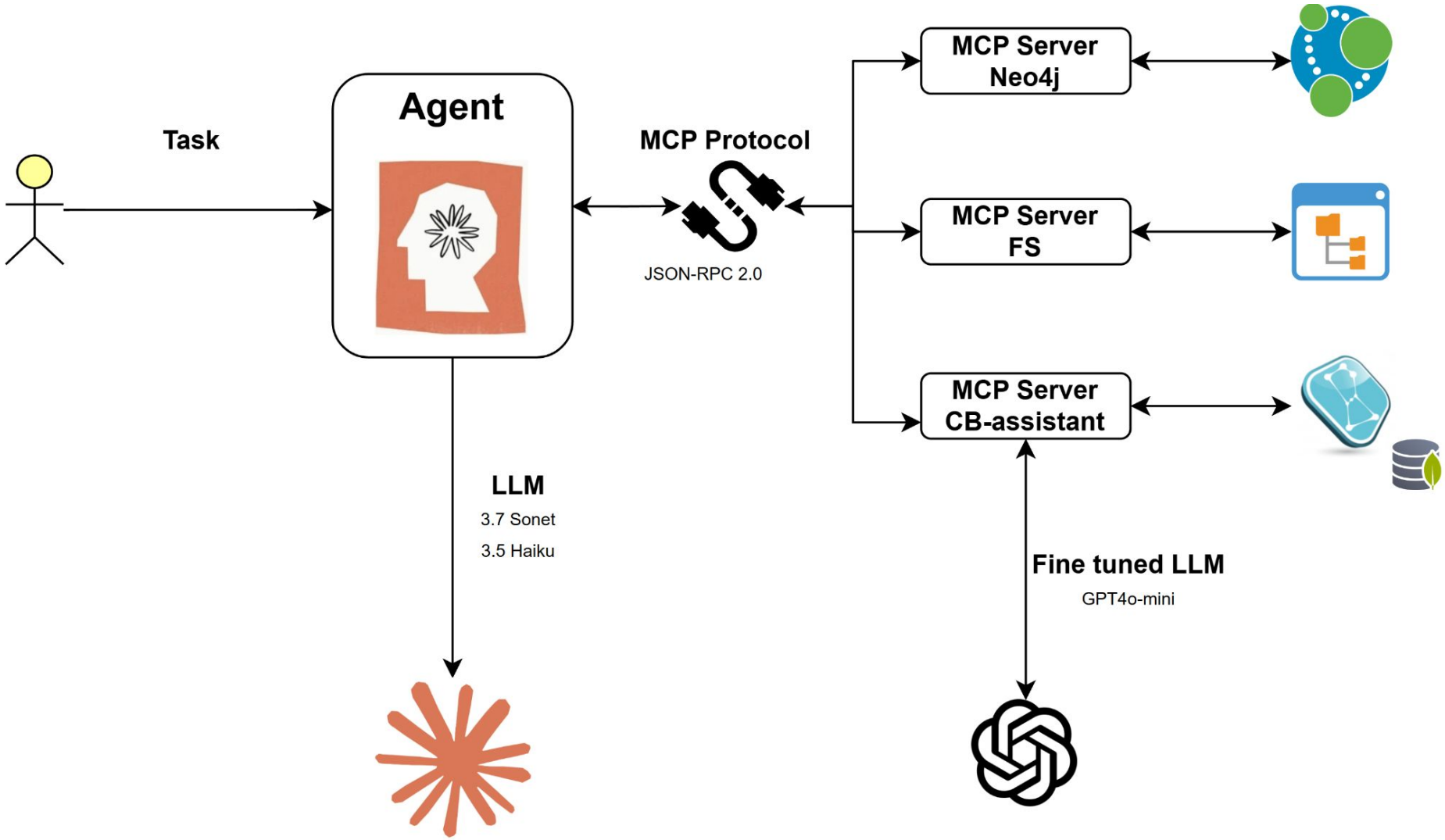
Implementation - MCP magic query CB

```python
# This tool asks an LLM to generate a valid NGSI-LD query from a natural language query
@mcp.tool()
def magic_query_context_broker(query: str= "", address: str="localhost", port:
int=1026, custom_headers: object) -> str:
    """
    If none of the other tools fit to interact with the context broker, this should be
used to generate a valid NGSI-LD query from a natural language query.
    …
    """
    try:
        # Load instructions from external file
        filepath = os.path.join(os.path.dirname(__file__),   "instructions.txt")
        with open(filepath,  "r") as file:
            INSTRUCTIONS_FOR_FTGPT4o_mini_FIWARE = file.read()

        # Prepare the request to the OpenAI API
        filepath = os.path.join(os.path.dirname(__file__),   ".env")
        logger.debug( f"Loading OpenAI API key from  {filepath}")
        env_vars = {}
        with open(filepath,  "r") as file:
            for line in file:
                . . .
        openai_client = openai.OpenAI(http_client=CustomHTTPClient(),
api_key=env_vars[ 'OPENAI_API_KEY' ])

        # Call the fine-tuned model which returns a curl command to run in the CB
        response = openai_client.chat.completions.create(
            model="ft:gpt-4o-mini-2024-07-18:fine-tuned:fiware-translator-v3-01:XXX",
            messages=[
                {"role": "system", "content": INSTRUCTIONS_FOR_FTGPT4o_mini_FIWARE},
                {"role": "user", "content": query}
            ]
        )
```

```python
        # Extract the NGSI-LD query from the model response
        ngsi_query = response.choices[ 0].message.content
        logger.debug( f"Generated NGSI-LD query from fine tuned OpenAI model:
{ngsi_query}")

        # Call helper method to prepare the request parameters
        broker_url, headers, method, data, params =
prepare_request_from_curl(ngsi_query, address, port, custom_headers)

        # Execute the query against the Context Broker
        if method == 'GET':
            cb_response = requests.get(broker_url, headers=headers, params=params)
        elif method == 'POST':
            cb_response = requests.post(broker_url, headers=headers, json=data,
params=params)
        elif method == 'PATCH':
            cb_response = requests.patch(broker_url, headers=headers, json=data,
params=params)
        else:
            raise ValueError(f"Unsupported HTTP method:  {method}")

        cb_response.raise_for_status()

        return json.dumps({
            "original_query" : query,
            "ngsi_query" : ngsi_query,
            "results": cb_response.json()
        })

except FileNotFoundError:
        . . .
```


FIWARE

Proposed architecture



Fine-tuning for magic query

```
{
  "messages": [
    {
      "role": "system",
      "content": "I need you to perform as an expert in NGSI-LD
        with a background on semantic modeling and Smart Data
        Models (SDM). I will then send you some queries in
        Natural Language and you should convert them into NGSI
        -LD fully executable code to run in a FIWARE system (a
        Context Broker). Assume you are connected to a context
        broker in localhost and port 1026 and that your answers
        will be directly forwarded to the context broker; so
        send only the set of commands that would be sent to the
        CB, no explanation at all unless requested."
    },
    {
      "role": "user",
      "content": "Find all Restaurant entities serving
        traditional cuisine within 500 meters of Santa Fe's
        iconic Suspension Bridge (El Puente Colgante) (31.6390S
        , 60.6927W)."
    },
    {
      "role": "assistant",
      "content": "curl -G -X GET 'http://localhost:1026/ngsi-ld
        /v1/entities' -H 'Accept: application/json' -d 'type
        =Restaurant' -d 'q=cuisine==\"argentinian\"' -d
        'geometry=Point' -d 'coordinates=%5B-60.6927,-31
        .6390%5D' -d 'georel=near%3BmaxDistance==500'"
    }
  ]
}
```

- We defined the instructions
- And the pair < prompt , LLM response >
- Generated a corpus of +2600 NGSI-LD queries for training
- All queries correspond to GETting data from the CB (no push, no update, no delete)
- Training on a small subset of possibilities (for example no training on temporal interfaces)
- The idea is to capture the structure of NGSI-LD queries based on pattern in questions.
- Obviously, without semantic this fail miserably. The LLM cannot guess the name of attributes nor entities

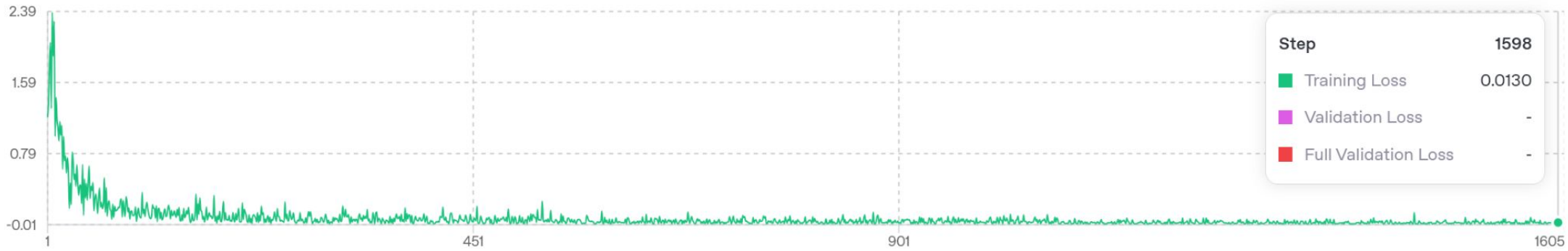
FIWARE

Fine-tuning for magic query

Messages Metrics

15:09:57	The job has successfully completed
15:09:51	New fine-tuned model created
15:09:51	Checkpoint created at step 1070
15:09:51	Checkpoint created at step 535
14:37:52	Fine-tuning job started
14:37:49	Files validated, moving job to queued state
14:35:08	Validating training file: file-TodnyNu7qhsUp3nUCmMqkA
14:35:08	Created fine-tuning job: ftjob-tYUWMv9Ib6IKoeJSFsXQdsXr

Training loss 0.0023



Agenda

1. Learning concepts
2. Problem and motivation
3. LLMs + MCP = Agents
4. Proposed architecture and implementation
5. **DEMO: Chatting with the CB**
6. Thoughts on improvements and future work
7. Summary

Demo

The demo consist in a Smart Farm app with some Animal entities (pig and cows), some buildings, person and relationships among them, along with attributes. For instance, cows have weight, owner, location, etc.

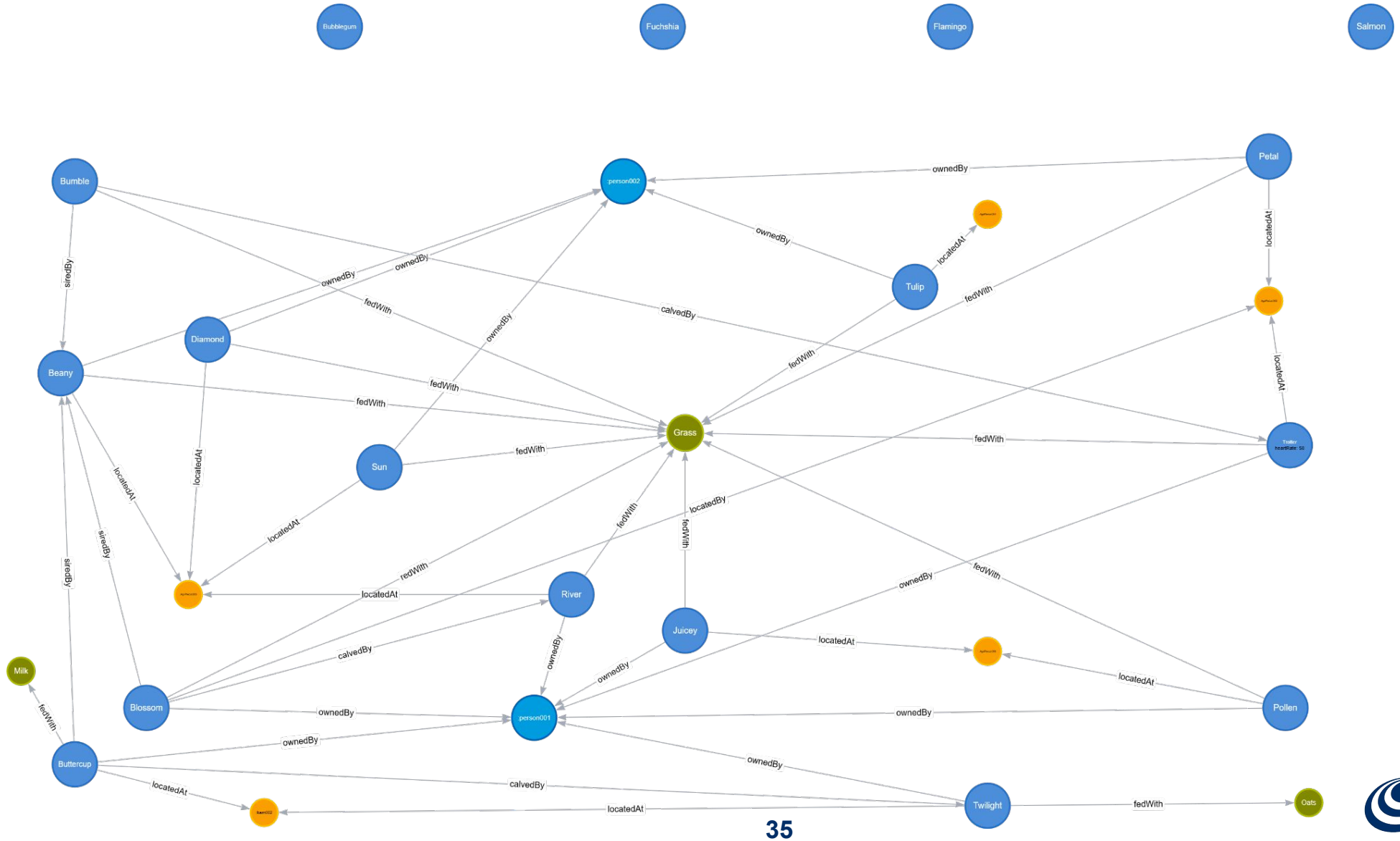
Buttercup - `urn:ngsi-ld:Animal:cow103`

Species:	dairy cattle		
Legal ID:	F-cow103-Buttercup		
Phenological Condition:	lactatingBaby	**Reproductive Condition:**	noStatus
Sex:	Female	**Date of Birth:**	unknown
Fed With:	Milk	**Weight**	20 kgs
Calved by:	urn:ngsi-ld:Animal:cow006		
Sired by:	urn:ngsi-ld:Animal:cow001		
Located At:	urn:ngsi-ld:Building:barn002		
Owned By:	urn:ngsi-ld:Person:001		

Comment

```
Buttercup was born today.
```

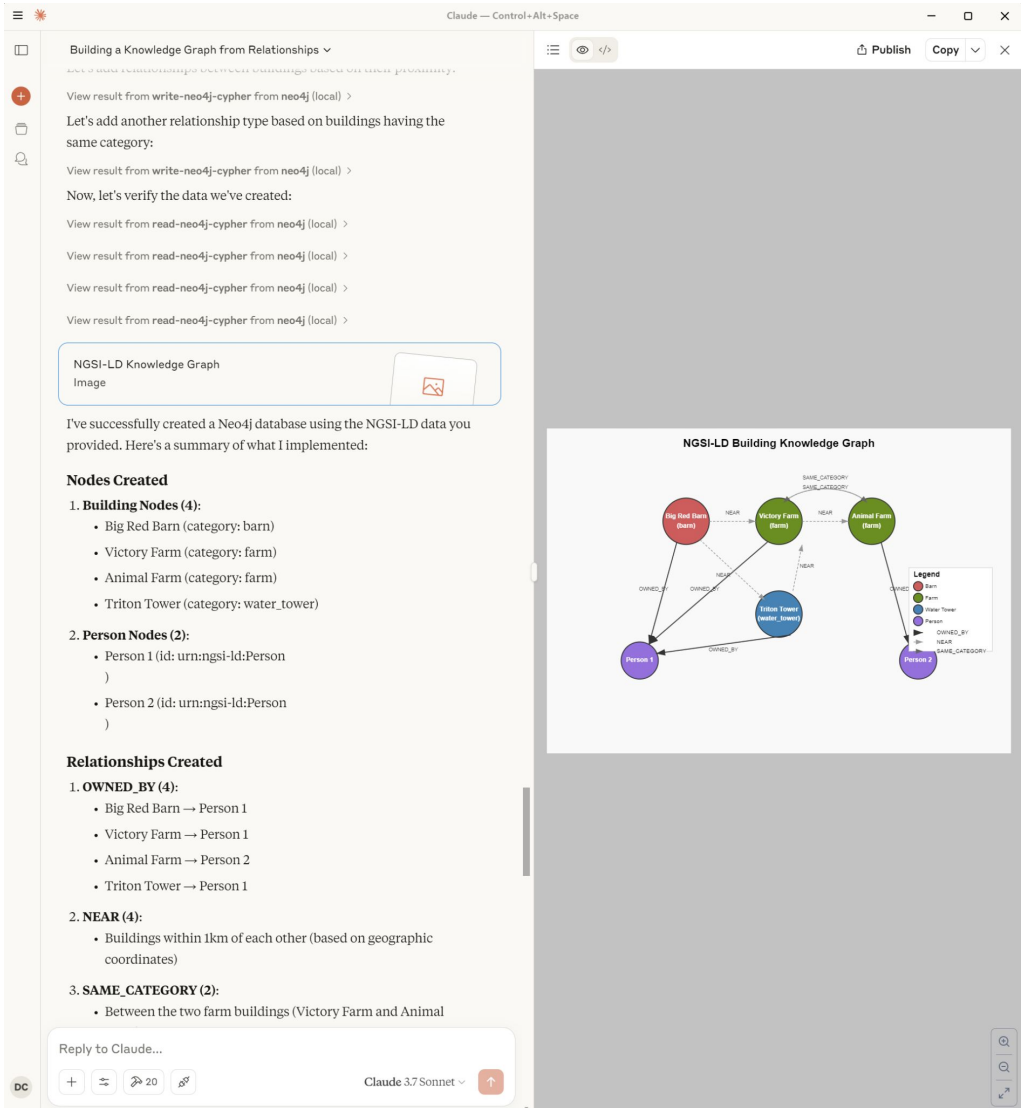
Demo - Relationship among entities in the Smart Farm



Demo

Tasks:

- Ask info or version of the CB
- List entity types to understand the domain of the content pointed by the CB
- Could you please list the Animals in-heat ready to be inseminated?
- What animals weight less than 50 KG?
- Could you create a knowledge graph with the info pointed by the CB using a neo4j database and cypher language? Store the nodes and relationships into the DB.
- …



Agenda

1. Learning concepts
2. Problem and motivation
3. LLMs + MCP = Agents
4. Proposed architecture and implementation
5. DEMO: Chatting with the CB
6. **Thoughts on improvements and future work**
7. Summary

Improvements and future work

- Nice to test this in local, deploying a LLM (ollama)
- Extending server capabilities (update entities, batch publish, delete🚧)
- Improve the RAG context, use of embeddings for chunks of data
- Add semantic support, follow SDM links definitions
- Add more integrations exploring other useful MPC Servers

FIWARE

Agenda

1. Learning concepts
2. Problem and motivation
3. LLMs + MCP = Agents
4. Proposed architecture and implementation
5. DEMO: Chatting with the CB
6. Thoughts on improvements and future work
7. **Summary**

Summary

- **LLM** ⊂ GenAI ⊂ Deep Learning ⊂ Machine Learning ⊂ Artificial Intelligence
- **RAG** (Retrieval-Augmented Generation) in LLMs refers to a technique that combines large language models with external retrieval systems to fetch relevant documents or data, used to enhance the model's response generation
- We have used **FIWARE** to perform the **RAG** and support the **LLM** with contextual data from a smart farm in real time
- The **CB** is used as a central piece that handles the contextual data
- This sets up the basis for a general architecture that could be used to enhance the experience in FIWARE solutions and could be the ground for other types of **powered-by-FIWARE** solutions that could be improved to interact with users.
- A nice take away is that we tackle syntax of NGSI-LD queries with fine-tuning and the semantics with RAG providing context (e.g. name if attributes).
- MCP is an open source technology that opens the door for agentic applications and boost solutions with ease.

FIWARE

Resources

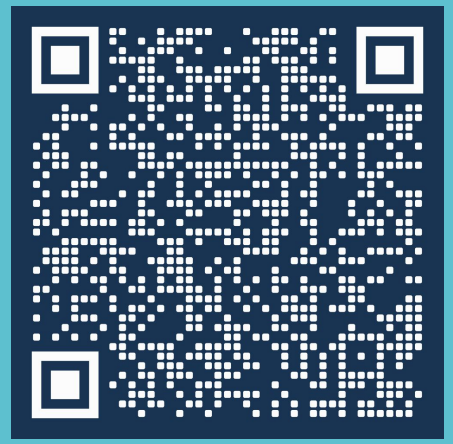
- Repository with code and slides: https://github.com/dncampo/webinar-chat-with-cb
- FIWARE MCP Server CB-assistant repo: https://github.com/dncampo/FIWARE-MCP-Server/tree/main
- More video demos: https://www.youtube.com/playlist?list=PLcPI_2NtrguWhkCXmGZyGOE05hfIM0o0Z
- MCP documentation, Client list, Server list
  - https://modelcontextprotocol.io/introduction
  - https://github.com/modelcontextprotocol/servers
  - https://modelcontextprotocol.io/clients
- LLMs reference papers (transformers, RAG, fine-tuning):
  - Attention Is All You Need Google scientific paper that defined Transformers architecture, 2017
  - Retrieval-Augmented Generation for Large Language Models: A Survey
  - The Ultimate Guide to Fine-Tuning LLMs from Basics to Breakthroughs: An Exhaustive Review of Technologies, Research, Best Practices, Applied Research Challenges and Opportunities

FIWARE

Thank you!

http://fiware.org
Follow @FIWARE on Twitter

Contact me!



david.campo@fiware.org

FIWARE

FIWARE Global Summit

HOSTED BY Rabat Smart City

Shaping Sustainable, Innovative and Secure Smart Cities

22-23 May, 2025

Rabat Smart City, Morocco

#FIWARESummit25

FIWARE UM6P