NMRG 86th meeting
IETF 125 Shenzhen

Agentic AI for Intent-Based Network Management

Abdelkader Mekrache, Adlen Ksentini
Communication Systems, EURECOM

Table of Contents

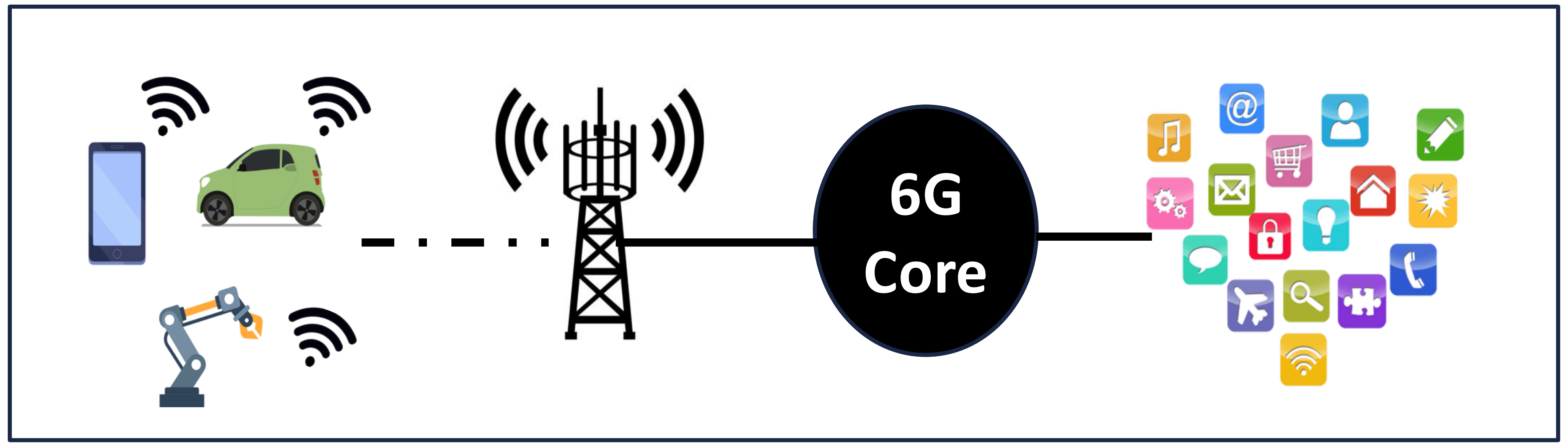
- Intent-Based Networking (IBN)
- Intent Translation with LLMs
- Intent Assurance with LLMs
- Conclusion

Table of Contents

- Intent-Based Networking (IBN)
- Intent Translation with LLMs
- Intent Assurance with LLMs
- Conclusion

Intent-Based Networking (IBN)

OSS: Operations Support System

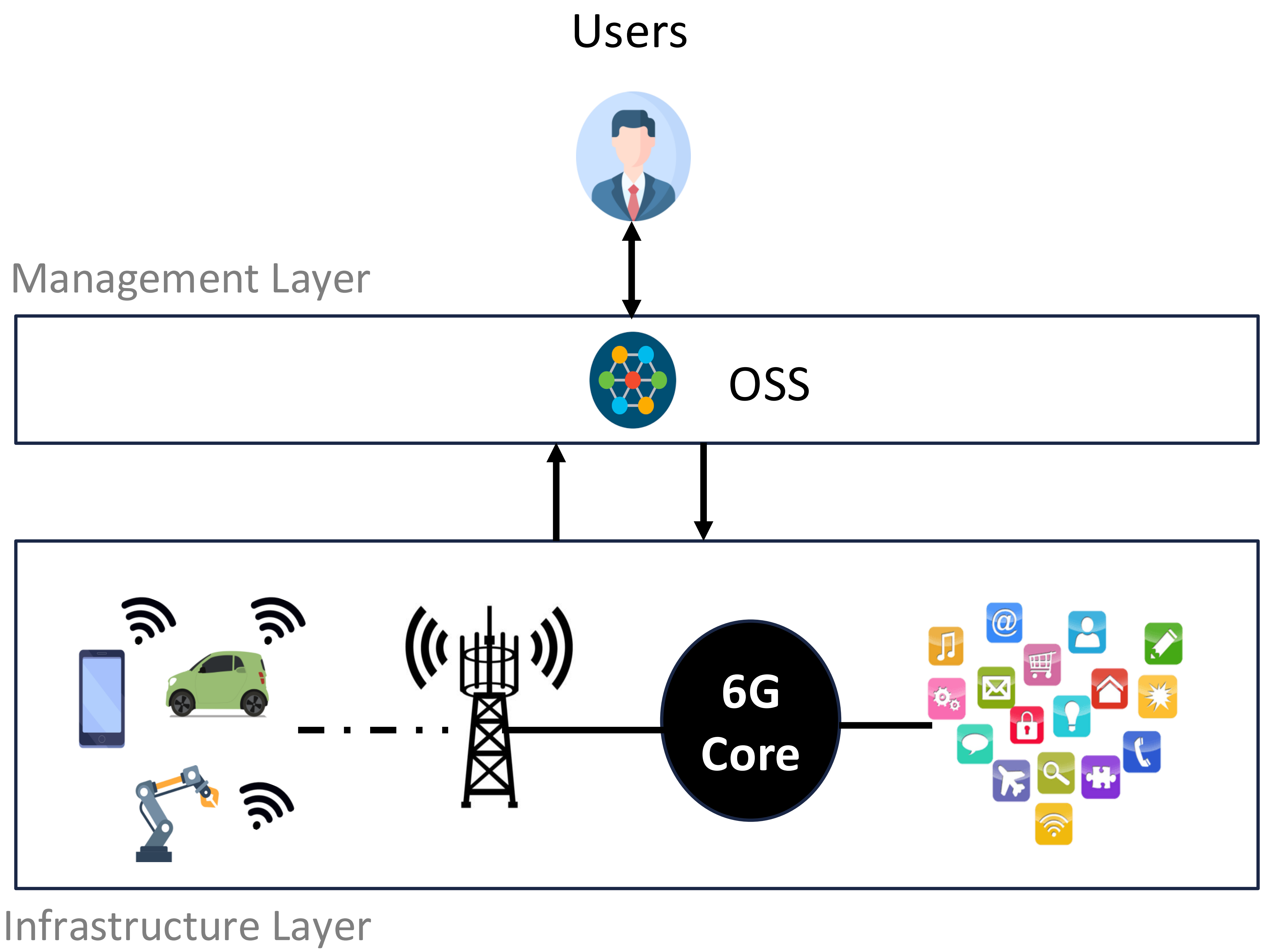


Infrastructure Layer

Intent-Based Networking (IBN)

OSS: Operations Support System

Traditional network management relies on low-level, and technology-specific configurations, making it error-prone, and poorly suited for autonomous networks.



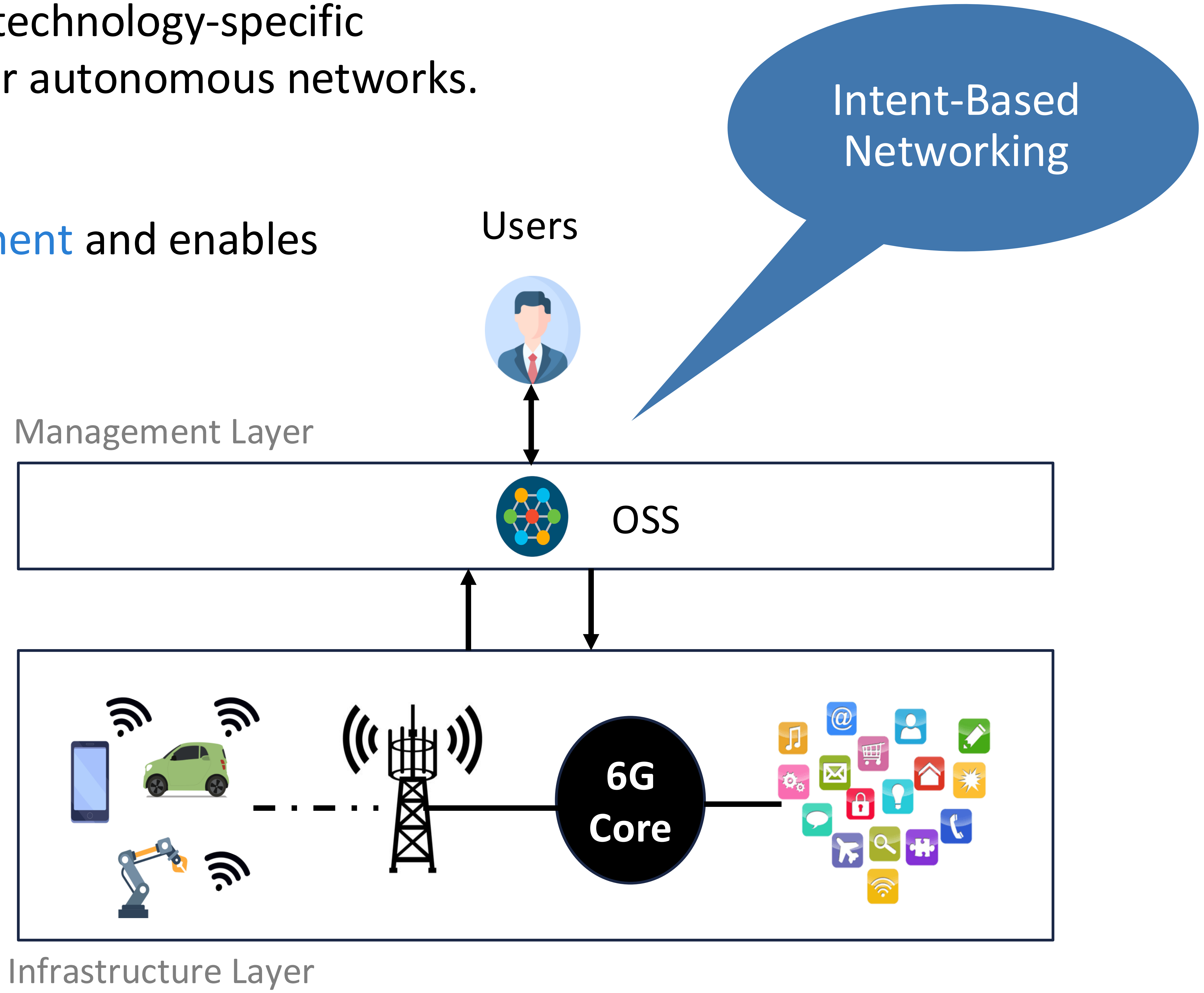
Intent-Based Networking (IBN)

OSS: Operations Support System

❗ Traditional network management relies on low-level, and technology-specific configurations, making it error-prone, and poorly suited for autonomous networks.

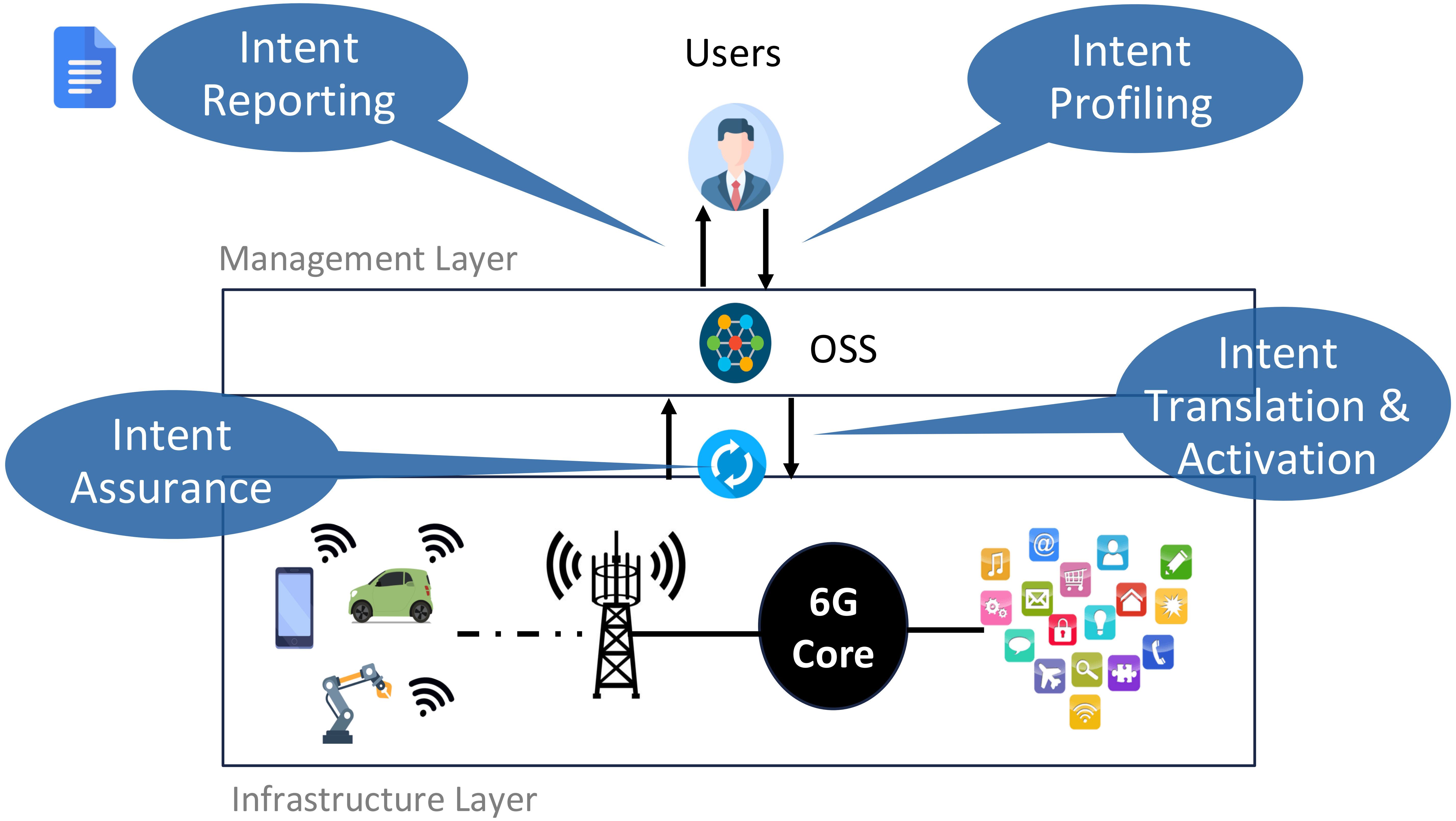
✓ Intent-based Networking (IBN) simplifies network management and enables the evolution toward autonomous networks

- ○ Intent abstracts complexity by allowing users to express *what they want* , not *how to implement it*.
- ○ It supports closed-loop control allowing systems to continuously adapt service objectives without manual intervention.



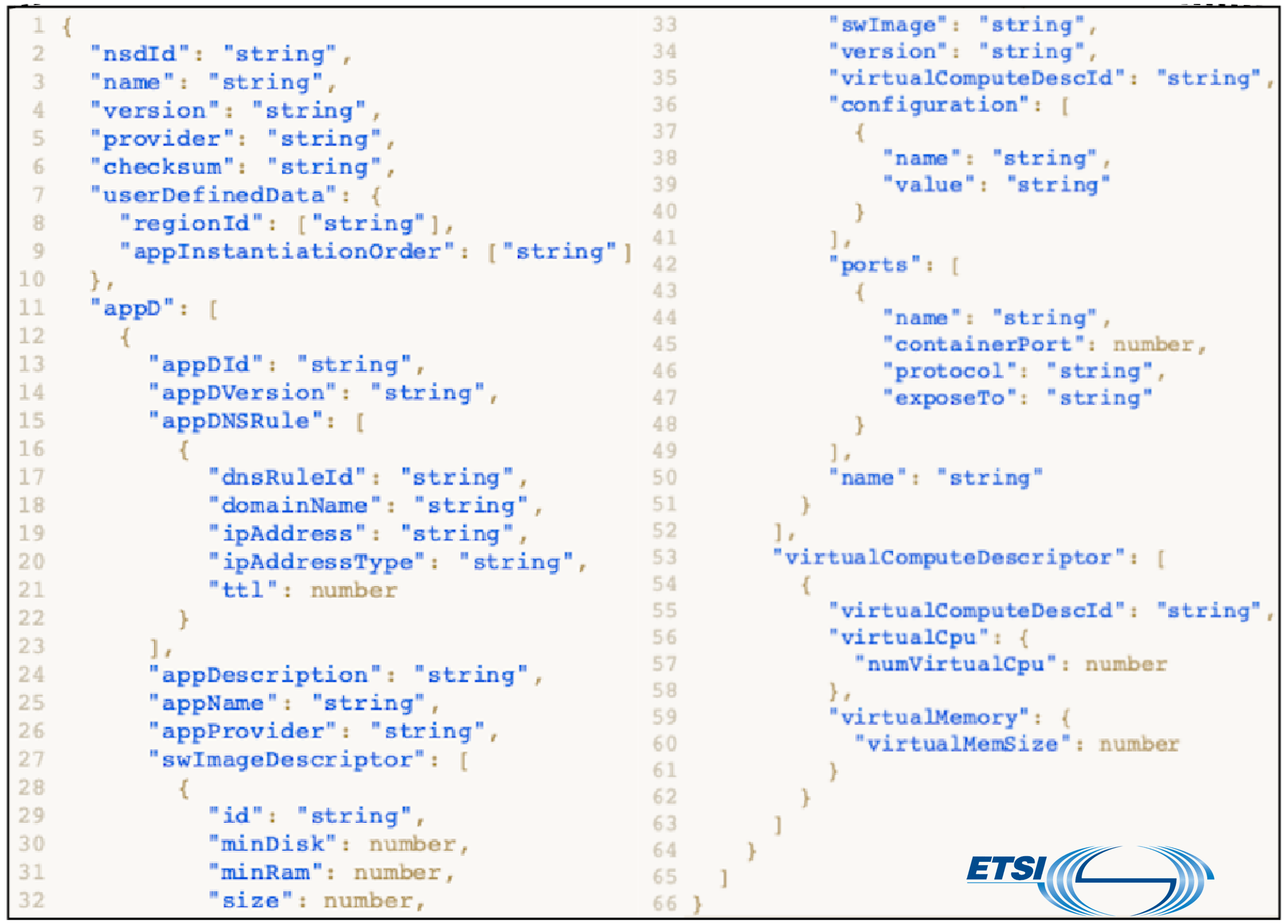
Intent-Based Networking (IBN)

- Intent Profiling, declare the intent using a declarative structure.
- Intent Translation, translate high-level intents into low-level configurations.
- Intent Activation, activate the intents within the infrastructure.
- Intent Assurance, ensure the intents respect the specified requirements.
- Intent reporting, provide feedback to users about the status of the intent.

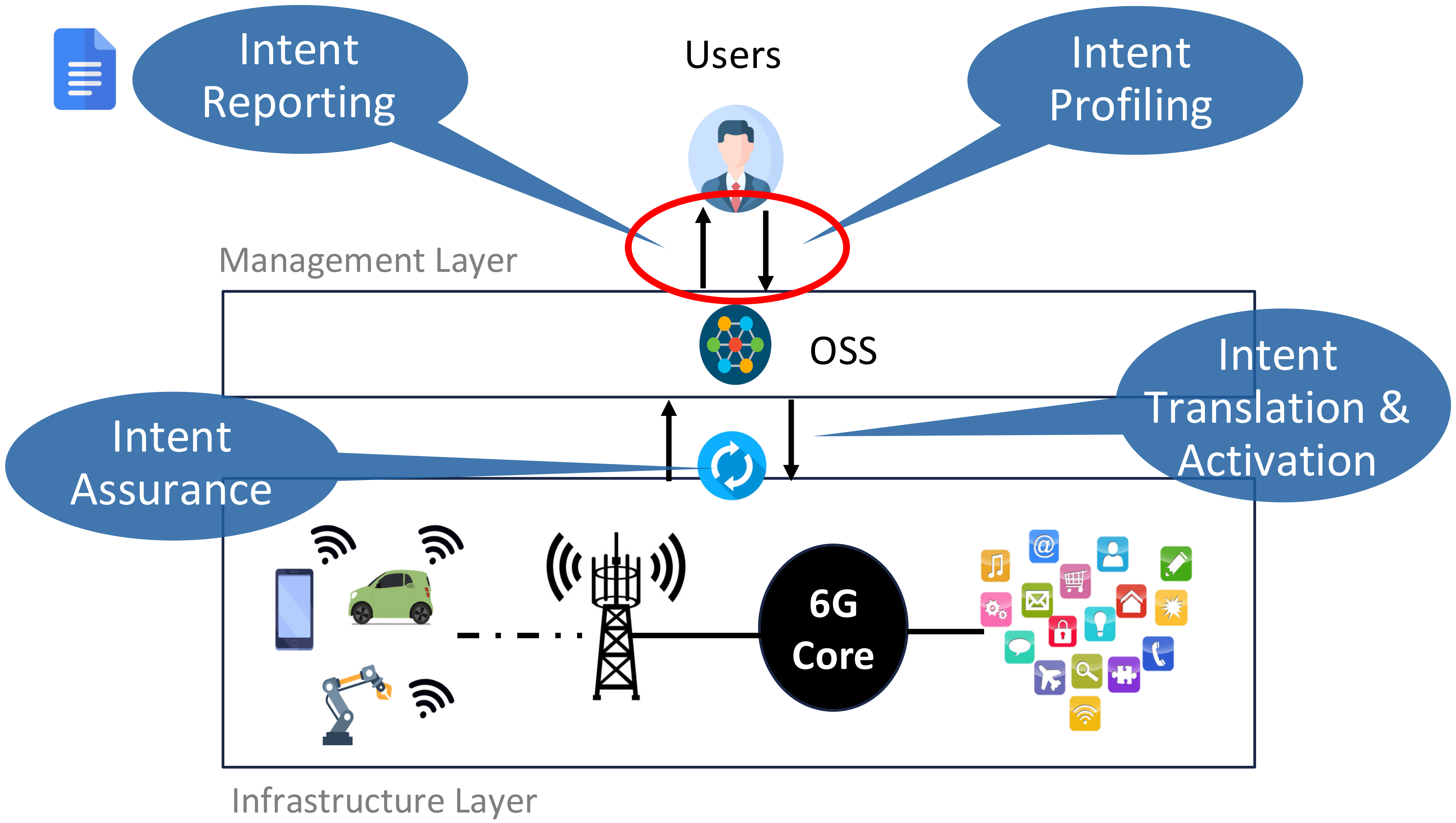


Challenges

Different standardization bodies (3GPP, TM Forum, ETSI) define heterogeneous OSS API endpoints and JSON structures.

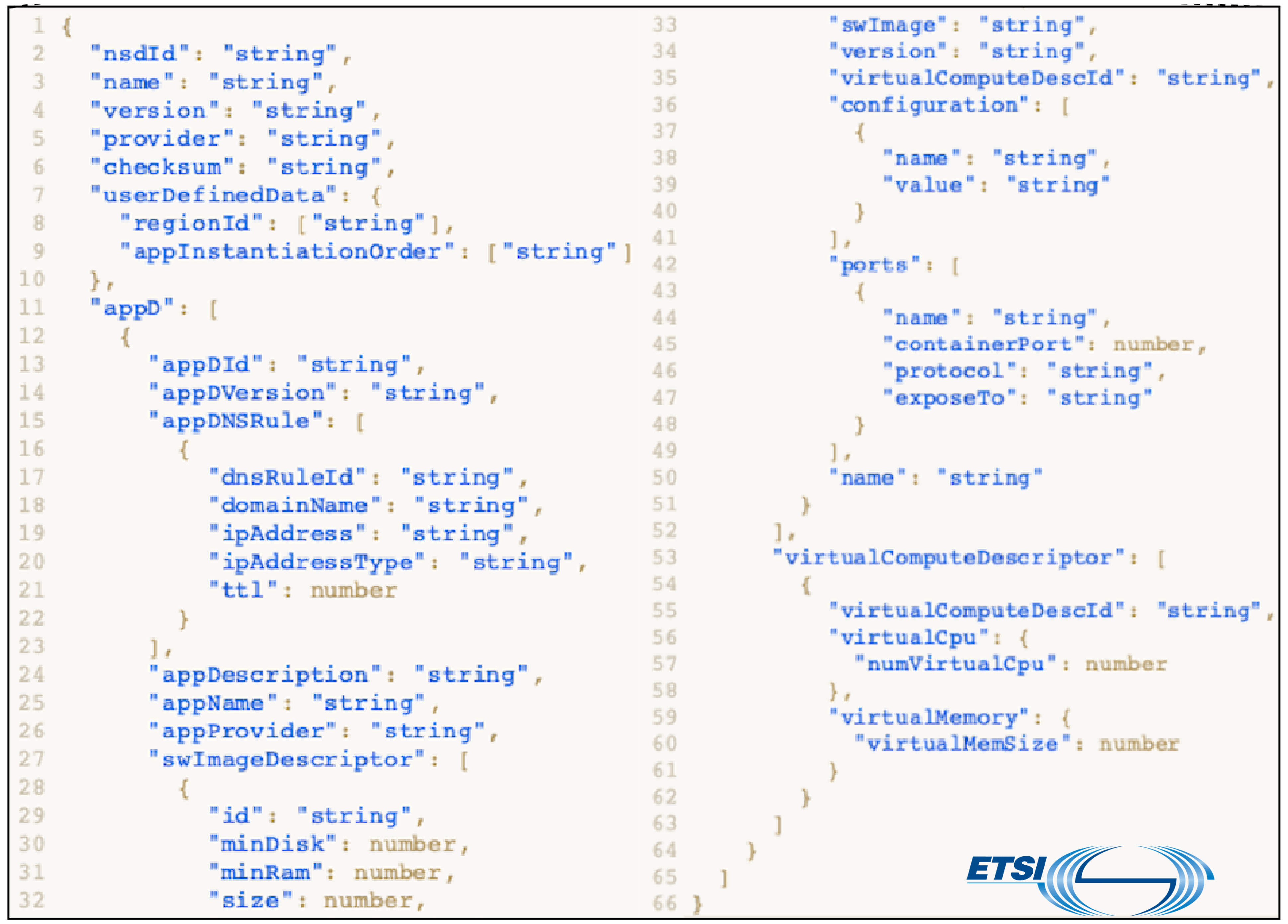


```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

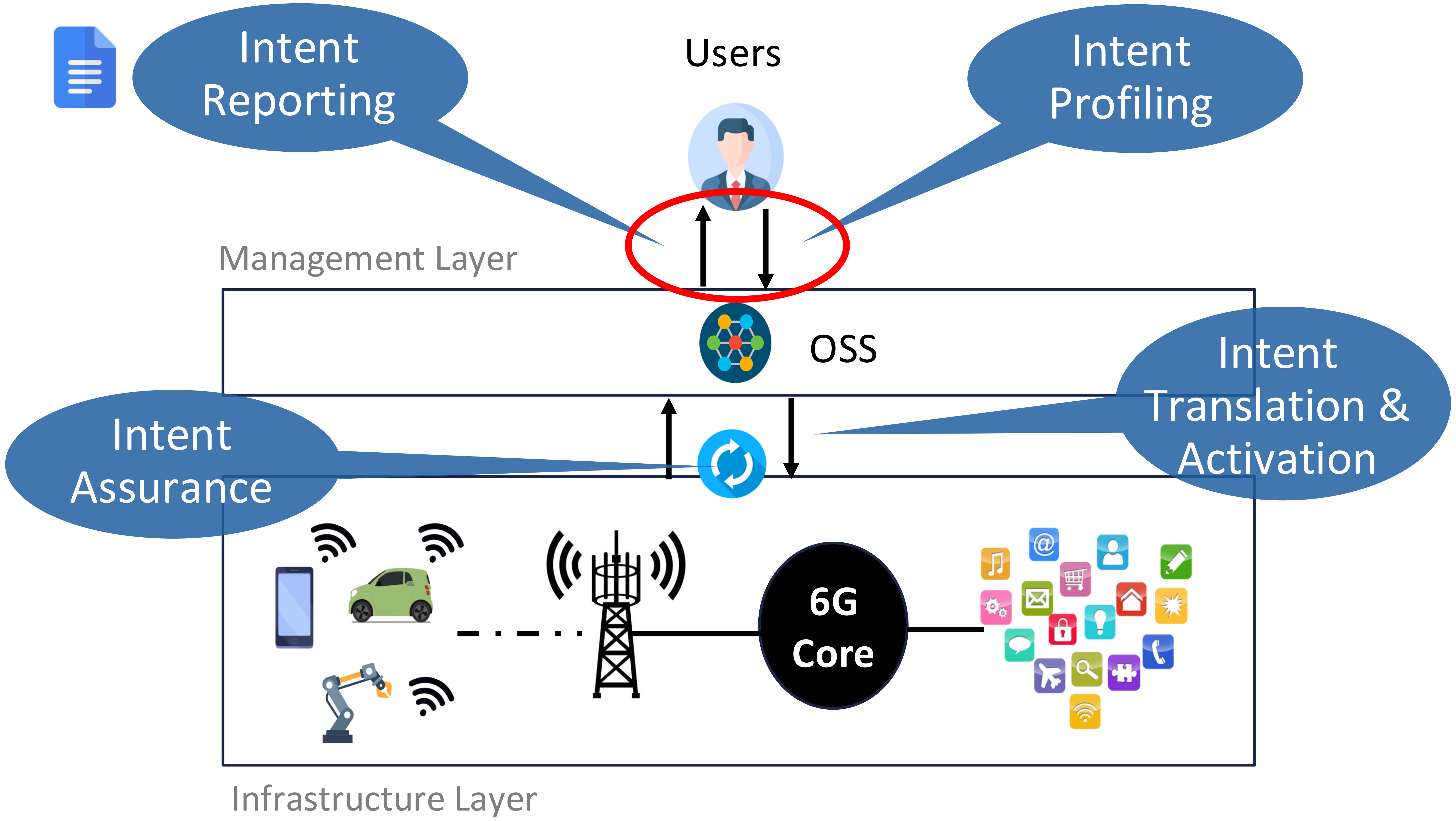


Challenges

Different standardization bodies (3GPP, TM Forum, ETSI) define heterogeneous OSS API endpoints and JSON structures.



```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```



Users must understand these APIs, which complicates intent-based orchestration

Challenges

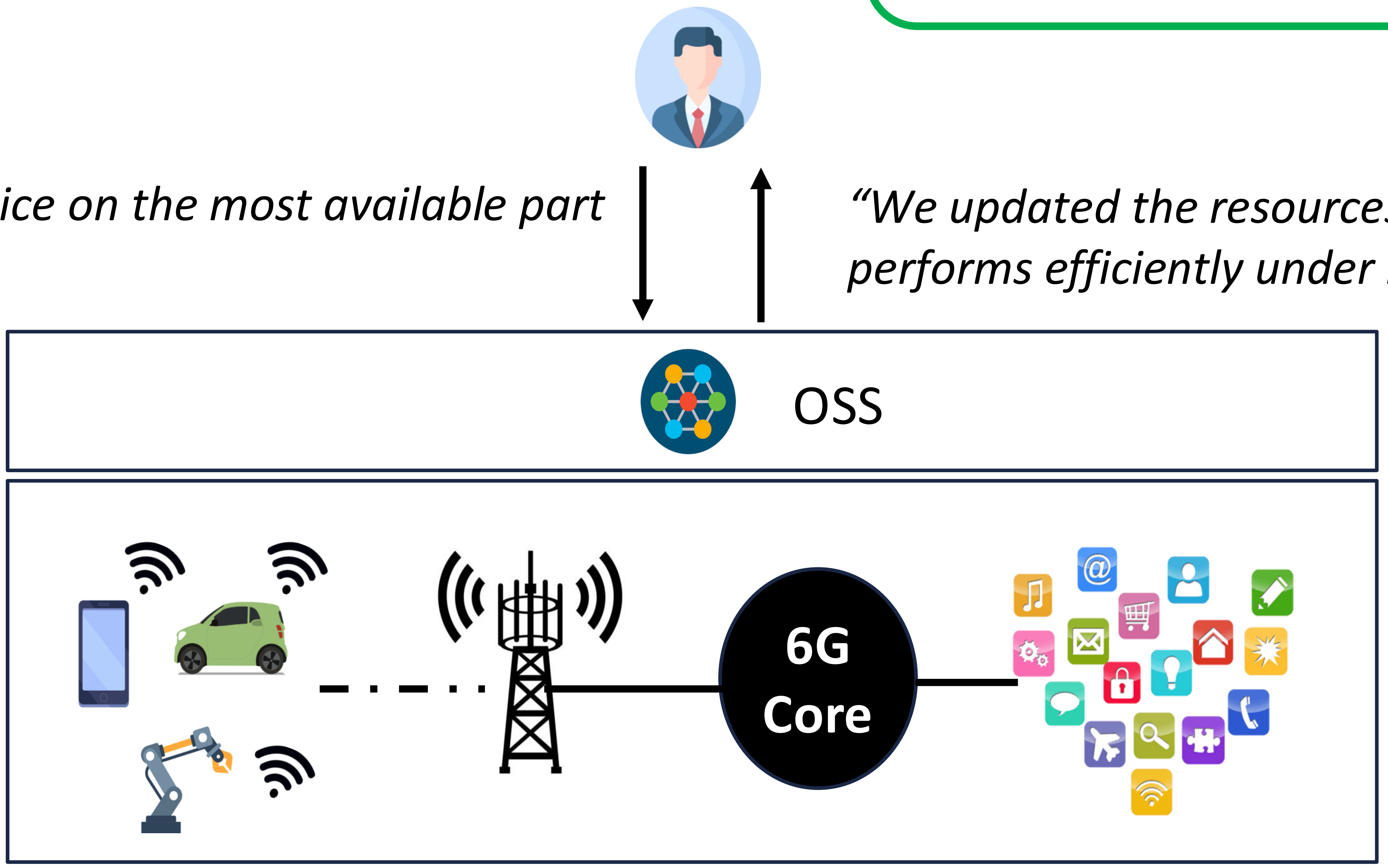
✓ Next evolution towards ‘‘Natural Language’’ intents.

- Removes API complexity.
- Allows users to manage networks, without prior knowledge.
- Enables more automated, and scalable IBN

Challenges

✓ Next evolution towards ''Natural Language'' intents.

- ○ Removes API complexity.
- ○ Allows users to manage networks, without prior knowledge.
- ○ Enables more automated, and scalable IBN



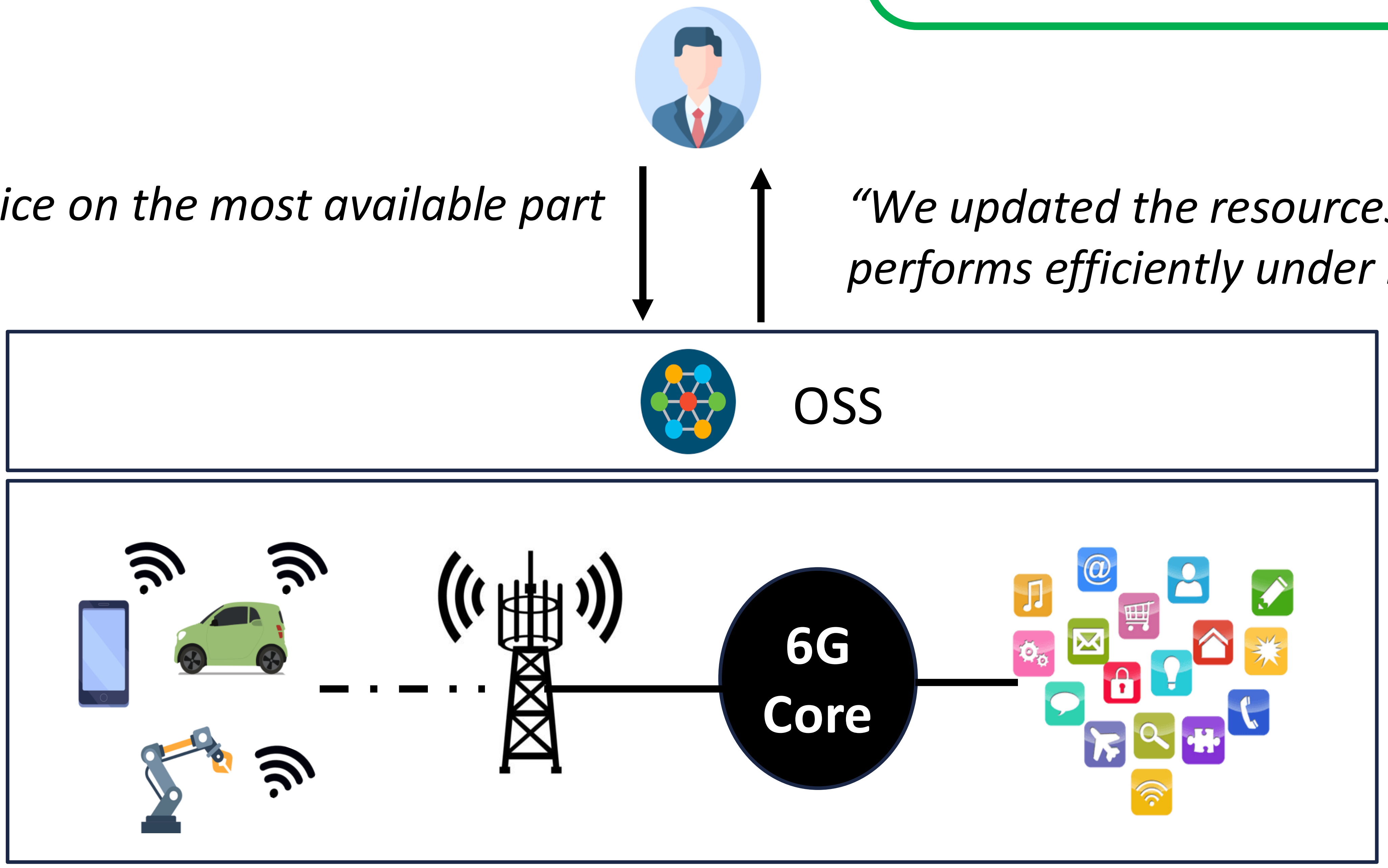
"Deploy a 5G communication service on the most available part of the virtualized infrastructure"

"We updated the resources of your XR service to ensure it performs efficiently under high user load."

Challenges

✓ Next evolution towards ''Natural Language'' intents.

- ○ Removes API complexity.
- ○ Allows users to manage networks, without prior knowledge.
- ○ Enables more automated, and scalable IBN



"Deploy a 5G communication service on the most available part of the virtualized infrastructure"

"We updated the resources of your XR service to ensure it performs efficiently under high user load."

✓ Large Language Models (LLMs) can understand, interpret, and generate natural language

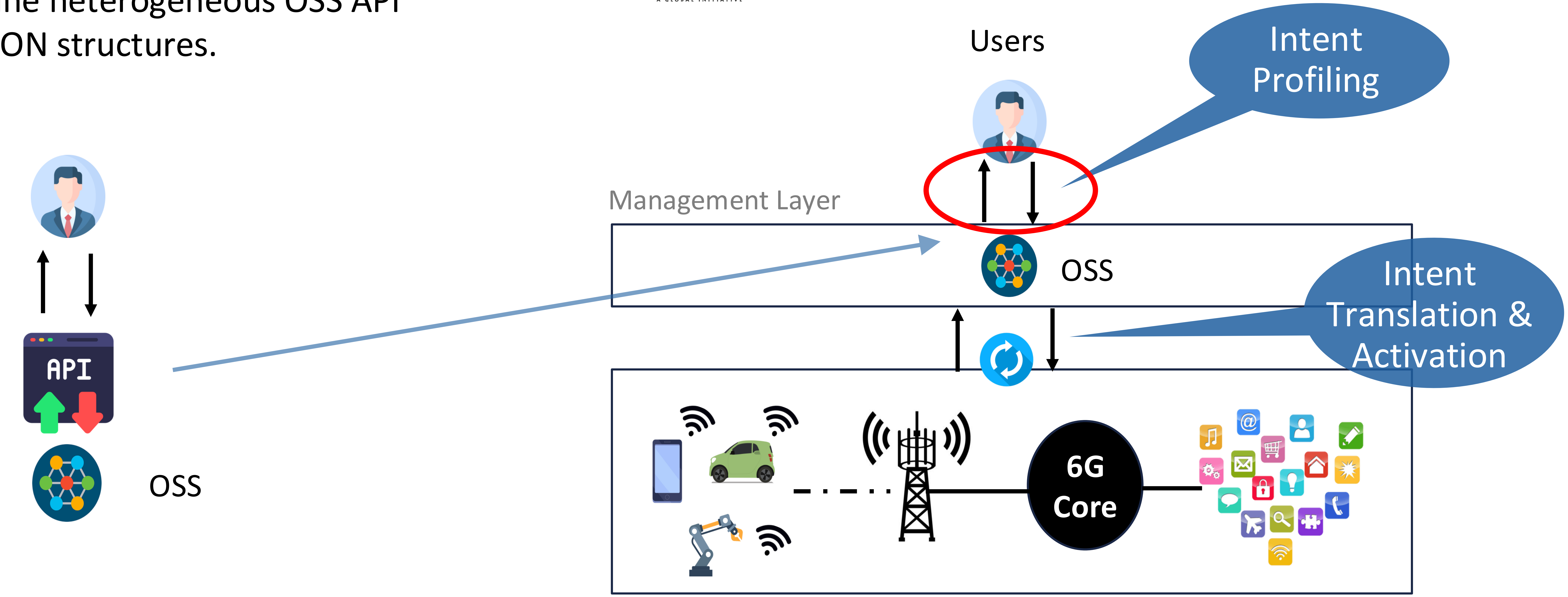
Table of Contents

- Intent-Based Networking (IBN)
- Intent Translation with LLMs
- Intent Assurance with LLMs
- Conclusion

Motivation

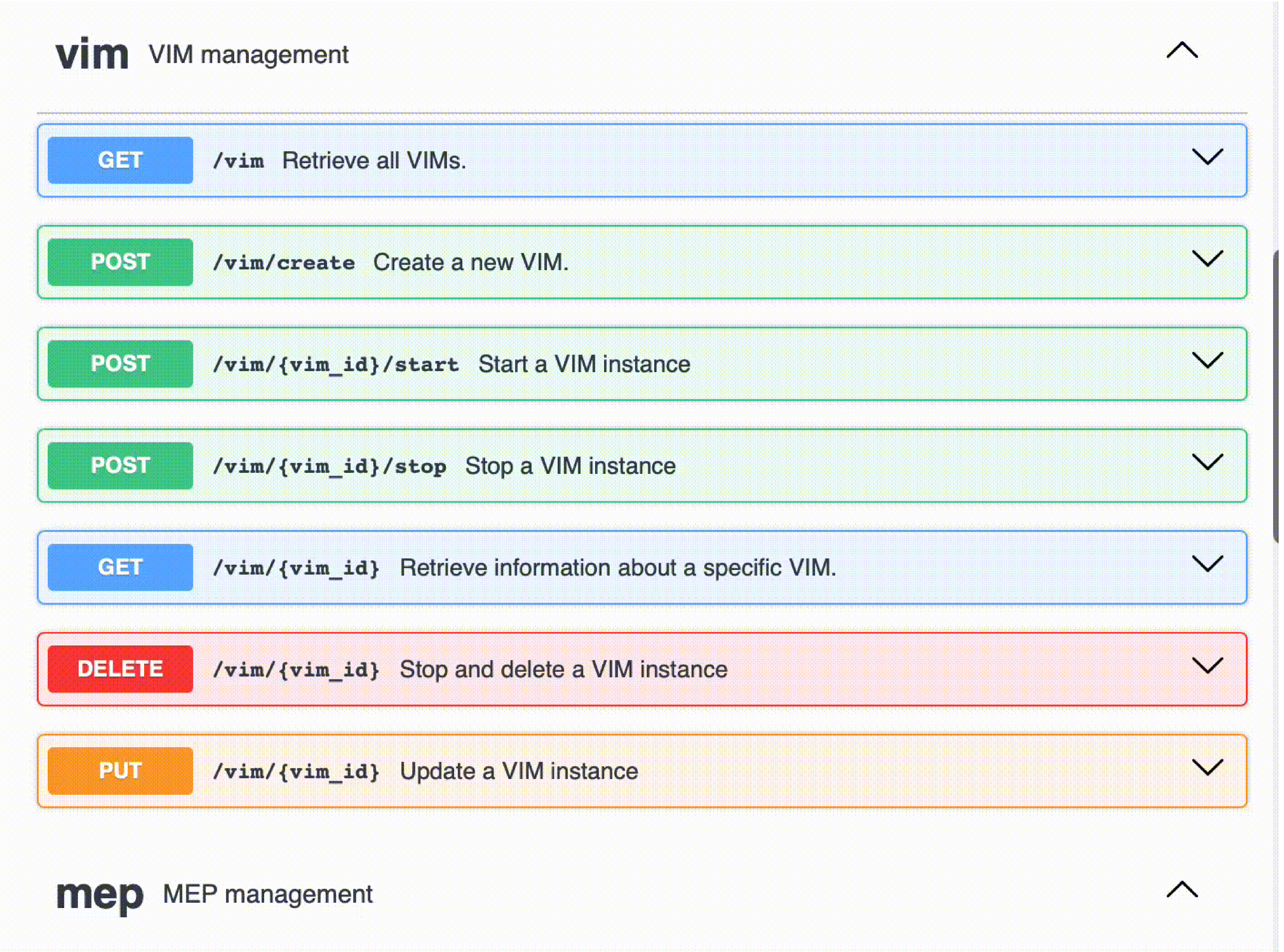
API: Application Programming Interface

Different standardization bodies (3GPP, TM Forum, ETSI) define heterogeneous OSS API endpoints and JSON structures.



Motivation

API: Application Programming Interface



vim VIM management

GET /vim Retrieve all VIMs.

POST /vim/create Create a new VIM.

POST /vim/{vim_id}/start Start a VIM instance

POST /vim/{vim_id}/stop Stop a VIM instance

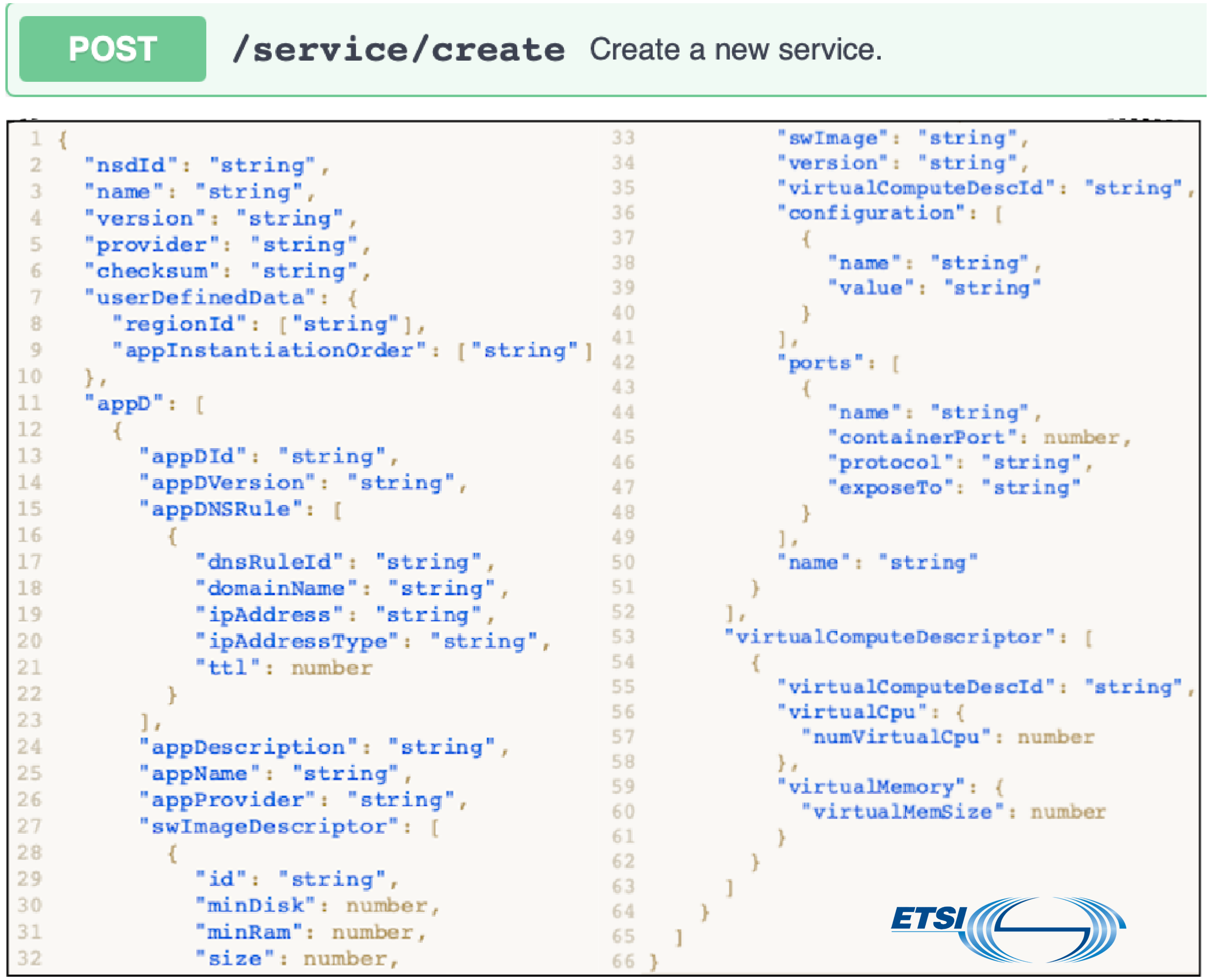
GET /vim/{vim_id} Retrieve information about a specific VIM.

DELETE /vim/{vim_id} Stop and delete a VIM instance

PUT /vim/{vim_id} Update a VIM instance

mep MEP management

OSS API endpoints



POST /service/create Create a new service.

```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

OSS “Create Service” endpoint with the Network Service Descriptor (NSD) as the JSON body

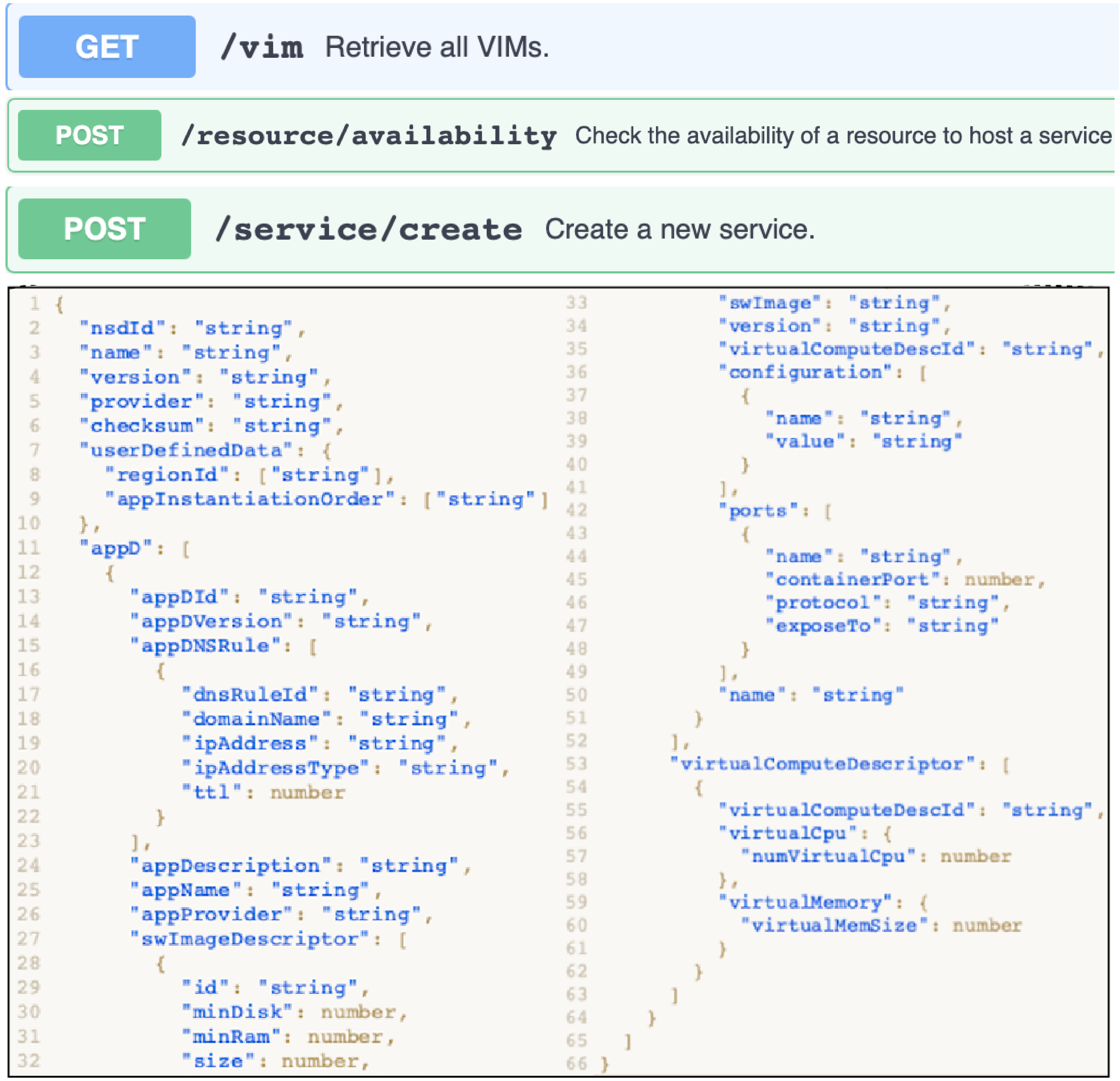
Motivation

API: Application Programming Interface

OSS APIs have many endpoints, each for a different functionality:

- Get infrastructures
- Check resources
- Create a service

Each endpoint has diverse JSON structures depending on the standard: 3GPP, TM Forum, ETSI



GET /vim Retrieve all VIMs.

POST /resource/availability Check the availability of a resource to host a service

POST /service/create Create a new service.

```json
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

Motivation

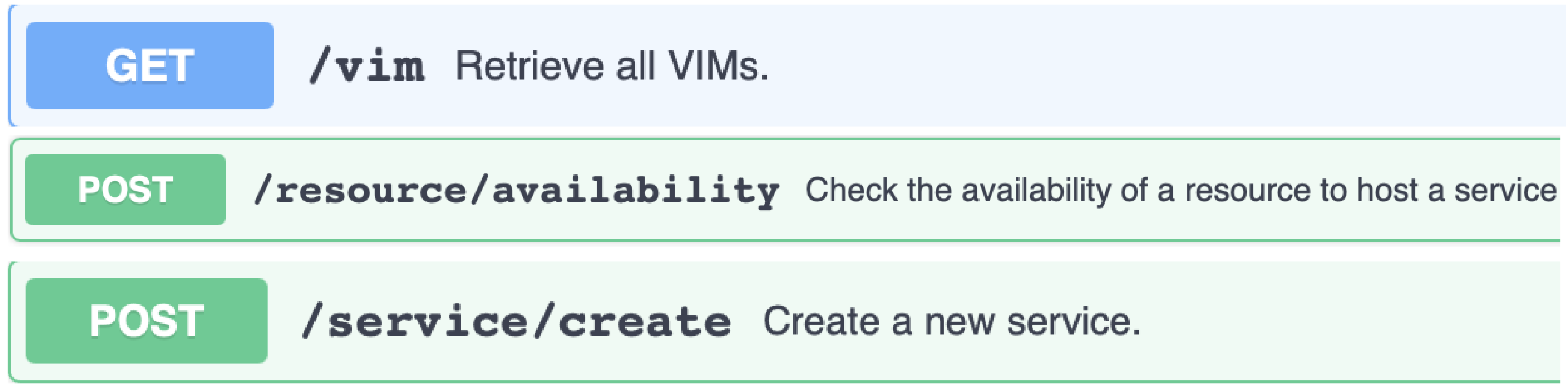
API: Application Programming Interface

OSS APIs have many endpoints, each for a different functionality:

- Get infrastructures
- Check resources
- Create a service

Each endpoint has diverse JSON structures depending on the standard: 3GPP, TM Forum, ETSI

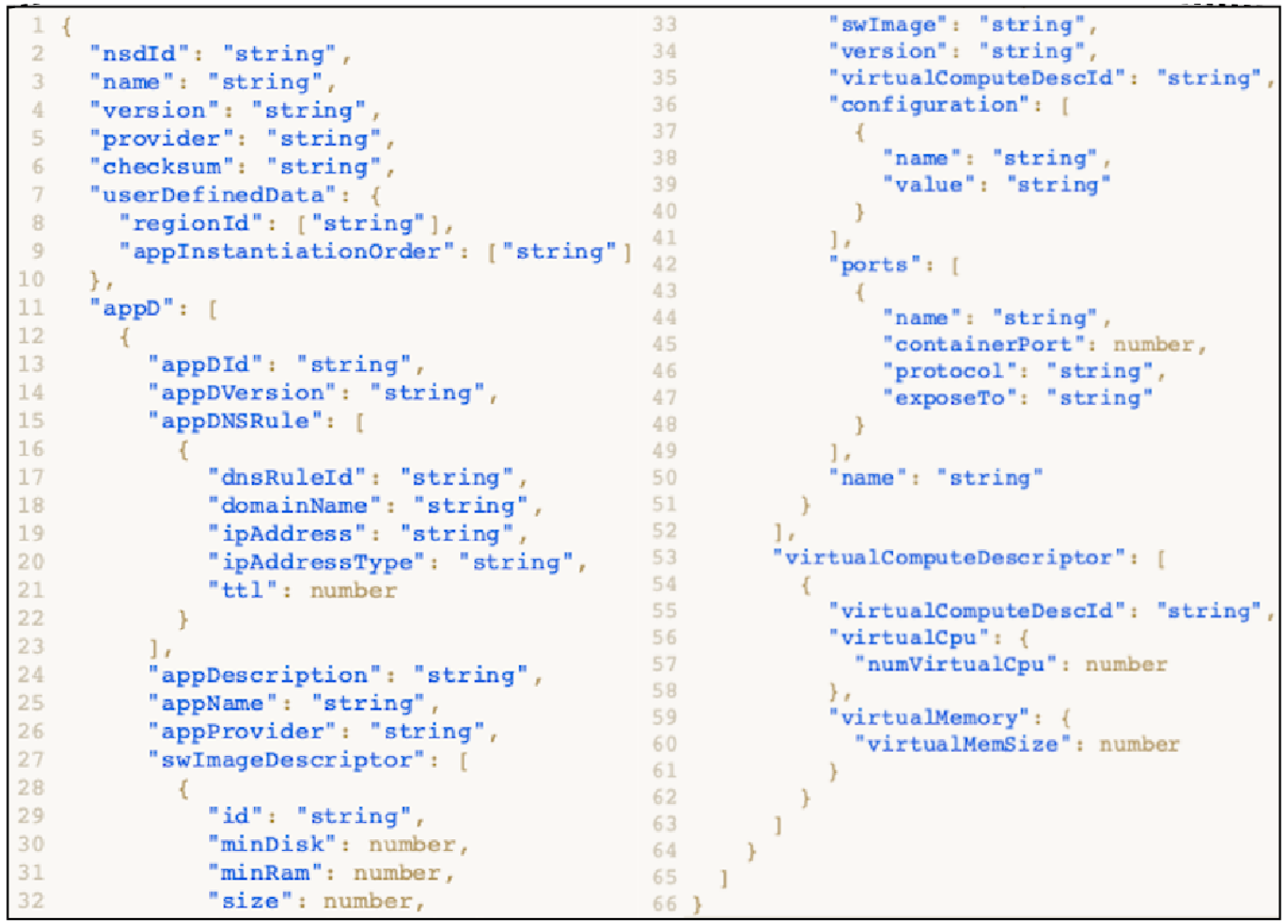
Users must understand endpoints and their JSON bodies, which complicates IBN adoption.



GET /vim Retrieve all VIMs.

POST /resource/availability Check the availability of a resource to host a service

POST /service/create Create a new service.

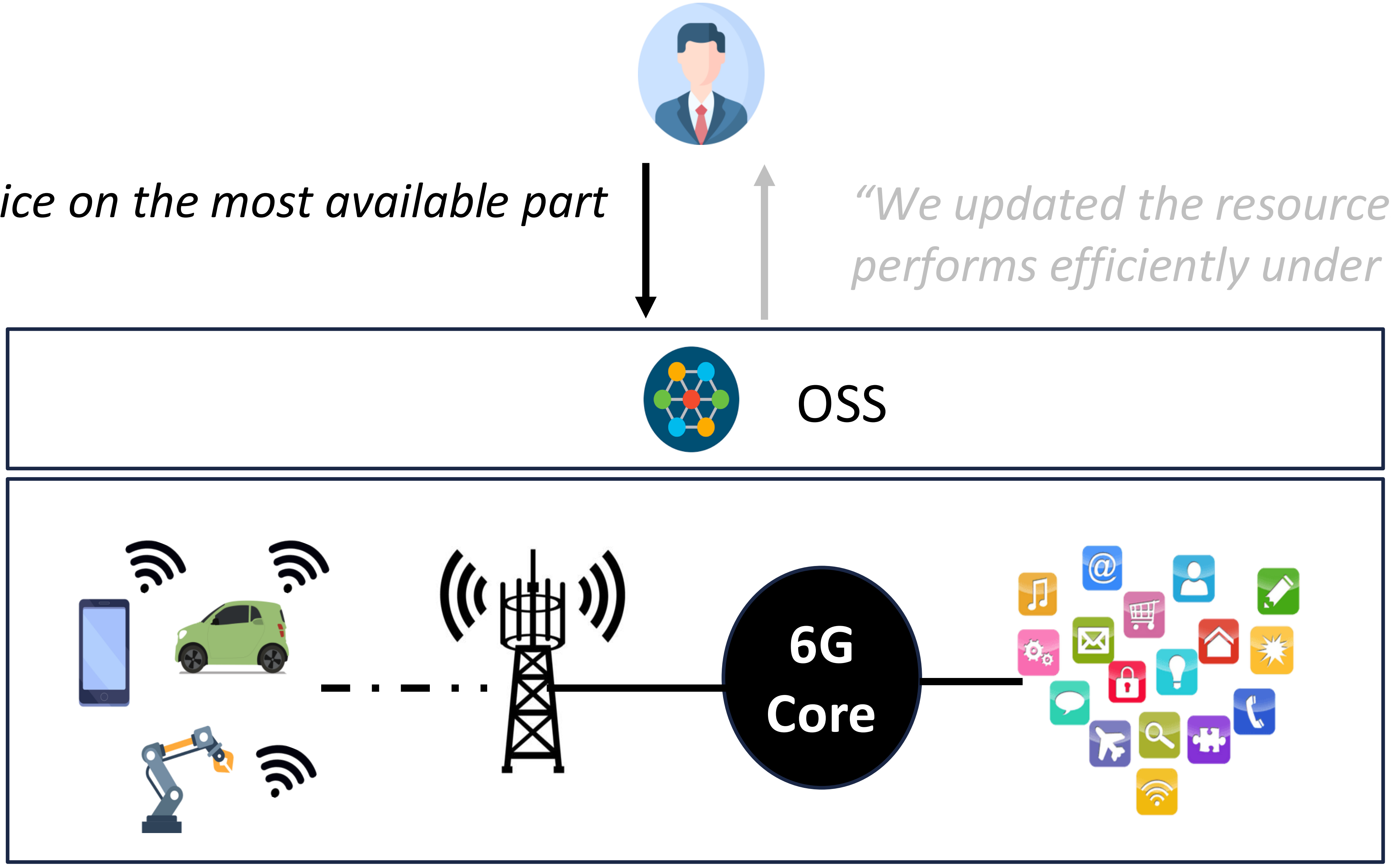


```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

Motivation

✓ Moving towards using natural language to express intents

“Deploy a 5G communication service on the most available part of the virtualized infrastructure”



“We updated the resources of your XR service to ensure it performs efficiently under high user load.”

Motivation

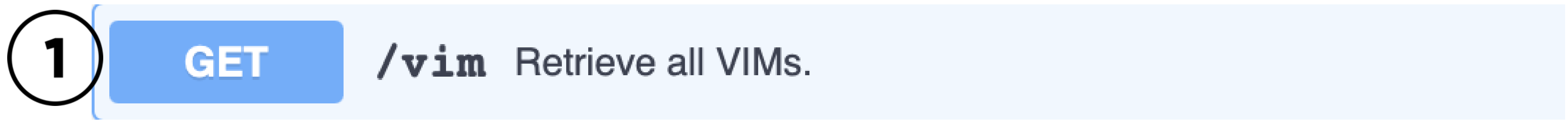
API: Application Programming Interface

"Deploy a 5G communication service on the most available part of the virtualized infrastructure"

Motivation

API: Application Programming Interface

1

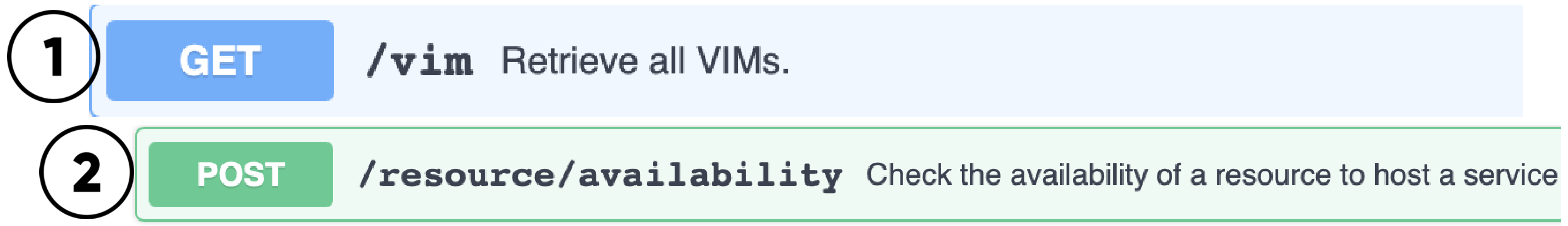


GET /vim Retrieve all VIMs.

“Deploy a 5G communication service on the most available part of the virtualized infrastructure”

Motivation

API: Application Programming Interface



1 GET /vim Retrieve all VIMs.

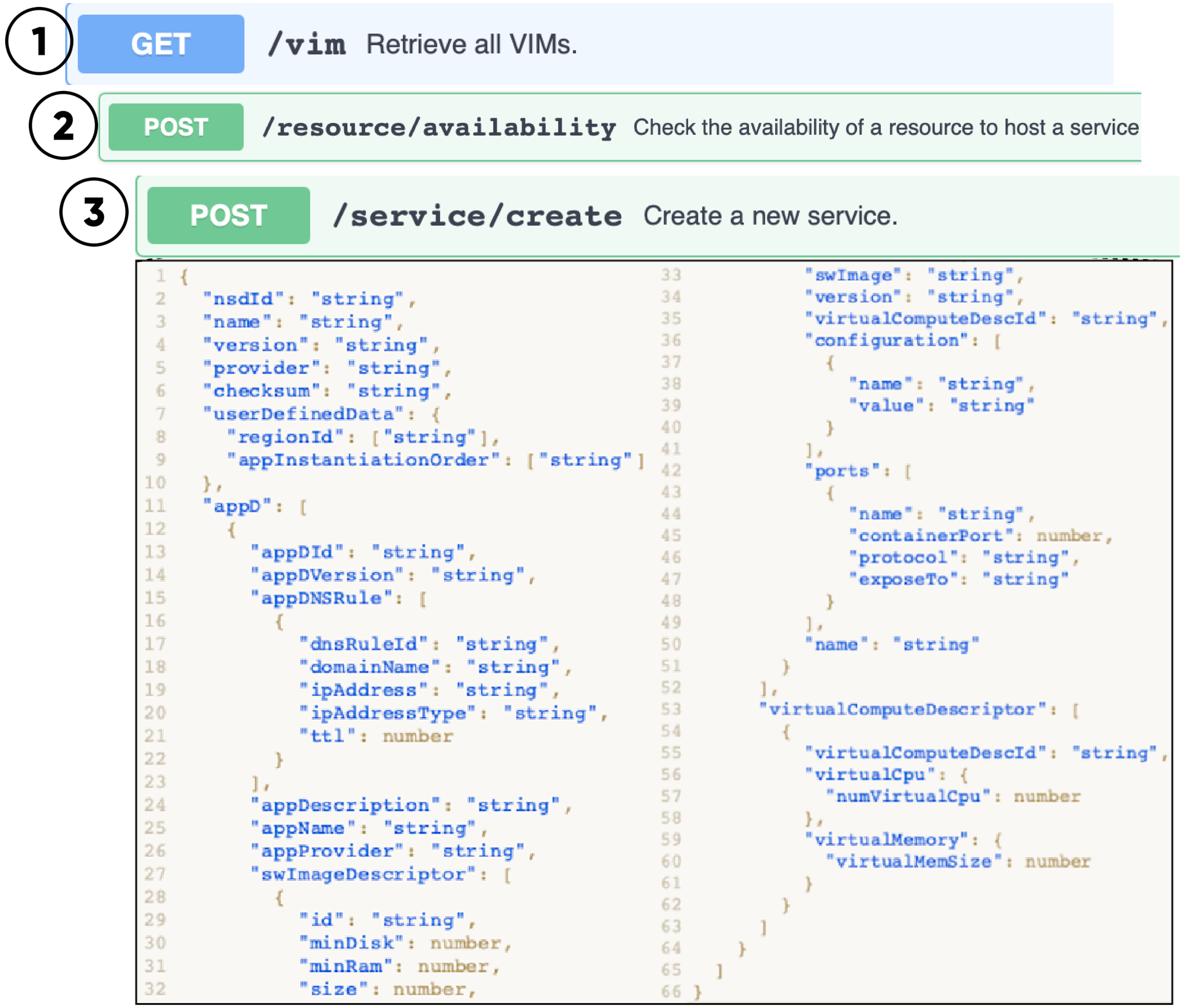
2 POST /resource/availability Check the availability of a resource to host a service

“Deploy a 5G communication service on the most available part of the virtualized infrastructure”

Motivation

API: Application Programming Interface

“Deploy a 5G communication service on the most available part of the virtualized infrastructure”



1 GET **/vim** Retrieve all VIMs.

2 POST **/resource/availability** Check the availability of a resource to host a service

3 POST **/service/create** Create a new service.

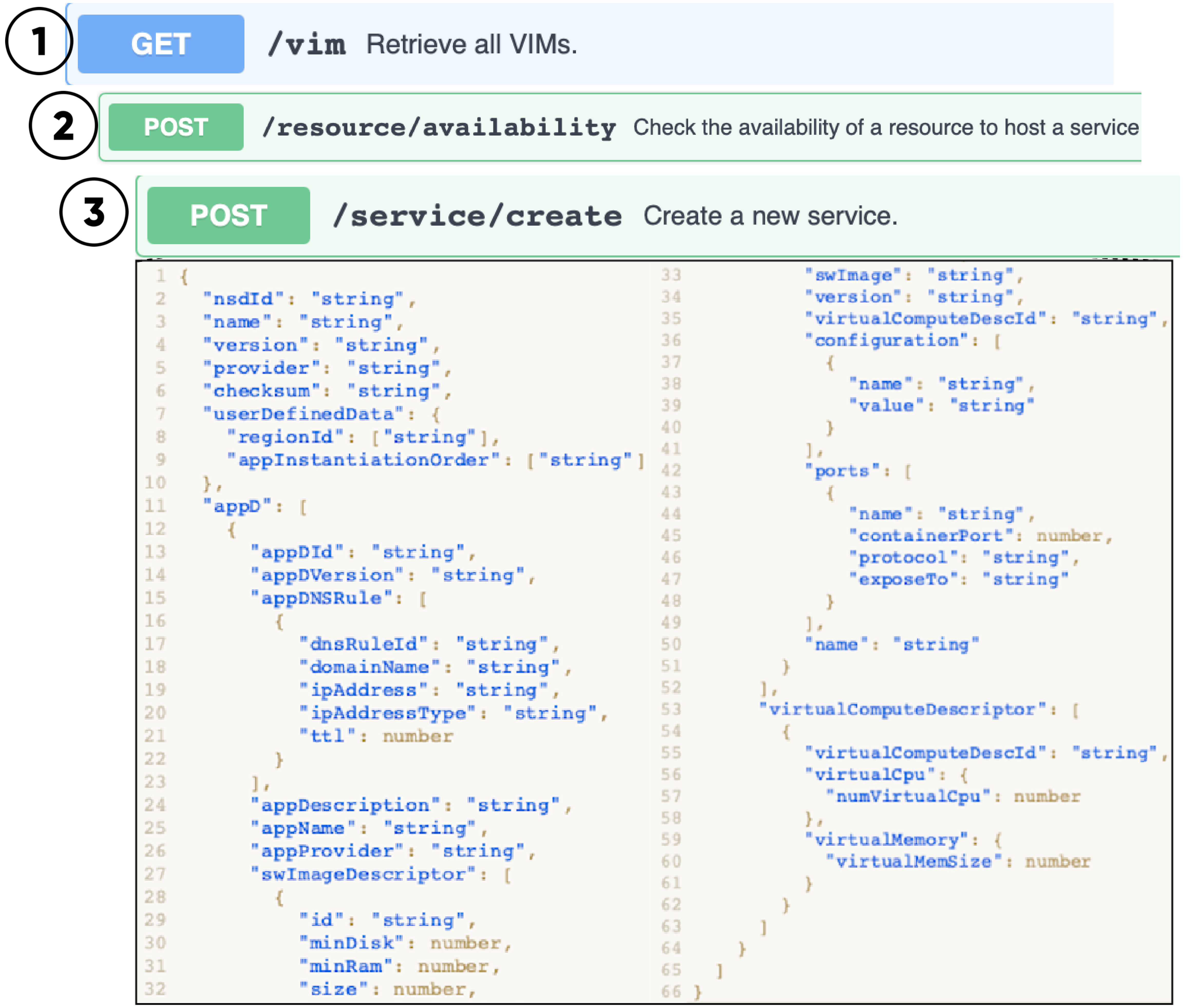
```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

Motivation

API: Application Programming Interface

Natural language Intents

“Deploy a 5G communication service on the most available part of the virtualized infrastructure”



1 GET `/vim` Retrieve all VIMs.

2 POST `/resource/availability` Check the availability of a resource to host a service

3 POST `/service/create` Create a new service.

```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

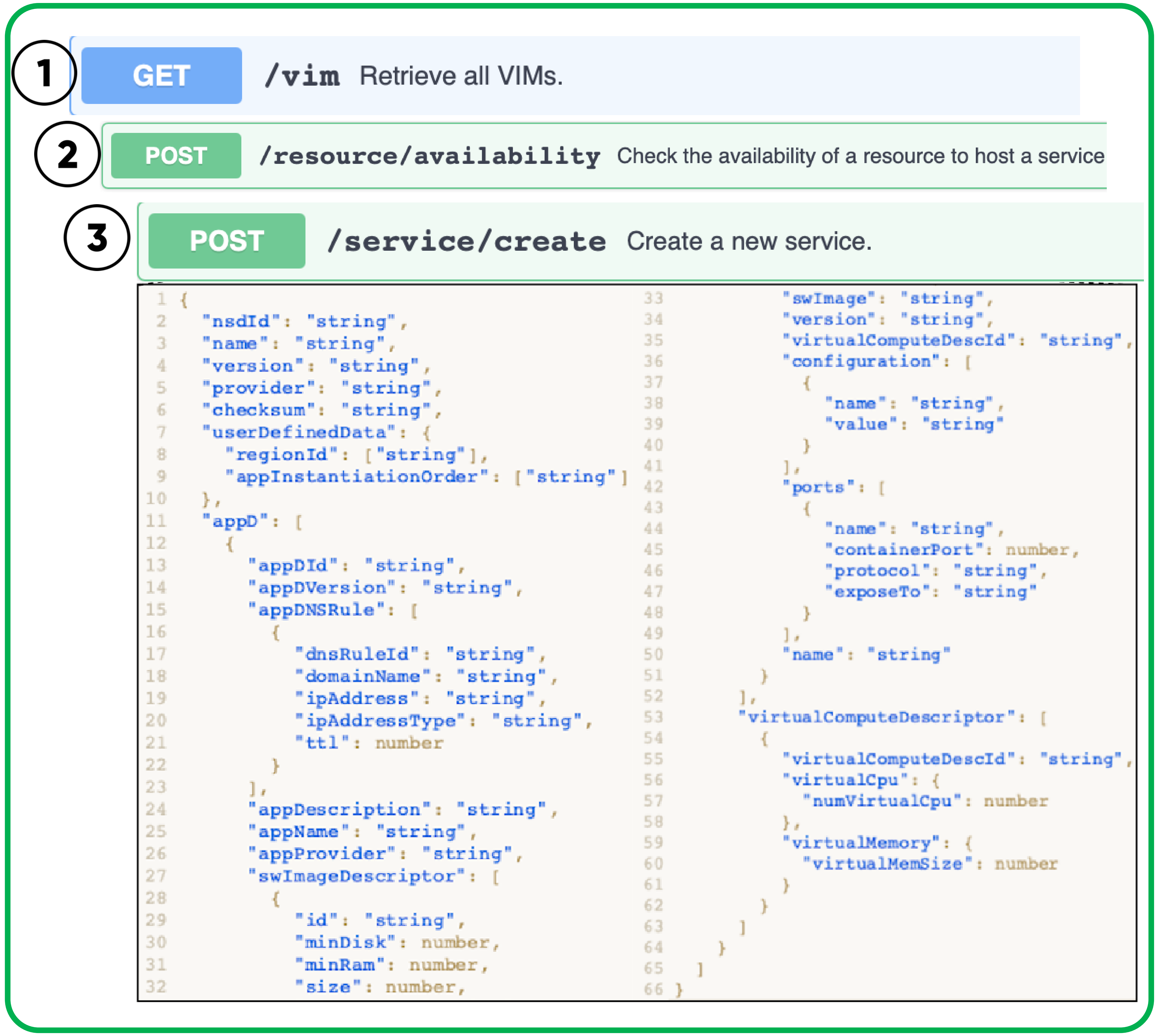
Motivation

API: Application Programming Interface

Natural language Intents

“Deploy a 5G communication service on the most available part of the virtualized infrastructure”

- ○ Multiple OSS-level API calls,
- ○ With the appropriate YAML/JSON bodies generated internally,
- ○ Execute the API calls, to fulfill the intent



1 GET `/vim` Retrieve all VIMs.

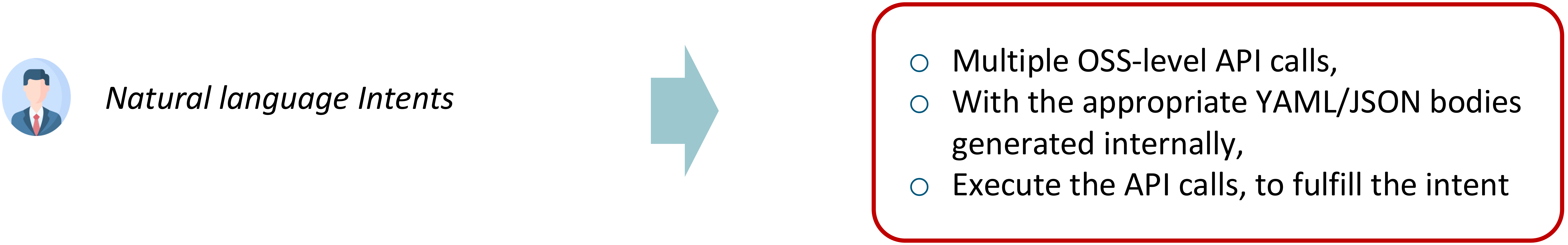
2 POST `/resource/availability` Check the availability of a resource to host a service

3 POST `/service/create` Create a new service.

```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

Approach

API: Application Programming Interface



Natural language Intents

- Multiple OSS-level API calls,
- With the appropriate YAML/JSON bodies generated internally,
- Execute the API calls, to fulfill the intent

Approach

API: Application Programming Interface

Natural language Intents

- ○ Multiple OSS-level API calls,
- ○ With the appropriate YAML/JSON bodies generated internally,
- ○ Execute the API calls, to fulfill the intent

! A complex task for one LLM agent to handle

✓ Multiple LLM agents (Agentic AI), each responsible for a specific role

Approach

API: Application Programming Interface

Natural language Intents

- ○ Multiple OSS-level API calls,
- ○ With the appropriate YAML/JSON bodies generated internally,
- ○ Execute the API calls, to fulfill the intent

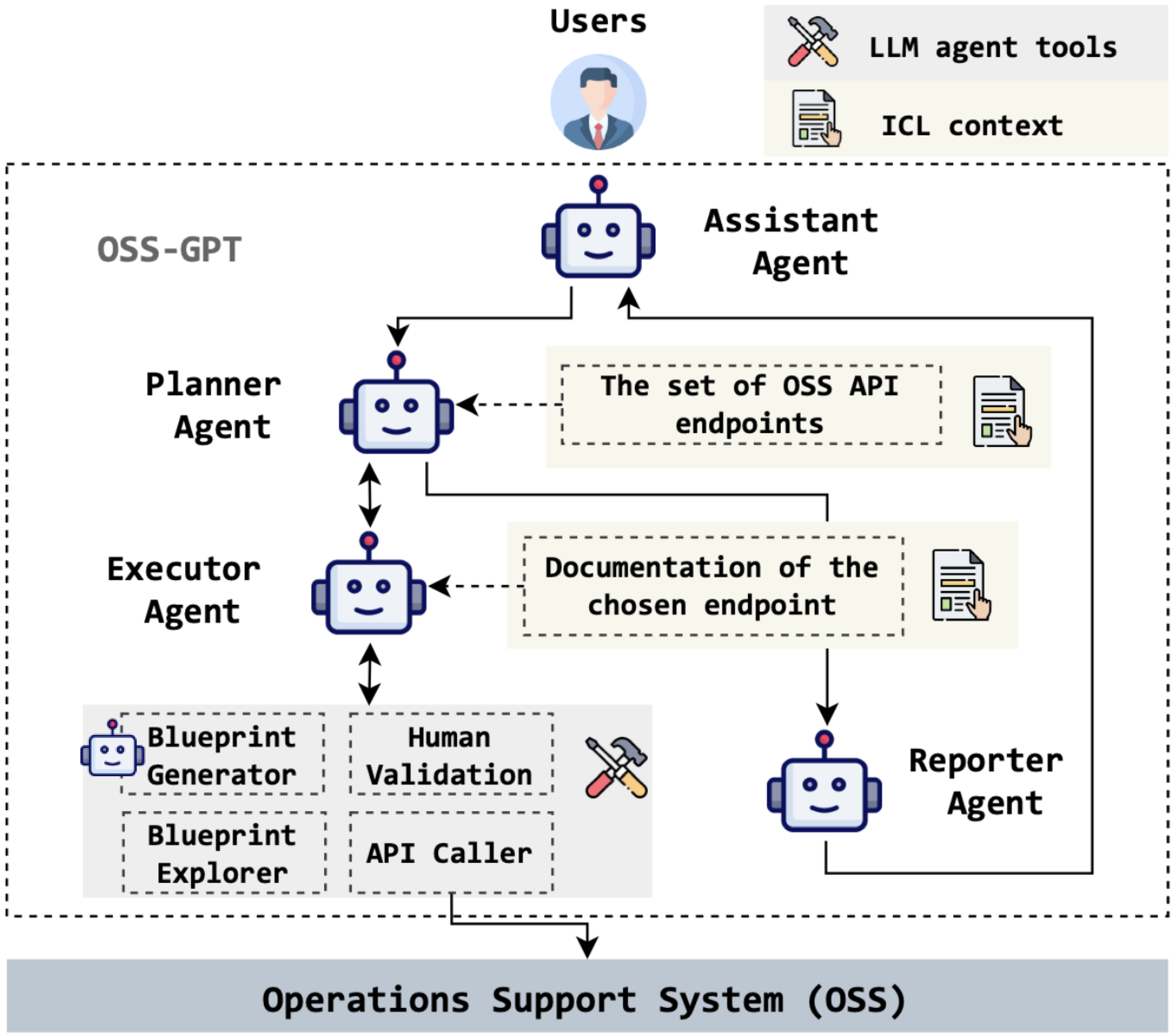
A complex task for one LLM agent to handle

✓ Multiple LLM agents (Agentic AI), each responsible for a specific role

To address this, we introduced **OSS-GPT**:

- ○ An agentic AI framework that translates natural language intents into multiple API calls.
- ○ Generates their request bodies (JSON), and executes them.

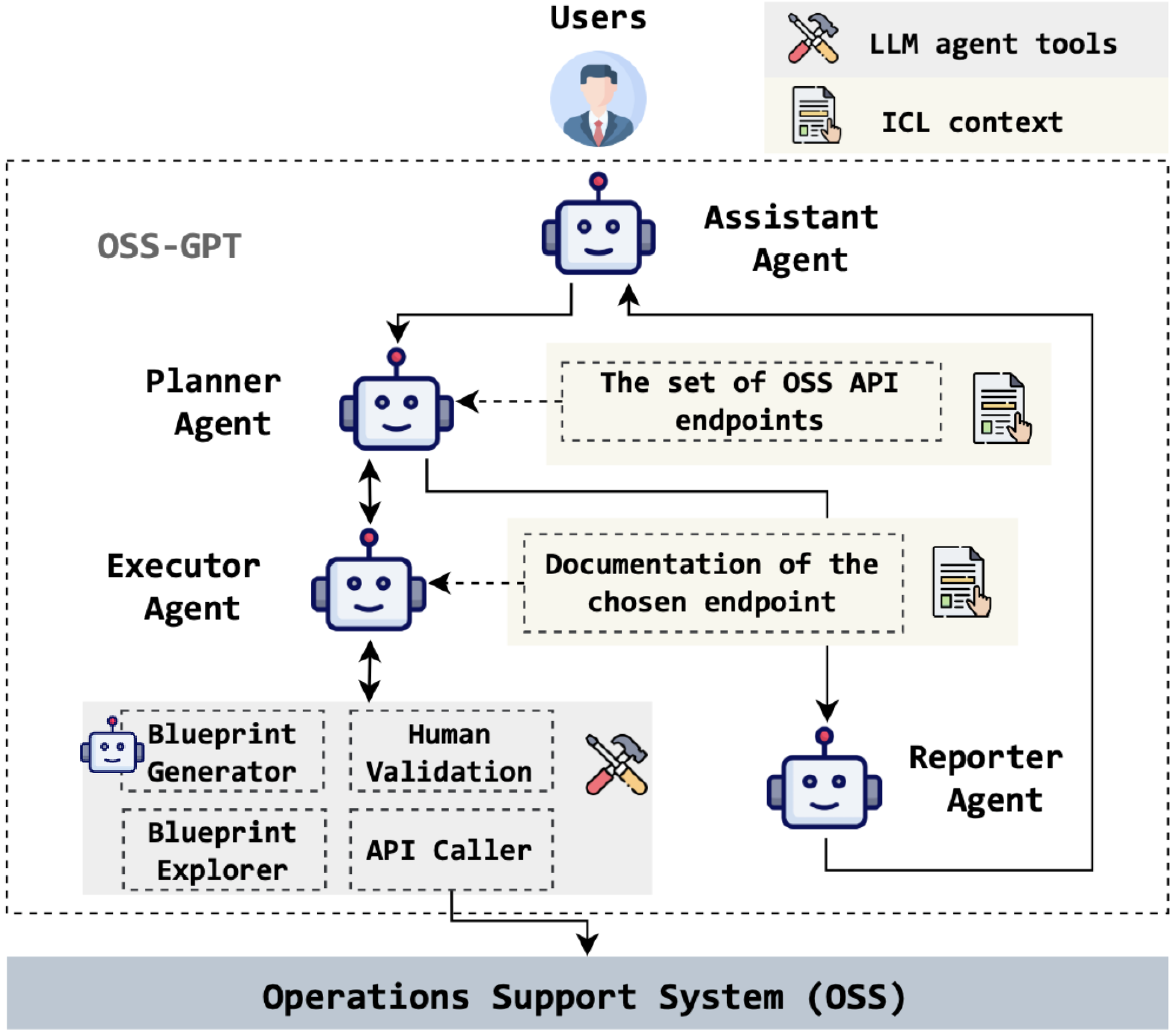
Approach



Users
LLM agent tools
ICL context
OSS-GPT
Assistant Agent
Planner Agent
The set of OSS API endpoints
Executor Agent
Documentation of the chosen endpoint
Blueprint Generator
Human Validation
Blueprint Explorer
API Caller
Reporter Agent
Operations Support System (OSS)

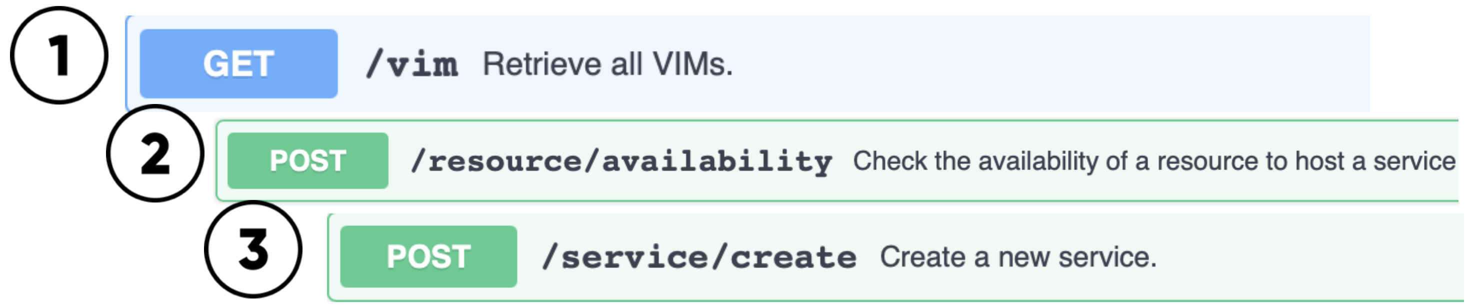
Approach

- Assistant: will interact with the users using natural language.



Approach

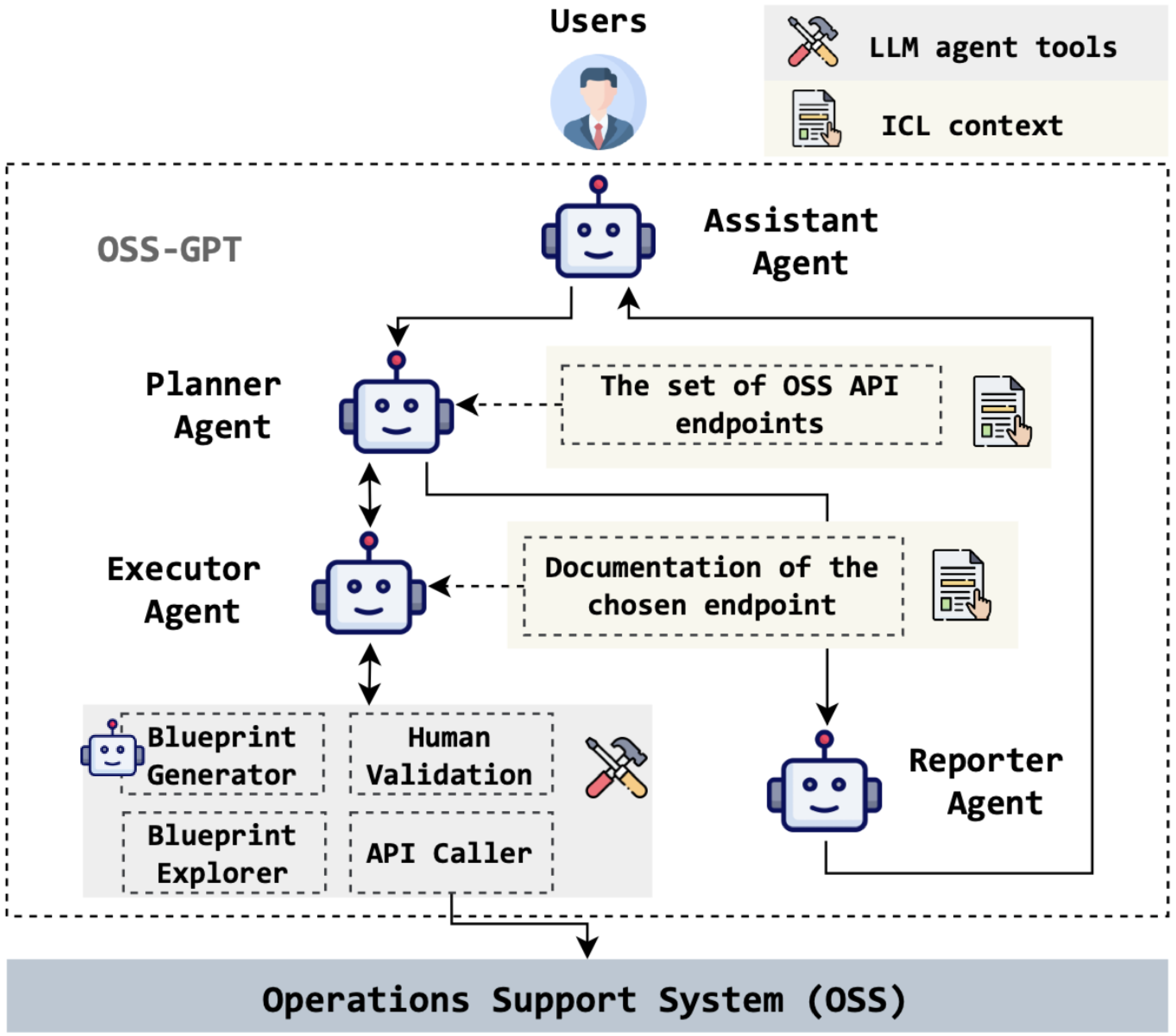
- Assistant: will interact with the users using natural language.
- Planner: will plan the set of API calls to fulfill the intent.



1 GET /vim Retrieve all VIMs.

2 POST /resource/availability Check the availability of a resource to host a service

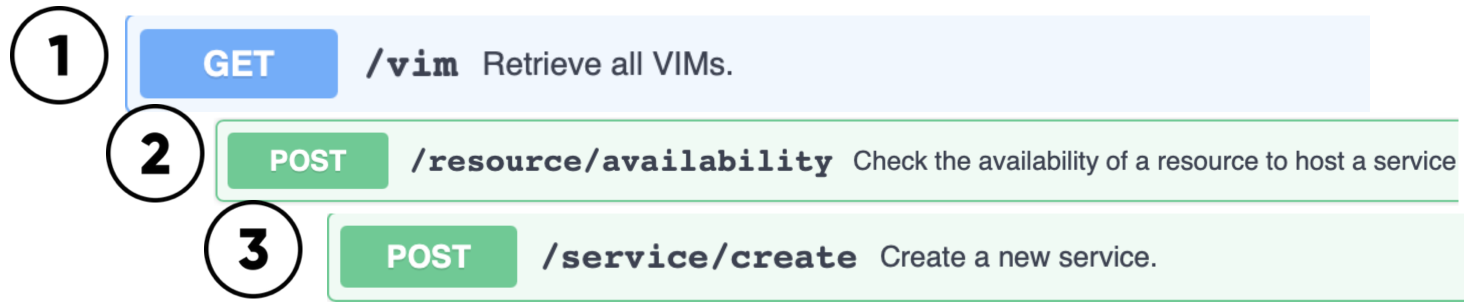
3 POST /service/create Create a new service.



Approach

- Assistant: will interact with the users using natural language.

- Planner: will plan the set of API calls to fulfill the intent.

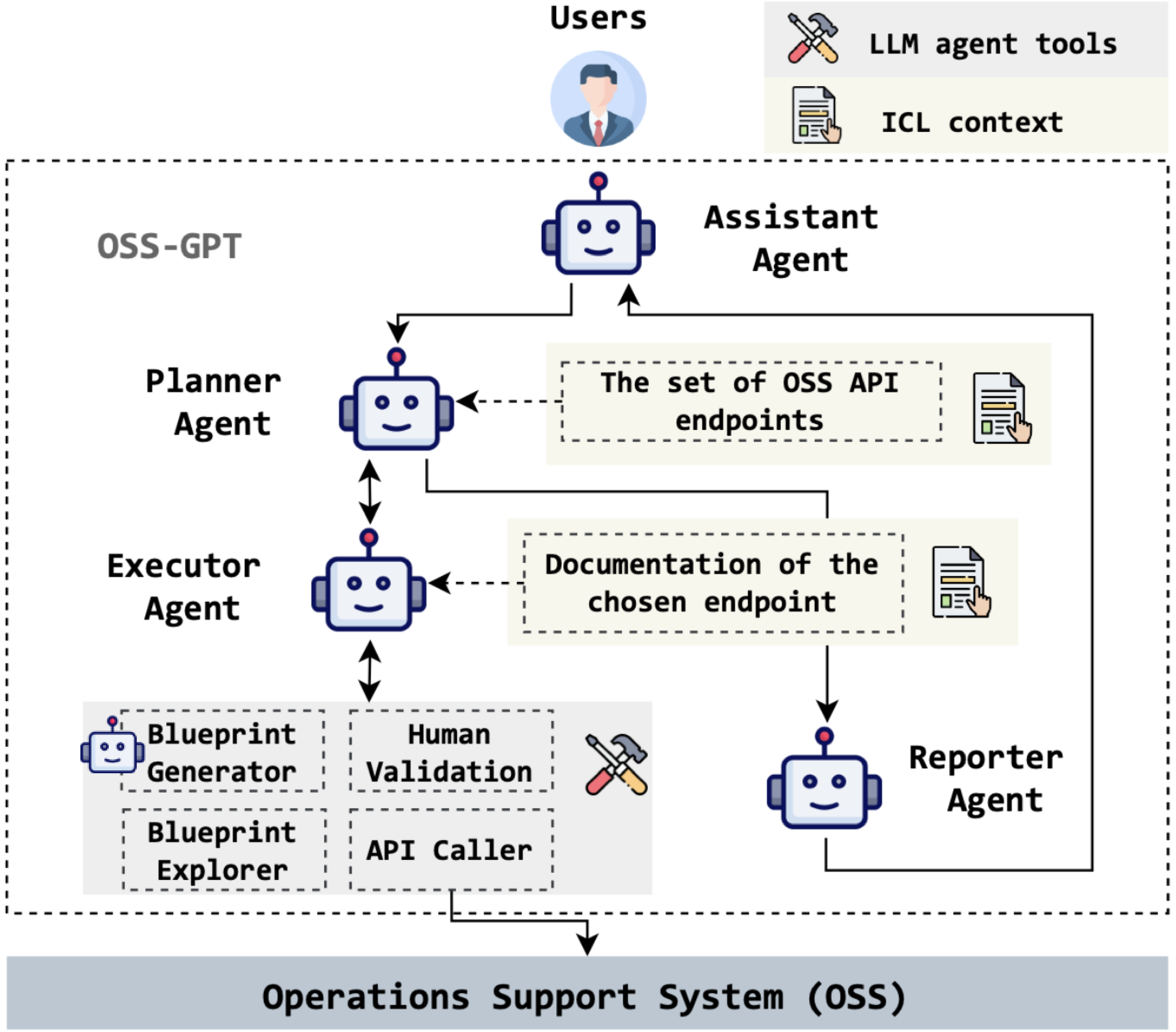


1 GET /vim Retrieve all VIMs.

2 POST /resource/availability Check the availability of a resource to host a service

3 POST /service/create Create a new service.

- Executor: will execute each API calls, using tools.



Users

LLM agent tools

ICL context

OSS-GPT

Assistant Agent

Planner Agent

The set of OSS API endpoints

Executor Agent

Documentation of the chosen endpoint

Blueprint Generator

Human Validation

Blueprint Explorer

API Caller

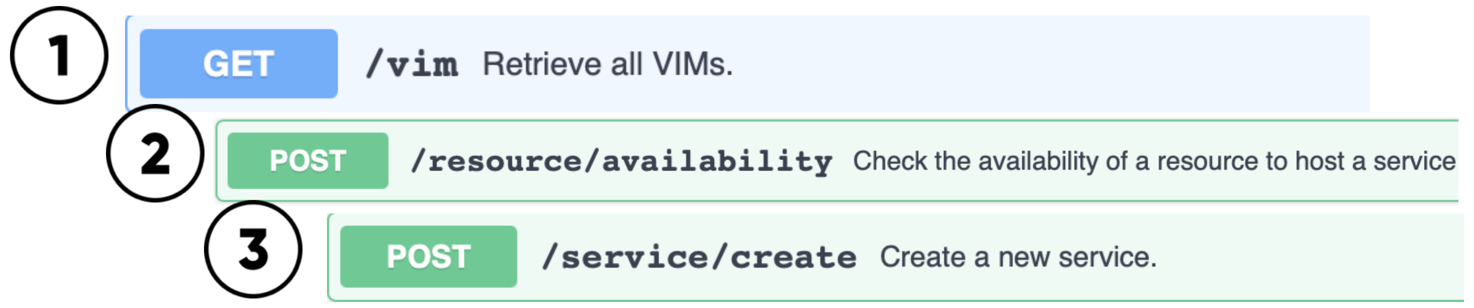
Reporter Agent

Operations Support System (OSS)

Approach

- Assistant: will interact with the users using natural language.

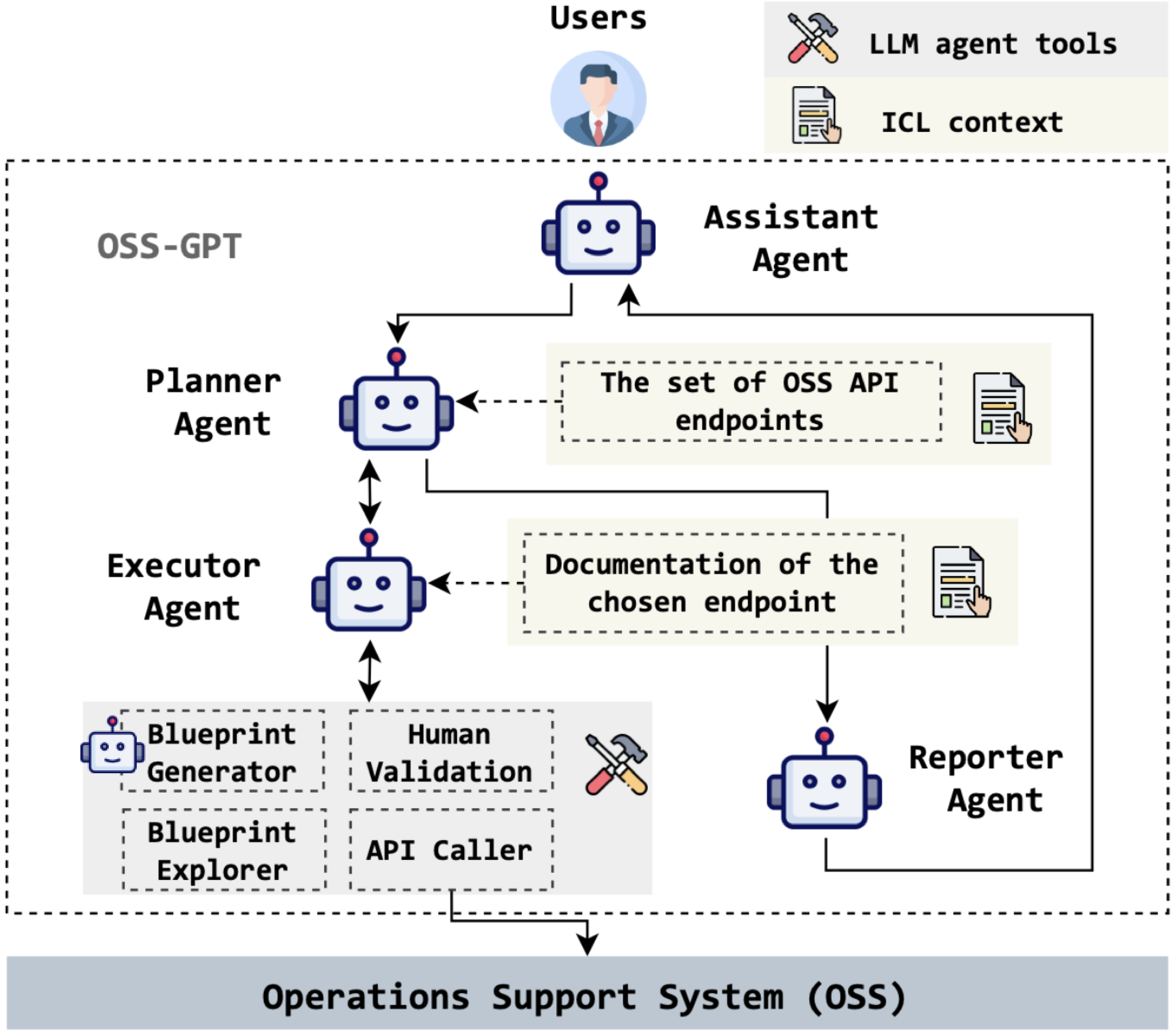
- Planner: will plan the set of API calls to fulfill the intent.



1. GET /vim Retrieve all VIMs.
2. POST /resource/availability Check the availability of a resource to host a service
3. POST /service/create Create a new service.

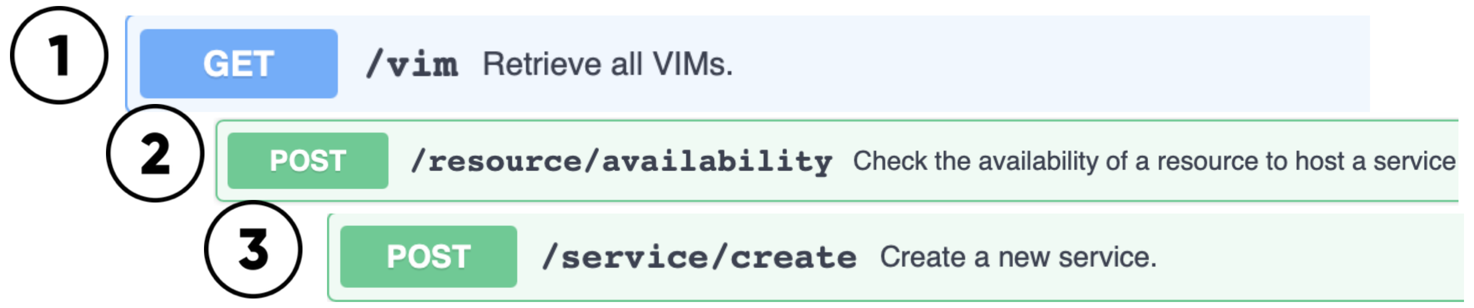
- Executor: will execute each API calls, using tools.

- Reporter: report the results to the user using natural language.



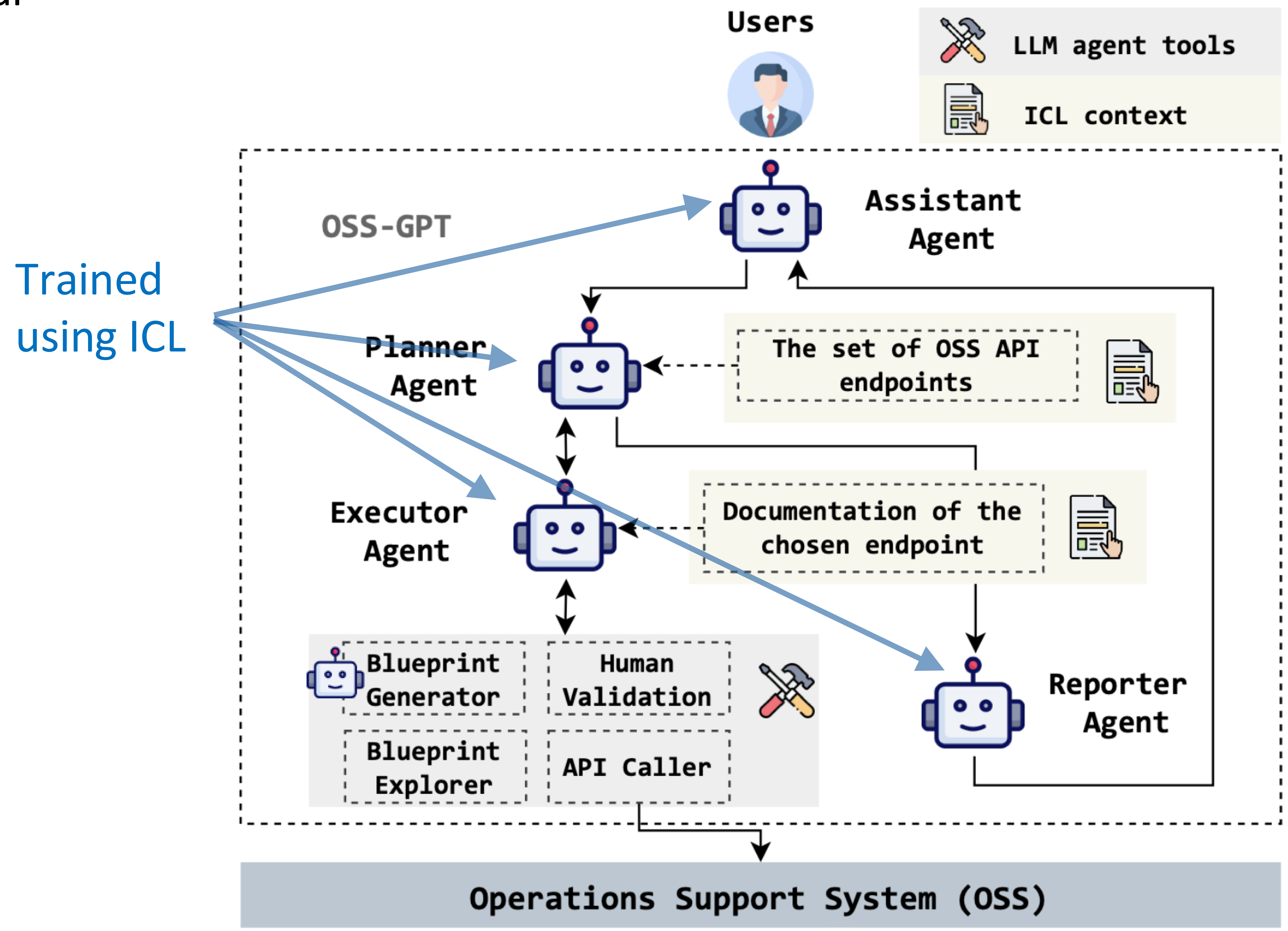
Approach

- Assistant: will interact with the users using natural language.
- Planner: will plan the set of API calls to fulfill the intent.



1. GET `/vim` Retrieve all VIMs.
2. POST `/resource/availability` Check the availability of a resource to host a service
3. POST `/service/create` Create a new service.

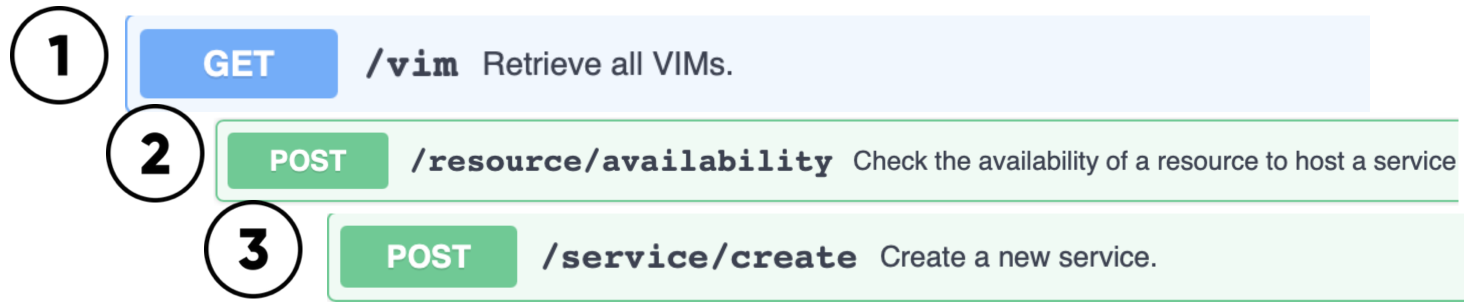
- Executor: will execute each API calls, using tools.
- Reporter: report the results to the user using natural language.



Trained using ICL

Approach

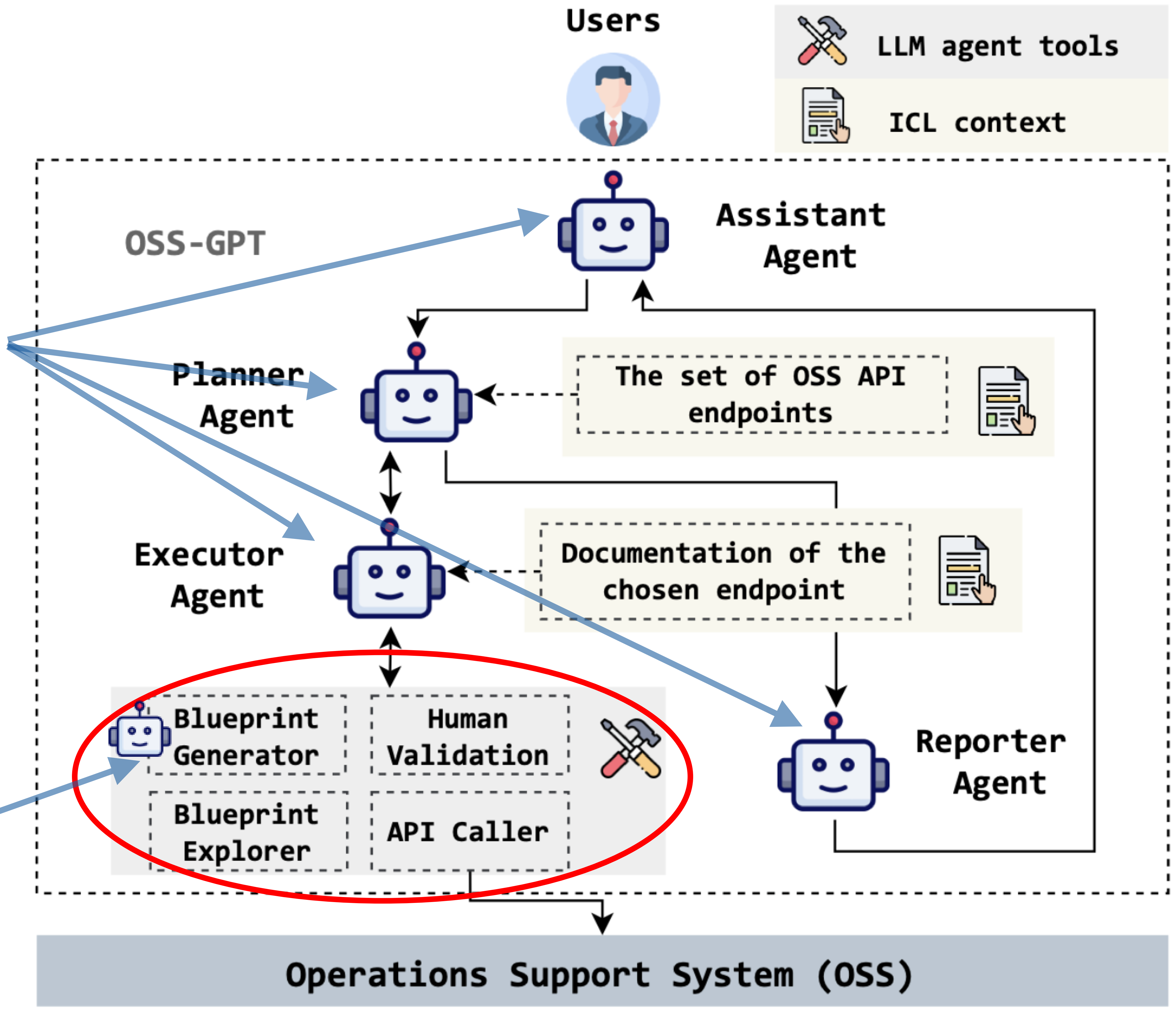
- Assistant: will interact with the users using natural language.
- Planner: will plan the set of API calls to fulfill the intent.

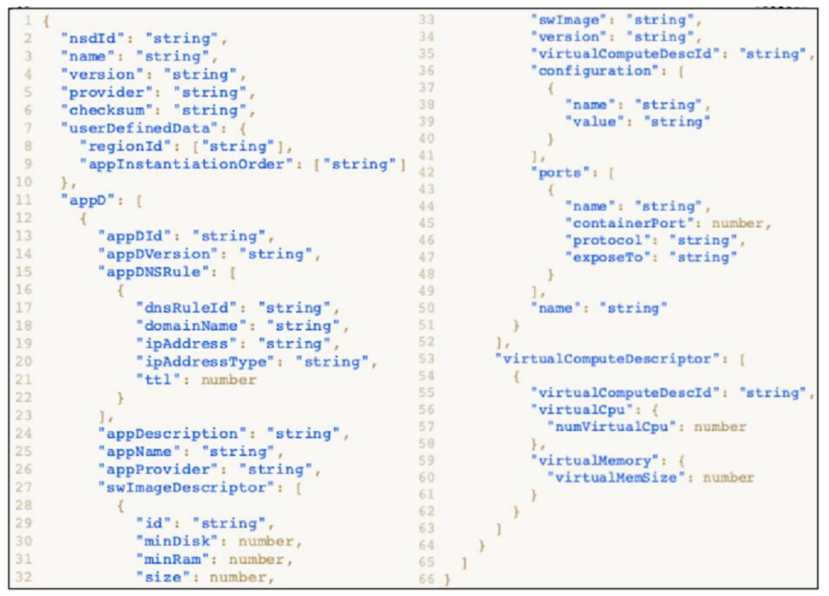


1. GET /vim Retrieve all VIMs.
2. POST /resource/availability Check the availability of a resource to host a service
3. POST /service/create Create a new service.

- Executor: will execute each API calls, using tools.
- Reporter: report the results to the user using natural language.

Trained using ICL



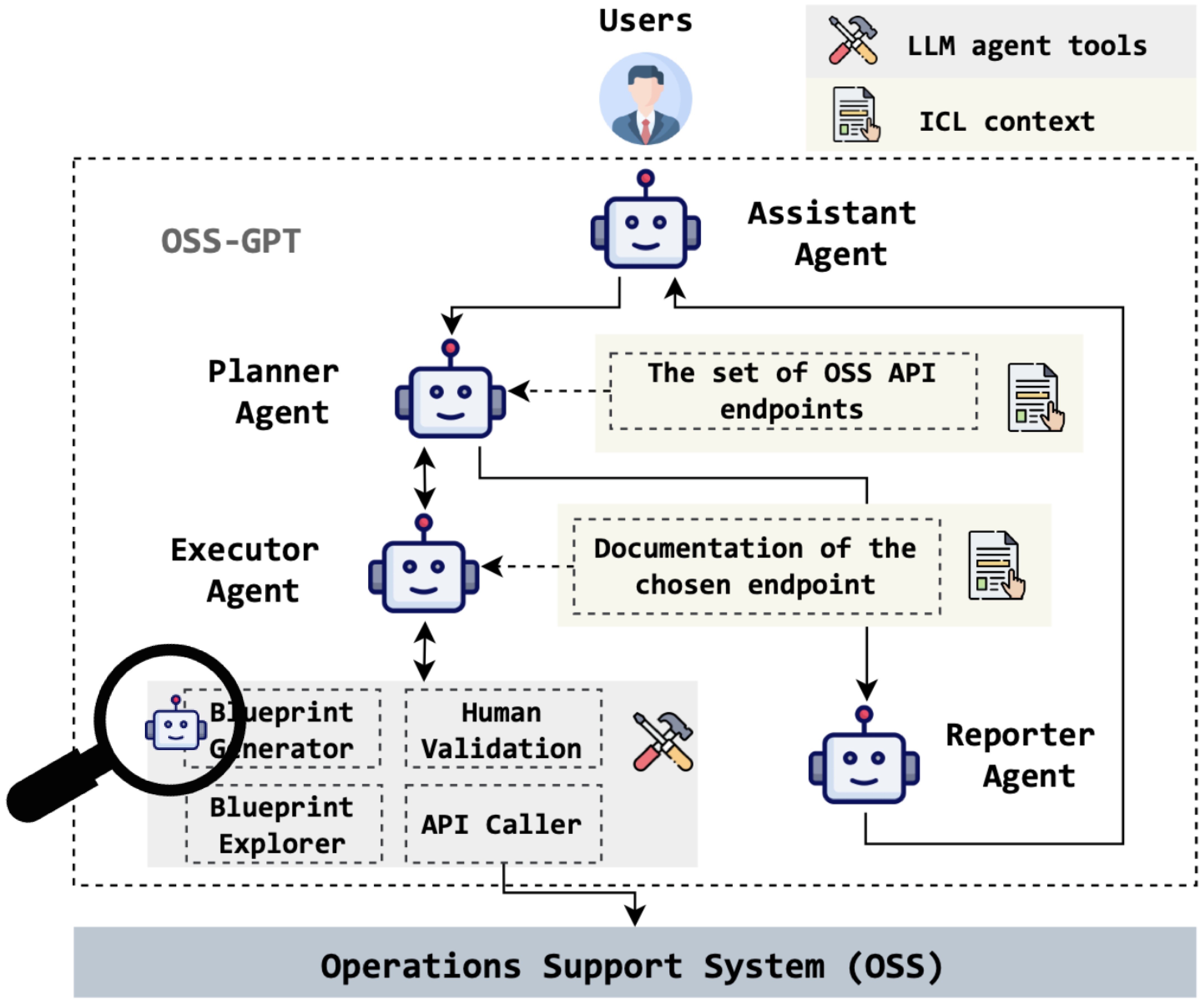


```
{
  "nsdId": "string",
  "name": "string",
  "version": "string",
  "provider": "string",
  "checksum": "string",
  "userDefinedData": {
    "regionId": ["string"],
    "appInstantiationOrder": ["string"]
  },
  "appD": [
    {
      "appDId": "string",
      "appDVersion": "string",
      "appDNSRule": [
        {
          "dnsRuleId": "string",
          "domainName": "string",
          "ipAddress": "string",
          "ipAddressType": "string",
          "ttl": number
        }
      ],
      "appDescription": "string",
      "appName": "string",
      "appProvider": "string",
      "swImageDescriptor": [
        {
          "id": "string",
          "minDisk": number,
          "minRam": number,
          "size": number,
          "swImage": "string",
          "version": "string",
          "virtualComputeDescId": "string",
          "configuration": [
            {
              "name": "string",
              "value": "string"
            }
          ],
          "ports": [
            {
              "name": "string",
              "containerPort": number,
              "protocol": "string",
              "exposeTo": "string"
            }
          ],
          "name": "string"
        }
      ],
      "virtualComputeDescriptor": [
        {
          "virtualComputeDescId": "string",
          "virtualCpu": {
            "numVirtualCpu": number
          },
          "virtualMemory": {
            "virtualMemSize": number
          }
        }
      ]
    }
  ]
}
```

Generates the NSD (request body) from natural language.

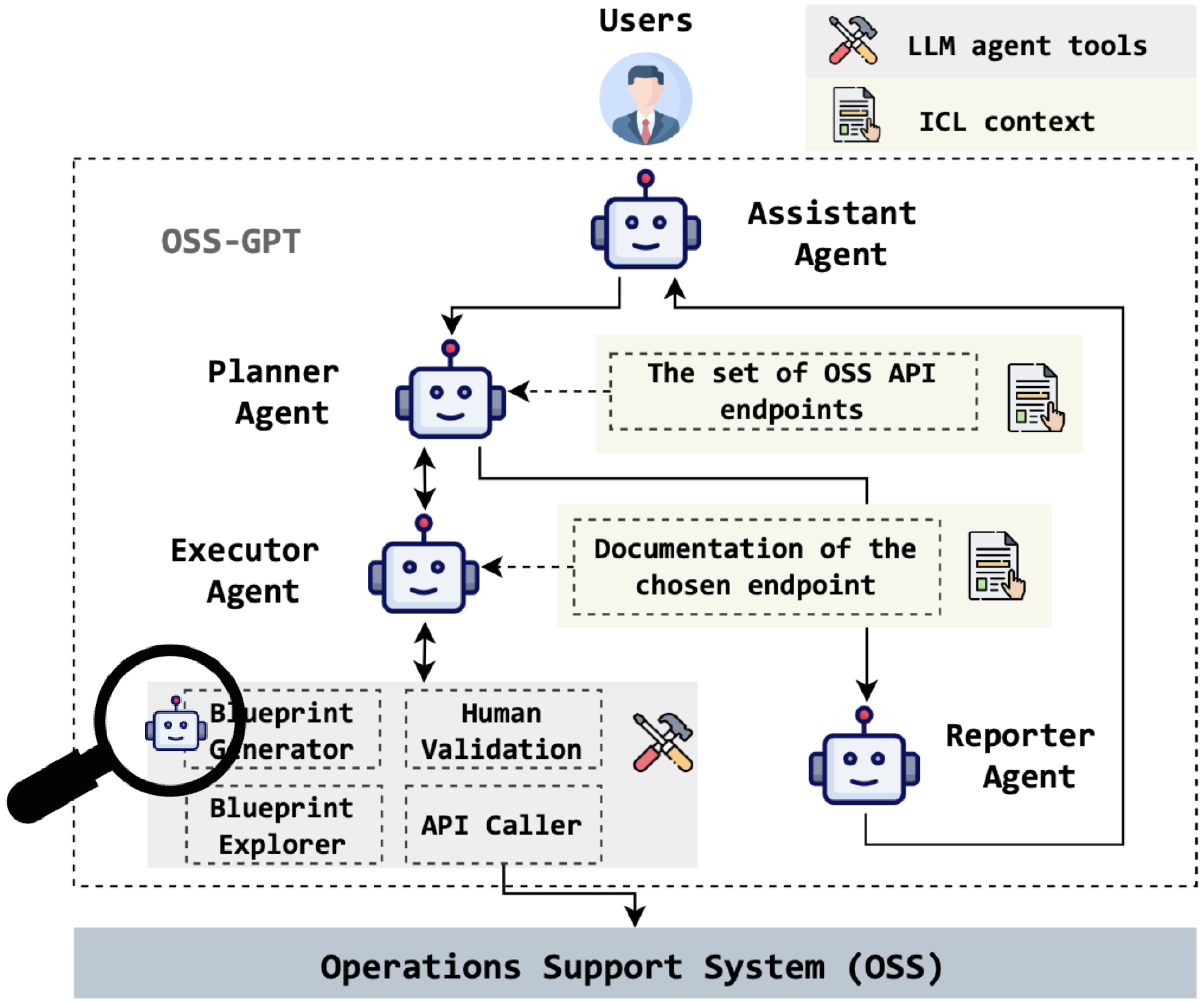
✓ We developed an LLM (NSD-expert) Trained using fine-tuning.

Approach


Users
LLM agent tools
ICL context
OSS-GPT
Assistant Agent
Planner Agent
The set of OSS API endpoints
Executor Agent
Documentation of the chosen endpoint
Blueprint Generator
Human Validation
Blueprint Explorer
API Caller
Reporter Agent
Operations Support System (OSS)

Approach

- Enable OSS-GPT to generate NSDs from natural language.
- Dataset Creation & Augmentation
  - ○ No public dataset available → dataset built from scratch.
  - ○ Initial dataset: 100 high-quality (intent, NSD) pairs.
  - ○ Augmentation via ICL: Used few-shot prompting
- Fine-Tuning with LoRA [1]
  - ○ LoRA (Low-Rank Adaptation): Efficient fine-tuning method
  - ○ Injects NSD expertise without retraining the entire model.

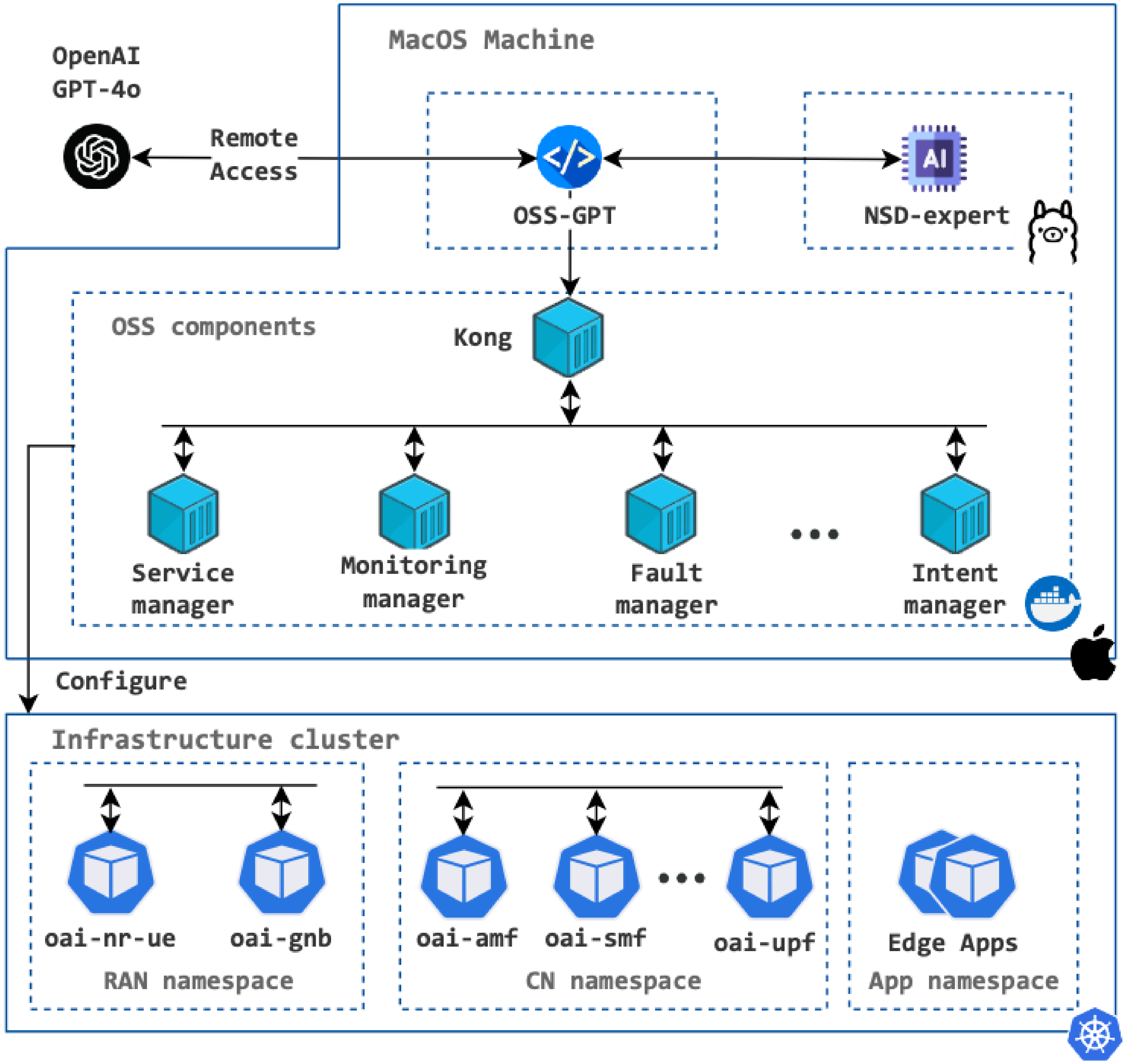


[1] Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." ICLR 1.2 (2022): 3.

Approach

vRAM: video Random Access Memory

- Machine 1: 6G Infrastructure (Kubernetes Cluster)
  - ○ Hosts single-node Kubernetes cluster with E2E 5G stack using OpenAirInterface [1]
- Machine 2: 6G Management (OSS & OSS-GPT)
  - ○ OSS-GPT built using LangGraph [2]
  - ○ GPT-4 for the Assistant, Planner, Executor, and Reporter
  - ○ NSD-expert for the Blueprint Generator (Ollama [3]).
- NSD-expert was fine-tuned using NVIDIA A100 (40GB vRAM)
  - ○ Base: Llama 3.2 - 3B Instruct
  - ○ Framework: Unsloth [4] + LoRA [5]



[1] https://openairinterface.org
[2] https://www.langchain.com/langgraph
[3] https://ollama.com
[4] https://unsloth.ai
[5] Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." ICLR 1.2 (2022): 3.

Demo

RAN: Radio Access Network
CN: Core Network
VIM: Virtualized Infrastructure Manager
O-RAN: Open RAN

- Three infrastructures (VIMs) are available in the EURECOM testbed.
- In this demo, a user will deploy an O-RAN app (RAN subservice) on the second VIM.

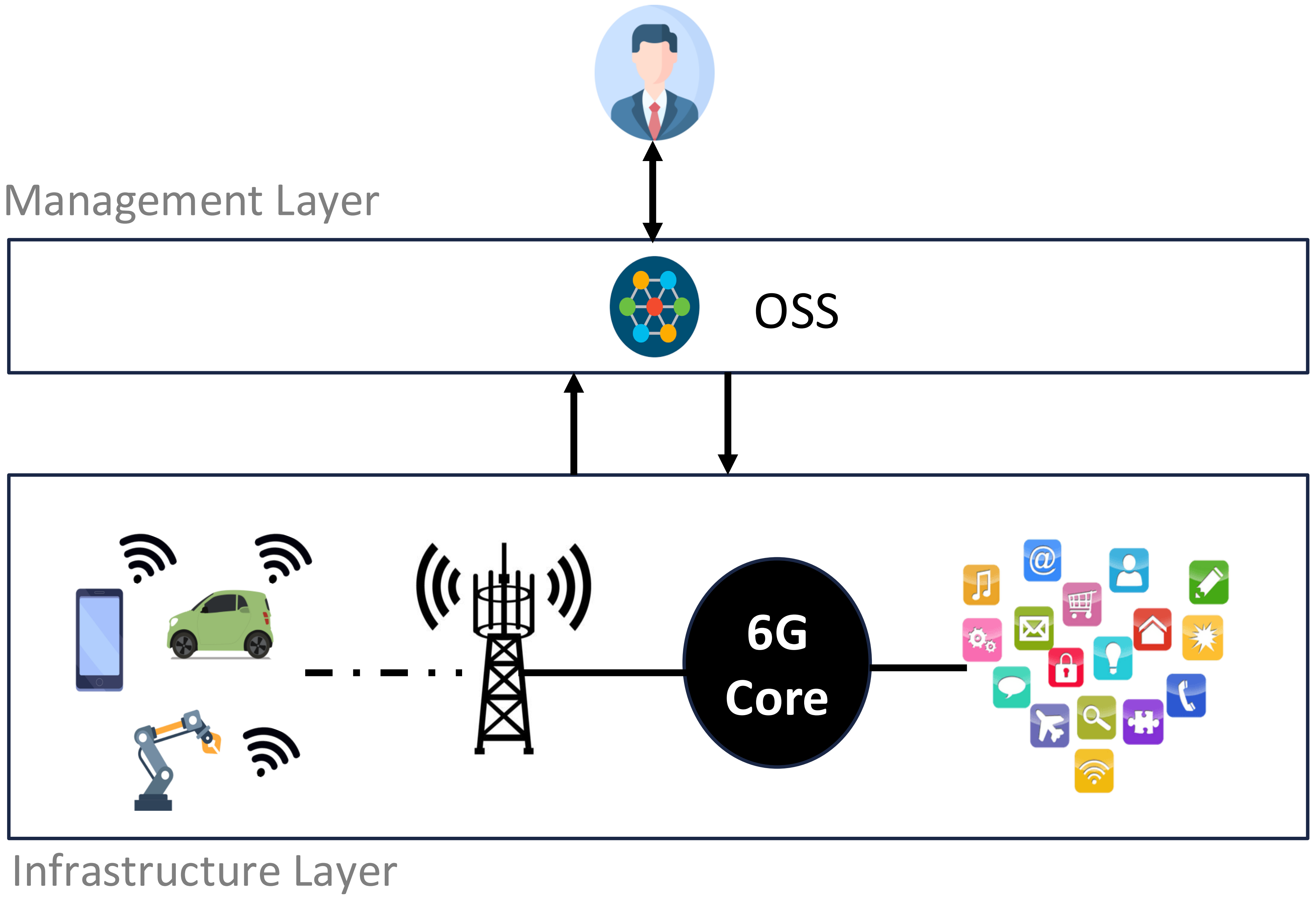
Demo

RAN: Radio Access Network
CN: Core Network
VIM: Virtualized Infrastructure Manager
O-RAN: Open RAN

- Three infrastructures (VIMs) are available in the EURECOM testbed.

- In this demo, a user will deploy an O-RAN app (RAN subservice) on the second VIM.

Terminology …

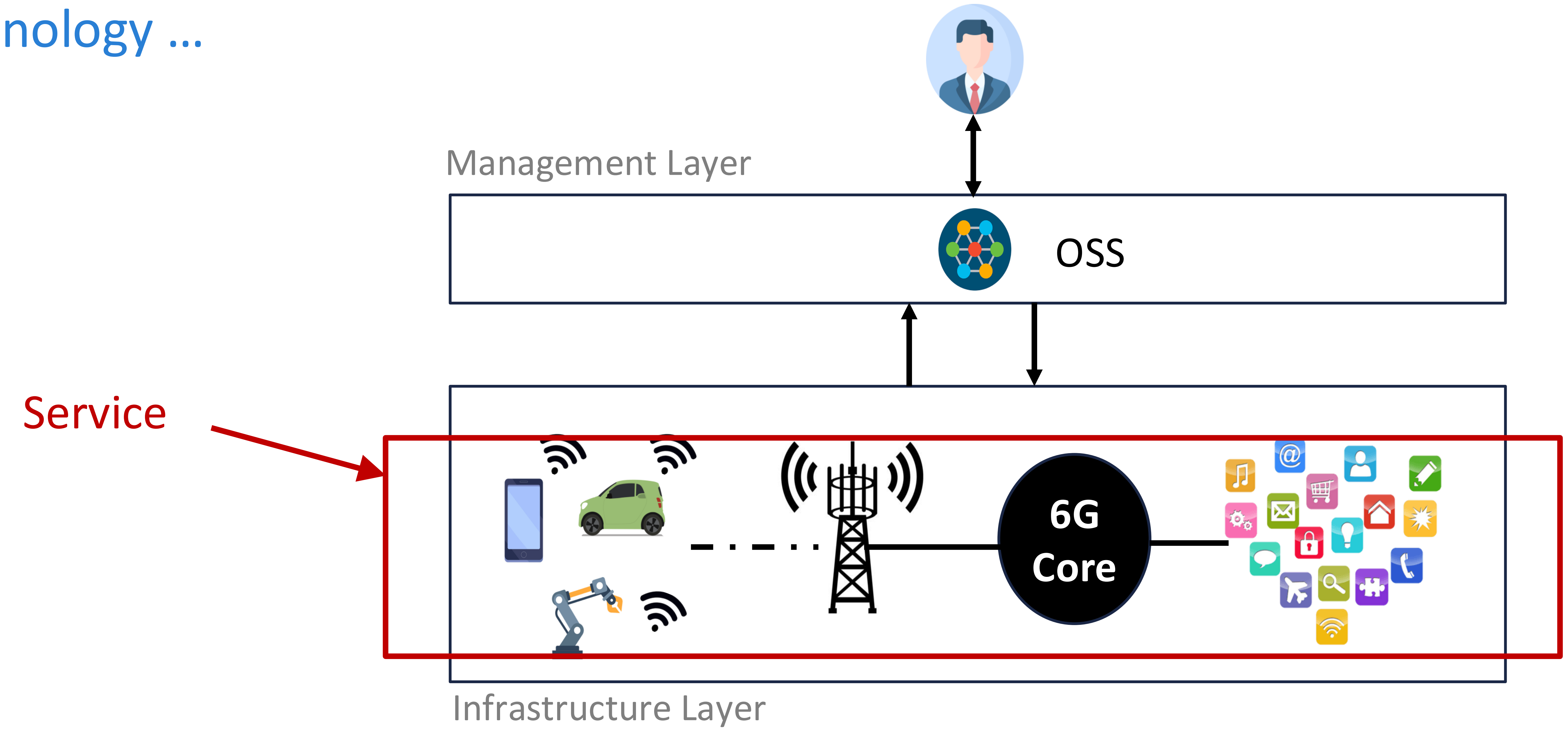


Demo

RAN: Radio Access Network
CN: Core Network
VIM: Virtualized Infrastructure Manager
O-RAN: Open RAN

- Three infrastructures (VIMs) are available in the EURECOM testbed.
- In this demo, a user will deploy an O-RAN app (RAN subservice) on the second VIM.

Terminology …

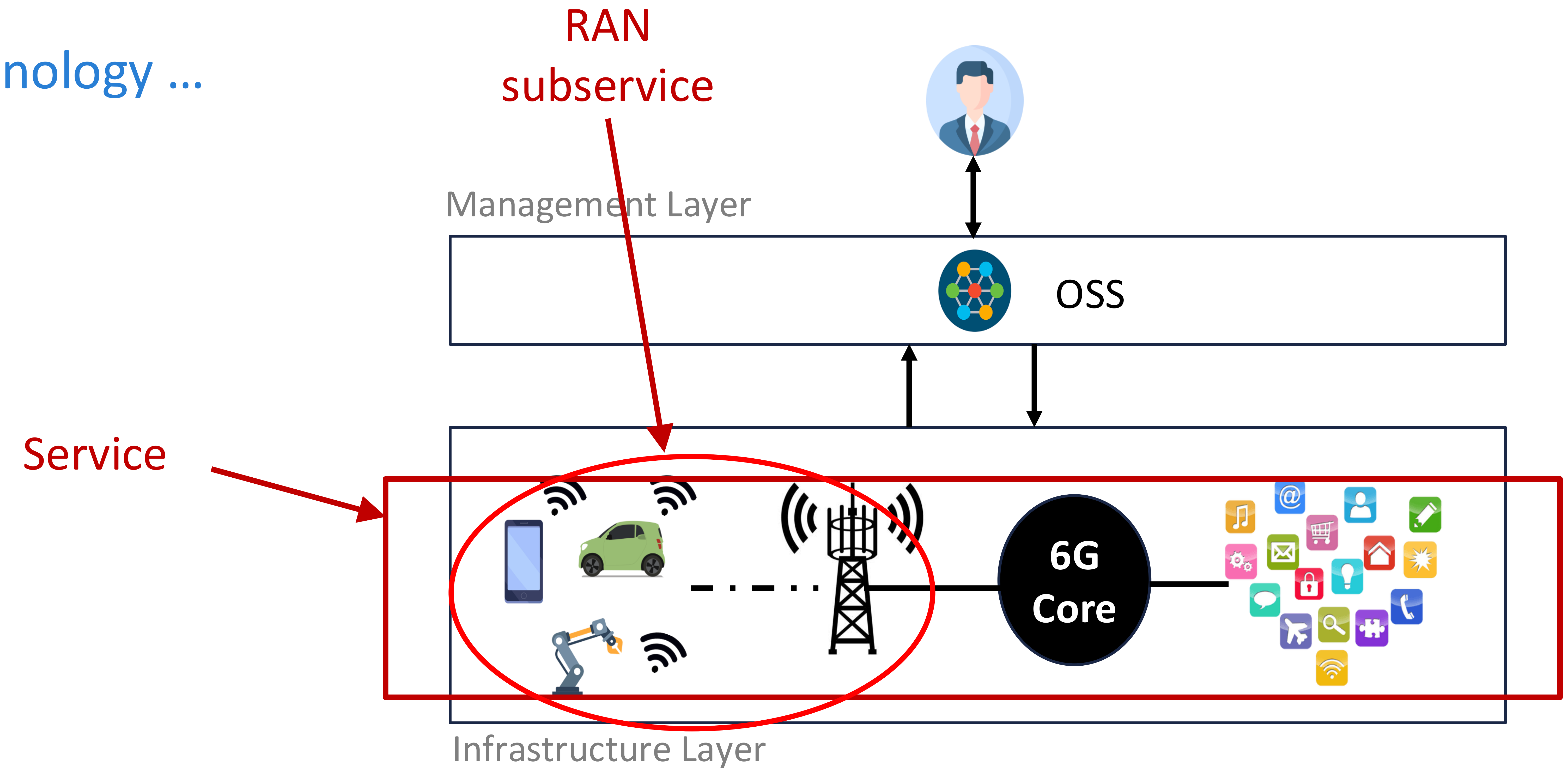


Demo

RAN: Radio Access Network
CN: Core Network
VIM: Virtualized Infrastructure Manager
O-RAN: Open RAN

- Three infrastructures (VIMs) are available in the EURECOM testbed.
- In this demo, a user will deploy an O-RAN app (RAN subservice) on the second VIM.

Terminology …



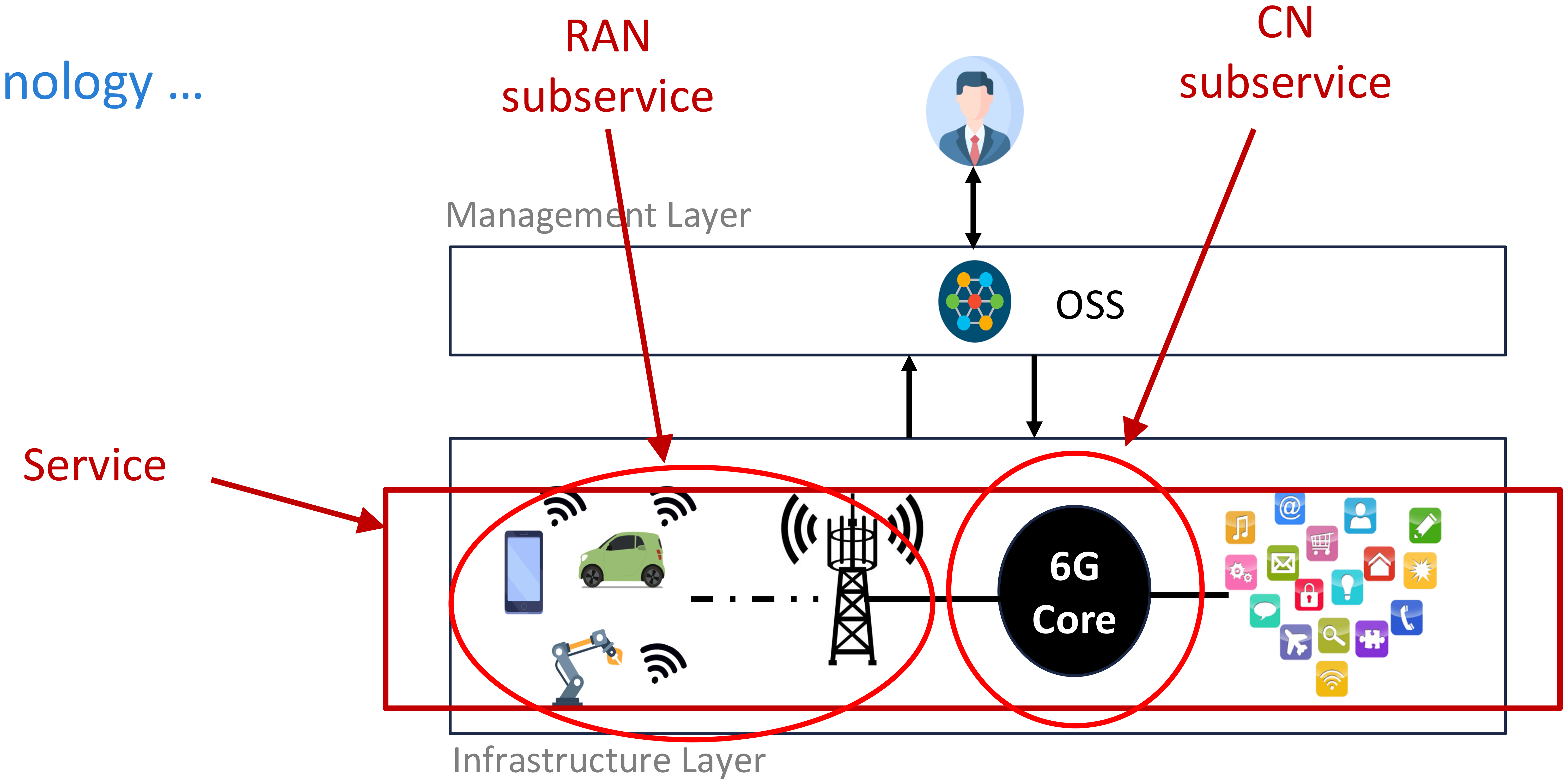
Demo

- Three infrastructures (VIMs) are available in the EURECOM testbed.

- In this demo, a user will deploy an O-RAN app (RAN subservice) on the second VIM.

RAN: Radio Access Network
CN: Core Network
VIM: Virtualized Infrastructure Manager
O-RAN: Open RAN

Terminology …

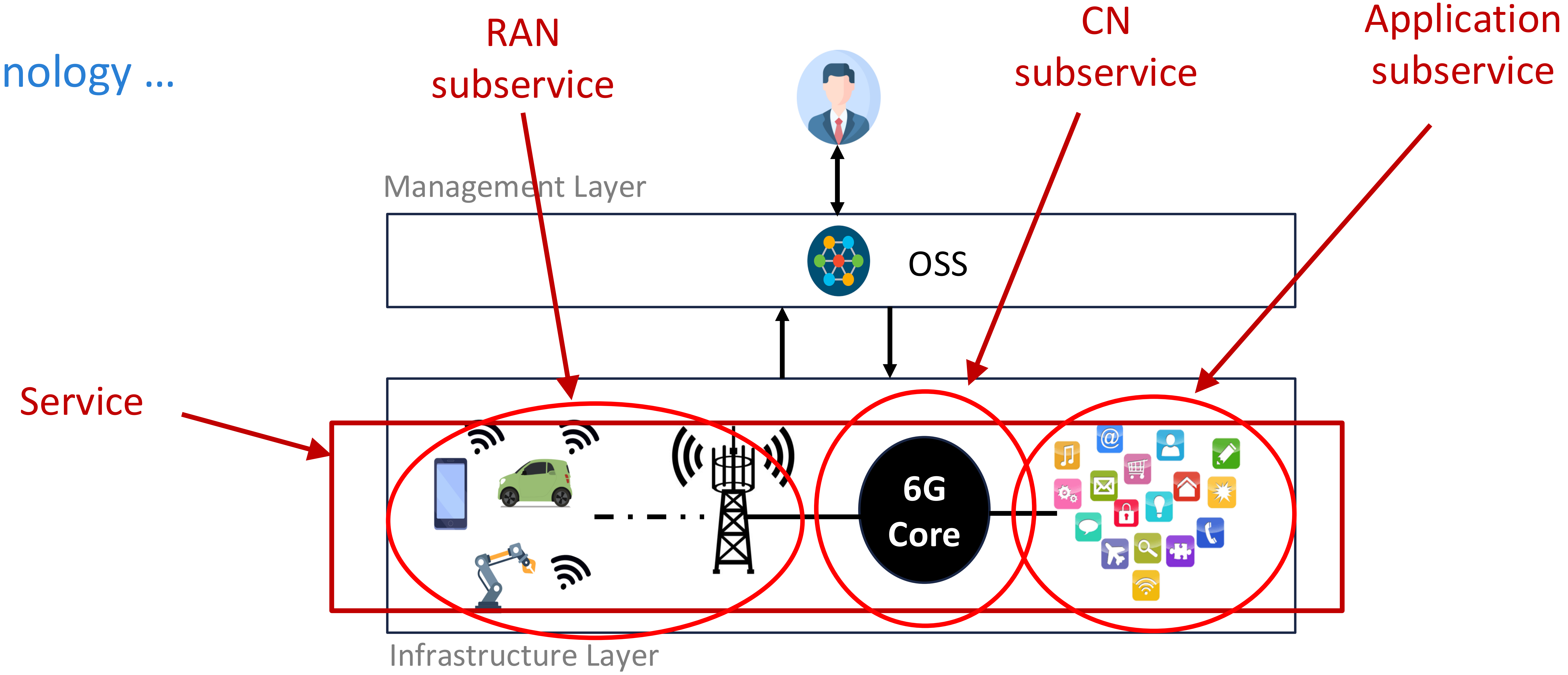


Demo

RAN: Radio Access Network
CN: Core Network
VIM: Virtualized Infrastructure Manager
O-RAN: Open RAN

- Three infrastructures (VIMs) are available in the EURECOM testbed.
- In this demo, a user will deploy an O-RAN app (RAN subservice) on the second VIM.

Terminology …

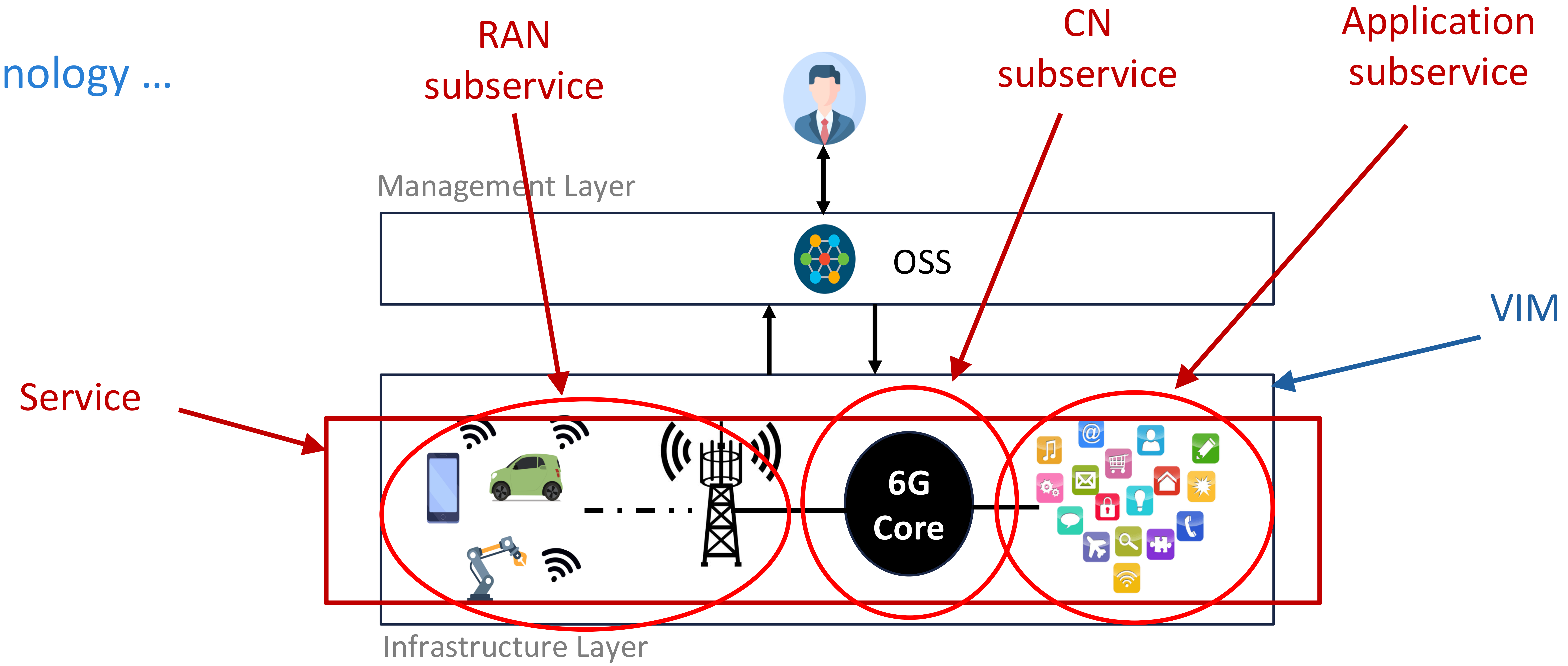


Demo

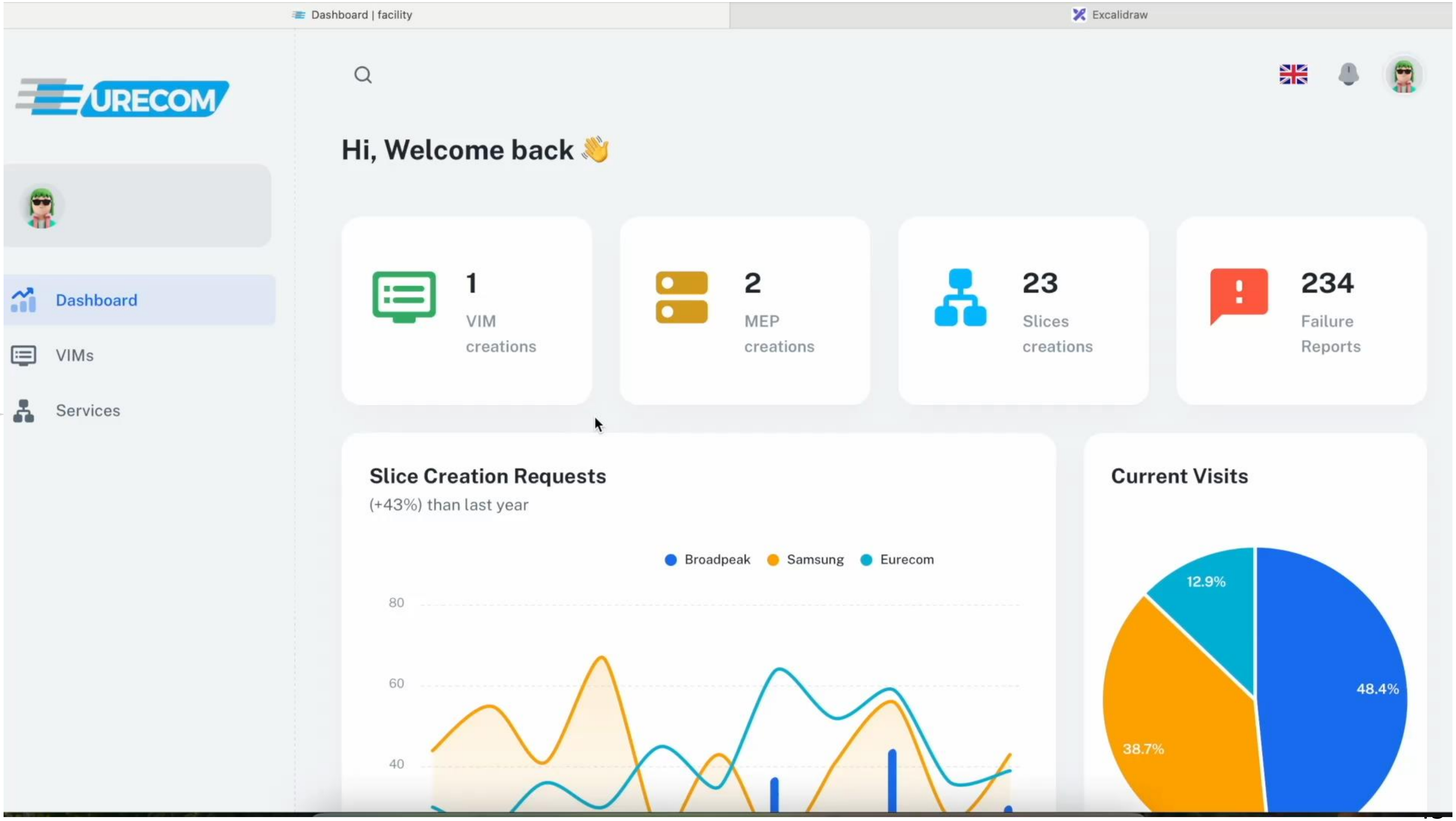
RAN: Radio Access Network
CN: Core Network
VIM: Virtualized Infrastructure Manager
O-RAN: Open RAN

- Three infrastructures (VIMs) are available in the EURECOM testbed.
- In this demo, a user will deploy an O-RAN app (RAN subservice) on the second VIM.

Terminology …



Demo



Dashboard | facility

Excalidraw

EURECOM

Dashboard

VIMs

Services

Hi, Welcome back 👋

1	VIM creations
2	MEP creations
23	Slices creations
234	Failure Reports

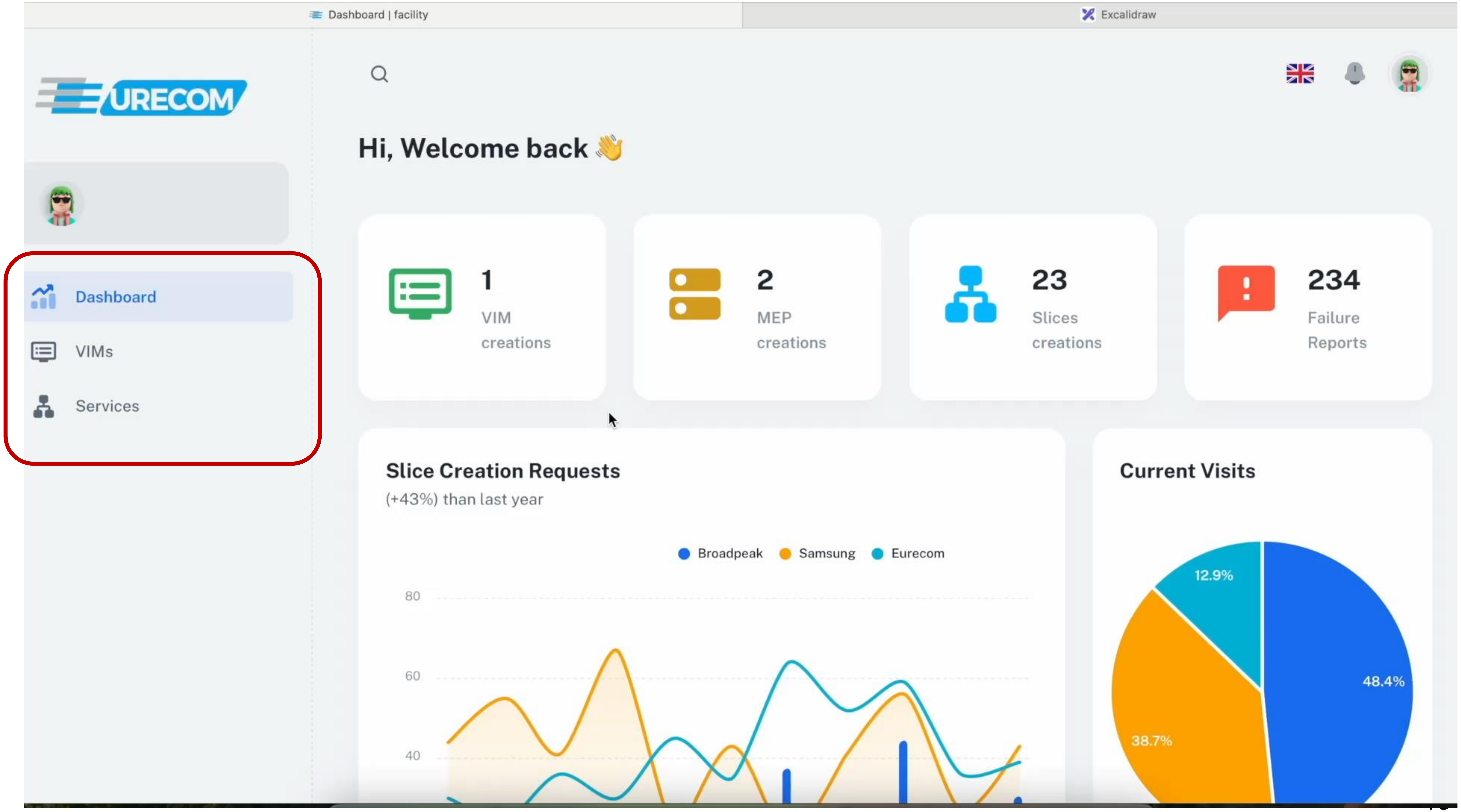
Slice Creation Requests

(+43%) than last year

Current Visits

OSS Frontend

Demo



Hi, Welcome back 👋

1 VIM creations

2 MEP creations

23 Slices creations

234 Failure Reports

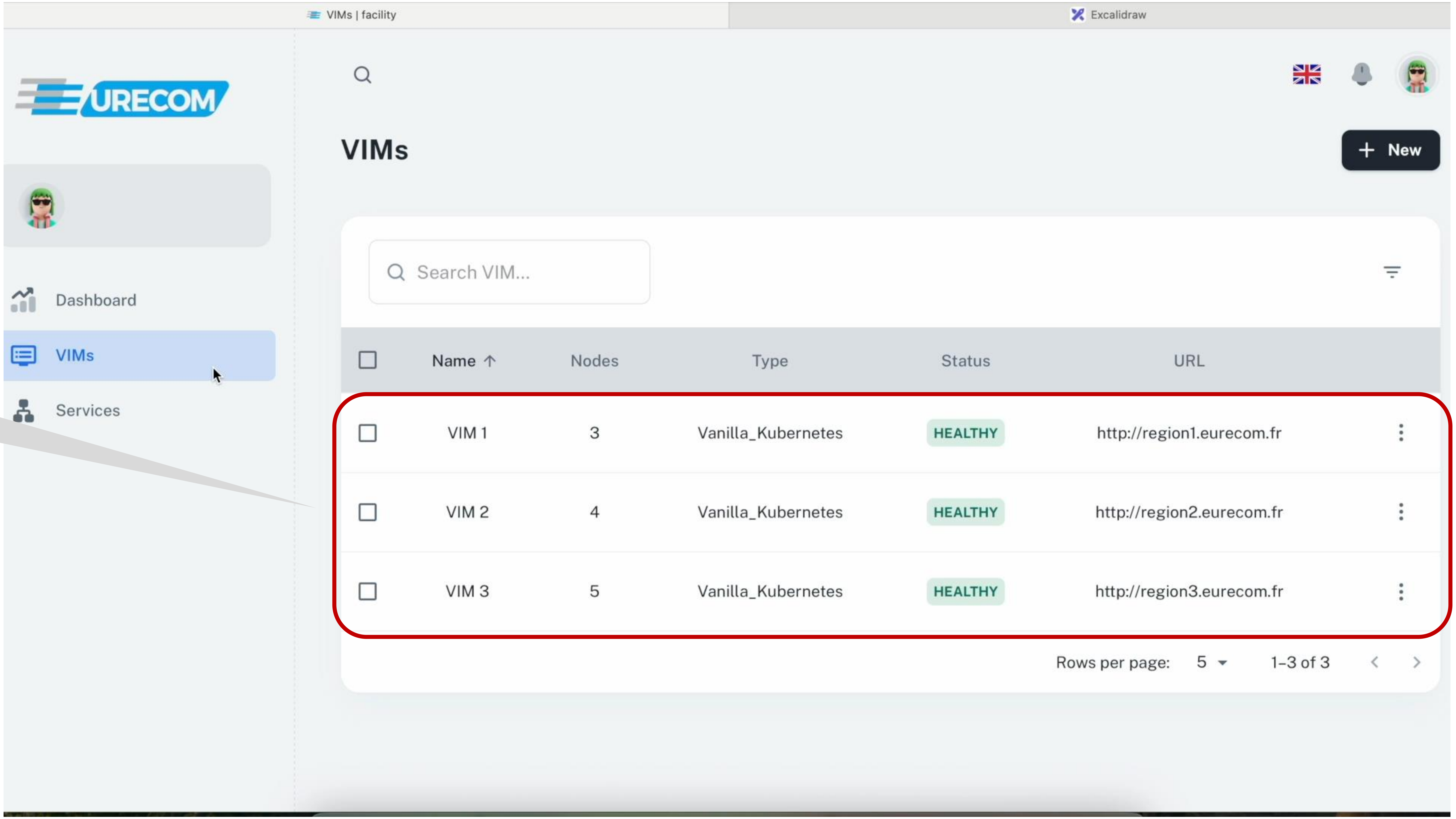
Slice Creation Requests

(+43%) than last year

Current Visits

Manage VIMs and Services

Demo



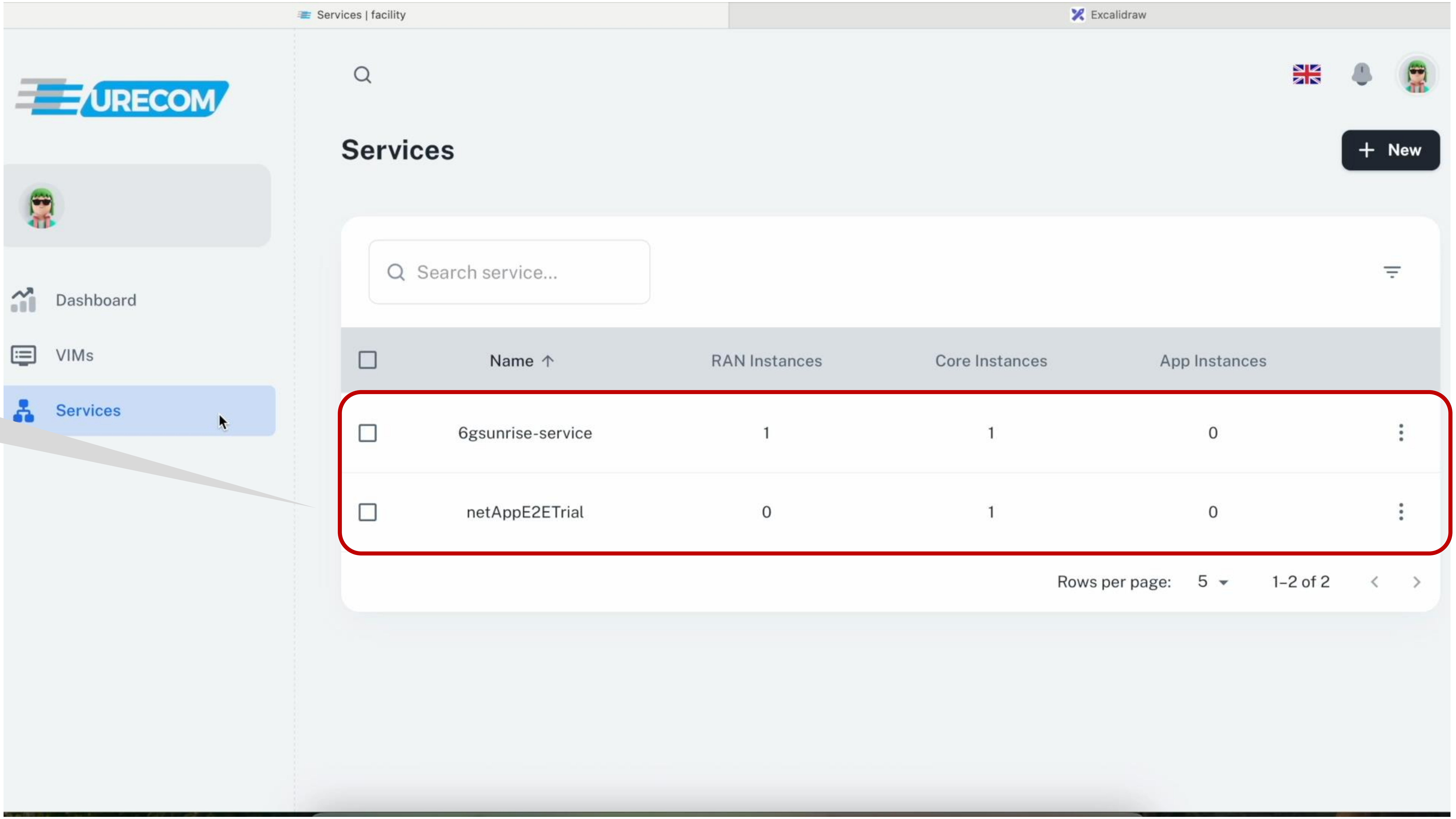
VIMs

□	Name ↑	Nodes	Type	Status	URL
□	VIM 1	3	Vanilla_Kubernetes	HEALTHY	http://region1.eurecom.fr
□	VIM 2	4	Vanilla_Kubernetes	HEALTHY	http://region2.eurecom.fr
□	VIM 3	5	Vanilla_Kubernetes	HEALTHY	http://region3.eurecom.fr

Rows per page: 5 1–3 of 3

List of VIMS

Demo



Services

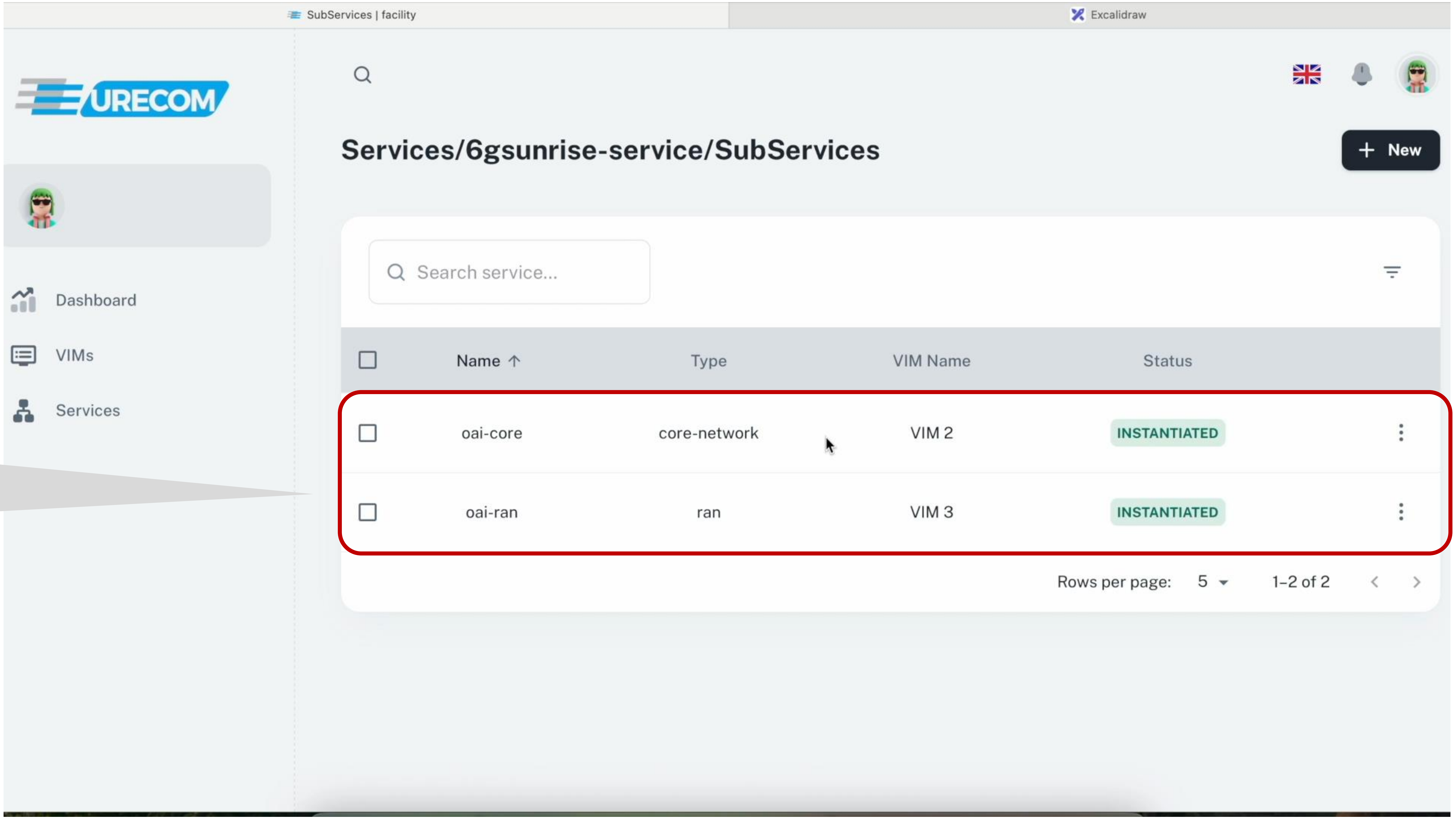
Search service...

□	Name ↑	RAN Instances	Core Instances	App Instances
□	6gsunrise-service	1	1	0
□	netAppE2ETrial	0	1	0

Rows per page: 5 1–2 of 2

List of Services

Demo



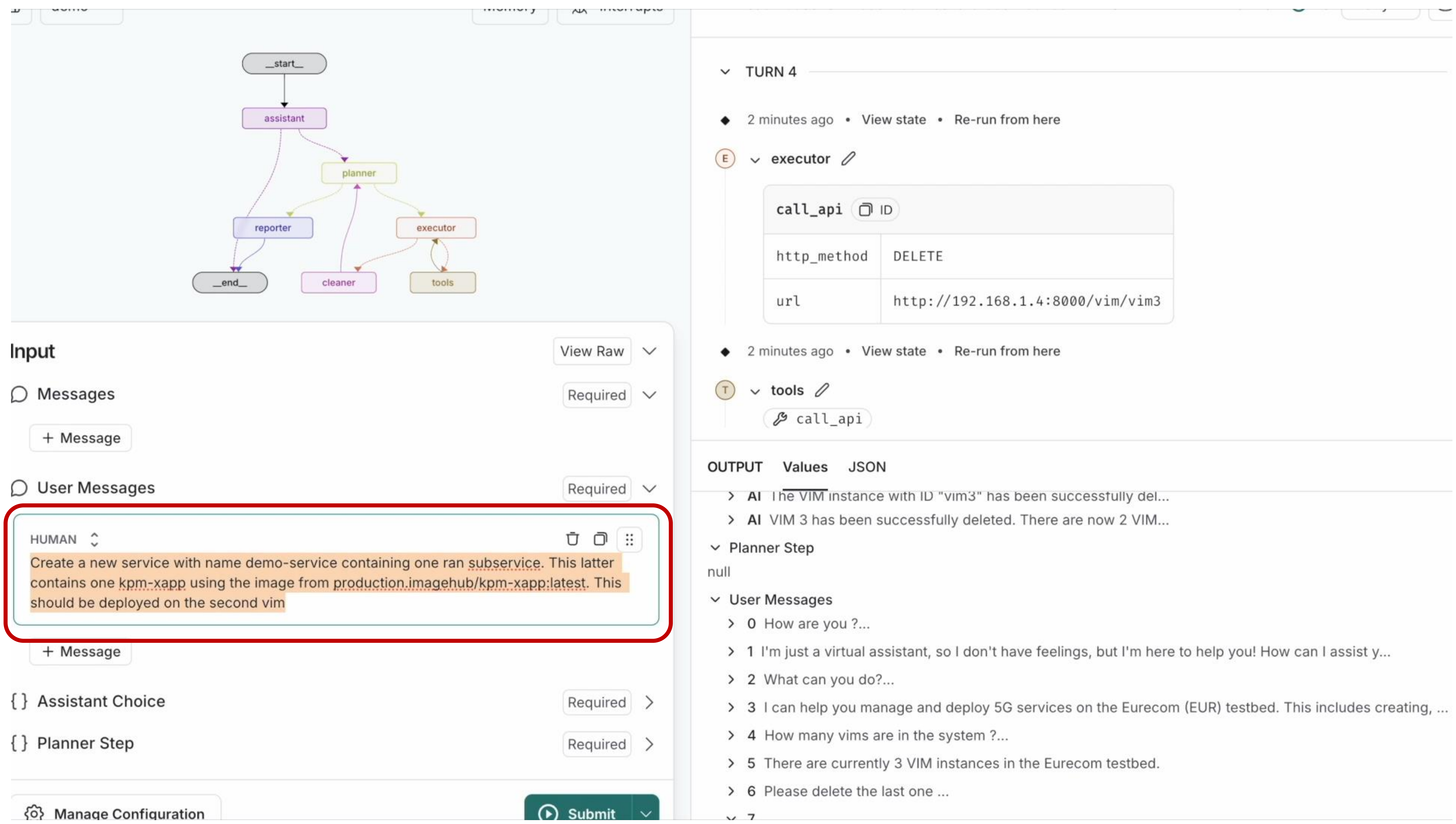
Services/6gsunrise-service/SubServices

□	Name ↑	Type	VIM Name	Status
□	oai-core	core-network	VIM 2	INSTANTIATED
□	oai-ran	ran	VIM 3	INSTANTIATED

- Core Network deployed on VIM 2
- RAN deployed on VIM 3

Demo

User's Intent



Input

View Raw

Messages Required

+ Message

User Messages Required

HUMAN

Create a new service with name demo-service containing one ran subservice. This latter contains one kpm-xapp using the image from production.imagehub/kpm-xapp:latest. This should be deployed on the second vim

+ Message

{ } Assistant Choice Required

{ } Planner Step Required

Manage Configuration

Submit

TURN 4

2 minutes ago • View state • Re-run from here

E executor

call_api ID

http_method	DELETE
url	http://192.168.1.4:8000/vim/vim3

2 minutes ago • View state • Re-run from here

T tools

call_api

OUTPUT Values JSON

AI The VIM instance with ID "vim3" has been successfully del...

AI VIM 3 has been successfully deleted. There are now 2 VIM...

Planner Step

null

User Messages

0 How are you ?...

1 I'm just a virtual assistant, so I don't have feelings, but I'm here to help you! How can I assist y...

2 What can you do?...

3 I can help you manage and deploy 5G services on the Eurecom (EUR) testbed. This includes creating, ...

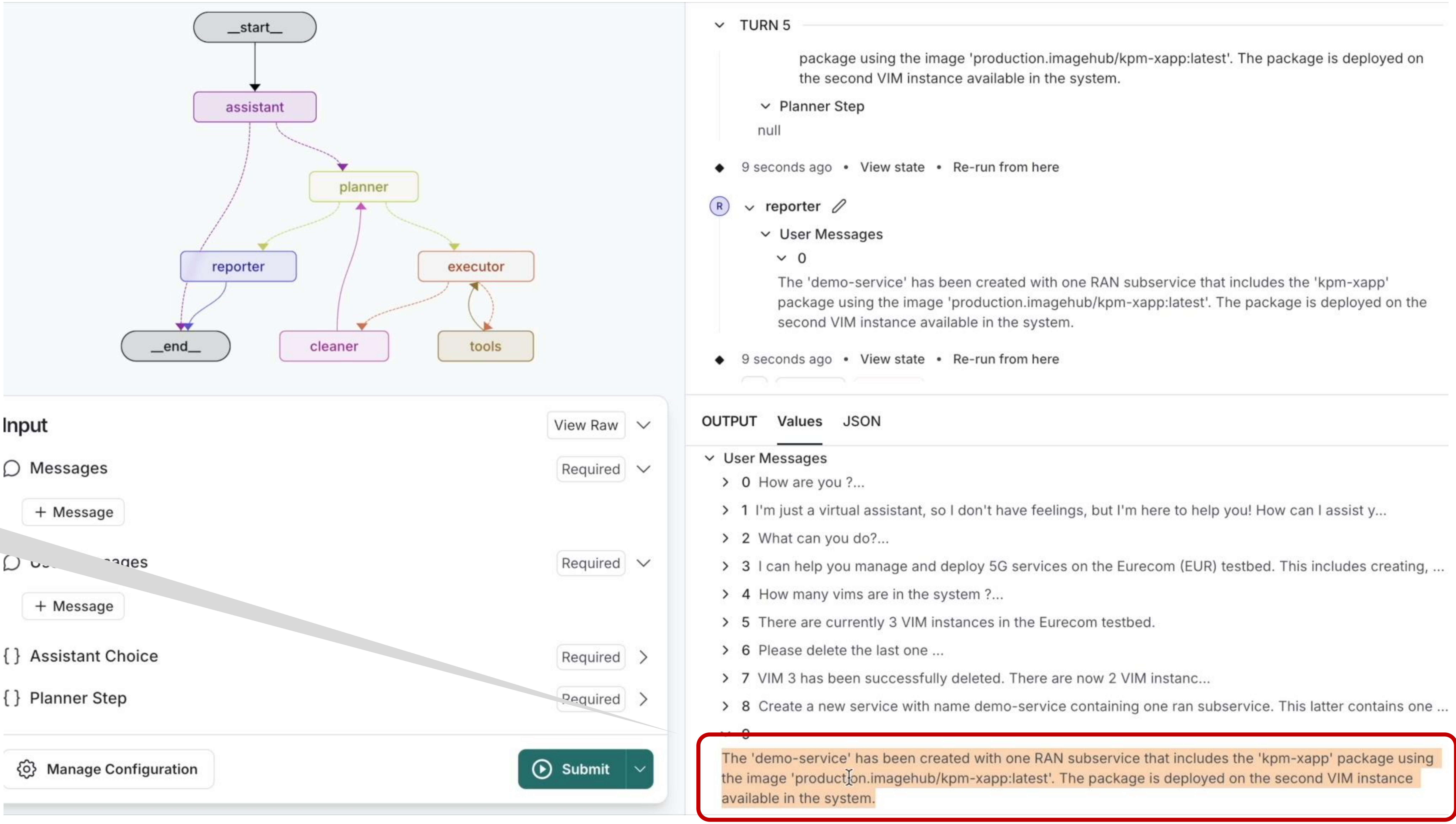
4 How many vims are in the system ?...

5 There are currently 3 VIM instances in the Eurecom testbed.

6 Please delete the last one ...

7

Demo



package using the image 'production.imagehub/kpm-xapp:latest'. The package is deployed on the second VIM instance available in the system.

null

The 'demo-service' has been created with one RAN subservice that includes the 'kpm-xapp' package using the image 'production.imagehub/kpm-xapp:latest'. The package is deployed on the second VIM instance available in the system.

0 How are you ?...

1 I'm just a virtual assistant, so I don't have feelings, but I'm here to help you! How can I assist y...

2 What can you do?...

3 I can help you manage and deploy 5G services on the Eurecom (EUR) testbed. This includes creating, ...

4 How many vims are in the system ?...

5 There are currently 3 VIM instances in the Eurecom testbed.

6 Please delete the last one ...

7 VIM 3 has been successfully deleted. There are now 2 VIM instanc...

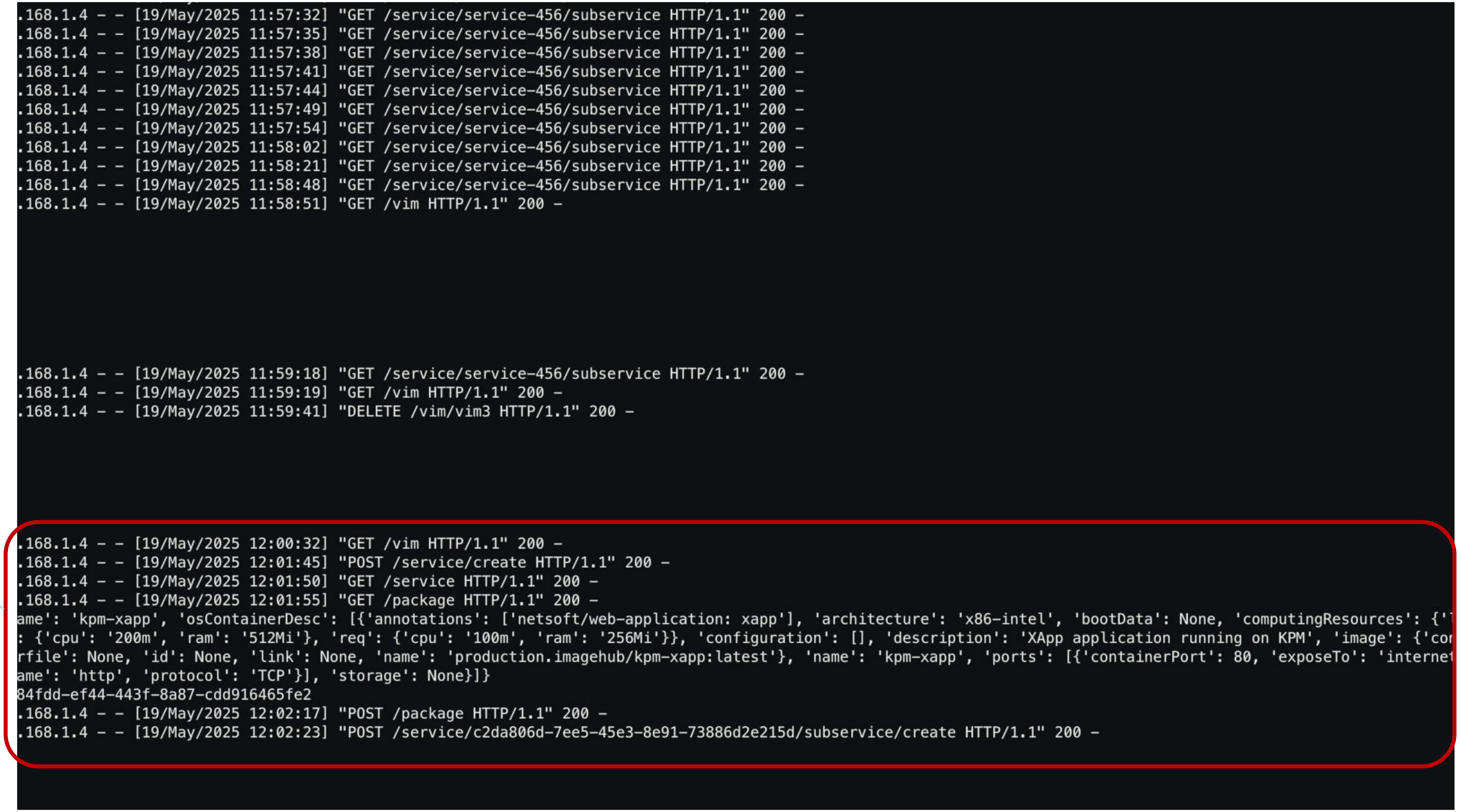
8 Create a new service with name demo-service containing one ran subservice. This latter contains one ...

9

The 'demo-service' has been created with one RAN subservice that includes the 'kpm-xapp' package using the image 'production.imagehub/kpm-xapp:latest'. The package is deployed on the second VIM instance available in the system.

OSS-GPT report

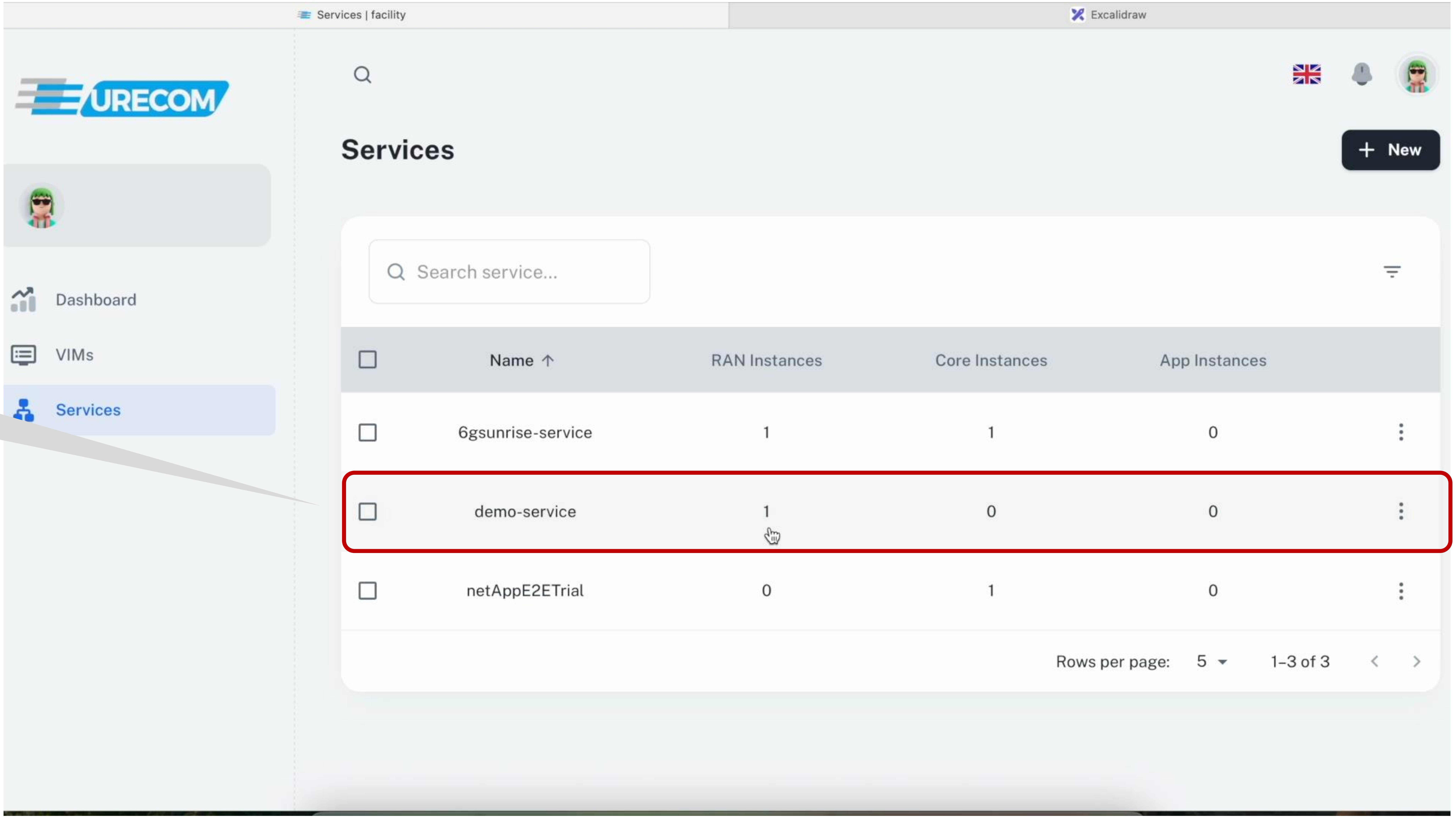
Demo



```
.168.1.4 - - [19/May/2025 11:57:32] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:57:35] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:57:38] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:57:41] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:57:44] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:57:49] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:57:54] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:58:02] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:58:21] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:58:48] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:58:51] "GET /vim HTTP/1.1" 200 -








.168.1.4 - - [19/May/2025 11:59:18] "GET /service/service-456/subservice HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:59:19] "GET /vim HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 11:59:41] "DELETE /vim/vim3 HTTP/1.1" 200 -






.168.1.4 - - [19/May/2025 12:00:32] "GET /vim HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 12:01:45] "POST /service/create HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 12:01:50] "GET /service HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 12:01:55] "GET /package HTTP/1.1" 200 -
ame': 'kpm-xapp', 'osContainerDesc': [{'annotations': ['netsoft/web-application: xapp'], 'architecture': 'x86-intel', 'bootData': None, 'computingResources': {'
: {'cpu': '200m', 'ram': '512Mi'}, 'req': {'cpu': '100m', 'ram': '256Mi'}}, 'configuration': [], 'description': 'XApp application running on KPM', 'image': {'co
rfile': None, 'id': None, 'link': None, 'name': 'production.imagehub/kpm-xapp:latest'}, 'name': 'kpm-xapp', 'ports': [{'containerPort': 80, 'exposeTo': 'interne
ame': 'http', 'protocol': 'TCP'}], 'storage': None}]}
84fdd-ef44-443f-8a87-cdd916465fe2
.168.1.4 - - [19/May/2025 12:02:17] "POST /package HTTP/1.1" 200 -
.168.1.4 - - [19/May/2025 12:02:23] "POST /service/c2da806d-7ee5-45e3-8e91-73886d2e215d/subservice/create HTTP/1.1" 200 -
```

List of OSS API endpoints used by OSS-GPT to fulfill the intent

Demo



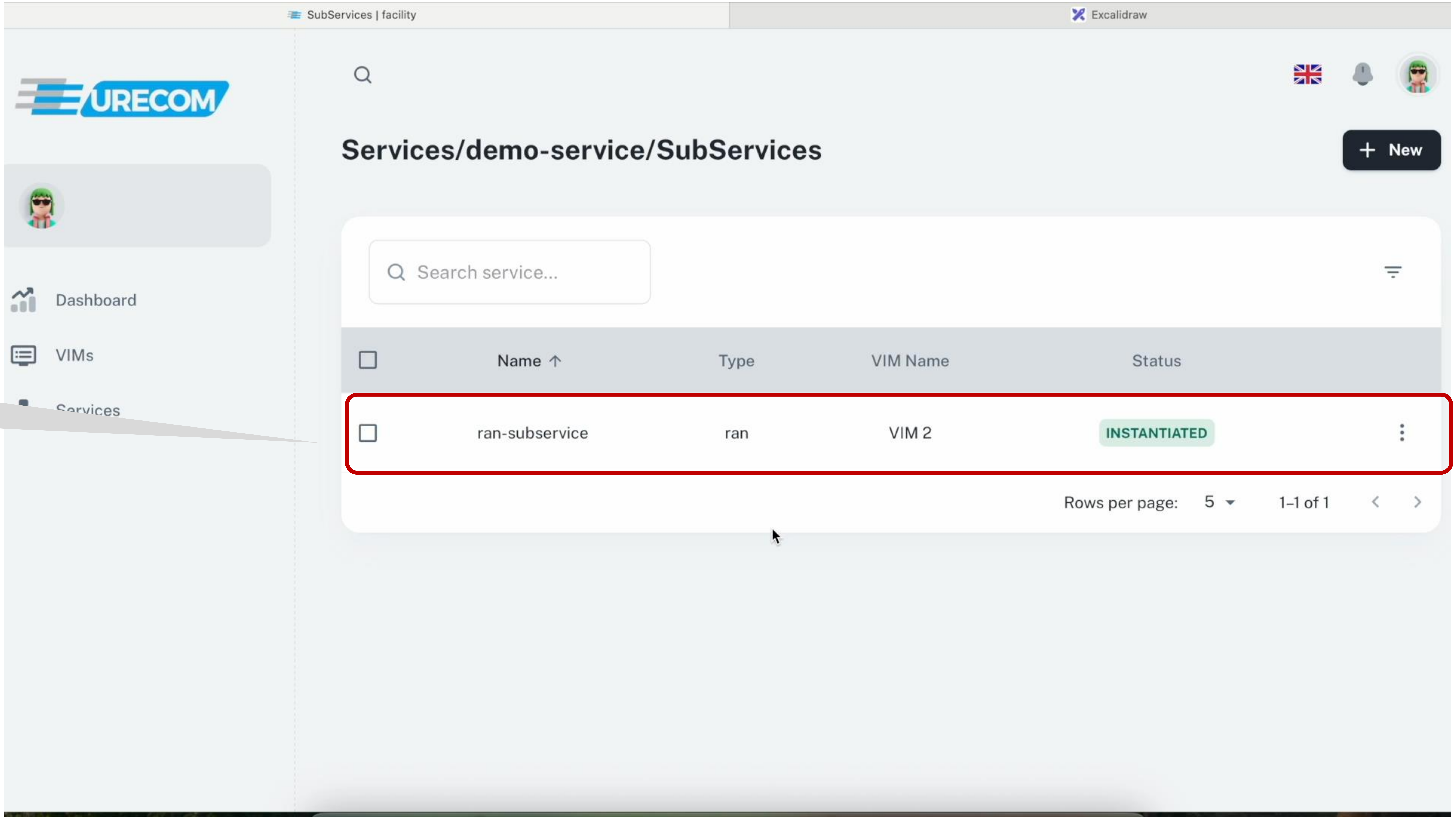
Services

□	Name ↑	RAN Instances	Core Instances	App Instances
□	6gsunrise-service	1	1	0
□	demo-service	1	0	0
□	netAppE2ETrial	0	1	0

Rows per page: 5 1–3 of 3

The new service is created

Demo



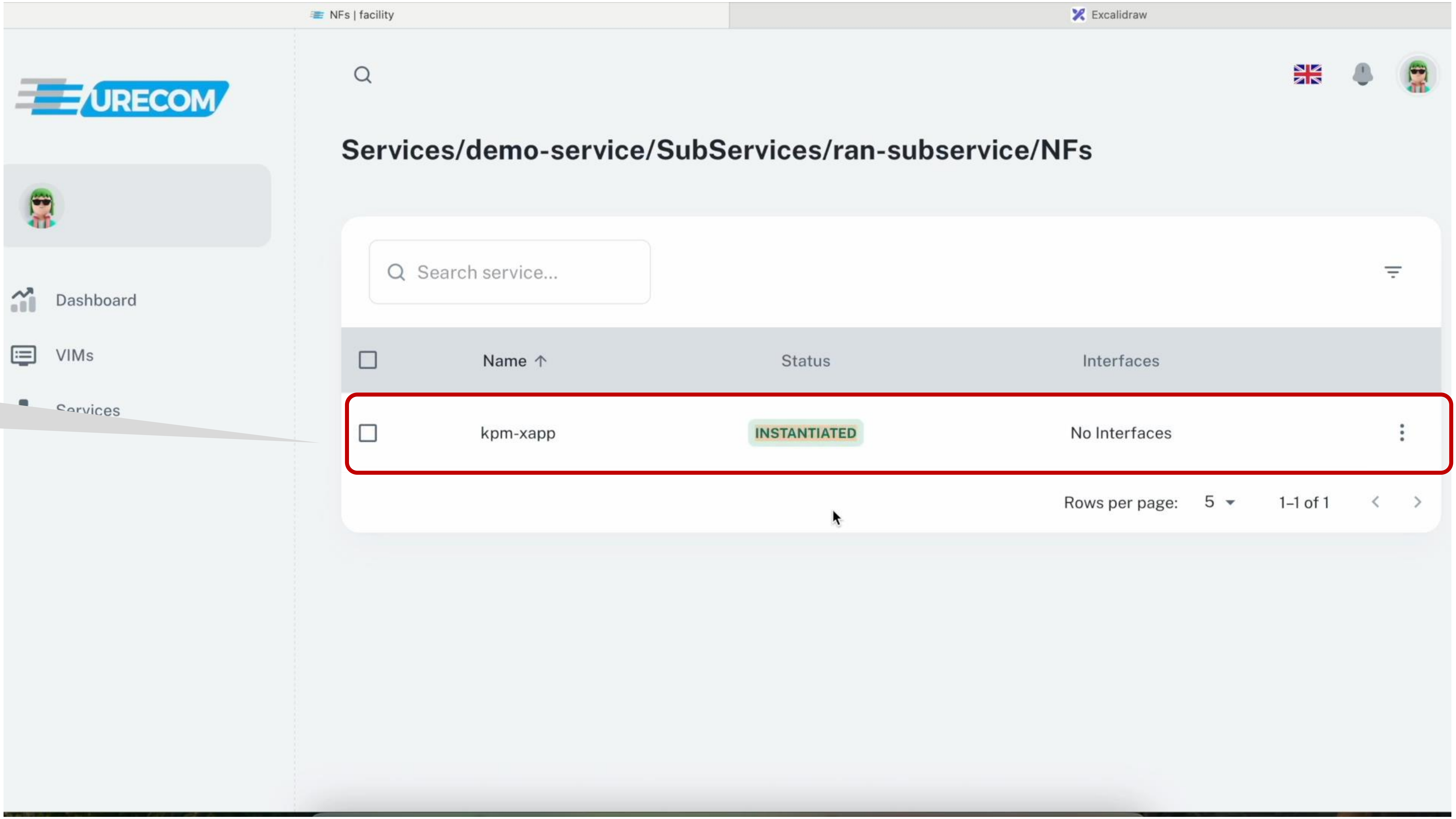
Services/demo-service/SubServices

□	Name ↑	Type	VIM Name	Status
□	ran-subservice	ran	VIM 2	INSTANTIATED

Rows per page: 5 ▾ 1–1 of 1

The subservice is deployed on VIM 2

Demo



Services/demo-service/SubServices/ran-subservice/NFs

Search service...

□	Name ↑	Status	Interfaces	
□	kpm-xapp	INSTANTIATED	No Interfaces	⋮

Rows per page: 5 ▾ 1–1 of 1 ‹ ›

The subservice contains 1 kpm-xapp

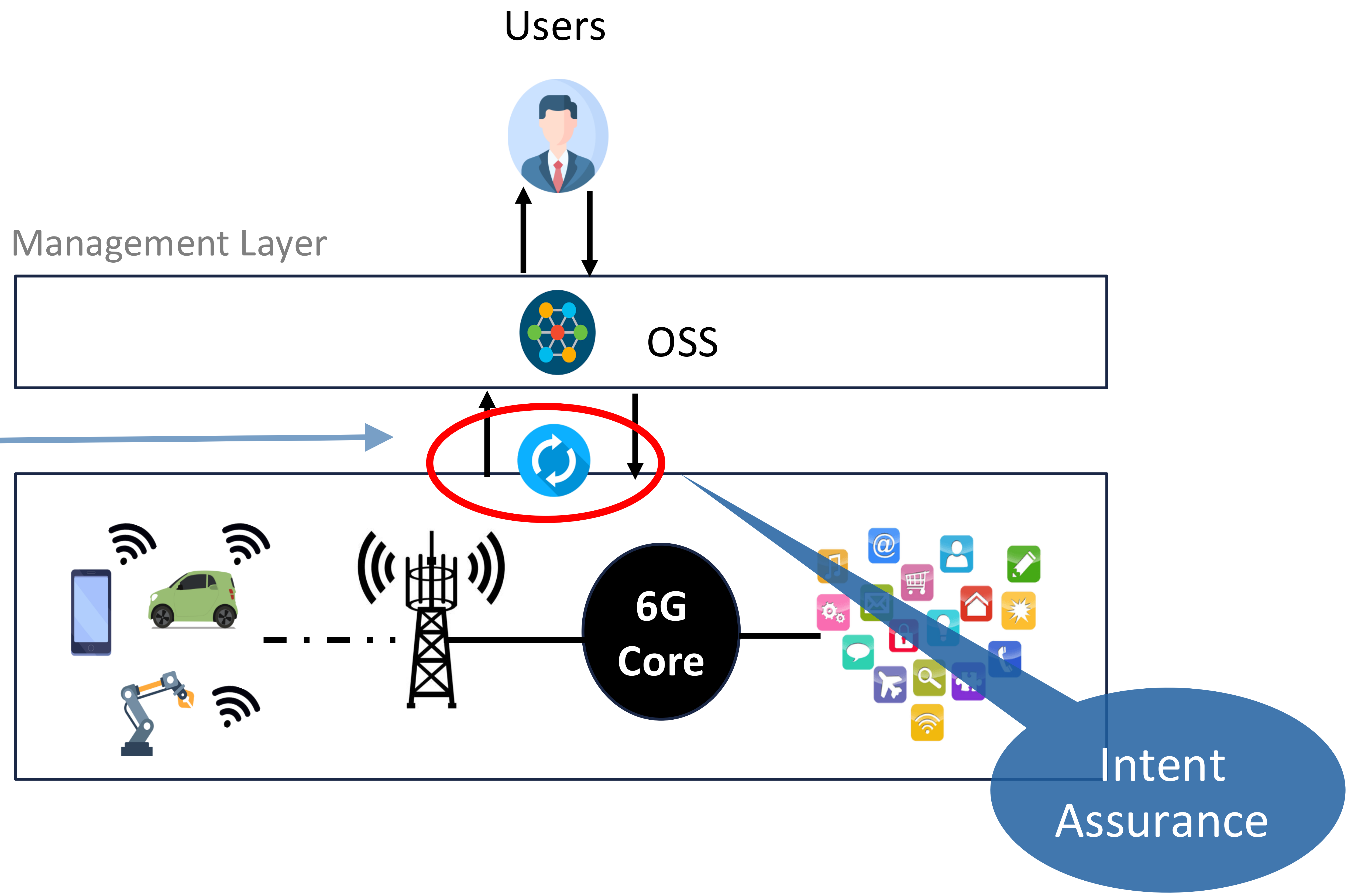
Table of Contents

- Intent-Based Networking (IBN)
- Intent Translation with LLMs
- Intent Assurance with LLMs
- Conclusion

Motivation

API: Application Programming Interface

Closed control loop to ensure intent requirements are satisfied throughout their lifecycle



Motivation

XAI: eXplainable AI

- Intent Assurance, or Zero-touch Network and Service Management (ZSM), enables autonomous network management without human intervention.

- Detecting anomalies, identifying their root causes, and resolving them autonomously.

Motivation

XAI: eXplainable AI

- Intent Assurance, or Zero-touch Network and Service Management (ZSM), enables autonomous network management without human intervention.

- Detecting anomalies, identifying their root causes, and resolving them autonomously.

AI methods have been widely used in research to detect anomalies.
However, AI are often black boxes lacking explainability, making it difficult to extract the root cause.

Motivation

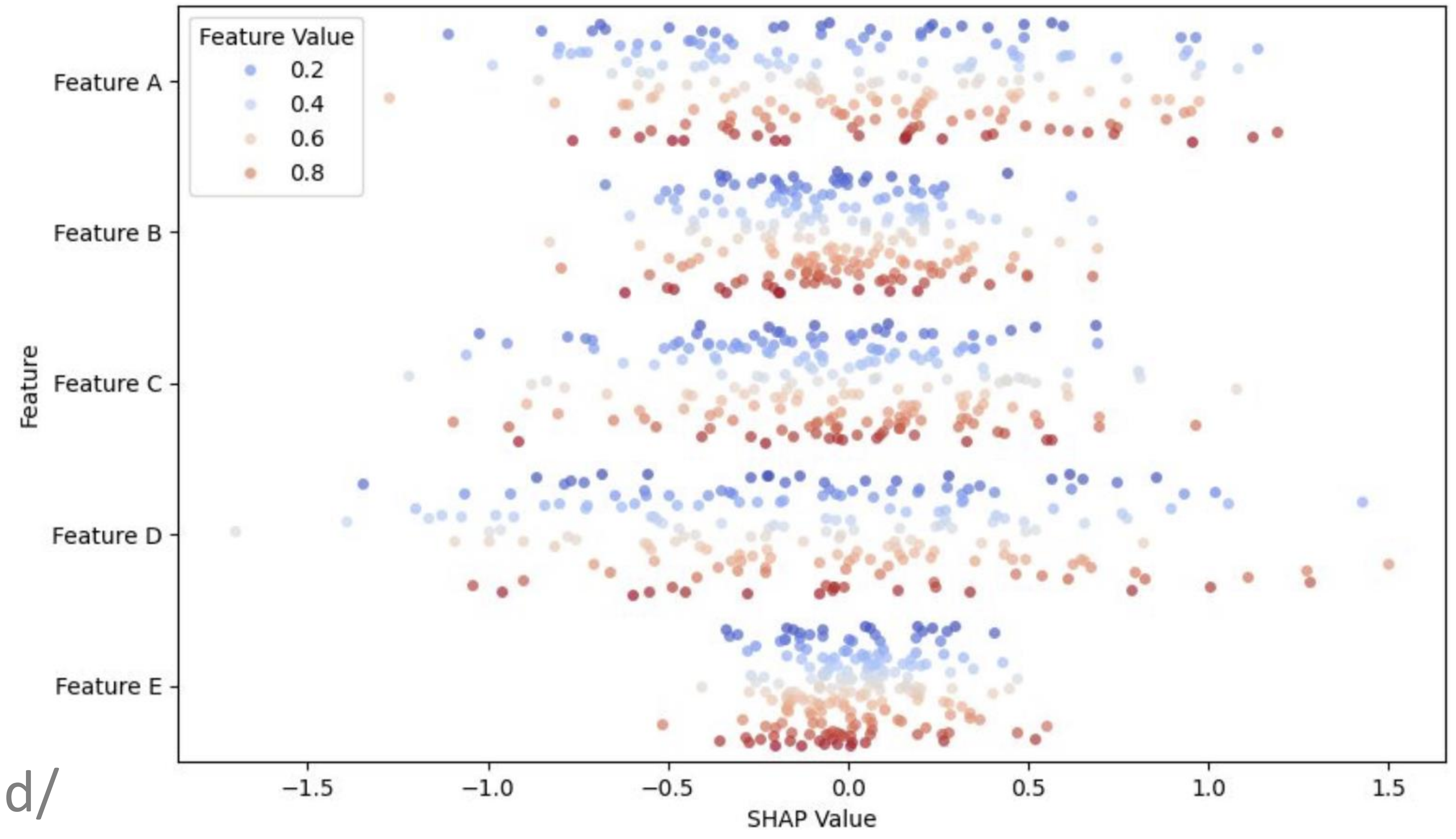
XAI: eXplainable AI

- Intent Assurance, or Zero-touch Network and Service Management (ZSM), enables autonomous network management without human intervention.

- Detecting anomalies, identifying their root causes, and resolving them autonomously.

AI methods have been widely used in research to detect anomalies.
However, AI are often black boxes lacking explainability, making it difficult to extract the root cause.

XAI methods have emerged to explain AI decisions, thus identify the root causes of anomalies.
However, XAI rely on numerical values to explain anomalies



https://aicompetence.org/understanding-xai-shap-lime-and-beyond/

Motivation

XAI: eXplainable AI

- Intent Assurance, or Zero-touch Network and Service Management (ZSM), enables autonomous network management without human intervention.

- Detecting anomalies, identifying their root causes, and resolving them autonomously.

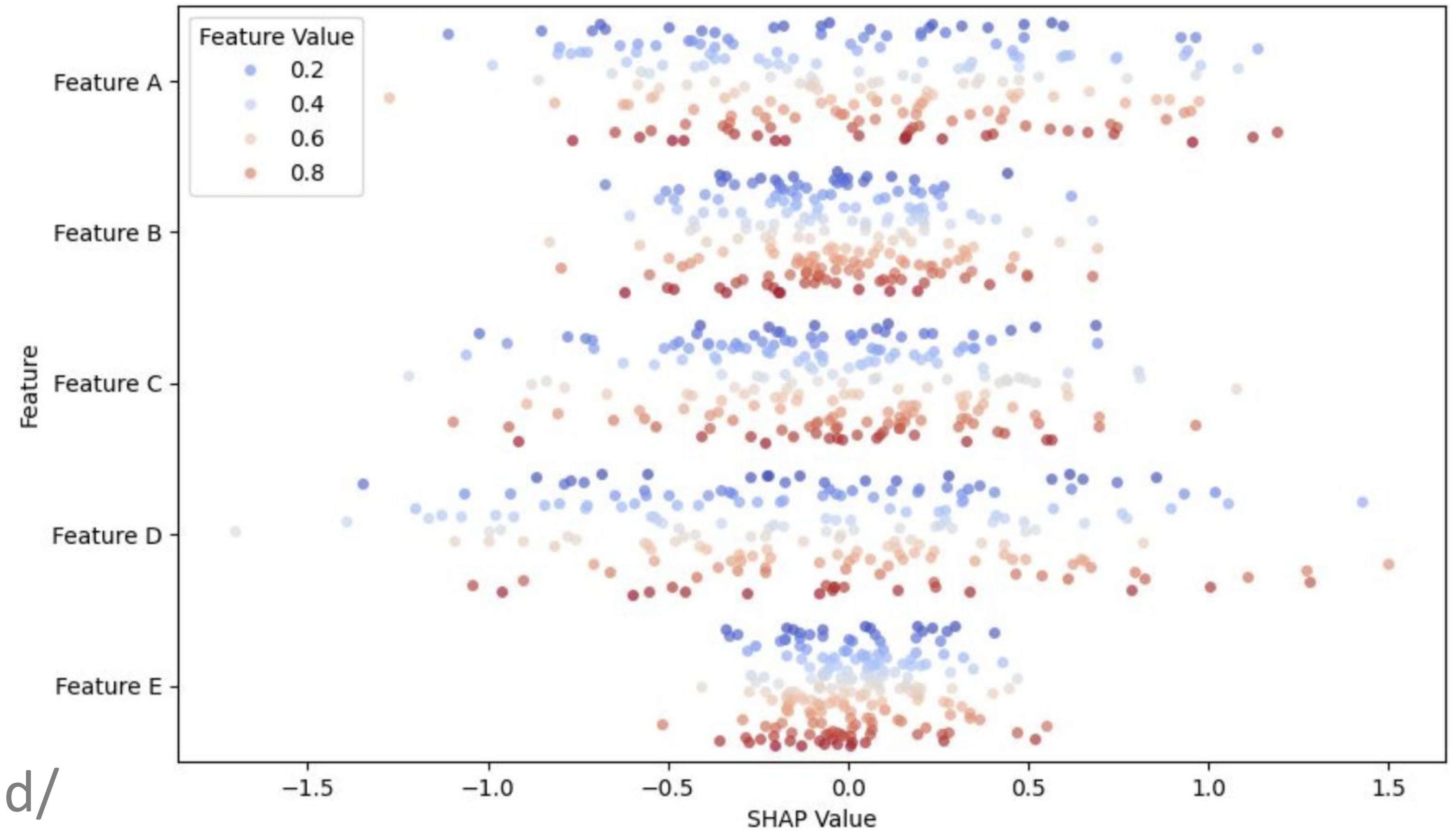
AI methods have been widely used in research to detect anomalies.
However, AI are often black boxes lacking explainability, making it difficult to extract the root cause.

XAI methods have emerged to explain AI decisions, thus identify the root causes of anomalies.
However, XAI rely on numerical values to explain anomalies

These XAI values are difficult for users with little domain knowledge to understand.

This removes trust in the autonomous system, which is mandatory in IBN

https://aicompetence.org/understanding-xai-shap-lime-and-beyond/

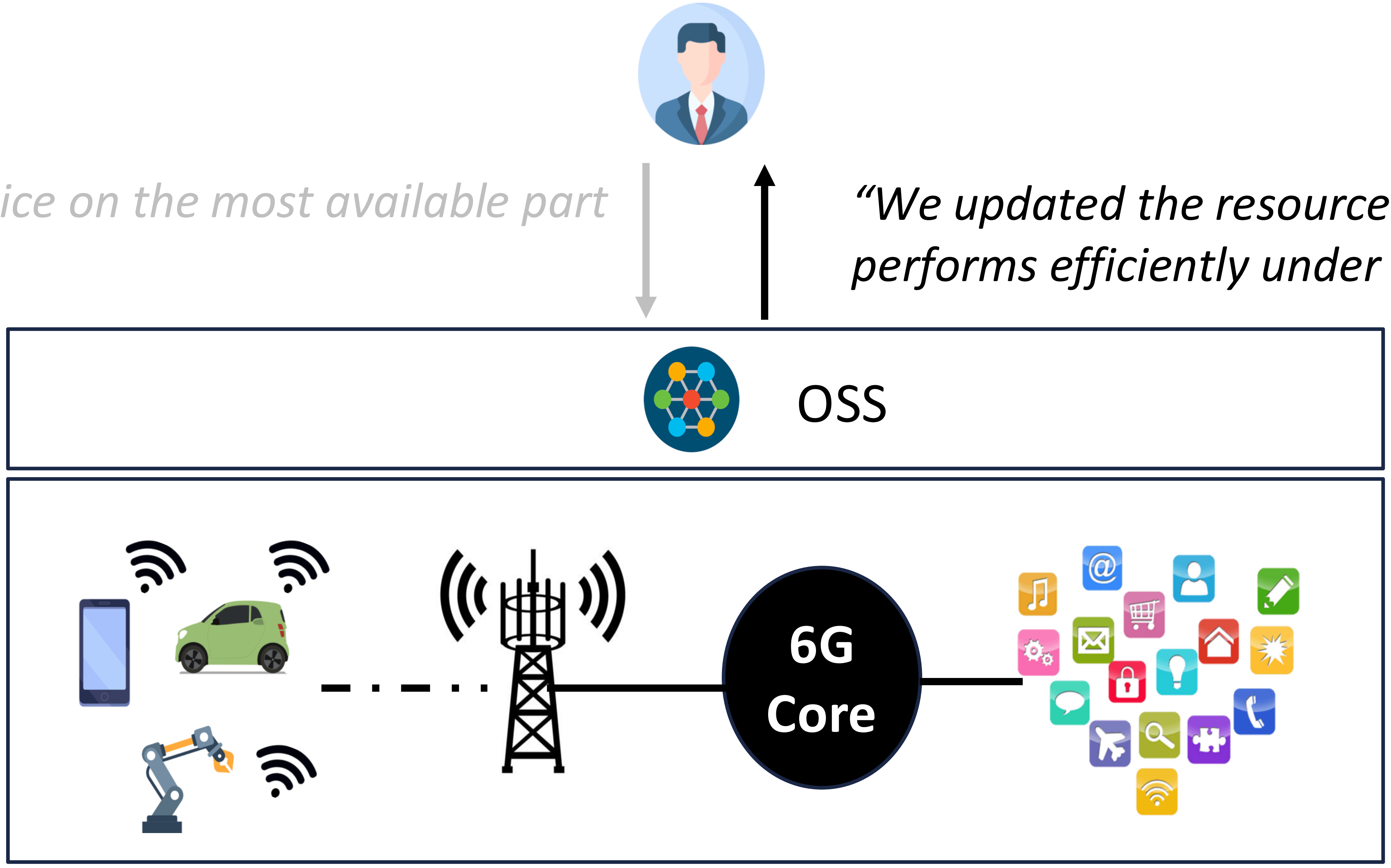


Motivation

✓ Moving towards using natural language to report the status of intents

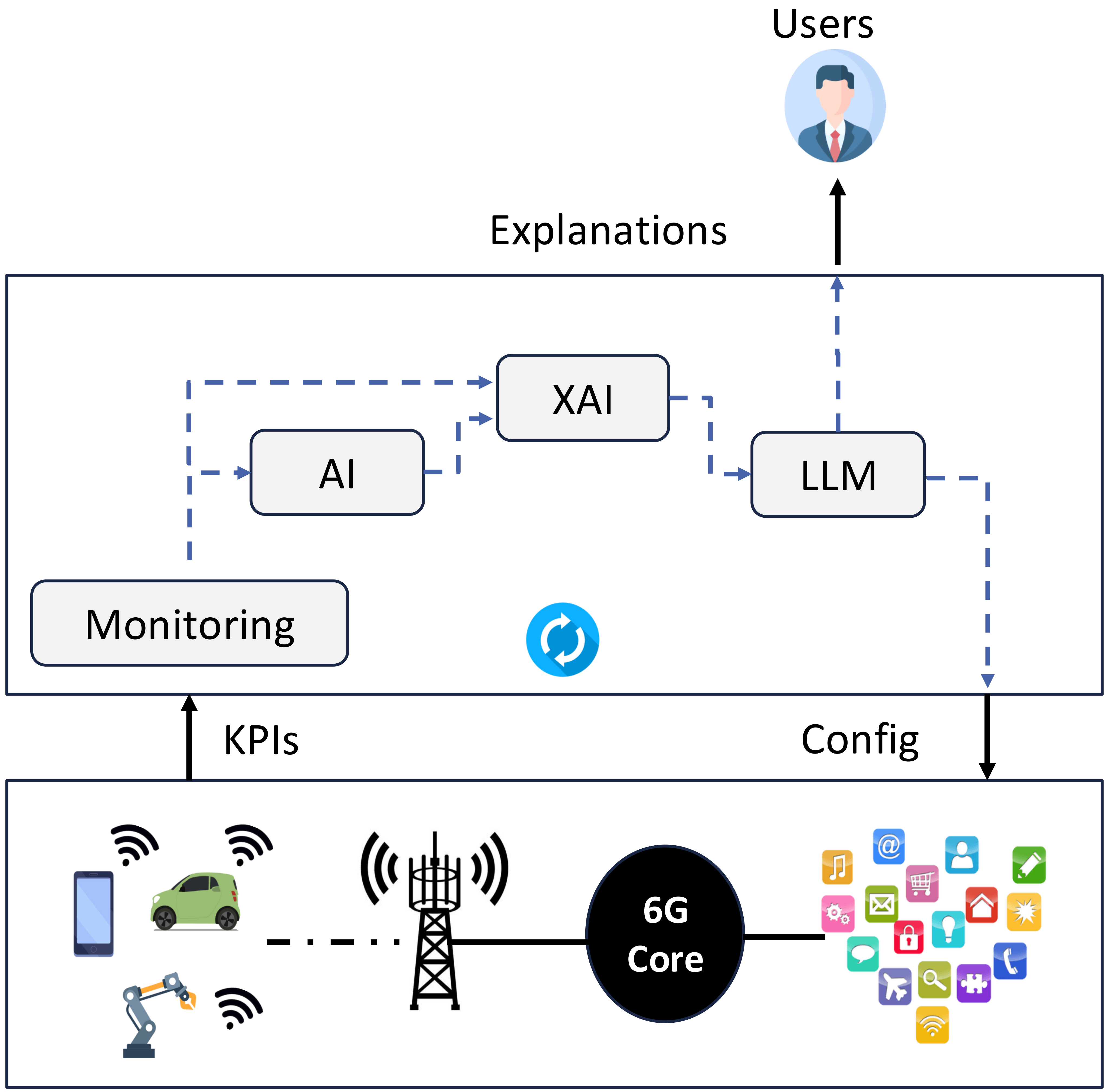
“Deploy a 5G communication service on the most available part of the virtualized infrastructure”

“We updated the resources of your XR service to ensure it performs efficiently under high user load.”



Approach

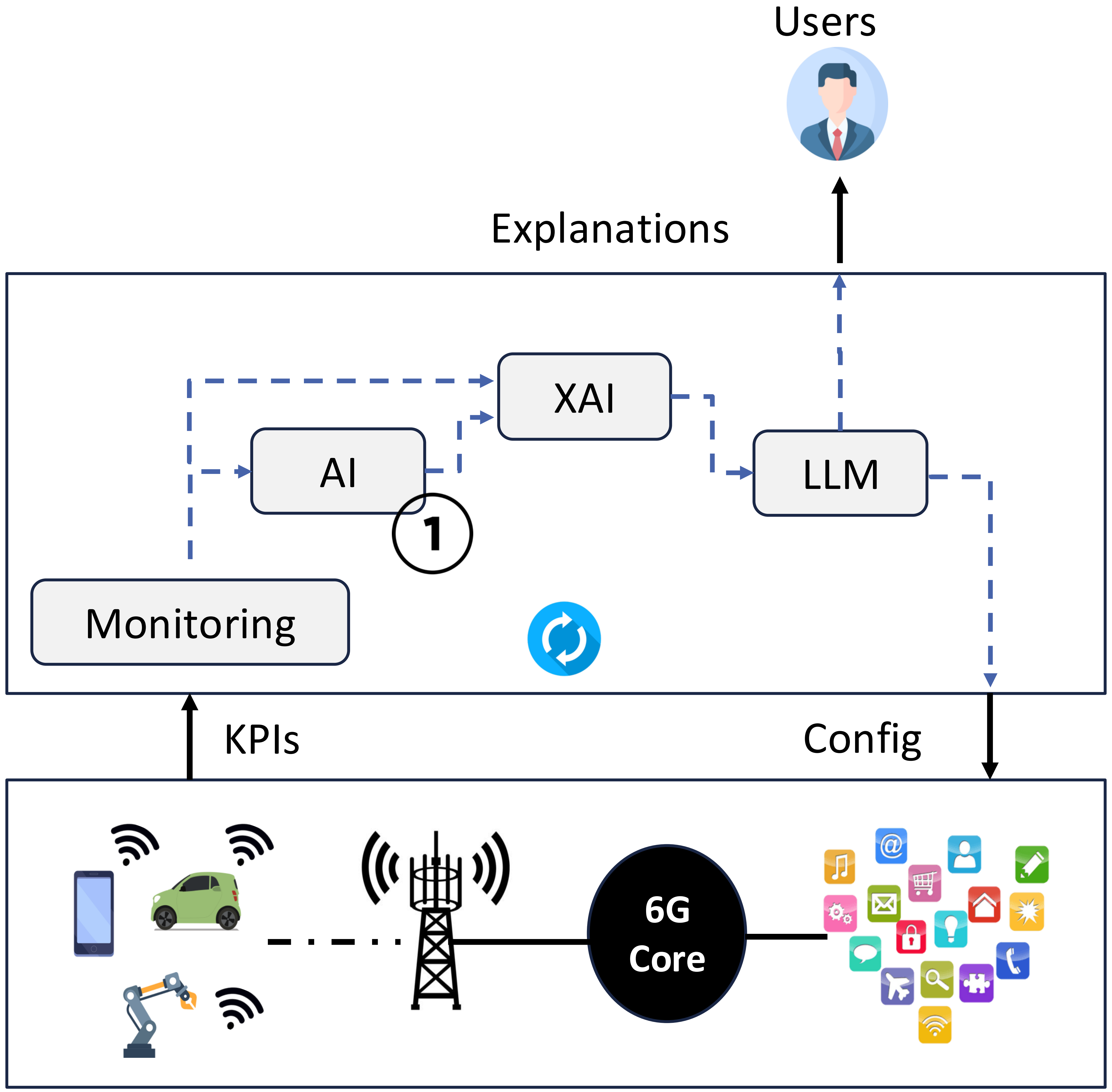
Pipeline :



Approach

Pipeline :

1 AI for anomaly detection and prediction

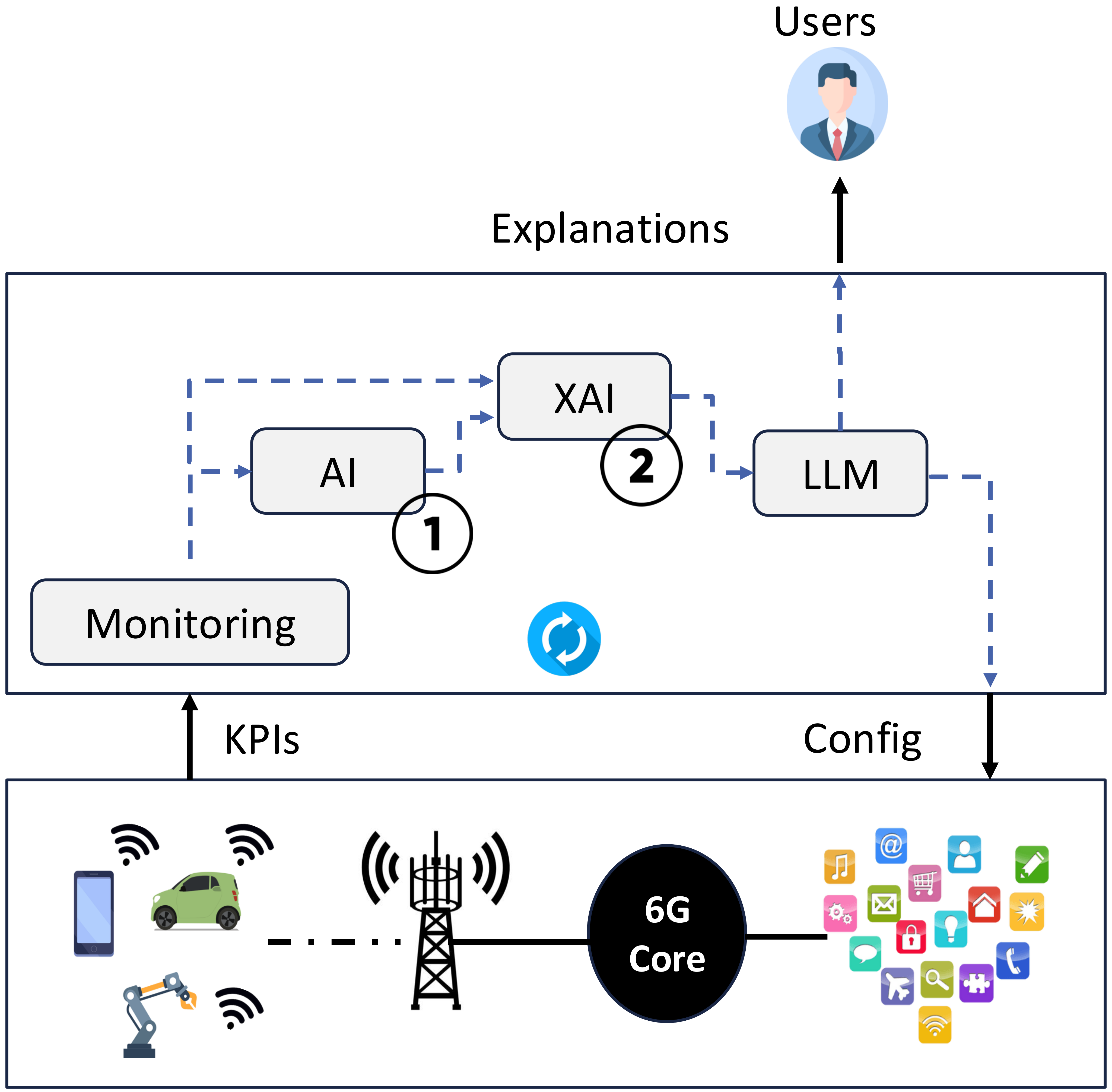


Approach

Pipeline :

1. AI for anomaly detection and prediction

2. XAI for anomaly root cause analysis



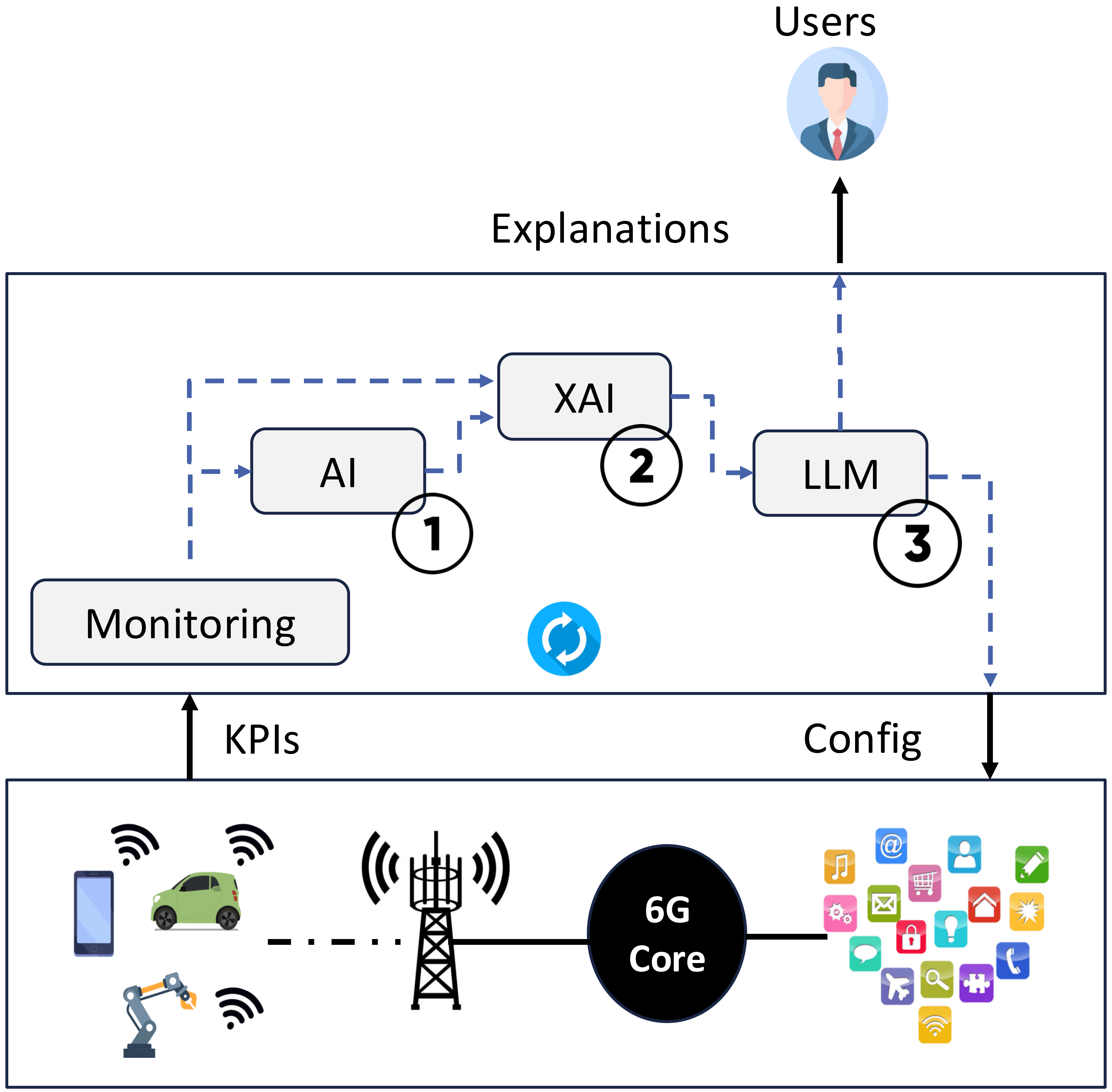
Approach

Pipeline :

1. AI for anomaly detection and prediction

2. XAI for anomaly root cause analysis

3. LLM for explaining anomalies, and providing corrective actions to resolve the anomalies, which are then applied autonomously without human intervention.



Approach

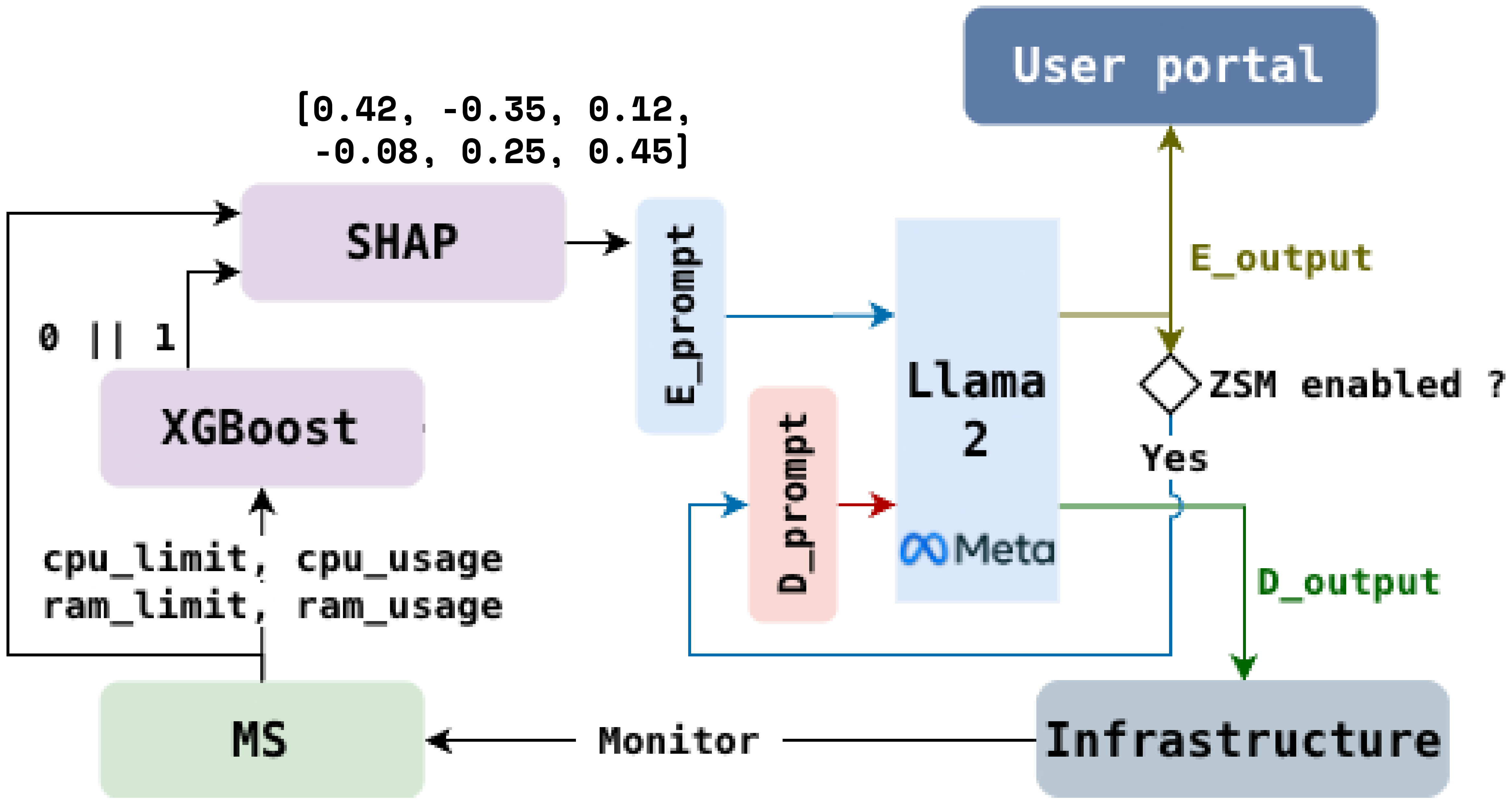
SLA: Service level Agreement
XGBoost: eXtreme Gradient Boosting
SHAP: SHapley Additive exPlanations
MS: Monitoring System

A 6G application running at the edge with strict SLA latency requirements.

XGBoost (AI) [1] is used to predict latency violations;

SHAP (XAI) [2] is applied to identify the root cause (e.g., CPU or RAM);

Llama2 (LLM) [3] provide human-understandable explanations of the violations along with corrective actions.



[1] Chen, Tianqi. "XGBoost: A Scalable Tree Boosting System." Cornell University (2016).
[2] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).
[3] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." arXiv preprint arXiv:2307.09288 (2023).

Approach

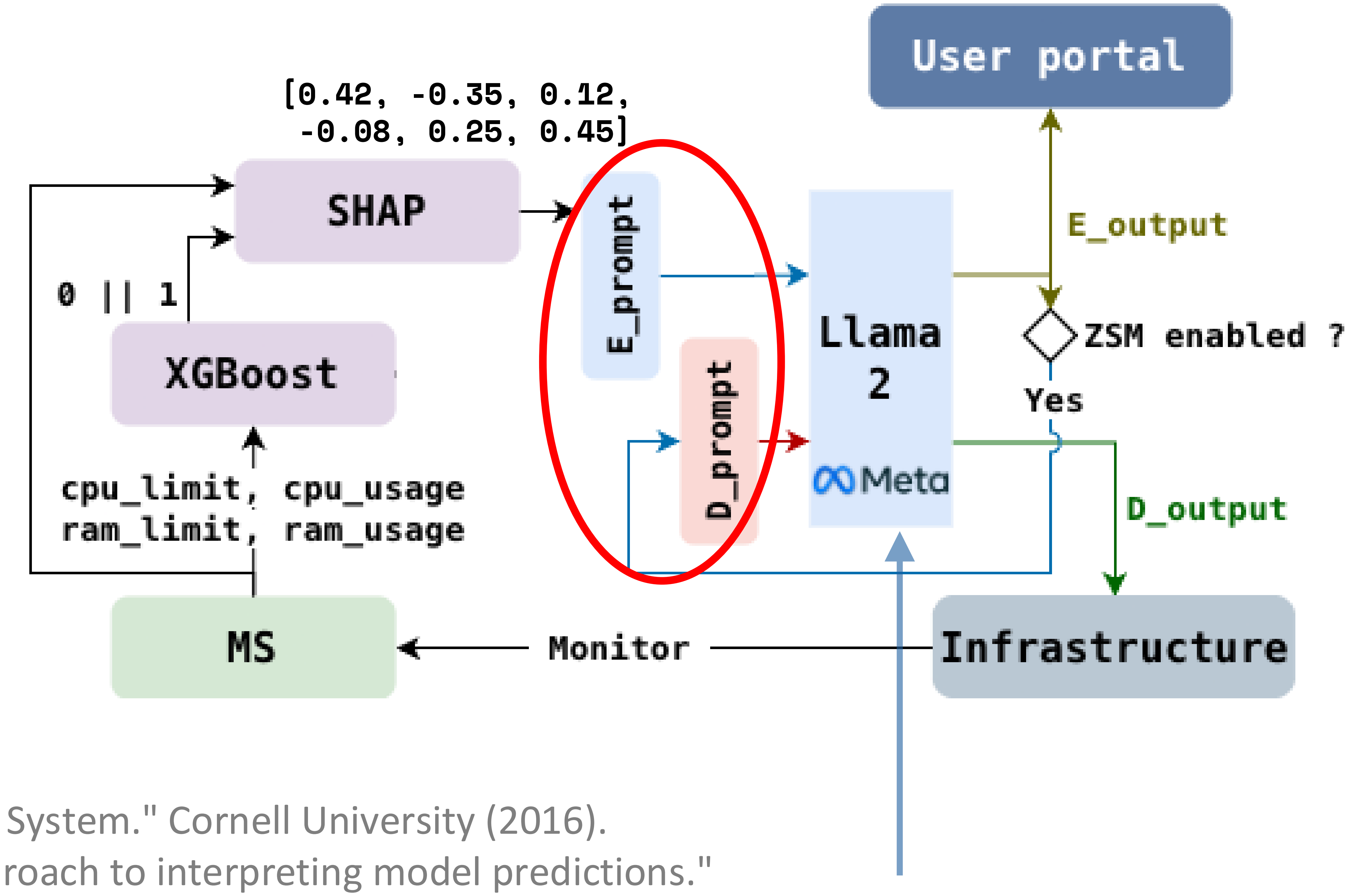
SLA: Service level Agreement
XGBoost: eXtreme Gradient Boosting
SHAP: SHapley Additive exPlanations
MS: Monitoring System

A 6G application running at the edge with strict SLA latency requirements.

XGBoost (AI) [1] is used to predict latency violations;

SHAP (XAI) [2] is applied to identify the root cause (e.g., CPU or RAM);

Llama2 (LLM) [3] provide human-understandable explanations of the violations along with corrective actions.



Trained using ICL

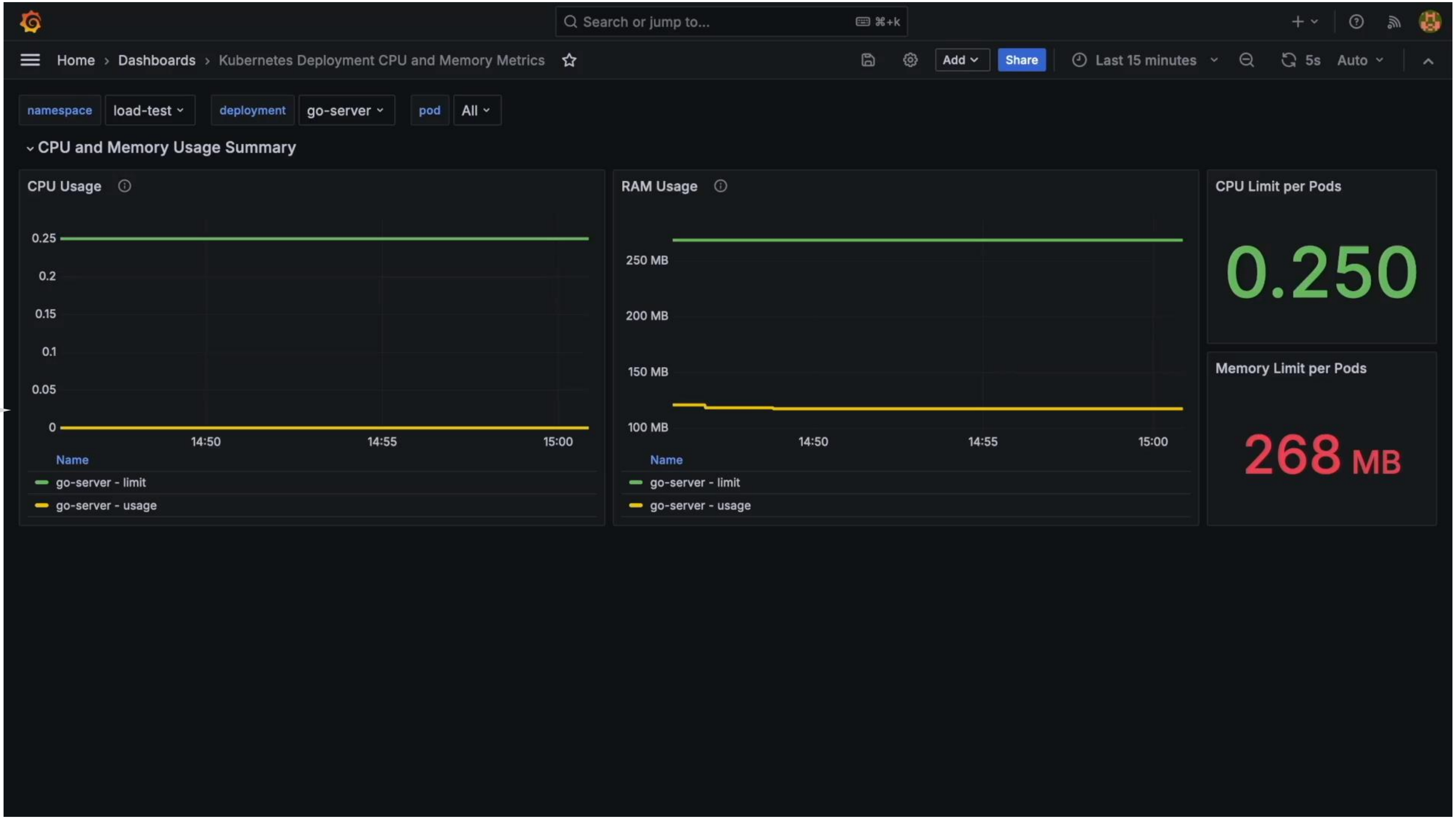
[1] Chen, Tianqi. "XGBoost: A Scalable Tree Boosting System." Cornell University (2016).
[2] Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).
[3] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." arXiv preprint arXiv:2307.09288 (2023).

Demo

- 0.25 CPU cores, 256 MB RAM for the microservice application.
- Grafana [1] and Prometheus [2] are used to visualize cloud resource KPIs related to the microservices.
- ApacheBench [3] is used to generate HTTP traffic load.
- We monitor CPU and RAM usage and limits, along with the outputs of the LLM.

[1] https://grafana.com/
[2] https://prometheus.io/
[3] https://httpd.apache.org/

Demo



Search or jump to...
⌘+k
Home › Dashboards › Kubernetes Deployment CPU and Memory Metrics ☆
Add
Share
Last 15 minutes
5s
Auto

namespace: load-test
deployment: go-server
pod: All

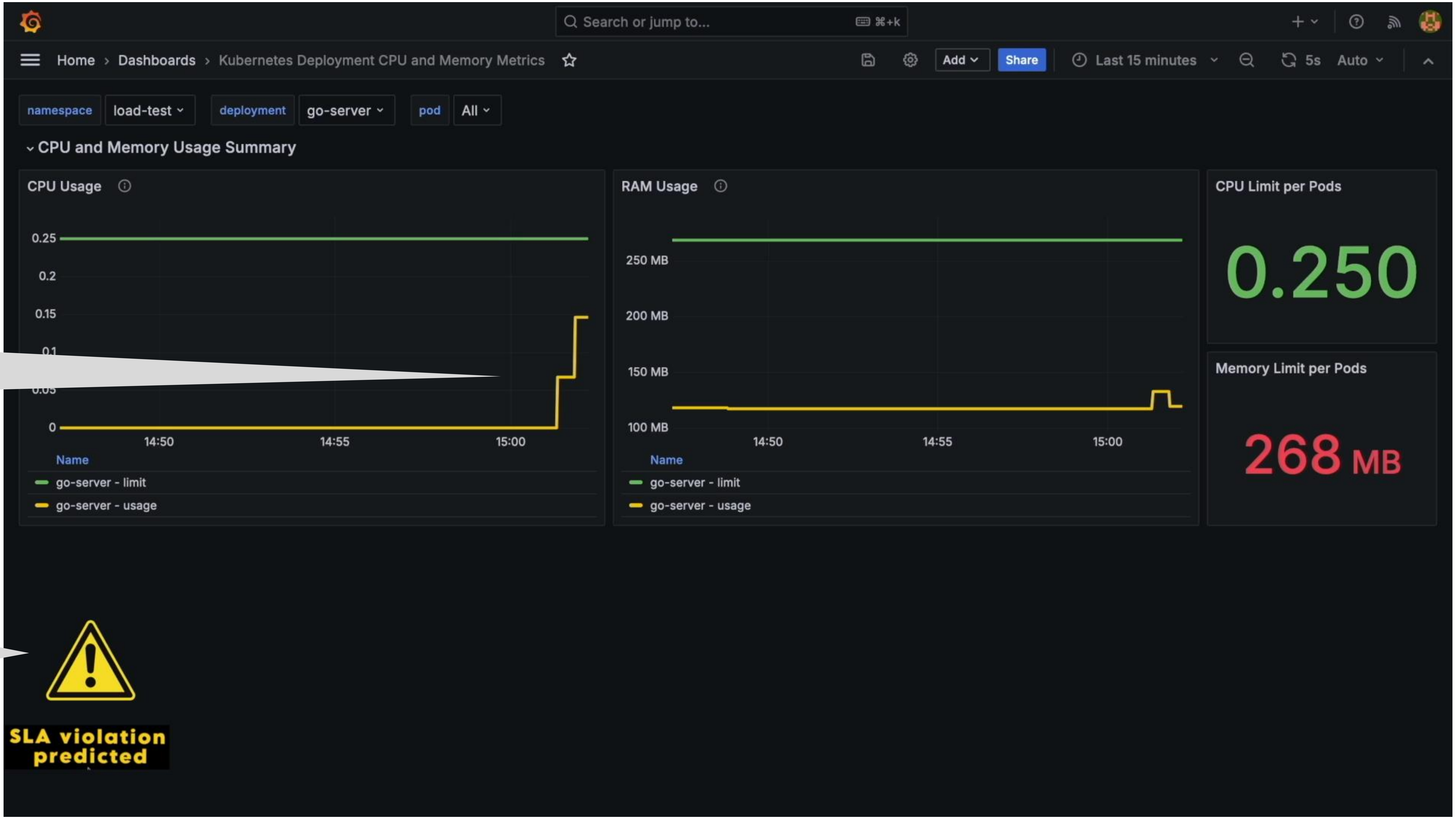
CPU and Memory Usage Summary

CPU Limit per Pods: 0.250

Memory Limit per Pods: 268 MB

Grafana dashboard

Demo



namespace: load-test

deployment: go-server

pod: All

CPU and Memory Usage Summary

CPU Limit per Pods: 0.250

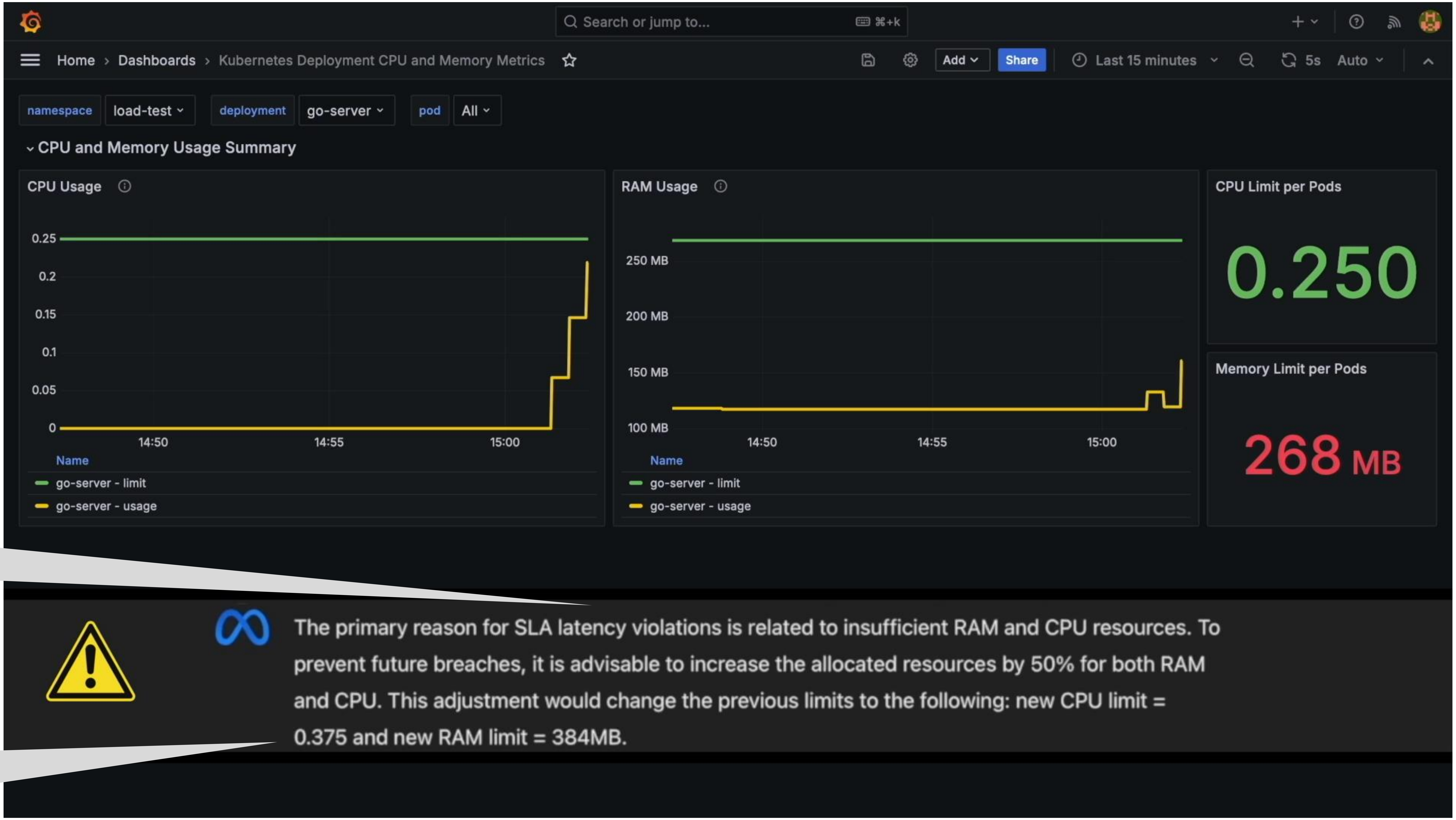
Memory Limit per Pods: 268 MB

SLA violation predicted

We stress the application with a load of HTTP requests

XGBoost predicted SLA latency violation

Demo

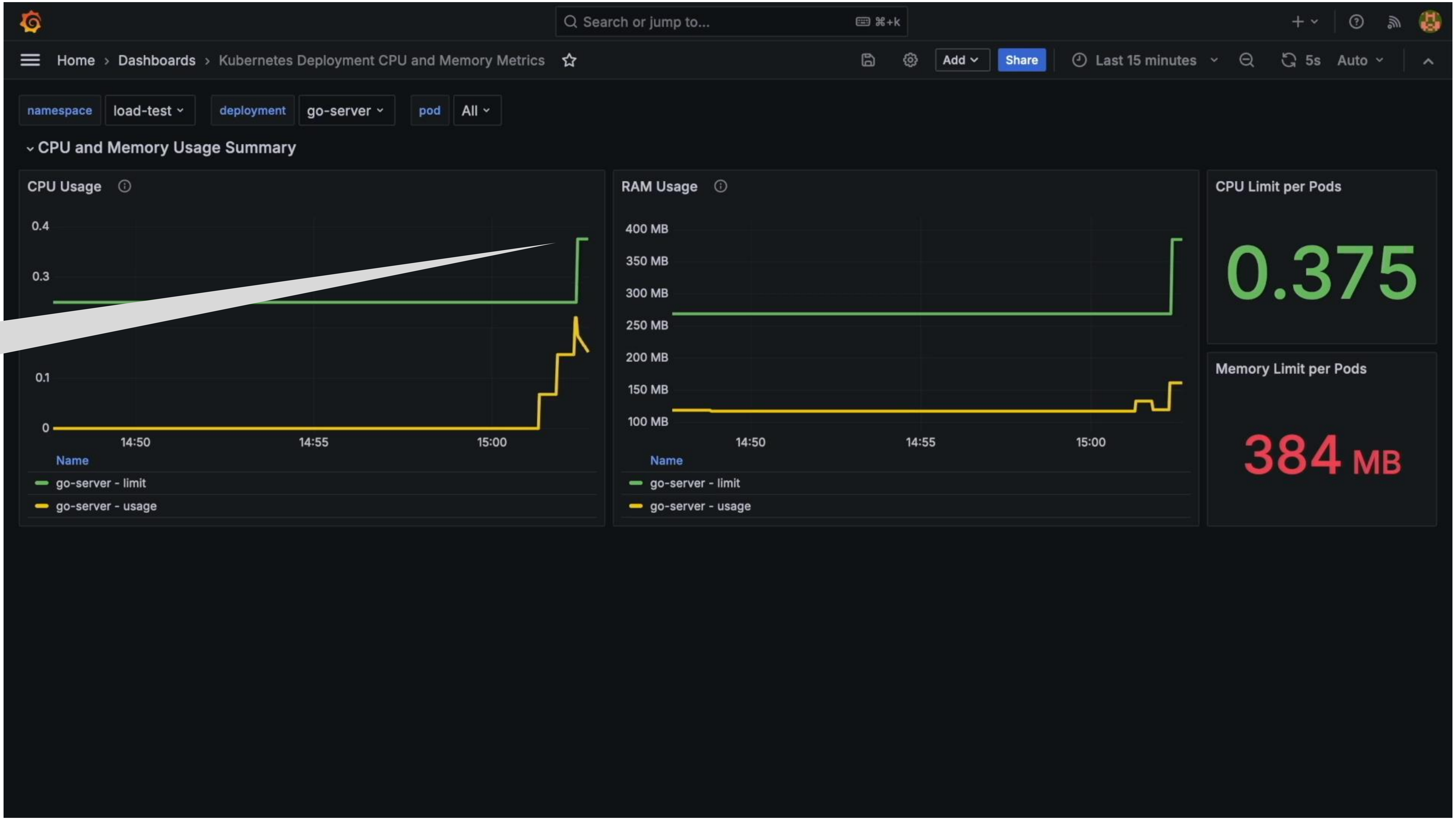


The primary reason for SLA latency violations is related to insufficient RAM and CPU resources. To prevent future breaches, it is advisable to increase the allocated resources by 50% for both RAM and CPU. This adjustment would change the previous limits to the following: new CPU limit = 0.375 and new RAM limit = 384MB.

The root cause is insufficient RAM and CPU resources

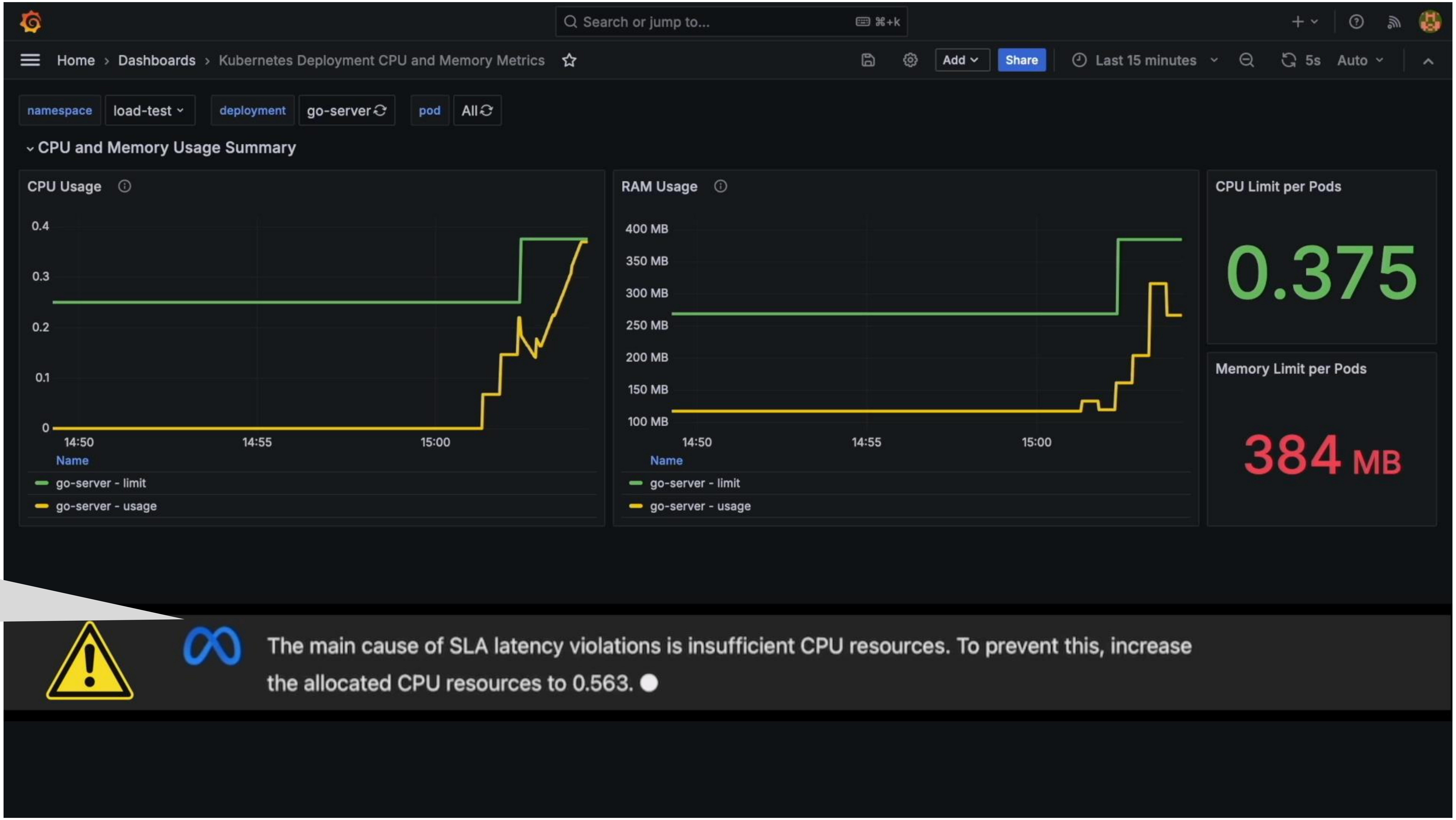
Decision: Increase both CPU and RAM to higher values

Demo



The new allocation was applied without human intervention.

Demo



namespace load-test

deployment go-server

pod All

CPU and Memory Usage Summary

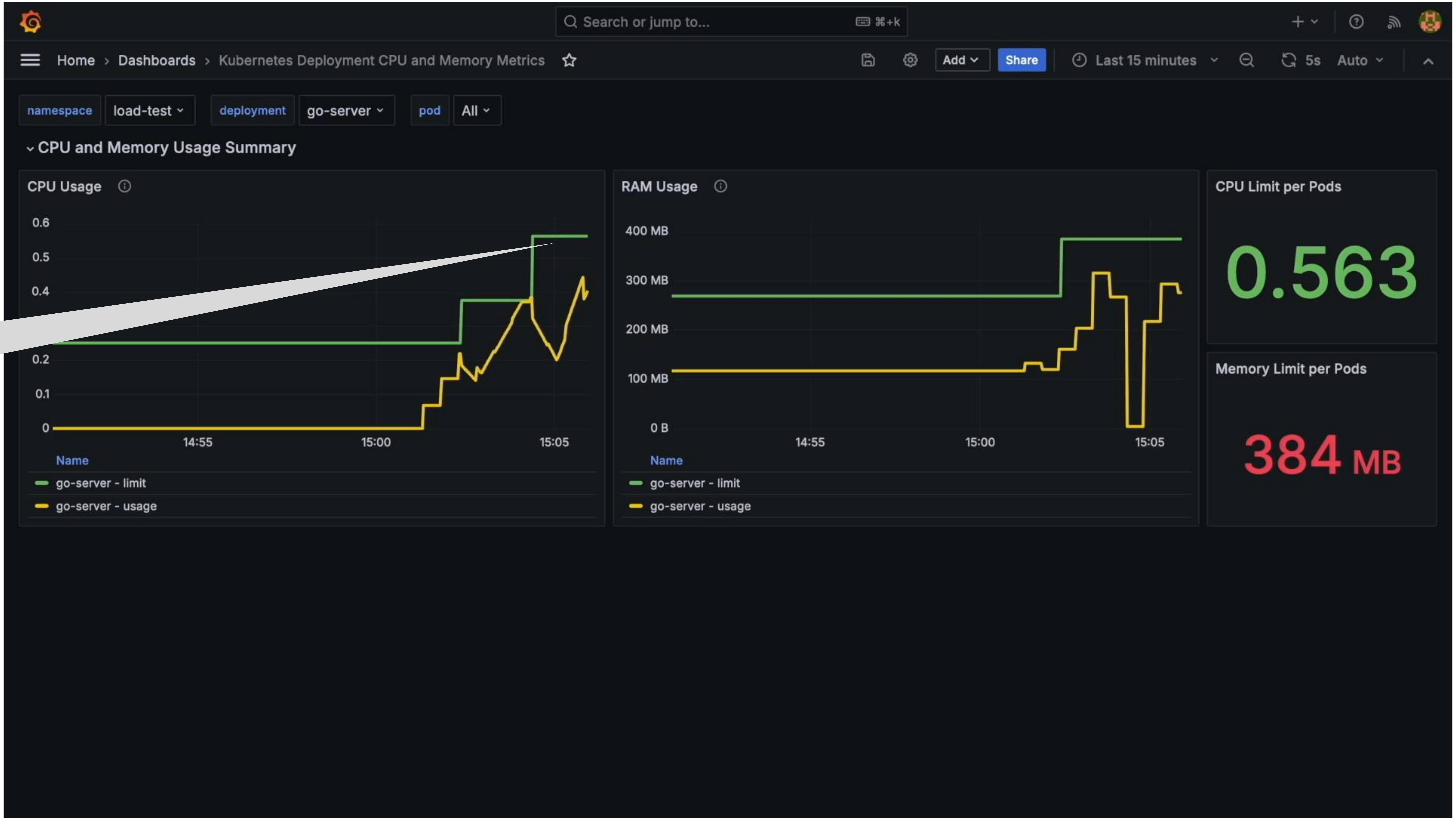
CPU Limit per Pods 0.375

Memory Limit per Pods 384 MB

The main cause of SLA latency violations is insufficient CPU resources. To prevent this, increase the allocated CPU resources to 0.563. ●

XGBoost predicted an SLA violation, but this time the root cause was solely insufficient CPU resources.

Demo

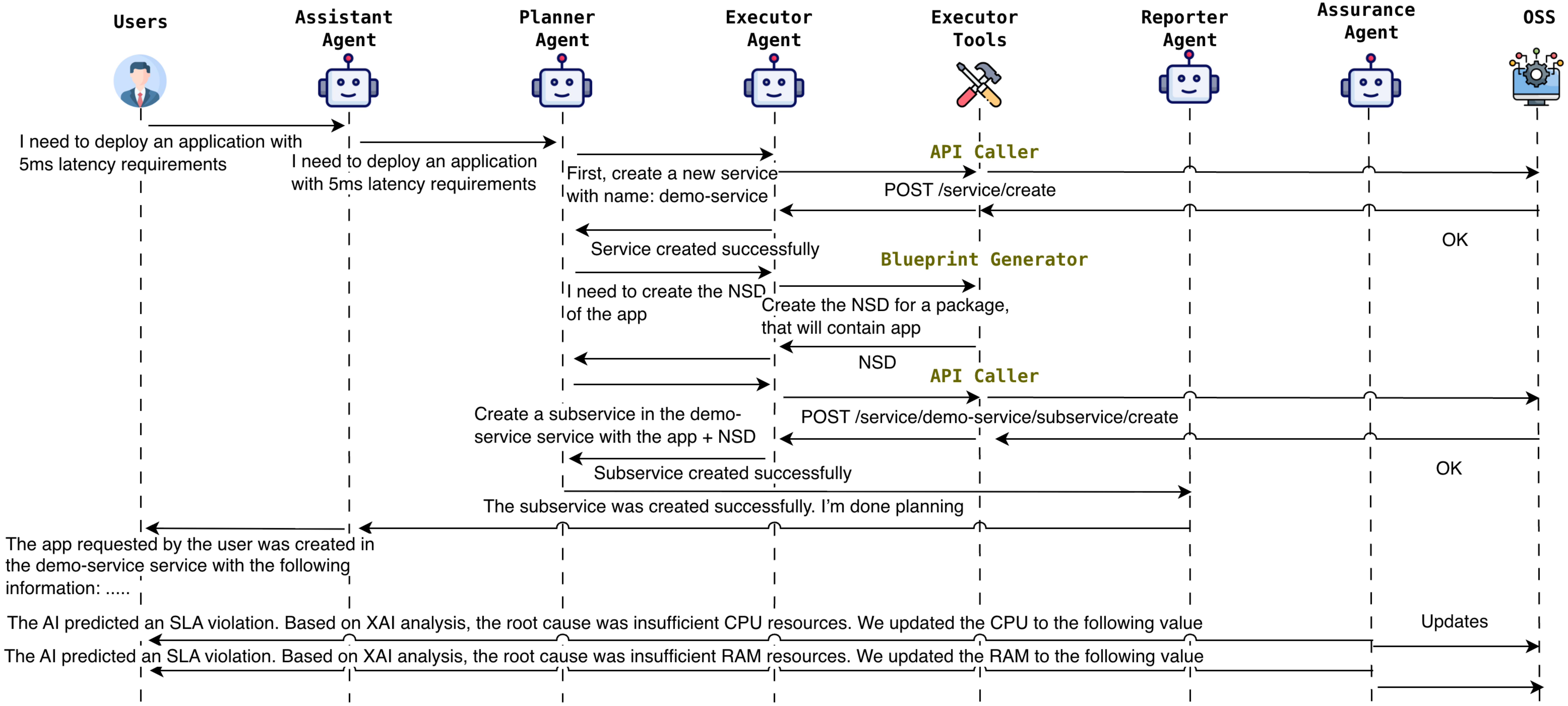


The new allocation was applied without human intervention.

Table of Contents

- Intent-Based Networking (IBN)
- Intent Translation with LLMs
- Intent Assurance with LLMs
- Conclusion

Conclusion



Users

Assistant Agent

Planner Agent

Executor Agent

Executor Tools

Reporter Agent

Assurance Agent

OSS

I need to deploy an application with 5ms latency requirements

I need to deploy an application with 5ms latency requirements

First, create a new service with name: demo-service

API Caller

POST /service/create

OK

Service created successfully

Blueprint Generator

I need to create the NSD of the app

Create the NSD for a package, that will contain app

NSD

API Caller

Create a subservice in the demo-service service with the app + NSD

POST /service/demo-service/subservice/create

OK

Subservice created successfully

The subservice was created successfully. I'm done planning

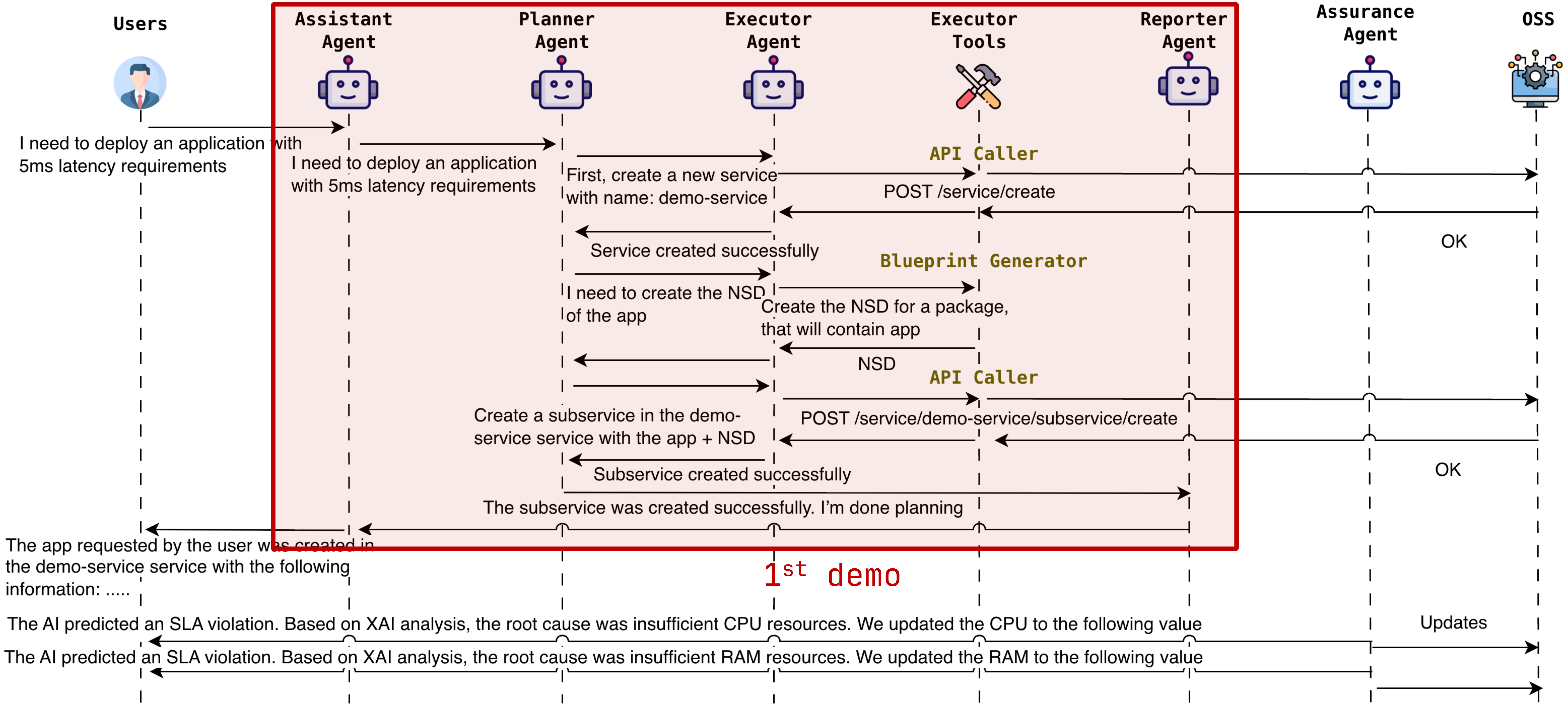
The app requested by the user was created in the demo-service service with the following information: .....

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient CPU resources. We updated the CPU to the following value

Updates

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient RAM resources. We updated the RAM to the following value

Conclusion



Users

Assistant Agent

Planner Agent

Executor Agent

Executor Tools

Reporter Agent

Assurance Agent

OSS

I need to deploy an application with 5ms latency requirements

I need to deploy an application with 5ms latency requirements

First, create a new service with name: demo-service

API Caller

POST /service/create

OK

Service created successfully

Blueprint Generator

I need to create the NSD of the app

Create the NSD for a package, that will contain app

NSD

API Caller

Create a subservice in the demo-service service with the app + NSD

POST /service/demo-service/subservice/create

OK

Subservice created successfully

The subservice was created successfully. I'm done planning

The app requested by the user was created in the demo-service service with the following information: .....

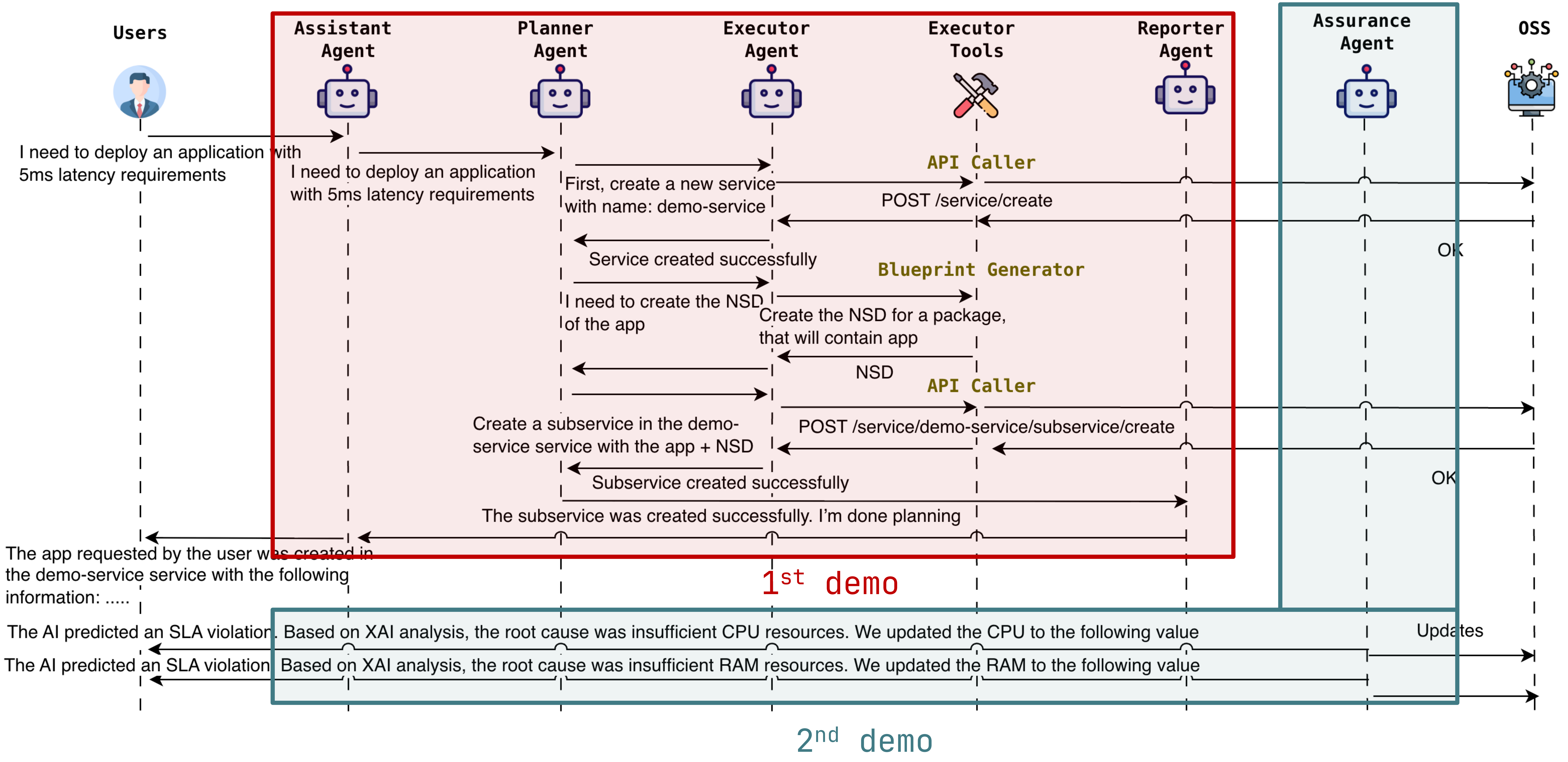
1st demo

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient CPU resources. We updated the CPU to the following value

Updates

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient RAM resources. We updated the RAM to the following value

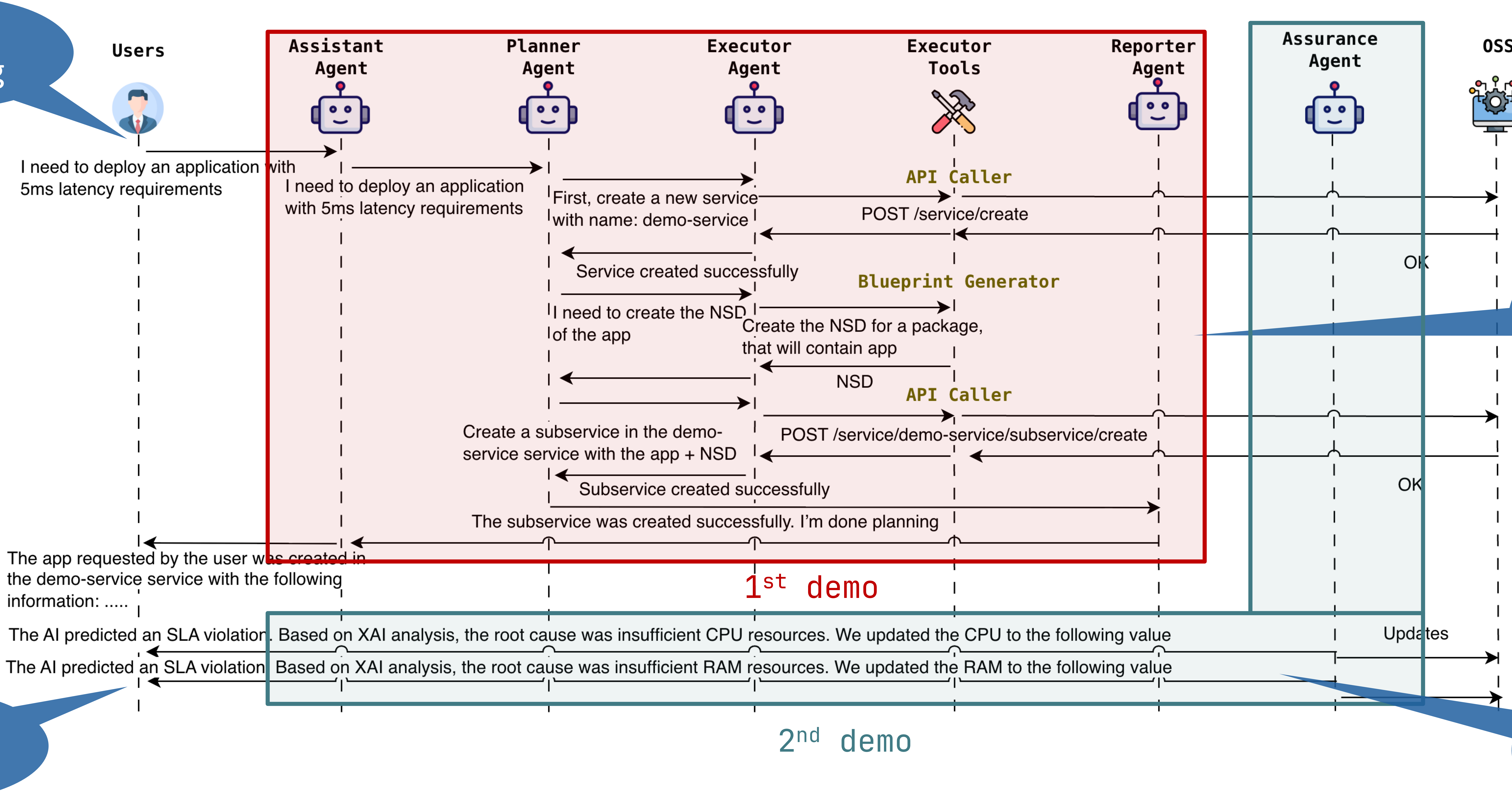
Conclusion



Users
Assistant Agent
Planner Agent
Executor Agent
Executor Tools
Reporter Agent
Assurance Agent
OSS
I need to deploy an application with 5ms latency requirements
I need to deploy an application with 5ms latency requirements
First, create a new service with name: demo-service
API Caller
POST /service/create
OK
Service created successfully
Blueprint Generator
I need to create the NSD of the app
Create the NSD for a package, that will contain app
NSD
API Caller
Create a subservice in the demo-service service with the app + NSD
POST /service/demo-service/subservice/create
OK
Subservice created successfully
The subservice was created successfully. I'm done planning
The app requested by the user was created in the demo-service service with the following information: .....
1st demo
The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient CPU resources. We updated the CPU to the following value
Updates
The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient RAM resources. We updated the RAM to the following value
2nd demo

Conclusion

Intent Profiling



Users

Assistant Agent

Planner Agent

Executor Agent

Executor Tools

Reporter Agent

Assurance Agent

OSS

I need to deploy an application with 5ms latency requirements

I need to deploy an application with 5ms latency requirements

First, create a new service with name: demo-service

API Caller

POST /service/create

OK

Service created successfully

Blueprint Generator

I need to create the NSD of the app

Create the NSD for a package, that will contain app

NSD

API Caller

Create a subservice in the demo-service service with the app + NSD

POST /service/demo-service/subservice/create

OK

Subservice created successfully

The subservice was created successfully. I'm done planning

The app requested by the user was created in the demo-service service with the following information: .....

1st demo

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient CPU resources. We updated the CPU to the following value

Updates

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient RAM resources. We updated the RAM to the following value

2nd demo

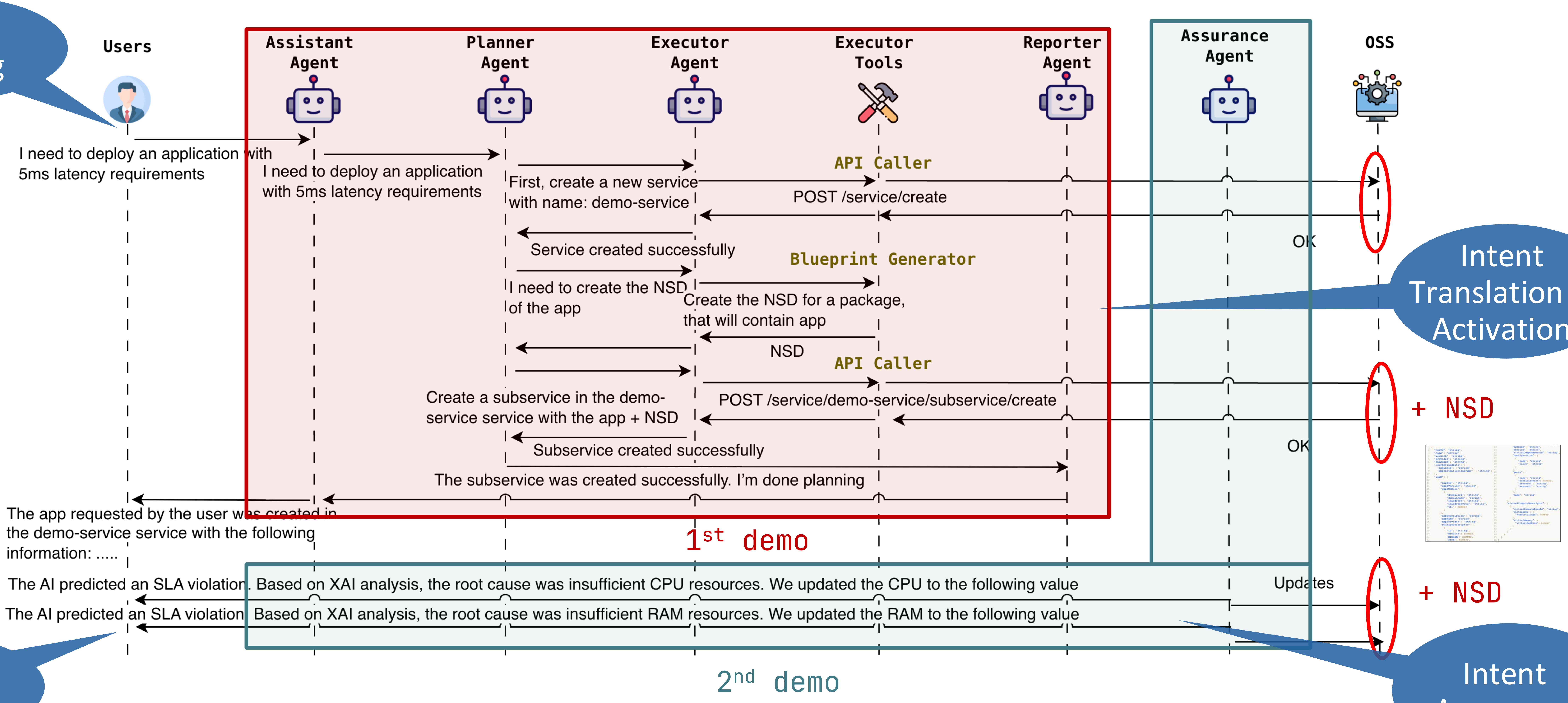
Intent Translation & Activation

Intent Reporting

Intent Assurance

Conclusion

Intent Profiling



Users

Assistant Agent

Planner Agent

Executor Agent

Executor Tools

Reporter Agent

Assurance Agent

OSS

I need to deploy an application with 5ms latency requirements

I need to deploy an application with 5ms latency requirements

First, create a new service with name: demo-service

API Caller

POST /service/create

OK

Service created successfully

Blueprint Generator

I need to create the NSD of the app

Create the NSD for a package, that will contain app

NSD

API Caller

Create a subservice in the demo-service service with the app + NSD

POST /service/demo-service/subservice/create

+ NSD

OK

Subservice created successfully

The subservice was created successfully. I'm done planning

The app requested by the user was created in the demo-service service with the following information: .....

1st demo

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient CPU resources. We updated the CPU to the following value

Updates

+ NSD

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient RAM resources. We updated the RAM to the following value

2nd demo

Intent Translation & Activation

Intent Assurance

Intent Reporting

Conclusion

Intent Profiling

Users

Assistant Agent

Planner Agent

Executor Agent

Executor Tools

Reporter Agent

Assurance Agent

OSS

I need to deploy an application with 5ms latency requirements

I need to deploy an application with 5ms latency requirements

First, create a new service with name: demo-service

API Caller

POST /service/create

OK

Service created successfully

Blueprint Generator

I need to create the NSD of the app

Create the NSD for a package, that will contain app

NSD

Intent Translation & Activation

API Caller

Create a subservice in the demo-service service with the app + NSD

POST /service/demo-service/subservice/create

+ NSD

OK

Subservice created successfully

The subservice was created successfully. I'm done planning

The app requested by the user was created in the demo-service service with the following information: .....

1st demo

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient CPU resources. We updated the CPU to the following value

Updates

+ NSD

The AI predicted an SLA violation. Based on XAI analysis, the root cause was insufficient RAM resources. We updated the RAM to the following value

Intent Reporting

2nd demo

Intent Assurance

Conclusion

Intent Translation

- Mekrache, Abdelkader, Adlen Ksentini, and Christos Verikoukis. “Demo: Next-Generation Network Management with OSS-GPT”. 2025 ACM SIGCOMM. Coimbra, Portugal.
- Mekrache, Abdelkader, Adlen Ksentini, and Christos Verikoukis. "OSS-GPT: An LLM-Powered Intent-Driven Operations Support System for 6G Networks." *2025 IEEE 11th International Conference on Network Softwarization (NetSoft)*. IEEE, 2025.
- ▶ Demo: https://www.youtube.com/watch?v=A1tTyHhyT80

Intent Assurance

- Mekrache, Abdelkader, et al. "On combining XAI and LLMs for trustworthy zero-touch network and service management in 6G." IEEE Communications Magazine 63.4 (2024): 154-160.
- ▶ Demo: https://www.youtube.com/watch?v=CtesBPSgT3c

Thank You

Agentic AI for Intent-Based Network Management

Abdelkader Mekrache, Adlen Ksentini

abdelkader.mekrache@eurecom.fr
adlen.ksentini@eurecom.fr