Deployment of AI-Agents in OSM

Ignacio Labrador (Ignacio.Labrador@atos.net)
Luis Gomez (luis.gomez.external@atos.net)
Auto-Scaling – Today.

- Simple metrics (cpu/ram/network)
- Simple and static thresholds to trigger alarms
- Reactive scaling
- AI-Agents using HELM-Charts is running in VNF Execution Environments.
- AI-Agent triggers automatically VNFs scaling in-out actions making possible to address complex data classification or prediction problems (e.g., based on CPU, RAM or network usage metrics) according to the concrete AI-models in the Server.
**AI model example Use Case**

**VNF Scaling Actions are triggered based on the AI-model response about traffic images classification:**

- Al-Model: Convolutional Neural Network.
- Data Set: 180 images with different framing and lighting conditions.
- Data Source: Public traffic camera from the Spanish Traffic Management Authority (DGT).

- **Upon High-density traffic detected → VNF Scale-out action**
- **Upon smooth traffic detected → VNF Scale-in action**
Use Case Implementation

1. **MODEL DEVELOPMENT**
   - Google colab + Google Drive
   - TensorFlow
   - Keras
   - Development Environment

2. **MODEL DEPLOYMENT**
   - AI Models Server
   - Tensorflow Serving
   - openstack

3. **MODEL SERVING**
   - Response
     - Traffic Jam / Smooth
   - Scaling Action
     - NBI
     - VNF
     - AI-Agent

- Scaling Request
  - Traffic Images
  - Images Server
  - Scaling Action
  - NBI
  - VNF
  - Images Server

- Request
  - (Evaluate traffic condition)

- Use Case Implementation

© ETSI
AI-Agent config, structure & logic

Helm Config

AI-Agent Helm Chart
- Cron windows to launch predictions periodically
- Hot change settings
- Docker containers
- Launch AI Agents actions

<table>
<thead>
<tr>
<th>Chart.yaml</th>
<th>values.yaml</th>
<th>\templates folder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Versioning</td>
<td>AI Agent config</td>
<td>cronjob.yaml Deployment file</td>
</tr>
</tbody>
</table>

VNF structure

vnf.tar.gz
- vnf.yaml
- checksums
- helm-charts/
  - eechart/
    - Chart.yaml
    - Values.yaml
    - templates/...
  - charts/
    - ai-agent
      - Chart.yaml
      - Values.yaml
      - templates/
      - cronjob.yaml
Helm Descriptor: configuring an AI-Agent

**Deployment**

- **jobs:**
  - **name**: ai-agent
  - **image**: repository: luisupm/ai-agent
tag: latest
  - **imagePullPolicy**: Always
- **schedule**: "* * * * *
- **failedJobsHistoryLimit**: 1
- **successfulJobsHistoryLimit**: 3
- **concurrencyPolicy**: Allow
- **restartPolicy**: OnFailure

**AI-Agent Conf**

# Configuration for AI-Agent

- **config**:
  - **version**: 1

# Select the AI Server to request

- **AIServer**:
  - **name**: TensorFlow
  - **type**: tensorflow
  - **url**: "http://192.168.137.46:8501"
  - **version**: v1

**Proactive monitoring**

- **predictions**:
  - # List of predictions
  - **model**: A2-traffic-model
  - **active**: True

- **monitoring**:
  - **url**: 192.168.137.34 # URL, IP or just vdu_name
  - **port**: 3000 # Optional

- **threshold**:
  - **function_name**: evaluator
  - **logic**: "evaluator = lambda x: True if (x['predictions'][0][0] > x['predictions'][0][1]) else False "

# External Sources

- ../ai-agent/values.yaml
- AI model Definition
- Prediction enabled
- Monitoring Data
- Threshold data
Key Takeaways.

• The PoC demonstrates an additional usage of the Execution Environment feature released in OSM8.

• Integration AI-Agents in OSM allows to act in a proactive way before a mitigation action is required.

• The new AI-Agents model is compatible with the current monitoring system in OSM.

• Possibility to integrate heterogeneous data sources (infrastructure metrics, NS data and external) to reach automated ambitious decisions.

• Open Solution:
  • It enables to deploy different ML models defined by the NS designer.
  • Possible to use different AI/ML frameworks at the OSM user's discretion.

• The solution provides a scalable architecture (multiple AI-Agents and ML models can be deployed to satisfy different requirements).
Future Work

• Further integration with the OSM architecture:
  • Integration with the native OSM alarms flow instead direct interface to NBI.
  • Integration with the OSM console and UI.

• Integration of OSM time-series data sources.

• Include support for other closed loop ML models (e.g., reinforcement learning).

• AI-Models catalogue (provide the possibility to select ready-to-use AI models from a catalogue for easier use).

• Other ML models to be considered so the scaling actions are based on e.g. CPU usage, RAM or network usage metrics.
