Cisco
The bridge to possible

Cisco DevNet

FOSDEM

Drag, Drop, and Deploy
Low-Code AI Agents for Network Ops

Alfonso Sandoval Rosas
Developer Advocate

```
!
username Alfonso-(Poncho)-Sandoval
!
role developer title "Developer Avocado"
organization "Cisco Systems Portugal" PT
!
interface LinkedIn0/0
 ip address linkedin.com/in/asandovalros
 no shutdown
!
interface GitHub0/1
 ip address github.com/ponchotitlan
 no shutdown
!
end
!
```

Cisco Office in Tokyo, Japan JP

Agenda

AgenticOps in the World of Networking

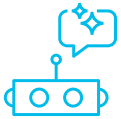
An AgenticOps Open Ecosystem, brick-by-brick

Demo 1: Agentic ChatOps for NetAuto

Demo 2: Agentic Reporting & Ticketing for NetAuto

Wrap-up

AgenticOps in the World of Networking



Speaking CLI

Modern LLMs arrive with surprisingly deep prior exposure to real-world network CLIs across vendors, so they can generate command sequences without prior tuning

Reasoning about intent

Given a high-level intent (e.g. “check interface health”), an LLM can decompose it into a structured sequence of diagnostic steps and commands

Low-code tools available

Mature low-code frameworks enable agentic automation, reducing boilerplate and enabling guardrails, while still requiring explicit logic

An AgenticOps Open Ecosystem, brick-by-brick

Ollama
(Local LLM runtime)

Open-source runtime for running LLMs locally, enabling self-hosted, data-local AI

n8n

n8n (Low-code orchestration)

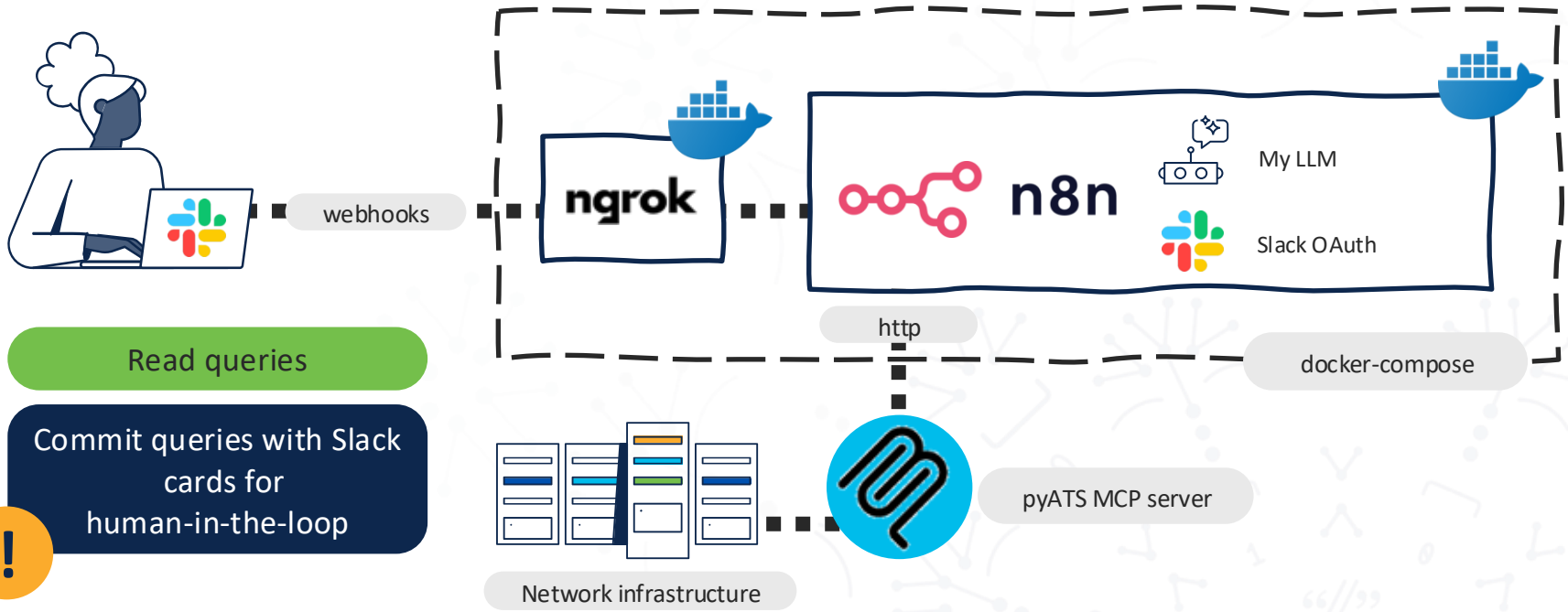
Open-core, self-hosted workflow engine for auditable, guardrailed agent workflows

+ pyATS
print(network.profile)

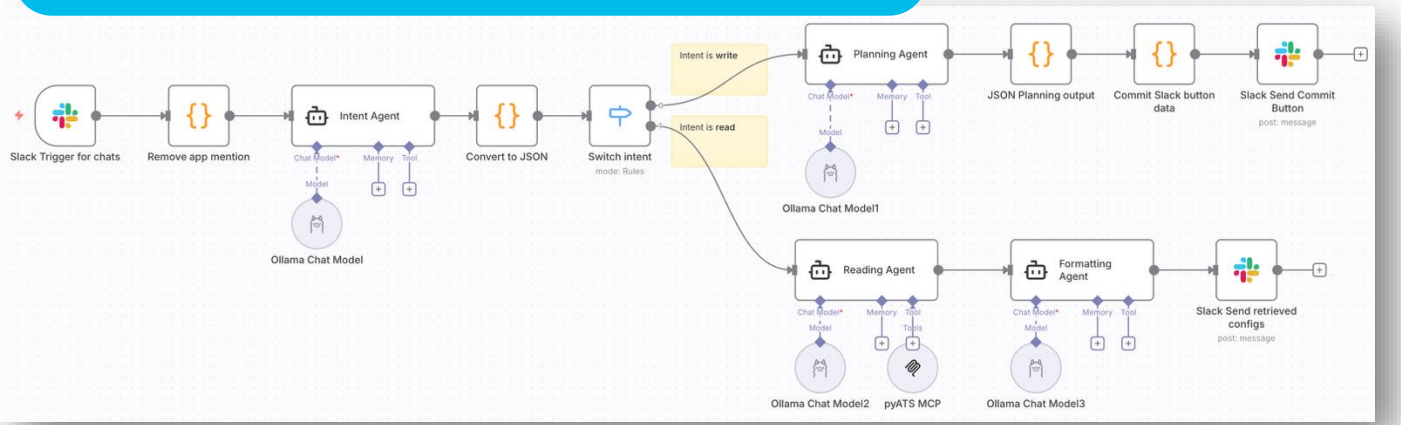
MCP server (based on pyATS framework)

pyATS-powered control plane enforcing structured, safe read/write access to network

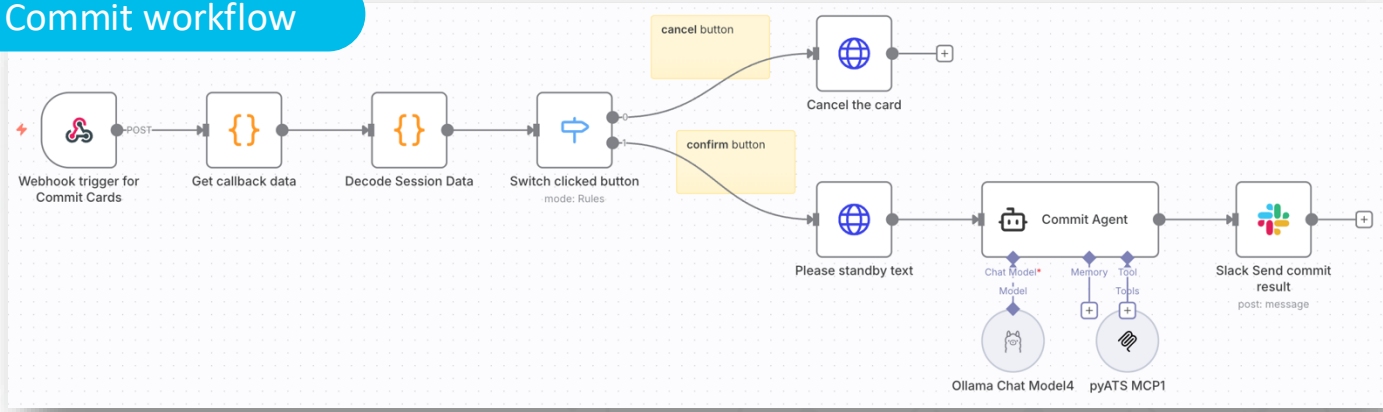
Demo 1: Agentic ChatOps for NetAuto



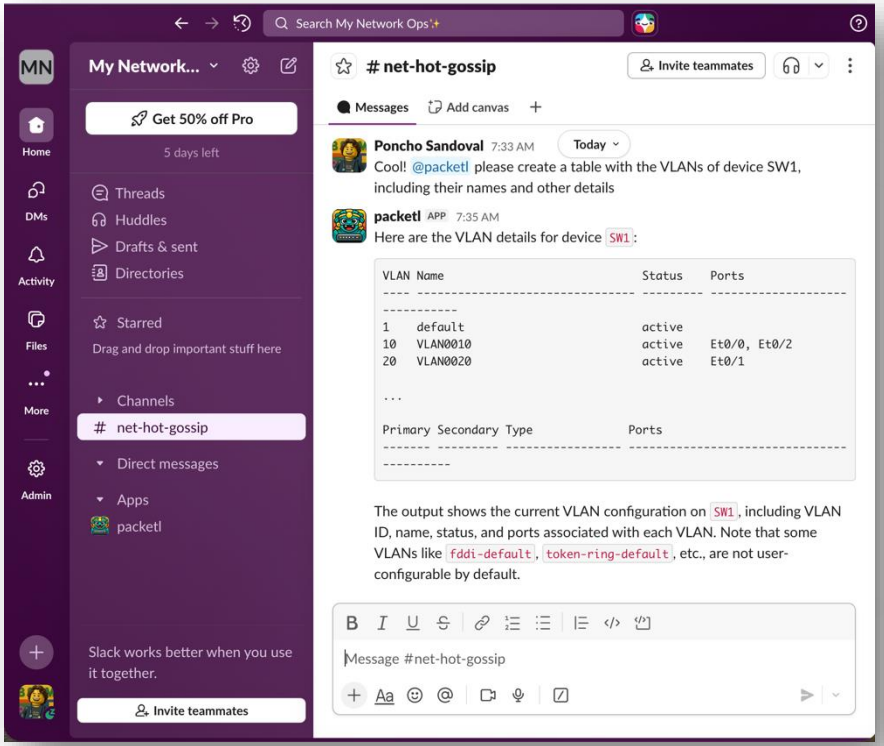
Reading + Commit intent (planning) workflow



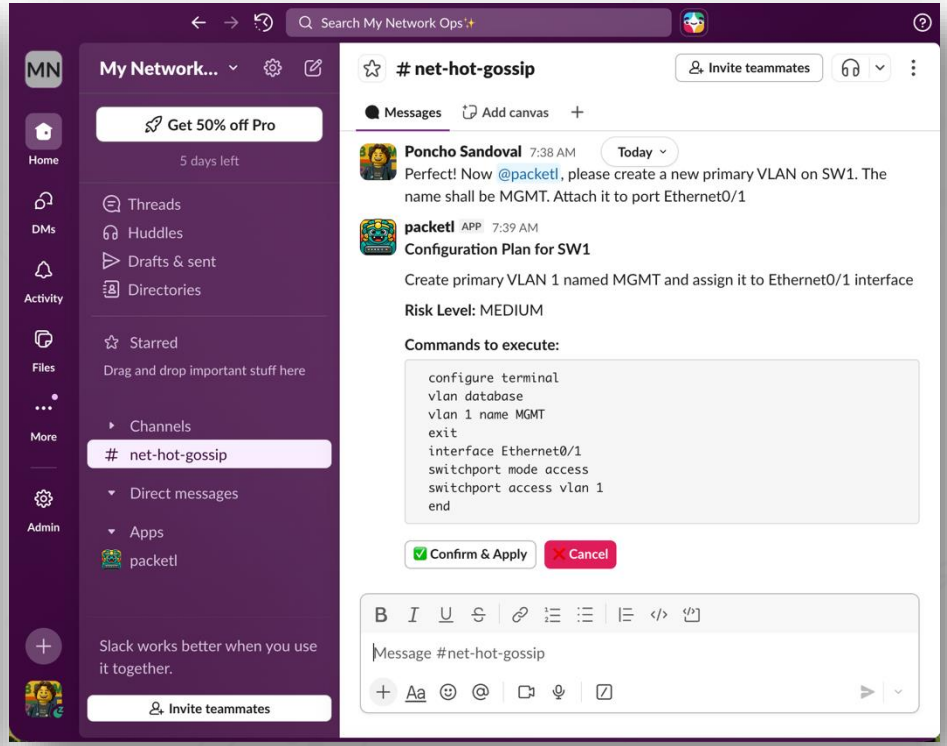
Commit workflow



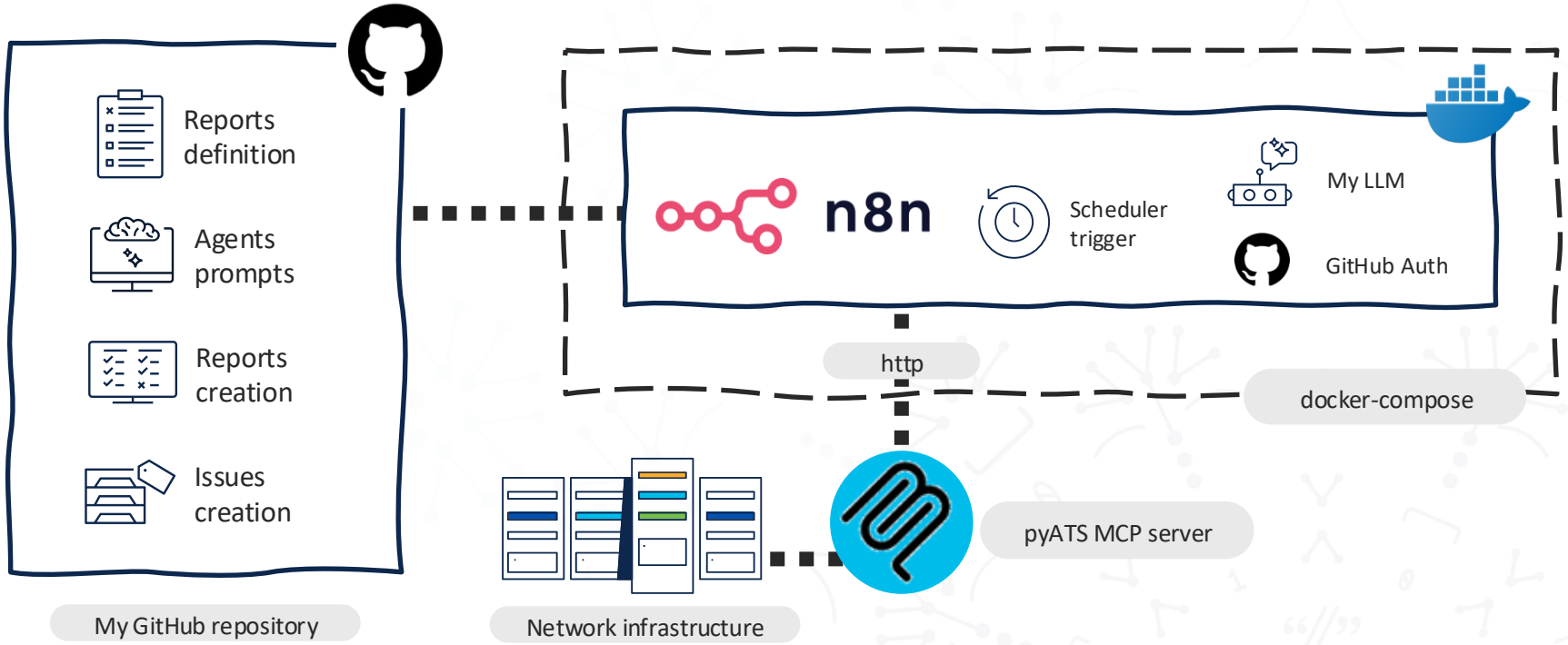
Reading interaction



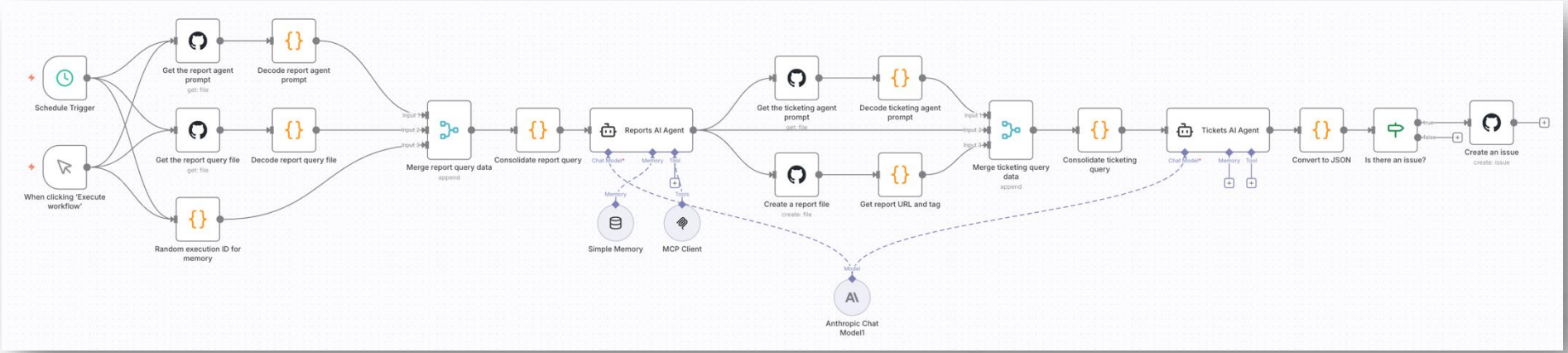
Commit interaction



Demo 2: Agentic Reporting & Ticketing for NetAuto



Reporting & ticketing workflow



pyATS-loves-agenticops / reports / files / interfaces_status_2026-01-13T18:19:22.447+00:00.md

ponchotitlan interfaces_status report gen

Preview | Code | Blame 95 lines (70 loc) · 4.28 KB

Report generation

Network Interface Status Summary Report

Executive Summary

Analysis of interface operational states across the network infrastructure reveals **14 operational interfaces** and **4 administratively down interfaces** across 4 devices. The network demonstrates healthy connectivity with active traffic on management interfaces and no packet drops or queue congestion detected.

Scope & Assumptions

- **Scope**: Complete interface status analysis for all devices (`R1` , `R2` , `SW1` , `SW2`)
- **Data Source**: Real-time interface status via `show ip interface brief` and `show interfaces summary`
- **Assessment Period**: Current operational state snapshot

Environment Overview

Infrastructure Components:

- **Routers**: 2 IOS-XE devices (`R1` , `R2`)
- **Switches**: 2 IOS-XE devices (`SW1` , `SW2`)
- **Total Interfaces**: 18 interfaces analyzed
- **Platform**: IOL (IOS on Linux) virtual environment

[interfaces_status report gen] [⚠️medium] Address interface redundancy and documentation gaps on Router R1 #17

Edit | New issue

Issue creation

Open

ponchotitlan opened last week — Owner

Context

Router R1 interface analysis reveals healthy operations with 3/4 interfaces functional and zero errors. However, several operational improvements are needed to enhance network reliability and maintainability.

Key Issues Identified

- **Single Point of Failure**: Ethernet0/3 remains unused, reducing redundancy options
- **Documentation Gap**: Missing interface descriptions on Ethernet0/0 and Ethernet0/1
- **Traffic Concentration**: All traffic flowing through only 3 active interfaces
- **Monitoring Gap**: Lack of standardized interface monitoring

Action Items

Immediate Actions

- [ ] Add descriptions to Ethernet0/0 and Ethernet0/1 for operational clarity
- [ ] Evaluate Ethernet0/3 - determine if it should be activated for redundancy or permanently removed

Operational Improvements

- [ ] Implement regular interface health checks focusing on error counters and utilization trends
- [ ] Establish consistent interface naming and description conventions

Assignees
No one - Assign yourself

Labels
No labels

Projects
No projects

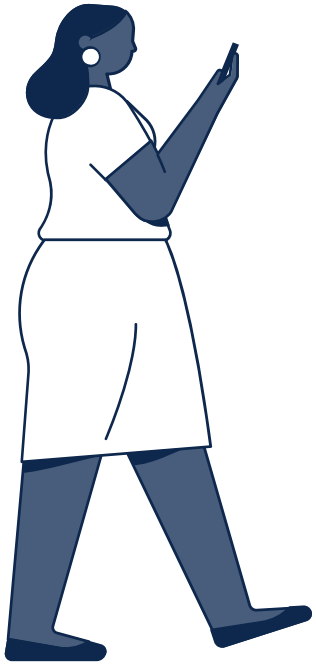
Milestone
No milestone

Relationships
None yet

Development
Code with agent mode

Create a branch for this issue or link a pull request.

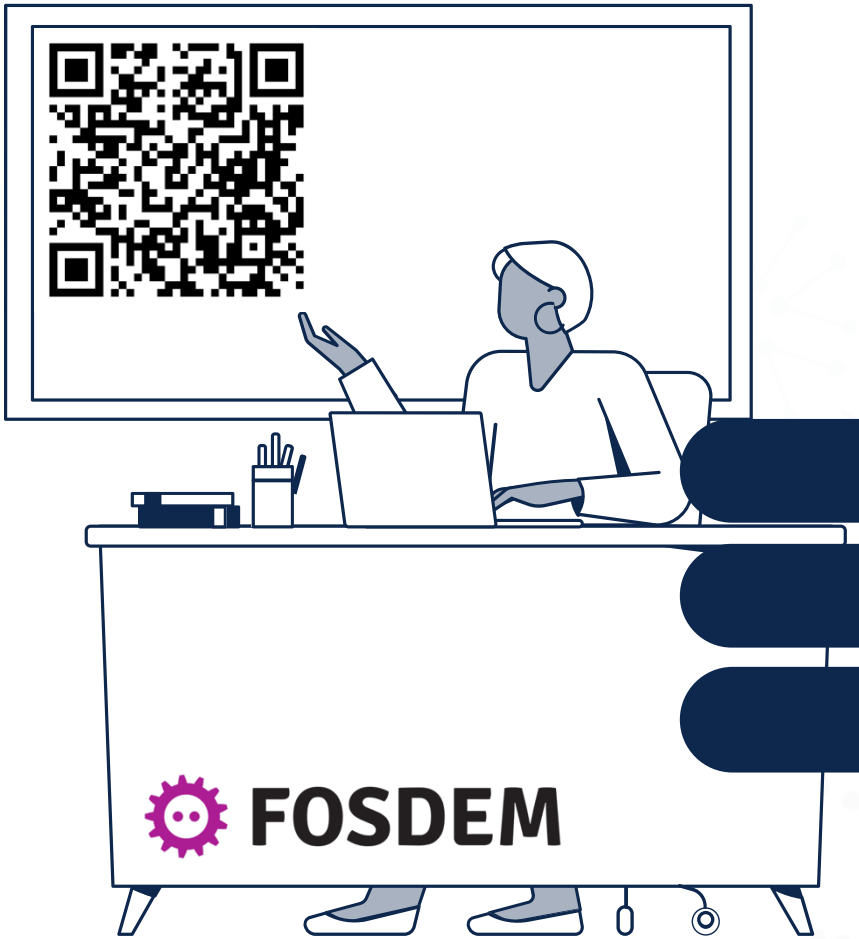
Notifications Customize
Unsubscribe
You're receiving notifications because you're subscribed to this thread.



Start small: build single-purpose agents for well-defined operational tasks

Treat LLM agents like junior network engineers: restrict what they can touch and how

Build agents on top of real network tools (CLI, APIs, pyATS) – Also, low-code is your ally!



About n8n
docs.n8n.io/

About pyATS
developer.cisco.com/docs/pyats/

Our demo repo: pyATS loves AgenticOps
cs.co/pyATS-loves-agenticops

The bridge to possible