WORKSHOP GARR 2026

NET MAKERS

Uso di LLM on-prem. L'esperienza del progetto AI4INFN

Marco Corvo

Istituto Nazionale di Fisica Nucleare

AI: fire walk with me

L'AI sta cambiando radicalmente il modo in cui tutti noi lavoriamo:

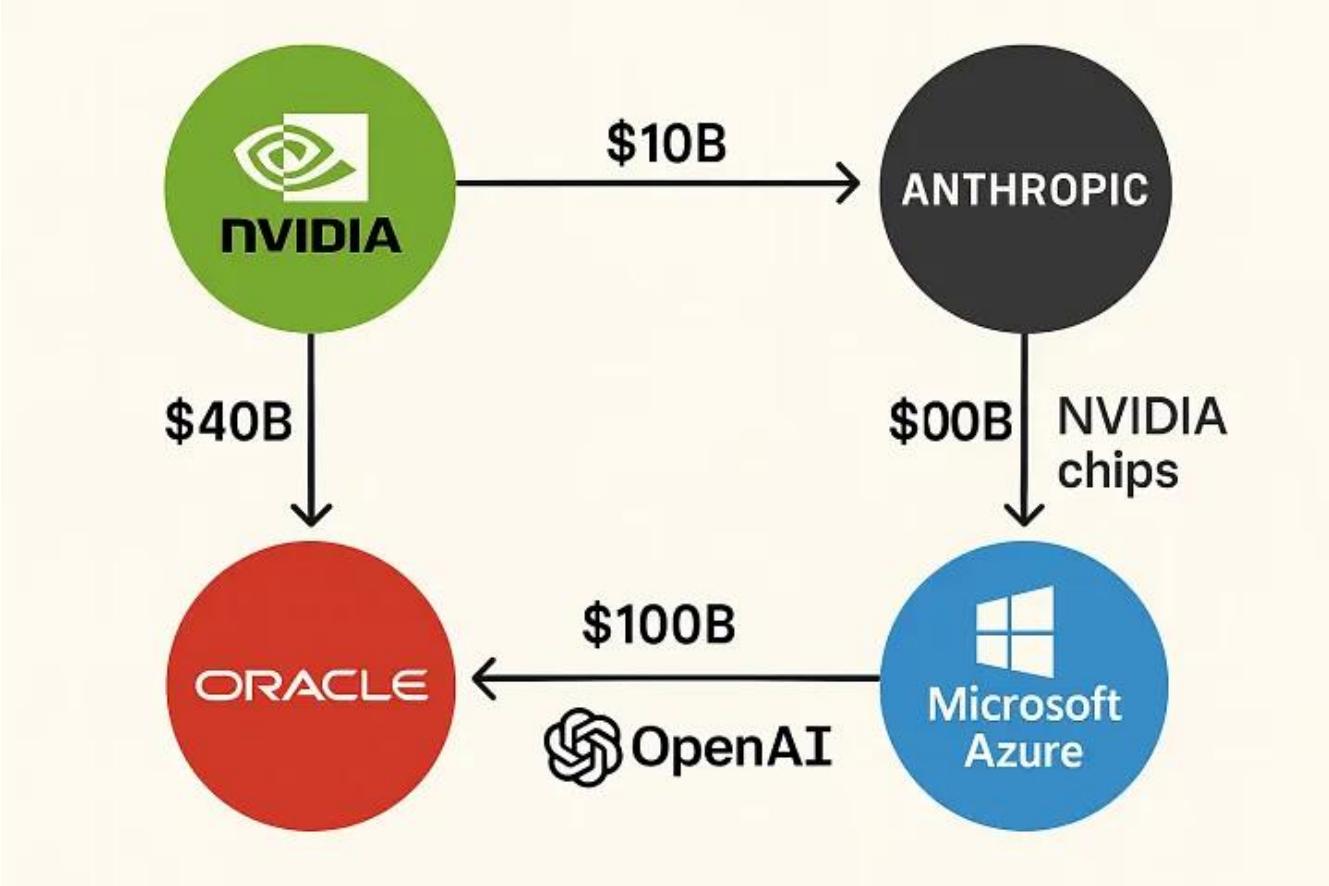
- Supporto alle attività amministrative (ad esempio verifica documenti, riassunti, controllo incrociato di condizioni contrattuali, sommario delle email con precise indicazioni su mancate risposte o richieste di supporto)
- Analisi dati (ad esempio con LLM allenati su dati di esperimenti di LHC)
- Coding agentico
- Supporto alle operation (tracking di email spam, analisi automatica di log ed eventi di auditing)

AI: un prezioso aiuto ma un gigante dai piedi di argilla

La figura mostra quello che, personalmente, continuo a ritenere un grosso problema:

Chi ripagherà i campioni dell'AI degli ingenti investimenti?

Io una risposta ce l'ho...

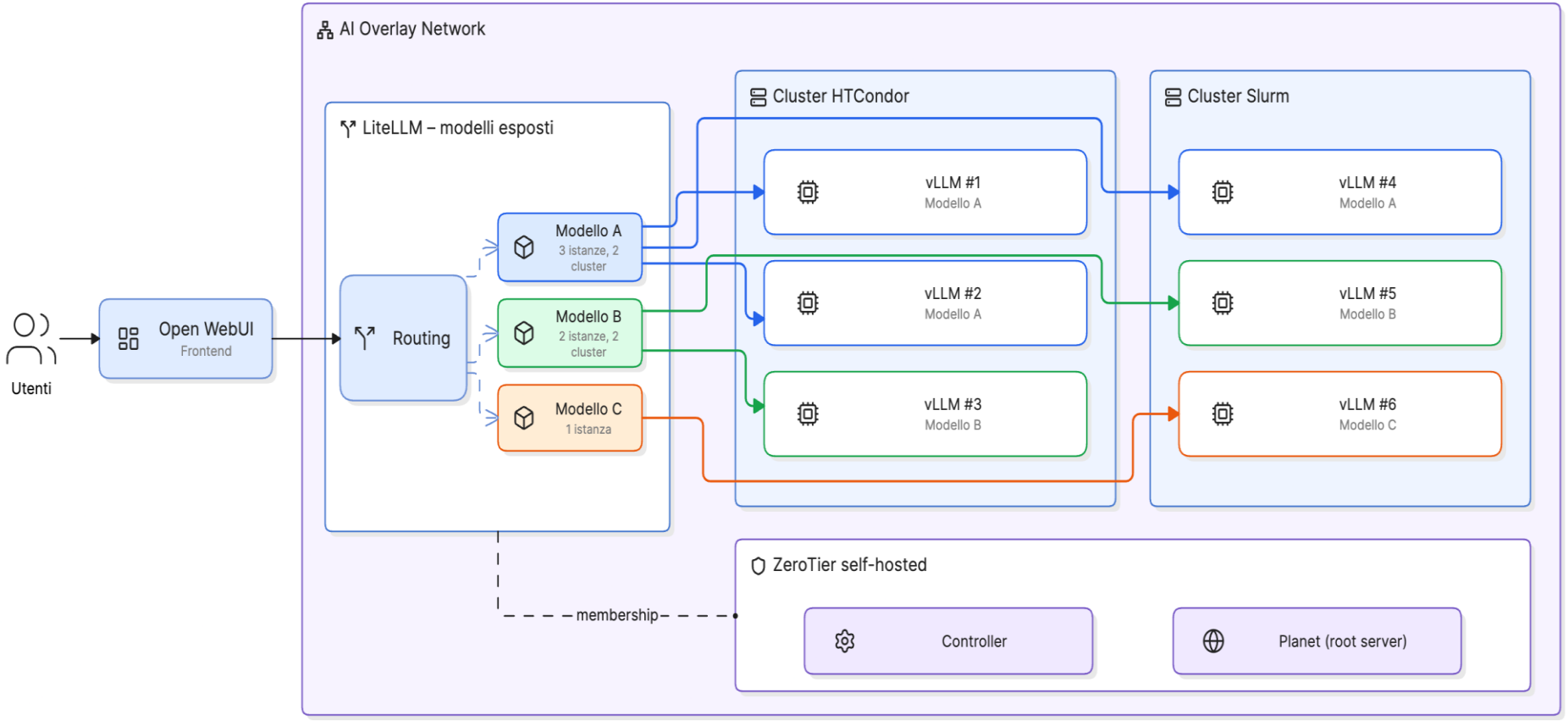


AI: ha senso per un Ente di ricerca dotarsi di una infrastruttura AI on-prem?

- E' quello che ci siamo chiesti nell'ambito del progetto **AI4INFN** e la risposta, per noi, è sì.
- Non pensiamo di sostituire i prodotti commerciali tout-court, ma di razionalizzarne l'utilizzo per i casi più importanti
- A fianco di questi modelli vogliamo proporne altri open-weight da servire nelle nostre infrastrutture con due scopi principali
  - **Mantenere il controllo dei dati e degli accessi**
  - **Limitare i costi**

Diversi approcci, un solo scopo: fornire un'infrastruttura AI per nostri utenti

Il primo approccio è un servizio AIaaS che si appoggia ad cluster Slurm o HTCondor



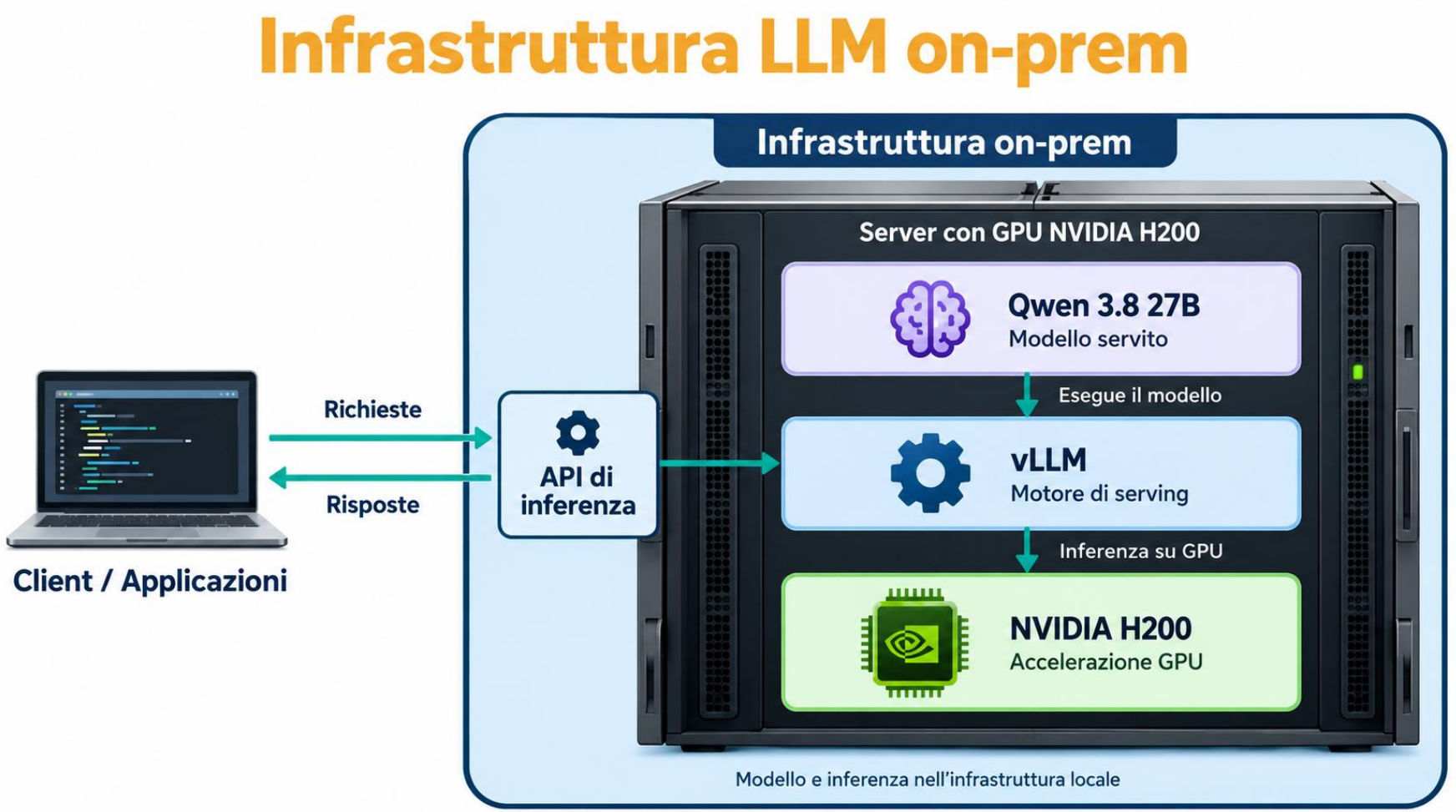
Diversi approcci, un solo scopo.

Il secondo è un accesso diretto attraverso un indirizzo pubblico



Esperienza al CNAF

- In questo momento al CNAF utilizziamo una H200 con 96GB RAM HBM3e e 480 GB RAM DDR5
- L'LLM (Qwen 3.8 27B BF16) viene servito da un'istanza di vLLM che gira in un container Docker
- Le richieste arrivano direttamente all'endpoint di vLLM sulla porta 8000 (no reverse proxy)

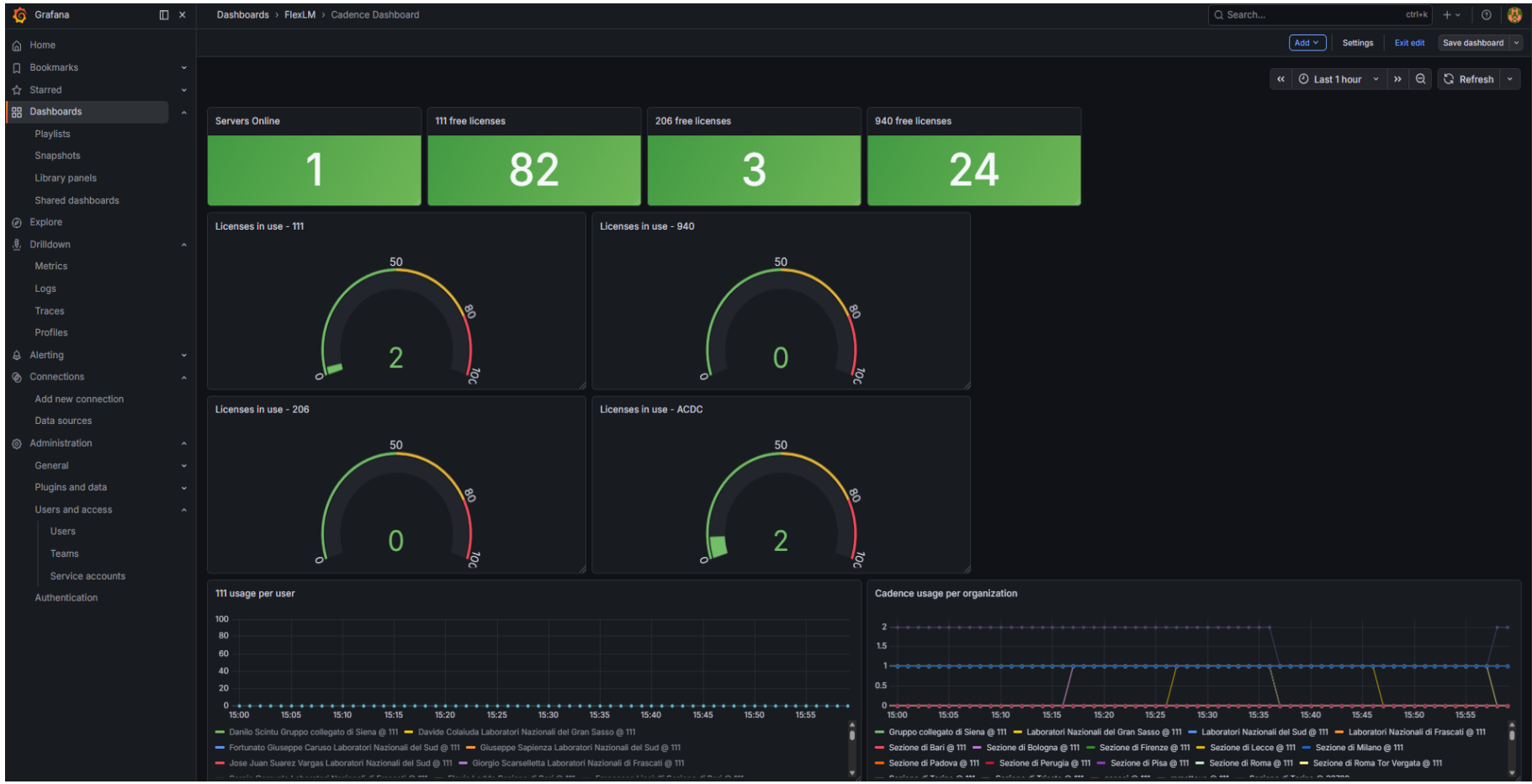


I casi d'uso

- Personalmente ho iniziato ad usare l'AI per risolvere problemi sistemistici e sviluppare codice
  - Indicazioni su come eseguire certe operazioni su OS Linux
  - Scrittura di un modulo Puppet
  - Aggiornamento di un servizio
  - Sviluppo di un poller per license server e successivo interfacciamento con Prometheus e Grafana
- Successivamente ho investigato l'uso di agenti per automatizzare:
  - Il build di una particolare classe di web pages
  - L'aggiornamento di un servizio su VM remota

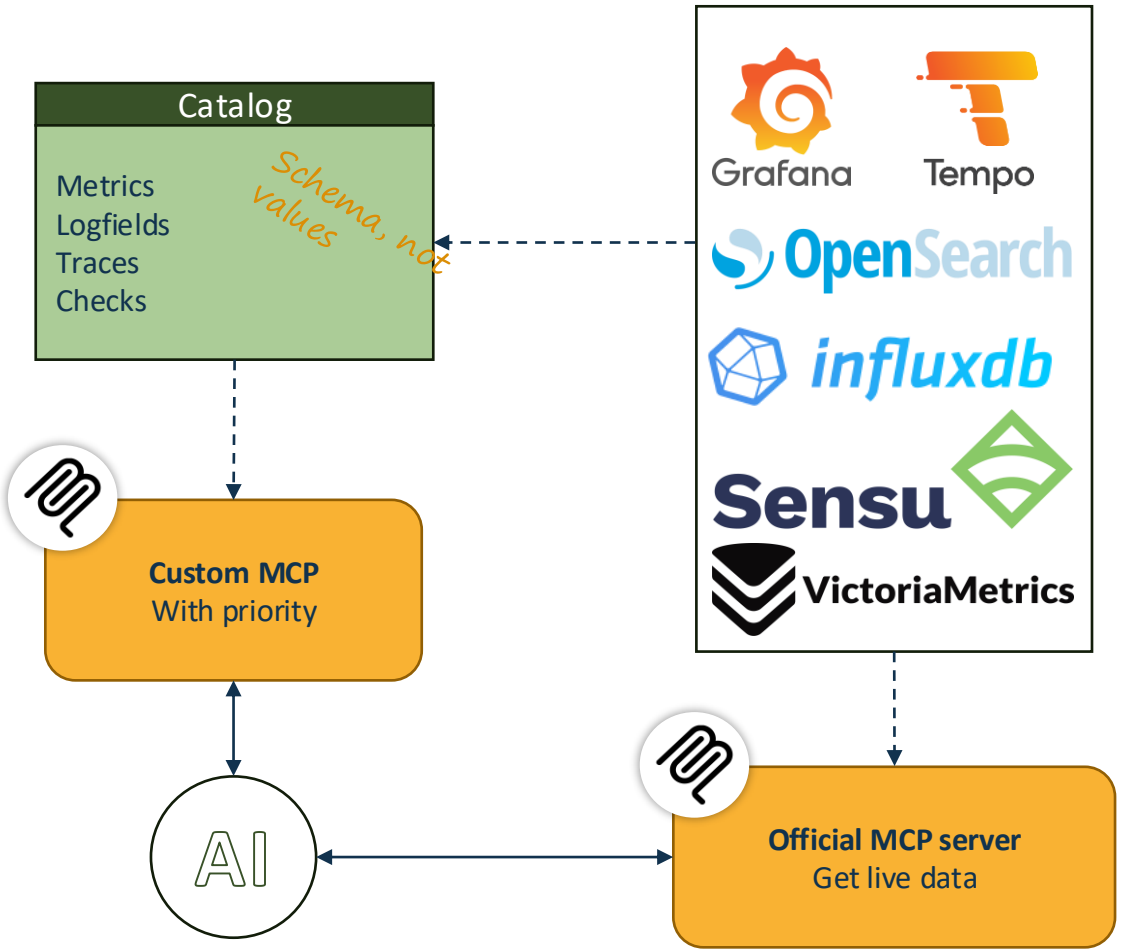
Monitoring di server Flex

- Il poller interroga ciclicamente una lista di server Flex
- Incrocia il dato dell'uid fornito dal server con le informazioni nell'LDAP nazionale per associare utenti e centri con l'uso dei prodotti
- Invia le informazioni ad un'istanza Prometheus che popola le dashboard Grafana



Un server MCP per esporre tool di monitoring

- Il catalogo con le metriche da monitorare è contenuto in un database Postgres
- Il server MCP espone il catalogo e i tool per interrogarlo
- Gli agenti AI di monitoring chiedono le metriche al server MCP che le fornisce attraverso chiamate ai tool
- Evita di dover “indovinare” quali metriche o campi usare e rende l’analisi più veloce e aprofondita



Agente per build da diversi repo GitHub

- L'agente gira in un container Docker dove viene installato Claude Code
- Ogni settimana controlla se compare un nuovo tag nel repo di Vaultwarden
- In caso affermativo applica patch e custom commit e lancia il build delle web pages

```
---
description: Update the Bitwarden-branded web vault to a new Vaultwarden release (revert rebrand commits, rebrand, add custom images, build).
argument-hint: "[version]"
---

You are running the **monthly Vaultwarden → Bitwarden web-vault update**. Perform the exact same
pipeline that was done for `v2026.7.0`, but for a new version. Work autonomously, report each
phase, and stop to confirm only when something is genuinely ambiguous (an unexpected rebrand
commit, a revert conflict you can't resolve confidently, or a destructive disk cleanup).

An optional version was passed as: `$ARGUMENTS`
(If empty, use the latest upstream version.)

## Layout (verify before relying on it)
The command assumes **no pre-existing working directory and no bind-mounted volume** — it
creates everything it needs. Prerequisites: network access (GitHub + baltig.infn.it), docker
access, and the two GitLab tokens (below).
- **WORK** = scratch work area, `${VW_WORK:-/tmp/vw_update}` (point the override only at an
  ephemeral scratch path). Created by Phase 1; every repository is **cloned into it**.
- **HARNESS_REPO** = `https://baltig.infn.it/ssnn/bitwarden-web-client.git` — the build
  harness (private GitLab, same host as SSNN_REPO; the SSH form is
  `git@baltig.infn.it:ssnn/bitwarden-web-client.git` — use the HTTPS form, never SSH: a
  container has no host SSH keys).
- **ROOT** = `$WORK/bw_web_builds` — a fresh clone of HARNESS_REPO (contains `Dockerfile`,
  `scripts/`, the rebrand build machinery). `.bw_clients.git` and `container_builds/` are
  created inside it during the run.
- **CLIENTS** = `$WORK/bw_clients` — a fresh shallow clone of VW_REPO at the `$target` branch
  tip (has `apps/web/`, `libs/`).
- **SSNN** = `$WORK/ssnn-vault` — a fresh shallow clone of SSNN_REPO (custom images + server
  SCSS), authenticated with a GitLab personal access token supplied in the docker compose
  project's `.env` file (see Phase 1).
- **VW_REPO** = `https://github.com/vaultwarden/vw_web_builds.git` (upstream version source,
  public).
- **SSNN_REPO** = `https://baltig.infn.it/ssnn/vault.git` (private GitLab repo; the SSH form
  is `git@baltig.infn.it:ssnn/vault.git` — use the HTTPS form, never SSH: a container has no
  host SSH keys).
- **SSNN_GITLAB_TOKEN** = personal access token for SSNN_REPO (GitLab, `read_repository`
  scope at minimum), provided in the docker compose tool's `.env` file.
- **HARNESS_GITLAB_TOKEN** = personal access token for HARNESS_REPO (GitLab,
  `read_repository` scope at minimum) — a *separate* token from SSNN_GITLAB_TOKEN, defined
  in the `environment:` section of the docker-compose file.

> Versions are **released as tags** upstream (3-part `vYYYY.M.P`). Branches are cut *before*
> release and may be untagged (still in development). A version is only built once it has a
> corresponding **tag** — the tag is the release gate; the build still checks out that version's
> **branch** (the tag and branch can point to different commits, and the branch is the proven
> build source).

## Phase 0 — Determine the target version
A version is buildable only once it is **released**, i.e. it has a corresponding **tag**
```

Dove ci piacerebbe arrivare: l'infrastruttura

- Integrare sistemi eterogenei e distribuiti in un unico pool di risorse
  - Sistemi con GPU disponibili vengono inseriti o, in caso di malfunzionamenti, rimossi garantendo sempre la fruibilità del servizio
    - Un servizio funzionante è già presente (vedi slide 5)
  - Sistemi «cloud-native» dietro batch system Htcondor e Slurm che costituiscono l'ossatura di DataCloud
  - In questo ambito è stato prezioso il lavoro dei colleghi di Roma2 che hanno implementato un meccanismo di load balancing che ridireziona le richieste sulle risorse attive e meno cariche

Dove ci piacerebbe arrivare: i casi d'uso

- Le applicazioni dell'AI sono limitate solo dalla fantasia
- Ci sono diverse iniziative per implementare sistemi RAG che possono essere allenati anche con i dati delle email per gestire il supporto utenti
  - Poter mantenere un'infrastruttura on-prem ci consente di controllare i dati e mantenerli all'interno del perimetro dell'INFN
- Rendere organico e coerente un insieme di dati disomogenei e in continua evoluzione attraverso «knowledge graph»
- Abilitare agenti che controllano strumenti di misura e fanno analisi dati

Caveat: cosa mi ha sorpreso

- Stavo cercando una risposta ad una domanda su una configurazione di un vCenter vSphere
- Claude Code mi ha avvisato che la VM su cui stava lavorando non era un vCenter, quindi gli ho detto «non ti trovi in un vCenter»
- «Curiosando» per la mia home directory, ha scovato un vecchio progetto per il provisioning di VM in Terraform, ha individuato le mie credenziali e mi ha chiesto se poteva usarle per collegarsi ad un vero vCenter
- Fortunatamente non era attivo l'auto mode e l'ho fermato

Conclusioni

- Dotarsi di infrastrutture on-prem, per chi ha le competenze per gestirle e mantenerle, può risultare strategico in termini di protezione dei dati e contenimento dei costi
- L'evoluzione di modelli e strumenti è, nel bene e nel male, estremamente rapida e impone anche all'utente una notevole e continua capacità di adattamento e di mantenere il passo dei cambiamenti
- E' innegabile che, con un uso corretto e consapevole, i tempi per svolgere particolari compiti possono essere notevolmente ridotti e ottimizzati

Approfondimenti

Accordo tra CERN e Mistral per allenare modelli LLM con dati di LHC
https://home.cern/cern-and-mistral-join-forces-to-accelerate-ai-for-science/

Presentazioni del gruppo CCR_AI4INFN
https://cells.infn.it/public/233cd60a3af7

Per le domande www.partici.fi e codice 83995288

WORKSHOP GARR 2026
NET MAKERS