



NetEye Conference 2026

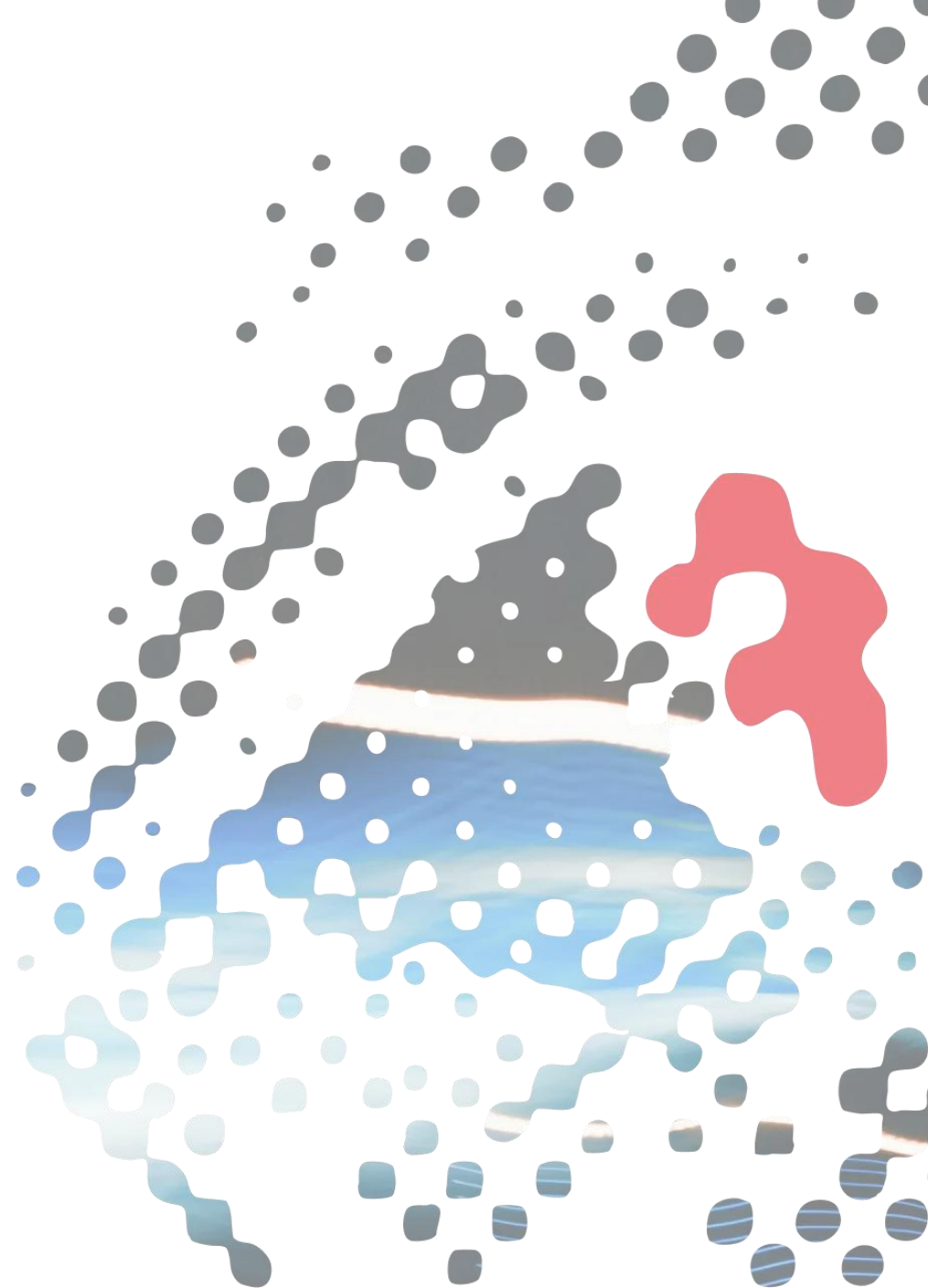
AI & Human Insight: la coesistenza strategica per le IT Operations

Ridurre il lavoro manuale negli upgrade:

conversione delle dashboard IcingaDB

con supporto MCP

Andrea Mariani



Agenda

- Il costo nascosto dell'upgrade
- IDO e IcingaDB: perché non è un trova-e-sostituisci
- Tre attori, e su cosa è stato addestrato il modello
- Demo: una dashboard, dieci query, una decisione
- Cosa sa fare il modello oggi, e come lo proviamo
- Dove mettere il confine fra AI e validazione umana

Observability

Il costo nascosto dell'upgrade

22

dashboard con pannelli SQL

132

query scritte a mano,
fino a 3.000 caratteri

217

riferimenti a un datasource
che verrà spento

Installazione NetEye reale

Nessuna di queste voci
compare nel piano di upgrade.

Una dashboard sbagliata non si spegne.

Non dà errore: mostra un numero. Magari plausibile, magari sbagliato.

Il rischio vero non è il tempo del rework: è l'errore silenzioso nel pannello che qualcuno guarda alle tre di notte.



IDO e IcingaDB: stesso monitoraggio, altro modello

	Icinga IDO	IcingaDB
Identità	tabella-perno icinga_objects (object_id)	id checksum binari su ogni tabella
Stato	current_state (numero)	hard_state / soft_state
Flag	problem_has_been_acknowledged = 0/1	is_acknowledged = 'y'/'n'
Tempo	timestamp in secondi	bigint in millisecondi
Custom var	una tabella	due salti: host_customvar → customvar
Dimensione	63 tabelle · 792 colonne	67 tabelle · 637 colonne

Non è una sostituzione di testo: è una traduzione. E si può sbagliare in modo elegante.



Prima e dopo: una query reale

Icinga IDO

```
SELECT COUNT(*) AS down
FROM icinga_objects AS ho
INNER JOIN icinga_hosts AS h
    ON h.host_object_id = ho.object_id
    AND ho.is_active = 1
    AND ho.objecttype_id = 1
INNER JOIN icinga_hoststatus AS hs
    ON hs.host_object_id = ho.object_id
WHERE hs.current_state = '1'
    AND (hs.problem_has_been_acknowledged
        + hs.scheduled_downtime_depth) = 0
```

IcingaDB • tradotta da ido-icingadb

```
SELECT COUNT(*) AS down
FROM icingadb.host AS h
INNER JOIN icingadb.host_state AS hs
    ON hs.host_id = h.id

WHERE hs.hard_state = 1
    AND hs.is_acknowledged = 'n'
    AND hs.in_downtime = 'n'
```

1 la tabella-perno sparisce:
join diretti sugli id

2 current_state → hard_state

3 somma di contatori →
due enum 'y'/'n'

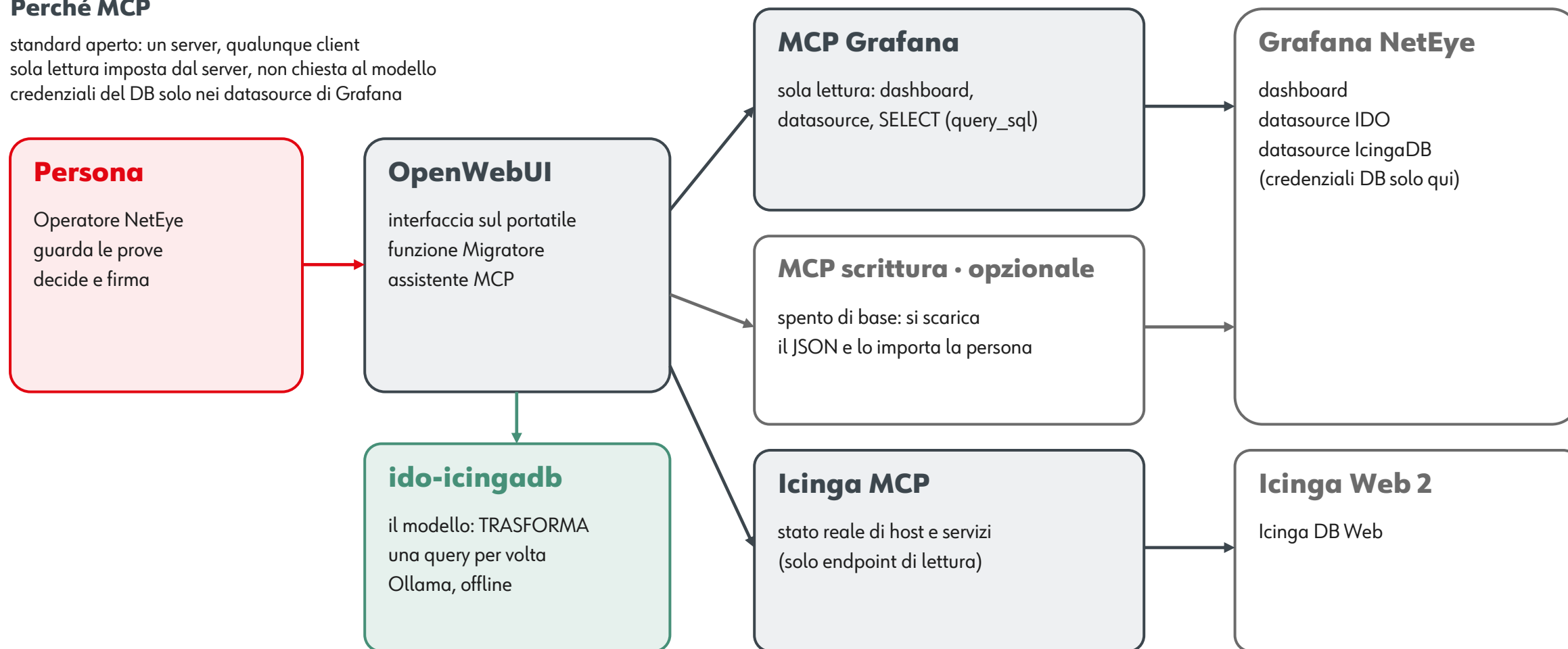
Nessun passaggio è difficile da solo. Il difficile è farli tutti, 132 volte, senza sbagliarne uno.



Tre attori, tre ruoli

Perché MCP

standard aperto: un server, qualunque client
sola lettura imposta dal server, non chiesta al modello
credenziali del DB solo nei datasource di Grafana



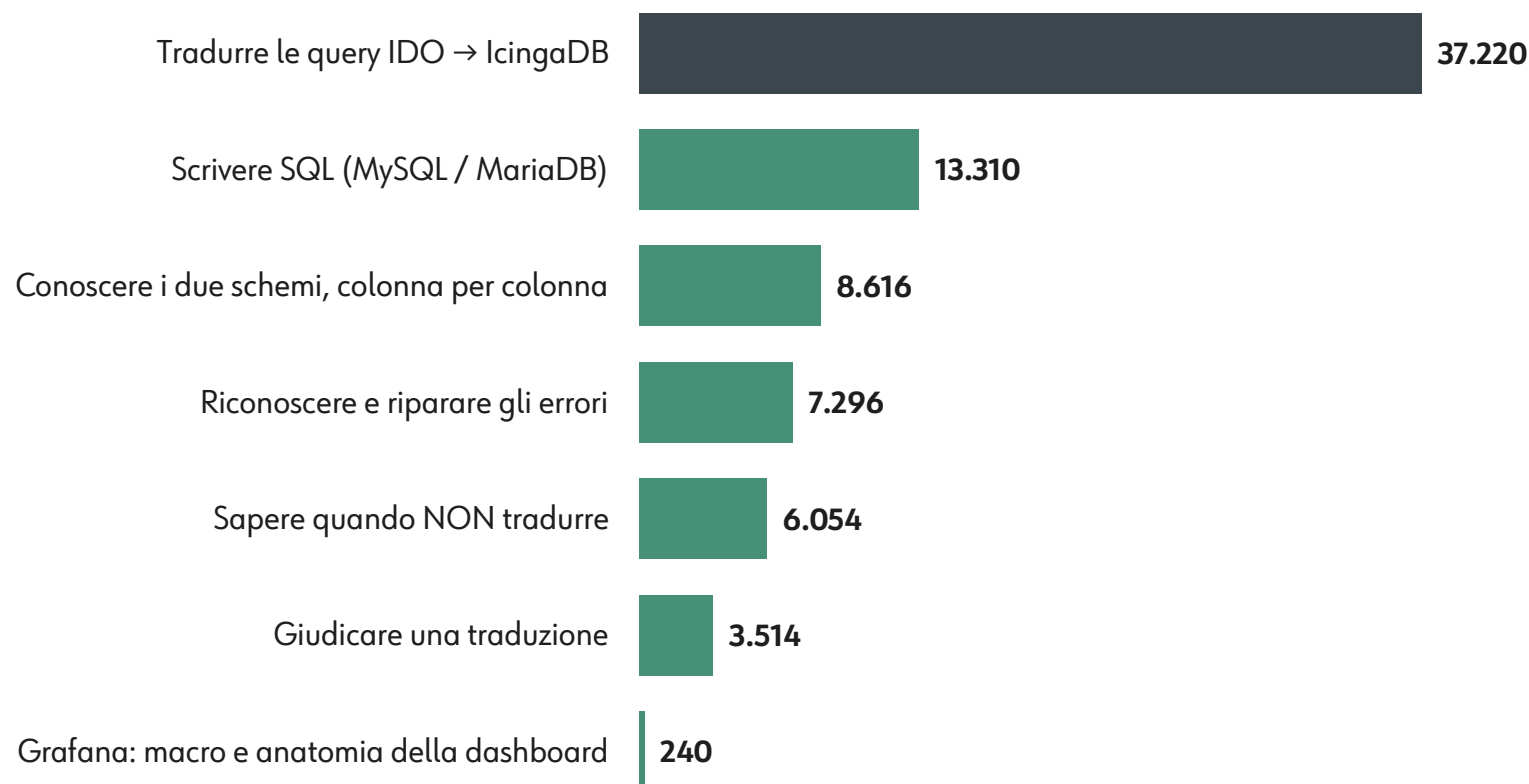
● modello (AI)

● strumenti MCP

● persona

Tutto in locale: Docker + Ollama. Nessun dato esce.

Il modello: su cosa è stato addestrato



3B

parametri · Qwen2.5-Coder + LoRA

2,2 GB

gira offline su 4 GB di VRAM

~76.000

esempi, ognuno verificato sul database vero

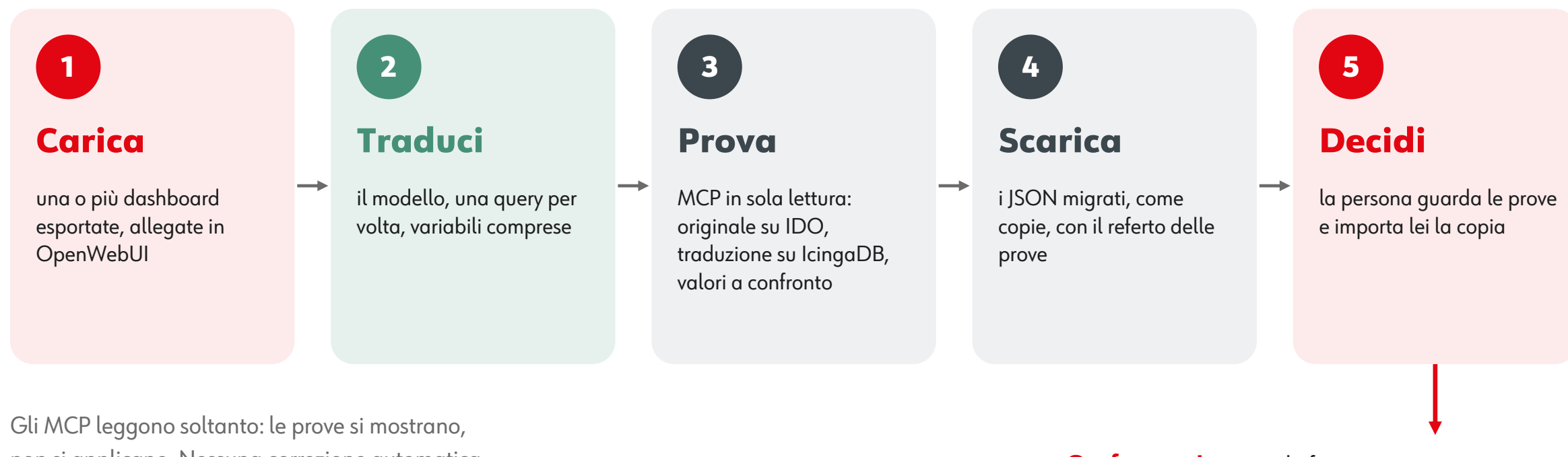
~1 giorno

per rifarlo da zero: 13-21 h di GPU

Tradurre è metà del lavoro. L'altra metà è ciò che serve per tradurre bene.



Il flusso: cinque passi, zero scritture su Grafana



Gli MCP leggono soltanto: le prove si mostrano, non si applicano. Nessuna correzione automatica.

Grafana → Import: lo fa una persona, come copia.
Opzionale, spento di base: creare le copie in una cartella dedicata via MCP.

● modello (AI)

● strumenti MCP

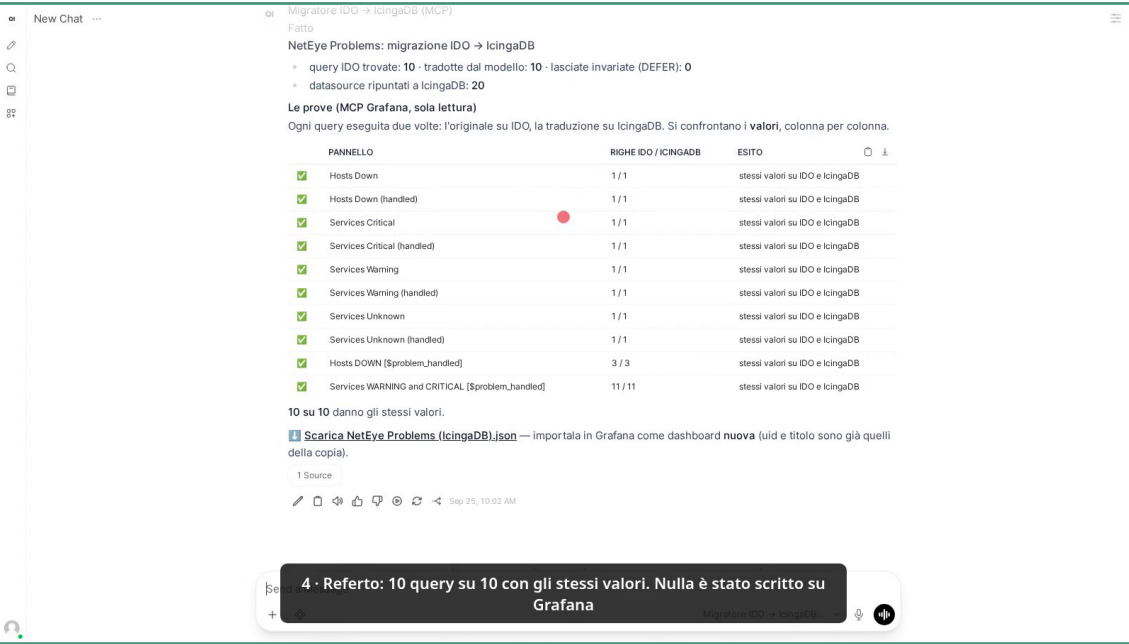
● persona



Demo: prove, non promesse

A • Il flusso di base

allego il JSON, migro, scarico e importo io



10

query tradotte dal modello

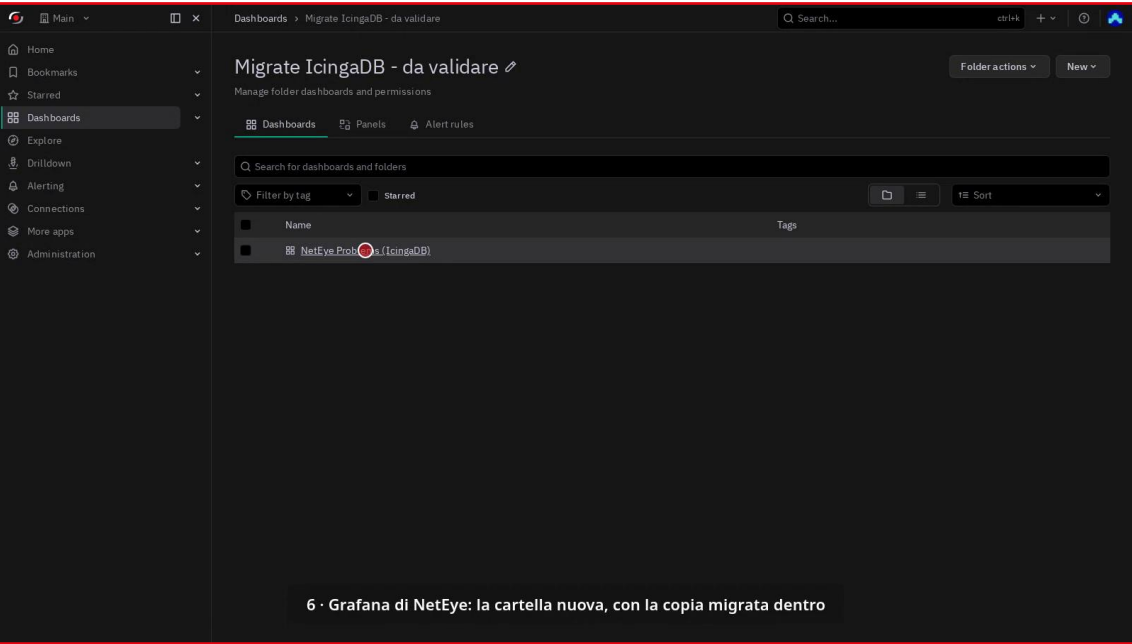
20

esecuzioni via MCP, in lettura

video • 1:30 ciascuno • clic per avviare • tempo reale, senza tagli

B • Sul Grafana di NetEye

opzione accesa: la copia va in una cartella nuova



10 / 10 • 8 / 10

stessi valori: demo • NetEye

0

originali toccati



Cosa sa fare il modello oggi

Oggetti e nomi

```
icinga_objects o ... o.name1
```

↓

```
icingadb.host h ... h.name
```

host, servizi, endpoint, zone: niente più icinga_objects

Tempi in millisecondi

```
$__timeFilter(sh.state_time)
```

↓

```
sh.event_time BETWEEN  
$__unixEpochFrom()*1000 AND  
$__unixEpochTo()*1000
```

anche \$__timeGroup e FROM_UNIXTIME(col/1000)

Stati e storico

```
sh.state · state_type = 1
```

↓

```
sh.soft_state · state_type = 'hard'
```

stati, prese in carico, downtime, notifiche

Variabili personalizzate

```
icinga_customvariables  
... varname = 'host_owner'
```

↓

```
host_customvar + customvar  
... name = 'host_owner'
```

il filtro sulla variabile resta al suo posto

Grafana resta intatto

```
${host_owner:sqlstring} · $__interval  
AS 'State time'
```

↓

```
${host_owner:sqlstring} · $__interval  
AS `State time`
```

variabili, macro, alias: il pannello li ritrova

Sa quando non tradurre

metriche, altri datasource,
query che non leggono IDO

↓

**lasciate come sono,
e segnalate nel referto**

meglio fermarsi che inventare

Il modello oggi, su 22 dashboard reali

126 / 126

query che eseguono su lcingaDB

60

danno gli stessi valori di IDO, colonna per colonna *

120 / 120

uguali alla traduzione di riferimento

0

pannelli vuoti, alias o filtri persi

5 / 5

query critiche: stessa risposta ripetendo la domanda

~20 s

per migrare e provare NetEye Problems (10 query)

Ogni prova gira su Grafana vero, via MCP, sui due database: **la prova deve girare dove gira la produzione.**

Zero difetti noti sulle nostre prove: che non vuol dire zero difetti. La prossima dashboard va provata come queste.

* le altre non hanno dati su cui confrontare (per quelle vale il confronto con il riferimento) o differiscono per i dati, non per la traduzione



Diverso non vuol dire sbagliato

Stessa prova sul NetEye del laboratorio, dove IDO e IcingaDB convivono:

colonna	IDO ↔ IcingaDB	causa	decisione (umana)
duration	70 d 14 h 2 m ↔ 70 d 14 h 1 m	le due query girano in momenti diversi	è il tempo che passa
output	Host Unreachable ↔ Packet loss 100%	i due database non sono allineati	è il dato: la query è giusta
timestamp	18:27:08 ↔ 18:27:07.531	precisione: secondi contro millisecondi	normalizzato dal confronto

La macchina trova le differenze. **Dare loro un significato è un giudizio.**



Dove mettere il confine: quattro domande

domanda	cosa prende	chi risponde	costo
1 La query gira?	tabelle o colonne inesistenti, sintassi	macchina	secondi
2 Il testo è fedele?	segnaposto Grafana, alias, filtri, stringhe	macchina	secondi
3 I valori sono gli stessi?	errori di significato: un nome, un'unità di tempo	MCP: IDO vs IcingaDB	minuti
4 La differenza ha senso?	errore, dato o precisione? e si importa?	persona	minuti per pannello

All'AI la trasformazione e la raccolta delle prove.

Alla persona il giudizio e la firma.

La domanda 3 esiste solo finché IDO è acceso.
Migrate le dashboard prima di dismetterlo.

Cosa portarsi a casa

1

Il lavoro manuale cambia natura

Non si riscrivono 130 query: se ne rivedono le due o tre che le prove segnalano.

2

Le prove contano quanto la trasformazione

Senza prove un modello è una promessa. Le prove girano dove gira la produzione, e si ripetono: il modello non è sempre deterministico.

3

Progettate il punto in cui decide la persona

MCP solo in lettura, nessuna scrittura su Grafana: la copia migrata la scarica e la importa una persona.

Modello, container MCP, OpenWebUI e demo girano su un portatile: **il pacchetto è a disposizione.**





Grazie per l'attenzione.

Domande?

