Clueless About The Cost of AI Model Usage?

OpenCost + llm-d = first of a kind inference cost tracking

Sima Nadler, IBM Research: LLM Inference Infrastructure & Cost Modeling

Alex Meijer, Principal Engineer, IBM Kubecost

Another KubeCon AI Talk?

Maybe, but not the way you might expect

We argue AI is entering "Day 2" phase

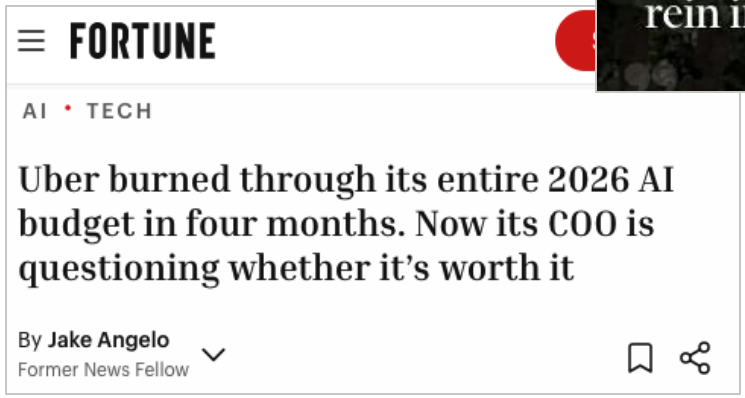
- Euphoria is subsiding
- Strengths and weaknesses becoming more apparent
- Costs becoming very apparent
  - "New Paradigm"/blank check days are over

Self-hosted models cheaper than hosted

- But what happens when *those* get expensive?







[1] [2] [3]

Agenda

1. OpenCost 101
2. llm-d 101
3. The Integration
4. Demo
5. Roadmap
6. Questions

OpenCost 101: What is it?

CNCF Incubating project

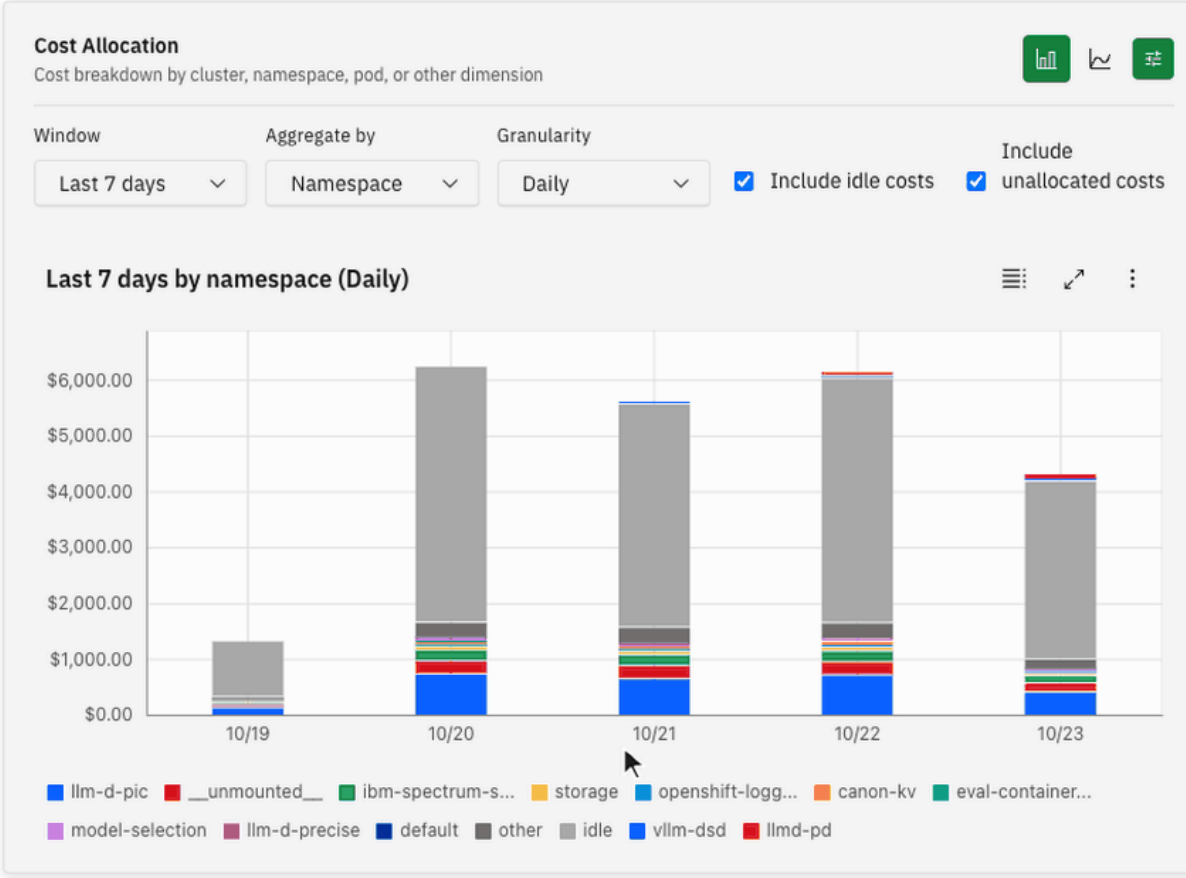
Launched in 2019

~6800 GitHub stars

FOSS FinOps tool for Cloud Native Spend

- Cloud provider costs
- SaaS costs
- Kubernetes Costs
- Any FOCUS data

REST APIs, UI, and exporters for consumption



Extending OpenCost to LLMs



Why extend a k8s/cloud spend tool to AI costing?

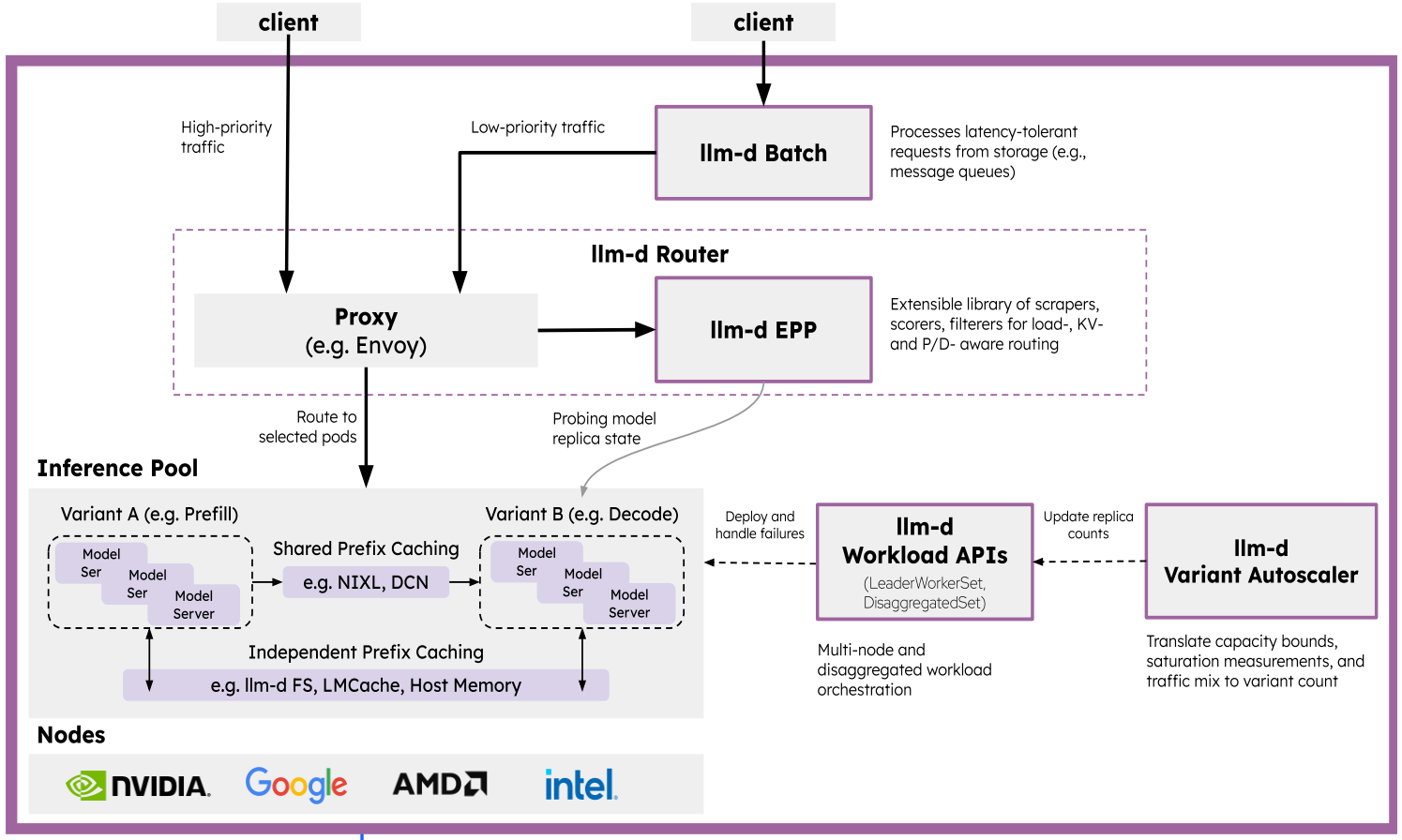
A supermajority (66%) of AI workloads run on k8s

OpenCost already has GPU costing features

- Building blocks in place for correlating GPU costs to LLM-level costs

Leverage existing community/expertise to deploy more quickly

llm-d 101: Distributed inference platform for serving at scale



v0.1

Launch, May 20th

(4 founding contributors + 7 launch partners)

AMD · cisco · CoreWeave · Google · Hugging Face · IBM · intel · Lambda · Mistral AI · NVIDIA · ...

Unique platform value proposition

Operationalizability

modular and resilient architecture with native integration into Kubernetes via Inference Gateway API

Flexibility

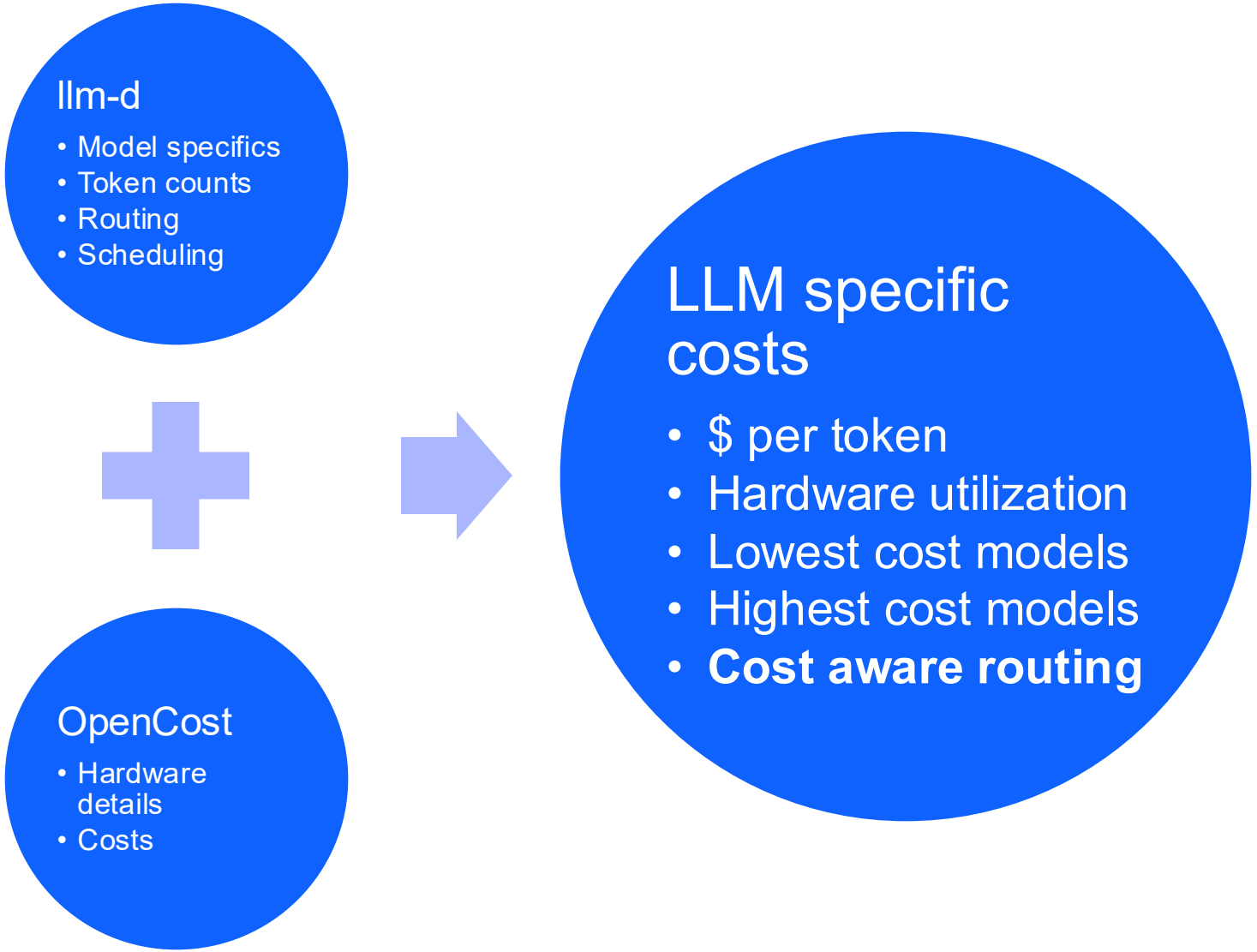
cross-platform with extensible implementations of key composable layers of the stack

Well-lit paths

Tested blueprints that guide adopters from experimentation to real-world, production-grade LLM deployments

Courtesy of Vita Bortnikov, IBM Research

llm-d + OpenCost Integration



Responsibility Model

Cost-aware model routing

Cost-aware LLM autoscaling

AI ROI calculations

AI Costing Observability

FinOps for AI

APIs

OpenCost

- Consumes LLM manager metrics
- Obtains provider pricing
- Obtains hosted model pricing/usage
- Computes cost based on pricing + allocation

Remains focused on costing and pricing

- **LLM internals remain abstract**

llm-d

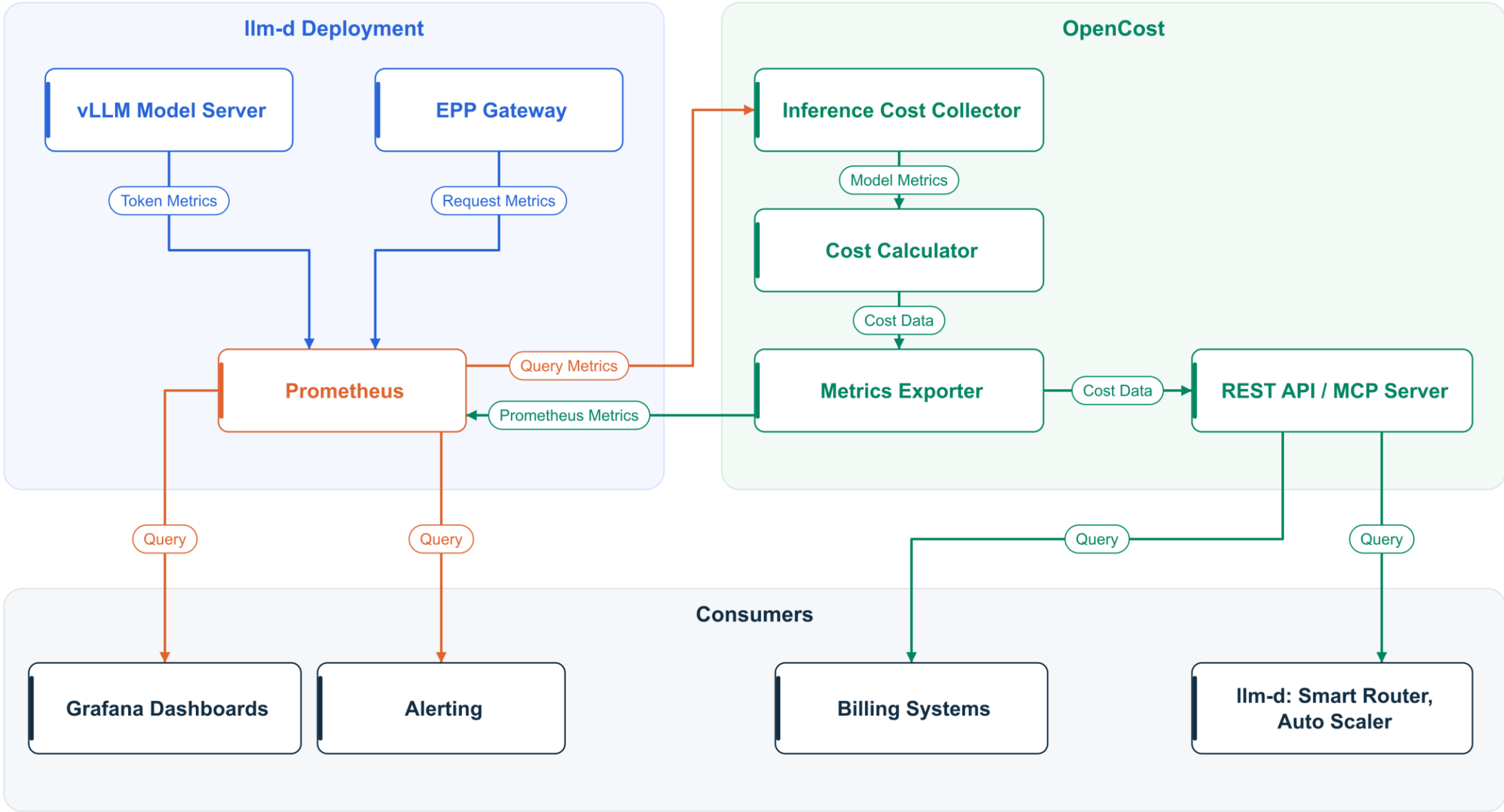
- Orchestrates LLMs
- Exports key metrics related to LLM performance
- Consumes normalized cost

Remains focused on LLM Orchestration

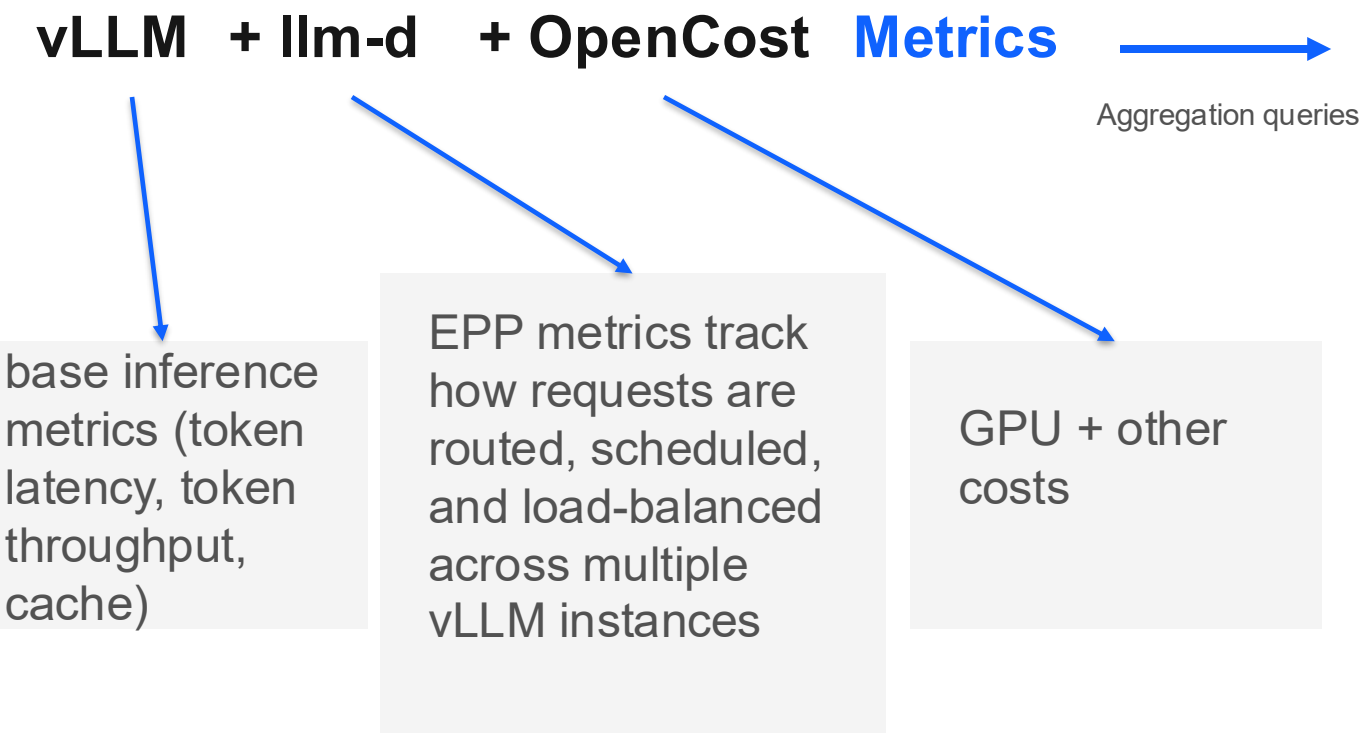
- **Pricing and costing code remains abstract**

Kubernetes

Integration Architecture



Turning Usage Metrics into Cost



Business Level Metrics

- Cost of self hosting AI models
  - and vs 3rd Party APIs
- Usage Inference Costs per Team
- Costs per Workload
- Costs per Model
- Costs per Model Variant
  - deployed with different hardware, LoRA, etc
- Cost per Token
  - Per Model, Model Variant
- KV Cache Costs and Savings

Allocation Vs Usage Cost Attribution

Allocation = max(request, usage)

- 'Production'/steady state metric
- Rewards right sizing
- Might result in under-provisioning

Usage cost = actual used resources

- 'R&D' metric
- No penalty for overprovisioning
- Might result in under-utilization

Cost Attribution	Number of CPUs requested	CPU Cost Per Hour	Average CPU Utilization	Total Cost
Allocation	2	$1	0.5	$2
Usage	2	$1	0.5	$1

Example Flow

K8s API

```
requests:
  cpu: 1
  memory: 1
  nvidia.com/gpu: 1
```

Container Runtime

```
container_cpu_usage_
seconds_total = 0.5

container_memory_
working_set_bytes = 2
```

Cloud Provider/CSV

```
"default,gpu": {
  "CPU": "1.0",
  "RAM": "2.0",
  "GPU": "10.0"
}
```

llm-d/vLLM

```
generation_tokens_
total = 75

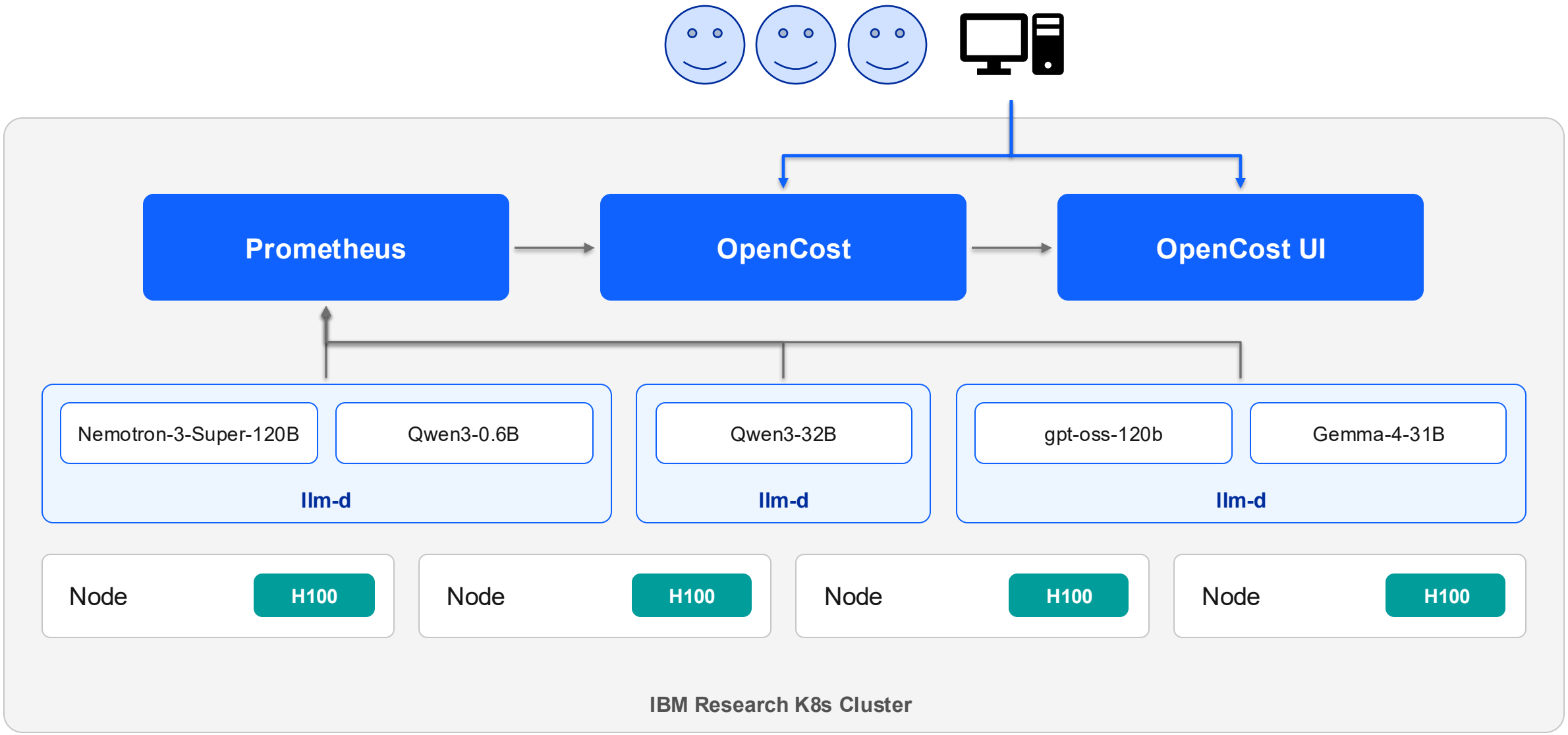
prompt_tokens_
total = 75
```

CPU Allocation: 1
RAM Allocation: 2
GPU Allocation: 1

CPU Cost: $1
RAM Cost: $4
GPU Cost: $10

$15 cost / (75 prompt tokens + 75 generation token) = $0.10 per token

Demo

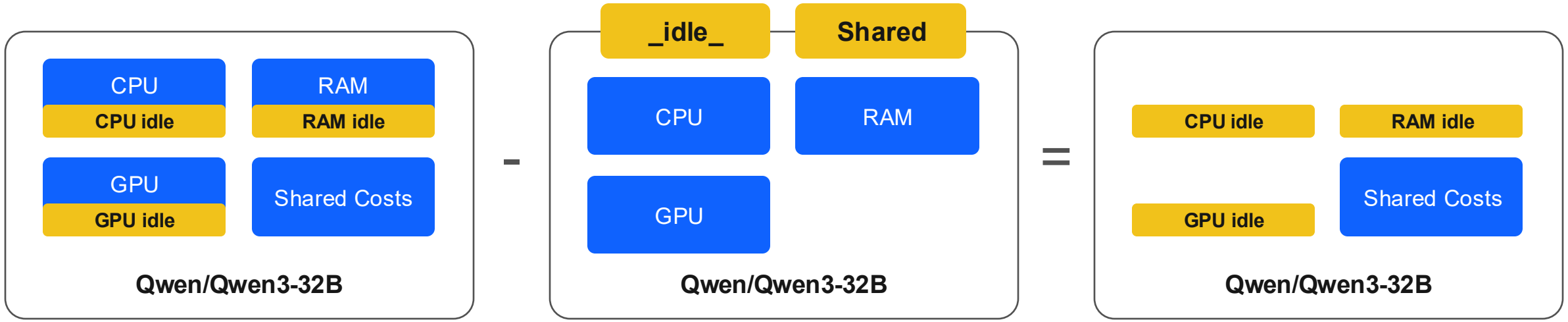

Prometheus
OpenCost
OpenCost UI
Nemotron-3-Super-120B
Qwen3-0.6B
llm-d
Qwen3-32B
llm-d
gpt-oss-120b
Gemma-4-31B
llm-d
Node
H100
Node
H100
Node
H100
Node
H100
IBM Research K8s Cluster

On The Topic of Idle

We are trying to answer the question of "what is the given model's share of idle costs"?

Idle costs are provided by OpenCost separately per node or per cluster, or shared by cost weight among containers

- How, then, do we get per-model idle if a model might run across many containers?




```
opts := &opencost.AllocationAggregationOptions{
  ShareIdle: opencost.ShareWeighted,
  ShareSplit: opencost.ShareWeighted,
  Share: shareFilter,
  SharedLabels: map[string][]string{c.config.SharedInfraLabel:
    {c.config.SharedInfraLabelValue}},
}
return as.AggregateBy("model_name", opts)
```

```
opts := &opencost.AllocationAggregationOptions{
  ShareIdle: opencost.ShareNone,
  ShareSplit: opencost.ShareNone,
  Share: shareFilter,
  SharedLabels: map[string][]string{c.config.SharedInfraLabel:
    {c.config.SharedInfraLabelValue}},
}
return as.AggregateBy("model_name", opts)
```

What's Next?

GPUs use request only for allocation

- Need to update to max(request, usage)

Support for GPU Sharing

- MIG
- Time slicing

Collect deeper GPU specific metrics

Improve idle calculations

- Multi dimensional understanding of GPU Idle

Inference cost attribution by more dimensions

- tenant, workload, user

Interested in exploring these concepts in the enterprise?

Want to talk more in depth about FinOps?

Come speak with us in booth 182!

Thank you!

Alex – alexander.meijer@ibm.com

Sima – sima@il.ibm.com

References

1. https://www.forbes.com/sites/jemmagreen/2026/07/02/ai-costs-more-than-the-people-it-replaced/
2. https://fortune.com/2026/05/26/uber-coo-ai-spending-tokens-claude-code/
3. https://www.ft.com/content/1d37cc08-e0aa-45a4-a45d-4ad282529314
4. https://www.cncf.io/wp-content/uploads/2026/01/CNCF_Annual_Survey_Report_final.pdf

Appendices

Appendix A

llm-d Intro Slides

llm-d

Scalable, cost-effective AI platform



Vita Bortnikov
IBM Fellow, IBM Research

IBM

The evolving cycle of operating AI

Data preparation



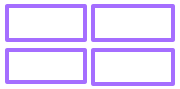
Workflow of steps for filtering and preparing data

Distributed training



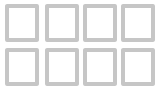
Long-running job on massive infrastructure

Model customization



Model customization with proprietary **sensitive** data (including real-time)

Inference



Highly sensitive to latency and throughput

Reinforcement Learning

Hours to days

Increasingly automated pipelines for cleaning, dedup, and synthetic data generation.



weeks to months

Training was the bottleneck — weeks of GPU time to build a model, then simple single-GPU serving.



hours to days

RL closes the loop between inference and training, requiring fast rollout generation at scale.



sub-second request

Frontier Models Need Multi-Node Inference, with Agents Multiply Inference 10–50x

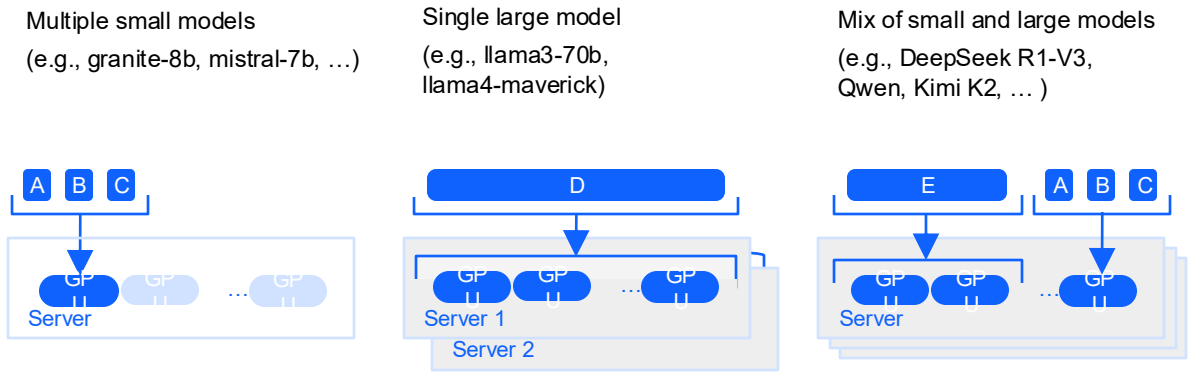
2/3 of all AI compute is now inference



The challenge of scaling inference

Distributed inference is essential for cost-effective scaling of AI,
but introduces unique **operationalization challenges**

- Load balancing **variable, resource-heavy** and hardware-affinity nature of requests
- **Ensuring SLO** and maintaining low latency under multi-tenant traffic
- Leveraging and managing **heterogeneous hardware** for better cost-efficiency
- Managing distributed **KV cache state** at scale as key part in inference efficiency
- **Orchestrating** prefill and decode phases; and multi-node serving



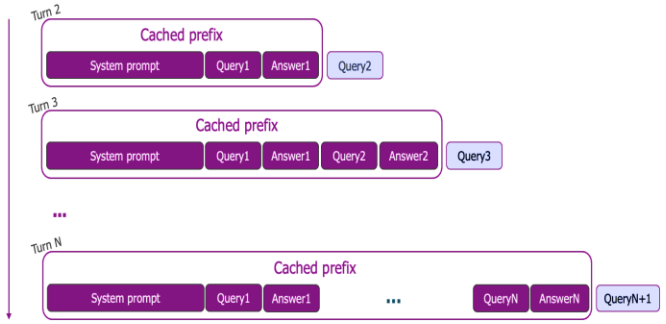
llm-d

KV cache management

KV cache hit rate is essential to make prefix-heavy workloads **cost-effective**

Conversational AI

Large input-to-output ratio



Agentic workflows

Very large input-to-output ratio (100:1)



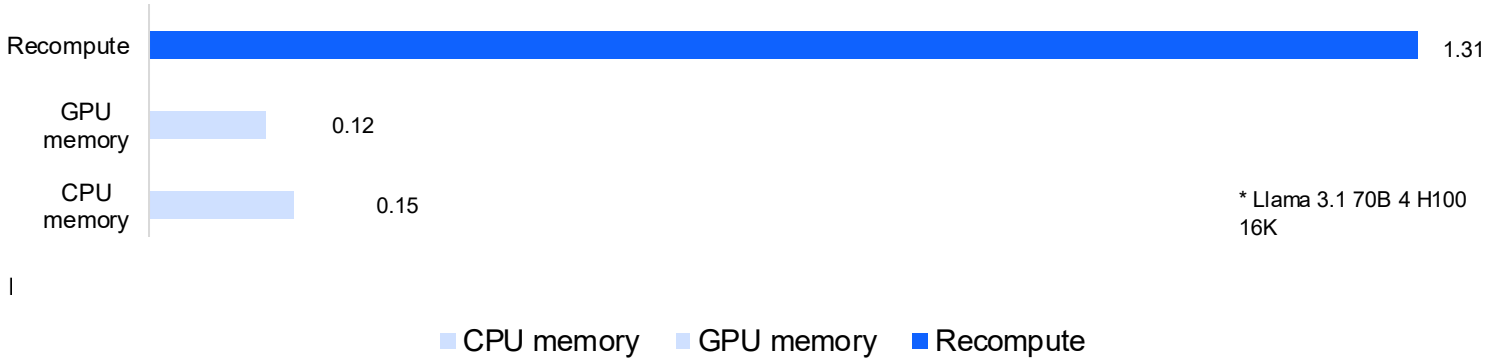
KV cache size grows linearly with context length and batch size

E.g., Llama 3.1 8B (fp8), 32K–128K context length results in 16–64 GB KV-cache size increase

KV cache grows to hundreds of terabytes for chatbot and agentic AI

Inference Time - TTFT (sec)

Inference speed-up with KV reuse

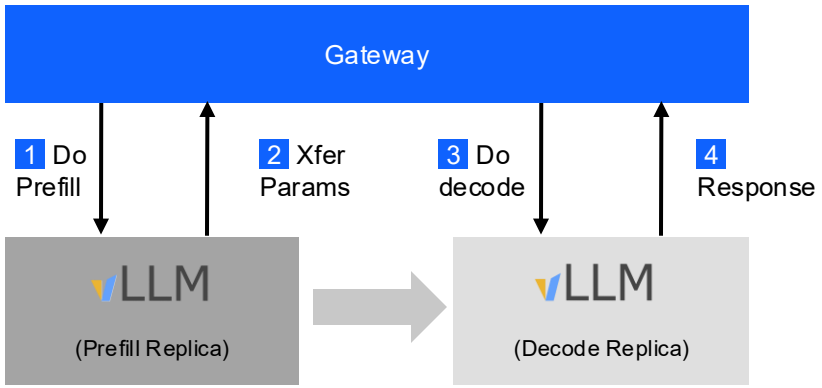


llm-d

Pipeline imbalance: prefill and decode behave very differently

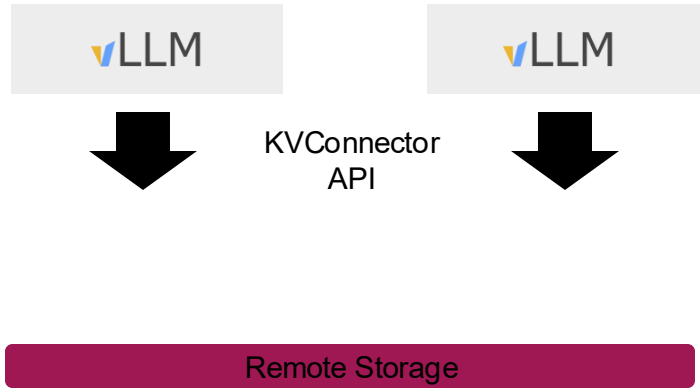
P/D disaggregation, from exploration to production operation

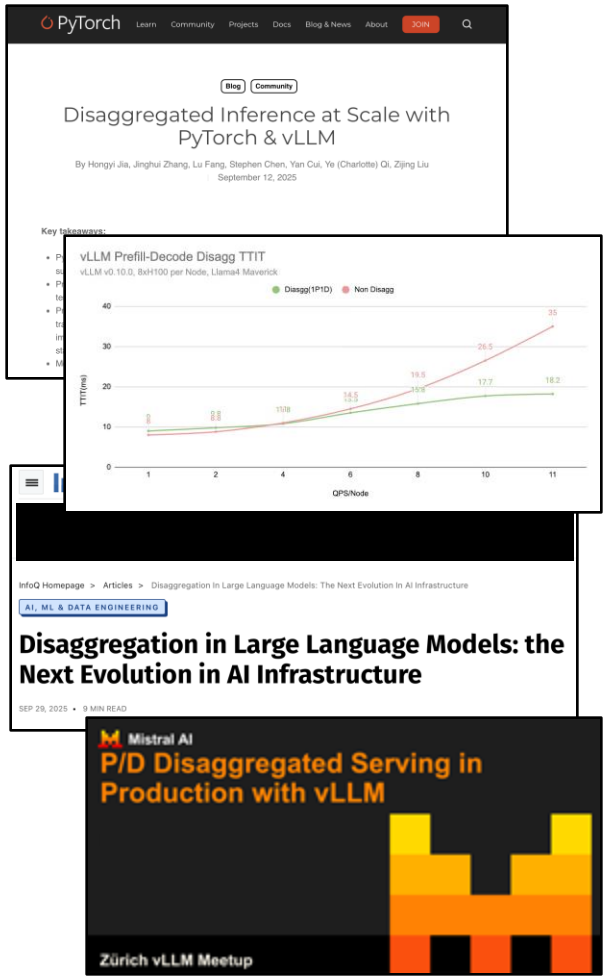
Prefill/Decode Disaggregation



Connector API

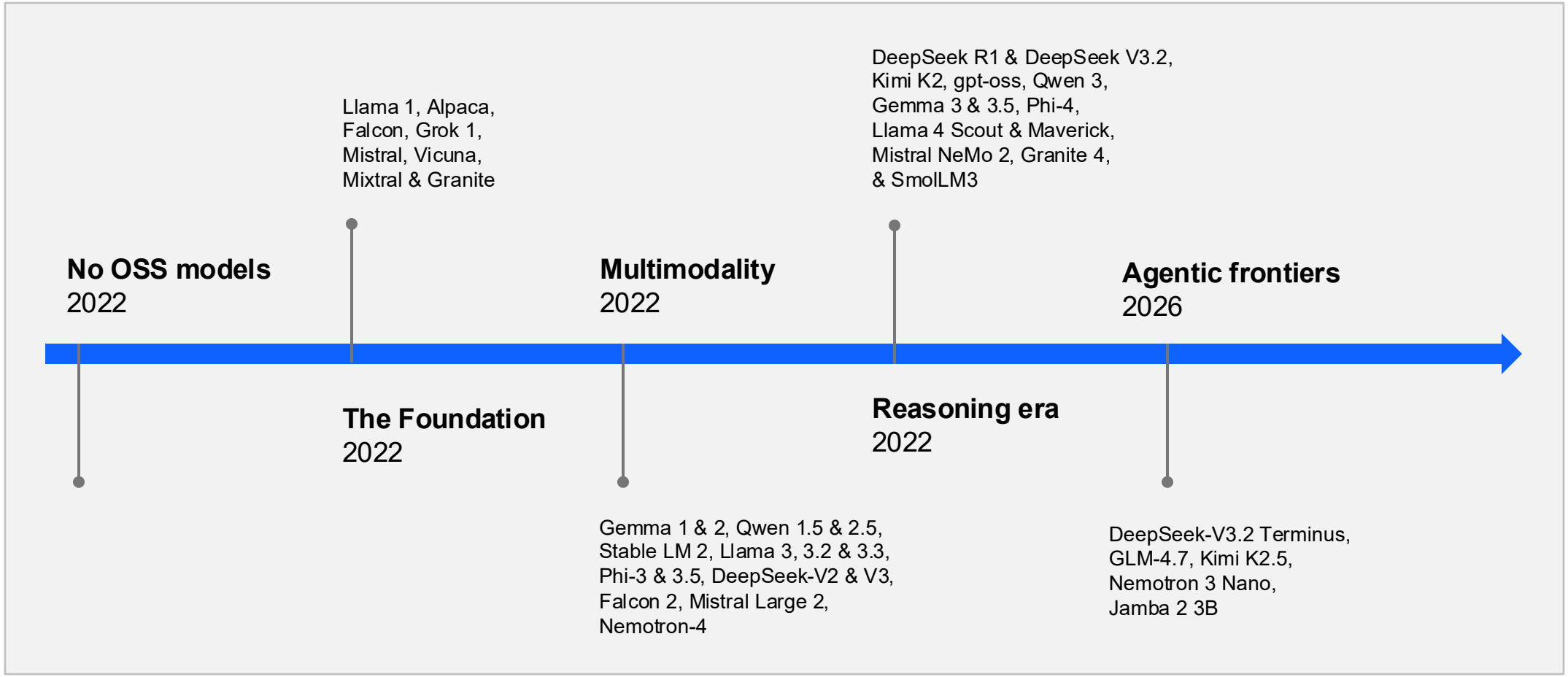
KV Cache Offloading





Key challenges on implementing and standardizing the optimizations/APIs needed for **disaggregated serving orchestration**

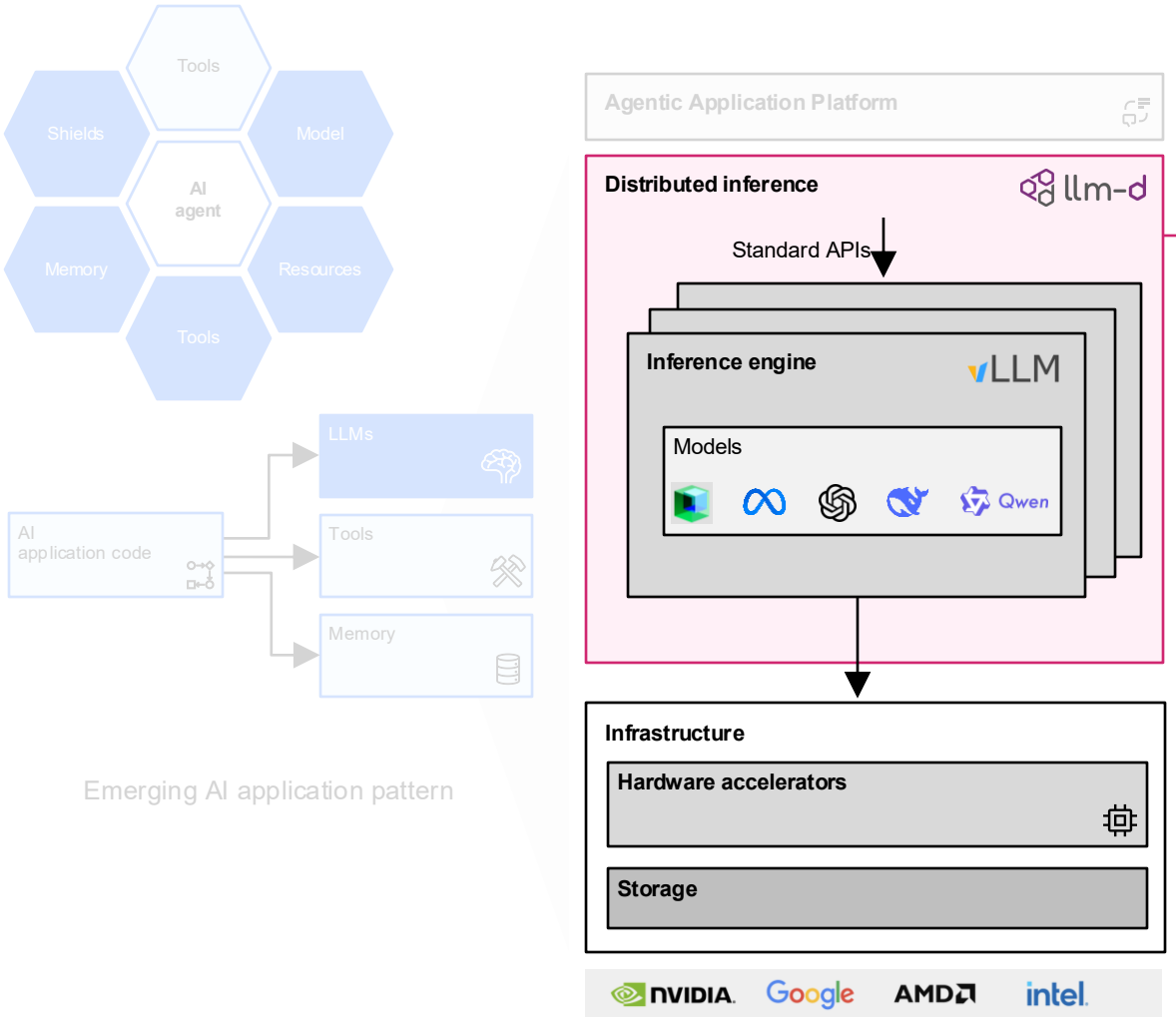
The OSS Model Ecosystem is Exploding



Models | Explosion in parameters
Reasoning capability

Agents | Multi-turn
Huge token demand

Establishing the foundation for a common distributed inference platform



Launch of llm-d project

May '25

Core components

Red Hat, IBM, Google

Expanded components

NVIDIA, intel, AMD

Validation, ecosystem, adoption exploration

CoreWeave, cisco, Hugging Face, Mistral AI, Lambda

Distributed inference layer that sits between AI applications and hardware

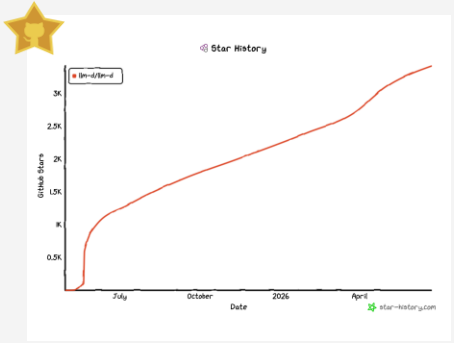
Delivering state-aware scheduling, prefill/decode disaggregation, and MoE parallelism through standard APIs

Maximize GPU utilization and **reduce serving costs** through intelligent request routing and resource disaggregation

Eliminating operational complexity with first true **cross-vendor** effort establishing a common stack for distributed inference

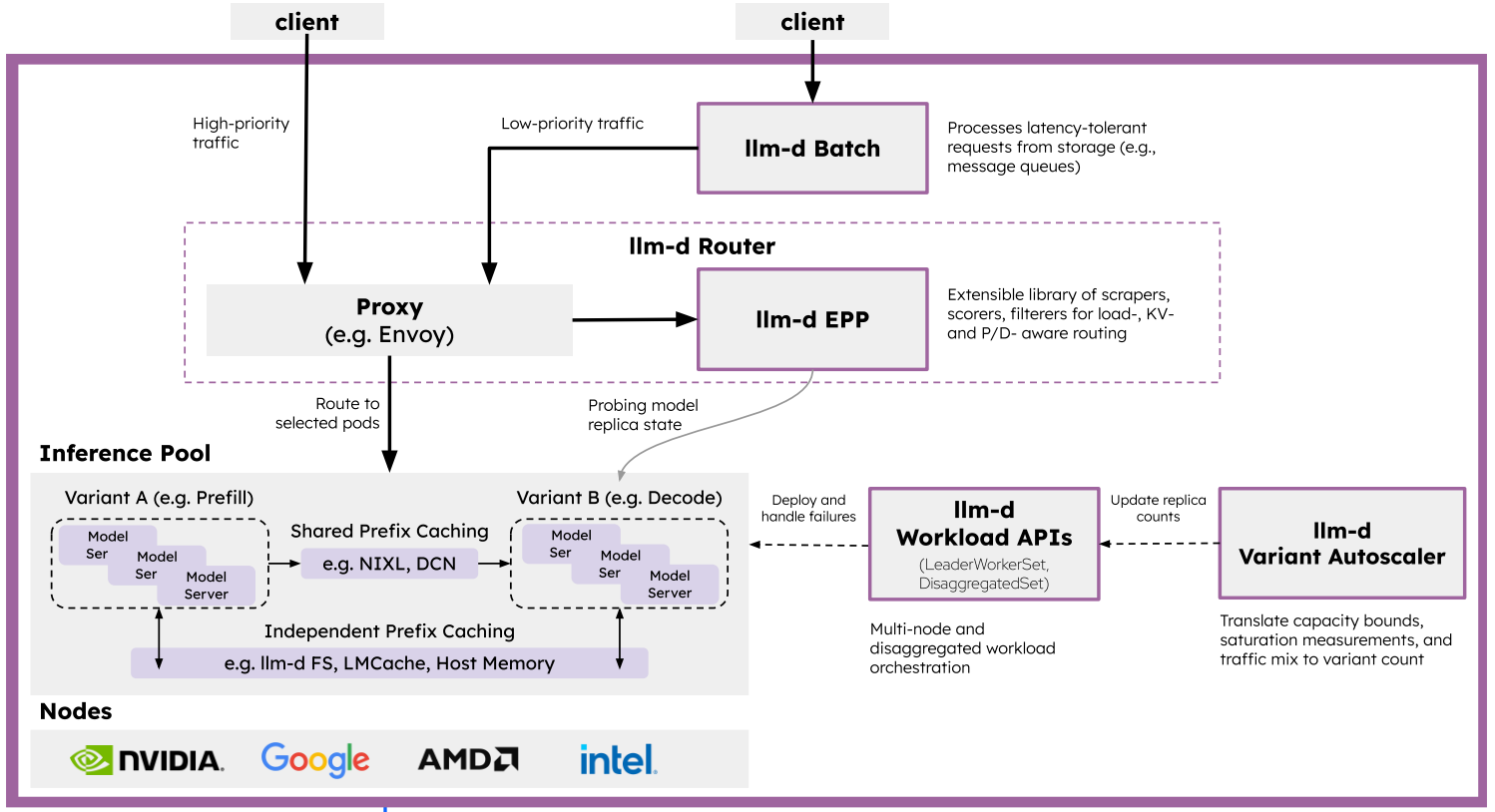
Fast growing adoption and contributor expansion among industry-leaders

Donated to CNCF in March'26, forming **one of the three pillars** for AI in cloud-native



CLOUD NATIVE COMPUTING FOUNDATION

llm-d Distributed inference platform for serving at scale



v0.1

Launch, May 20th

(4 founding contributors + 7 launch partners)

AMD · cisco · CoreWeave · Google · Hugging Face · IBM · intel · Lambda · Mistral AI · NVIDIA · ...

Unique platform value proposition

Operationalizability

modular and resilient architecture with native integration into Kubernetes via Inference Gateway API

Flexibility

cross-platform with extensible implementations of key composable layers of the stack

Well-lit paths

Tested blueprints that guide adopters from experimentation to real-world, production-grade LLM deployments

llm-d highlights

Technical and Community leadership

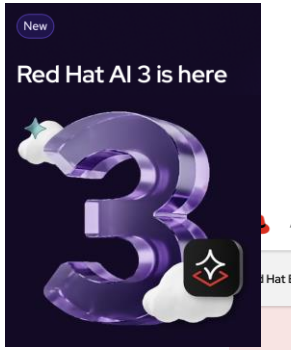
CNCF donation (March) and industry reaction

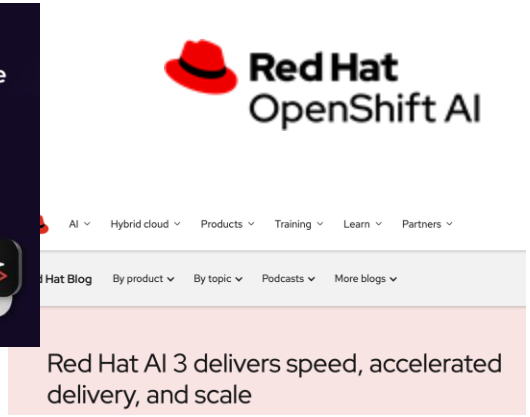


"The Common Blueprint for production-grade inference"
AI Inference Takes Center Stage At KubeCon Europe 2026
Forbes

"llm-d owns distributed orchestration"
Mark Collier,
Executive Director, PyTorch Foundation

"…one of three pillars of CNCF's AI strategy"
Jonathan Bryce
Executive Director, Linux Foundation / CNCF





Cutting edge innovation leading to real gains in production

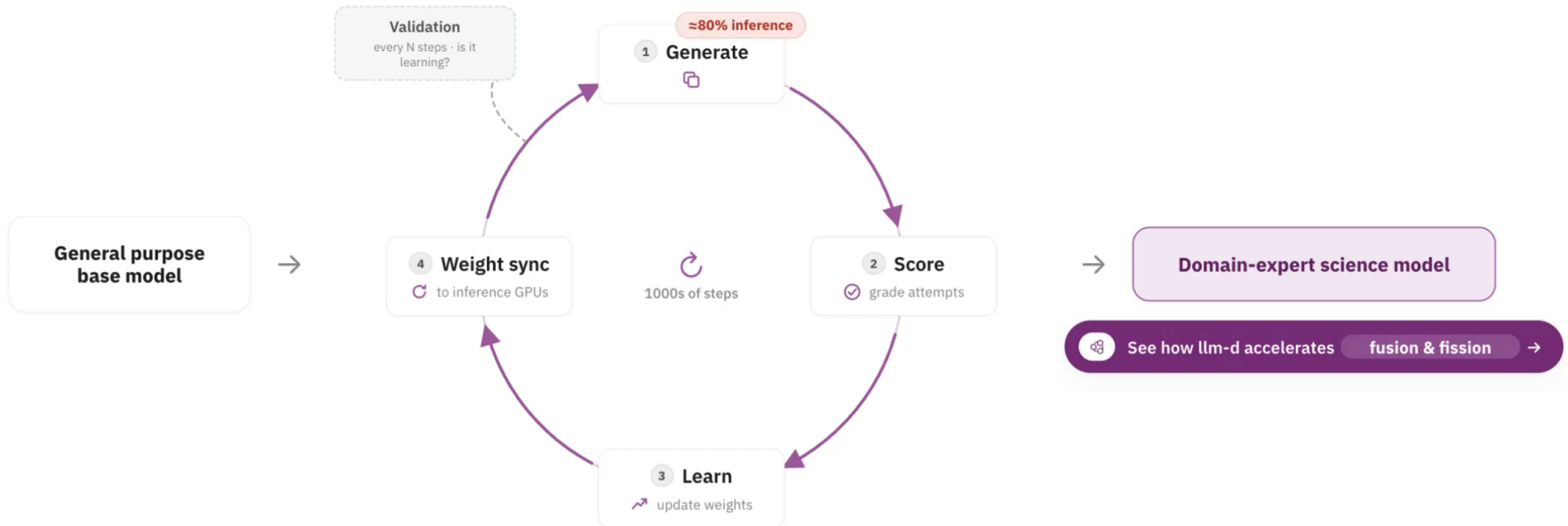
joint work with partners

3x higher output throughput and **2x faster TTFT**	**Up to 70% higher tokens/sec**	**30% throughput improvement**	**40% reduction in TTFT and ITL in Google Vertex**	**Deployment blueprint for running inference on CKS**	**llm-d powering purpose-built inference service for agentic workloads**

Growing list of users and contributors

MISTRAL AI_ · TESLA · cohere · snowflake · CapitalOne · JETBRAINS

PRIME Intellect · Novita · rebellions_ · MOREH · STELIA · Reflection



The loop runs on inference, so **Generate** sets the speed of discovery.

Thank You!