KubeCon

CloudNativeCon

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT

AI_dev

Open Source GenAI & ML Summit

China 2024

KubeCon

CloudNativeCon

THE LINUX FOUNDATION

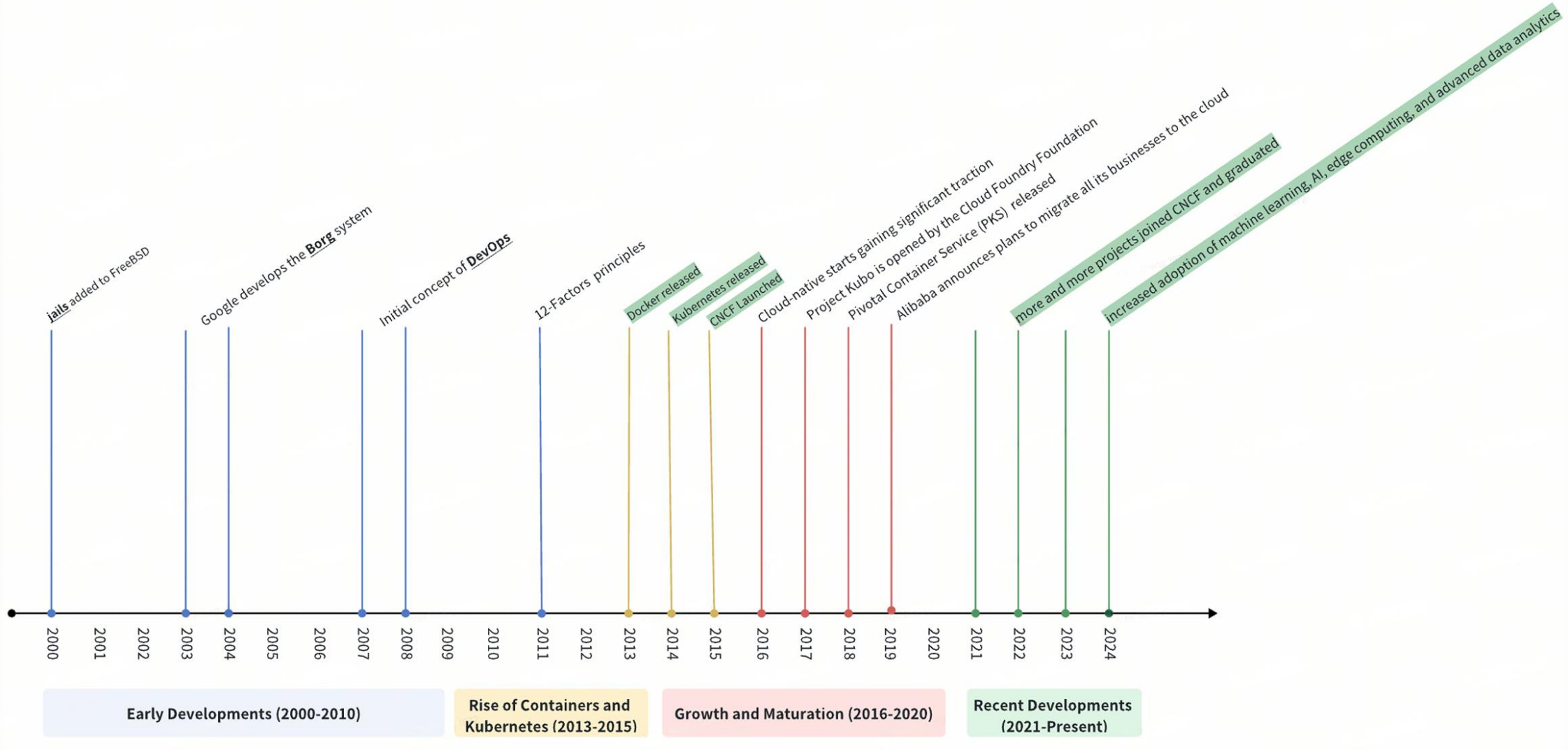
China 2024

AI Inference Performance Acceleration: Methods, Tools, and Deployment Workflows

Yifei Zhang / Lei Qian

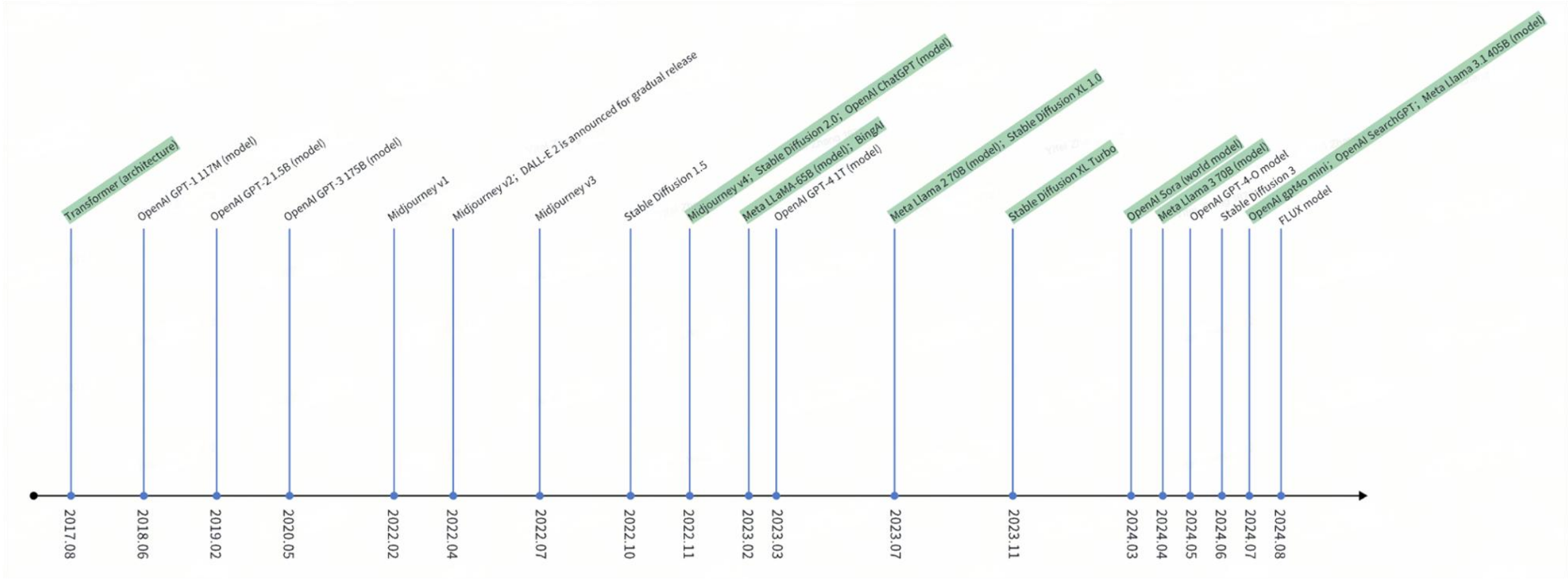
ByteDance Cloud Native Development Engineer

AI + Cloud Native Technology Trends



https://www.hivenet.com/post/cloud-computing-history
https://www.redapt.com/blog/cloud-native-applications-a-brief-history

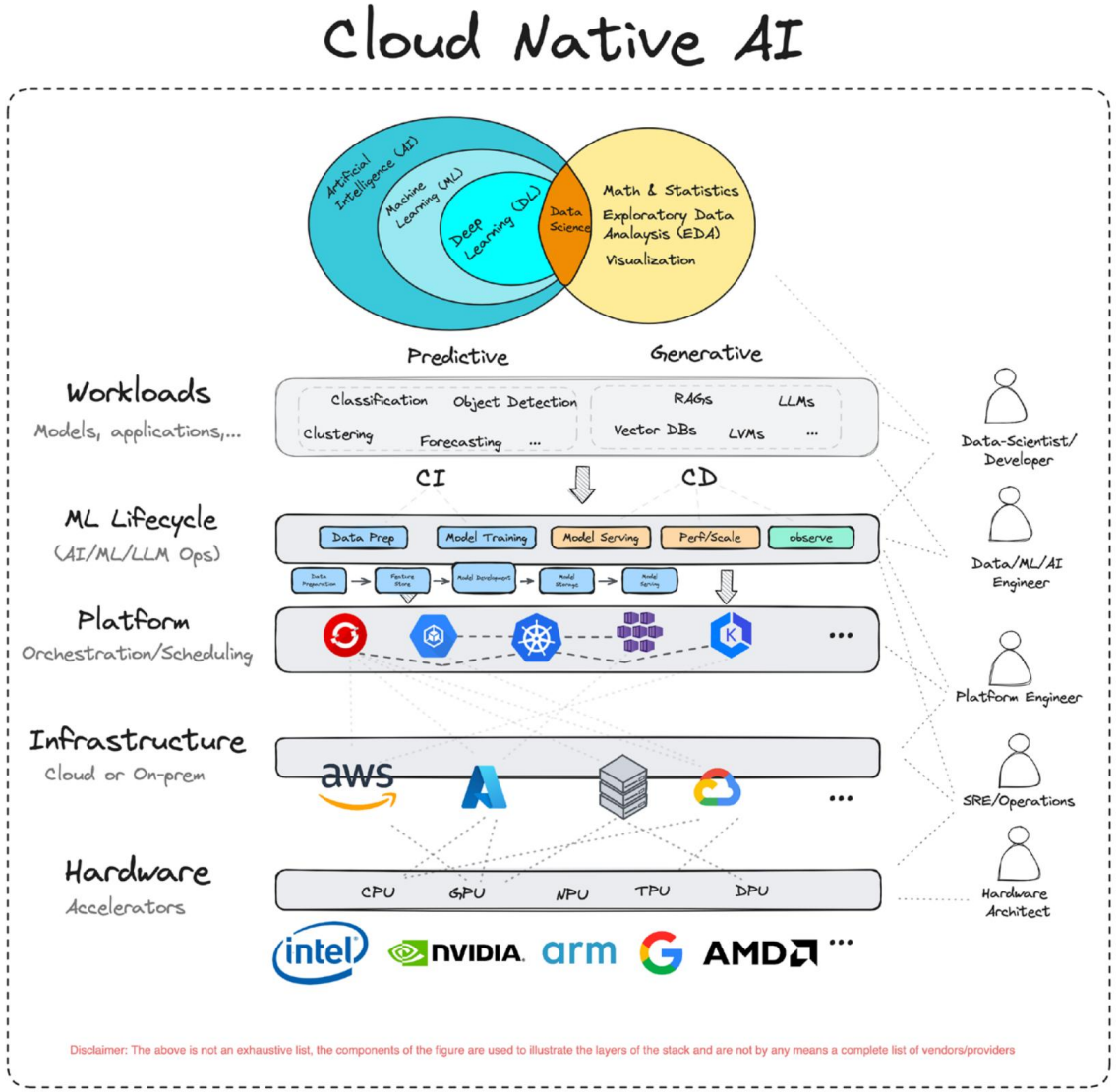
AI + Cloud Native Technology Trends



https://lifearchitect.ai/timeline/
https://nhlocal.github.io/AiTimeline/

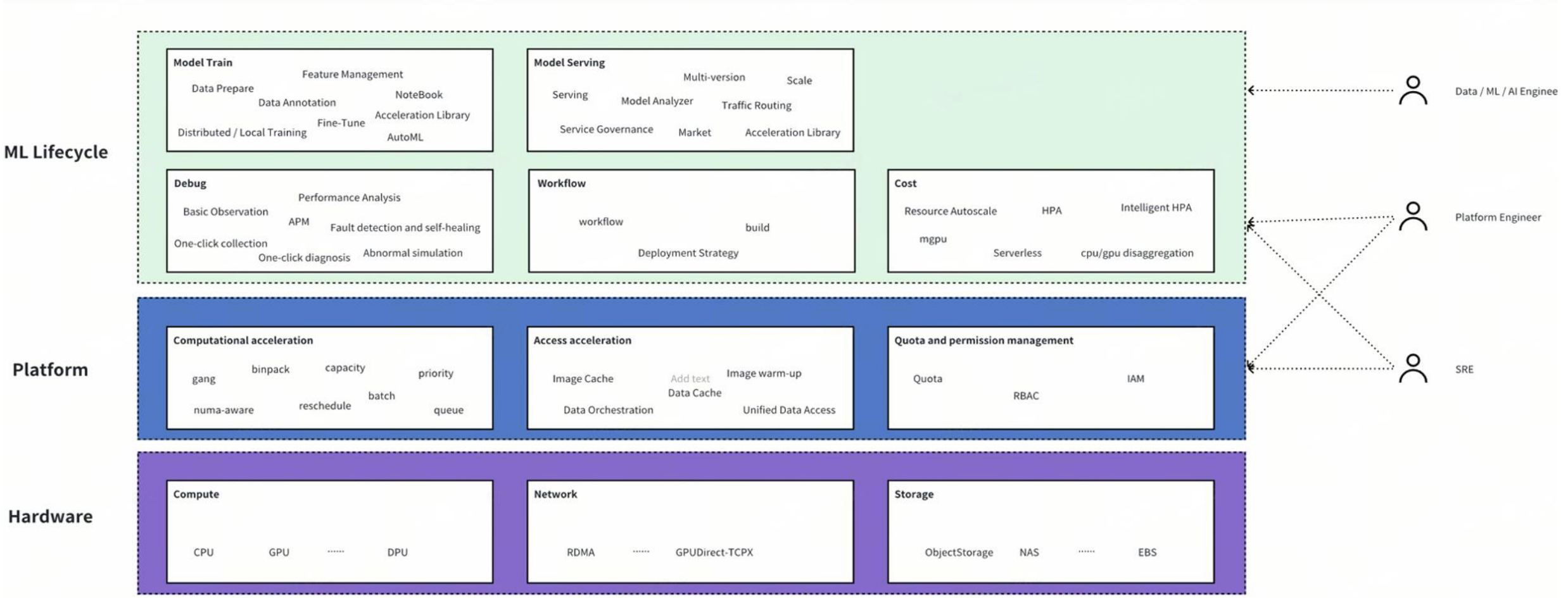
AI + Cloud Native Technology Trends

2024.03

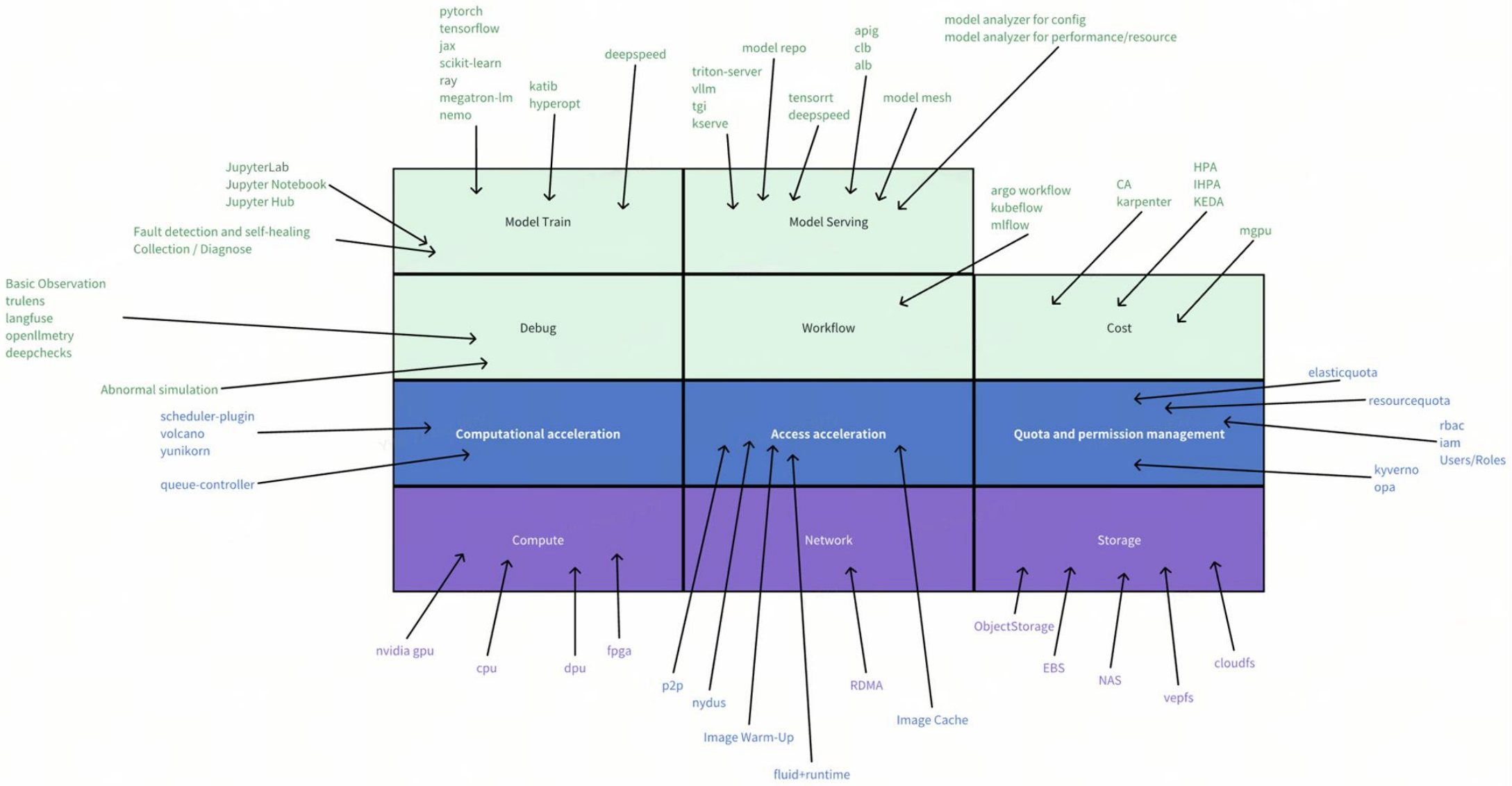


https://www.cncf.io/reports/cloud-native-artificial-intelligence-whitepaper/

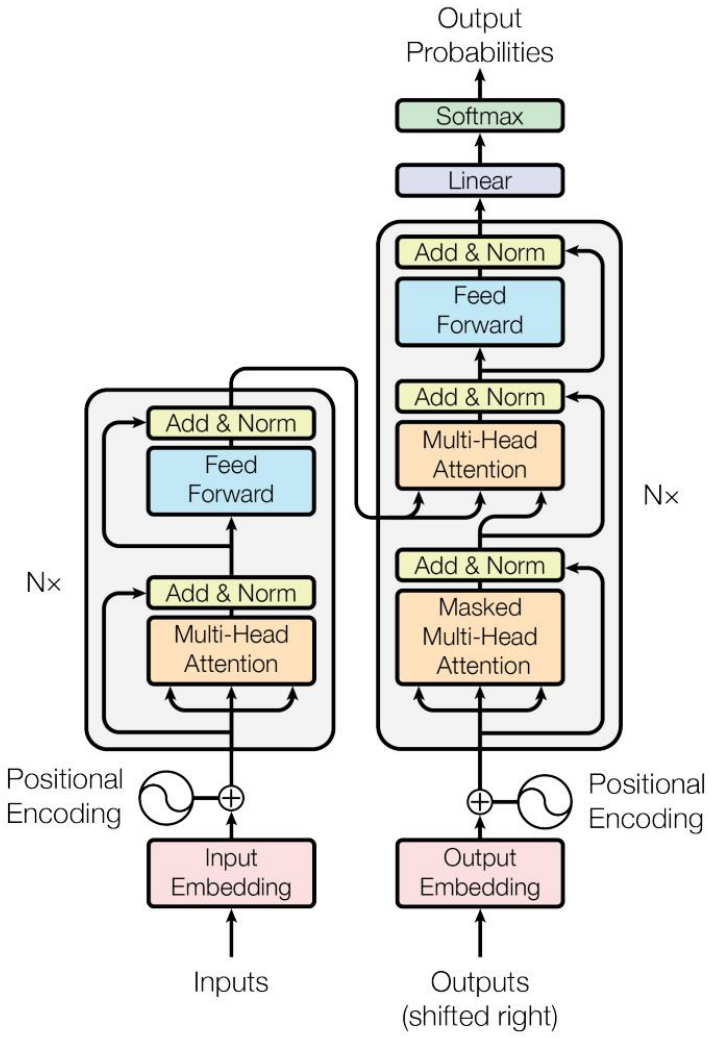
AI + Cloud Native Technology Trends

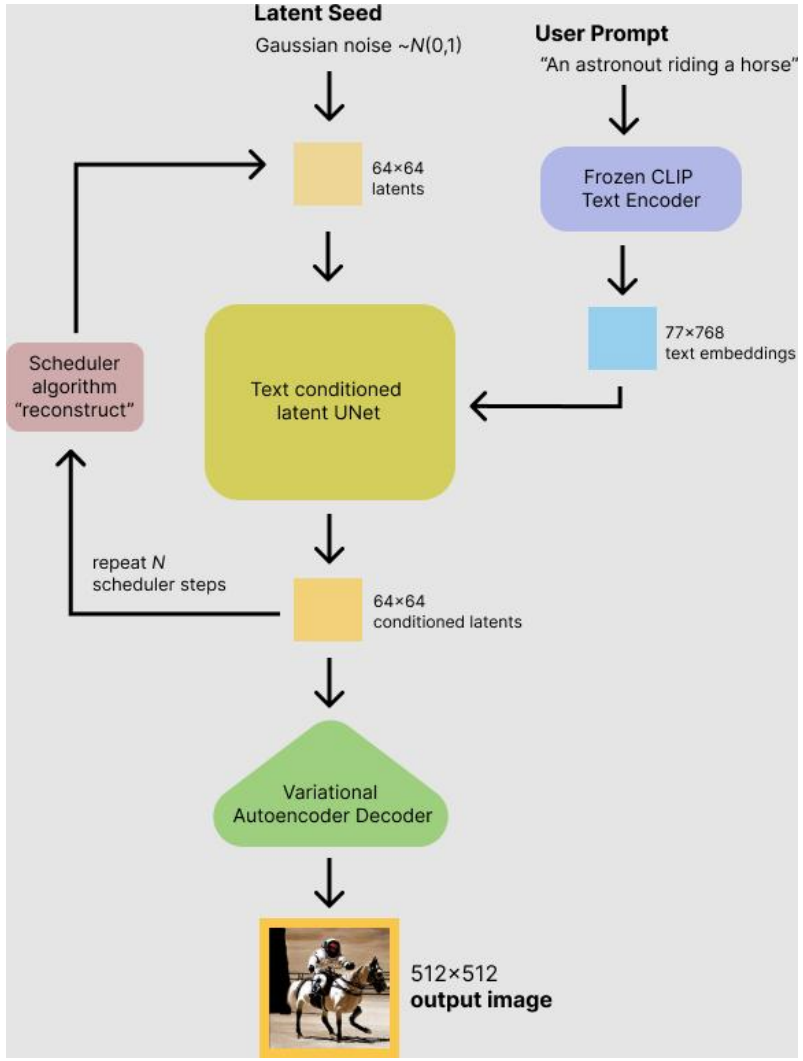


AI + Cloud Native Technology Trends



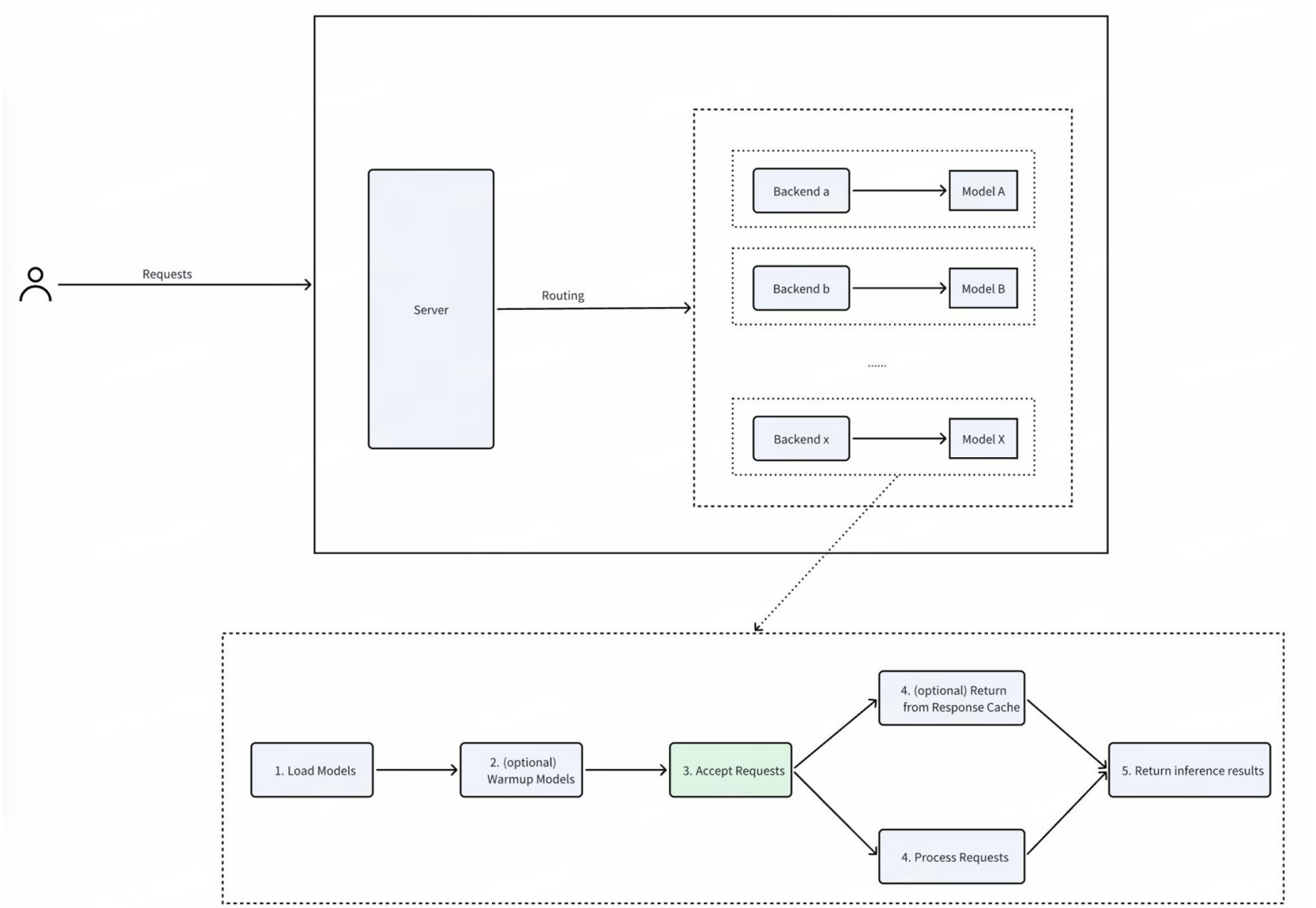
AI Inference





https://arxiv.org/abs/1706.03762
https://huggingface.co/blog/stable_diffusion

AI Inference



AI Inference

Methods	Description
Batching	Batching is one of the most effective ways to improve throughput for models that support it.
TensorRT / TensorRT-LLM Optimization	Converting models to TensorRT / TensortRT-LLM format can significantly improve performance.
Model Instances	Creating multiple instances of the same model allows concurrent execution and increases throughput, especially for real-time inferencing.
Response Caching	Enabling response caching at the server or model level can reduce latency by storing and retrieving responses for repeated requests.
Model Warmup	Avoid slow initial inference requests due to deferred initialization of models.
Quantization	Quantization techniques focus on representing data with less information while also trying to not lose too much accuracy. This often means converting a data type to represent the same information with fewer bits. Lower precision can also speed up inference because it takes less time to perform calculations with fewer bits.
Torch compile	`torch.compile` is a feature introduced in PyTorch 2.0 designed to optimize PyTorch code by JIT-compiling it into highly efficient kernels. This process aims to enhance the performance of PyTorch models with minimal code changes.
Distributed KV Cache	A distributed key-value (KV) cache in AI inference is a system designed to store and manage intermediate computational results across multiple nodes or servers to optimize the performance and efficiency of AI models, particularly large language models (LLMs). This technique is crucial for handling the high computational demands and large memory requirements of modern AI models.
Storage Access Acceleration	Optimize the performance of remotely pulling the model.

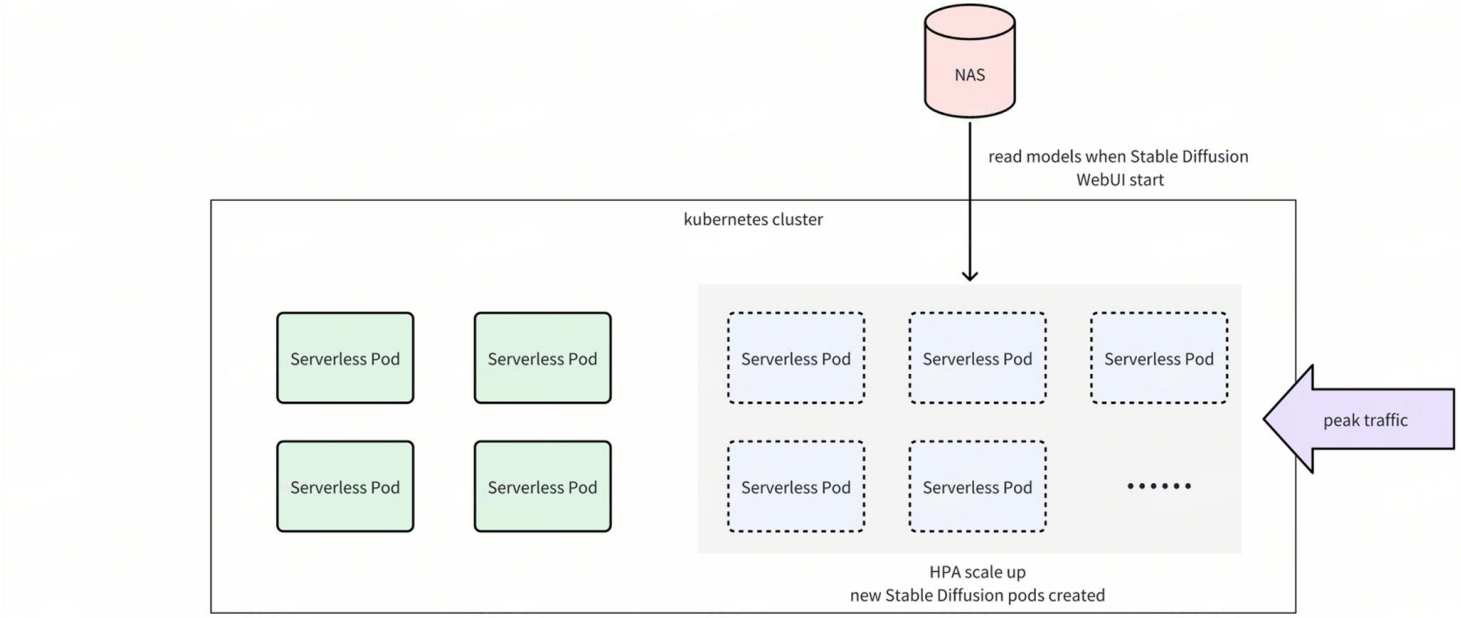
Storage Access Acceleration

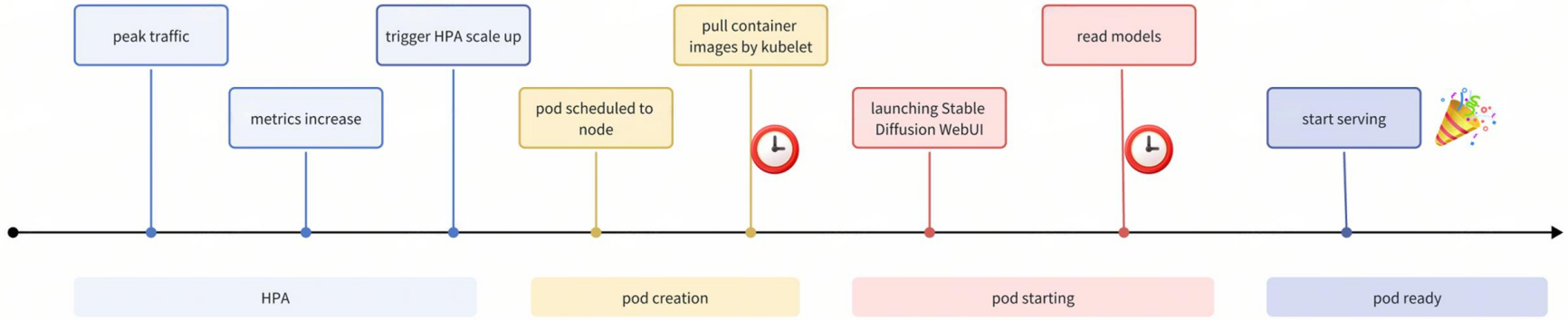
Scenario:
running Stable Diffusion WebUI with serverless Pods

Problem:
slow scaling speed

Time Consume:
- pull container images
- read Stable Diffusion models

Solutions:
- Image Cache
- Fluid + Alluxio

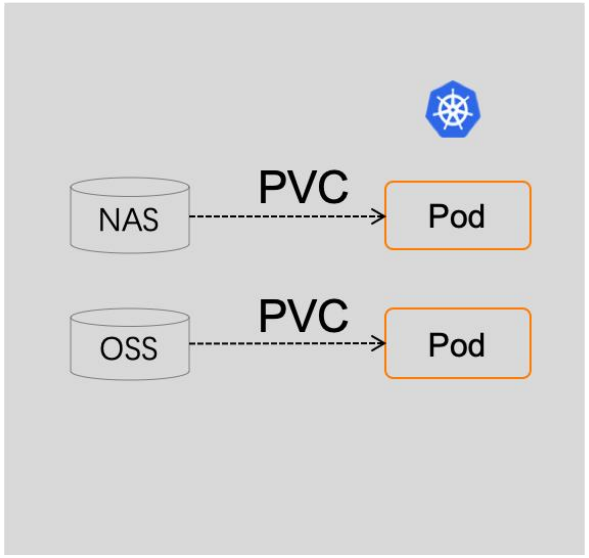




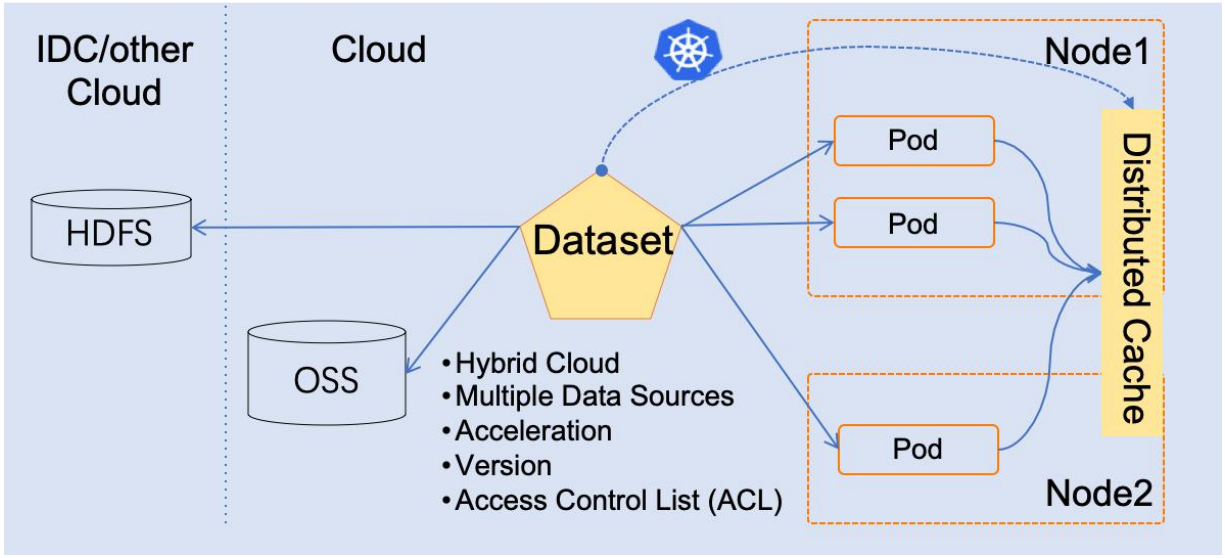
Storage Access Acceleration

- Kubernetes native
- manage dataset(compose、accelerate)
- support multiple storage runtime
- CNCF sandbox project

The perspective of CSI

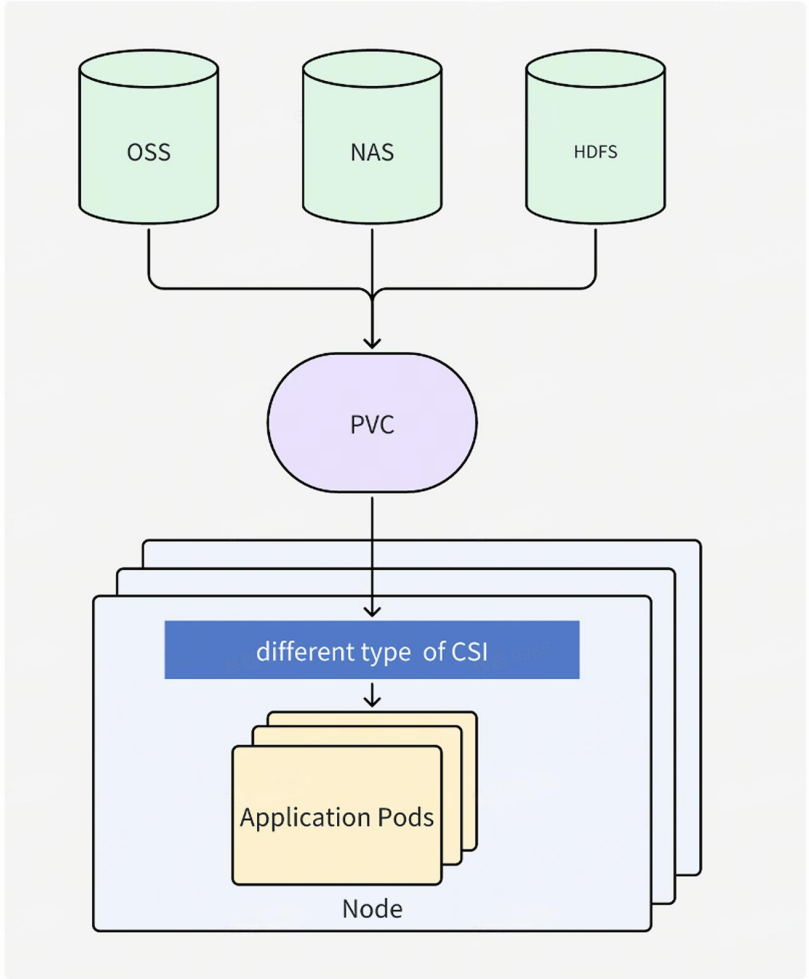


The data usage perspective of Fluid

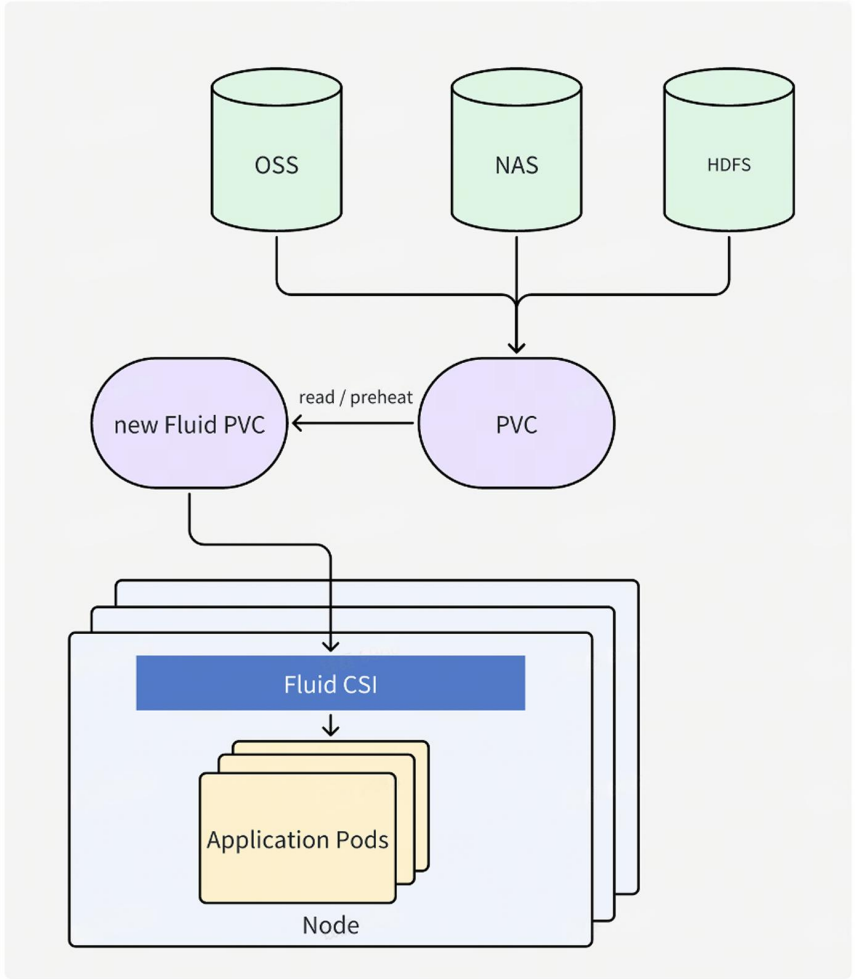


https://fluid-cloudnative.github.io/

Storage Access Acceleration



shift in usage patterns

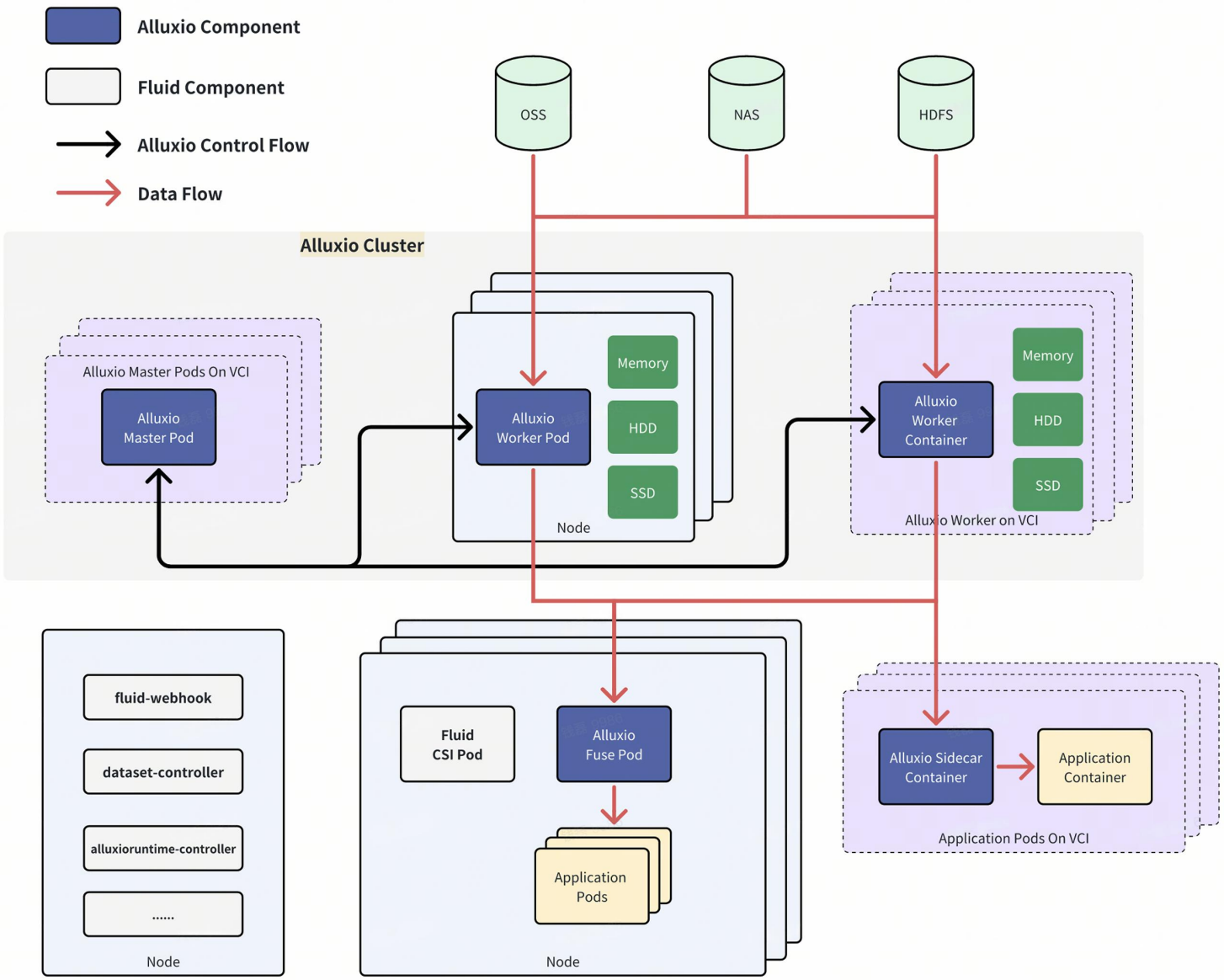


Storage Access Acceleration

Fluid Components:
deploy on node / using serverless pod

Alluxio Master Pod:
use serverless pod to avoid node unavailable

Alluxio Worker Pod:
- deploy on existing nodes to increase node utilization
- scale up for high performance and scale down to minimize costs



Storage Access Acceleration

time consumed for first text2img request

Scenario	P90 time consumed (compared with reading from NAS directly)
read from NAS directly	-
3 Alluxio Worker Pods **with** preheating	↑ 284%
6 Alluxio Worker Pods **with** preheating	↑ 480%
6 Alluxio Worker Pods **without** preheating	↑ 305%

What's next:

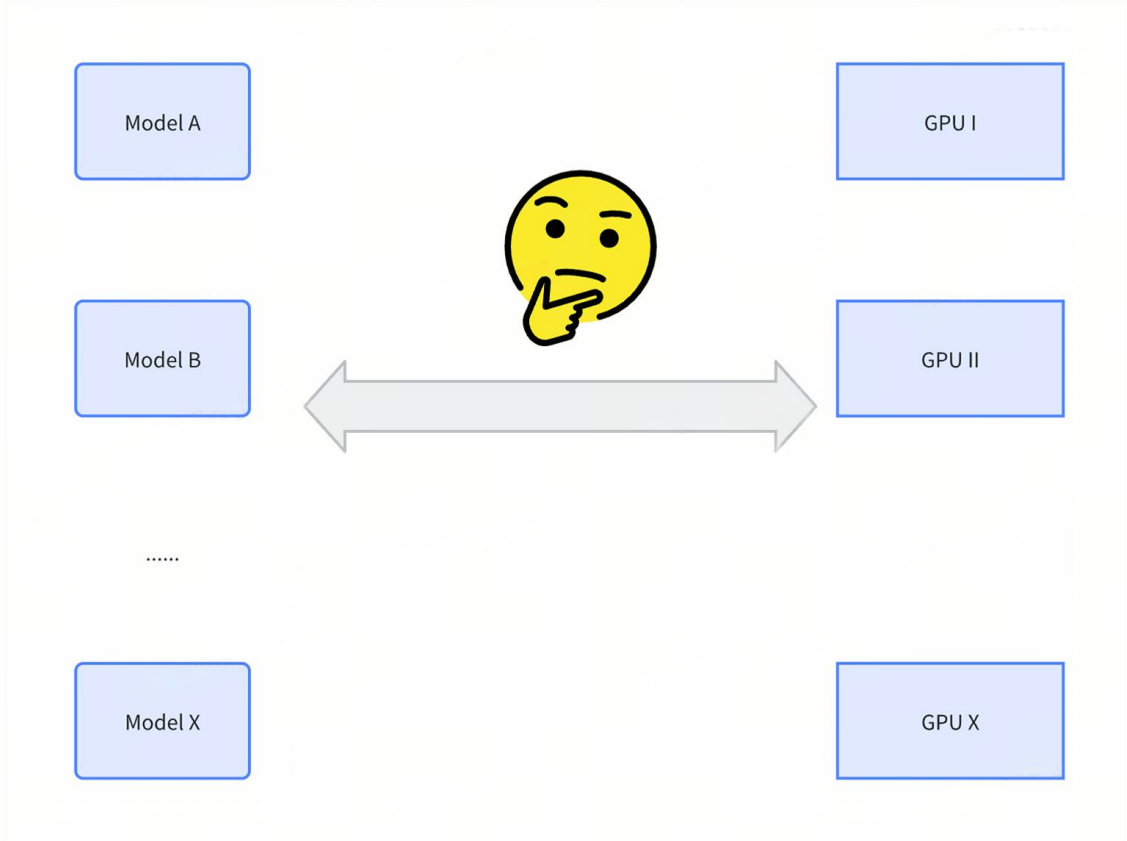
- faster and more stable storage runtime
- integrated with veTurboIO, an open source、high performance library for reading/writing models
- productization, more user-friendly
- validation at a larger scale

Benchmark Test

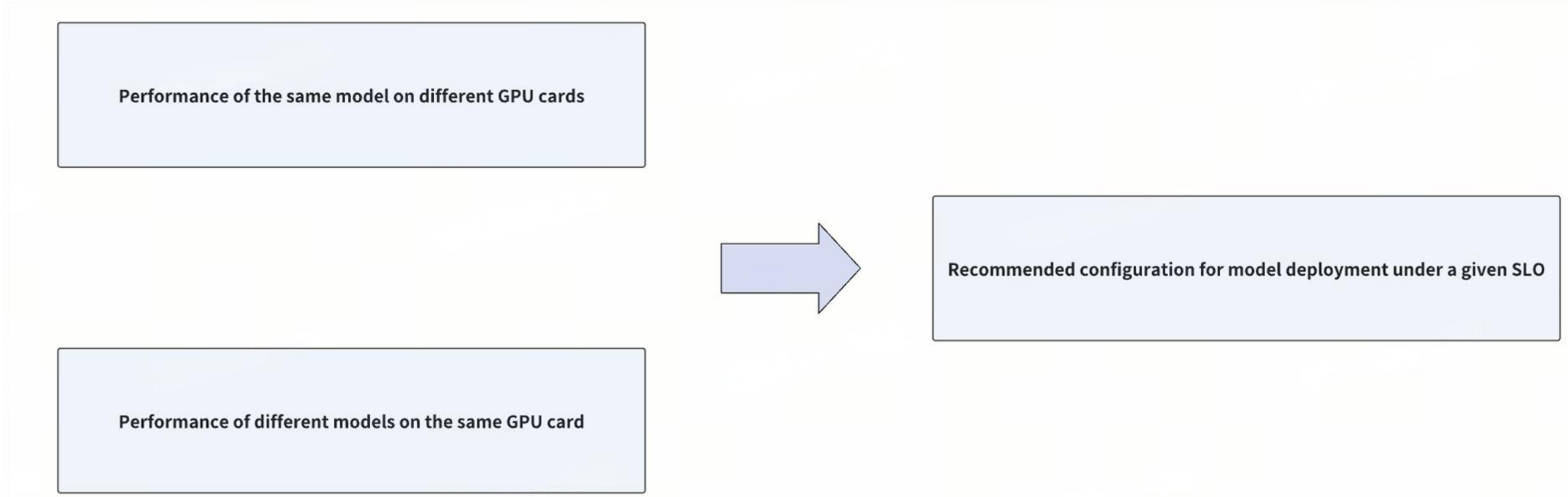
Customer scenario:
- Choose the right card for inference for different models

Question:
- What is the performance of the model?
- What kind of model is suitable for different cards?
- How to choose the best deployment configuration for the model?

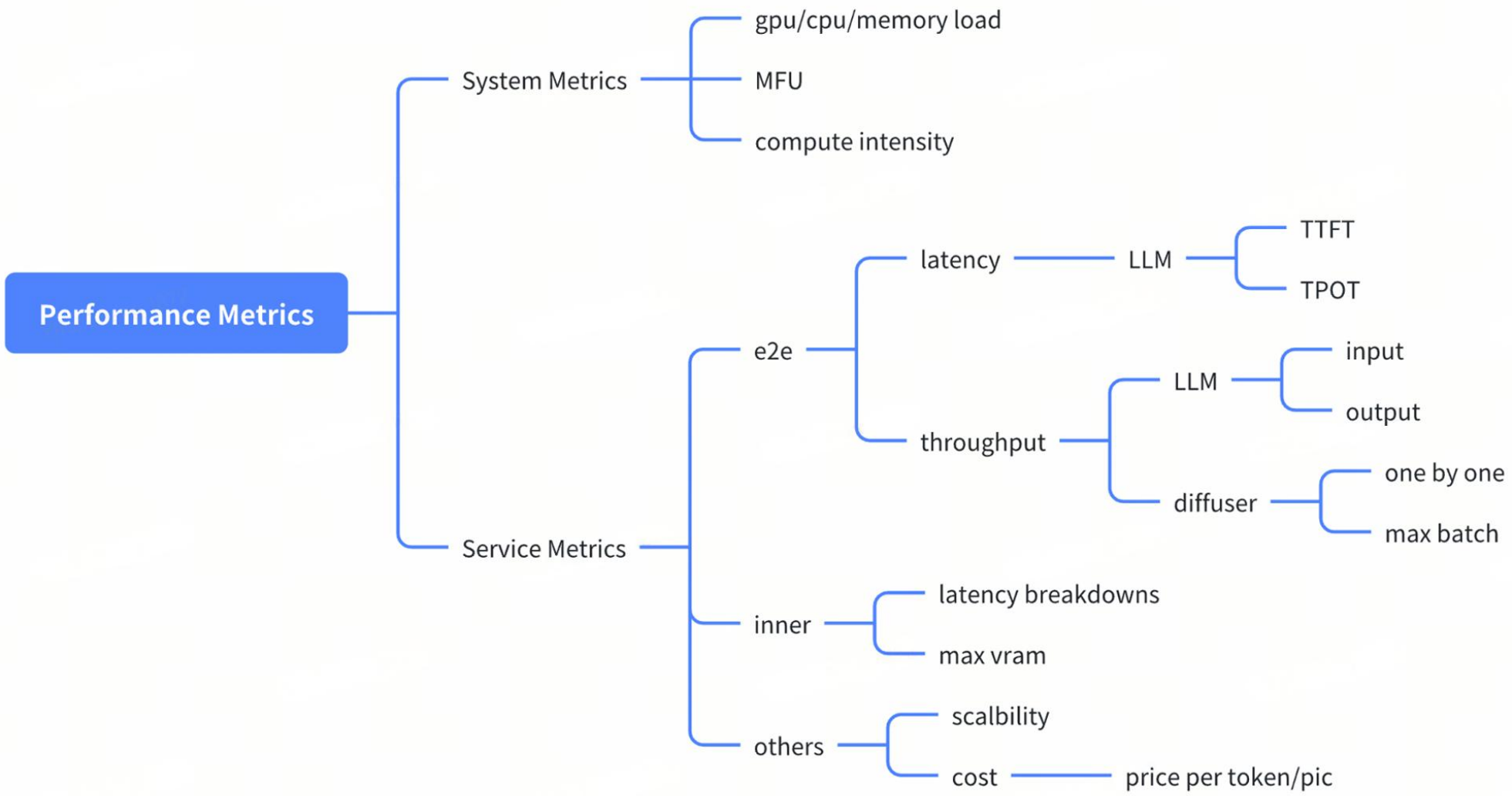
Solution:
- Unified performance evaluation tool
- Deployment configuration recommendation



Benchmark Test

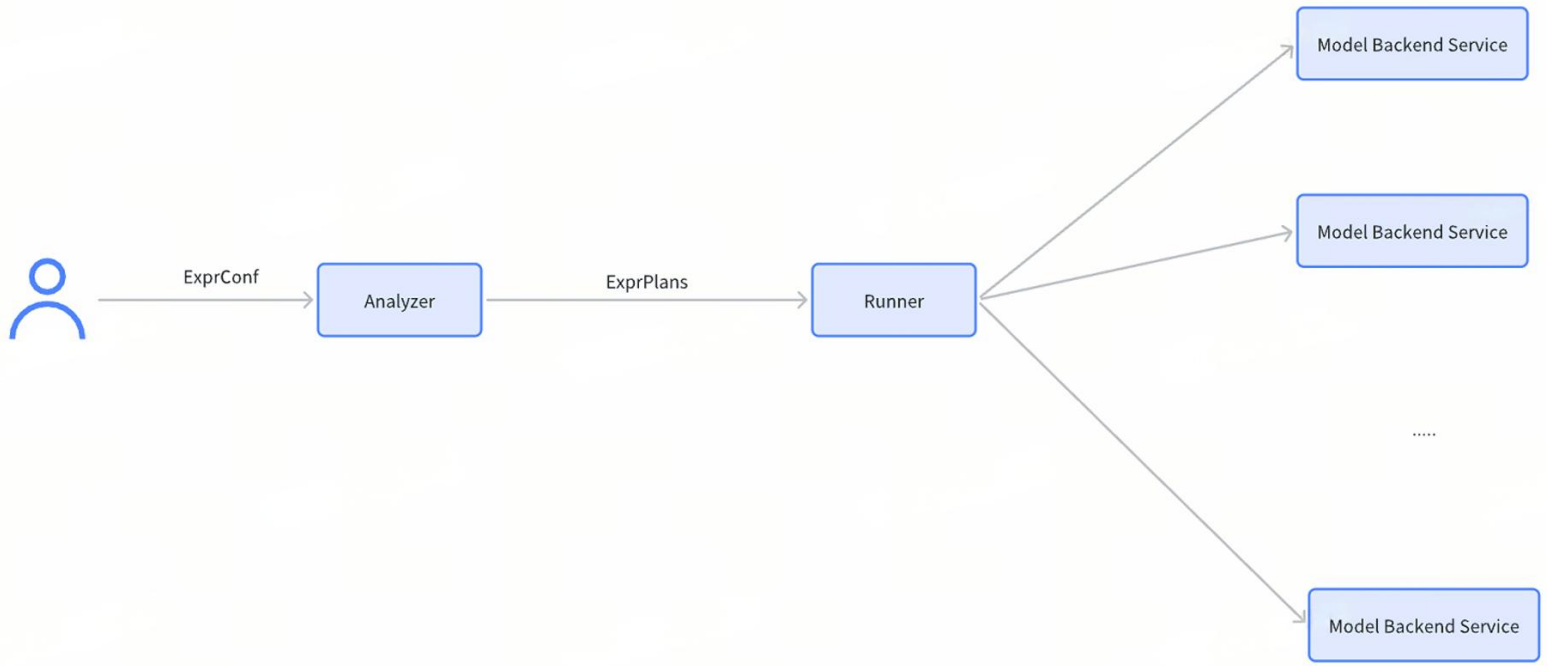


Benchmark Test



Benchmark Test


```
objectives:
 - MaxThroughput

backends:
 - name: VLLMOpenAIOnline
   models:
   - llama3
   task: TextGeneration
   environment_parameters:
     - name: DeviceIds
       value: [all]
   parameters:
   - name: TensorParallelSize
     mutate_policy: Fixed
     value: [1,2,4,8]
   - name: max_num_seqs
     mutate_policy: Fixed
     value: [128, 256, 512]
 - name: HuggingfaceTgi
   models:
   - llama3
   task: TextGeneration
   parameters:
   - name: TensorParallelSize
     mutate_policy: Fixed
     value: [1,2,4,8]

scenarios:
 - total_requests: 300
   warmup_requests: 30
   request_profile:
     distribution: Poisson
     arrival_rate: 15
     parameters:
     - name: MaxPromptToken
       mutate_policy: Fixed
       value: [100,1000]
     - name: PromptDistribution
       value: [Gamma]
     - name: GammaShape
       value: [1]
     - name: GammaScale
       value: [200]
```

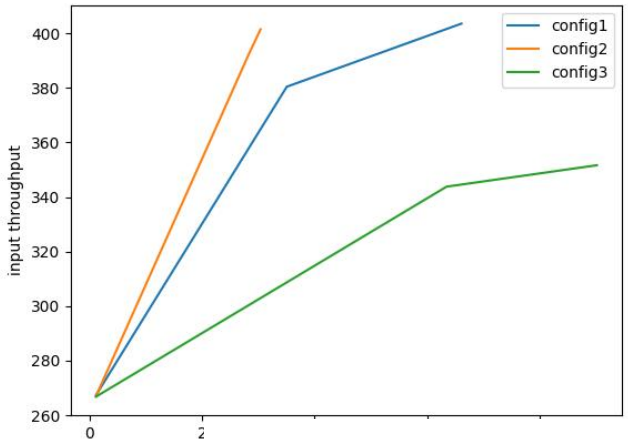


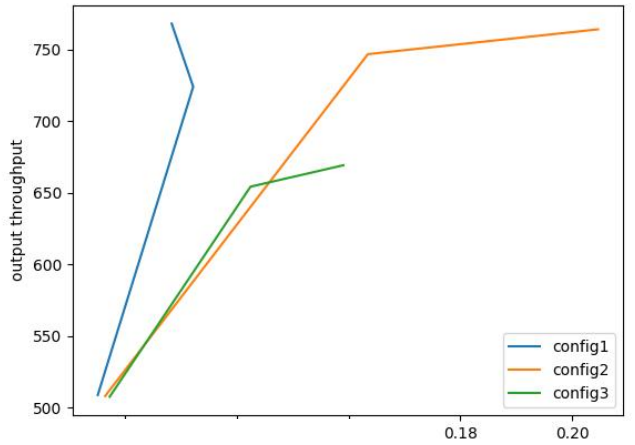
Benchmark Test

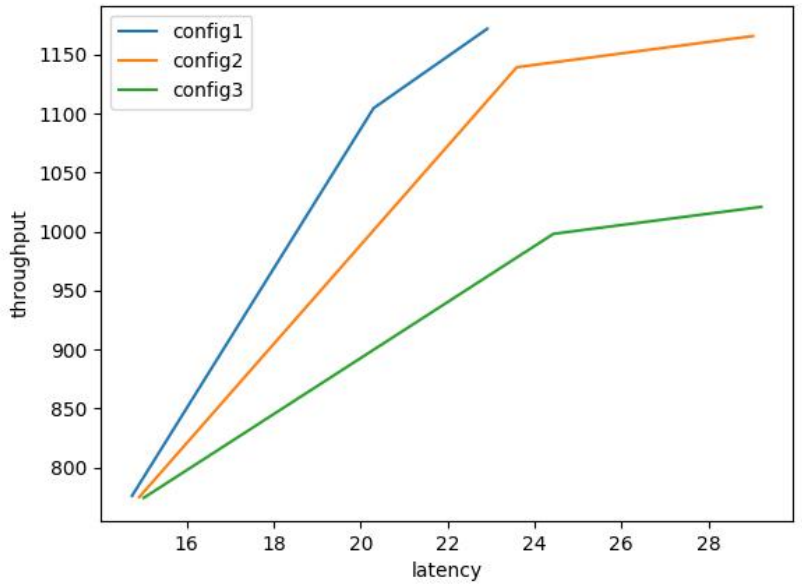

```
------
Result for ExprPlan 1
------
TTFT
count: 500
mean: 1.825434488832 s
stddev: 0.14498870880203713
p99: 1.968870144 s
p95: 1.968870144 s
p90: 1.968870144 s
p50: 1.95446656 s

TPOT
count: 500
mean: 0.09616399742588977 s
stddev: 0.00122213186684615
p99: 0.09747481725648295 s
p95: 0.09747417327874015 s
p90: 0.09747354984146982 s
p50: 0.09690578746456693 s

JCT
count: 500
mean: 14.03826216192 s
stddev: 0.29742351646087456
p99: 14.348171935573333 s
p95: 14.348090150400001 s
p90: 14.348010973866666 s
p50: 14.261501568 s

tokenize time
count: 500
mean: 0.000138018038 s
stddev: 6.355090792551172e-05
p99: 0.0002505621500000000 7 s
p95: 0.0002294281499999999 4 s
p90: 0.0002130584333333333 2 s
p50: 0.000131658 s

input_throughput
value: 495.2658436064178 token/s

output_throughput
value: 1139.9343303895107 token/s
```

```
------
Result for ExprPlan 2
------
TTFT
count: 500
mean: 1.381456792064 s
stddev: 0.7493186622894569
p99: 2.1823961053866676 s
p95: 2.148490496 s
p90: 2.148490496 s
p50: 1.598062592 s

TPOT
count: 500
mean: 0.17353241527836222 s
stddev: 0.05295802498117623
p99: 0.30187345284367456 s
p95: 0.28689303594330706 s
p90: 0.2686527295832021 s
p50: 0.15085138544881893 s

JCT
count: 500
mean: 23.420073532416 s
stddev: 6.842699711382243
p99: 39.93599110314667 s
p95: 38.0334781568 s
p90: 35.71695924906667 s
p50: 19.246099328 s

tokenize time
count: 500
mean: 0.00013838738 s
stddev: 6.276053674548014e-05
p99: 0.0002643714900000000 43 s
p95: 0.0002275032 s
p90: 0.0002124118333333333 2 s
p50: 0.0001306035 s

input_throughput
value: 457.0821161414881 token/s

output_throughput
value: 1052.048314502454 token/s
```

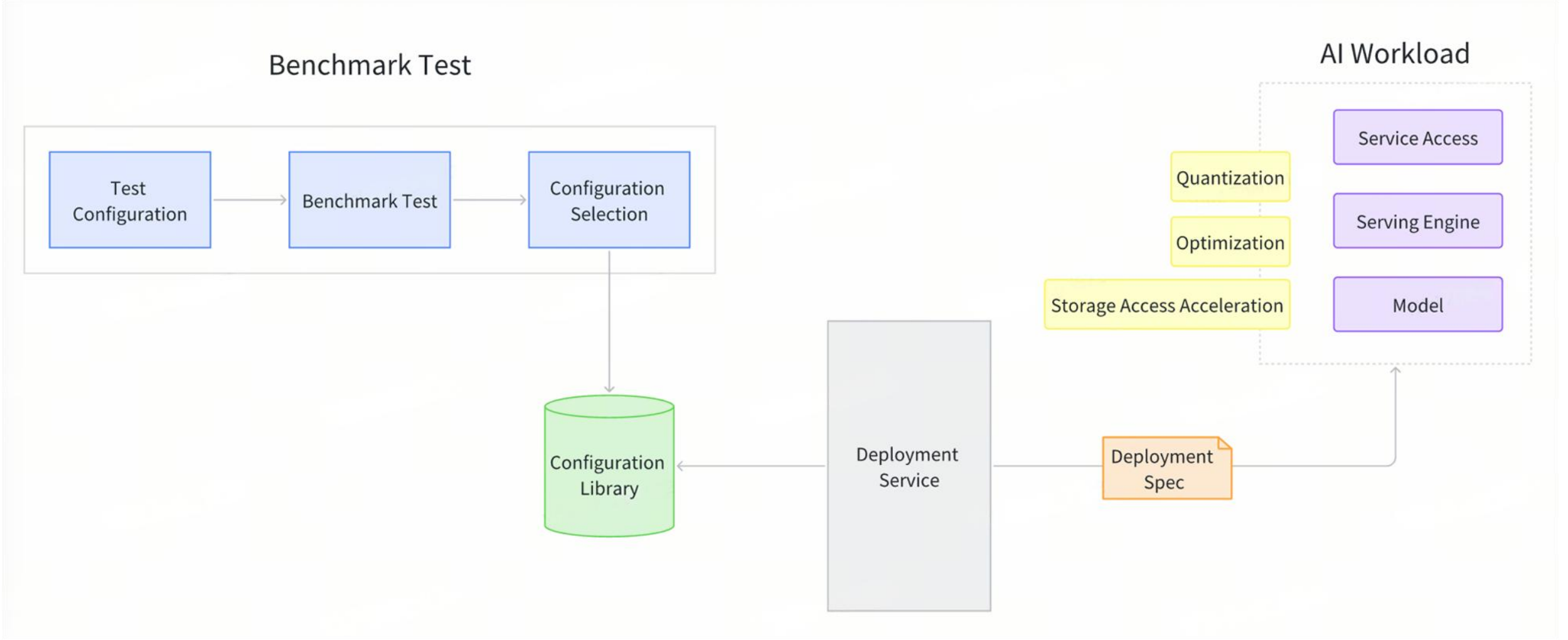
```
------
Result for ExprPlan 3
------
TTFT
count: 500
mean: 0.942095332352 s
stddev: 0.8856858033330711
p99: 2.204497408 s
p95: 2.204497408 s
p90: 2.204497408 s
p50: 0.283148288 s

TPOT
count: 500
mean: 0.17581645832869292 s
stddev: 0.05930062742010439
p99: 0.3337383068690814 s
p95: 0.30376179885354326 s
p90: 0.2549980297070866 s
p50: 0.15912877152755905 s

JCT
count: 500
mean: 23.270785540096 s
stddev: 7.637014045701006
p99: 43.68257130837334 s
p95: 39.875554790399995 s
p90: 33.6825561088 s
p50: 21.42635904 s

tokenize time
count: 500
mean: 0.000138566612 s
stddev: 6.550952676029779e-05
p99: 0.0002682519900000000 1 s
p95: 0.00022738425 s
p90: 0.0002115468333333333 3 s
p50: 0.0001299945 s

input_throughput
value: 352.14318735801106 token/s

output_throughput
value: 810.5144210211 token/s
```







Deployment Workflow



Thanks!

Cloud Native AI Suite User Group

扫描群二维码，立刻加入该群

该二维码永久有效