



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

REST in Peace: AI Needs to Be Async - Meet Asya



Artem Yushkovskiy, Delivery Hero SE

```
start() ->
  spawn(ch1, init, []).
```

```
alloc() ->
  ch1 ! {self(), alloc},
  receive
    {ch1, Res} ->
      Res
  end.
```

```
free(Ch) ->
  ch1 ! {free, Ch},
  ok.
```

```
init() ->
  register(ch1, self()),
  Chs = channels(),
  loop(Chs).
```

```
loop(Chs) ->
  receive
    {From, alloc} ->
      {Ch, Chs2} = alloc(Chs),
      From ! {ch1, Ch},
      loop(Chs2);
    {free, Ch} ->
      Chs2 = free(Ch, Chs),
      loop(Chs2)
```

What Do You Think When You Hear “Actors”?



```
public class HelloWorld extends AbstractBehavior<HelloWorld.Greet> {

  public static record Greet(String whom, ActorRef<Greeted> replyTo) {}
  public static record Greeted(String whom, ActorRef<Greet> from) {}

  public static Behavior<Greet> create() {
    return Behaviors.setup(HelloWorld::new);
  }

  private HelloWorld(ActorContext<Greet> context) {
    super(context);
  }

  @Override
  public Receive<Greet> createReceive() {
    return newReceiveBuilder().onMessage(Greet.class, this::onGreet).build();
  }

  private Behavior<Greet> onGreet(Greet command) {
    getContext().getLog().info("Hello {}", command.whom);
    command.replyTo.tell(new Greeted(command.whom, getContext().getSelf()));
    return this;
  }
}
```



What Do You Think When You Hear “Actors”?

```
start() ->
  spawn(ch1, init, []).
```

```
alloc() ->
  ch1 ! {self(), alloc},
  receive
    {ch1, Res} ->
      Res
  end.
```

```
free(Ch) ->
  ch1 ! {free, Ch},
  ok.
```

```
init() ->
  register(ch1, self()),
  Chs = channels(),
  loop(Chs).
```

```
loop(Chs) ->
  receive
    {From, alloc} ->
      {Ch, Chs2} = alloc(Chs),
      From ! {ch1, Ch},
      loop(Chs2);
    {free, Ch} ->
      Chs = free(Ch, Chs),
      loop(Chs2)
  end.
```

```
public class HelloWorld extends AbstractBehavior<HelloWorld.Greet> {

  public static record Greet(String whom, ActorRef<Greeted> replyTo) {}
  public static record Greet(String whom, ActorRef<Greet> from) {}

  public static P<Greet>
    return P<Greet>()
  }

  private ActorRef<Greet>
    super()
  }

  @Override
  public ActorRef<Greet>
    return this::onGreet).build();
  }

  private Behavior<Greet>
    and) {
    getContext().getSelf().tell(command.whom,
    command.replyTo.tell(new Greeted(command.whom, getContext().getSelf()));
    return this;
  }
}
```



asya.sh



AI-powered food image enhancement at Delivery Hero



Our Edge Cases



Who I am



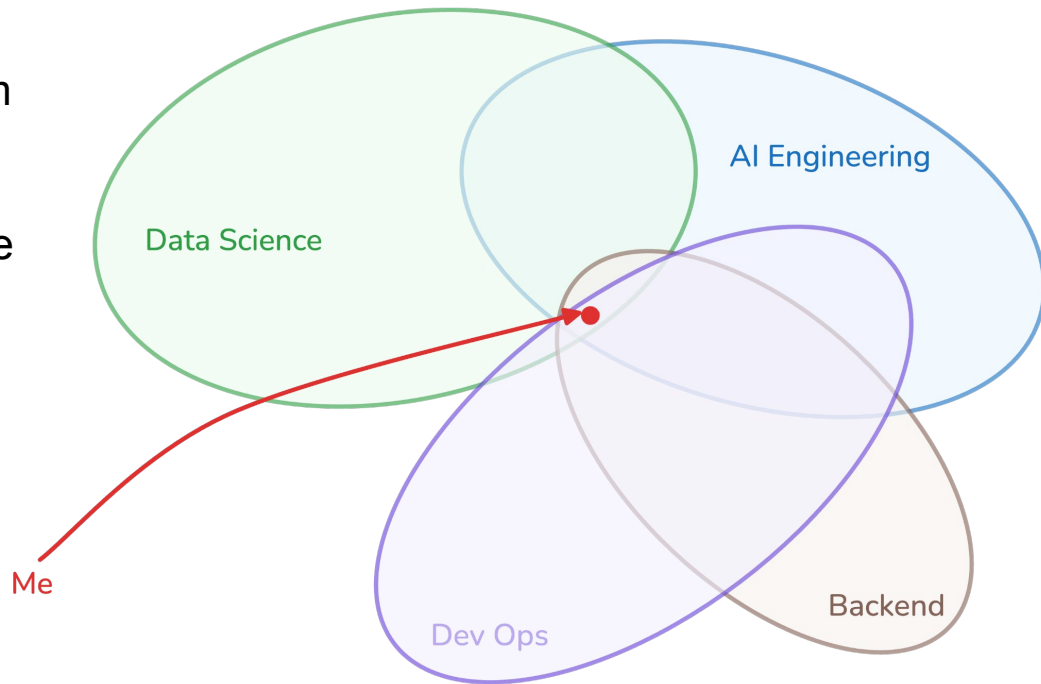
KubeCon



CloudNativeCon

Europe 2026

- **@atemate** aka. Artem Yushkovskiy
- **The bridge** between DS and platform worlds
- **~2015-2018-ies**: Web Security, Distributed systems, Automated Code Analysis
- **~2020-2023**: MLOps, ML Platforms
- **~2023-today**: Around-AI Engineering (High-load serving, Ops, Experimentation and Delivery)



[1/3] The Starting Point

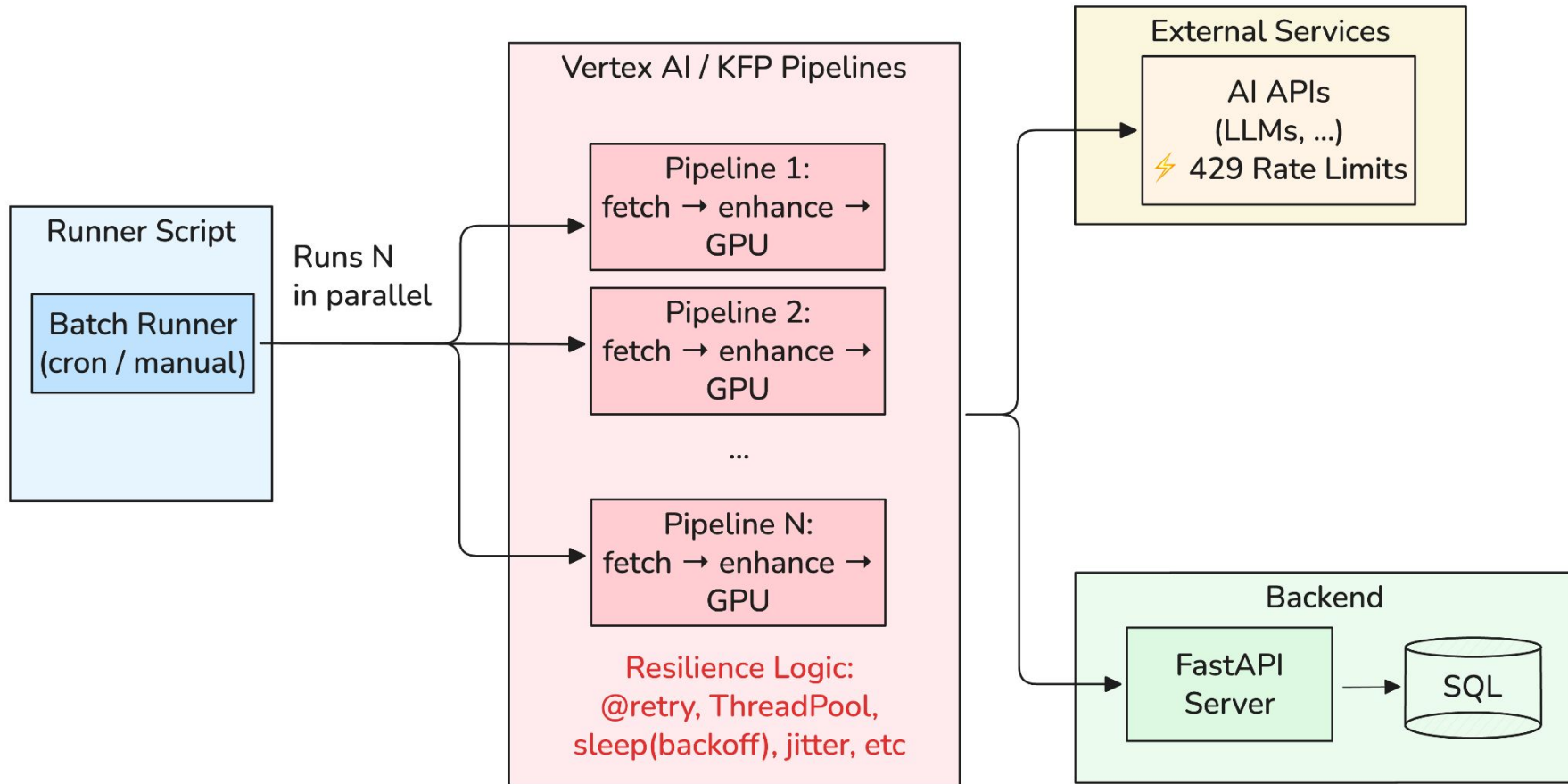


KubeCon



CloudNativeCon

Europe 2026



[1/3] The Nightmare at Scale



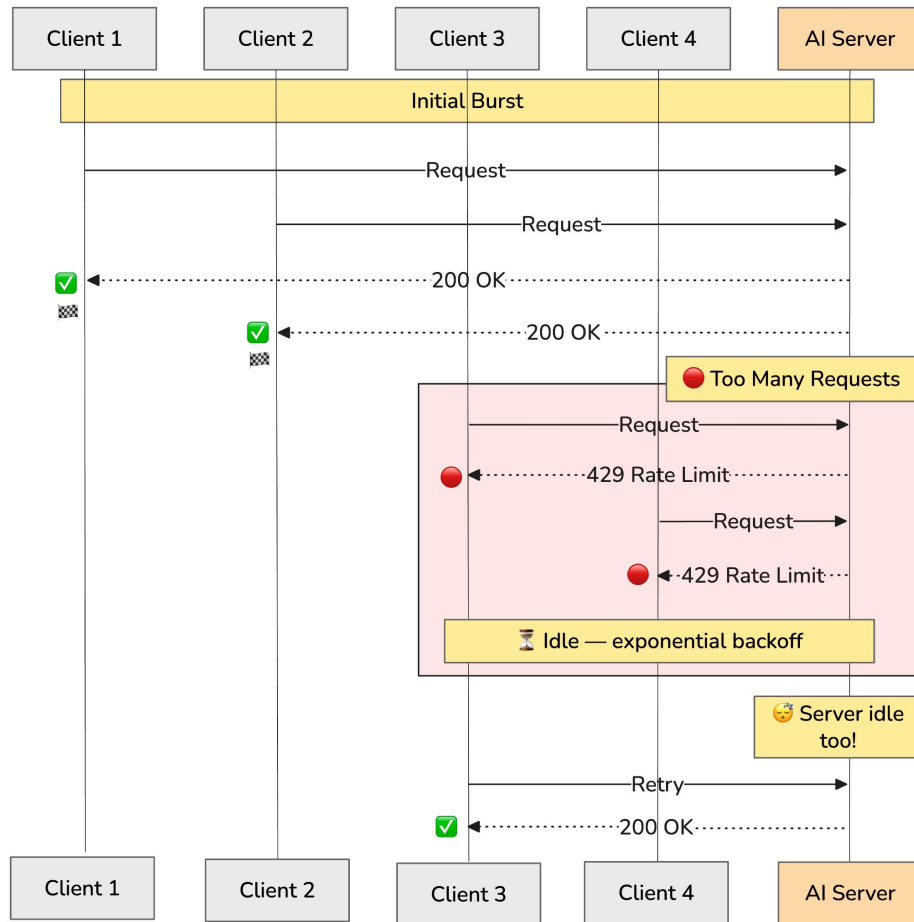
KubeCon



CloudNativeCon

Europe 2026

Exponential backoff =
everyone waits, including the
server



[2/3] The Obvious Fix

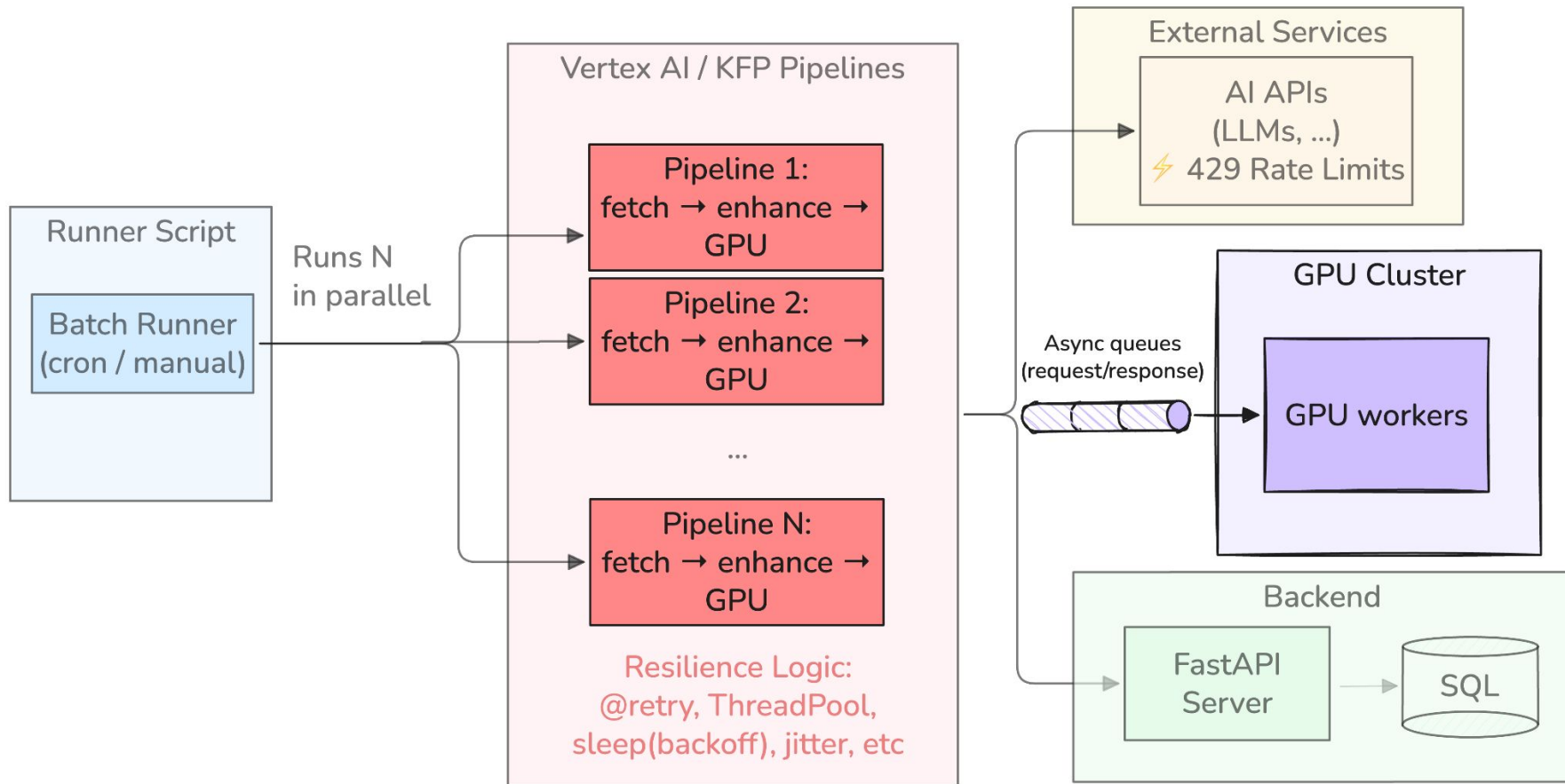


KubeCon



CloudNativeCon

Europe 2026



[2/3] The Obvious Fix



KubeCon



CloudNativeCon

Europe 2026

RESET ZOOM

1 hour

✓ 6 hours

1 day

1 week

1 month

6 weeks

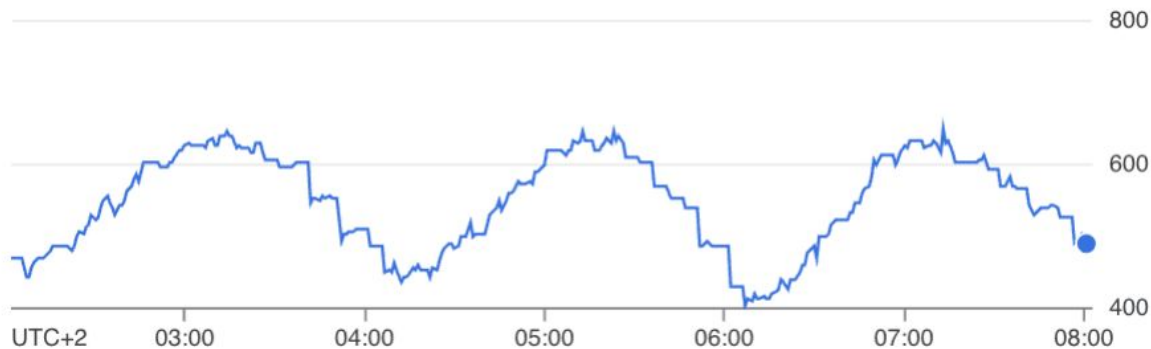
Custom



Unacked messages by region



[Create alerting policy](#)



num_unacked_messages_by_region us-central1 : 489

[3/3] Actor Mesh 🤖 — Flatten Everything!

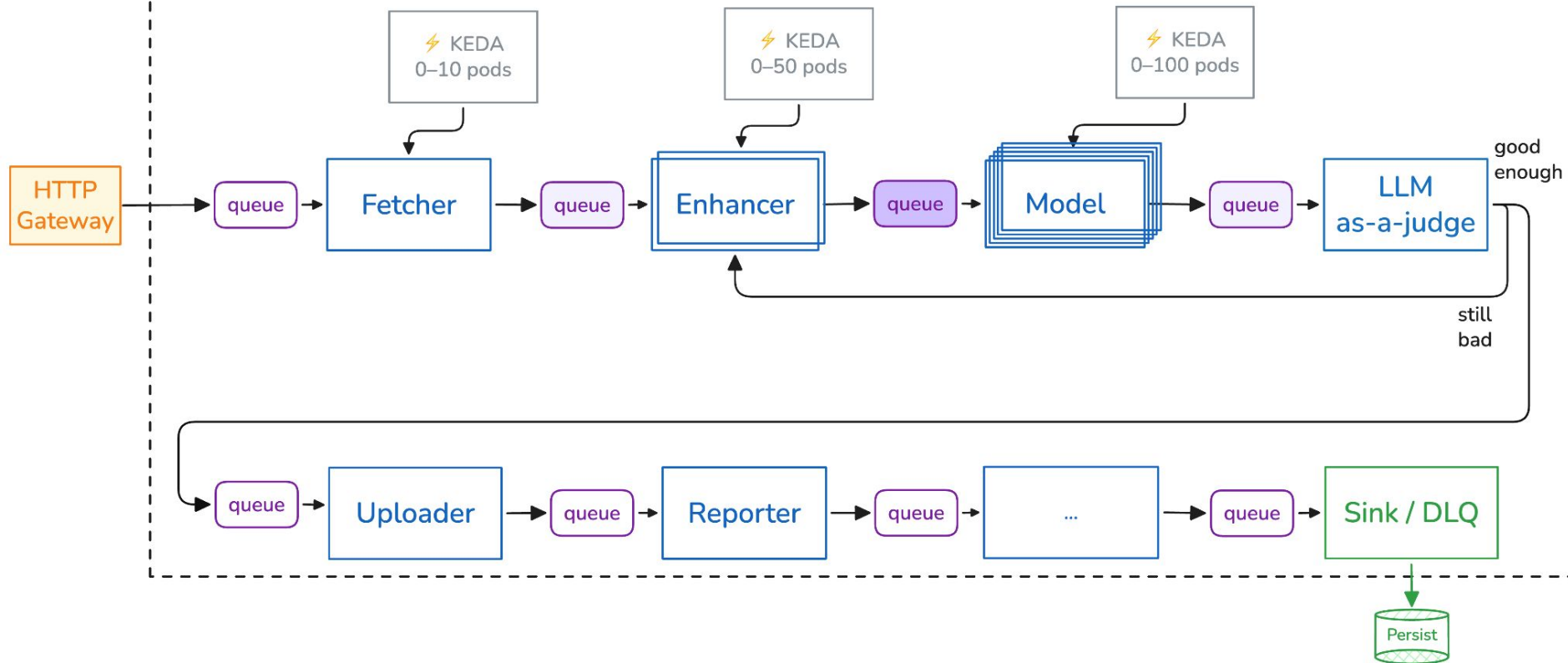


KubeCon

CloudNativeCon

Europe 2026

Actor Mesh — All Uniform, All Async



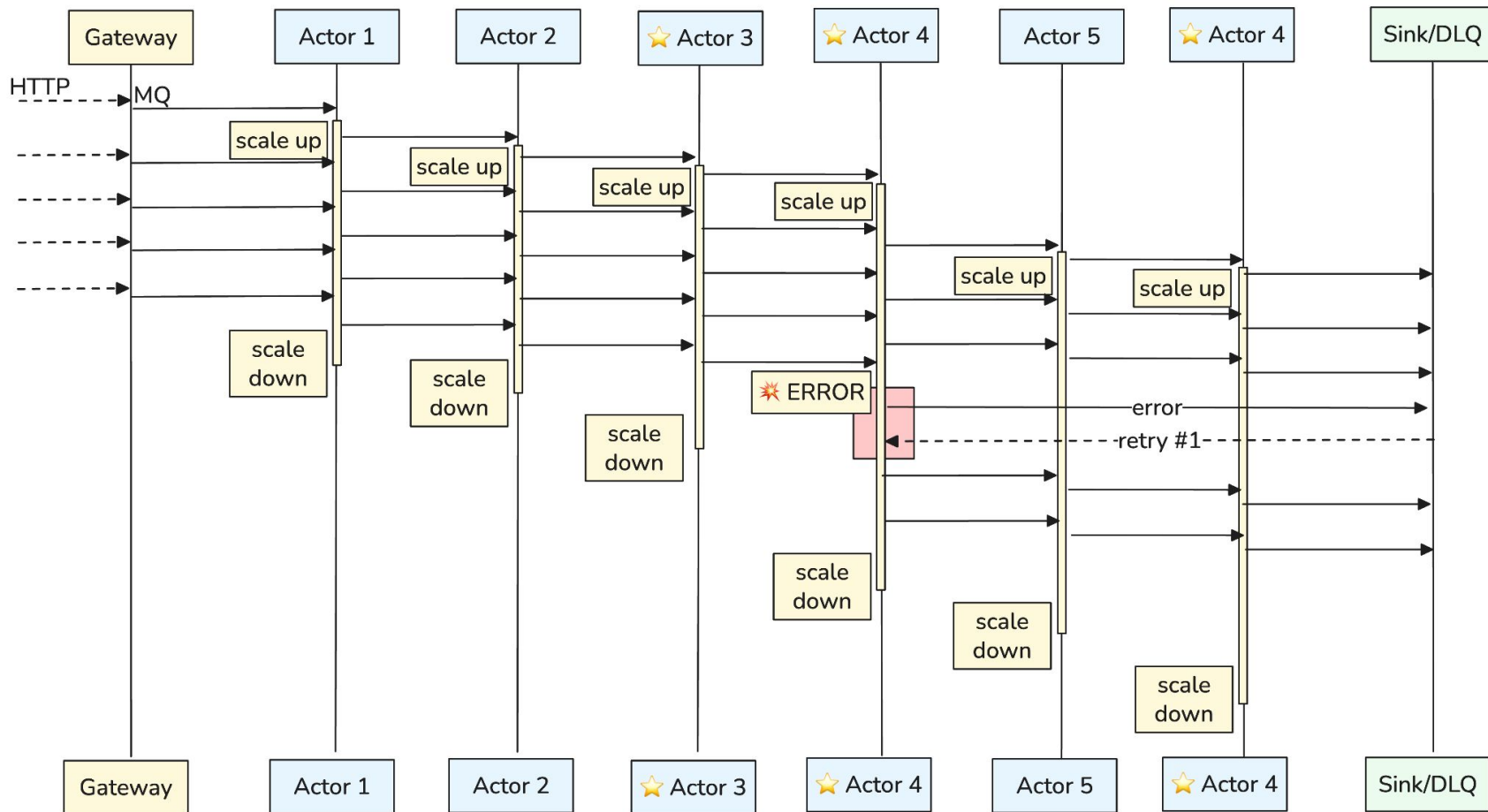
[3/3] Actor Mesh 🤖 — Flatten Everything!



KubeCon

CloudNativeCon

Europe 2026



[3/3] Actor Mesh 🤖 — In Action



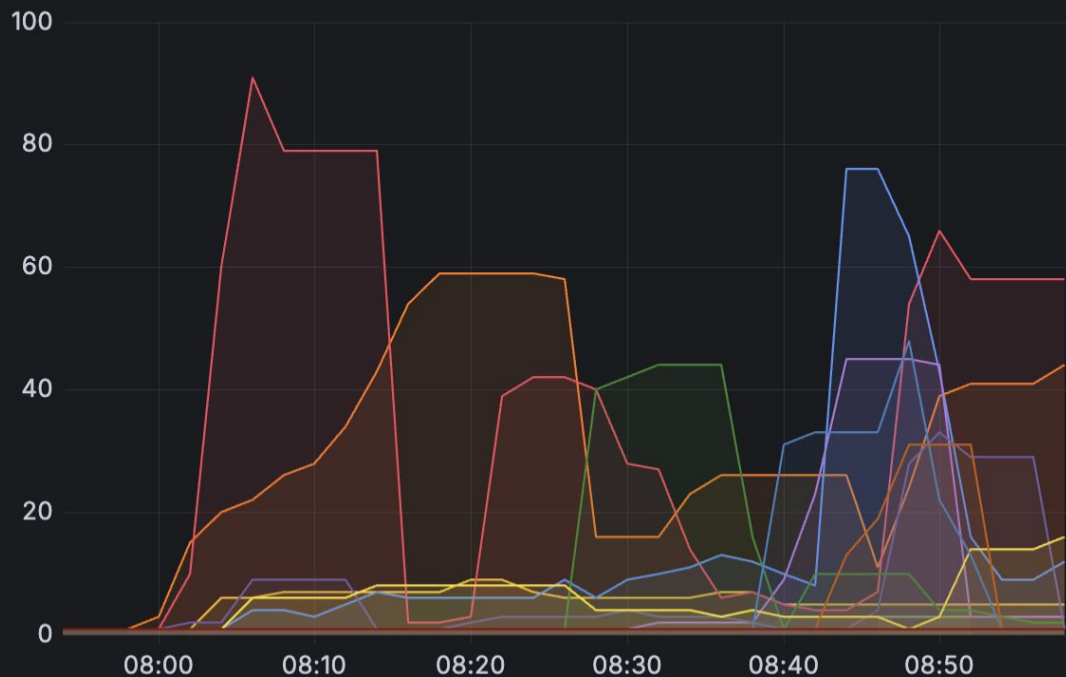
KubeCon



CloudNativeCon

Europe 2026

Runnings Pods



Name

Last *

auto-acceptor-eu	3
data-ingester-eu	5
dlq-consumer-eu	1
enhancer-pasd-upscale-eu	44
image-register-eu	58
ist-scorer-eu	12
ist-uploader-eu	3
retriever-eu	1
scorer-easyocr-watermarks-eu	2
scorer-opencv-blur-eu	16
scorer-ramp-tags-eu	1
suggestion-creator-eu	1
suggestion-undater-eu	1



Actor Pod Anatomy

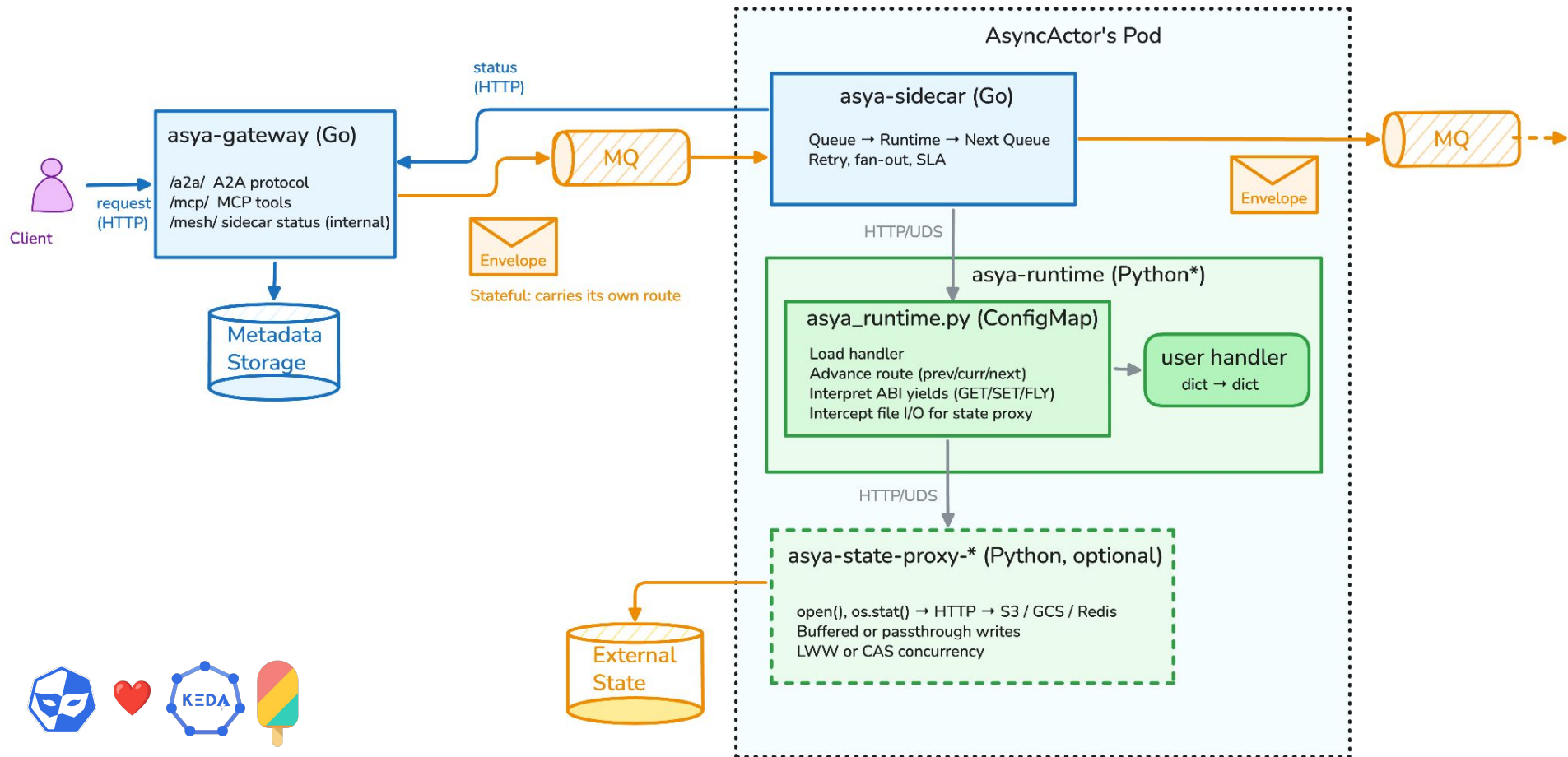


KubeCon



CloudNativeCon

Europe 2026



Two Files, Two Owners



KubeCon



CloudNativeCon

Europe 2026

```
# pip install tenacity
@retry(
    stop=stop_after_attempt(5) | stop_after_delay(300),
    wait=wait_exponential(min=1, max=60)
)
def answer_questions(payload: dict) -> dict:
    questions = [q["text"] for q in payload["questions"]]

    with ThreadPoolExecutor(max_workers=3) as pool:
        try:
            answers = list(pool.map(call_model, questions, timeout=120))
        except TimeoutError:
            logging.error("Batch processing exceeded 120s limit")
            raise
        except requests.exceptions.HTTPError as e:
            logging.error(f"HTTP error: {e.response.status_code}")
            raise

    payload["answers"] = answers
    return payload
```

Two Files, Two Owners



KubeCon



CloudNativeCon

Europe 2026

```
# pip install tenacity
@retry(
    stop=stop_after_attempt(5) | stop_after_delay(300),
    wait=wait_exponential(min=1, max=60)
)
def answer_questions(payload: dict) -> dict:
    questions = [q["text"] for q in payload["questions"]]

    with ThreadPoolExecutor(max_workers=3) as pool:
        try:
            answers = list(pool.map(call_model, questions, timeout=120))
        except TimeoutError:
            logging.error("Batch processing exceeded 120s limit")
            raise
        except requests.exceptions.HTTPError as e:
            logging.error(f"HTTP error: {e.response.status_code}")
            raise

    payload["answers"] = answers
    return payload
```

Two Files, Two Owners



KubeCon

CloudNativeCon

Europe 2026

Pure CRD

Pure function

```
def answer_questions(payload: dict) -> dict:
    questions = [
        call_model(q["text"])
        for q in payload["questions"]
    ]
    payload["answers"] = answers
    return payload
```

```
apiVersion: asya.sh/v1alpha1
kind: AsyncActor
metadata:
  name: model-caller
spec:
  actor: model-caller
  image: call-model:latest
  handler: handler.answer_questions

  scaling:
    minReplicaCount: 0
    maxReplicaCount: 3
    queueLength: 1

  resiliency:
    retry:
      policy: exponential
      maxAttempts: 5
      initialInterval: 1s
      maxInterval: 60s
      actorTimeout: 300s
```

Two Files, Two Owners



KubeCon



CloudNativeCon

Europe 2026

Actor flavors (overlays)

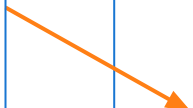
```
apiVersion: apiextensions.crossplane.io/v1beta1
kind: EnvironmentConfig
metadata:
  name: llm-resilient
  labels:
    asya.sh/flavor: llm-resilient
data:
  resiliency:
    retry:
      policy: exponential
      maxAttempts: 5
      initialInterval: 1s
      maxInterval: 60s
      jitter: true
    nonRetryableErrors:
      - "ContentPolicyViolation"
      - "InvalidAuth"
    actorTimeout: 120s
```

Pure CRD

```
apiVersion: asya.sh/v1alpha1
kind: AsyncActor
metadata:
  name: model-caller
spec:
  actor: model-caller
  image: call-model:latest
  handler: handler.answer_questions

  scaling:
    minReplicaCount: 0
    maxReplicaCount: 3
    queueLength: 1

  flavors: [llm-resilient]
```



The Message Knows the Way



KubeCon



CloudNativeCon

Europe 2026

Envelope route mutation

```
{
  "id": "a1b2c3d",
  "route": {
    "prev": [],
    "curr": "enhance",
    "next": [
      "score",
      "validate"
    ]
  },
  "headers": {
    "trace_id": "abc-123"
  },
  "payload": {
    "image_url": "s3://cat.jpg"
  }
}
```



```
{
  "id": "a1b2c3d",
  "route": {
    "prev": [
      "enhance"
    ],
    "curr": "score",
    "next": [
      "validate"
    ]
  },
  "headers": {
    "trace_id": "abc-123"
  },
  "payload": {
    "image_url": "s3://cat.jpg",
    "enhanced_img": "supercat.jpg"
  }
}
```



```
{
  "id": "a1b2c3d",
  "route": {
    "prev": [
      "enhance",
      "score"
    ],
    "curr": "validate",
    "next": []
  },
  "headers": {
    "trace_id": "abc-123"
  },
  "payload": {
    "image_url": "s3://cat.jpg",
    "enhanced_img": "supercat.jpg",
    "score": 0.75
  }
}
```

The Message Knows the Way



KubeCon



CloudNativeCon

Europe 2026

Agentic actors: stream live, route dynamically — pure Python.

```
def llm_judge(payload: dict):
    # Stream real time
    for chunk in llm.stream(payload["question"]):
        yield "FLY", {"text": chunk} # SSE event -> directly to gateway

    payload["answer"] = llm.get_full_response()

    # Route based on the result
    if needs_human_review(payload["answer"]):
        yield "SET", ".route.next", ["x-pause"] # human-in-the-loop

    yield payload # next actor
```

Python In, Actors Out



KubeCon

CloudNativeCon

Europe 2026

```
@flow
async def routing_classifier(state: dict) -> dict:
    # Classify the incoming request
    state = await classifier(state)

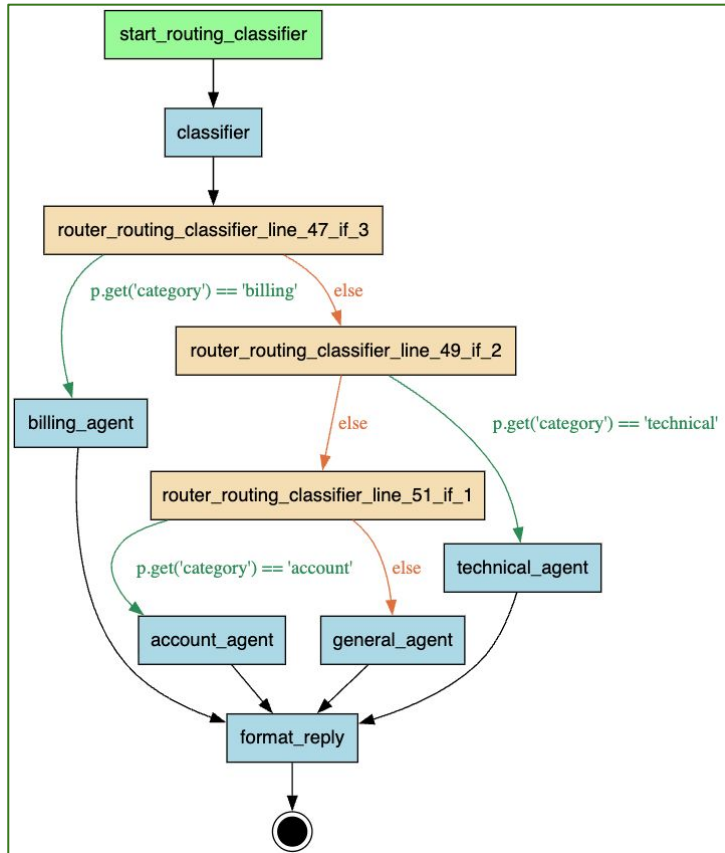
    # Route to specialized handler
    if state["category"] == "billing":
        state = await billing_agent(state)

    elif state["category"] == "technical":
        state = await technical_agent(state)

    elif state["category"] == "account":
        state = await account_agent(state)

    else:
        state = await general_agent(state)

    # Unified post-processing
    state = await format_reply(state)
    return state
```



```
$ asya compile classifier -f src/routing_classifier.py
```

Long Live the Queue!



- POST /predict and wait
- Static pre-built pipeline
- Retry/timeout/backoff in your code
- One pipeline process, one failure domain
- Framework or AI provider lock-in
- Complex abstractions
- The **message knows the way**
- Dynamic mesh — **actors write the future**
- Retry policy is the **deployment information**
- Independent actors, **independent scaling**
- **Pure Python function** + Pure K8s manifest
- Single CRD. **Composition** over abstraction

Open source • Apache 2.0 • github.com/deliveryhero/asya ★





KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon



asya.sh



Artem Yushkovskiy

Thank you!

Session feedback

