NETCELL EDA PLATFORM

Event-Driven AI Platform Events Processor

NETCELL-EDA Architecture

Events Processor

This microservice design is perfect for Cloudera

Kafka integration is native

Schema Registry fits perfectly

Ranger can enforce tenant isolation

Spark/Flink can generate offline features

Iceberg tables store historical data

Kubernetes handles scaling

AI pipeline plugs into Cloudera storage

NETCELL-EDA Architecture

Events Processor

1. Internal Modules (inside the microservice)

A. Kafka Consumer Module

Responsible for:

connecting to contacts-events topic

deserializing messages

validating schema

handling partition assignment

committing offsets

Key behaviors:

uses ContactId as key → ensures ordering

part of consumer group contact-processor-group

supports parallelism via partitions

B. Event Validator

Responsible for:

validating event structure

checking schema registry version

rejecting malformed events

ensuring required fields exist (TenantId, ContactId, etc.)

C. Tenant Router

Responsible for:

reading TenantId from event

loading tenant-specific config

selecting tenant-specific rules

selecting tenant-specific model set (if applicable)

This is where multi-tenancy becomes real.

NETCELL-EDA Architecture

Events Processor

D. Profile Loader

Responsible for:

fetching profile from Document DB

creating a new profile if none exists

merging basic fields (ContactId, AccountId, etc.)

E. Feature Builder

Responsible for:

computing real-time features

updating Online Feature Store

preparing feature vector for AI scoring

Examples:

time since last state change

number of state changes in last 30 days

login frequency

billing/security signals

F. Inference Client

Responsible for:

calling Inference Service

sending feature vector

receiving scores (risk, churn, engagement)

handling model versioning

G. Profile Updater

Responsible for:

updating profile fields:

- State
- StateHistory
- Scores
- LastEventAt

applying tenant-specific rules

preparing final document for upsert

NETCELL-EDA Architecture

Events Processor

H. Document DB Writer

Responsible for:

upserting updated profile

ensuring atomicity

handling retries

I. Metrics & Logging

Responsible for:

Prometheus metrics

structured logs

tracing (TraceId)

error reporting

NETCELL-EDA Architecture

Events Processor

2. Internal Flow (inside the microservice)

Here's the exact flow for a ContactStateChanged event:

Code

KafkaConsumer.poll()

↓

EventValidator.validate(event)

↓

TenantRouter.loadConfig(event.TenantId)

↓

ProfileLoader.load(event.TenantId, event.ContactId)

↓

FeatureBuilder.build(profile, event)

↓

InferenceClient.score(features)

↓

ProfileUpdater.update(profile, event, scores)

↓

DocumentDbWriter.upsert(updatedProfile)

↓

Metrics.logSuccess()

This is the real backbone of event-driven architecture.

NETCELL-EDA Architecture

Events Processor

3. Internal Data Structures

Event object (after validation)

json


```
{
  "TenantId": "tenantA",
  "ContactId": 12345,
  "AccountId": 6789,
  "OldState": "New",
  "NewState": "Active",
  "Reason": "First login",
  "OccurredAt": "2026-08-04T19:40:00Z"
}
```


Feature vector

json


```
{
  "state_change_count_last_30_days": 4,
  "time_since_last_state_change_hours": 2.5,
  "total_logins_last_7_days": 12,
  "security_incidents_last_90_days": 1
}
```


Scores from AI

json


```
{
  "Risk": 0.18,
  "Churn": 0.07,
  "Engagement": 0.82
}
```


Updated profile

json


```
{
  "TenantId": "tenantA",
  "ContactId": 12345,
  "State": "Active",
  "Scores": {
    "Risk": 0.18,
    "Churn": 0.07,
    "Engagement": 0.82
  },
  "LastEventAt": "2026-08-04T19:40:00Z"
}
```

NETCELL-EDA Architecture

Events Processor

4. Deployment Behavior on Kubernetes

The processor runs as:

4 replicas

part of **one consumer group**

Kafka partitions distributed across pods

automatic scaling based on:

- CPU
- event backlog
- latency

If traffic spikes:

add more replicas

Kafka rebalances partitions

throughput increases linearly

This is how achieve enterprise-grade scalability.

NETCELL-EDA Architecture

Events Processor

Thank you