EVENFLOW

Robust Learning and Reasoning for Complex Event Forecasting

EVENFLOW White Paper

On Scalable Neurosymbolic Streaming Analytics

www.evenflow-project.eu

WHITE PAPER

On Scalable Neurosymbolic Streaming Analytics

Date:	22 October 2025
Authors:	Nikos Giatrakos

Executive Summary

Organizations operating at the intersection of Big streaming Data and intelligent decision-making face a triad of challenges: extremely high data stream ingestion rates, complex analytics (neural and symbolic), and strict end-to-end latency/service-level objectives. This white paper proposes a unified architecture that composes three complementary systems: **SDEaaS (Synopses Data Engine-as-a-Service)**, **SuBiTO (Synopsis-based Training Optimization)**, and **NeuroFlinkCEP** to deliver scalable, adaptive, and interpretable streaming intelligence across the cloud-to-edge continuum. It achieves that, by integrating the virtues of data stream summaries, data-parallel training and geo-distributed neurosymbolic (NeSy) analytics.

- **SDEaaS** provides a synopsis-as-a-service layer that maintains and serves approximate summaries (sketches, samples, quantile digests, etc.) for thousands of streams. This decouples downstream analytics from raw-rate ingestion, enabling low-latency, resource-efficient neural or neurosymbolic model updates.
- **SuBiTO** is the control plane for continuous, scalable learning. It tracks accuracy vs latency trade-offs in real time, detects concept drifts, and dynamically tunes neural and NeSy training pipelines, also configuring how aggressively to draw from synopses and the parallelism of the training pipeline, so that models keep pace with the stream while respecting time and resource budgets.
- **NeuroFlinkCEP** composes *neural* detectors (primitive events) with *symbolic* complex event recognition (CER) patterns. It compiles patterns to streaming jobs and optimizes operator placement across edge and cloud resources.

Together, these components form a feedback-rich, synopsis-aware NeSy stack. The result is a platform that scales to massive input rates, preserves expressivity (incorporating knowledge in symbolic rules), adapts to drift, and optimizes application accuracy, compute, memory, and bandwidth consumption, while maintaining human oversight.

Audience and Reading Map

- **Executives & product owners (5–7 min):** *Executive Summary*, *Business Value & TCO* (Section 9), *Mini Case Study* (Section 9.1).
- **Architects & platform leads (15–25 min):** *Components* (Section 2), *Integration Blueprint* (Section 3), *MVP Engineering* (Section 5), *Security/Privacy/Governance* (Section 7.1), *Operational Model & SLAs* (Section 7.2).
- **Data scientists & ML engineers (15–20 min):** *SuBiTO* (Section 2.2), *KPIs/Benchmarks* (Section 6), *Roadmap* (Section 8).
- **Ops/SRE (10–15 min):** *MVP Engineering* (Section 5), *Operational Model & SLAs* (Section 7.2), Appendix B – *FAQ*.

About the Author

Nikos Giatrakos is an Assistant Professor at the School of Electrical and Computer Engineering of the Technical University of Crete (Greece). His research interests are in the broad area of Big Data Management algorithms, software architectures and systems including Big streaming Data & Real-Time Analytics, Distributed/Decentralized Big Data Processing, Federated Machine Learning, Edge-to-Cloud Big Data Management, Synopses for Massive Data/Approximate Query Processing, Complex Event Processing. He was a recipient of the Best Demo Award in ACM CIKM 2020. He has contributed as one of key investigators in several recent EU projects, he has served as the co-coordinator of the EU H2020 project INFORE and as a Principal Investigator for Athena Research Center at the EU Horizon project EVENFLOW.

Table of Contents

Table of Figures

Definitions, Acronyms and Abbreviations

Acronym/ Abbreviation	Title
CE	Complex Event
CEP	Complex Event Processing
CER	Complex Event Recognition
DAG	Directed Acyclic Graph
DS	Data Science/Scientist
FTE	Full Time Equivalent
HLL	HyperLogLog Sketch
MTTR	Mean Time to Response
NeSy	Neurosymbolic
ONNX	Open Neural Network Exchange
p95/p99	95th/99th percentile
PII	Personally Identifiable Information
QPS	Queries Per Second
SDEaaS	Synopses Data Engine as a Service
SE	Simple Events
SLA/SLO	Service-Level Agreement / Objective
SRE	Site Reliability Engineer / Engineering
SuBiTO	Synopsis-Based Training Optimization

Term	Definition
Synopsis	A compact streaming summary (e.g., sketch, sample) that enables fast, approximate queries with bounded error.
Quantile (t-digest)	Data structure that estimates percentiles (e.g., p95 latency) accurately on streams with limited memory.
Heavy hitter (HH)	An item (e.g., station transition, endpoint) that occurs with high frequency; tracked via HH sketches.
Cardinality (HLL)	Estimated count of distinct elements in a stream using HyperLogLog.
Selection strategy	CEP policy for which overlapping matches are eligible (e.g., strict vs. relaxed contiguity).
Consumption policy	CEP policy for how matched events are consumed
Watermark	Event-time progress indicator used to handle out-of-order data and close windows safely.
Guardrail	Enforced constraint (latency, accuracy, privacy, cost) that triggers block/rollback when violated.
Service Level Objective (SLO)	Internal target for a reliability/quality metric (e.g., p95 ≤ 1.5 s).
Error Budget	Allowed deviation from an SLO (e.g., % of time above latency target) used for go/no-go decisions.

Term	Definition
Burn-rate	Speed at which the error budget is being spent; primary signal for escalation.
Canary deployment	Staged rollout of a new model/pattern to a small slice before full promotion.
Rollback	Automatic/manual reversion to a previous model/pattern when KPIs regress.
Model freshness window	Maximum allowable age for a deployed model before retrain/refresh is required.
Concept/covariate/prior drift	Changes in relationship, features, or class priors that degrade model performance.
Load sweep	A planned test where you systematically vary input rate and/or concurrency to map system behaviour across loads.
Surge profile	Predefined, temporary operating mode for spikes that prioritizes SLOs. In the context of this white paper this means downshifting models, increasing synopsis compression/ sampling and adjusting placement/resources until load normalizes.

1 Motivation and Problem Setting

1.1 The Real-Time Intelligence Gap

Conventional stream processing excels at low-level aggregations but struggles to:

- Continuously train and update neural models on data in motion while meeting real-time SLAs.
- Detect complex, temporally extended situations from heterogeneous sources using both soft predictions (neural) and hard constraints (symbolic/temporal patterns).
- Scale to thousands of streams, thousands of concurrent analytics across numerous network nodes without prohibitive resource costs.

1.2 Why Combining Synopses & Data Parallel Learning & NeSy CER

- **Synopses** reduce the memory footprint from raw-rate to compact representations with quantifiable error bounds, enabling downstream tasks to query *representative* information at *low* latency. They often also reduce the computational complexity of the problem at hand, speeding up analytics tasks.
- **Data Parallel Training** ensures neural components keep up with concept drift and workload variation by tuning neural model topology depth and types of layers, epochs, sampling/compression via synopses, and synchronization schedules in the context of federated, data parallel learning settings.
- **Neurosymbolic CER** merges the pattern-level interpretability of symbolic rules with the perceptual acuity of neural models, offering accuracy and explainability during critical decision making procedures.

2 Component Overviews

2.1 SDEaaS: Synopsis-as-a-Service for Streaming Analytics

Role. SDEaaS is a long-running, horizontally (with the volume and velocity), vertically (with the number of streams) and federated (with the number of network nodes) scalable engine that maintains many synopses per stream and exposes them via APIs for *reuse* across multiple analytics. Rather than recomputing summaries per consumer, synopses are shared, versioned and queried as first-class citizens.

Key capabilities. ► **Breadth of synopses.** Count–Min sketches, HyperLogLog, sampling, quantile digests (e.g., t-digest), heavy-hitter summaries, Bloom/Counting Bloom structures, sliding-window variants, and multi-resolution hierarchies. ► **Concurrency at scale.** Thousands of synopses across thousands of streams; multi-tenant isolation; per-synopsis update policies (event-time windows, tumbling/sliding, watermark-aware). ► **Reuse and composition.** Downstream operators (training, CEP, dashboards) query the same synopsis instances; synopses can be *composed* (union/merge for federated views) or *refined*. ► **Hot-swap and evolution.** Synopsis definitions can be registered/ unregistered at runtime; parameters (e.g., sample size) are adjustable. ► **Placement & federation.** Deployed in cluster(s) with optional edge caches; supports aggregation of partial synopses for geo-distributed analytics.

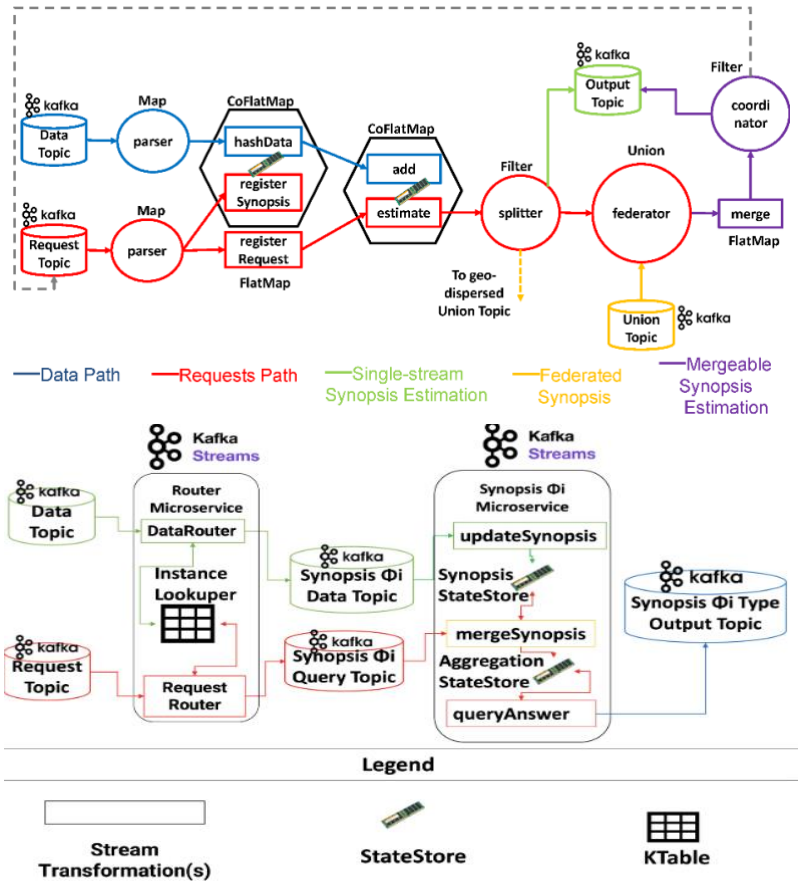


Figure 1:SDEaaS Internal Architecture on Apache Flink (up) and Kafka Streams (down)

Interfaces. The Synopses Data Engine (SDE) consumes JSON formatted **Data Streams** and **Requests** from separate Kafka topics. Requests are JSON messages with a *requestID* and synopsis parameters. Operations:

► (1) *ADD* — add a synopsis ► (2) *DELETE* — delete a synopsis ► (3) *ESTIMATE* — query a synopsis ► (4) *ADD* (random partitioning) ► (5) *ADD* (continuous) ► (7) *UPDATE* — update synopsis state

For facile integration into Python ecosystems, including Tensorflow and PyTorch training pipelines, the SDE has also been implemented on top of Apache Dask.

2.2 SuBiTO: Synopsis-Based Training Optimization

Role. SuBiTO is the *learning control plane* for scalable continuous training over streaming data. It monitors latency/throughput/accuracy, detects drift, and interactively steers the training pipeline by tuning: (i) **model complexity** (types and number of layers), (ii) **stream ingestion** (sampling/compression via synopses, class-balanced draws, window sizes), (iii) **training schedule** (epochs, mini-batch size, parallelism), and (iv) **deployment knobs** (smart synchronization protocols for data parallel learning, mini-batching for high-throughput predictions).

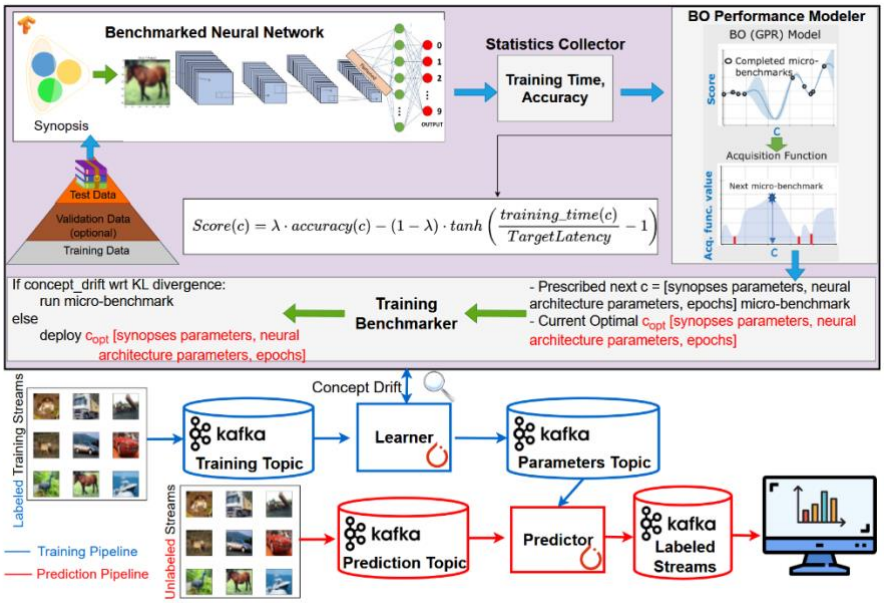


Figure 2: SuBiTO Architecture

SuBiTO Optimizer. Observes the current status of ingested synopses and of the training pipeline → Detects concept drifts → Explores/Exploits alternatives to optimize synopses parameters, neural model architectures, epoch durations and parallelism across the training pipeline → Prescribes a properly configured model for the training pipeline → Keeps monitoring.

SuBiTO Training Pipeline. Continuously receives the prescribed model from the SuBiTO Optimizer and deploys it to sample and train in a data parallel/federated fashion on labeled streams.

SuBiTO Prediction Pipeline. Continuously receives the most up-to-date trained neural model from the Training Pipeline and uses it to tag unlabeled data streams across parallel predictors.

Human-in-the-loop. Dashboards expose Pareto fronts (accuracy vs execution time), with manual override when human operators prefer a particular configuration (e.g., during imminent application incidents).

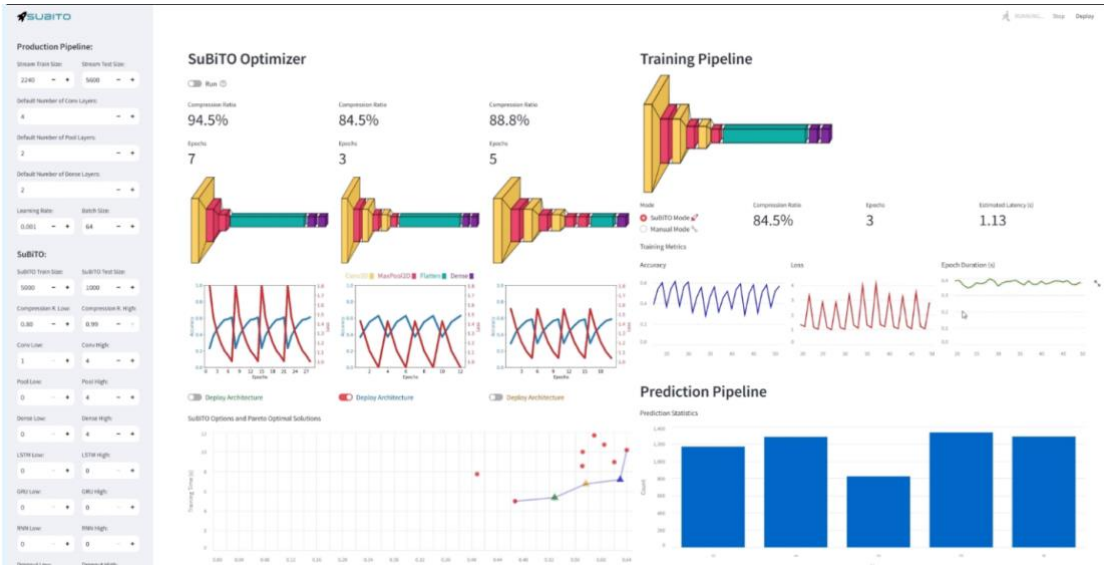


Figure 3: SuBiTO Dashboard

2.3 NeuroFlinkCEP — NeSy Complex Event Recognition across the Continuum

Role. NeuroFlinkCEP compiles extended regular expressions/temporal patterns into streaming jobs that combine: **neural primitives** (SynapSEflow nested operator) producing primitive events by dynamically learning from perceptual data instead of relying on crisp definitions; and **symbolic operators** (FlinkCEP nested operator) that match complex events.

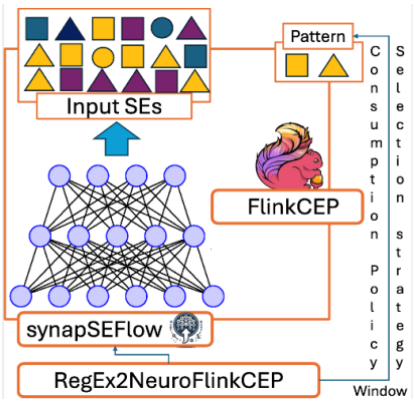


Figure 4: Internal Structure of a NeuroFlinkCEP Operator

The DAG*4CER optimizer, tailored for CEP workflows composed of connected NeuroFlinkCEP operators, is used to optimally place operators across cloud-to-edge devices under latency, bandwidth, and device constraints.

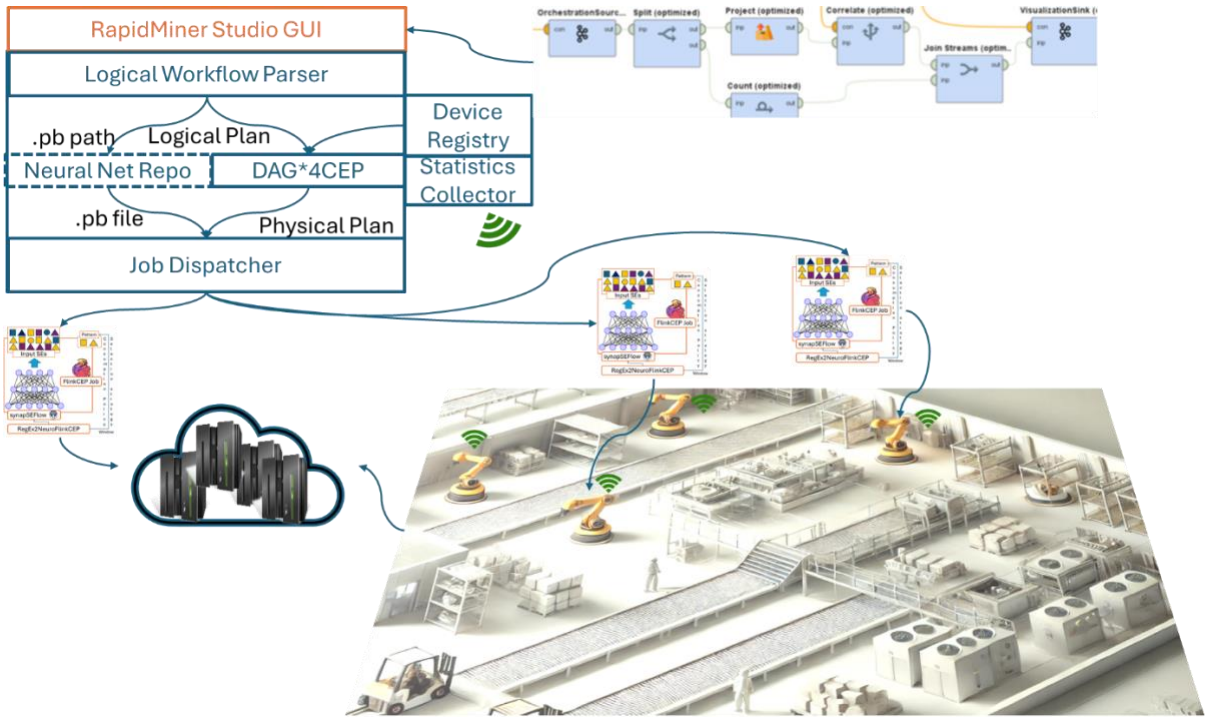


Figure 5: NeuroFlinkCEP for CER Workflow Optimization

Simple vs Complex Events and Patterns (robotic reference scenario).

► **Simple Events (SEs):** A (collision), B (stopped/unknown), C–L (moving to station 0–9), M–V (stopped at station 0–9).

► **Complex Events (CEs) – Pattern examples:**

1) CollisionRecovery

```
PATTERN CollisionRecovery := A B [C-L] WITHIN 2u
SELECTION_STRATEGY = STRICT_CONTIGUITY
CONSUMPTION_POLICY = SKIP_PAST_LAST
```

Meaning: after a collision and a stop, the robot resumes movement towards a station within 2 time units.

2) StationSkipping

```
PATTERN StationSkipping := ([C-L])\1*(?!\1)(?![M-V])([C-L]) WITHIN 1
0u
SELECTION_STRATEGY = RELAXED_CONTIGUITY
CONSUMPTION_POLICY = NO_SKIP
```

Meaning: the robot moves towards a station and, without stopping to that target station heads to a different station.

3) **ProlongedStop**

```
PATTERN ProlongedStop := ([M-V](?![C-L])[A-BM-V])* WITHIN 5u
SELECTION_STRATEGY = STRICT_CONTIGUITY
CONSUMPTION_POLICY = SKIP_TO_SPECIFIC
```

Meaning: the robot remains at a station (no movement events) for a prolonged period.

4) SuccessfulStop

```
PATTERN SuccessfulStop := (?:[C-L]+[^AB]*[M-V])+ WITHIN 5u
SELECTION_STRATEGY = STRICT_CONTIGUITY
CONSUMPTION_POLICY = SKIP_PAST_LAST
```

Meaning: one or more movement sequences that end with a valid stop at a station.

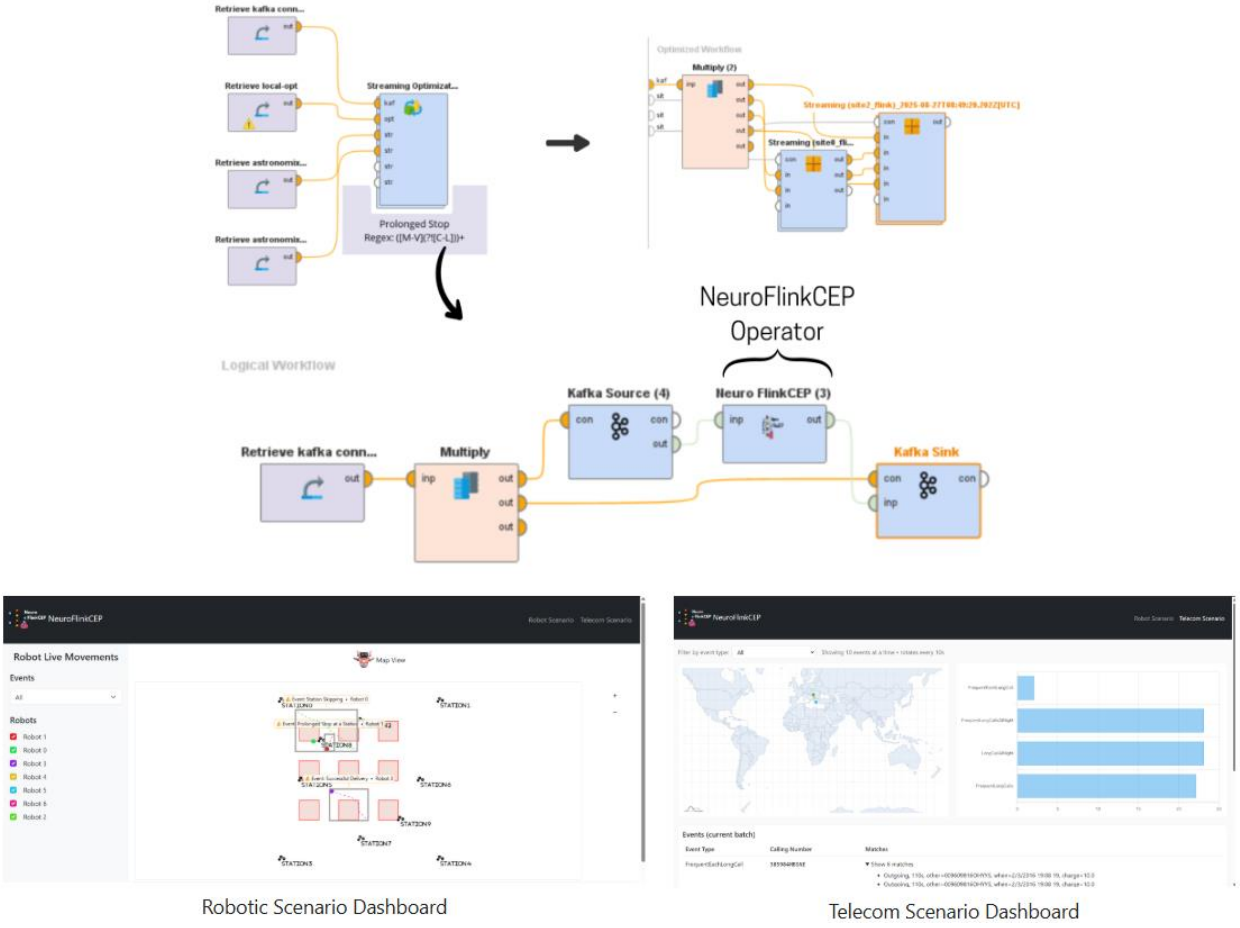
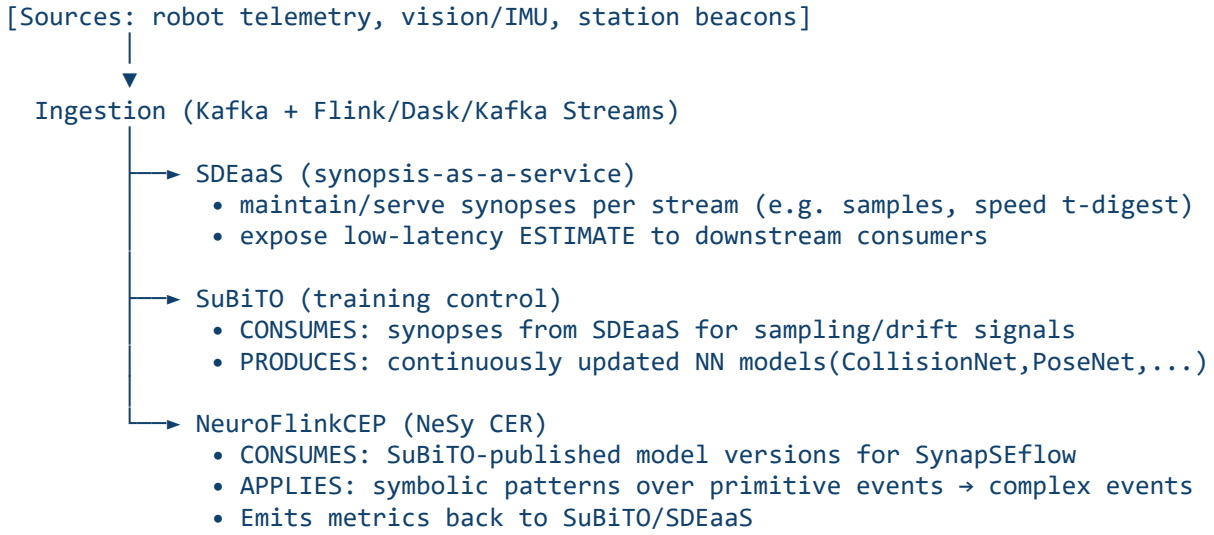


Figure 6: NeuroFlinkCEP in Action

3 An Integration Blueprint

This section instantiates the proposed architecture for a **robotic fleet** scenario (SEs A,B,C–L,M–V; CEs CollisionRecovery, StationSkipping, ProlongedStop, SuccessfulStop) and shows how SDEaaS, SuBiTO, and NeuroFlinkCEP would interact with concrete request/response examples.

3.1 Reference Architecture



Control/Feedback Plane (extended): ► SDEaaS → SuBiTO: synopsis and cost to inform training pipeline. SuBiTO → NeuroFlinkCEP: up-to-date models ► NeuroFlinkCEP → SuBiTO: detection quality, latency, accuracy ► SuBiTO → SDEaaS: required synopsis precision/retention to meet accuracy/latency targets.

3.2 Primary Data and Data Flows

2) Register synopses in SDE (once per fleet or per zone)

- *speed quantiles per robot* (10-min sliding window):

```
{"streamID":"robot-telemetry","synopsisID":101,"requestID":1,  "dataSetkey":"robots","param":["robot_id","speed","TDIGEST","600s","δ=0.01"],
 "noOfP":5,"uid":5001}
```

- *station transition heavy hitters* (from→to, 15-min sliding window):

```
{"streamID":"robot-telemetry","synopsisID":102,"requestID":1,  "dataSetkey":"robots","param":["from_station","to_station","HH","900s","w=10","d=5"],
 "noOfP":6,"uid":5002}
```

- *collision counts per zone* (1-min tumbling):

```
{"streamID":"robot-telemetry","synopsisID":103,"requestID":1,
 "dataSetkey":"robots","param":["zone_id","is_collision","COUNT","60s"],
 "noOfP":4,"uid":5003}
```

- *image reservoir for training refresh* (per robot):

```
{"streamID":"robot-vision","synopsisID":104,"requestID":1,
 "dataSetkey":"robots","param":["robot_id","frame","RESERVOIR","k=500"],
 "noOfP":4,"uid":5004}
```

2) Adaptive and Parallel training on synopses (SuBiTO)

Samples from parallel synopses, monitors drift, and continuously trains/fine-tunes models and training configurations. Publishes versioned models and budgets.

3) Neural primitive detection (NeuroFlinkCEP)

CollisionNet(vision+IMU) emits **A**; PoseNet/BeaconNet emit **C–L** (movement to station) and **M–V** (stopped at station); a watchdog emits **B** (stopped/unknown) when pose is uncertain.

4) CEP complex event matching (NeuroFlinkCEP)

- *CollisionRecovery:* `A B [C-L] WITHIN 2u` → CE `{robot_id, from_station, to_station, recovery_time}`
- *StationSkipping:* `([C-L])\1*(?!\1)(?![M-V])([C-L]) WITHIN 10u` → CE `{robot_id, from_station, to_station}`
- *ProlongedStop:* `([M-V](?![C-L])[A-BM-V])* WITHIN 5u` → CE `{robot_id, station_id, dwell_time}`

5) CEP could consult SDEaaS at runtime (pre-prune & enrich)

- Heavy hitters to guide StationSkipping search:

```
{"synopsisID":102,"requestID":3,
"param":["topk","k=20","window=now-15m..now"]}
```

Example response:

```
{"synopsisID":102,"result":[
  {"from":"S3","to":"S7","freq":0.18},
  {"from":"S2","to":"S5","freq":0.12},
  {"from":"S7","to":"S8","freq":0.09}
]}
```

- Quantile call to parameterize ProlongedStop threshold (near-zero speed):

```
{"synopsisID":101,"requestID":6,
 "param":["quantile","q=0.95","robot_id=R042","window=now-10m..now"]
}
```

6) Feedback & adaptation (SuBiTO control loop)

If CollisionRecovery recall drops, SuBiTO increases image_reservoir *k* and lowers t-digest δ (finer quantiles), schedules a *partial fine-tune* of CollisionNet on the last 2h reservoir frames, adjusts edge inference batch size, and temporarily raises the CEP timeout for `WITHIN 2u` by +0.5u.

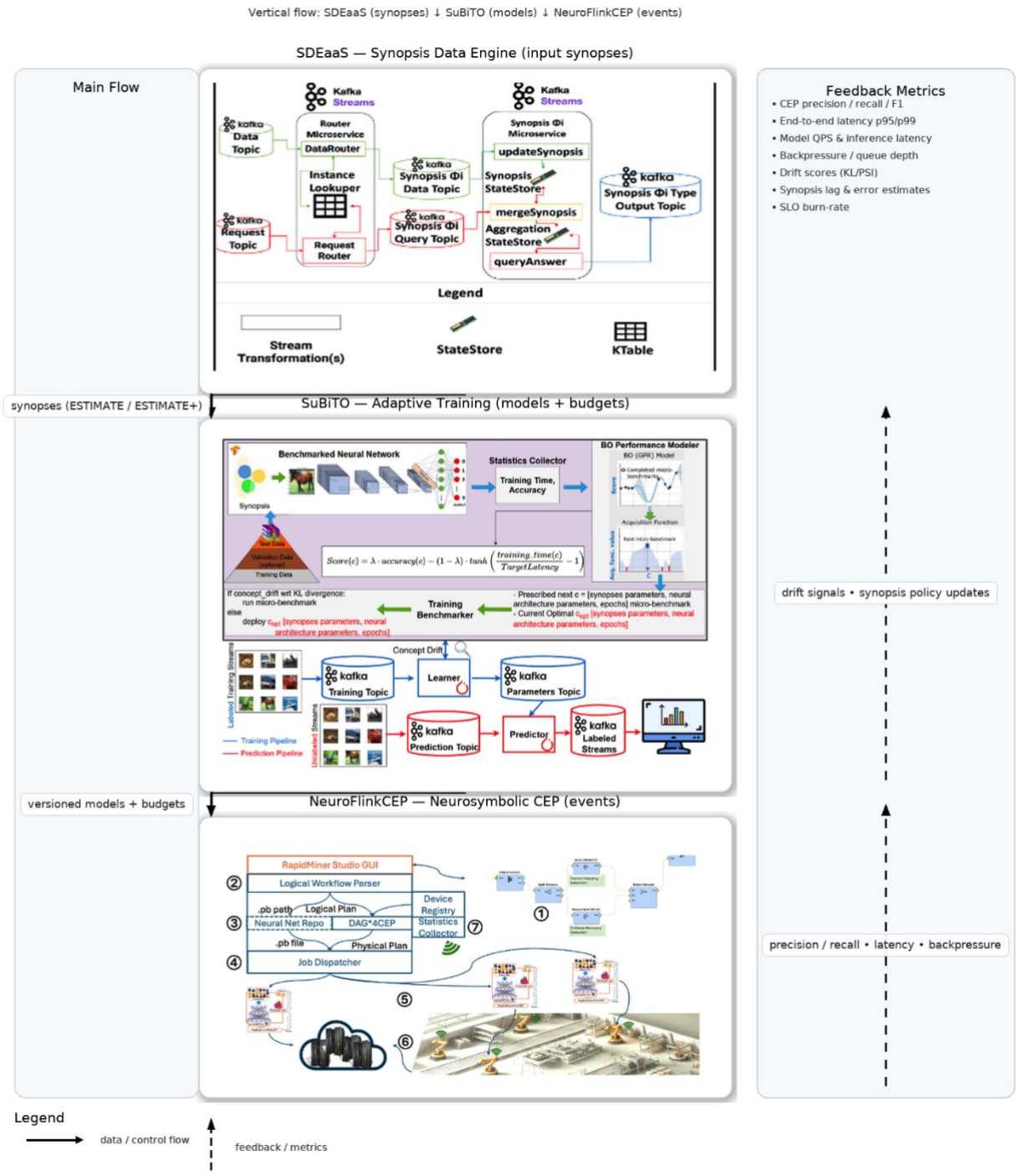


Figure 7: Streaming Intelligence - Architectural Reference Blueprint

3.3 How to Create Interfaces Among Components

- **SDEaaS ⇄ NeuroFlinkCEP**

  Query heavy hitters to bias StationSkipping search:

  ```
  {"synopsisID":102,"requestID":3,
   "param":["topk","k=10","window=now-10m..now"]}
  ```

  Quantile call for ProlongedStop speed threshold:

```
{"synopsisID":101,"requestID":6,
 "param":["quantile","q=0.90","robot_id=R021"]}
```

- **SDEaaS ⇄ SuBiTO**
  SamplerSpec drawing class-balanced collision/non-collision from the image reservoir:

```
{"sampler":"balanced","sources":[{"synopsisID":104,"label":"collisio
n"},
                                 {"synopsisID":104,"label":"non_
collision"}],
 "target_per_class":800,"max_latency_ms":5000}
```

- **SuBiTO ⇄ NeuroFlinkCEP**
  Model registry event (new CollisionNet version + inference budget):

```
{"model":"CollisionNet","version":"1.4.2",
 "device_hint":"edge","max_qps":120,
 "batch_size":4,"sla_ms":80}
```

4 Target Use Cases

4.1 Telecom Call/Event Monitoring

- **SEs:** A (premium), B (night), C (long), E (aggregate ≥ 60 min/day).
- **CEs:** `AB?C`, `(AB?C){3}`, `(AE){5}`, `(AC){5}` with the stated selection/consumption policies.
- **Neural primitives:** premium-destination classifier, caller-behavior anomaly model, duration estimator.
- **Synopsis leverage:** call samples, SDE heavy hitters (caller/callee), quantile digests for durations, approximate join cardinalities for pattern pre-pruning.
- **Adaptivity:** drift in calling behavior triggers SuBiTO to adjust sampling windows and refresh models with class-balanced synopsis samplers.

4.2 Robotic Fleet Coordination

- **SEs:** A (collision), B (stopped/unknown), C–L (move to station 0–9), M–V (stopped at station 0–9).
- **CEs:** CollisionRecovery `A B [C-L]`; StationSkipping `([C-L])\1*(?!\1)(?![M-V])([C-L])`; ProlongedStop `([M-V](?![C-L])[A-BM-V])*`; SuccessfulStop `(?:[C-L]+[^AB]*[M-V])+`.
- **Neural primitives:** collision detection (IMU/vision), station/pose recognition, motion intent classification.
- **Synopsis leverage:** position and vision samples, windowed quantiles of speed, heavy hitters for station transitions, synopsis-backed sampling for replay/debug.
- **Adaptivity:** SuBiTO tightens/loosens synopsis compression based on robot swarm load and link quality; placement favours edge for primitives, central for high-fan-in patterns.

4.3 Industrial IoT Quality & Safety

- **Inputs:** multivariate sensor streams (temperature, vibration, pressure), inspection images.
- **Neural primitives:** anomaly detectors, defect classifiers.
- **Symbolic patterns:** co-occurrence of temperature spikes with vibration change and component-state flags.
- **Synopsis leverage:** stratified reservoir samples per machine state; sliding quantile estimates for thresholds.
- **Adaptivity:** during maintenance windows, SuBiTO relaxes latency constraints to train deeper models with larger epochs and finer synopsis resolution.

4.4 Flood Response with UAV/UUV and Sensors

- **Inputs:** drone video frames, radar rain rates, river gauges, LiDAR tiles.
- **Neural primitives:** debris classifiers, water/road segmentation, shoreline change detectors.

- **Symbolic patterns:** sequences linking rising river levels, heavy rain, and clustered debris detections within spatiotemporal bounds.
- **Synopsis leverage:** video/frame/sensor samples, t-digests for level quantiles per gauge; sketches for debris-class frequencies per tile; multi-resolution synopses to zoom in during alarms.
- **Adaptivity:** when storms intensify (input surge), SuBiTO increases synopsis compression and reduces model depth/epochs to preserve end-to-end latency; then restores richer intake post-surge.

4.5 Infrastructure Life-Cycle Assessment

- **Inputs:** bill of materials, logistics telemetry, site sensors (energy/water), construction schedule, satellite tiles.
- **SEs/CEs:** SEs MaterialArrival, EnergySpike, WaterUse; CEs FootprintExceeds Budget, AnomalousMaterialRoute, StormRiskImpact.
- **SDEaaS:** rolling aggregates and t-digests per phase; heavy hitters for supplier routes; reservoir samples for audit.
- **SuBiTO:** forecasters for phase-level footprint; retunes with seasonal shifts and schedule slips.
- **NeuroFlinkCEP:** thresholds + forecasts → compliance events with provenance.

4.6 Maritime Domain

- **Inputs:** AIS tracks, coastal radar, weather/sea-state, port calls.
- **SEs/CEs:** SEs CourseChange, SpeedDrop, ZoneEntry; CEs LoiteringNearRestricted Area, DarkVesselPattern, CollisionRisk.
- **SDEaaS:** route heavy hitters (port→port); speed quantiles per vessel class; gap-duration histograms.
- **SuBiTO:** adapts anomaly nets to seasonal routes and weather; learns priors for false AIS gaps.
- **NeuroFlinkCEP:** zone & CPA rules; AIS radar correlation.

4.7 Content Moderation at Scale (SuBiTO-centric)

- **Inputs:** text/video/image streams, user reports, creator metadata.
- **SEs/CEs:** SEs ToxicityScore, NudityScore, ReportBurst; CEs Coordinated Harassment, EscalatingSeverity, BanEvasion.
- **SDEaaS:** topic heavy hitters; class-balanced reservoirs for rare classes; score distribution quantiles.
- **SuBiTO:** continuous class re-balancing, locale-specific thresholds, active-learning fine-tunes.
- **NeuroFlinkCEP:** policy rules convert neural scores to actions and audit trails.

4.8 Smart Retail Loss Prevention

- **Inputs:** PoS events, RFID gates, camera detections, staff schedules.
- **SEs/CEs:** SEs UnscannedItem, ExitGateEvent, StaffNearby; CEs PushoutRisk, TicketSwitching, ORCPattern.
- **SDEaaS:** store/aisle heavy hitters; quantile baskets; join-cardinality estimates for PoS↔gate windows.
- **SuBiTO:** adapts vision/sequence models to layout changes and seasonality.
- **NeuroFlinkCEP:** store policy, dwell-time rules, multi-sensor alignment.

4.9 Rail Network Operations

- **Inputs:** train telemetry, switch states, track occupancy, maintenance logs, weather.
- **SEs/CEs:** SEs SpeedLimitBreach, SwitchFlip, TrackHotBox; CEs NearMissPattern, CascadingDelay, WorkzoneIntrusion.
- **SDEaaS:** per-segment speed quantiles; heavy hitters for delay causes; reservoirs for rare failures.
- **SuBiTO:** drift-aware forecasters for delay propagation; anomaly nets for sensor drift.
- **NeuroFlinkCEP:** temporal logic on headways, work-windows, switch sequences.

4.10 Cybersecurity

- **Inputs:** auth logs, netflow, EDR alerts, DNS, cloud audit.
- **SEs/CEs:** SEs FailedLoginBurst, RareProcessTree, BeaconingScore; CEs Lateral Movement, CredentialStuffingWave, DataExfil.
- **SDEaaS:** heavy hitters/HLL for source/destination pairs; sketch-based cardinalities for unique principals; quantiles for session sizes.
- **SuBiTO:** adaptive thresholds and online fine-tuning during incidents.
- **NeuroFlinkCEP:** playbook rules (kill chain stages), suppression policies to reduce alert fatigue.

5 Engineering the MVP

5.1 Minimal Viable Stack

- **Messaging & execution:** Kafka (ingest), Flink/Dask/Kafka Streams (stream compute), object store for cold paths.
- **SDEaaS:** persistent Flink/Dask/Kafka Streams job set with REST/gRPC front-end; RocksDB/Kafka StateStores or other for synopses state maintenance.
- **SuBiTO:** control service with Prometheus for statistics collection; model registry (MLflow/W&B); trainers on Ray (on Kubernetes optional).
- **NeuroFlinkCEP:** Workflow authoring + FlinkCEP runtime; inference at the edge (ONNX/TensorRT) and cloud (PyTorch/TensorFlow) per optimizer.

5.2 Data Contracts & Schemas

- **Event envelope:**
  {ts,event_type,source_id,geo,features,provenance} with protobuf or Avro schemas and evolution policy.
- **Model I/O:**
  {model_name,version,input_schema,output:{class,score,explain}} + embedding side-channels if needed.
- **Synopsis descriptors:**
  {synopsis_id,type,params,window,retention,error_budget,mergeable}.

5.3 Deployment Topology

- Edge tier runs *light* SynapSEflow models and selected synopses (counting, heavy hitters) with periodic uplink of merged synopses.
- Regional tier hosts SDEaaS shards and selected CEP operators; cloud tier hosts SuBiTO controllers, model training, and global CEP aggregations.

5.4 Operator Placement & Scheduling

- DAG*4CER-style placement informed by latency budgets, accelerator locality, and bandwidth costs. Prefer edge placement for low-latency neural primitives; centralize high-fan-in pattern operators.

6 KPIs, Benchmarks, and Evaluation Methods

The KPIs, benchmarks, and error-budgeting guidance presented here are intended as planning boundary conditions for deployments of SDEaaS, SuBiTO, and NeuroFlinkCEP. Prior to rollout, target values (e.g., p95 latency, synopsis error bounds, model accuracy) are established, and a benchmark plan (baseline vs. integrated, load sweeps, drift drills) is defined to validate them. During pilots, any KPI breach is treated as a gating issue, with error budgets informing whether to tighten synopses, resize models, or adjust placement. In production, the same KPIs remain on dashboards, and burn-rate against SLOs is reviewed to trigger retraining, reconfiguration, or rollback**.**

Core KPIs. Throughput (events/sec), end-to-end latency (p95/p99), model accuracy, CEP precision/recall/F1, resource costs (CPU/GPU hours, memory, bandwidth), synopsis error bounds, time-to-adapt (drift→stable), SLA abidance.

Benchmark design. Baseline vs integrated; load sweeps; drift drills (covariate/prior/concept); edge/cloud toggles and network impairment tests.

Error budgeting. Map accuracy SLOs to error budgets (e.g., acceptable relative error in heavy hitters or quantiles) so SDEaaS and SuBiTO can adapt neural training parameters without violating business SLAs.

7 Risk Analysis and Mitigation

The practices below are intended as deployment planning inputs. During design reviews and pre-production readiness, security/privacy policies should be defined (7.1), then they should be translated into operational SLOs, runbooks, and change controls (7.2), eventually confirming that these choices differentiate the solution versus alternatives (7.3). Each item should be tied to concrete acceptance criteria (e.g., latency SLOs, synopsis error budgets, audit requirements – see Section 6) and verified in staging through targeted tests (canaries, chaos drills, load sweeps) before promotion to production.

7.1 Security, Privacy, and Governance

- **Data minimization by design:** Prefer synopses over raw data when possible; treat synopsis parameters (error budgets, retention) as *policy-controlled* entities.
- **Access control & tenancy:** Per-tenant namespaces; row-level security on raw streams; capability-based synopsis queries.
- **Provenance & audit:** Attach lineage to complex events (rules fired, model versions, synopsis IDs + parameters). Immutable audit logs for incident review.
- **PII handling:** Field-level encryption and tokenization; ensure synopses exclude direct identifiers or use privacy-preserving sketches.
- **Retention & right-to-erasure:** GC raw shards aggressively; version synopses to allow targeted invalidation.
- **Adversarial robustness:** Adversarial testing for neural primitives; symbolic sentry rules; rate-limit untrusted inputs.
- **Compliance:** Align with ISO 27001, SOC 2, GDPR/AI Act (risk management, transparency, human-in-the-loop), and sectoral regs.

7.2 Operational Model & SLAs

- **Runbook:** Declarative *policies* set latency/accuracy/cost targets; SuBiTO and DAG*4CER optimizer should reconcile at runtime.
- **SLOs:** p95 latency per pattern; model freshness windows; synopsis lag (e.g., ≤ 5s of watermark).
- **Change management:** Canary deployments (i.e., staged releases) for new models/patterns; automatic rollback on KPI regression; weekly chaos drills[1].
- **On-call & escalation:** Tiered alerts (synopsis lag, backlog growth, CEP miss rate); golden dashboards with burn-rate indicators.
- **Capacity planning:** Periodic load sweeps; cost/throughput curves; autoscaling policies for edge and cloud tiers.

[1] Regular, scheduled resilience tests where you intentionally inject faults into your system to prove you can detect, contain, and recover

7.3 Competitive Advantage

- **Competitors:** Raw-first streaming stacks; rule-only CEP engines; monolithic MLOps + batch retrains; agentic workflow tools.
- **Differentiation:** cutting edge research delivered into state-of-the-art prototypes, synopsis reuse at scale, adaptive training control and NeSy CEP + edge/cloud placement with explicit error/latency budgets.

8 Roadmap

This roadmap reflects a plan for the evolution of SDEaaS, SuBiTO, and NeuroFlinkCEP and should be iterated quarterly based on user feedback, field deployments, and research results. Priorities may shift with new requirements (e.g., compliance, security), performance learnings (cost/latency/error budgets), and ecosystem changes (hardware/runtimes). Backward compatibility and migration paths must be provided wherever feasible.

0–6 months. Harden SDEaaS APIs in Apache Dask/Kafka; multi-resolution synopses; synopsis-aware samplers in SuBiTO; initial NeuroFlinkCEP patterns for one flagship use case; automated model registry wiring.

6–12 months. Probabilistic CEP with uncertainty propagation; DAG*4CER optimizer extended with synopsis cost models; edge accelerators (Jetson/TPU) with ONNX runtimes; forecasting operators.

12–24 months. Federated synopsis merges; privacy-preserving summaries; auto-pattern synthesis; meta-learning for synopsis selection.

9 Business Value and TCO Rationale

Throughput per dollar: synopsis reuse and inference placement cut CPU/GPU-hours and network egress by avoiding redundant heavy compute and raw data transport.

Time-to-insight: NeSy patterns compress analyst intent into executable detectors; adaptive training keeps models relevant without manual retuning.

Reliability & trust: symbolic layers provide audit trails; synopses provide stable, bounded approximations; guard railed control loops provide predictability.

Portability: modular design compatible with mainstream streaming and MLOps ecosystems; avoids lock-in.

9.1 Mini Case Study

Context. City flood-response pilot (4 weeks); inputs: 120 camera streams, 45 gauges, radar tiles.

Before (raw-first baseline). p95 latency 3.1 s; GPU hours/day 420; egress 8.4 TB/day; Accuracy 0.81.

After (integrated stack). p95 latency 1.2 s; GPU hours/day 210 (−50%); egress 3.6 TB/day (−57%); Accuracy 0.87 (+6 pts).

Drivers. Edge inference placement; synopsis-assisted sampling; CEP pre-pruning with synopsis cardinality estimates; SuBiTO-guided drift responses.

9.2 Costing Sketch

Edge tier: ~60× Jetson-class nodes; avg 35% GPU utilization; ~500€/month power.

Regional/cloud: ~8x g5.xlarge equivalents for training bursts; 3x r6g.2xlarge for SDEaaS/CEP state; object storage ~1500€/month.

Ops: 0.4 FTE for SRE; 0.6 FTE for DS/ML AI Engineer/AI Expert.

10 Conclusion

SDEaaS, SuBiTO, and NeuroFlinkCEP are complementary building blocks of a next-generation, real-time AI analytics platform: synopses deliver scale and responsiveness, continuous and adaptive data parallel training sustains model quality in dynamic environments and NeSy CER supplies efficient execution across the cloud-to-edge continuum along with expressivity and explainability. Integrating them yields a scalable NeSy stack that can meet rigid SLAs while controlling cost and complexity across the cloud-to-edge continuum.

11 References

[REF-01]	Georgios Panagiotis Kalfakis, Nikos Giatrakos: SaaMS: The synopses-as-a-microservice paradigm for scalable adaptive streaming analytics across the cloud to edge continuum. Inf. Syst. 136: 102629 (2026) [https://github.com/geok1999/Synopses-as-a-MicroService-SaaMS]
[REF-02]	Michael Kratimenos: A Toolkit for Scalable Preprocessing and Neural Learning with Tensorflow and Dask, Diploma Thesis, Technical University of Crete, July 2025. https://doi.org/10.26233/heallink.tuc.104033
[REF-03]	Ourania Ntouni, Dimitrios Banelas, Nikos Giatrakos: NeuroFlinkCEP: Neurosymbolic Complex Event Recognition Optimized across IoT Platforms. Proc. VLDB Endow. 18(12): 5355-5358 (2025) [https://neuroflinkcep.github.io/]
[REF-04]	Errikos Streviniotis, Dimitrios Banelas, Nikos Giatrakos, Antonios Deligiannakis: DAG*: A Novel A*-Alike Algorithm for Optimal Workflow Execution Across IoT Platforms. ICDE 2025: 807-820
[REF-05]	Errikos Streviniotis, George Klioumis, Nikos Giatrakos: SuBiTO: Synopsis-based Training Optimization for Continuous Real-Time Neural Learning over Big Streaming Data. AAAI 2025: 29697-29699 [https://subito-ai-for-bigdata.github.io/]
[REF-06]	George Klioumis, Nikos Giatrakos: Data-driven Synchronization Protocols for Data-parallel Neural Learning over Streaming Data. IEEE Big Data 2024: 988-997
[REF-07]	Antonios Kontaxakis, Nikos Giatrakos, Dimitris Sacharidis, Antonios Deligiannakis: And synopses for all: A synopses data engine for extreme scale analytics-as-a-service. Inf. Syst. 116: 102221 (2023) [https://sdeaas.github.io/]
[REF-08]	EVENFLOW-project-EU. *Scalability-Toolkit*. GitHub, https://github.com/EVENFLOW-project-EU/Scalability-Toolkit. Accessed 22 Oct. 2025.

Appendix A – Example CEP Patterns

A.1 Telecom

```
// At least 3 long premium calls at night (per caller)
PATTERN FrequentLongCallsAtNight := ( A B? C ){3} WITHIN 24h
SELECTION_STRATEGY = RELAXED_CONTIGUITY
CONSUMPTION_POLICY = NO_SKIP
EMIT { caller_id, window_start, window_end, count }
```

A.2 Robotic

```
// Collision → stop → movement within 2 time units
PATTERN CollisionRecovery := A B [C-L] WITHIN 2u
SELECTION_STRATEGY = STRICT_CONTIGUITY
CONSUMPTION_POLICY = SKIP_PAST_LAST
EMIT { robot_id, from_station, to_station, recovery_time }
```

A.3 Flood (multimodal)

```
// Detect flood risk from multi-modal inputs
PATTERN FLOOD_RISK :=
  SEQ(
    RainRate(r) WHERE r > θ_rain,
    RiverLevel(l) WHERE Δq95(l) > θ_q,
    DebrisDetected(d) WHERE cluster_size(d.geo,150m,10min) ≥ k
  ) WITHIN 15min
  EMIT {
    risk_score := f(p_rain, p_level, p_debris, synopsis_error_
bounds),
    area := union(d.geo),
    provenance := {matched_rules, contributing_models}
  }
```

Appendix B - Frequently Asked Questions (FAQs)

- **Q:** Can synopses hurt accuracy?
  A: Yes if mis-tuned; mitigate with sentry rules and error-aware thresholds.

- **Q:** Do we need GPUs at the edge?
  A: Yes, for real-time training and inference purposes; CPU-only modes exist with higher latency.

- **Q:** How do we roll back a bad model?
  A: Re-run the SuBiTO optimizer to reconsider currently deployed models and training configurations.

- **Q:** What if connectivity drops?
  A: Edge caches maintain synopses and local CEP; results reconcile on reconnect.

- **Q:** Who decides which synopses to maintain?
  A: Start with a policy catalog (quantiles, HHs, cardinalities, reservoirs). SuBiTO and SDEaaS can be trivially extended to co-optimize: drop low-value synopses, increase precision for those driving KPIs.

- **Q:** How is drift detected and handled?
  A: Combine statistical drift tests with task metrics (precision/recall, latency). SuBiTO triggers partial fine-tunes, data re-balancing, or architecture swaps; CEP thresholds can auto-rebase using synopsis quantiles.

- **Q:** How does multi-tenancy/isolation work?
  A: Per-tenant namespaces, resource quotas, and access control on both raw topics and synopsis endpoints. No cross-tenant merges without explicit policy.

- **Q:** What's an error budget in this context?
  A: A contract tying business KPIs (e.g., F1 ≥ 0.85, p95 ≤ 1.5s) to approximation limits (e.g., HH $\varepsilon \leq 2\%$, q95 error ≤ 0.5%). SDEaaS/SuBiTO adapt sketch parameters within that budget.

- **Q:** How do we reduce CEP false positives/negatives?
  A: Calibrate neural thresholds via SuBiTO, propagate uncertainty into pattern matches via NeSy training, require multi-sensor cross correlations, and add cooldowns/debounce rules.

- **Q:** Can we reproduce a historical decision?
  A: Yes, with extensions. If model/version, pattern version, synopsis IDs + params, and data hashes with each CE are stored. Replays can use frozen artifacts to regenerate outcomes.

- **Q:** What happens during input surges?
  A: SuBiTO switches to surge profiles: lighter models, higher synopsis compression,

adaptive sampling, and back-pressure aware CEP windows, while respecting hard SLOs first.

- **Q:** How is cost optimized?
  A: Place inference where it is cheapest for SLA, shrink/expand synopses based on marginal value, prefer reservoirs over raw retention.

- **Q:** How do we integrate with existing stacks?
  A: Use Kafka topics and REST/gRPC for SDEaaS; model registry integrates with Kafka/Ray; NeuroFlinkCEP compiles to FlinkCEP jobs; dashboards consume CE topics.

- **Q:** What telemetry should we watch?
  A: p95/p99 end-to-end, CE throughput, CEP miss/FP rates, synopsis lag and error estimates, model QPS/latency, drift scores, and burn-rate for SLOs.

- **Q:** How do we secure synopses and models?
  A: One can Encrypt at rest/in transit, capability-scoped tokens for queries, private model registries, and PII-safe synopsis designs (no direct identifiers).

- **Q:** Are we compliant-ready (GDPR/AI Act)?
  A: Yes, via data minimization, transparent provenance, human-in-the-loop overrides, and retention/erasure controls; perform DPIAs for high-risk use cases.

- **Q:** Vendor lock-in concerns?
  A: Components are loosely coupled (Kafka/Flink/Dask, ONNX, Ray). Synopses and patterns can be portable; training and CEP runtimes can be swapped with adapters.

- **Q:** How do we measure success in production?
  A: Compare baseline vs integrated on latency, accuracy, cost per decision, and time-to-adapt; review monthly error-budget reports and incident MTTR.

This project is funded by the EU under Horizon Europe Grant Agreement No 101070430.
Views and opinions expressed are those of the EVENFLOW consortium only and
do not necessarily reflect those of the EU or the European Commission, neither the EU
nor the granting authority can be held responsible for them.