trategy & tactics to steer Nortal toward a Lattice-like federated mesh COP (vendor-neutral)

North-star outcome

A national COP composed of **many autonomous nodes** (edge + regional + national), where:

- nodes keep operating during partitions (no SPOF),
- data sharing is **subscription-based** (publish/subscribe), not "everything funnels to the center,"
- integration happens via an **SDK + conformance harness** (translator/adapter ecosystem),
- deployment is **topology-parity**: the same services run on-prem, in sovereign cloud, or hybrid.

This directly builds on Nortal's own stated principles: no SPOF, replaceable subsystems, regional + central usable even if either is down, edge device requirements, and x-tee-like edge behavior.

---

1) Reframe the "distributed architecture" requirement into *operationally testable* behavior

What Nortal already has (good base)

- "No single point of failure (SPOF)" and replaceable subsystems.

NORTAL

- Regional + central can work even if one is unavailable.

NORTAL

- Edge device must do detection/classification, encrypt/sign traffic, remote management.

NORTAL

- Wants x-tee-like edge security servers that can send even without central.

NORTAL

The nudge: make "mesh" explicit via acceptance tests

Insert into the analysis outputs and later procurement spec as *MUST-pass tests*:

Partition resilience tests

- **T1:** With national/central node offline, each region continues COP for its AO (ingest → fuse → display → alert).

- **T2:** With region isolated, it continues; when reconnected, it reconciles without operator hand-holding (store-and-forward + conflict rules).
- **T3:** With a subset of edge nodes disconnected from region, local edge COP persists at minimum “alerts + last-known tracks.”

Multi-level fusion

- **T4:** Fusion runs at **edge + regional + national** concurrently. Center is not a mandatory chokepoint.

This converts “distributed” from a diagram into something vendors must *demonstrate*.

---

2) Separate the planes: governance/control vs real-time data

Nortal’s x-tee-like idea is strong for governance, but it can accidentally drag real-time sensor flows into a document-exchange mindset.

NORTAL

Required framing in the analysis

- **Governance plane:** identity, authentication, authorization, audit, policy distribution, legal traceability.
- **Data plane:** low-latency pub/sub for tracks/events/video-metadata streams.

Tactic

Ask Nortal to deliver **two architecture views** (C4-style works):

1. control plane (x-tee-like or equivalent pattern),
2. data plane (pub/sub, selective subscription, priority/QoS).

That one move makes “Lattice-like mesh” feel like the natural solution, while keeping your legal/security stakeholders happy.

---

3) Elevate cloud-native and SDK without making it smell like “cloud mandate”

Nortal already flags classified vs public networks and the need to test cross-domain behavior.

NORTAL

Use “deployment topology parity” language:

Insert into evaluation / requirements

- “Core services must be deployable **unchanged** on-prem or sovereign cloud (containerized microservices).”
- “Hybrid mode must be supported: some services in restricted networks, some in public, with controlled replication.”

Weighting (do it ethically)

Instead of “cloud-native 25%,” weight **capabilities**:

- **Disconnected ops + partition tolerance**
- **Integration velocity (new sensor in ≤2 weeks via published adapter contract)**
- **Deployment flexibility (on-prem / cloud / hybrid parity)**
- **Operability (telemetry, audit, reproducible deployments)**

Cloud-native architectures tend to win these, but you aren’t explicitly biasing to one vendor.

---

4) Force the “translator ecosystem” by making SDK + conformance harness a deliverable

Your own standards plan already emphasizes adapters + canonical schema + conformance tests. Make Nortal’s analysis produce procurement-ready artifacts:

A. Canonical model

- Canonical **Entity/Track/Event** schema with: position/velocity, confidence, lineage/provenance, policy tags (classification/releasability, precision downgrade rules). (This supports multi-agency masking.)

B. Adapter contract

- One “primary” integration surface (recommend: **gRPC/protobuf**) plus one fallback (REST/JSON).
- Adapters publish capabilities via a manifest (versioning + supported outputs).

C. Certification harness

- Golden test vectors + replayable datasets.

- “Adapter passes harness” becomes procurement gate.

This is the single most reliable way to “nudge toward Lattice-like” behavior without naming Lattice.

---

5) Shift the edge paradigm: edge nodes are autonomous fusion participants

Nortal already requires edge to classify detections, sign/encrypt, and remote-manage.

NORTAL

Nudge the analysis to treat edge as a **first-class node**:

Add three explicit edge requirements

- Edge node can maintain a **local track store** + local alerting rules.
- Edge-to-edge and edge-to-region **peer comms** is supported (not only edge→center).
- Edge supports **inference slot** (hardware-agnostic): “deployable model inference container” for rapid algorithm updates.

---

6) Standards prioritization: keep the “OR” clauses and avoid lock-in

Your plan already recommends flexibility: ASTERIX or SAPIENT; RIST or SRT; PTP baseline; TSN where justified.
Tactically, ensure Nortal doesn’t collapse the options into a single mandated stack during analysis.

Procurement-safe phrasing

- “System SHALL ingest **ASTERIX CAT-062/240** and MAY ingest **SAPIENT**; internal normalization is required either way.”
- “Backhaul SHALL support **SRT or RIST**.”
- “Timebase SHALL support PTP; TSN is required only for segments with proven need.”

This preserves competition and still aligns with Anduril-strength integrations.

---

7) Ukraine interviews: steer the questions toward degraded comms + decentralization

Nortal already plans Ukraine expert interviews.

NORTAL

Give them a **question set** that naturally validates mesh/edge autonomy:

- “What failed first under EW: links, central nodes, or sensors?”
- “What minimum picture was still usable when central C2 was degraded?”
- “What local/edge processing mattered most?”
- “What data products were shared peer-to-peer vs pushed centrally?”
- “How did you handle provenance + trust when data was partial/spoofed?”

You’re not saying “Anduril did X,” you’re making the analysis discover the conditions where mesh wins.

---

8) Security reframing: zero trust + edge-held keys, expressed as testable requirements

Nortal is already deep on stakeholders, access levels, auditability, and legal constraints.
Turn “zero trust” into checks:

- mutual auth everywhere (service-to-service and node-to-node),
- per-message signing for critical events/tracks,
- ABAC enforced at the API and subscription layer (not only UI),
- audit logs include “who saw what / subscribed to what.”

Practical intervention plan (what to do immediately)

A) Submit an “Architecture Guardrails” annex (1–2 pages)

Non-negotiables:

- partition tolerance + offline-first,
- multi-level fusion,
- two-plane separation,
- SDK + conformance harness,

- deployment parity (on-prem / cloud / hybrid),
- policy-tagged data model.

Anchor it to Nortal’s own SPOF + edge + distributed intent so it reads like refinement, not derailment.

B) Force the prototype to prove mesh (not just ingest→map)

Prototype success criteria:

- 2 regions + 1 national node + 3 edge clusters,
- run T1–T4 partition tests,
- integrate 3 sensor types via adapter contract (radar track, RF, Remote ID),
- show ABAC masking across agencies.

C) Make “integration time” a scored capability

Score vendors on:

- time to onboard a new sensor using the SDK/contract,
- ability to run the conformance harness,
- ability to deploy the same stack in a disconnected edge site and in cloud.

This is where monoliths sweat.

[KOMMENTAAR XTEE-1 | paigutus: 1.6 „Sensorvõrgustiku vaade” lõiku, kohe pärast lauset „Nortal eelistab x-tee laadset lahendust …”]
X-tee/turvaserveri analoogia on tugev *governance* (identiteet, autentimine, audit, õiguslik jälgitavus) jaoks, kuid reaalaja sensorvõrgu *andmeplaanina* (tracks/events/video-metadata) tekitab see vale arhitektuurilise paradigma: X-tee on valdavalt päringu-/dokumendivahetuse loogikaga, mis on keeruline sobitada madala latentsuse, QoS-i, selektiivse tellimise (subscription) ja katkestustaluvusega (partition-tolerant) voogudele. Risk on, et “keskserver pole kättesaadav” muutub praktikas siiski koordineerimis- ja jõudlus-chokepoint’iks või sunnib ehitama suure hulga erandeid, mis lõhuvad ühtset arhitektuurimustrit. Seetõttu soovitame X-tee-laadse lähenemise *asendada andmeplaanis* standardse pub/sub (publish-subscribe) ja

“tõlkijate/adapterite” mudeliga ning jätta X-tee-laadne mehhanism *juhtimis- ja kontrollplaani* (poliitikad, ABAC, audit, sertifikaadid) jaoks.

NORTAL

Soovituslik asendus (Anduril-laadne, kuid hankes/vormistuses neutraalne):

1. **Data plane (reaalaja pilt):** pub/sub sõnumivahendus + versioonitud integratsiooni-API. Tehniliseks liideseks määrata *programmi integraatsioonileping* (SDK): **gRPC/Protobuf** reaalajaks + **REST/OpenAPI** laiemaks ökosüsteemiks; “translator/adapter” konteinerid kirjutatakse selle lepingu vastu. See on otseselt kooskõlas MOSA põhimõtetega ja võimaldab turul tekkinud “open architecture + developer kit” integratsioonimudelit, kus kolmandad osapooled lisavad sensoreid adapterite kaudu.
2. **Payload-standardid (mida üle võrgu kantakse):** säilitada mitme standardi tugi ja vältida ühtset “tõlkekeskust”: **ASTERIX** (nt CAT-062/CAT-240) ja/või **SAPIENT** (kui allikad seda toetavad) sisendina *igal fusion-tasemel* (EDGE/regionaal/keskne), normaliseerides sisemisse kanonilisse mudelisse adapterite kaudu.
3. **Control/Governance plane (õigused ja jälgitavus):** X-tee-laadne muster võib jääda *poliitikate ja auditi* tarbeks, kuid peab olema disainitud nii, et **reaalaja töö ei sõltu alati-onlines keskssetest teenustest** (edge peab suutma töötada poliitikate vahemäluga).

NORTAL

Miks see on parem kui X-tee andmeplaanina (argument “tough”):

- Pub/sub annab loomuliku “tellimuspõhise” jaotusmudeli (ainult vajalik info liigub), mis vähendab koormust ja toetab katkestustaluvust (võrk võib jaguneda ja hiljem koonduda).
- SDK + adapterite leping teeb integratsiooni *korratavaks*: iga uus sensor ei ole eraldi projekt, vaid sertifitseeritav adapter.
- ASTERIX/SAPIENT jäävad “turunõuetele” ja liitlasstandarditele vastavaks väliseks keeleks; sisemine mudel ja API kontrollivad evolutsiooni ja ühilduvust.

Soovituslik lisatav vastuvõtutest (üks lause lõppu):
“Tulevases prototüübis tuleb demonstreerida, et EDGE-sõlm publitseerib rajajälgi pub/sub kanalisse gRPC streaminguga, regionaalne sõlm tellib ja fuseerib; keskse komponendi väljalülitamisel säilib lokaalne COP ja hilisem resünkroniseerimine toimub automaatselt.”

[Tugevus: väga tugev; Anduriliga ühilduvus: kriitiline (sama “open API + translator/adapter + pub/sub mesh” muster, kuid sõnastus jääb vendor-neutraalseks).]

Võimalik pitfall / alternatiiv: DDS/RTPS annaks kõige “päris” reaalaegse pub/sub QoS-i, aga selle juurutus ja operatiivne keerukus võib olla suurem; pragmatiline vahevariant on gRPC streaming + message-bus (nt NATS/Kafka) ning DDS hoida segmentides, kus QoS/latentsus nõuab.