INTERNATIONAL TELECOMMUNICATION UNION
TELECOMMUNICATION STANDARDIZATION SECTOR
STUDY PERIOD 2025-2028

FOCUS GROUP ON AI NATIVE FOR TELECOMMUNICATION NETWORKS (FG-AINN)

FG-AINN-I-xxx

Original: English

Question(s):	N/A	New Delhi, 17th February, 2026

INPUT DOCUMENT

Source:	Manipal Institute of Technology, Bengaluru
Title:	Build-a-thon 4.0 2026: Intent-Driven AI-Native Precise Positioning-as-a-Service (ATSC 3.0 broadcast + RTK/bitmap + closed-loop AI control).

Contact:	Anirudh Nishtala Dept. of ECE, 4th Year Manipal Institute of Technology, Bengaluru	E-mail: anirudh2.mitblr2022@learner.manipal.edu
Contact:	Rishiraj Rakesh Kumar Dept. of ECE, 4th Year Manipal Institute of Technology, Bengaluru	E-mail: rishiraj.mitblr2022@learner.manipal.edu
Contact:	Tarunika D Dept. of ECE, 4th Year Manipal Institute of Technology, Bengaluru	E-mail: tarunika.mitblr2022@learner.manipal.edu
Contact:	Dr. Shweta Singh Asst. Prof., Dept of ECE Manipal Institute of Technology	E-mail: shweta.s@manipal.edu

Keywords: AI Native Network; PoC;

Abstract: This document describes a submission towards Build-a-thon 4.0, 2026 conducted in New Delhi on February 17th, 2026.

1. Introduction

1.a) Background to the Topic: The Precise Positioning Imperative

Precise positioning is a foundational dependency for safety-critical and mission-critical mobility services such as autonomous driving, advanced driver assistance (ADAS), fleet automation, emergency response, and intelligent logistics. Under open-sky conditions, conventional GNSS can provide acceptable positioning, but its performance degrades sharply in tunnels, urban canyons, dense city blocks, and signal-blocked zones, where satellite visibility is reduced and multipath reflections dominate. In such conditions, positioning errors can escalate from centimetres to metres within seconds, forcing systems to fall back to dead-reckoning and other sensors, increasing cost and operational complexity.

High-precision techniques such as RTK (Real-Time Kinematic) can deliver centimetre-level accuracy by applying real-time corrections derived from reference stations. However, RTK depends on continuous availability of correction messages and stable radio delivery, which is often unreliable or economically inefficient in challenging coverage environments. Consequently, there is a practical need for a robust, scalable, and cost-efficient correction delivery and control mechanism that sustains high-precision positioning even when GNSS observability and connectivity fluctuate.

This PoC addresses that need by leveraging broadcast delivery (ATSC 3.0) for wide-area dissemination of correction and positioning assistance data, combined with an AI-native control loop that dynamically adapts the delivery strategy based on observed fleet positioning outcomes.

1.b) Need for AI-Native Positioning Automation (Non-Radio Focus)

Traditional positioning augmentation systems typically operate with static configurations: fixed correction message types, fixed update rates, fixed robustness settings, and limited feedback regarding whether field receivers actually achieve the intended RTK performance. This “configure once, operate forever” approach is misaligned with real environments where the optimal trade-off between accuracy, reliability, latency, and broadcast resource usag**e** varies by geography (tunnels vs open sky), time (traffic density, blockage patterns), and receiver conditions.

The proposed PoC adopts an AI-native, intent-driven paradigm, where the system exposes a service intent interface (e.g., *maximise accuracy*, *maximise reliability*, *minimise bandwidth*, *prioritise tunnel corridors*) and continuously translates that intent into real-time operational decisions for the positioning assistance pipeline. The key shift is the introduction of a closed-loop feedback controller that uses fleet telemetry to evaluate outcomes (e.g., RTK FIX availability, positioning degradation events) and then adjusts broadcast and edge delivery parameters accordingly.

This architecture is explicitly aligned with AI-native principles for non-radio functions: it embeds intelligence into the operational fabric via continuous observation → decision → actuation → validation, enabling the system to (i) adapt to real-time conditions, (ii) avoid wasteful over-provisioning during “good” conditions, and (iii) increase robustness when the environment deteriorates.

1. c) Proposed PoC Overview: Intent-Driven AI-Native Precise Positioning-as-a-Service

This PoC demonstrates an **Intent-Driven AI-Native Precise Positioning-as-a-Service** platform that sustains high-precision vehicle positioning in extreme GNSS environments (tunnels, urban canyons, blockage zones) by combining:

- **GNSS reference station corrections** (RTK-style error vectors; RTCM-like correction streams),
- **Broadcast delivery using ATSC 3.0** (one-to-many dissemination of corrections and positioning assistance),
- **Bitmap / grid-based positioning assistance** (coverage-aware augmentation, location-dependent robustness and assistance payloads),
- **AI-driven adaptive control** of correction delivery policies (e.g., update rate, redundancy, tile/bitmap resolution, robustness/FEC-style settings, and bandwidth allocation).

The core innovation is that a **real-time AI controller** adapts the positioning assistance strategy based on fleet-level observed performance. Instead of assuming that “more corrections” always help, the system continuously measures whether vehicles maintain high-precision states (e.g., RTK FIX) and dynamically adjusts the pipeline to maximise outcome metrics under resource constraints.

Target outcomes for the PoC are:

- **3–10 cm positioning accuracy** under representative conditions,
- **high reliability** in rural and obstructed zones (including tunnel-like signal loss events),
- **efficient utilisation of ATSC 3.0 broadcast capacity**, prioritising robustness only where and when needed.

2. State of the Art in High-Precision Positioning Augmentation

2.a) Current High-Precision GNSS and Correction Delivery Approaches

Standard GNSS typically provides metre-level accuracy, which is insufficient for autonomy and safety-critical mobility. RTK-class positioning improves accuracy to centimetre-level by using carrier-phase measurements together with real-time corrections from a reference/base station; however, RTK performance depends on timely delivery of those corrections and degrades when conditions deteriorate (e.g., blockage, multipath). This PoC positions **ATSC 3.0** as a scalable one-to-many correction distribution channel, leveraging wide-area broadcast range and configurable robustness/capacity trade-offs (with practical broadcast latency on the order of seconds) to disseminate correction and assistance data to many receivers simultaneously.

2.b) Broadcast-Delivered Positioning Assistance and Adaptive Broadcasting

Academic work has investigated GNSS error modelling (ionospheric/tropospheric effects), multipath-aware positioning, map-assisted localisation, and learning-based techniques to predict or detect positioning degradation. Research has also explored adaptive communications and edge processing to optimise reliability and latency for distributed systems. Despite these advances, many solutions remain siloed: either optimising GNSS estimation without controlling delivery resources, or optimising delivery without closing the loop on positioning outcomes at the receiver.

2.c) Practical Standardisation and Interoperability Considerations

High-precision correction ecosystems often depend on established message formats and receiver interoperability constraints. Broadcast delivery introduces additional considerations around packaging, scheduling, and robustness configuration. A key practical requirement is therefore an architectural approach that preserves interoperability at the positioning layer while enabling policy-level adaptation at the orchestration layer.

3. Research Gaps and System Limitations Addressed by this PoC

Analysis of real-world deployment constraints reveals four core gaps that this PoC addresses:

- **GPS Signal Loss in Tunnels and Deep Blockage Zones:** Vehicles can lose centimetre-level positioning for extended intervals when satellite visibility collapses. A static correction delivery strategy does not provide sufficient robustness across such transitions.
- **Uncertainty in Urban Canyons (Multipath and NLOS):** GNSS performance degrades due to reflections and non-line-of-sight reception. Without environment awareness, systems cannot selectively increase robustness where multipath risk is highest.
- **Wasted Broadcast Resources Under Good Conditions:** Broadcasting at a uniformly high robustness level everywhere is inefficient. The system needs to scale broadcast robustness and payload composition according to real-time needs.
- **No Closed-Loop Feedback from Field Outcomes:** Traditional systems do not directly optimise for fleet-level outcome metrics (e.g., RTK FIX availability, degradation duration).

This PoC introduces a telemetry-driven feedback loop so that decisions are validated against observed positioning performance.

4. Technological Foundations and Proposed Architecture (AI-Positioning PoC)

4.a) A Closed-Loop Intent-Driven Control System

The PoC is designed as a closed-loop system that continuously translates service intent into operational control actions:

Observe: Vehicles/receivers provide positioning telemetry (e.g., accuracy indicators, FIX/float state, degradation events) and contextual signals (location corridor, blockage risk).
Decide: An AI decision engine selects policies that best satisfy the declared intent (accuracy/reliability/bandwidth objectives).
Act: The controller generates a broadcast assistance configuration (payload composition, update strategy, robustness level, bitmap/tile policies).
Validate: Outcomes are measured at the fleet level; the controller updates policies to improve performance over time.

This delivers AI-native behaviour by construction: the system is outcome-driven, adaptive, and continuously validated against real telemetry.

4.b) Broadcast-Assisted Positioning Delivery Using ATSC 3.0

ATSC 3.0 is utilized as the wide-area broadcast mechanism to disseminate positioning assistance data efficiently at scale. The PoC design uses broadcast delivery to reduce per-vehicle dependency on unicast connectivity while preserving the ability to adapt delivery robustness dynamically. Where required, the architecture supports fallback policies (e.g., selective unicast supplementation) without undermining the core broadcast-first design.

4.c) System Specifications and Success Metrics

i. **Target accuracy:** Maintain **centimetre-class positioning (≈3–10 cm, where conditions permit)**, with controlled degradation behaviour under deep blockage and severe multipath (e.g., tunnel segments, dense urban canyon).
ii. **Reliability objective:** Increase **RTK FIX availability** and **valid correction availability** by ensuring broadcast-delivered corrections/assistance remain decodable and fresh across varying channel conditions, reducing time spent in degraded states (FLOAT / standalone).
iii. **Continuity objective:** Minimise service interruptions by reducing **dropout/degradation events per corridor/trip** and reducing outage-induced error spikes, with faster recovery back to FIX after GNSS loss or impaired reception.
iv. **Efficiency objective:** Optimise ATSC 3.0 broadcast resource usage by dynamically tuning the **positioning datacast policy**—including **correction update rate**, **redundancy/repetition**, **FEC/robustness settings**, and **bitmap/tile resolution/refresh strategy**—based on fleet telemetry, channel state, and geo-context, while staying within an intent-dependent broadcast budget.
v. **Closed-loop control specification:** The **Data Aggregator & AI Feedback Controller (AI Agent)** shall ingest fleet/receiver telemetry and environment features, run inference with a confidence check, and output actionable configuration updates for the ATSC 3.0 positioning service (PLP distribution/robustness, payload

strategy, and scheduling). The broadcast chain shall apply these updates and the resulting positioning outcomes shall be fed back to the controller to close the loop.

5. Current Status & Demo Proposal

Our project, **"Intent-Driven AI-Native Precise Positioning-as-a-Service over ATSC 3.0,"** will culminate in a demonstration of a complete, closed-loop **adaptive positioning datacast** workflow in a controlled simulation/testbed. The demo will show how fleet telemetry and operator intent drive **dynamic configuration of the ATSC 3.0 positioning service**—including correction delivery strategy and assistance/bitmap policy—so that **centimetre-class positioning performance is sustained where feasible** while **broadcast resources are kept within a configured budget**.

The demonstration will visualise the following sequence:

- **The system observes the fleet and channel state**, using aggregated positioning outcomes and broadcast/network indicators (e.g., FIX/FLOAT distribution, convergence time, correction availability/age, packet loss patterns, and geo-context such as open-sky vs tunnel/urban canyon via assistance maps).
- **The Data Aggregator & AI Feedback Controller executes the decision loop**:
  - gathers fleet telemetry and environment features,
  - runs **neural-network inference** to determine broadcast configuration,
  - performs a **confidence check** and falls back to conservative/rule-based logic when confidence is low,
  - generates a broadcast command that adjusts the positioning datacast policy (e.g., **redundancy**, **correction update frequency**, **bitmap/tile resolution**, **FEC overhead**, and **PLP distribution**).
- **The ATSC 3.0 Broadcast Encoder applies the AI command** by packaging RTCM correction frames and assistance/bitmap tiles according to the selected policy (priority, scheduling, redundancy insertion, and robustness settings), and transmits them over the ATSC 3.0 RF channel using the configured protection settings (e.g., FEC/PLP choices).
- **The receiver / client pipeline consumes the datacast** (corrections + assistance), updates the positioning solution (e.g., RTK engine / fusion parameters as applicable), and reports the resulting performance back to the controller to close the loop (telemetry → decision → broadcast update → improved outcome).
- **The results are presented as baseline vs AI-adaptive comparative runs**, showing time-series improvements in positioning stability and continuity under impairments (tunnel-like outages / urban-canyon multipath) alongside controlled broadcast overhead and spectrum use.

Success metrics measured by:

- **Positioning Quality:** ΔHPE_p95 (and where applicable ΔVPE_p95), RTK FIX availability (% time FIX), and **TTR** (time-to-recover after outage/degradation).
- **Reliability / Correction Availability:** valid correction availability at the receiver and correction age / freshness under stress scenarios (tunnel/urban canyon).
- **Correction Delivery Latency:** broadcast → receiver decode latency and end-to-end time to usable correction/assistance at the client.
- **ATSC 3.0 Spectrum Efficiency / Overhead:** relative broadcast resource usage versus baseline (e.g., effective bits/s/Hz for correction PLP and incremental overhead from redundancy/FEC/tiles).
- **Service Continuity:** dropout/degradation event rate per trip/corridor and reduction in outage-induced error spikes (especially in tunnel/urban canyon segments).

Target: Maintain **centimetre-class performance (≈3–10 cm where conditions permit)**, improve **RTK FIX availability** and reduce **TTR**, while keeping broadcast usage **within the configured budget** via adaptive redundancy/FEC/update-rate/tile policy decisions.

Submission Id		*FG-AINN-PoC-xxx* *(Id number is to be assigned by the secretariat)*
Title		Intent-Driven AI-Native Precise Positioning-as-a-Service over ATSC 3.0 (RTK + Bitmap Assistance with Closed-Loop Orchestration)
Created by		Team: NoWiresAttached Team lead: Ms. Tarunika D. Team members: 1.Mr. Anirudh Nishtala 2.Mr. Rishiraj Rakesh Kumar Internal guide: Asst. Prof. Dr. Shweta Singh
Creation date		13 January 2026
Category		Intent-Based AI Agents; AI-Native Orchestration; Distributed Decision-Making; Resilience-Centric AI Inference (Service Continuity under GNSS Impairments)
PoC Objective		To demonstrate an **intent-driven AI-native precise positioning system delivered through ATSC 3.0 broadcast**, where operator/service intents (e.g., **maximise accuracy**, **maximise reliability in challenging environments**, **optimise ATSC 3.0 spectrum usage**) are **automatically translated by an AI agent into real-time configurations** of the positioning pipeline (broadcast PLPs, correction strategy, and edge policies).
Description		This PoC implements a **closed-loop broadcast-assisted RTK positioning service**. A base/reference segment generates **RTCM/RTK corrections** and **bitmap/grid error maps**, which are delivered over **ATSC 3.0 datacast**. Vehicles/clients compute positions using an RTK engine (e.g., RTKLIB) and send **telemetry feedback** (e.g., fix availability/quality indicators) back to the AI decision function. The **AI agent adapts the broadcast strategy in real time** by tuning parameters such as **PLP robustness/FEC, redundancy, correction update rate, and tile resolution**, so that centimeter-level performance is sustained in **tunnels/urban canyons/signal blockage zones**, while controlling broadcast resource usage.
Feedback to WG1	*Gaps Addressed*	• Demonstrates **intent-driven orchestration** where high-level service intents are mapped to **concrete broadcast + positioning controls** (PLP configuration, RTK/SSR/bitmap policy, edge processing). • Establishes a measurable **closed-loop control model** for AI-native positioning: fleet/receiver telemetry → AI decision → broadcast adaptation → improved positioning outcomes. • Defines **PoC-level KPIs** spanning accuracy, reliability, latency, spectrum efficiency, and continuity for evaluation consistency.
POCs Test Setup		**Software-defined PoC stack (open-source-first):** • **GNSS/RTK:** RTKLIB-based baseline and assisted positioning runs producing .pos outputs. • **Broadcast pipeline (ATSC 3.0 data):** ALP encapsulation +

		ROUTE/DASH packaging + service signalling; positioning data carried as datacast objects/streams. • **AI control loop:** intent parser + decision engine/optimiser + controller that updates broadcast/positioning parameters (e.g., PLP robustness, correction rate, tile resolution). • **Receiver/client pipeline:** ATSC 3.0 client (ALP/ROUTE/DASH) feeds correction/bitmap inputs into positioning client (RTK/SSR + bitmap consumer). • **Orchestration/analytics interface:** REST/gRPC style intent input plus real-time analytics outputs for demo visualization.
Data Sets		• **GNSS observation logs** for baseline and assisted runs (sample logs sufficient for PoC). • **RTK correction frames (RTCM/MSM):** from RTKLIB sample RTCM or synthesised generator. • **Bitmap tiles / grid maps:** small synthetic tiles (e.g., 100×100) representing correction coverage / environmental error masks. • **Telemetry features:** correction age, position residuals, integrity indicators, PLP loss, and environmental state indicators used as AI inputs.
Feedback to WG2	*Simulated Use cases*	The PoC evaluates intent-driven adaptation under representative scenarios: 1. **High-Accuracy Rural Drone:** intent "guarantee sub-10 cm accuracy" → increase correction rate, high-resolution tiles, stronger PLP robustness. 2. **Low-Bandwidth Rural Coverage:** intent "optimize ATSC spectrum use" → reduce correction bitrate, switch to lower-rate policy, reduce PLP bandwidth. 3. **Urban Canyon Reliability:** intent "maximize reliability" → strengthen grids/tiles + integrity updates, adjust multipath mitigation/fusion weighting.
Feedback to WG3	*Architectural concepts*	**Reference architecture** comprising: • GNSS reference/base station segment producing **RTCM corrections** and **grid/bitmap error models** with timing/latency awareness. • Broadcast core hosting intent-driven control and selecting **PLP strategy**, RTK vs SSR ratio, bitmap resolution/update rate, redundancy and FEC. • ATSC 3.0 broadcast RAN with dedicated positioning PLPs (e.g., RTK correction stream; bitmap/error maps/integrity). • Receiver/vehicle stack consuming broadcast corrections and producing RTK solution + telemetry feedback loop.
Demo and Evaluation		**Demo flow:** Baseline static broadcast/assistance policy versus **AI-intent-driven adaptive policy**, shown through time-series plots and comparative runs demonstrating (a) improved positioning performance/continuity and (b)

	controlled broadcast resource usage. **Evaluation KPIs (reported per scenario and overall):** • **Positioning Accuracy:** horizontal/vertical error. • **Reliability:** valid correction availability. • **Correction Delivery Latency:** broadcast → UE decode. • **ATSC 3.0 Spectrum Efficiency:** bits/s/Hz for correction PLP. • **Service Continuity:** dropout rate (rural/urban canyon).
PoC Observation and discussions	During implementation, the team will document: (i) which intents produced the most measurable gains, (ii) observed trade-offs between robustness and broadcast overhead, (iii) stability of the feedback loop under changing channel conditions, and (iv) limitations encountered due to synthetic vs real field datasets and transmitter control constraints.
Conclusion	The PoC demonstrates that **intent-driven AI control** can dynamically adapt **ATSC 3.0 broadcast assistance** (PLP robustness, redundancy, correction strategy and tile policy) using fleet telemetry, enabling **centimetre-class RTK positioning** to be sustained more reliably in degraded GNSS environments while maintaining **efficient use of broadcast resources**.
Open Problems and Future Work	• Validate with **real field GNSS + mobility datasets** and real urban canyon/tunnel traces (beyond synthetic tiles). • Integrate with **actual transmitter control interfaces** to vary PLP/FEC parameters in real time at broadcast side. • Extend integrity/assurance: explicit integrity messaging, fail-safe policies, and robustness to telemetry loss. • Standardization alignment: specify data models for bitmap tiles and intent→control mappings for interoperability.
References	[1] "ATSC A/321:2023-03 — System Discovery and Signaling," ATSC (PDF), 2023. https://www.atsc.org/wp-content/uploads/2023/04/A321-2023-03-System-Discovery-and-Signaling.pdf [2] "ATSC A/322:2021 — Physical Layer Protocol," ATSC (PDF), 2021. https://www.atsc.org/wp-content/uploads/2021/04/A322-2021-Physical-Layer-Protocol.pdf [3] "ATSC A/331:2024-04a — Signaling, Delivery, Synchronization, and Error Protection," ATSC (PDF), 2024. https://www.atsc.org/wp-content/uploads/2025/01/A331-2024-04a-Signaling-Delivery-Sync-FEC.pdf [4] "A/360:2025-07 — ATSC 3.0 Security and Service Protection," ATSC (Document page), 2025. https://www.atsc.org/atsc-documents/3602018-atsc-3-0-security-service-protection/ [5] "3.0 Standards (ATSC 3.0 document library)," ATSC (Standards index), n.d.

	https://www.atsc.org/atsc-documents/type/3-0-standards/ [6] "An Overview of the ATSC 3.0 Physical Layer Specification," ResearchGate, n.d. https://www.researchgate.net/publication/290472101_An_Overview_of_the_ATSC_30_Physical_Layer_Specification [7] "RTKLIB — An Open Source Program Package for GNSS Positioning (Project Site)," RTKLIB, n.d. https://www.rtklib.com/ [8] "RTKLIB ver. 2.4.2 Manual," RTKLIB (PDF), 2013. https://www.rtklib.com/prog/manual_2.4.2.pdf [9] "tomojitakasu/RTKLIB," GitHub (source repository), n.d. https://github.com/tomojitakasu/RTKLIB [10] "Ntrip — Networked Transport of RTCM via Internet Protocol," BKG / IGS NTRIP, n.d. https://igs.bkg.bund.de/ntrip/ [11] "Ntrip Documentation (HTTP-based protocol for dissemination of DGNSS/GNSS streaming data)," ESA GSSC (PDF), 2004. https://gssc.esa.int/wp-content/uploads/2018/07/NtripDocumentation.pdf [12] "RTCM 3 Message List (RTCM 10403.x overview)," Use-SNIP Knowledge Base, 2023. https://www.use-snip.com/kb/knowledge-base/rtcm-3-message-list/ [13] "RTCMv3 Standard Logs (RTCM1005, etc.)," NovAtel OEM7 Documentation, n.d. https://docs.novatel.com/OEM7/Content/Logs/RTCMV3_Standard_Logs.htm [14] "RTCMv3 Messages (Supported / MSM overview)," Septentrio Customer Support, n.d. https://customersupport.septentrio.com/s/article/RTCMv3-messages [15] "3GPP TS 38.305 (ETSI delivery) — NG-RAN; Stage 2 functional specification of UE positioning," ETSI (PDF), v17.7.0. https://www.etsi.org/deliver/etsi_ts/138300_138399/138305/17.07.00_60/ts_138305v170700p.pdf

6. Implementation proposal (Build-a-thon Stage-2 proposal)

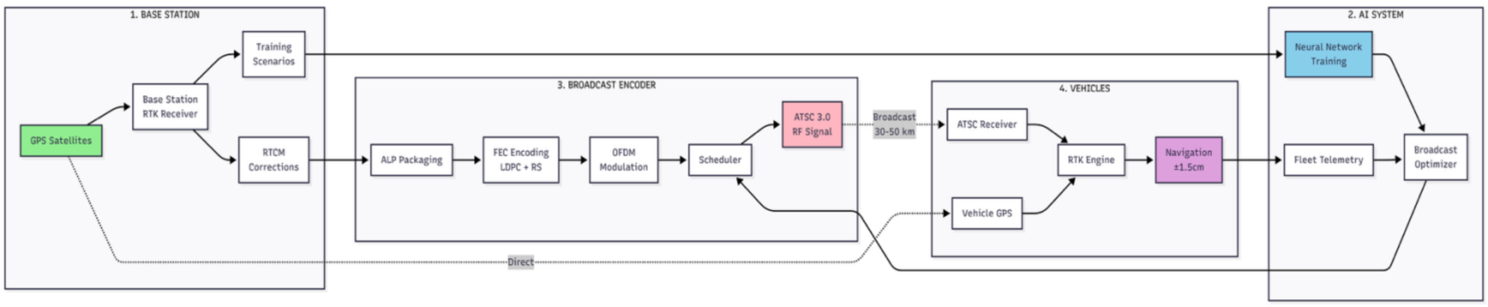
This clause describes the implementation proposal for the submission above. Implementation is done in the following phases. Phase-1 includes preparation and demo of the test setup. Phase-2 includes a demo of (opensource) base code in as-is-where-is condition. Phase-3 includes modification of the base code and test setup to suit the requirements of the demo mentioned in clause-1. Phase-4 includes a documentation of results corresponding to the test cases executed.

6.a) Description of the test setup

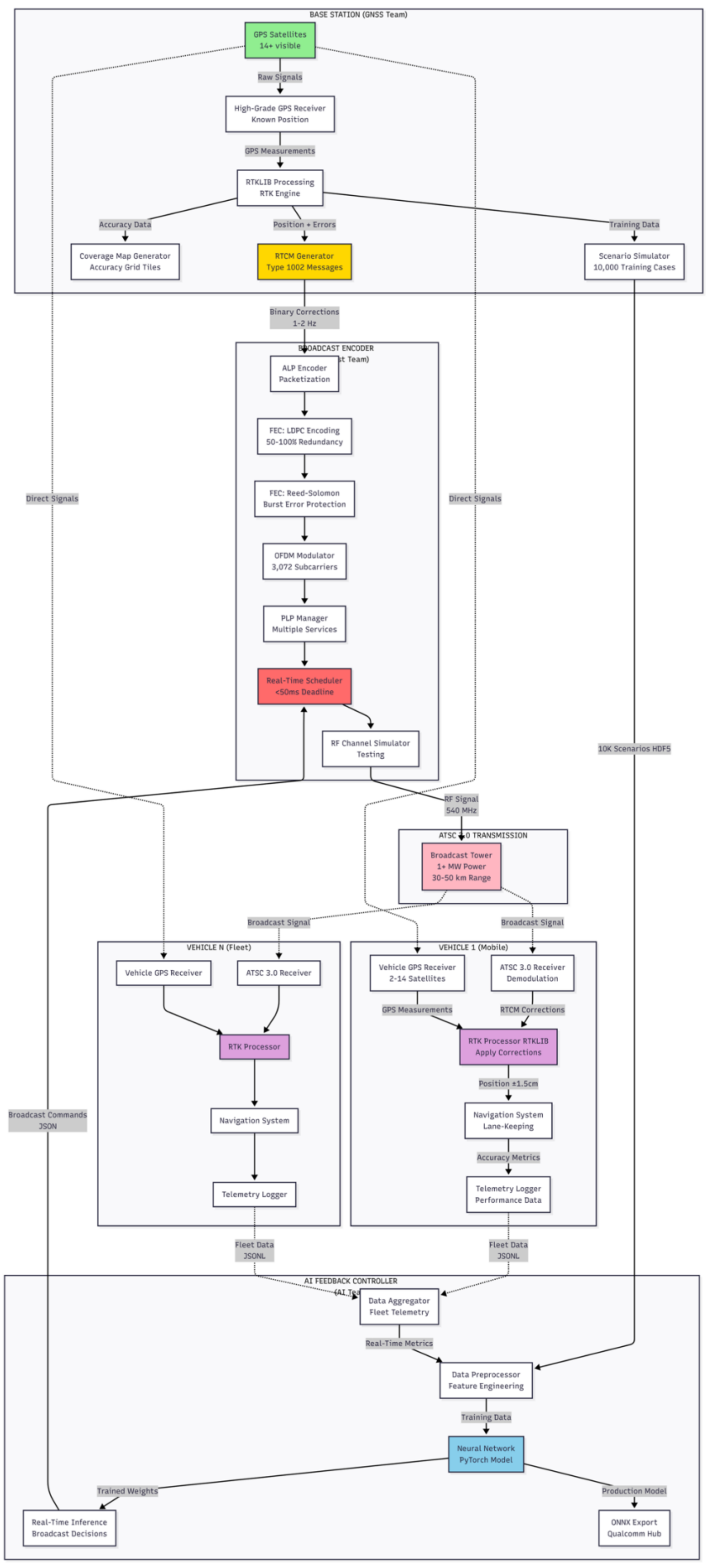
The test setup involves the following:

- **Code generation model from**:
  N/A for runtime (implementation is engineered in Python). Optional developer productivity tools (e.g., GitHub Copilot / ChatGPT) may be used during coding, but **are not part of the PoC runtime, evaluation, or control loop**.
- **Service orchestrator from**:
  A **custom Intent → Policy Orchestrator (Python)** that (i) ingests receiver/fleet telemetry, (ii) applies intent and policy logic, and (iii) emits broadcast configuration updates (e.g., correction update rate, robustness/redundancy level, bitmap/tile granularity, prioritisation rules). The orchestrator runs as a standalone service (Dockerised) and persists decisions + metrics for evaluation.
- **Agent framework from**:
  A **lightweight modular "controller + adapters" framework (Python)** implemented within the repo (no external MAS framework required). Modules are separated as: Telemetry Ingest, KPI Computation, Policy Engine (intent-driven), Broadcast Config Generator, and Experiment Runner (baseline vs adaptive).
- **Simulator from**:
  A controlled **positioning + impairment simulation/testbed** comprising:
  1. **Receiver/vehicle positioning emulator** (open-sky vs urban-canyon vs tunnel-like outage profiles), producing FIX/FLOAT-like states, error indicators, and correction age/validity signals;
  2. **ATSC 3.0 delivery emulation** (broadcast datacast pipeline abstraction) to model correction/assistance packaging and configurable robustness/overhead;
  3. **Experiment harness** to run baseline (static policy) and AI-adaptive policy runs under identical impairment traces and compare KPI time-series.
- **Datasets from**:
  Synthetic and replayable scenario traces generated for reproducibility, including:
  - tunnel-like GNSS outage windows (signal loss / reacquisition),
  - urban-canyon multipath / NLOS degradation profiles,
  - open-sky stable segments (control periods), and
  - correction/assistance payload logs (RTK-style / RTCM-like streams and bitmap/tile assistance payload variants).
    All scenario traces and configuration files used for the demo are stored/versioned in the GitHub repo under the dataset/scenario directory structure.

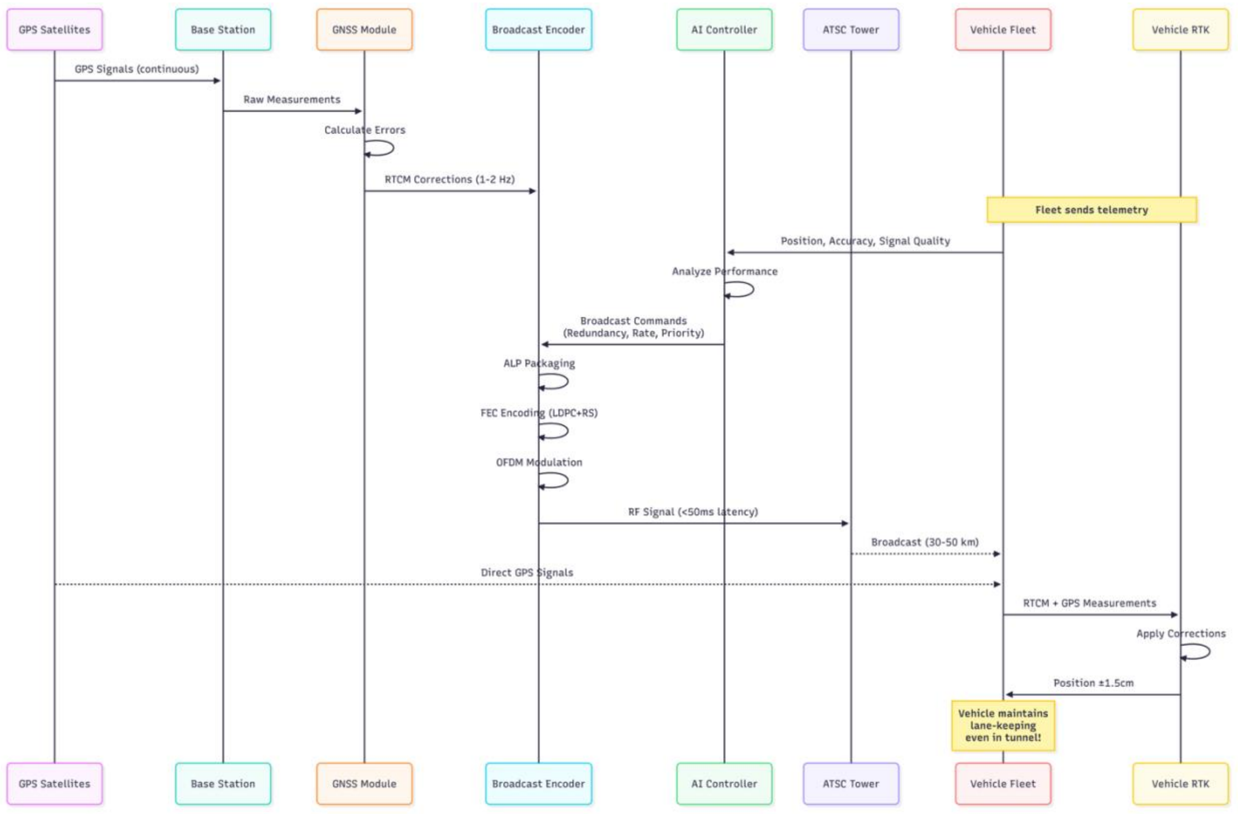
The following figure shows the high level architecture in the test setup:-



The complete system architecture is as follows:


BASE STATION (GNSS Team)
GPS Satellites 14+ visible
Raw Signals
High-Grade GPS Receiver Known Position
GPS Measurements
RTKLIB Processing RTK Engine
Accuracy Data
Position + Errors
Training Data
Coverage Map Generator Accuracy Grid Tiles
RTCM Generator Type 1002 Messages
Scenario Simulator 10,000 Training Cases
Binary Corrections 1-2 Hz
BROADCAST ENCODER
t Team)
ALP Encoder Packetization
FEC: LDPC Encoding 50-100% Redundancy
FEC: Reed-Solomon Burst Error Protection
OFDM Modulator 3,072 Subcarriers
PLP Manager Multiple Services
Real-Time Scheduler <50ms Deadline
RF Channel Simulator Testing
Direct Signals
Direct Signals
10K Scenarios HDF5
RF Signal 540 MHz
ATSC 3.0 TRANSMISSION
Broadcast Tower 1+ MW Power 30-50 km Range
Broadcast Signal
Broadcast Signal
VEHICLE N (Fleet)
Vehicle GPS Receiver
ATSC 3.0 Receiver
RTK Processor
Navigation System
Telemetry Logger
VEHICLE 1 (Mobile)
Vehicle GPS Receiver 2-14 Satellites
ATSC 3.0 Receiver Demodulation
GPS Measurements
RTCM Corrections
RTK Processor RTKLIB Apply Corrections
Position ±1.5cm
Navigation System Lane-Keeping
Accuracy Metrics
Telemetry Logger Performance Data
Broadcast Commands JSON
Fleet Data JSONL
Fleet Data JSONL
AI FEEDBACK CONTROLLER
(AI Tea
Data Aggregator Fleet Telemetry
Real-Time Metrics
Data Preprocessor Feature Engineering
Training Data
Neural Network PyTorch Model
Trained Weights
Production Model
Real-Time Inference Broadcast Decisions
ONNX Export Qualcomm Hub

The sequence diagram for the above architecture is as follows:-



6.b) Description & reference to base code

Base code used from:
Our own GitHub repository: https://github.com/Rishi8520/ai-positioning-atsc3.git.

It has the following components:
(i) RTK correction generation / ingestion interface (RTCM-like), (ii) ATSC 3.0 assistance payload packaging and scheduling layer (emulated or tool-backed), (iii) vehicle-side RTK application pipeline (RTKLIB-based), (iv) fleet telemetry logger and aggregator, (v) intent-driven policy engine (rules baseline + ML model), and (vi) evaluation harness for baseline vs AI-native comparison.

Important for the demo:
The intent interface, telemetry aggregation, policy engine that selects broadcast/assistance parameters, and the KPI evaluation harness (ΔHPE_p95, RTK FIX%, TTR, ΔBroadcast_Overhead%) are the core demo-critical elements.

Used as-is:
Upstream libraries/tools (e.g., RTKLIB binaries/APIs, Python ML libraries such as PyTorch/ONNX runtime, and any ATSC 3.0 toolchain components used for packaging/modulation or emulation) will be used as-is wherever possible, with only configuration-level tuning.

Changed/implemented as below:
Because the repository starts empty, the primary 'changes' in this PoC are net-new modules and integration glue: (a) telemetry schema + real-time aggregator, (b) intent-to-policy translation logic, (c) adaptive control that tunes update rate/redundancy/payload policy based on observed fleet outcomes, and (d) baseline vs adaptive experiment runner and reporting.

Justification:
These additions are required to demonstrate AI-native, closed-loop behaviour for a non-radio broadcast-assisted positioning service: decisions must be outcome-driven (fleet telemetry), resource-aware (broadcast budget), and verifiable through measurable KPI deltas.

6.c Mapping to the demo proposal

Functional and performance requirements corresponding to the demo

- The demo will show an end-to-end loop where GNSS/RTK corrections and positioning assistance are delivered over an ATSC 3.0 broadcast pipeline and adapted using an intent-driven controller, based on observed fleet positioning outcomes. The functional and performance requirements below are directly mapped to the demo steps and the KPIs defined in Clause-2 (Positioning Accuracy, Reliability, Correction Delivery Latency, Spectrum Efficiency, Service Continuity).

Any datasets needed

- The PoC uses (i) simulated/controlled GNSS impairment scenarios (open-sky baseline, urban-canyon multipath, tunnel-like outage), (ii) RTK/RTCM correction streams generated from a reference/base-station processing chain, and (iii) fleet telemetry logs containing positioning state and quality indicators (FIX/FLOAT, horizontal error statistics, correction age, outage duration, packet loss). A minimum dataset will be generated during the PoC runs and stored in the repository as reproducible logs (CSV/JSON/HDF5 as applicable) for baseline vs AI-native comparison.

Toolsets including open source or proprietary

- Open-source toolsets are used for GNSS/RTK processing and for the AI/control implementation. The ATSC 3.0 pipeline is modelled at the scheduling/robustness/payload level sufficient for evaluation of broadcast budget vs positioning outcomes. The implementation artefacts, scripts, logs, and evaluation notebooks will be hosted in the project GitHub repository as mentioned above.

Test cases which correspond to requirements needs to be added at this stage

- The test cases below correspond to the functional requirements and directly measure the KPIs (accuracy, reliability, latency, spectrum efficiency, continuity) under identical impairment scenarios, comparing baseline static policies vs intent-driven adaptive policies.

Functional requirements are as below:

Req-1 (Critical): It is critical that the **end-to-end positioning assistance loop** is integrated in the test setup: **GNSS/RTK correction generation → ATSC 3.0 packaging and transmission → vehicle-side RTK processing → fleet telemetry return → AI broadcast optimiser decisions → updated broadcast policy applied back to the encoder/scheduler**. This is required so that the AI control loop can *measure outcomes (FIX/accuracy/continuity)* and *actuate broadcast configuration changes (rate/redundancy/priority/bitmap policy)* within the same experiment.

Req-2 (Expected): It is expected that **service intent templates / policy profiles** are defined and versioned in the project repository (e.g., *Maximise Accuracy*, *Maximise Reliability in Tunnels*, *Minimise Broadcast Overhead*, *Corridor Priority*). Each intent must map to concrete, configurable broadcast parameters such as **correction update rate**, **redundancy level (FEC**

configuration / repetition policy), **payload composition**, and **bitmap/tile resolution or prioritisation**.

Req-3 (Added value): It is of added value that the system performs **safety-style impact checking of broadcast policy changes** by comparing *baseline vs optimised* runs and ensuring that improvements in challenged zones (tunnel/urban canyon scenarios) do **not** cause unacceptable degradation in normal/open-sky scenarios. This is demonstrated by fleet-level deltas in **HPE_p95 / RTK FIX availability / time-to-recover (TTR)** while keeping **broadcast usage within a configured budget**.

Any datasets needed

- **Synthetic scenario dataset (HDF5):** 10,000+ simulated scenarios covering open-sky, urban-canyon multipath, and tunnel-like GNSS outage/attenuation events, with labelled corridor segments and impairment profiles.
- **Correction stream data:** RTCM correction messages generated from RTK processing (or replayed traces where applicable).
- **Fleet telemetry logs (JSON/CSV):** time-stamped **RTK solution state (FIX/FLOAT), estimated error (HPE), correction age/latency indicators, degradation events, and recovery times**, aggregated per trip / per corridor.

Toolsets including open source or proprietary

- **Open-source GNSS/RTK:** RTKLIB (correction generation / RTK engine integration).
- **ATSC 3.0 broadcast processing chain (testbed/simulation):** ALP packaging + FEC (LDPC + RS) + OFDM + scheduler/PLP configuration (implemented as test modules/simulated blocks as per PoC scope).
- **AI/ML stack:** Python + PyTorch for model training; ONNX export for deployment validation (as targeted in the project outline acceptance criteria).
- **Data handling and evaluation:** HDF5 tooling, plotting/metrics scripts for **HPE_p95, RTK FIX%, TTR, Broadcast_Overhead%**.
- **Repository:** GitHub project repository will be used for code, configs, datasets (or dataset generation scripts), and documentation: https://github.com/Rishi8520/ai-positioning-atsc3.git

Test cases which correspond to requirements

The test cases are as below:

1. **TST-1 (Baseline vs Intent-Driven in Tunnel Impairment):** The tunnel/outage impairment scenario is triggered (reduced satellite visibility / forced degradation window). Expected result: under the *"Maximise Reliability in Tunnels"* intent, the optimiser increases robustness (e.g., redundancy / prioritisation / update policy) such that **RTK FIX availability increases and TTR decreases** versus baseline, while maintaining centimetre-class performance where conditions permit.
2. **TST-2 (Resource Budget Compliance / Efficiency Trade-off):** A constrained broadcast budget is applied (configured payload/bandwidth ceiling). Expected result: under *"Minimise Broadcast Overhead"* intent, the system reduces **Broadcast_Overhead%** relative to baseline while keeping **positioning quality within acceptable bounds**(HPE_p95 and RTK FIX% not breaching policy thresholds). This demonstrates dynamic scaling instead of static over-provisioning.
3. **TST-3 (Corridor Prioritisation / Fleet-Level Optimisation):** A corridor-priority scenario is enabled (subset of route segments marked as high-risk or high-value). Expected result:

under *"Corridor Priority"* intent, the optimiser reallocates robustness/payload emphasis to the corridor, producing **measurable improvement in corridor metrics**(degradation events per trip/corridor reduce; outage-induced error spikes reduce) while keeping non-priority areas within baseline-acceptable performance.

Note 1: The references may include PoC reports, whitepapers, public repositories, newsletters or articles. The PoC can be public or private. In such cases, references such as published materials corresponding to the PoC may be added as reference as appropriate.

Note 2: PoCs from this report may be demonstrated or showcased in FG meetings or other collaborative events organised along with the FG.

Bibliography

[FGAINN intro] https://www.itu.int/en/ITU-T/focusgroups/ainn/Pages/default.aspx