1. Introduction

1.1 Purpose

This document provides the architectural and design details of the TelcoPulse real-time analytics system for monitoring mobile network metrics. The system processes real-time streams from telecom providers, derives insights, and visualizes them through a live dashboard.

1.2 Scope

The solution focuses on real-time ingestion, transformation, storage, and visualization of network performance data using AWS-native services. It includes:

- Real-time data ingestion via Amazon Kinesis
- Data transformation with AWS Glue and Spark Streaming
- Storage in Amazon S3
- Querying via AWS Athena
- Visualization using Streamlit on Amazon ECS

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition
KPI	Key Performance Indicator
ECS	Elastic Container Service
S3	Simple Storage Service
GPS	Global Positioning System
ETL	Extract, Transform, Load
SQL	Structured Query Language

1.4 References

- AWS Documentation
- Apache Spark Documentation
- Streamlit Documentation
- Project README

1.5 Overview

The document outlines a modular architecture using AWS-native services to ensure scalability, reliability, and low-latency analytics on mobile network metrics.

2. Architectural Goals and Constraints

2.1 System-wide Design Decisions

- Real-time streaming architecture
- Serverless and fully managed services where possible
- Decoupling components for maintainability and scalability

2.2 Architectural Constraints

- AWS-only implementation
- Near real-time latency (<5 minutes)
- Autoscaling components to support variable input loads

3. Use-Case View

3.1 Use-Case Model

Actors:

- Data Engineering Team
- Network Operations Analysts

Use Cases:

- Monitor signal strength trends
- Identify network outages by postal code
- Analyze GPS accuracy by operator

4. Logical View

4.1 Major Logical Components

- **Data Ingestor** (Amazon Kinesis)
- **Stream Processor** (AWS Glue with Spark)
- **Data Lake** (Amazon S3)
- **Data Catalog** (Glue Crawlers)
- **Query Engine** (AWS Athena)
- **Dashboard UI** (Streamlit on ECS)

5. Process View

5.1 Concurrent Processes

- Kinesis continuously streams data
- Spark Streaming processes events in micro-batches
- ECS runs containerized dashboard with auto-refresh logic

6. Implementation View

6.1 Development Environment

- Python 3.x for Spark and Streamlit
- AWS SDK / Boto3
- AWS CLI, Terraform

6.2 Configuration Management

- Git for source control
- Versioning of ETL jobs and dashboards via tags/branches

7. Data View

7.1 Storage & Retrieval

- Raw and processed data in S3

- Athena queries provide real-time metrics to dashboard

7.2 Data Integrity and Security

- S3 bucket policies for access control
- IAM roles for Glue, Athena, ECS
- Encryption at rest and in transit enabled

8. Size and Performance

8.1 Performance Benchmarks

- Data latency: < 5 minutes from ingestion to visualization
- Throughput: Scalable with stream size via Kinesis shard scaling

8.2 Scalability Constraints

- Number of Kinesis shards
- Glue job parallelism
- ECS service auto-scaling settings

9. Quality

Attribute	Strategy
Usability	Interactive, auto-refresh dashboard UI
Reliability	Managed services with fault tolerance
Performance	Micro-batch processing and tuning
Maintainability	Modular service design
Security	IAM, encryption, private networking

10. Appendices

10.1 Revision History

Date	Version	Description
2025-05-15	1.0	Initial draft generated