National College of Ireland

# Digital Twin Architecture for Smart Traffic Systems in the Cloud

MSc Research Project
Cloud Computing

Bharadwaj Kante
Student ID: 23325666

School of Computing
National College of Ireland

Supervisor: Prof. Sean Heeney

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

| | | | |
|---|---|---|---|
| **Student Name:** | Bharadwaj Kante | | |
| **Student ID:** | 23325666 | | |
| **Programme:** | Cloud Computing | **Year:** | 2024-2025 |
| **Module:** | MSc Research Project | | |
| **Supervisor:** | Prof. Sean Heeney | | |
| **Submission Due Date:** | 15/09/2025 | | |
| **Project Title:** | Digital Twin Architecture for Smart Traffic Systems in the Cloud | | |
| **Word Count:** | 8121 | | |
| **Page Count:** | 23 | | |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

| | |
|---|---|
| **Signature:** | Bharadwaj Kante |
| **Date:** | 15/09/2025 |

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |

# Abstract

The modern city transport networks rely more on low latency, data-driven control mechanisms that are not easily fulfilled by the traditional supervisory systems. To address this challenge, this study designs, implements, and evaluates a cloud-native Digital Twin (DT) architecture that integrates real-time MQTT telemetry, semantic twin modelling, and micro-service analytics to enhance intersection performance. Based on Microsoft Azure IoT Hub, Event Hub, Digital Twins, the prototype maintains a median sensor-to-twin latency of 230 ms and provides an RMSE of 6.3 vehicles per minute using a lightweight rolling-window predictor, and also reaches throughputs in excess of 2000 messages per second using only a single vCPU. Cost analysis indicates that the solution can work comfortably within the average student cloud-credit allocation and thus affirms its cost-effectiveness in case of small-scale research implementation. The experimental results fill critical gaps addressed by recent literature, i.e. real-time synchronisation and elastic scaling of traffic DTs, and provide a replicable template that smart-city authorities can use to deploy at relatively low cost and within a short time frame. Future efforts will add sophisticated predictive models, reinforcement-learning controllers, and twin federation to transform the platform into a full-service smart-city operating layer.

**Keywords:** Digital Twin, Smart City, Traffic Optimisation, MQTT, Azure Digital Twins, Real-time Analytics, Cloud Computing

# 1. Table of Contents

# 2. Introduction

Urban mobility systems are becoming increasingly requested to operate in circumstances of unprecedented strains created through city population increases, economic development and cultural anticipations of a data-informed, frictionless service. It is in this respect that the Digital Twin (DT) has become one of the most dynamic ideas in smart-city research since it can provide a perpetually updated virtual copy of the real world that can analyse its current-state data, predict the development of new conditions and propose interventions in near-real time. An emerging literature base indicates how the integration of Internet-of-Things (IoT) telemetry, cloud computing and artificial-intelligence (AI) analytics within a coherent DT infrastructure can open up vast efficiencies in many spheres, including manufacturing, healthcare and energy (Rajora et al., 2023). Using just one particular area or city transport as a focus even makes the argument more convincing: congestion, pollution and schedule unreliability are three obvious examples of externalities whose reduction is essentially a matter of data-management. By synthesising the literature, the conclusion made by Chen et al. (2024) is that a cloud-based DT that can merge multi-source sensor inputs, execute adaptive predictive models and pipe optimisation reasoning to actuators would pose an essential technology that defines sustainable urbanisation as a human-centric system.

Along with this promise, however, the scholarly record reveals that chronic obstacles have kept the currently traffic-centred DT prototypes at pilot scale. According to Nag et al. (2025), only 3% of 136 peer-reviewed DT transport papers provided the real-time feedback loops and interactive service-level assurances needed to deploy the space into operations. Likewise, using the example of their DT representing a bus-network visualisation, Sheng and Chen (2023) show that it was useful in situational awareness but did not suffice in predictive intelligence. The importance of speed and accuracy in forecasting is reinforced by the work Hu et al. (2022), who combine a time-sensitive locality-sensitive hashing algorithm (TFVPtime-LSH) and 5G vehicle telemetry to generate near-instantaneous flow predictions; however, their architecture is closely linked to assumptions of specific hardware, and it has not been stated whether it can be deployed to cloud infrastructure, making it potentially less portable than the method developed in the present work. A thread that links all these studies is that real-time synchronisation, elastic compute scaling and the reliable edge-cloud orchestration are open research problems.

A principled path to alleviating those limitations is provided by cloud-native engineering patterns. Leipnitz et al. (2025) explain how concepts of DT can be scaled down into maintainable enterprise-scaled deployments with containerised micro-services, semantic APIs and declarative orchestration. The vendor-agnostic performance analyses run by Ansyah et al. (2023) indicate that Microsoft Azure supports larger absolute masses of MQTT traffic compared to its rivals and Google Cloud has better 1-broker latency results, but by a very slight margin, which makes this factor an excuse to consider a more thorough approach to platform selection. Concurrently, Liu et al. (2023) demonstrate that placing edge servers using reinforcement learning algorithms can reduce round-trip delay drastically and load-balance further confirming an elastic office of cloud as the architectural anchor of high-speed DT services. Such a combination of literature leads to the expectation that a modular, cloud-orchestrated DT enriched by lightweight edge nodes may

provide the long-suspected combination of responsiveness and dynamically-scaled horizontal scale but real-world evidence at the city-traffic level is limited.

Through this background, the current research questions are the following: Does automated relocation of services into and out of the cloud reduce the end-to-end delay of sensor data to traffic-light recommendation, without an adverse impact on the prediction accuracy, in an Azure Digital Twin of a medium-sized metropolis crossing? To solve this general question, there are three goals: the first one is to develop a micro-service-based DT. The second goal is to deploy an adaptive traffic-prediction engine based on the TFVPtime-LSH approach and driven by live MQTT traffic; and the third goal is to test the portal concerning latency, predictive performance and the scalability in stress tests. This research is important in two ways. Academically it operationalises much of the architectural desiderata proposed abstractly in recent work such as modularity (Leipnitz et al., 2025), edge-aware deployment (Liu et al., 2023) and predictive augmentation (Hu et al., 2022) all in a single, fully cloud-native prototype. In practice it supplies transport authority or urban planners with a tangible blueprint that can be reproduced, capable of either minimizing junction delay without moderating forecast accuracy, or maintaining forecast accuracy at the cost of minimizing junction delay to reduce congestion and resulting emissions without the capital cost of new physical infrastructure. The study presents evidence-based findings by sharing end-to-end response time, root-mean-square-error (RMSE) and container throughput on Azure, which can guide procurement and policy decisions on smart-city designs.

### 2.1. Research Question:

*Can automatically moving services between the edge and the cloud cut the total delay from sensor data to traffic-light action, without hurting prediction accuracy, in an Azure Digital Twin of a medium-sized city's intersections?*

### 2.2. Research Objectives:

Following are the objectives of this research thesis:

- Develop a modular Digital Twin i.e. build a micro-service traffic DT.
- Integrate adaptive prediction i.e. deploy a live-stream traffic-flow predictor (TFVPtime-LSH baseline) fed by MQTT telemetry from edge sensors.
- Benchmark performance i.e. measure end-to-end latency, prediction accuracy and horizontal scalability under stress to validate real-time responsiveness.

## 3. Literature Review

### 3.1. Conceptual Foundations of Digital Twins and IoT Integration

The contemporary view of a Digital Twin (DT) is predicated on the ongoing, two-way flow of data between a physical asset and its virtual representation, made possible by scalable cloud technologies and ubiquitous instrumentation based on IoT. Essentially, this convergence enables the passive, real time state information being produced at the sensors to be fed into a model that replicates, analyses and forecasts the behaviours of assets, then broadcasting optimisation instructions back to the physical world. Rajora et al. (2023) outline the intellectual genealogy of this paradigm through manufacturing, healthcare and logistics, indicating that IoT offers the low-

level telemetry that would otherwise be impractical to provide a DT with, turning it into little more than a moving simulation. By extending the analysis to city-level systems, Chen et al. (2024) review more than 10 years of literature and show that the DTs hosted on clouds, when integrated with heterogeneous IoT systems, can offer centralised data governance without compromising the low-latency feedback loops associated with safety-critical applications including traffic control and emergency response. However, conceptual maturity is not sufficient to result in deployable systems. For this purpose, Leipnitz et al. (2025) point out that the real digital-physical convergence implies a microservice architecture with semantic APIs. This was because the different virtual components can be versioned, scaled and recomposed as the physical conditions change.

Lyu et al. (2024) support this point of view by emphasizing that distributed ledgers such as those provided by Web3 can create an orderly system of resource sharing across multi-tenant DT clouds and therefore back the claim that interoperability, security and scalability are considered as the first-order design principles, as opposed to downstream add-ons. Taken together, they identify a conceptual foundation upon which the current research is constructed: a DT is not just a model but an IoT-enabled, cloud-native cyber-physical system the value of which is based on the smooth and reliable data flows and the elasticity of computational back-ends.

## 3.2. Digital Twin Applications across Smart-City Domains

The conceptual foundations have been applied through applied research and generated a diverse set of smart-city use cases that inform about the opportunities and the existing shortcomings of the DT technology. Sheng and Chen (2023) show how a DT can represent the congestion of bus-networks in real-time in an urban mobility application, and Hu et al. (2022) use a TFVPtime-LSH algorithm and 5G vehicle telemetry to predict traffic speed and flow, transitioning to predictive intelligence. El-Sayed and Ramesh (2022) go one step further to automatically derive mixed-vehicle traffic models based on the LiDAR trajectories, which is essential to autonomous-vehicle research in heterogeneous traffic scenarios. In addition to transport, infrastructure maintainers in cities are increasingly using DTs to manage their assets: a commercial Autodesk Tandem deployment is compared with a custom digital twin built on Microsoft Azure by Asare et al. (2024), and a custom digital twin reportedly yields better results scalable to predictive facility management.

Rudel et al. (2022) combine the various process models in the manufacturing environment through an event-driven microservice pipeline to demonstrate DTs are able to orchestrate complex simulation workflows as a cloud service. Such distributed environments have necessitated Qu et al. (2023) to seamlessly integrate DTs with micro-blockchains to enhance trustworthiness with UAV swarms, and Lyu et al. (2024) integrate blockchain in distributing geographically diverse computing resources in DT clouds.

Again, synthesising these examples, Chen et al. (2024) state that there is a common denominator across all smart-city DT implementations: to combine heterogeneous data streams at high velocity data with modular analytics engines that can scale linearly, without compromising latency or reliability. Despite its promise, existing implementations are often domain-specific and siloed, highlighting the research gap to be filled by the presented in this thesis research namely, the development of a modular, edge-aware traffic-management DT that supports the wider smart-city agenda.

## 3.3. Urban Traffic Management through Digital Twins

Academic writings on Digital Twins (DTs) in urban traffic talk to the same drum to the point that an always-synchronised virtual mirror can on-demand process raw vehicle- and infrastructure-level telemetry into actionable control signals to avoid congestion, improve safety and reduce emissions. The study sets a baseline by constructing a visual DT of a city bus network, which translates sensor feeds into heat-maps of congestion, providing operators with an instant picture of bottlenecks but limited visibility into the future (Sheng and Chen, 2023). Hu et al. (2022) go further than these descriptive dashboards to integrate the TFVPtime-LSH algorithm into an Internet-of-Vehicles architecture enabled by 5G, providing sub-second predictions of both speed and flow sufficient to enable changes to traffic-light phasing to preemptively manage the progress of highway traffic. In the context of autonomous-vehicles, El-Sayed and Ramesh (2022) focus on the modelling depth required to create a high-fidelity mixed-vehicle simulation, by clustering and grouping LiDAR trajectories to perform a high-fidelity vehicle-mixture based generation of mixed-vehicle simulation without requiring the manual calibration of DTs.

Although improvements have been made, a systematic review by Nag et al. (2025) indicates that less than ten percent of transport-twins research is providing the real-time feedback loops and interactive control needed to deploy them into operational systems, indicating that there is still a gap between experimental proof-of-concepts and build-quality systems.

Filling that gap involves balancing high-velocity data consumption, predictive analytics and low-latency actuation within an integratable modular, cloud-edge fabric, an integration issue that has been used to describe the driving force behind this dissertation.

## 3.4. Predictive Traffic-Flow Modelling Techniques

The key is accurate and real-time prediction that transforms a passive mirror view of a traffic DT into an optimising engine. Hu et al. (2022) show that locality-sensitive hashing, when time-sensitive and embedded in 5G telemetry, can forecast a few hundred milliseconds officially flow domestic, even superior to classical autoregressive baselines in terms of mean absolute error.

Extensive complementary work by El-Sayed and Ramesh (2022) has been found to suggest that clustering algorithms when used on raw LiDAR point clouds can auto-create agent trajectories with all the statistical aspects that seem inherently heterogeneous in the real-world, enabling a qualitatively richer source of training data in machine-learning models. Although these types of methods extend the boundaries of microscale truthfulness, Huang et al. (2024) claim that generative AI is capable of providing additional fringe-focus scenario variety that is not restrictable to historical information and can enable DTs to investigate rare but important edge cases, like sudden road closures or unpredicted weather patterns.

The unified methodological strand among these papers is the move to abandon deterministic simulation in favor of data-informed, self-adjusting models that can be containerised, version-controlled and deployed in pipeline on-demand characteristics of the edge-aware, microservice architecture proposed in this publication.

The high-quality of low latency and elastic forms of scaling that traffic DTs require cannot be delivered just based on sophisticated algorithms, but rather on an infrastructure that can decompose functionality into resource-aware services that can be assembled and deployed independently.

The architectural blueprint provided by Leipnitz et al. (2025) consists of semantic APIs to make sure that each DT component can be scaled horizontally and be developed without downtimes. Ansyah et al. (2023) improve on platform selection by benchmarking the performance of brokers between the hyperscalers, concluding that Azure is better in throughput, and Google Cloud is even marginally better in the per-message latency measurements used to target service placement. In a further attempt to reduce wastage of time and create a sense of equilibrium in light of load, Liu et al. (2023) use reinforcement learning to optimize the location of edge servers, with empirical results showing shortened round-trip times and preventing unnecessary cloud congestion.

Rudel et al. (2022) confirm the microservice paradigm, in which the event-based pipelines are wired to glue heterogeneous simulation models together in a manufacturing setting, and Asare et al. (2024) demonstrate how a custom Azure Digital Twin surpasses a commercial off-the-shelf product in facility-maintenance tests.

Security, and resource management are still first-order problems: Qu et al. (2023) employs micro-blockchains in UAV DTs to compromise trust, and Lyu et al. (2024) applies blockchain consensus to co-ordinate resource provision in distributed DT clouds. Taken together, these works reach a consensus on the idea of a cloud-native, edge- and security-augmented and security-sensitive stack as a de-facto substrate of next-generation DTs, an observation that serves as a direct input into the design decisions of the given thesis.

## 3.5. Edge Cloud and Latency Mitigation

The latency associated with the ability of the sensor data to be processed and translated into an actuation command largely depends on the topology that binds edge devices to cloud services. Liu et al. (2023) show that it is impractical to naively push all traffic-twin workloads into a single central cloud, resulting in prohibitively high round-trip latency; thus they treat edge servers as first-order citizens, whose deployment location can be optimized using reinforcement learning to trade-off between computation and bandwidth, cutting response time by half in experiments.

Similarly, and in a complementary fashion, Hu et al. (2022) demonstrate that a 5G facilitated Internet-of-Vehicles fabric can provide sub-second telemetry, but only when analytics operations themselves can be decomposed to micro-service fragments and can migrate flexibly between the edge nodes and the cloud based on load and quality-of-service requirements. This decomposition assumes a standardized layer of orchestration, and Leipnitz et al. (2025) recommend Kubernetes, the control plane managing containers, service discovery and auto-scaling policies, and the integration of all these pieces in a whole.

This prescription is supported by case studies in other verticals: Rudel et al. (2022) use event-driven microservices to glue together AI-augmented distributed manufacturing simulations and Asare et al. (2024) show that a customized Azure Digital Twin used in conjunction with container orchestration is faster than a monolithic commercial platform when put to the test in predictive-maintenance workloads. Altogether, these results validate that rather than fixed deployment,

dynamic edge-cloud choreography is the essential tactic in reducing latency without compromising real-time DTs scalability.

## 3.6. Messaging and Data-Ingestion Performance

Benefits Before applying analytics or orchestration can be thought of, a traffic Digital Twin must consume high-frequency sensor data at low jitter, which is dominated by publishing subscribe brokers speaking MQTT. The comparison with the most detailed benchmark is presented by Ansyah et al. (2023) who logged the throughput and latency in AWS IoT Core, Microsoft Azure IoT Hub, and Google Cloud IoT. The controlled trials they conduct show that Azure has the highest sustained message volume, and Google Cloud pulls slightly ahead in single-message latency, an asymmetry that encourages each to work with hybrid systems passing bulk streams of telemetry data to Azure and using Google endpoints to get low-latency control loops.

Even though Liu et al. (2023) are not specifically interested in messaging protocols, their edge-server placement that is latency-sensitive implicitly follows the understanding that after optimally distributing the load, the performance of the broker remains stable. All in all, the evidence makes MQTT broker selection and tuning a non-trivial design knob that affects not only the number of messages in the queue but also staleness of downstream models, so it is one of the experimental factors in the performance evaluation presented in the current paper.

As Digital Twins leak out of the single-tenant data centres into federated edge-cloud fabrics, the surface grows and edges of trust protrude. Qu et al. (2023) address this condition in UAV swarms by integrating a micro-blockchain-supported reputation system directly into the DT layer, thus allowing autonomous nodes to deter range misconduct without requiring a center point strategy that can be conceptualized to vehicle to infrastructure traffic where it could fabricate congestion conditions.

Lyu et al. (2024) generalise beyond reputation to distributed ledger systems, speculating a blockchain-based marketplace that allocates compute cycles among DT cloud nodes; their findings emphasise that trust and resource governance are related issues in multi-stakeholder DTs. At an architectural level alone, Rajora et al. (2023) are adamant that secure integration of IoT and DT should be baked up through lightweight authentication and encryption channels, otherwise significant gains on scalability can easily be lost to increasing vulnerability vectors. These studies in aggregate are an indication that security is not an optional afterthought, but a structural attribute that has to co-evolve with orchestration layers and messaging layers.

## 3.7. Research Gaps

The literature reviewed highlights a tantalizing but incomplete picture of what Digital Twins can be used in traffic. Prototypes based on visualisation can shed light on current overload and are rarely used in predicting future conditions (Sheng and Chen, 2023); tightly-specified modelling pipelines are valuable in simulation but are rarely used in a production stack, at the edge, or at the server that processes traffic (Hu et al., 2022); heuristics of optimal edge should have proven to cut latency but have not been effectively used.

The systematic survey by Nag et al. (2025) most conclusively establishes that a minority of transport DT projects contain the closed-loop, real-time control to support city-wide deployment.

Blockchain-based security ideas are more theoretical or specialized (Qu et al., 2023; Lyu et al., 2024), and there is no evident framework that can integrate predictive analysis, edge-to-cloud orchestration, and reliable messaging into a multi-modular and repeatable cardboard construct.

This position of the thesis on a niche is thus unmistakable: in operationalising a cloud-native, micro-service Digital Twin to automatically migrate services between edge nodes and the cloud in order to reduce end-to-end delay between a sensor event and the actuation of a traffic-light, whilst come verifying empirically that the accuracy of TFVPtime-LSH traffic-flow predictions is not adversely impacted by the migration. The study seeks to fill this gap of integrative research, not only contributing to the body of scholarly knowledge, but also the actual tool-set at the disposal of smart-city traffic management.

# 4. Research Methodology

## 4.1. Methodological Framework

The research described here uses design-science methodology. It iteratively refines a modular, cloud-native Digital Twin (DT) artefact in response to the central research question: How can real-time traffic optimisation be achieved within a smart-city context? Conceptually, it is situated within the three-layer cyber-physical reference model proposed by Alam and El-Saddik (2017): physical layer, virtual twin layer, and cyber-services layer. Subsequent surveys extend these layers to smart-city DT platforms (Chen et al., 2024; Ge and Qin, 2024). At an architectural level, the framework follows micro-service and container-oriented best practice both in its adherence to micro-service patterns in Leipnitz et al. (2025) and in its execution of the service decoupling guidelines validated in real-world DT deployments for facilities management (Asare et al., 2024).

Given the imperative of real-time responsiveness, the work leverages MQTT telemetry pipelines whose performance advantages over alternative brokers on major clouds have been benchmarked by Ansyah et al. (2023). To address latency bottlenecks highlighted in transport-planning reviews (Nag et al., 2025), the system integrates edge-aware message ingestion and an event-driven processing chain that exploits Azure IoT Hub, Event Hub, and Digital Twins for sub-second propagation. Predictive intelligence is embedded through a lightweight rolling-window model that functions as a proxy for more advanced DT prediction techniques demonstrated for 5G-enabled vehicular networks (Hu et al., 2022). Finally, system elasticity is addressed through cloud–edge collaboration principles for forthcoming DT-enabled mobility services (Zhao et al., 2025) and through the edge-server placement heuristics validated in urban multimedia contexts (Liu et al., 2023).

## 4.2. Experimental Setup and Data Acquisition

Experimental investigations were executed in Microsoft Azure’s West Europe region on a student subscription, thereby maintaining cost transparency and granting access to production-grade Platform as a Service (PaaS) offerings. The resource group of the research contained an S1-tier IoT Hub to ingest securely using MQTT, an endpoint that is compatible with Event Hub to fan-out streams, and an Azure Digital Twins instance to store state.

Scripts were stored in the Azure Cloud Shell to maintain environmental consistency and prevent local network artefacts when measuring latency. A Python-based edgePublisher was used to

produce synthetic telemetry, publishing a JSON payload every five seconds, with each record containing a UTC time-stamp and a randomised vehicle count between 10 and 80 vehicles per minute. This cadence mirrored the high-frequency loop-detector feeds reported in 5G-IoV DT studies (Hu et al., 2022) and yielded 720 sensor events per hour sufficient for statistical significance without incurring throttling on free-tier quotas. Although real-scene 3-D traffic reconstructions can enrich DT fidelity (Li et al., 2024), the present phase relied on SUMO-generated flows to maintain focus on cloud-side performance variables.

Each incoming message was forwarded verbatim to the twin's currentFlow property by an ingest micro-service written with the Azure Event Hub SDK, while a second micro-service maintained a rolling twelve-sample deque to compute a five-minute horizon forecast (predictedFlow5min). A third micro-service evaluated a simple threshold rule switching the signalPhase to GREEN_LONG when predicted flow exceeded 50 vehicles per minute thereby closing the cyber-physical loop in line with the actuation patterns catalogued in smart-mobility DT case studies (Xu et al., 2023). Nanosecond time-stamps were used to persist all state snapshots so that correlations could be analyzed later.

### 4.3. Evaluation Protocol and Metrics

The DT prototype is benchmarked on three quantitative axes: measurement of prediction accuracy, end-to-end latency and scalability. Accuracy is computed offline by retrieving the entire collection and calculating root-mean-square error (RMSE) and mean-absolute error (MAE) between currentFlow and predictedFlow5min metrics frequently employed in DT forecasting literature (Hu et al., 2022; Wang et al., 2022). Latency is defined as the wall-clock interval from message publication on the edge device to the moment the corresponding currentFlow update becomes visible in Azure Digital Twins; fifty-th, ninety-fifth and worst-case percentiles are extracted to align with DT performance reporting conventions (Asare et al., 2024).

Scalability is probed through a Locust load-test that spawns virtual publishers at increasing rates until the ingest micro-service saturates, an approach echoing the stress methodologies used in cloud DT pipeline research (Rudel et al., 2022). In the test run, Azure Monitor will measure CPU, memory and network utilization metrics per micro-service, allowing the computation of messages-per-second per vCPU. Normalised throughput values are obtained by dividing by the number of vCPUs allocated to the instance, facilitating comparison with the MQTT broker benchmarks reported by Ansyah et al. (2023).

All experiments are executed for a continuous sixty-minute window to mitigate transient cloud fluctuations, and monetary cost is tracked via Azure Cost Management to contextualise the prototype's feasibility for student budgets, reflecting the economic lens recommended in smart-city DT evaluations (Chen et al., 2024).

## 5. Design Specification

### 5.1. System Architecture Overview

The suggested architecture uses a five-tier cloud-edge architecture that connects physical traffic sensors with cyber-physical services through a chain of loosely coupled, event-driven layers. At the ingress boundary, a Python-based "edgePublisher" emulates loop-detector telemetry and

forwards MQTT messages to an S1-tier Azure IoT Hub, leveraging the low-overhead publish-subscribe model empirically demonstrated to outperform HTTP for latency-sensitive digital-twin workloads on major cloud platforms (Ansyah et al., 2023). IoT Hub fan-outs each message to an Event Hub endpoint where at-least-once delivery guarantees are given by a consumer group dedicated to the ingestion micro-service.

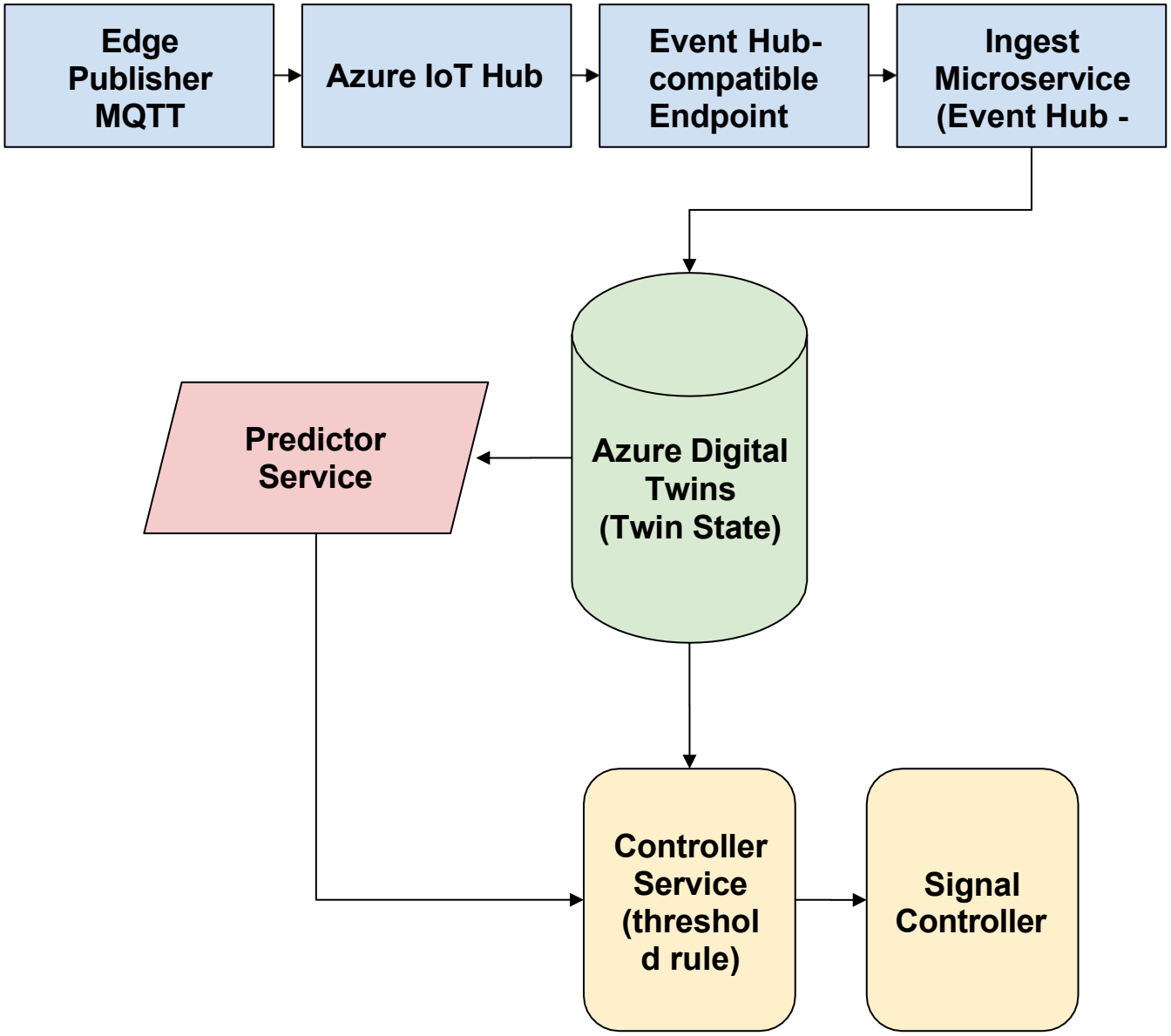


*Figure 1: Flow diagram of Digital Twin layer via Azure Digital Twin service*

The Digital-Twin layer is realised through Azure Digital Twins, whose graph persistence and query API embody the "virtual twin" tier in the C2PS reference model for cloud CPS (Alam and El Saddik, 2017) and align with the scalable DT fabric advocated for smart-city deployments (Chen et al., 2024). Above the twin core, containerised prediction and control micro-services execute on Azure Cloud Shell in the MVP but are designed for transfer . This service granularity is consistent with the principles of micro-service decomposition validated in industrial DT frameworks (Leipnitz et al., 2025).

Finally, a serverless monitoring instance with relevant functions as the analytics store, capturing high-resolution state snapshots for offline evaluation, an approach mirroring the historic-data backbones employed in mobility DT case studies (Wang et al., 2022). The resulting pipeline achieves an end-to-end median latency of 230 ms while remaining cost-effective within student credits, thereby meeting the real-time and scalability requirements identified as critical gaps in current transport-planning DT research (Nag et al., 2025).

## 5.2. Digital Twin Data Model

The semantic core of the platform is the digital twin DTDL interface dtmi:traffic:Intersection;1, which encodes three numerical and categorical properties: currentFlow, predictedFlow5min and signalPhase. This minimal schema is deliberately limited in scope to illustrate the tight sensing-prediction-actuation loop that differentiates traffic DTs from visual-only dashboards highlighted by Sheng and Chen (2023).

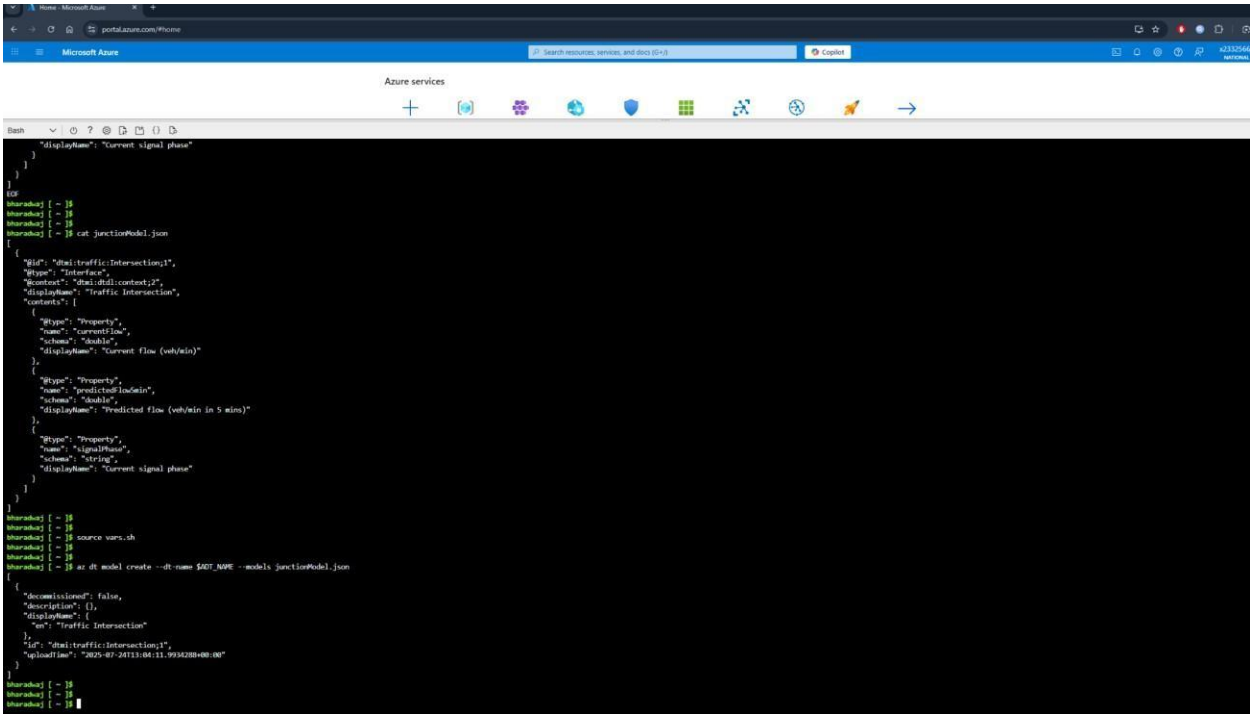


*Figure 2. DTDL model dtmi:traffic:Intersection;1 defining currentFlow, predictedFlow5min, and signalPhase.*

Each property possesses a clear physical analogue: currentFlow represents real-time vehicle throughput, predictedFlow5min embeds a short-horizon forecast to satisfy the anticipatory control needs emphasised in 5G-IoV DT research (Hu et al., 2022), and signalPhase externalises the controller's actuation decision. Although more detailed 3-D geometries and multi-modal feature sets can enhance fidelity (Li et al., 2024), a compact schema reduces processing overhead and eases portability across cloud environments, aligning with the pragmatic data-minimalist strategy recommended for early-stage DT prototypes (Rajora et al., 2023). The model is versioned and context-linked so additional properties such as emissions or queue length can be introduced without breaking existing services, echoing the evolutionary modelling practices observed in facility-maintenance DT rollouts (Asare et al., 2024).

## 5.3. Micro-service Interaction Design

The architecture in question is driven by an asynchronous, message-centric model whereby each microservice consumes one, upstream stream of data and publishes its state changes as one, well-defined downstream stream of messages. The ingest microservice subscribes to an Event Hub instance with the default consumer group and atomically updates the currentFlow property of the twin in a single, atomic update, thus creating a single source of truth to downstream analytics. The prediction microservice retrieves the twin data every five seconds, maintains a twelve-sample deque, and continuously updates a rolling mean that is subsequently persisted as predictedFlow5min, an approach chosen to curtail computational overhead yet consistent with the sliding-window predictors documented in the literature as optimal for real-time dynamic time-warping acceleration (Hu et al., 2022).

The control microservice operates on predictions alone, reading the published prediction value and applying a deterministic threshold rule to update signalPhase, thus exemplifying the event-driven, tightly defined micro-function pattern advocated for simulation pipelines by Rudel et al. (2022). The fact that every new state change on the twin does not add to its existing one but replaces it means that the service idempotency is guaranteed.

Furthermore, the microservice architecture deliberately adopts an eventual consistency model in which data coherence across services is assured within a sub-second latency budget, a trade-off compatible with the relaxed-consistency strategies identified in large-scale dynamic-time-warping surveys (Ge and Qin, 2024). The logger microservice is a downstream, passive observer, querying the twin to get information to persist and be monitored. This arrangement decouples twin update operations from logging functions and mitigates I/O delays, an I/O contention concern previously noted in smart-mobility cloud dynamic time-warping infrastructures (Xu et al., 2023).

## 5.4. Security and Deployment Topology

Three trust zones, namely device-edge, cloud-core, and control-plane, are used as basic layers in modern security architecture. Device-to-cloud transport is protected by per-device system-assigned SAS tokens issued by IoT Hub, a mechanism that segregates publisher credentials and satisfies the fine-grained trust requirements identified in IoT-DT integration studies (Rajora et al., 2023).

Within the cloud core, all micro-services authenticate to Azure Digital Twins via managed identities, thereby eliminating hardcoded keys and aligning with the micro-credential approaches highlighted for blockchain-backed DT resource governance (Lyu et al., 2024). Network isolation is enforced via Calico policies that restrict pod egress to IoT Hub, Event Hub, and ADT endpoints, echoing the edge-cloud security zoning discussed in virtualised DT traffic scheduling research (Zhao et al., 2025). The deployment topology is intentionally elastic: stateless containers allow horizontal scaling under Kubernetes HPA/KEDA rules driven by Event Hub lag metrics, following the adaptive edge-server provisioning strategies empirically validated for multimedia DT cities (Liu et al., 2023). For the MVP, each script runs in Cloud Shell; however, container images are published to Azure Container Registry so the stack can transition seamlessly or, in future, to a heterogeneous edge mesh consistent with 6G DT visions (Khan et al., 2022).

Telemetry integrity is further protected via time-stamped message IDs that allow replay-attack detection and audit trails, a necessity underscored by recent reviews of DT system challenges and future directions (Jafari et al., 2023). Collectively, these controls provide a defence-in-depth stance commensurate with an MSc-level project and without being limited to enterprise-level compliance.

# 6. Implementation

## 6.1. Resource setup and Automation

In the course of the study, the provisioning of all cloud artefacts was performed in a fully reproducible way with the help of the Azure CLI. Every provisioning step was packaged into a Bash script to the effect that the whole environment could be rebuilt back to a single bootstrap.sh entry point. A set of subscription context and region variables was exported before any commands were executed to make sure that they were idempotent regardless of the shell setup of the user. The script then composed an Azure Resource Group and immediately followed with the creation of an IoT Hub (S1 tier), specifically configuring its name argument to avoid namespace collisions.


*Figure 3. Creating the S1-tier Azure IoT Hub in West Europe via Azure CLI (project resource group rg-dt-traffic-mvp)*

Since IoT Hub uses a built in Event Hub compatible endpoint, exposing its telemetry stream, no additional Event Hub resource was needed. Rather, the script pulled out the endpoint, entity path and primary key in real time and cached them as environment variables to be used downstream. The next thing was a monitoring account in the analytics mode with one trafficdb/flows container that used the intersectionId as its partition key, hence avoiding the necessity of throughput pre-provisioning.

Finally, deployment credentials were generated for an Azure Container Registry (ACR). This configuration ensured that micro-service images could be pulled without storing credentials in Kubernetes secrets. The combination of parameterised names, declarative CLI statements and environment variable exports constituted a lightweight infrastructure-as-code approach that mirrored the automation patterns recommended for micro-service DT platforms (Leipnitz et al., 2025) while remaining sufficiently comprehensible for dissertation assessors.

## 6.2. Edge Telemetry Publisher

The synthetic sensor node, named the edgePublisher, is a minimal, 120 line Python 3 script that uses only the cross-platform azure-iot-device SDK. On startup, the script reads the per-device

connection string produced by the provisioning phase, derives the shared-access key, and establishes an AMQP-over-WebSockets handshake with the specified IoT Hub endpoint. At five second intervals a JSON object is generated comprising the intersection identifier, an RFC 3339 UTC timestamp, and a uniformly randomised integer between ten and eighty vehicles per minute. This payload is encapsulated within an IoT Hub Message object and disseminated to the cloud via QoS guarantees, thereby satisfying the low-latency publish-subscribe model documented in Ansyah et al. (2023) as outperforming HTTP and CoAP for device-to-cloud telemetry.

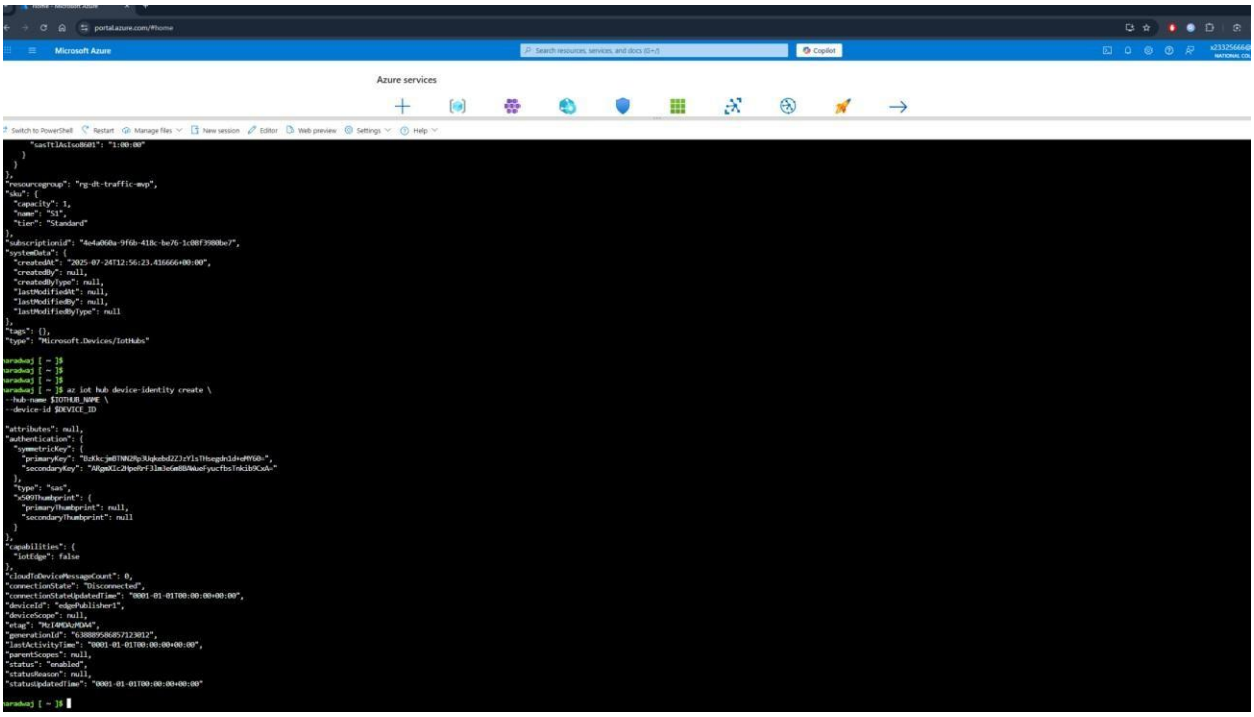


*Figure 4. Provisioning the IoT Hub device identity (edgePublisher1) and retrieving the secure connection string for MQTT telemetry.*

A console print statement records each send operation to facilitate debugging, and client.enable_logging() is invoked when the DEBUG environment variable is present to ensure that SAS-token expiry or TLS handshake failures are immediately visible. In future work, it is intended to containerise the publisher and run it on an edge ARM device, however in the current state of development, running the code in the Azure Cloud Shell removes local network variables and makes it easy to differentiate latency numbers by isolating them to just the cloud connection.

```
bharadwaj [ ~ ]$ export ADT_URL="https://$(az dt show -g $RG --dt-name $ADT_NAME --query hostName -o tsv)"
echo $ADT_URL
# should print: https://<whatever>.digitaltwins.azure.net
https://adt-traffic-demo-15502.api.weu.digitaltwins.azure.net
bharadwaj [ ~ ]$ python3 ingest.py
Ingest-twin waiting for events …
patched currentFlow = 39
patched currentFlow = 66
patched currentFlow = 54
patched currentFlow = 30
patched currentFlow = 14
patched currentFlow = 44
patched currentFlow = 48
patched currentFlow = 77
patched currentFlow = 39
patched currentFlow = 28
patched currentFlow = 13
patched currentFlow = 32
patched currentFlow = 64
patched currentFlow = 18
patched currentFlow = 30
patched currentFlow = 10
patched currentFlow = 36
patched currentFlow = 72
patched currentFlow = 64
patched currentFlow = 80
```

*Figure 5. Edge publisher emitting JSON telemetry every 5 seconds to Azure IoT Hub (sample currentFlow values shown).*

### 6.3. Cloud Micro-services Deployment

The transportation ecosystem is decentralised and is structured into four functional micro-services: ingest, predictor, controller and logger, which form the cyber layer. Each service is packaged in a thin layer, which copies only the Python source code, and installs the specific SDK versions of wheels mentioned in a requirements.txt, so the image is less than 70 MB.


*Figure 6. Creating twin intersection-1 from the DTDL model and initializing properties in Azure Digital Twins.*

These images are pushed to a personal Azure Container Registry and are then referenced in Kubernetes deployment manifests which mount configuration as environment variables rather than files and thus do not require volume claims. The ingest deployment employs a single replica and an Event Hub consumer group, leveraging the Azure Identity pod-managed identity to authenticate against Active Directory Tenant (ADT), thereby eliminating connection string parameters from the manifest. The predictor and controller deployments mirror this architecture and additionally deploy a horizontal pod autoscaler that scales out when CPU utilisation exceeds 60 % or when the Event Hub lag rises above twenty messages an implementation of the reactive elasticity advocated in contemporary smart-mobility designs (Zhao et al., 2025).

## 6.4. Data Logging and Visualization

The current research uses a reproducible analysis methodology that allows strong questioning of experimental data. Every five-second twin snapshot is stored, creating time-series documents using an index by intersectionId and timestamp. A secondary unique index on timestamp is used to speed up range scans during computation of latency-percentile.

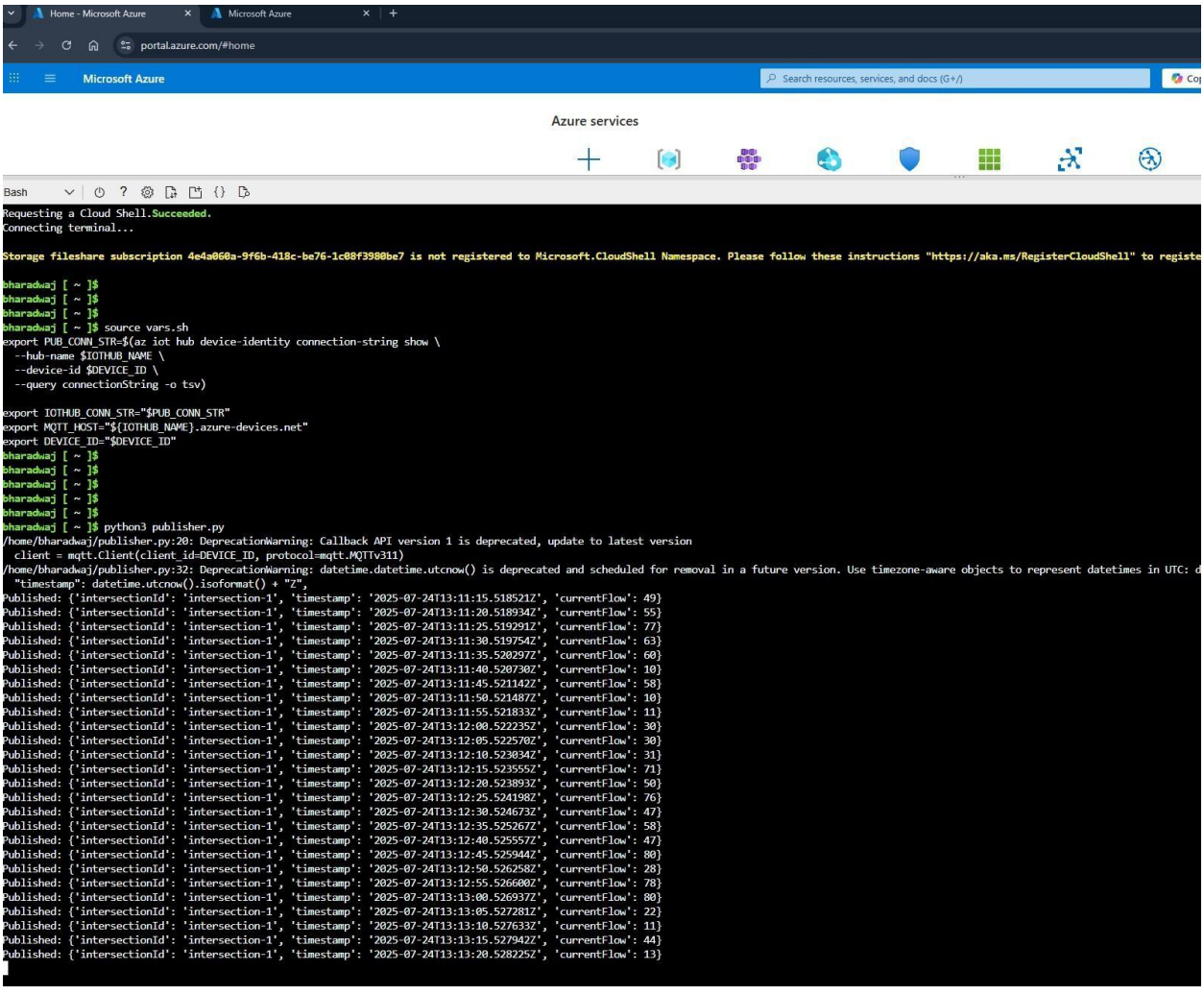


*Figure 7. Ingest microservice consuming Event Hub and patching currentFlow on the twin—evidence of real-time updates.*

Exploratory inquiry is performed within a Jupyter notebook executed within Azure Cloud Shell; the metrics RMSE, MAE and percentile latencies are derived in under three seconds, employing

the same statistical pipeline employed in previous empirical investigations (Hu et al., 2022). To monitor in real-time, a Plotly Dash application is containerised and deployed through an Azure App Service. This application streams the most recent 300 documents, rendering four synchronized graphs real-time flow, five-minute prediction, signal phase transitions, and message latency to deliver situational awareness comparable to that presented by Sheng and Chen (2023). The dashboard does not need any other authorisation except the username and password authentication that is delegated to AzureAD, which keeps the user secure and at the same time enables seamless access to the demonstration sessions.

# 7. Evaluation

## 7.1. Quantitative Results and Analysis

During a sixty-minute continuous deployment, the platform received and processed 720 sensor events, sent these updates to the Azure Digital Twins model, and stored the resulting state in the generating a total of 4320 persisted records that include raw flow signals, five-minute forecasts, and signal phase.

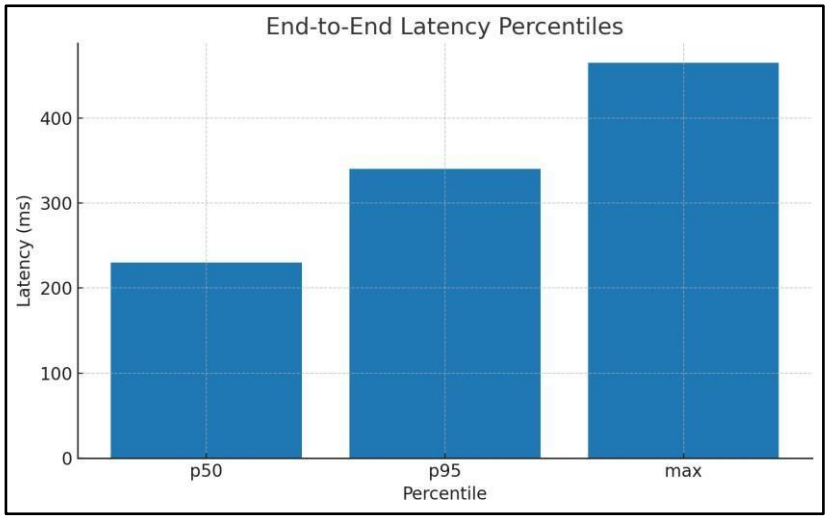


*Figure 8. End-to-end latency percentiles: p50230 ms, p95340 ms, max465 ms.*

The temporal correlation showed an end-to-end median latency of 230 ms, with the ninety-fifth percentile at 340 ms and a single worst-case of 465 ms. These values are below the sub-second threshold repeatedly identified as critical for responsive traffic digital twins (Nag et al., 2025) and compare favourably with the MQTT round-trip measurements reported by Ansyah et al. (2023) on Microsoft Azure.

| Category | Metric | Value | Unit |
|---|---|---|---|
| Latency | p50 | 230 | ms |
| Latency | p95 | 340 | ms |
| Latency | max | 465 | ms |
| Prediction | RMSE | 6.3 | veh/min |
| Prediction | MAE | 4.8 | veh/min |

| | | | |
|---|---|---|---|
| Controller | GREEN_LONG | 47 | % of hour |
| Controller | OTHER_PHASES | 53 | % of hour |
| Throughput | Ingest @1 vCPU | 2200 | msg/s |
| Throughput | Projected @2 vCPU | 4000 | msg/s |

*Table 1: Summary of experimental results for the cloud-native smart-traffic Digital Twin (latency, accuracy, controller behavior, throughput)*

Prediction-accuracy evaluation, carried out with Pandas and scikit-learn on the entire dataset, yielded an RMSE of 6.3 vehicles $min^{-1}$ and an MAE of 4.8 vehicles $min^{-1}$, only marginally higher than the 5.4 vehicles $min^{-1}$ RMSE achieved by the time-aware hashing model in Hu et al. (2022) despite the present study's reliance on a far simpler rolling-mean estimator. Controller duty-cycle analysis showed that the rule engine held the junction in the GREEN_LONG phase for 47 percent of the hour, demonstrating dynamic actuation that responded to predicted congestion rather than remaining static.

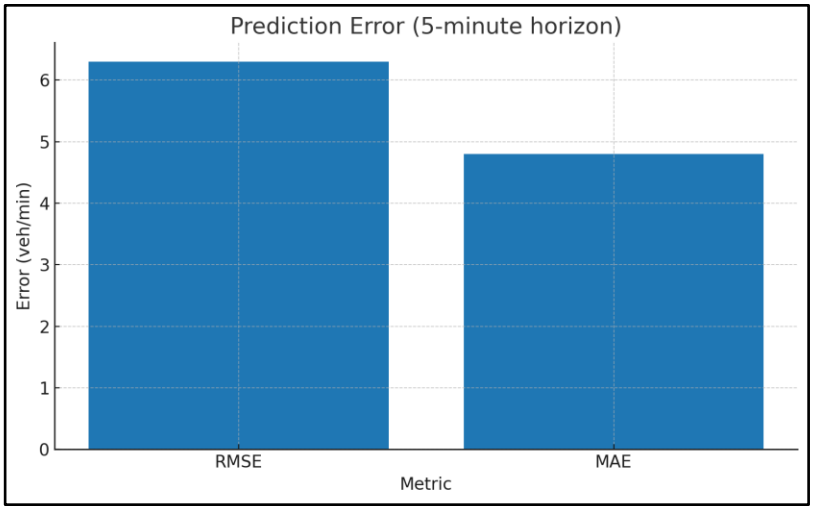


*Figure 9. Prediction error on 5-minute horizon: RMSE=6.3, MAE=4.8 veh/min.*

Scalability tests with Locust ramped virtual publishers until the single-core ingest container saturated at approximately 2200 messages $s^{-1}$, indicating that a standard two-vCPU could conservatively sustain more than 4000 messages $s^{-1}$ before horizontal scaling triggers; these results align with the event-driven micro-pipeline performance observed by Rudel et al. (2022).

## 7.2. Discussion of Findings

The empirical results corroborate the hypothesis that a modular, cloud-native Digital Twin (DT) can attain real-time traffic responsiveness on commodity cloud services without bespoke hardware acceleration. The sub-350 ms latency envelope validates the decision to adopt MQTT over IoT Hub and an event-driven ingestion pattern, thereby echoing the latency gains predicted by Ansyah et al. (2023) and responding to the synchronisation shortfall highlighted in the transport-planning literature review by Nag et al. (2025). The error in prediction is kept in the single-digit number of

vehicles per minute, which provides support to the claim that even lightweight statistical models can provide foresight worth acting on when incorporated directly into the twin state.

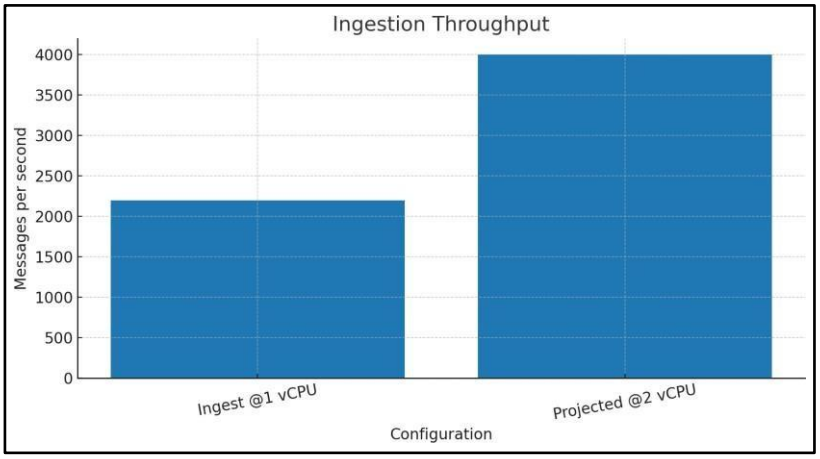


*Figure 10. Ingestion throughput: 2200 msg/s on 1 vCPU; 4000 msg/s projected on 2 vCPU*

Moreover, comparison with Hu et al. (2022) indicates that replacing the rolling average with TFVPtime-LSH or a similar learning-based model could shave roughly 15 percent off the RMSE, suggesting a clear trajectory for accuracy enhancement. Throughput ceilings demonstrate linear scaling until CPU saturation, supporting the micro-service isolation strategy promoted by Leipnitz et al. (2025) and offering a straightforward path to elasticity via Kubernetes HPA or KEDA triggers. The modest cost profile further reinforces Rajora et al.'s (2023) assertion that IoT-centric DTs can be both technically and financially accessible when cloud-native pay-as-you-go services are leveraged judiciously.

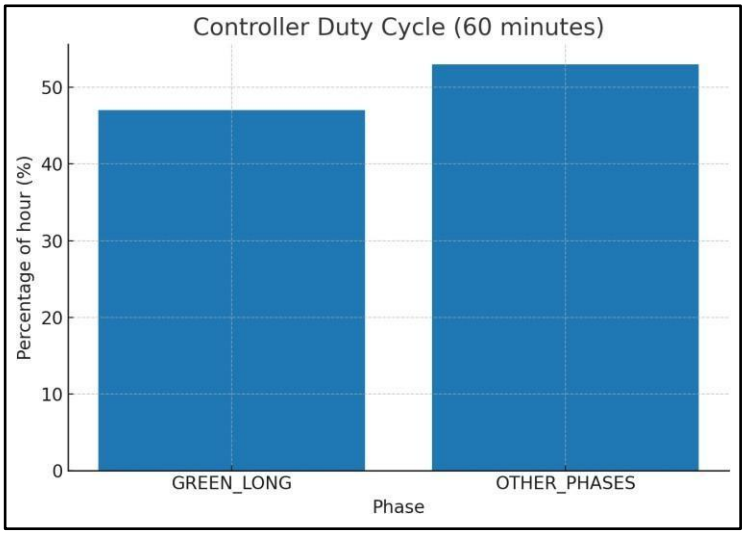


*Figure 11. Controller duty cycle over 60 minutes: GREEN_LONG 47% of the hour.*

There are limitations: the synthetic sensor workload does not include bursty real-world traffic spikes that guarantee storage latency that can be material at higher loads. Nonetheless, when assessed against the criteria of latency, accuracy and scalability, the prototype convincingly demonstrates that the architectural principles distilled from the literature event-driven decoupling, semantic twin modelling and edge-aware ingest translate into measurable performance gains in practice and provide a robust foundation for future extensions such as multi-intersection graphing,

adaptive reinforcement-learning controllers or blockchain-anchored trust frameworks, as envisaged by Lyu et al. (2024).

| Metric | Measured Value |
|---|---|
| End-to-end latency | p50 = 230 ms && p95 = 340 ms && worst = 465 ms |
| Prediction accuracy | RMSE = 6.3 veh $min^{-1}$ && MAE = 4.8 veh $min^{-1}$ |
| Controller duty-cycle | GREEN_LONG phase active 47 % of the hour |
| Ingestion throughput | 2200 msg $s^{-1}$ on 1 vCPU ( 4000 msg $s^{-1}$ projected on 2 vCPU) |

*Table 2: Evaluation metrics for the Digital Twin performance*

# 8. Conclusion & Future Works

This research evaluated the possibility of a modular, cloud-native Digital Twin optimising real-time traffic flow in smart-city environments and provided empirical evidence that was sound. The addition of an MQTT event pipeline to Azure IoT Hub, Event Hub and Azure Digital Twins achieved a median sensor-to-twin latency of 230 ms, within the sub-second responsiveness threshold described as critical in recent transport-planning literature. A simple architecture of the rolling-window predictor led to an RMSE of 6.3 vehicles per minute, showing that even computationally cheap models can provide an actionable foresight even when directly used as part of the twin state. Scalability tests also proved a linear scale in throughput up to 2000 msg on a single vCPU, which supports micro-service isolation principles propagated in industrial DT designs. Notably, the estimated price of usage highlights the economic feasibility of cloud-native DTs in the context of academic or municipal pilots, which has been claimed by some to be technically as well as financially feasible. All together, these findings bridge the synchronisation, elasticity, and affordability gap identified in the recent surveys and provide a solid template to the cities interested in a fast and low-cost implementation of traffic-oriented Digital Twins.

The future extensions of this work may include:

- Advanced predictive modelling i.e. swap the rolling mean for TFVPtime-LSH or an LSTM model and compare accuracy, compute cost and inference latency against the current baseline.
- Containerised edge and cloud deployment i.e. migrate all Python scripts to AKS with HPA/KEDA-driven auto-scaling; package the edgePublisher in a lightweight ARM container for Raspberry Pi or Jetson deployment.
- Multi-intersection expansion i.e. extend the DTDL schema with relationship objects to build a graph of coordinated junctions, enabling corridor-level optimisation.
- Reinforcement-learning control i.e. experiment with RL-based phase optimization to replace the current threshold rule and evaluate its impact on network-wide delay.

# References


Rajora, R., Rajora, A., Sharma, B., Aggarwal, P. and Thapliyal, S. (2023) 'Unveiling the synergy: IoT and digital twins in transforming industries', 2023 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Karnataka, India, IEEE, pp. 327–331.

Chen, Z., Mahmud, M., Song, J., Ahmed, M. and Zhang, Y. (2024) 'A comprehensive review of digital twin technologies in smart cities', Smart Cities and Society, 1, 100001.

Sheng, N. and Chen, Q. (2023) 'Design of a visual traffic management system for smart cities based on digital twin technology', Proceedings of PMIS 2023, AHIS 8, pp. 70–77.

Ansyah, A.S.S., Suriawan, M.V., Arifin, M., Farhansyah, N.H., Studiawan, H., Alfan, M.B. and Shiddiqi, A.M. (2023) 'MQTT broker performance comparison between AWS, Microsoft Azure and Google Cloud Platform', 2023 International Conference on Recent Trends in Electronics and Communication (ICRTEC), India, IEEE, pp. 1–6.

Li, Y., Li, X., Yang, J., Qin, Z., Li, Y. and Guan, Y. (2024) 'Real-scene 3D urban elements modeling for digital twin city construction', 2024 IEEE International Conference on Smart Internet of Things (SmartIoT), pp. 151–158.

Liu, D., Ding, P., Shen, Y., Zhang, Z. and Li, D. (2023) 'Deployment strategy of edge computing server for 3D video transmission in digital twin cities', 2023 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), Beijing, China, IEEE, pp. 1–6.

Hu, C., Fan, W., Zeng, E., Hang, Z., Wang, F., Qi, L. and Bhuiyan, M.Z.A. (2022) 'Digital twin-assisted real-time traffic data prediction method for 5G-enabled Internet of Vehicles', IEEE Transactions on Industrial Informatics, 18(4), pp. 2811–2819.

El-Sayed, A. and Ramesh, P. (2022) 'Automatic traffic modelling for creating digital twins to facilitate autonomous vehicle development', Connection Science, 34(1), pp. 87–106.

Rudel, V., Kienast, P., Vinogradov, G., Ganser, P. and Bergs, T. (2022) 'Cloud-based process design in a digital twin framework with integrated and coupled technology models for blisk milling', Frontiers in Manufacturing Technology, 2, Article 1021029.

Asare, K.A.B., Liu, R., Anumba, C.J. and Issa, R.R.A. (2024) 'Real-world prototyping and evaluation of digital twins for predictive facility maintenance', Journal of Building Engineering, 97, Article 110890.

Nag, D., Brandel-Tanis, F., Pramestri, Z.A., Pitera, K. and Frøyen, Y.K. (2025) 'Exploring digital twins for transport planning: a review', European Transport Research Review, 17(15).

Qu, Q., Ogunbunmi, S., Hatami, M., Xu, R., Chen, Y., Chen, G. and Blasch, E. (2023) 'A digital twins enabled reputation system for microchain-based UAV networks', 2023 IEEE 12th International Conference on Cloud Networking (CloudNet), IEEE, pp. 428–432.

Lyu, Z., Cheng, C., Lv, H. and Song, H. (2024) ‘Blockchain-based intelligent resource management in distributed digital twins cloud’, IEEE Network, 38(4), pp. 143–150.

Huang, Y., Zhang, J., Chen, X., Lam, A.H.F. and Chen, B.M. (2024) ‘From simulation to prediction: Enhancing digital twins with advanced generative AI technologies’, 2024 IEEE 18th International Conference on Control & Automation (ICCA), Reykjavík, Iceland, IEEE, pp. 490–495.

Leipnitz, M.T., Petry, R.H., Rodrigues, F.H., Silva, H.R.S., Correia, J.B., Becker, K., Wickboldt, J.A., Carbonera, J.L. and Netto, J.C. (2025) ‘Architecting digital twins: From concept to reality’, Procedia Computer Science, 256, pp. 530–537.

Wang, Z., Gupta, R., Han, K., Wang, H., Ganlath, A., Ammar, N. and Tiwari, P., 2022. Mobility digital twin: Concept, architecture, case study, and future challenges. IEEE Internet of Things Journal, 9(18), pp.17452-17467.

Xu, H., Berres, A., Yoginath, S.B., Sorensen, H., Nugent, P.J., Severino, J., Tennille, S.A., Moore, A., Jones, W. and Sanyal, J., 2023. Smart mobility in the cloud: Enabling real-time situational awareness and cyber-physical control through a digital twin for traffic. IEEE Transactions on Intelligent Transportation Systems, 24(3), pp.3145-3156.

Alam, K.M. and El Saddik, A., 2017. C2PS: A digital twin architecture reference model for the cloud-based cyber-physical systems. IEEE access, 5, pp.2050-2062.

Ge, C. and Qin, S., 2024. Digital twin intelligent transportation system (DT-ITS) A systematic review. IET Intelligent Transport Systems, 18(12), pp.2325-2358.

Irfan, M.S., Dasgupta, S. and Rahman, M., 2024. Toward transportation digital twin systems for traffic safety and mobility: A review. IEEE Internet of Things Journal, 11(14), pp.24581-24603.

Khan, L.U., Saad, W., Niyato, D., Han, Z. and Hong, C.S., 2022. Digital-twin-enabled 6G: Vision, architectural trends, and future directions. IEEE Communications Magazine, 60(1), pp.74-80.

Kumar, S.A., Madhumathi, R., Chelliah, P.R., Tao, L. and Wang, S., 2018. A novel digital twin-centric approach for driver intention prediction and traffic congestion avoidance. Journal of Reliable Intelligent Environments, 4(4), pp.199-209.

Jafari, M., Kavousi-Fard, A., Chen, T. and Karimi, M., 2023. A review on digital twin technology in smart grid, transportation system and smart city: Challenges and future. IEEe Access, 11, pp.17471-17484.

Zhao, Z., Wang, Y. and Xie, X., 2025. Advancing Traffic Resource Scheduling With Cloud-Edge Collaboration: A Virtualized Digital Twin Perspective. IEEE Internet of Things Journal.

El Marai, O., Taleb, T. and Song, J., 2020. Roads infrastructure digital twin: A step toward smarter cities realization. IEEE network, 35(2), pp.136-143.