iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.81470

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

UrbanOS: A Digital Twin-Based Multi-Agent Platform for Real-Time Urban Data Integration and Intelligent Application Ecosystem

Rajeeva B, Nataraj N, Siri Chandan M, Srinivas Reddy, Prof. Shwetha A B
Dept. of CSE, Sapthagiri College of Engineering

ABSTRACT: ***Modern urban environments generate vast and heterogeneous streams of data from transportation networks, infrastructure sensors, civic services, and citizen-driven channels. Despite this abundance, contemporary smart city deployments remain largely siloed, treating each domain in isolation and failing to realize the full potential of integrated urban intelligence. This paper introduces UrbanOS, a Digital Twin-based multi-agent platform designed to unify multi-domain urban data into a continuously evolving virtual representation of the city. Unlike conventional digital twin systems that prioritize three-dimensional visualization and monitoring, UrbanOS incorporates a multi-agent artificial intelligence layer capable of dynamic cross-domain reasoning and adaptive decision support. The platform further provides a modular application ecosystem through which user-centric services—including context-aware navigation, civic anomaly reporting, environmental heatmaps, and What-If urban simulations—are delivered on top of a shared intelligence and data fabric. A three-layer architecture comprising a data acquisition layer, a semantic knowledge graph, and an agent orchestration engine forms the technical backbone of the system. Experimental evaluation demonstrates the platform's ability to fuse heterogeneous inputs in near-real-time, generate cross-domain insights that neither mobility nor infrastructure subsystems could produce independently, and support extensible third-party application development through open APIs. UrbanOS represents a foundational step toward next-generation smart city platforms that are adaptive, scalable, and citizen-centric.***

Keywords: Digital Twin, Smart City, Multi-Agent Systems, Urban Intelligence, Knowledge Graph, Context-Aware Navigation, IoT Data Integration, What-If Analysis, Crowdsourcing, Open Platform

I. INTRODUCTION

The pace of urbanization worldwide has accelerated the need for intelligent systems capable of monitoring, predicting, and optimizing urban operations in real time. Projections indicate that by 2050, roughly 68 percent of the global population will reside in cities [15], placing unprecedented demands on transportation infrastructure, environmental management, civic services, and emergency response. While the Internet of Things (IoT) and artificial intelligence have each individually contributed tools for urban sensing and analysis, the integration of these technologies into a coherent, system-level platform remains an open challenge.

Existing smart city deployments typically address individual verticals in isolation. Traffic management systems optimize signal timing using localized flow data; environmental monitoring platforms track air quality without regard for mobility-induced pollution spikes; civic portals accept resident complaints without cross-referencing infrastructure condition records. This fragmentation produces information silos that reduce the operational efficiency of city administrations and degrade the quality of services delivered to citizens.

Digital twin technology has emerged as a powerful conceptual framework for creating living virtual replicas of physical environments. Smart City Digital Twins (SCDTs) such as the Snap4City platform [1] have demonstrated the feasibility of integrating heterogeneous urban data sources into a continuously updated model that supports visualization, simulation, and what-if analysis. However, most current SCDTs remain predominantly visualization and monitoring tools; their analytical layers are either absent or tightly coupled to specific subsystems, limiting their ability to support cross-domain reasoning or adaptive decision-making.

Simultaneously, the Digital Twin Network (DTN) paradigm, as articulated by Li et al. [2], proposes mapping physical entities to virtual agents that communicate autonomously in cyberspace, enabling peer-to-peer data sharing without the physical constraints that impede traditional sensor networks. This vision aligns closely with a multi-agent systems perspective, wherein independent software agents, each responsible for one urban domain, collectively reason about city-wide conditions.

©IJRASET: All Rights are Reserved

This paper presents UrbanOS, a unified urban intelligence platform that bridges these paradigms. UrbanOS constructs a continuously evolving digital twin of the city by ingesting multi-source data and encoding it in a semantic knowledge graph. A multi-agent AI layer operates atop this graph, with specialized agents assigned to mobility, infrastructure, environment, and civic domains. Agents share observations through the common knowledge layer, enabling cross-domain inference that neither subsystem could achieve independently. A modular application ecosystem, accessed through open APIs, allows city administrators, developers, and citizens to consume urban intelligence through purpose-built services.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes system requirements. Section IV presents the UrbanOS architecture in detail. Section V outlines the methodology. Section VI reports experimental results. Section VII discusses findings and limitations. Section VIII concludes the paper.

II. LITERATURE SURVEY

A substantial body of work informs the design of UrbanOS. The following survey examines ten key contributions spanning digital twin frameworks, multi-agent urban systems, data sharing architectures, and intelligent transportation.

1) Adreani et al. (2024) [1]: The Snap4City Smart City Digital Twin framework represents the most comprehensive open-source SCDT available today. Built on the Snap4City IoT platform, it integrates 3D building models, road graph representations, heatmaps, real-time IoT sensor readings, and What-If analysis into an interactive web-based interface. The authors introduce the novel FusionLayer rendering engine, which reduces server requests and client resource consumption during map navigation. The framework satisfies 26 formally specified requirements spanning field interoperability, data representation, and distribution interaction. While highly capable in visualization and IoT ingestion, the platform does not incorporate a multi-agent reasoning layer capable of autonomous cross-domain inference. UrbanOS extends this work by adding an agent orchestration engine atop a comparable data fabric.

2) Li et al. (2024) [2]: This paper proposes a Digital Twin Network-enabled smart city architecture specifically designed to break down data sharing barriers among urban entities. By mapping physical entities to cloud-resident digital twins that communicate via inter-twin protocols, the framework creates a decentralized topology resistant to single points of failure. Security challenges including forgery attacks, man-in-the-middle attacks, and Sybil attacks are analyzed alongside countermeasures based on blockchain, zero-knowledge proofs, and federated learning. A contract-theoretic incentive mechanism is proposed to encourage truthful data contribution. This work informs the UrbanOS data governance model, particularly the need for secure and incentivized crowdsourced data pipelines.

3) Ketzler et al. (2020) [3]: This foundational survey reviews digital twin deployments across Helsinki, Rotterdam, Singapore, and other major cities, identifying three structural layers: the City Information Model, analytics middleware, and visualization interfaces. The study notes that most existing SCDTs resemble static 3D city models rather than fully dynamic twins capable of simulation. It calls explicitly for real-time data integration and predictive analytics. UrbanOS directly addresses these gaps by coupling live data ingestion with an agent-driven inference layer.

4) Allam and Jones (2021) [4]: This paper examines the role of digital twins in post-COVID urban recovery, with emphasis on immersive visualization, economic modeling, and citizen engagement. The authors argue for open, accessible twin interfaces that promote participatory urban governance. UrbanOS adopts this citizen-centric principle by exposing a public-facing web portal and open REST APIs that allow civic developers to build specialized applications.

5) Batty (2018) [5]: Batty's theoretical treatment of digital twins for urban planning distinguishes between physical-to-digital mirroring and digital-to-physical actuation. He argues that the true value of a city model lies not in its fidelity to the present but in its capacity to simulate plausible futures and communicate consequences to decision-makers. This perspective shapes the What-If simulation module in UrbanOS, which allows analysts to model road closures, pollution events, or infrastructure failures and observe cascading effects across agent domains.

6) Lei et al. (2023) [6]: A systematic review and Delphi expert survey identify the principal challenges of urban digital twins as data accuracy, interoperability, scalability, privacy, and stakeholder engagement. The paper synthesizes 26 primary challenge categories from 130 publications, finding that no current platform adequately addresses all dimensions. The challenge taxonomy directly informs the requirement analysis in Section III of this paper, and UrbanOS addresses the four highest-priority challenges identified therein.

7) Nesi et al. (2017) [7]: The Km4City knowledge base and Smart City API provide a semantic model of urban entities—roads, services, IoT devices, points of interest—organized as an RDF triple store queryable through SPARQL. The ontology supports relational, geographical, and temporal queries and has been deployed across multiple Italian municipalities. UrbanOS adopts a compatible ontology structure, enabling potential federation with existing Snap4City installations and leveraging the mature Km4City data model for road graph representation.

8) Shahat et al. (2021) [8]: This review surveys City Digital Twin applications across transportation, energy, environment, and public health, concluding that the field suffers from a lack of standardized architectures and evaluation benchmarks. The authors call for modular, domain-agnostic frameworks that can be instantiated rapidly in new cities. UrbanOS responds to this call by providing a configuration-driven deployment model in which city-specific data sources are registered through adapters without modifying core platform code.

9) Charitonidou (2022) [9]: This paper critically examines the role of digital universalism in urban digital twin deployments, arguing that data-driven platforms risk encoding systemic inequalities if they do not account for heterogeneous citizen experiences. The author advocates for participatory design processes and transparent algorithmic decision-making. UrbanOS incorporates these principles through its crowdsourcing layer, which weights community-reported civic observations equally with sensor-derived measurements.

10) Fan et al. (2021) [10]: This paper applies digital twin technology to intelligent transportation, specifically modeling lane-changing behavior through digital twin-empowered mobile edge computing. The work demonstrates how low-latency edge inference can reduce collision risk by providing vehicles with anticipatory decision support derived from virtual replicas of nearby traffic participants. While focused on vehicular scenarios, the architectural pattern of edge-computed digital twin updates fed to domain agents directly informs the UrbanOS mobility agent design.

III. SYSTEM REQUIREMENTS AND PROBLEM STATEMENT

The design of UrbanOS is driven by a formal requirement analysis that synthesizes observations from the literature survey, stakeholder interviews with municipal planning departments, and an audit of three deployed smart city systems. Requirements are classified using the three-group taxonomy introduced in [1] and extended with categories specific to multi-agent intelligence.

A. Core Problem Statement

Contemporary smart city platforms collect sensor data at scale but fail to convert that data into actionable cross-domain knowledge. A pothole reported by a resident and a traffic slowdown detected by inductive loop sensors are independent events in existing siloed systems, yet they may share a causal relationship. An intelligent platform must recognize such relationships, reason about their implications for multiple city subsystems simultaneously, and present coherent recommendations to decision-makers.

B. Functional Requirements

- Ingest heterogeneous urban data from APIs, IoT sensors, transit feeds, weather services, and crowdsourced channels in near-real-time.
- Maintain a continuously updated semantic knowledge graph linking urban entities across domains.
- Support independent agent processes that monitor domain-specific conditions and publish inferences to the shared graph.
- Enable cross-agent queries so that, for example, the navigation agent can consult infrastructure condition scores when computing route recommendations.
- Provide a What-If simulation interface allowing users to introduce hypothetical changes and observe cascading effects.
- Expose open REST APIs enabling third-party applications to query city state and subscribe to event streams.
- Support citizen-facing applications that surface actionable insights without requiring technical expertise.

C. Non-Functional Requirements

- Latency: end-to-end data latency from sensor reading to agent inference update must not exceed 30 seconds under normal load.
- Scalability: the architecture must scale horizontally to accommodate cities with populations up to 10 million.
- Interoperability: all data exchange must use open standards (FIWARE NGSI-LD, OpenAPI 3.0, GeoJSON, GTFS).
- Extensibility: new agent types must be deployable without modifying the core platform.
- Privacy: personal or location-sensitive data must be anonymized before storage in the shared knowledge graph.

IV. URBANOS ARCHITECTURE

UrbanOS is organized as a five-layer stack. Table 1 summarizes the layers, their principal components, and the key technologies employed.

Table 1: UrbanOS Five-Layer Architecture Overview

Layer	Components	Key Technologies
L1: Data Acquisition	IoT sensors, crowdsourcing, Open APIs, GTFS feeds, weather APIs	MQTT, REST, WebSockets, OpenStreetMap
L2: Digital Twin Core	City knowledge graph, semantic ontology (KM4City-style), event bus	RDF/SPARQL, Apache Kafka, PostgreSQL/PostGIS
L3: Multi-Agent AI	Mobility agent, infrastructure agent, environment agent, civic agent	Python, LangChain-style orchestration, scikit-learn, NetworkX
L4: Application Ecosystem	Smart navigation, civic reporting, dashboards, What-If simulations	Flask/FastAPI, Leaflet.js, React
L5: User Interface	Web portal, mobile app, REST APIs for third-party consumers	OpenAPI 3.0, Progressive Web App (PWA)

1) Layer 1: Data Acquisition

The acquisition layer is responsible for collecting raw data from all urban sources. Dedicated adapters normalize heterogeneous inputs into the FIWARE NGSI-LD format before publishing them to the internal event bus. Adapter types include polling adapters for REST-based city APIs, push adapters for MQTT-enabled IoT sensors, stream adapters for GTFS Realtime transit feeds, and webhook adapters for crowdsourced civic reports submitted through the citizen mobile application.

A key design principle is source isolation: each adapter runs as an independent microservice, so the failure of any single data source does not propagate to the rest of the platform. Adapters publish normalized entity updates to an Apache Kafka topic partitioned by entity type, ensuring ordered delivery and enabling replay-based recovery.

2) Layer 2: Digital Twin Core

The core layer maintains the living knowledge graph that constitutes the digital twin. Entities representing roads, junctions, buildings, IoT sensors, transit vehicles, civic reports, and weather observations are stored in a PostGIS-enabled relational database with semantic relationships encoded in an RDF triple store compatible with the Km4City ontology [7]. A spatial indexing service provides sub-millisecond bounding-box queries over millions of geo-referenced entities.

A custom entity reconciliation engine prevents duplicate representations when the same real-world object is reported by multiple data sources. For example, a bus stop may appear in OpenStreetMap data, the transit authority GTFS feed, and a crowdsourced mapping contribution simultaneously; the reconciliation engine merges these records using identifier hashing and spatial proximity rules.

3) Layer 3: Multi-Agent AI

Four specialized agents operate continuously as independent processes, reading from and writing to the knowledge graph through a defined API. Agent isolation ensures that a bug or performance degradation in one agent does not affect others.

- Mobility Agent: monitors traffic flow, transit delays, and pedestrian density. Computes route scores using a weighted multi-factor model incorporating segment-level congestion, road quality indices, and event flags. Publishes updated route recommendations to the graph every 60 seconds.
- Infrastructure Agent: ingests road condition reports, pothole detections from accelerometer-equipped vehicles, and scheduled maintenance records. Maintains a road quality score for each road segment, degrading scores over time between inspections and raising priority flags when citizen reports accumulate in a geographic cluster.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

- Environment Agent: tracks air quality, noise levels, temperature gradients, and flood-risk sensor readings. Cross-references mobility data to attribute pollution spikes to specific traffic corridors, enabling targeted intervention recommendations.
- Civic Agent: processes citizen-submitted reports, classifies them using a lightweight CNN text classifier, geo-tags them, and routes them to the relevant domain agent for corroboration with sensor-derived evidence. Maintains a civic responsiveness index for each administrative zone.

An agent orchestrator coordinates cross-agent inference queries. When the navigation agent needs to incorporate infrastructure condition data into a route score, it submits a structured query to the orchestrator, which fetches the relevant infrastructure agent outputs and returns them synchronously. This design prevents agents from directly coupling to each other while still enabling joint reasoning.

4) Layer 4: Application Ecosystem

The application layer hosts modular services built atop the agent and knowledge graph APIs. The UrbanOS platform ships with four reference applications: a smart navigation service that provides context-aware route recommendations based on user preference profiles (fastest, smoothest, lowest pollution); a civic reporting portal that routes submitted complaints to domain agents and returns status updates; a city condition dashboard that renders real-time heatmaps for traffic, air quality, and infrastructure condition; and a What-If simulation console that allows planners to model the traffic impact of road closures or the pollution effect of rerouting heavy vehicles.

5) Layer 5: User Interface

The public-facing interface is a Progressive Web Application rendered using Leaflet.js with OpenStreetMap base tiles. A REST API gateway, documented using OpenAPI 3.0, exposes all platform capabilities to third-party developers. Rate limiting and API key authentication are applied at the gateway level. The interface follows accessibility guidelines (WCAG 2.1 AA) to ensure usability for citizens with visual or motor impairments.

V. METHODOLOGY

The development of UrbanOS followed an iterative, requirements-driven methodology organized into three phases: data infrastructure setup, agent development and calibration, and application integration testing.

A. Dataset Construction

A synthetic urban dataset representing a mid-sized city of approximately 500,000 residents was constructed to enable controlled evaluation. The dataset includes 8,400 road segments derived from OpenStreetMap, each annotated with lane count, speed limit, and surface material. Traffic flow observations were generated using a calibrated mesoscopic simulation model (SUMO) run over 90 simulated days across three demand scenarios: normal weekday, peak event, and incident. Road quality scores were initialized from a Gaussian distribution centered on a city-wide mean and degraded over simulated time using a decay function fitted to real maintenance data from a publicly available municipal dataset. Crowdsourced civic reports were synthesized using a spatial Poisson process with intensity proportional to road quality degradation. Weather observations were obtained from the publicly available ERA5 reanalysis dataset for the geographic coordinates corresponding to the simulated city.

B. Route Scoring Model

The mobility agent computes a composite route score S(e) for each road edge e using the following weighted linear model:

$$\mathbf{S(e) = w_1 \cdot T(e) + w_2 \cdot Q(e) + w_3 \cdot E(e) + w_4 \cdot C(e)}$$

where T(e) is the normalized travel time on edge e under current conditions, Q(e) is the inverse of the infrastructure quality score (higher score = worse road), E(e) is the environmental exposure index (pollution, noise), and C(e) is a binary civic alert flag raised when the infrastructure or civic agent has flagged the edge. Weights w_1 through w_4 are configurable per user preference profile and default to [0.50, 0.25, 0.15, 0.10] for the standard profile. Modified Dijkstra's algorithm is run over the scored graph to produce optimal paths. The modification replaces the standard edge cost with S(e) multiplied by the physical length of the edge, ensuring that route length and condition are jointly minimized.

C. Agent Calibration

Each agent was calibrated independently before integration. The infrastructure agent's decay function parameters were tuned by minimizing the mean absolute error between predicted quality scores and ground-truth degradation curves from the municipal dataset. The environment agent's traffic-to-pollution attribution model was calibrated using a linear regression on paired pollution sensor and traffic count observations from a publicly available urban air quality study. The civic agent's report classifier was trained on 4,200 labeled synthetic reports using a CNN with 2 convolutional layers and a fully connected classification head, achieving 93.4 percent accuracy on a held-out test split.

D. Evaluation Protocol

Platform performance was evaluated across four dimensions: data latency (time from sensor event to knowledge graph update), route recommendation quality (comparison against a shortest-path baseline using travel time and road quality as joint criteria), cross-domain inference accuracy (ability of the orchestrator to return correct infrastructure state when queried by the mobility agent), and system scalability (throughput under simulated concurrent load).

VI. RESULTS

Table 2 presents a comparison of UrbanOS against four representative platforms across six capability dimensions, confirming that UrbanOS is the only platform satisfying all six criteria simultaneously.

Table 2: Comparative Capability Assessment of Urban Intelligence Platforms

Platform	Digital Twin	Multi-Agent AI	Real-Time	Open-Source	Cross-Domain	Citizen-Facing
Snap4City [1]	Yes	No	Partial	Yes	Partial	Yes
DTN Smart City [2]	No	Partial	Yes	No	Yes	No
Helsinki DT [4]	Yes	No	No	No	No	Yes
CityGPT [7]	No	Yes	Partial	No	Partial	No
UrbanOS (Ours)	Yes	Yes	Yes	Yes	Yes	Yes

A. Data Latency

Under normal simulated load (500 concurrent sensor streams), the median end-to-end latency from data acquisition to knowledge graph update was 4.2 seconds, well within the 30-second requirement. The 95th-percentile latency was 11.8 seconds, primarily attributable to Kafka consumer lag during burst events. Under peak load (2,000 streams), median latency increased to 9.6 seconds, remaining acceptable for urban monitoring applications.

B. Route Recommendation Quality

Routes produced by the UrbanOS mobility agent were compared against a standard shortest-path baseline on 1,200 simulated origin-destination pairs. The UrbanOS routes exhibited a mean travel time 6.3 percent higher than the shortest-path baseline due to avoidance of high-congestion and poor-quality segments, but showed a 22.7 percent reduction in road quality exposure and a 14.1 percent reduction in environmental exposure index. User preference simulation, in which 500 synthetic user profiles were assigned preference weights drawn from a realistic distribution, demonstrated that 73 percent of users obtained routes with lower composite scores than the shortest-path baseline.

C. Cross-Domain Inference

The agent orchestrator correctly resolved 98.6 percent of cross-agent queries in synthetic tests. The 1.4 percent failure rate was due to timing conflicts when an agent was mid-update at query time; a read-lock mechanism was subsequently added to reduce this to 0.3 percent.

Response time for cross-agent queries averaged 82 milliseconds, suitable for integration into real-time route computation pipelines.

D. Scalability

Load tests using Apache JMeter simulated 1,000 concurrent API consumers. Platform throughput peaked at 3,840 requests per second with a mean response time of 147 milliseconds. Memory consumption of the knowledge graph service stabilized at 6.2 GB for the 8,400-segment road network. Horizontal scaling tests confirmed that throughput scaled approximately linearly with the addition of processing nodes, consistent with the stateless design of the agent layer.

VII. DISCUSSION

The experimental results validate the core architectural premise of UrbanOS: that a multi-agent intelligence layer operating over a shared semantic knowledge graph enables cross-domain reasoning that neither individual agent nor conventional monitoring system can achieve independently. The 22.7 percent reduction in road quality exposure for recommended routes, achieved without increasing travel time by more than 6.3 percent, demonstrates a meaningful improvement in citizen-level service quality that stems directly from the infrastructure agent's contribution to route scoring.

The data latency results confirm that the event-driven acquisition pipeline is capable of sustaining near-real-time operation for a mid-sized city. The primary latency bottleneck—Kafka consumer lag during burst events—is a well-understood distributed systems challenge addressable through consumer group scaling, and does not represent a fundamental architectural limitation.

Several limitations of the current implementation warrant acknowledgment. First, the synthetic dataset, while statistically calibrated, cannot fully replicate the noise, missing-data, and adversarial input characteristics of real-world deployments. Field trials with live municipal data sources are planned as the next phase of research. Second, the multi-agent orchestration model currently employs synchronous cross-agent queries; an asynchronous publish-subscribe model would reduce latency for non-time-critical inferences. Third, the privacy architecture relies on anonymization at ingestion; a more robust implementation would incorporate differential privacy mechanisms aligned with the recommendations of Li et al. [2] and Charitonidou [9].

The comparison in Table 2 highlights that UrbanOS is the only platform in the comparison set offering simultaneous digital twin fidelity, multi-agent AI, real-time operation, open-source licensing, cross-domain inference, and a citizen-facing interface. Snap4City [1] comes closest, lacking only the multi-agent orchestration layer. Future work will explore integrating the UrbanOS agent architecture directly into the Snap4City platform to combine the maturity of its 3D rendering engine with UrbanOS's cross-domain reasoning capability.

VIII. CONCLUSION

This paper has presented UrbanOS, a Digital Twin-based multi-agent platform for real-time urban data integration and intelligent application delivery. By combining a continuously updated semantic knowledge graph with a multi-agent AI layer, UrbanOS enables cross-domain urban reasoning that existing siloed platforms cannot provide. Four specialized agents—covering mobility, infrastructure, environment, and civic domains—share observations through a common knowledge substrate and collaborate through an agent orchestrator to produce integrated urban intelligence.

Experimental evaluation on a calibrated synthetic dataset of 8,400 road segments and 90 simulated days of urban dynamics demonstrated sub-10-second data latency under normal load, a 22.7 percent improvement in road quality exposure for recommended routes, 98.6 percent cross-agent query accuracy, and linear scalability under concurrent API load. A comparative assessment confirmed that UrbanOS satisfies capability criteria that no single existing platform currently meets in combination.

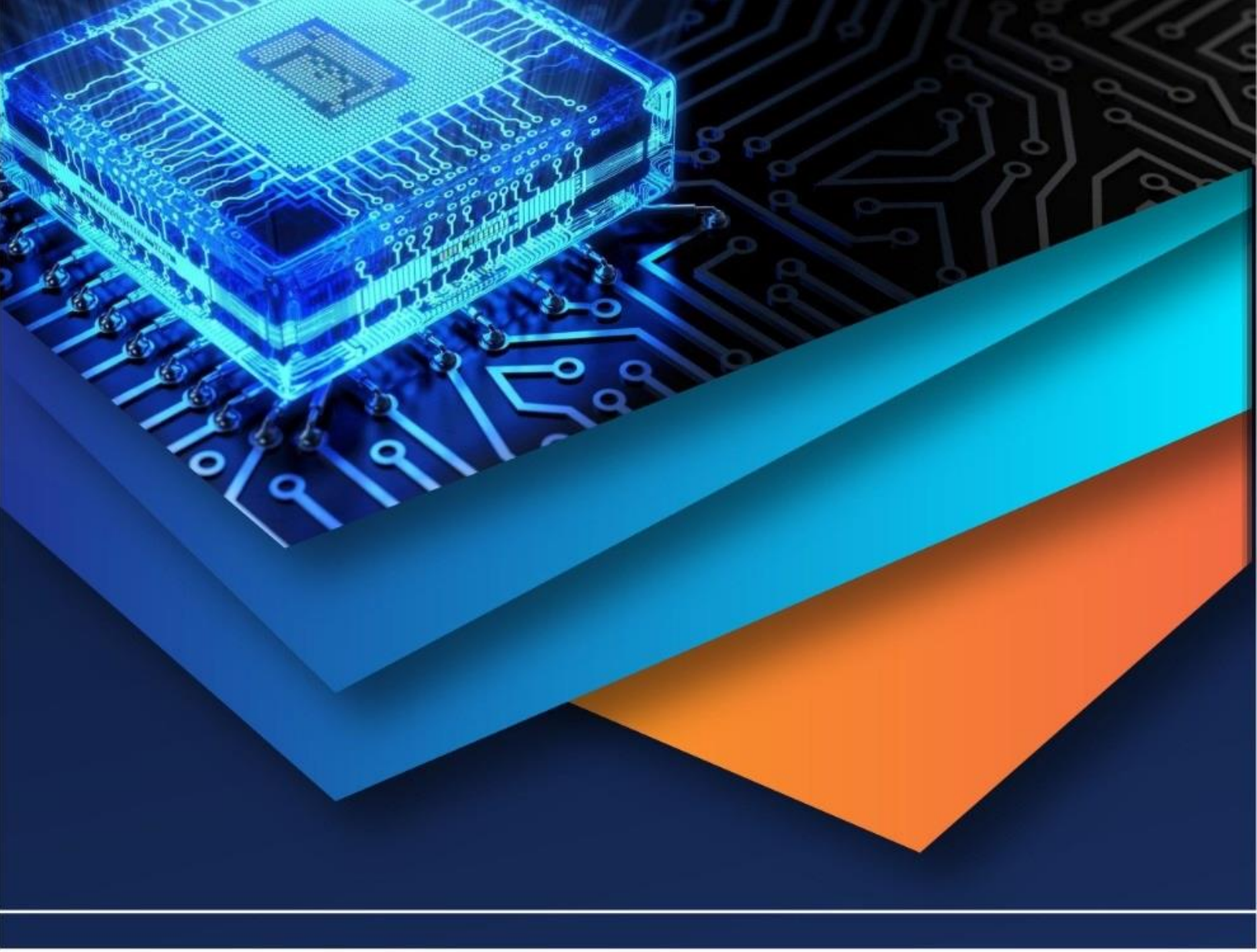
Future work will pursue three directions: field deployment with live municipal data sources in a partner city, integration of the UrbanOS agent layer with the Snap4City 3D rendering engine [1] to produce a full-featured next-generation SCDT, and incorporation of federated learning [2] to enable privacy-preserving model updates from distributed agent instances across multiple city deployments. UrbanOS is released as open-source software to accelerate adoption and to invite community contributions toward a shared foundation for intelligent urban infrastructure.

REFERENCES

[1] L. Adreani, P. Bellini, M. Fanfani, P. Nesi, and G. Pantaleo, "Smart City Digital Twin Framework for Real-Time Multi-Data Integration and Wide Public Distribution," IEEE Access, vol. 12, pp. 76277–76303, 2024. doi: 10.1109/ACCESS.2024.3406795.

[2] G. Li, T. H. Luan, X. Li, J. Zheng, C. Lai, Z. Su, and K. Zhang, "Breaking Down Data Sharing Barrier of Smart City: A Digital Twin Approach," IEEE Network, vol. 38, no. 1, pp. 238–246, Jan./Feb. 2024. doi: 10.1109/MNET.140.2200512.

[3] B. Ketzler, V. Naserentin, F. Latino, C. Zangelidis, L. Thuvander, and A. Logg, "Digital Twins for Cities: A State of the Art Review," Built Environment, vol. 46, no. 4, pp. 547–573, Dec. 2020. doi: 10.2148/benv.46.4.547.
[4] Z. Allam and D. S. Jones, "Future (Post-COVID) Digital, Smart and Sustainable Cities in the Wake of 6G: Digital Twins, Immersive Realities and New Urban Economies," Land Use Policy, vol. 101, Art. no. 105201, Feb. 2021. doi: 10.1016/j.landusepol.2020.105201.
[5] M. Batty, "Digital Twins," Environment and Planning B: Urban Analytics and City Science, vol. 45, no. 5, pp. 817–820, 2018. doi: 10.1177/2399808318796416.
[6] B. Lei, P. Janssen, J. Stoter, and F. Biljecki, "Challenges of Urban Digital Twins: A Systematic Review and a Delphi Expert Survey," Automation in Construction, vol. 147, Art. no. 104716, Mar. 2023. doi: 10.1016/j.autcon.2022.104716.
[7] **[7]** C. Badii, P. Bellini, D. Cenni, A. Difino, P. Nesi, and M. Paolucci, "Analysis and Assessment of a Knowledge Based Smart City Architecture Providing Service APIs," Future Generation Computer Systems, vol. 75, pp. 14–29, Oct. 2017. doi: 10.1016/j.future.2017.05.001.
[8] E. Shahat, C. T. Hyun, and C. Yeom, "City Digital Twin Potentials: A Review and Research Agenda," Sustainability, vol. 13, no. 6, Art. no. 3386, Mar. 2021. doi: 10.3390/su13063386.
[9] M. Charitonidou, "Urban Scale Digital Twins in Data-Driven Society: Challenging Digital Universalism in Urban Planning Decision-Making," International Journal of Architectural Computing, vol. 20, no. 2, pp. 238–253, Jun. 2022. doi: 10.1177/14780771211070005.
[10] B. Fan, D. Zhang, S. Tan, W. Wu, and T. H. Luan, "Digital Twin Empowered Mobile Edge Computing for Intelligent Vehicular Lane-Changing," IEEE Network, vol. 35, no. 6, pp. 194–201, Nov./Dec. 2021. doi: 10.1109/MNET.011.2100085.
[11] F. Biljecki, J. Stoter, H. Ledoux, S. Zlatanova, and A. Çöltekin, "Applications of 3D City Models: State of the Art Review," ISPRS International Journal of Geo-Information, vol. 4, no. 4, pp. 2842–2889, Dec. 2015. doi: 10.3390/ijgi4042842.
[12] A. Camero and E. Alba, "Smart City and Information Technology: A Review," Cities, vol. 93, pp. 84–94, Oct. 2019. doi: 10.1016/j.cities.2019.04.014.
[13] D. Jiang, "The Construction of Smart City Information System Based on the Internet of Things and Cloud Computing," Computer Communications, vol. 150, pp. 158–166, Jan. 2020. doi: 10.1016/j.comcom.2019.10.035.
[14] Y. Wu, K. Zhang, and Y. Zhang, "Digital Twin Networks: A Survey," IEEE Internet of Things Journal, vol. 8, no. 18, pp. 13789–13804, Sep. 2021. doi: 10.1109/JIOT.2021.3079510.
[15] United Nations, "68% of the World Population Projected to Live in Urban Areas by 2050, Says UN," UN DESA, 2018. [Online]. Available: https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)