Proceedings of the 20th Conference on Computer Science and Intelligence Systems (FedCSIS) pp. 351–356 DOI: 10.15439/2025F8576 ISSN 2300-5963 ACSIS, Vol. 43

Digital Twin Micro-Services Architecture to Support a Scenario-based Integration of Smart Mobility Services and Simulations

Damien Nicolas, Ruben J. Marques, Christophe Feltus
0000-0002-6388-032X, 0009-0004-5842-9544, 0000-0002-7182-8185
Luxembourg Institute of Science and Technology (LIST)
Avenue des Hauts–Fourneaux, 5, L–4362 Esch–sur–Alzette, Luxembourg
Email: *firstname.lastname*@list.lu

***Abstract*—This paper proposes a microservices-based digital twin architecture for flexible, scalable, and real-time integration of smart mobility services in urban centers. By decoupling functions into modular services embedded in scenario-driven simulations, the digital twin overcomes limitations of monolithic systems, enhancing adaptability and interoperability. Preliminary results demonstrate its effectiveness for dynamic urban mobility management.**

***Index Terms*—Digital Twin, Mobility, Reference architecture, Micro-services approach, Scenario-driven approach, Mobility Simulation.**

I. Introduction

As CITIES adopt smart mobility solutions to tackle congestion, reduce emissions, and improve transport efficiency, the ecosystem of services—ride-sharing, public transport tracking, electric vehicle infrastructure, and autonomous systems—is rapidly expanding[1]. These services require real-time interaction, but most current integration strategies are rigid, tightly coupled, and difficult to scale or update. This limits cities' ability to adapt their mobility infrastructure as technologies and needs evolve[2]. Digital twins—virtual representations of physical systems—offer a promising approach to dynamically model, simulate, and manage urban mobility. Paired with microservices architecture, digital twins (DT) can evolve from static models into flexible, modular platforms supporting continuous integration of new services[3]. However, many existing DT implementations for mobility rely on monolithic or semi-monolithic systems lacking modularity and real-time interoperability. Adding new services often requires major redevelopment, causing bottlenecks and hindering innovation. There is a need for an architecture that enables seamless integration, fast iteration, and scalability without compromising performance or reliability[4]. This paper presents a microservices-based DT architecture designed for flexible, scalable integration of smart mobility services. Each service can operate independently while contributing to a unified city-wide model, enabling real-time coordination and dynamic updates with minimal friction. A key feature of the architecture is embedding simulation within workflow-driven scenarios and sub-scenarios, rather than as a separate module. This modular design enables configurable, extensible simulation pipelines that capture urban mobility's complexity and support context-specific experimentation and decision-making. Section III defines the concept of Simulation and Scenario. Section IV proposes the design principles for the architecture. Section V details the proposed architecture and its key components. Section VI describes the implementation of the case study to validate the approach. Section VII discusses the results, challenges, and insights gained and concludes with key takeaways and directions for future work.

II. Related work

The *Digital Twin* concept was first introduced in a NASA roadmap[1] [5] as a probabilistic, multi-physics simulation of a space system, combining physical models, sensor data, and operational history. Grieves [6] later formalised it through three core elements: *a physical product, a virtual counterpart, and the data links between them.* As summarised by [7], a DT is a dynamic virtual representation of a physical system, connected throughout its lifecycle for continuous data exchange. Without this temporal link, the model remains a static simulation rather than a true DT. In *Ceci n'est pas un jumeau numérique*, [8] describes the progression from design models to data-enriched *shadows*, with unidirectional data flow. Twin status requires feedback enabling the virtual to affect the physical. While bidirectional exchange is essential, [9] argues that full understanding requires five dimensions: physical entities, virtual models, data, services, and interconnections. Beyond representation, DTs support decision-making—through monitoring (e.g., vehicle status), simulation (e.g., traffic impacts), and prediction (e.g., mobility demand).

Micro-Services Architecture in Smart Systems enables scalability and maintainability by dividing applications into loosely coupled components that communicate via lightweight APIs. This modularity helps manage complexity in domains like cities, energy grids, or mobility, where subsystems—such as traffic control or route planning—can evolve independently.

[1] $https:\\www.nasa.gov\sites\default\files\501321main_TA11-ID_rev4_NRC-wTASR.pdf$

TABLE I
TECHNICAL LIMITATIONS OF CURRENT APPROACHES

Challenge	Description
Legacy system integration	Difficulty integrating DTs with older infrastructure lacking IoT/sensor support
Scalability	Maintaining accuracy across expanding systems with diverse data volumes
Real-time synchronisation	Latency issues when combining Edge (fast) and Cloud (slow) DTs
Data quality and Governance	Errors in data pipelines lead to inaccurate DT representations

Recent use cases show improved iteration and fault isolation, such as in smart grids for energy forecasting and pricing [10], or in IoT cities for managing sensor and edge data streams [11]. Still, challenges like orchestration and service discovery persist but are mitigated with tools like Kubernetes and service meshes. For mobility-focused DTs, microservices enable real-time simulation, modular updates, and integration of varied services [12].

Integration Frameworks for Mobility Services face challenges due to data mismatches, conflicting services, and diverse tech stacks, limiting real-time performance. Automated methods offer potential but struggle with incomplete data, and domain-specific issues persist. In healthcare, DTs encounter uncertain model parameters and data ownership concerns, needing to balance AI insights with regulations like HIPAA [13].Urban DTs face difficulty simulating emergent behaviours across systems and resolving stakeholder conflicts [14]. Table I summarises key technical challenges in integrating SoS and DTs.

III. SIMULATION AND SCENARIO CONCEPT

This section defines simulation and scenario concepts, key to the proposed microservice architecture, enabling modular and scalable system modelling.

A. Simulation

The simulation is a central element of the digital twin concept. It enables the virtual reproduction of the real system's behaviour using various specialised simulation tools. In this work, four simulators have been integrated: SUMO[2], AIMSUN[3], MnMS[4], and MATSim[5]. These tools cover a wide range of scenarios and analyses in the field of mobility.

The simulation component was modelled using UML and consists of several inputs: parameters, context, initial conditions, constraints, and metrics for performance evaluation. The simulation produces outputs such as key performance indicators (KPIs) or visual results. These outputs can be displayed by the digital twin and support decision-making processes.

Table II provides a detailed description of the different concepts that make up a scenario.

[2]https://www.eclipse.dev/sumo
[3]https://www.aimsun.com
[4]https://github.com/licit-lab/MnMS
[5]https://www.matsim.org

TABLE II
SIMULATION CONCEPT DEFINITIONS

Name of the Concept	Description
Parameter	This class specifies parameters or settings that control the behaviour of the simulation. These parameters may include simulation duration, time step size, numerical methods, model parameters, and other configuration options.
Outcome	This class specifies the anticipated results, findings or insights that are expected to emerge from the simulation. This provides a basis for comparing the actual outcomes of the simulation with the desired or expected outcomes.
Metrics	This class outlines the criteria or measures used to evaluate the simulation (performance, effectiveness, etc.). It defines the key indicators, metrics, or outcomes that will be analysed to assess the success of the simulation in achieving its objectives.
Constraint	This class specifies the constraints, limitations or assumptions that apply to the simulation.
Initial Condition	This class defines the initial states or conditions at the beginning of the simulation. It includes parameters such as initial values, states, positions, velocities, and other relevant variables that characterise the system.
Context	This class contains the information related to the context of the simulation. This includes details about the environment, stakeholders, resources, constraints, and any other factors that may influence the simulation.

B. Scenario

The concept of a scenario is a core part of the proposed microservice architecture. A scenario can include multiple sub-scenarios to support complex simulation workflows. Each scenario consists of sub-scenarios, which are composed of individual steps called ScenarioStep. Each step represents a specific mobility-related service, such as calibration, stress test generation, traffic management, or simulation. This modular structure allows fine control over scenario behavior and execution. Additionally, each sub-scenario is defined by a workflow that manages the sequence and dependencies of its steps.

IV. DESIGN PRINCIPLES

A. Functional and non-functional Requirements

The functional and non-functional requirements stem from four pilot partners: the City of Amsterdam[6], Athens (via DAEM[7]), Helsinki (via Forum Virium Helsinki), and the Luxembourgish bus operator Sales-Lentz[8]. Each contributes with their domain expertise—from smart city development to digital infrastructure and autonomous transport. User-driven features are grouped by topic and labeled as essential or optional. Key functional needs include data collection from diverse sources, real-time dynamic simulations, scenario execution (e.g., traffic events), and 2D/3D visualization. Non-functional needs emphasize data fusion for enhanced insights and strong data privacy safeguards to prevent unauthorized access to personal

[6]https://www.amsterdam.nl/en/
[7]https://www.daem.gr/
[8]https://sales-lentz.lu/en/

TABLE III
FUNCTIONAL NEEDS BY THEME, WITH KEY (K) OR OPTIONAL (O) PRIORITY

Subj.	Features	K/O
Data	-As a data scientist (DS), I want to browse, upload and download data in different formats (JSON, CSV, XML) in the DT.	K
	-As a DS, I want data to be available according to well-defined access rules	O
	-As a DS, I want to get relevant infrastructure data (occupancy of parking lots, pricing, requests for parking places, etc) using open data	K
	-As a DS from Helsinki, I want the Helsinki DT data catalogue to be connected to the DT data management	K
Scenario	-As a civil servant (CS) or as a DS, I want to access a list of scenarios	K
	-As a DS, I want the scenarios to be labelled, timestamped, related to input, and generated by a simulation tool	K
	-As a DS, I want the scenarios to be stored in a database managed by the DT.	K
	-As a CS from Helsinki or as a DS, I want the simulation to be calibrated to better represent the traffic management strategies for Helsinki's harbour.	K
Simulation	-As a CS or as a DS, I want to realise various traffic simulations	K
	-As an employee from Luxembourg or as a DS, I want to simulate autonomous shuttle operations in the Luxembourg area. Therefore, I want: (1) to calibrate the model with data, (2) to collect and to store data from the national open data portal, (3) to simulate the shuttle mobility and the position of the stop stations	K
	-As a CS from Amsterdam or as a DS, I want to simulate the scenarios when the IJ-tunnel is open/closed.	K
	-As a CS from Helsinki or as a DS, I want to simulate different traffic management strategies and to simulate mobility environments when ferries arrive from Tallinn to Helsinki's harbour.	K
Visualization	-As a CS or as a DS, I want to visualise the different traffic management strategies on a 2D map	K
	-As a CS or as a DS, I want to visualise the different traffic scenarios on a 2D map	K
	-As a CS or as a DS, I want to visualise the KPIs associated with the scenario in a dashboard	K
	-As a CS or as a DS, I would like to make simple queries, such as to highlight the evolution of different KPIs	O
	-As a CS from Athens or as a DS, I want to view the different traffic management strategies in 3D	O
	-As a CS from Helsinki, I want to visualise 1 KPI that expresses the number of recommendations followed by the users (50 users for the pilot)	K
Feedback loop	-As an employee from Luxembourg, I would like the shuttle's position to be sent to the passengers' mobile application	K
	-As a CS from Helsinki, I want the trajectory recommendation to be sent to citizens arriving and living in the Helsinki harbour through a dedicated mobile application	K
Real time	-As a CS from Athens or as a DS, I want to use real-time data	O
	-As a CS from Helsinki, I want to visualise traffic management strategies, traffic flow in real-time, and a 3D viewer	O

data. These requirements shape the architecture supporting the Digital Twin (DT) systems used in the pilot projects.

B. Design of the digital twin

The design of a DT for mobility management requires a robust, scalable, and flexible architecture.

1) Microservices as an Architectural Foundation: Microservices offer flexibility, scalability, and resilience for data-intensive mobility systems ([15]). They support independent deployment and integration. Key design principles include **API-first** development, **domain-driven design** for mobility alignment, **containerization** for isolation, and **service decoupling** to reduce dependencies and ease maintenance. These ensure efficient, cloud-native, and modular architecture.

2) Layered Logical Architecture of the DT: To ease the deployment of the DT, we have decided to divide the architecture into layers, each supporting a dedicated functional principle. The seven layers considered are: 1) **Data Ingestion:** Collects real-time data from IoT sensors, GPS, traffic cameras, etc. 2) **Data Processing and Storage:** Processes and stores data in scalable systems (e.g., Apache Kafka, Cassandra) with harmonised formats.3) **Simulation and Analytics:** Provides insights via analytics and machine learning for forecasting and planning.4) **Integration:** Exposes APIs for system interaction and real-time responses.5) **User Interface:** Offers dashboards for operators to explore data and support decisions.6) **Security and Compliance:** Ensures data privacy and integrity with access controls and encryption. 7) **Scalability and Resilience:** Adapts to growth and evolving needs without major disruptions.

V. PROPOSAL ARCHITECTURE

A. Overview of the Digital Twin Micro-Services Framework

The digital twin platform adopts a high-level microservices architecture to ensure modularity, scalability, and maintainability. Each microservice functions as an autonomous unit with a specific role and may include its own user interface, business logic, and database.

Service communication is centralised through an API Gateway. Instead of direct IP-based interaction, all requests pass through the gateway, enabling unified access control, routing, and monitoring—enhancing both security and operational clarity. The gateway also enforces access policies and simplifies service discovery.

Inter-service communication includes authentication tokens, allowing services to verify the legitimacy of incoming requests. This ensures only authenticated and authorised interactions occur within the platform.

All services expose RESTful APIs, enabling standard HTTP communication and supporting independent development, deployment, and scaling.

To ensure platform independence and ease of deployment, all services are containerised. Containers encapsulate each service and its dependencies, enabling consistent deployment across environments and the use of orchestration tools like Kubernetes for automated deployment, scaling, and management.

B. Micro-Service Components

- **Mobility Data Ingestion:** The *Data Manager* provides a central point for accessing and managing data. It includes a user interface, business logic, and storage system. Users can upload or download datasets, while services can query or consume data for processing or simulation tasks.

- **Simulation & Prediction Engine (external):** This component offers unified access to simulation tools such as SUMO, Aimsun Next, MATSim, and MnMS. It abstracts the specifics of each tool, allowing higher-level services to request simulations through a standard, tool-agnostic interface.
- **Service Registry & API Gateway:** This module centralises all communication within the platform and serves as the access point for external clients. Services register their endpoints with the gateway, which manages routing, exposure control, and access policies. Inter-service communication passes through the gateway and includes authentication tokens to verify identity and permissions.
- **Scenarios Engine:** The *Scenarios Manager* orchestrates simulation scenarios by retrieving datasets from the Data Manager and initiating tasks through the Simulation Engine. It includes user interface components for scenario configuration and results visualisation.
- **User Interface / Visualisation Module:** A central front-end portal integrates visual components from services like the Data Manager and Scenarios Manager via `iframes`. This enables services to independently develop their interfaces while ensuring a unified and seamless user experience.

C. Integration Mechanism

The microservices described earlier form the core of the platform, providing essential functions such as data ingestion, scenario orchestration, simulation execution, and communication routing. Together, they ensure interoperability across all modules. The platform is designed for extensibility: developers can introduce new services using their preferred technology stack, provided each service encapsulates its internal components—such as business logic and storage—and exposes a clear HTTP RESTful API for system-wide interaction. To integrate a new service, it must be registered with the API Gateway, which centralises access control, routing, and service discovery. Developers specify which endpoints are exposed to the platform, ensuring new functionalities remain decoupled, manageable, and aligned with established security and communication standards. This modular design allows the platform to evolve, supporting the integration of new tools, analytics, or decision-support modules as the Digital Twin's requirements grow.

VI. Case Study Implementation

A. Implementation Context and Technology Stack

The reference architecture described in this paper has been instantiated in the context of the ACUMEN use case, which aims to enable advanced mobility functionalities such as traffic monitoring, dynamic route optimisation, public transport coordination, and seamless mobility services. The platform's modular microservices approach, centralised access control, and robust data handling, provides a reliable foundation to meet these operational requirements.

The implementation leverages a carefully selected technology stack to support these capabilities, ensuring scalability, interoperability, and maintainability. Table IV summarises the key technologies and tools used across the platform's core components and supporting infrastructure.

TABLE IV
Technology stack used in the implementation

Module	**Technology / Tool**	**Purpose**
API Gateway	Java, Spring Boot, Spring Cloud Gateway, MongoDB	Centralizes communication, manages routing, service registry and discovery
Authentication Service	Keycloak, Java	Handles user authentication, access control, and token management (JWT)
Data Manager	Java, Spring Boot, PostgreSQL, MinIO	Provides a central hub for dataset storage, retrieval, and sharing across services
Scenarios Manager	Java, Spring Boot, ArangoDB	Orchestrates scenario execution, including data fetching and simulation delegation
Simulations Engine	Python, ClickHouse	Interfaces with external tools to manage and process simulation requests
Front-End Shell	Vue.js, Vuetify	Hosts the main portal and integrates service-specific UIs via iframe-based inclusion
Simulation Tools	SUMO, Aimsun Next, MATSim, MNMS	Simulate real-world mobility scenarios and provide output for analysis
Containerization	Docker	Packages services into containers to ensure portability and consistency
CI/CD	GitLab CI/CD, Docker Compose	Automates build, test, and deployment workflows

B. First results of SUMO Simulation Scenario: Greek National Road 91 Closure

This section presents the first results of the scenario execution for a simulation using SUMO. The objective of this scenario is to simulate traffic evolution when the Greek National Road 91 in Athens is closed to traffic, as illustrated in Fig. 2. The scenario was subdivided into several sub-scenarios to visualise the impact of different types of closures (e.g., one lane from north to south, one lane from south to north, or both lanes simultaneously) and to determine the optimal mitigation strategy.

To perform these sub-scenarios, the necessary parameters were introduced. Specifically, the simulation was configured with the following inputs: *rush hour* was enabled, *weather conditions* were set to *rainy*, and an *incident* was introduced in the form of a *vehicle breakdown* located at the *midpoint of the tunnel*. A *dynamic, individually aware rerouting strategy* was applied. *Pedestrian presence* was considered *low*, and there were *no disruptions to public transport*. The simulation was run for *120 minutes* with detailed output including *vehicle*

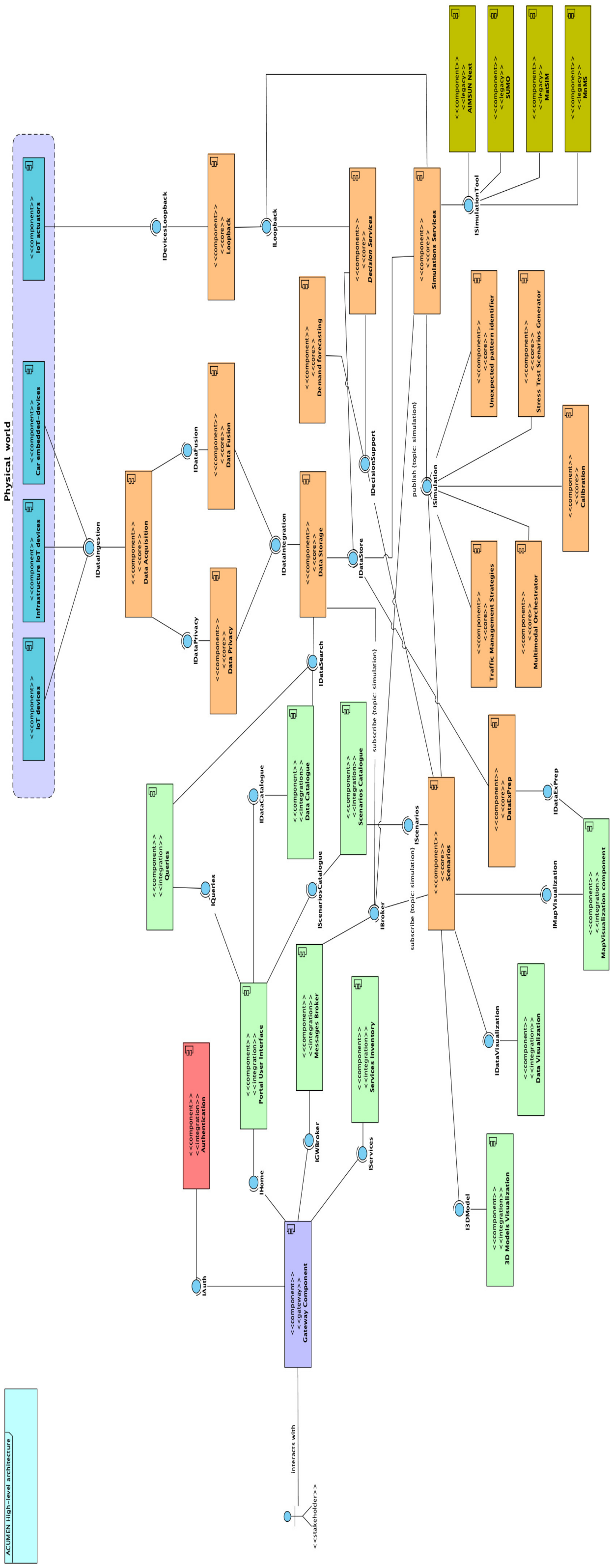

Fig. 1. ACUMEN microservices-based reference architecture

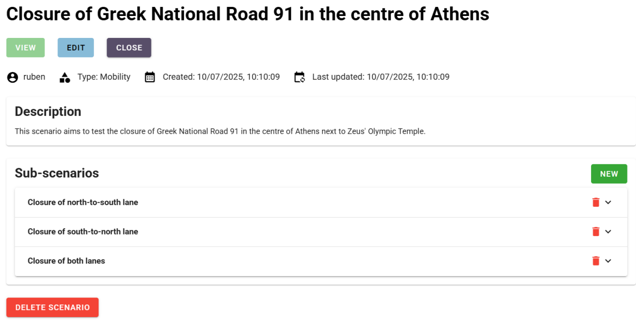

Fig. 2. Global view of the scenario

positions and teleports. The *traffic lights* followed an *adaptive signal control program*, as shown in Fig. 3.

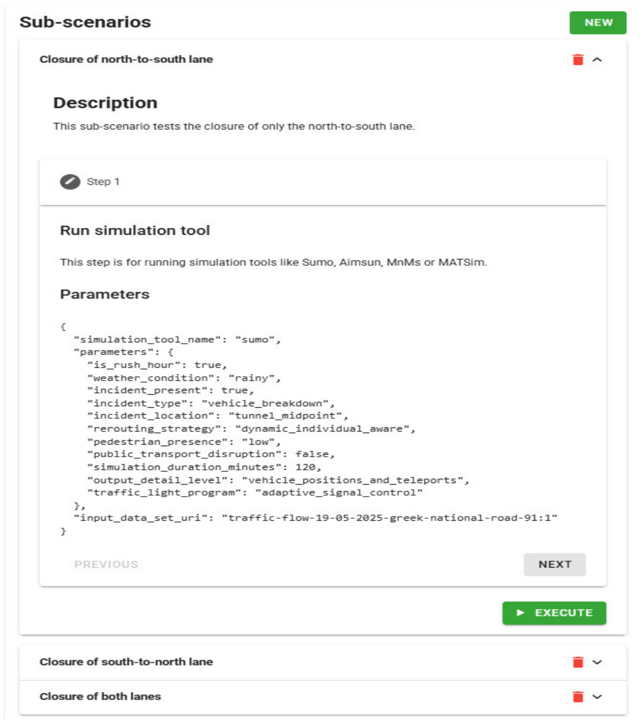

Fig. 3. Sub-scenario parameters

The scenario execution was initiated by clicking the *EXECUTE* green button shown in Fig. 3. After a few minutes of computation, simulation results were available for each sub-scenario, allowing a comparative visualisation of the different traffic configurations. As illustrated in Fig. 4, the results depict the movement of various types of vehicles: *cars in blue*, *buses in green*, and *trams in orange*. These visualisations help to understand how traffic flows redistribute across the city of Athens depending on the specific sub-scenario configuration.

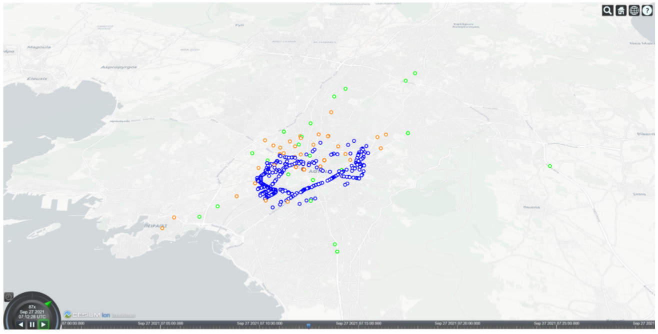

Fig. 4. Visual result of the simulation

VII. Conclusion and Future Works

A Digital Twin Microservices Architecture provides a scalable, modular framework ideal for smart mobility systems. It breaks down complex services—like traffic control or ride-sharing—into independent microservices, improving fault tolerance, development speed, and system updates. When combined with DT, real-world elements like vehicles and infrastructure are mirrored virtually, enabling real-time monitoring, predictive modeling, and system optimization in fast-changing urban environments (transport of dangerous good [16]).

The architecture promotes interoperability across diverse platforms used by different stakeholders, synchronizing data via standard APIs. A shared digital twin layer keeps services aligned with real-time conditions, supporting informed, coordinated decision-making. Its modularity also enables advanced analytics and machine learning, using rich data streams to forecast traffic, optimize routes, and assess infrastructure—without disrupting core operations.

However, several challenges remain. Integrating massive, heterogeneous data sources requires sophisticated pipelines, validation tools, and common standards to ensure accuracy. Orchestrating many microservices also demands complex coordination, secure communication, and real-time responsiveness. Additionally, the approach is resource-intensive, requiring significant investment in computing power, cloud infrastructure, and cybersecurity. As cities scale up smart mobility, managing cost, system complexity, and security becomes increasingly difficult. Overcoming these hurdles will require technical innovation, strong governance, and collaboration across sectors.

This paper proposes a microservices-based digital twin for smart mobility, featuring modular simulations for adaptability and interoperability. A case study on the Athens tunnel closures shows its effectiveness. Future work aims to target automation, self-adaptation, cloud-edge performance, data privacy, and AI explainability, replacing rigid systems with agile urban mobility solutions.

Acknowledgment

We gratefully acknowledge that project ACUMEN (project number 101103808), funded by the European Union as Research and Innovation Action through the Horizon Europe Cluster on Climate, Energy and Mobility, has been instrumental in shaping the ideas and insights presented in this work.

References

[1] C. Lai, F. Boi, A. Buschettu, and R. Caboni, “Microservices for multimobility in a smart city,” *ICIW 2019: The Fourteenth International Conference on Internet and Web Applications and Services*, 2019.

[2] C. Goumopoulos, “Smart city middleware: A survey and a conceptual framework,” *IEEE Access*, vol. 12, pp. 4015–4047, 2024. doi: 10.1109/ACCESS.2023.3349376

[3] E. Faliagka, E. Christopoulou, D. Ringas, T. Politi, N. Kostis, D. Leonardos, C. Tranoris, C. P. Antonopoulos, S. Denazis, and N. Voros, “Trends in digital twin framework architectures for smart cities: A case study in smart mobility,” *Sensors*, vol. 24, no. 5, p. 1665, 2024. doi: 10.3390/s24051665

[4] A. Raghunandan, D. Kalasapura, and M. Caesar, “Digital twinning for microservice architectures,” in *ICC 2023-IEEE International Conference on Communications*. IEEE, 2023. doi: 10.1109/ICC45041.2023.10279802 pp. 3018–3023.

[5] M. Shafto, M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne, and L. Wang, “Draft modeling, simulation, information technology & processing roadmap,” *Technology area*, vol. 11, pp. 1–32, 2010.

[6] M. Grieves, “Digital twin: Manufacturing excellence through virtual factory replication. researchgate; unknown,” 2015.

[7] J. Trauer, S. Schweigert-Recksiek, C. Engel, K. Spreitzer, and M. Zimmermann, “What is a digital twin?–definitions and insights from an industrial case study in technical product development,” in *Proceedings of the design society: DESIGN conference*, vol. 1. Cambridge University Press, 2020, pp. 757–766.

[8] P. Bibow, M. Dalibor, C. Hopmann, B. Mainz, B. Rumpe, D. Schmalzing, M. Schmitz, and A. Wortmann, “Model-driven development of a digital twin for injection molding,” in *International Conference on Advanced Information Systems Engineering*. Springer, 2020. doi: 10.1007/978-3-030-49435-3_6 pp. 85–100.

[9] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, and A. Y. Nee, “Enabling technologies and tools for digital twin,” *Journal of Manufacturing Systems*, vol. 58, pp. 3–21, 2021. doi: 10.1016/j.jmsy.2019.10.001

[10] A. F. Veloso, J. V. Júnior, M. M. d. N. Costa, R. A. Rabelo, and P. R. Pinheiro, “A comparative evaluation of monolithic and microservice architectures for load profiling services in smart grids,” in *Handbook of Artificial Intelligence and Data Sciences for Routing Problems*. Springer, 2024, pp. 17–36.

[11] H. Siddiqui, F. Khendek, and M. Toeroe, “Microservices based architectures for iot systems-state-of-the-art review,” *Internet of Things*, vol. 23, p. 100854, 2023. doi: 10.1016/j.iot.2023.100854

[12] M. Liu, S. Fang, H. Dong, and C. Xu, “Review of digital twin about concepts, technologies, and industrial applications,” *Journal of manufacturing systems*, vol. 58, pp. 346–361, 2021. doi: j.jmsy.2020.06.017

[13] M. De Domenico, L. Allegri, G. Caldarelli, V. d’Andrea, B. Di Camillo, L. M. Rocha, J. Rozum, R. Sbarbati, and F. Zambelli, “Challenges and opportunities for digital twins in precision medicine from a complex systems perspective,” *npj Digital Medicine*, vol. 8, no. 1, p. 37, 2025. doi: 10.1038/s41746-024-01402-3

[14] M. F. Jahangir, C. P. Leslie Schultz, and A. Kamari, “A review of drivers and barriers of digital twin adoption in building project development processes.” *Journal of Information Technology in Construction*, vol. 29, 2024. doi: j.itcon.2024.008

[15] E. Ferko, A. Bucaioni, and M. Behnam, “Architecting digital twins,” *IEEE Access*, vol. 10, pp. 50 335–50 350, 2022. doi: 10.1109/ACCESS.2022.3172964

[16] A. Imeri, C. Feltus, D. Khadraoui, N. Agoulmine, and D. Nicolas, “Solving the trust issues in the process of transportation of dangerous goods by using blockchain technology,” in *Proceedings of the 11th International Conference on Security of Information and Networks*, 2018. doi: 10.1145/3264437.3264470 pp. 1–2.