[COR: Number] [FN: Number]

Development and Evaluation of a FIWARE-based Digital Twin Prototype for Road Systems

Juliana Hildebrandt[a,*,1], Ludwig Maximilian Leibl[a], Dirk Habich[a,1] and Wolfgang Lehner[a]

[a]*Dresden University of Technology (TUD), Database Research Group, Germany*

Abstract

The mobility of people and goods is a central basis of our modern society with increasingly global and diverse networked processes. In its present form, mobility, especially with regard to road traffic, is currently confronted with global challenges (durability, safety, efficiency, ecology, costs, automation, etc.) that urgently require fundamental solutions. In particular, we have to revise the way we plan and build roads and increasingly take the potential of technical innovations (digitalization or sensors) into account. To enable a more sustainable, safer, and more efficient construction and operation of roads in the future, a digital twin for the road of the future is currently under development in the Collaborative Research Center 339 at TU Dresden and RWTH Aachen. This paper describes an initial comprehensive case study we conducted on such a digital twin road using the open-source platform FIWARE.

Keywords

Digital twin, Road system, FIWARE

1. Introduction

The road system is a backbone of the entire transportation infrastructure and it is currently facing a multitude of problems and critical challenges. One key issue is that the road system is at its limit and availability is often restricted, although road-based passenger and freight traffic has increased significantly in recent years. Other challenges include climate change, the goal and need to conserve resources, and the mobility transition with regard to automated driving. Therefore, the question arises how the road system of the future can be optimally designed and operated. With reference to a Chinese proverb *One generation builds the road on which the next will drive*, this question must be researched and answered now.

The Collaborative Research Center 339[2] at TU Dresden and RWTH Aachen addresses this question and envisions a solution based on the concept of the digital twin of the road system [1]. As illustrated in Figure 1, our vision of a digital twin road introduces interactions between the physical road system and its virtual representation through real time monitoring and optimization of the traffic flow [1]. While the virtual representation (models of the reality in space and time) contains physical and data-driven models of the road-tire-vehicle system, the physical road system objects (e.g., road structure, tires, vehicles, and surrounding objects) are equipped with sensors measuring diverse parameters

---

DiDiT 2024 (1st International Workshop on Distributed Digital Twins), June 17, 2024, Groningen, the Netherlands

[*] Corresponding author.

juliana.hildebrandt@tu-dresden.de (J. Hildebrandt); dirk.habich@tu-dresden.de (D. Habich); wolfgang.lehner@tu-dresden.de (W. Lehner)

0000-0001-7198-8552 (J. Hildebrandt); 0000-0002-8671-5466 (D. Habich); 0000-0001-8107-2775 (W. Lehner)

[1] These authors contributed equally.

CEUR Workshop Proceedings (CEUR-WS.org)

[2] Website: https://www.sfbtrr339.de/en/

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

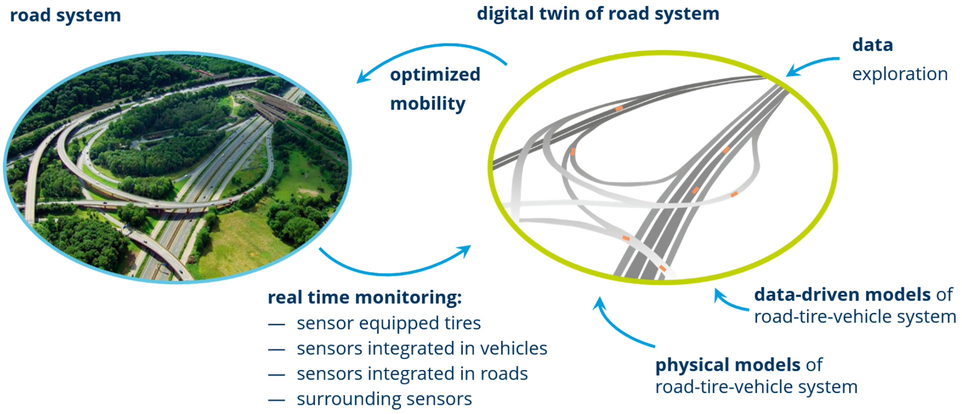

Figure 1: Vision on a digital twin road: interaction between the real road system and its virtual twin by real time monitoring and optimization of the traffic flow [1].

such as load, pressure, deformation, temperature, or humidity. That means, the digital twin road continuously collects and provides real time as well as precise traffic data, information on locations with critical driving maneuvers, and information on state changes such as road damage or changing friction conditions [1]. Based on that, our digital twin road will allow for more precise forecasts of road status and durability, optimization of maintenance intervals, as well as optimization of the loading of the road (e.g., by influencing the tire load distribution on the road surface by prescribing a lateral displacement for autonomous vehicles) [1].

From a technical point of view, there is no consensus with regard to the implementation of digital twins in general and for road systems in particular [2, 3]. Generally, there is a wide range of possible infrastructures [4] and usable technologies [5] to choose from to build a digital twin.

Our Research Question and Contribution: So, our research question is: How suitable is the open-source platform FIWARE[3] [6, 7] for implementing a digital twin for road infrastructure in terms of scalability and handling a large volume of data and sensors?To answer this research question, we present a case study of a digital twin road using FIWARE in this paper. The goal of this case study was to conduct an initial evaluation of the application potential of the FIWARE platform for the technical implementation of a comprehensive digital twin road system. Fundamentally, FIWARE provides a rather simple yet powerful set of APIs (Application Programming Interfaces) and combines components enabling the connection to the Internet of Things with Context Information Management and Big Data services in the Cloud. Specifically, the *Next Generation Service Interfaces Linked Data API (NGSI-LD API)* [8, 9] is used as the standardized programming interface.

Outline: In the following section, we present two representative use cases of the digital twin road in more detail that are decisive for our FIWARE-based use case study. Afterwards, we briefly introduce FIWARE in Section 3, while Section 4 describes the implementation of our digital twin road prototype for the representative use cases applying FIWARE. Then, we present selective experimental results in Section 5. Finally, we conclude the paper with a short conclusion in Section 6.

2. Digital Twin Road and Representative Use Cases

Generally and also specifically for road systems, digital twins can and should be applied over the whole life cycle of the physical counterpart (planning, construction, operation, and maintenance). For our

---

[3] FIWARE Website: https://www.fiware.org

initial case study, we mainly focus on the life cycle aspects of *operation* and *maintenance*. In particular, we assume a sensor-based (or IoT-based) physical road system as a starting point, whereby a road is hierarchically subdivided into road segments, each of which can have any number of lanes. Moreover, several sensors as stand-alone IoT-devices are permanently installed near the road or integrated into the road surface for each road segment and these sensors can be used to continuously record data about the road system itself and its environment. For interaction with the physical road system, we assume that traffic signs are installed on each segment as actuators.

2.1. Representative Use Cases

Based on an IoT-based physical road system, we present two representative use cases for a digital twin road corresponding to the *operation* and the *maintenance* of the road system in the following.

Operation Use Case: The operational life cycle aspect is usually characterized as a real time application by monitoring a road to automatically influence the physical twin by its virtual counterpart. One representative example is the warning of slippery conditions. For this, we require an appropriate computational model for assessing the risk of slippery conditions as e.g., presented in [10]. To apply this model, the approach requires sensors measuring the road surface temperature, the air temperature, dew point temperature — or alternatively the air humidity — and a sensor measuring the condition of the road surface (dry, humid, or wet). Based on the most current values, the model is regularly updated and the output of the model is used to activate or deactivate a slippery road warning signal as actuator.

Maintenance Use Case: A use case for a long-term maintenance application is, e.g., to give a forecast of road conditions (from intact to deteriorated). Here, we also need a model that might be physical or a data-driven model [1]. Our case study focuses on a data-driven forecast. That means, to assess road conditions, an established performance indicator, such as the Pavement Condition Index (PCI) [11] or other indexes, can be utilized. To create data-driven forecasting models for the material condition of roads, machine learning methods have been successfully employed [12, 11]. These methods can take into account several factors such as the most recent information about the road condition and about signs of road pavement fatigue, the structure and material of the road pavement, as well as (historical) weather data such as the number of freeze-thaw cycles per year and traffic data as suitable input parameters. Moreover, these approaches might be used to forecast the road conditions for an assumed traffic load, such that an actuator would inform the maintainers about current or future damages to be expected and divert traffic or reduce the traffic density for individual roads for example to minimize sum of the future repair costs.

2.2. Challenges of a Digital Twin Road

The complexity of designing and implementing digital twins for roads that are able to deal with the introduced use cases implies various challenges, such as the **highly distributed environment**. The sensors themselves often have to be integrated in the road surface to determinate slippery conditions or pavement for physical models to assess the material fatigue [1]. In order to use a simple slippery warning sign as an actuator, the data collected on the roadside does not necessarily have to be collated and processed centrally (cloud computing), but might be consolidated on site (edge and fog computing) to switch the actuator on site with low latency. However, we have a different situation when it comes to predicting the material fatigue of a road depending on the traffic load, as the latter depends on environmental conditions, such as closures of other roads. Similarly, long-term traffic diversions can only be implemented in a central node for an entire road network. In this use case, (pre-processed) sensor data must be consolidated and processed at a central location in the twin architecture.

A second challenge is **data heterogeneity**. The combination of sensor data (also called data in motion) from road-level sensors measuring temperature, traffic density, and other parameters with diverse data sets such as weather conditions poses a challenge due to the inherent data heterogeneity. Here, robust methods to harmonize and reconcile disparate data for accurate prediction of future road conditions are required. In addition, we have to face **scalability** as a third challenge. The monitoring and the road condition estimation for a single road segment is a completely different scenario compared to the monitoring and the road condition prediction of an entire city or at a nation-wide level. Thus, it is necessary to employ a scalable digital twin infrastructure to be able to manage and process high data volumes (data in motion as well as data in rest).

3. FIWARE Platform

From a technical perspective, the FIWARE platform [13] offers a curated framework of open-source software components that can be assembled and combined with other third-party platform components to accelerate the development of smart solutions in different domains such as cities, manufacturing, or utilities [6]. Central building blocks are generic enablers (software components) and NGSI-LD [8], standardized by ETSI ISG CIM [14], as API for the integration of generic enablers [6]. The NGSI-LD API and the corresponding information model has been developed (i) to give application simple access to real-world information by specifying what information they need and (ii) to support various deployment architectures on different scales. As described in [6], the NGSI-LD API is composed of three main parts:

Core-API: has the functionality for synchronously requesting information or for subscribing to relevant changes and asynchronously receiving respective notifications.

Registry API: enables distributed operations and has functionalities for registration and discovery of NGSI-LD context sources.

Temporal API: allows accessing history information by specifying the time interval of interest.

All these properties are very interesting for the development of a digital twin for roads. Moreover, a vibrant global FIWARE community has formed since the FIWARE Foundation was established at the end of 2016.

4. FIWARE-based Digital Twin Road Prototype

Figure 2 shows the developed platform architecture for our case study of a digital twin road. This three-layer architecture follows the reference architecture of a FIWARE-based vertical smart solution [6, 15]. As illustrated at the bottom of this figure, the physical IoT-based road system forms the foundation of the digital twin road. This is followed by the subsequent layers: *data acquisition*, *context broker*, and *application*. The individual components for the layers of *data acquisition* and *context broker* are introduced next, subsequently we describe the implementation of the chosen use case application in more detail.

4.1. Platform Architecture Components

The central component of our three-layer architecture is the **context broker** that is responsible for the integration of data from multiple systems and for creating a holistic view of information/data.

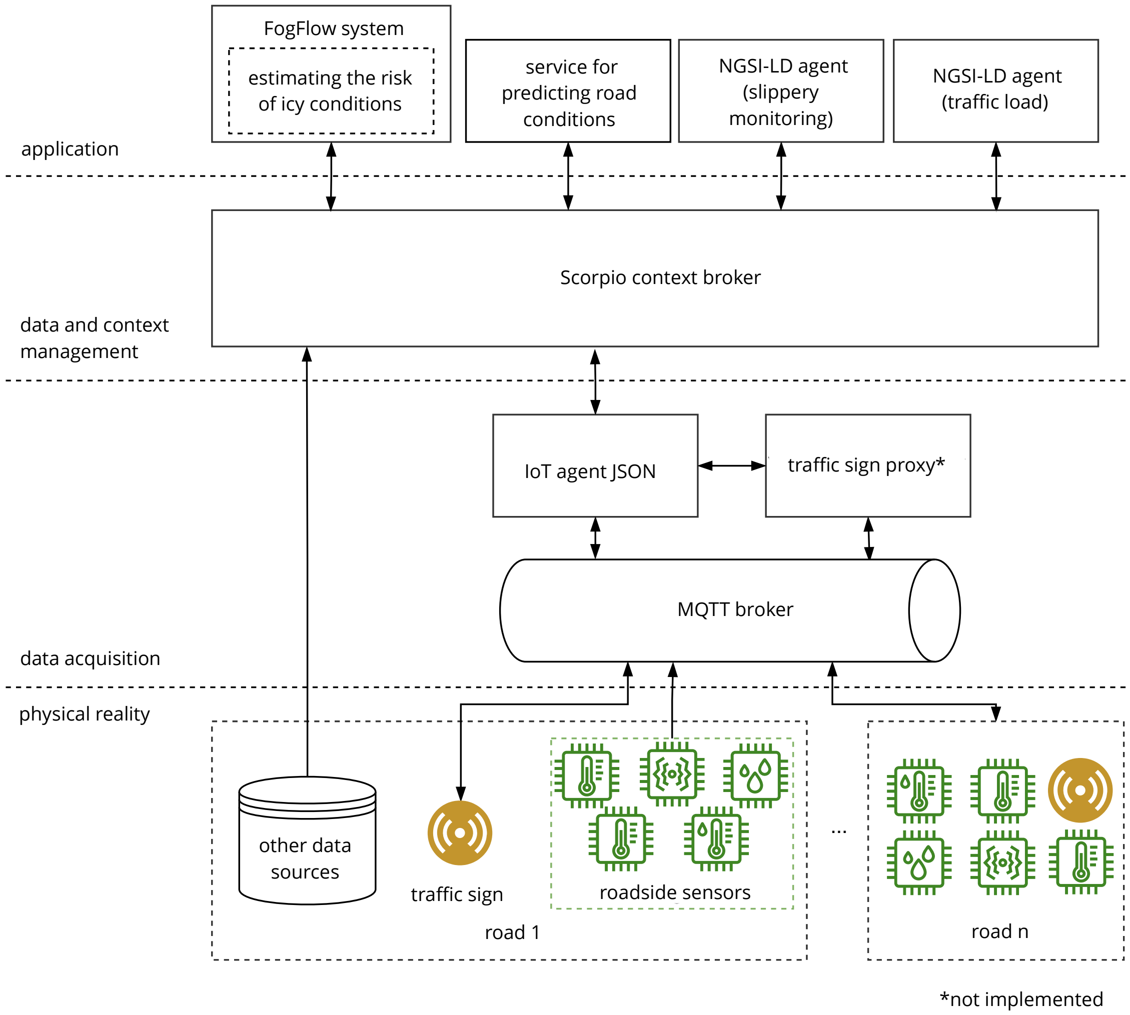

Figure 2: FIWARE-based platform architecture.

This component can be set up as centralized or decentralized system. For our prototype, we basically opted for a centralized system using the NGSI-LD compliant context broker Scorpio[4].

To virtually represent the physical reality of a road system within this context broker, we designed an appropriate NGSI-LD-based data model as schematically depicted in Figure 3. Our road system (cf. Section 2) is primarily based on the Road Smart Data Model [16], which introduces the NGSI-LD entity type `Road` as shown in Figure 3. According to this model, a road is made up of a number of road segments. From this road model, we selected specific NGSI-LD properties and relationships relevant to our case study. Thus, the virtual road is only characterized by its name (`name`) and by the subdivision into individual segments (`refRoadSegment`) in our digital twin. To enable referencing from a road entity to several road segment entities, the corresponding NGSI-LD relationship must be represented as a so-called multi-attribute of an NGSI-LD entity for such a set-based structure.

For the virtual representation of a road segment, the data model for the NGSI-LD entity type `RoadSegment` is modified and extended in accordance with the Road Segment Smart Data Model (cf.

---

[4] https://github.com/ScorpioBroker/ScorpioBroker

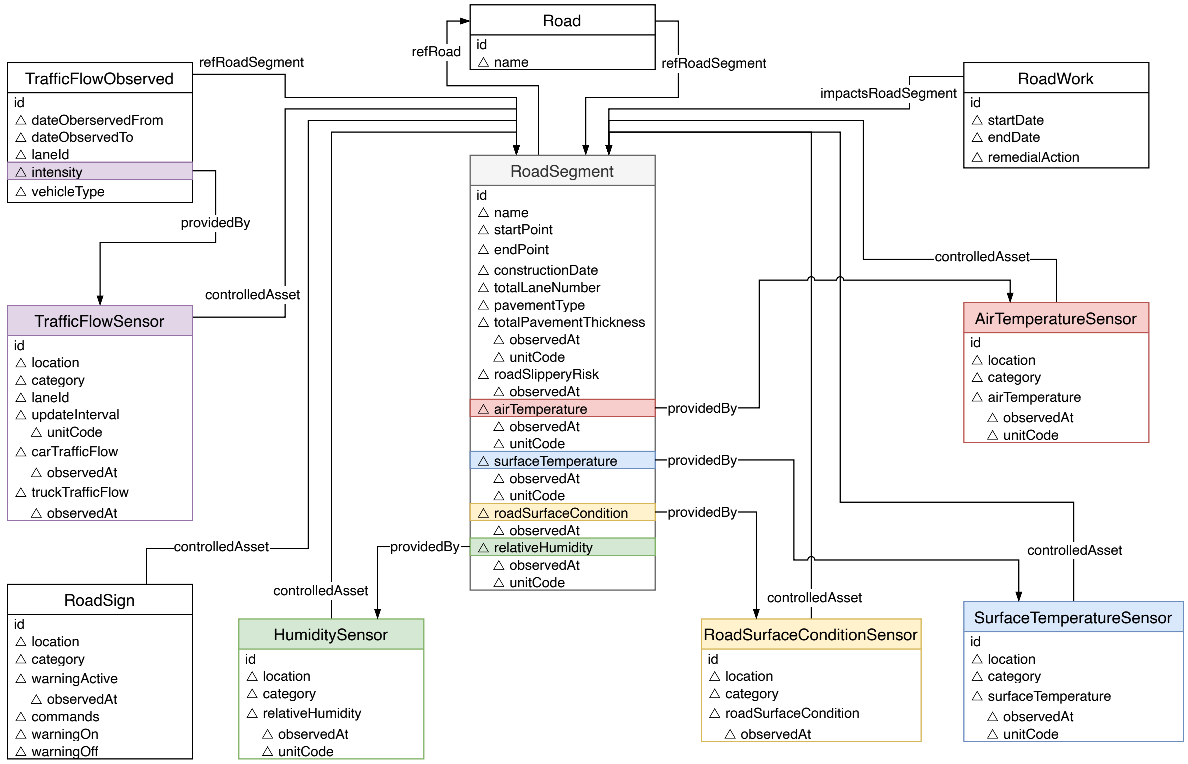

Figure 3: Schematic representation of the NGSI-LD-compliant data model for the developed digital twin road.

Figure 3) for our case study. For example, the geographical start (`startPoint`) and end position (`endPoint`), a name (`name`), the total number of lanes (`totalLaneNumber`) and a reference to the higher-level road entity (`refRoad`) are taken unchanged for road segments from the Road Segment Data Model. Moreover, additional NGSI-LD properties (e.g., `airTemperature`, `humidity`, `surfaceTemperature`, or `roadSurfaceCondition`) are directly integrated into the road segment for modeling weather and environmental observations. This simplification makes it easier to consider the data correlation when implementing the specific use cases.

As we are assuming an IoT-based road system, we use the Device Smart Data Model [17] for the virtual representation of an IoT device. The Device Smart Data Model is a generic data model with which sensors and actuators can be represented. Based on this Device Smart Data Model, independent NGSI-LD entity types are defined for each of the roadside sensors considered in the case study (`AirTemperatureSensor`, `SurfaceTemperatureSensor`, `HumiditySensor`, `RoadConditionSensor` and `TrafficDetector`) and the traffic sign, as shown in Figure 2. In addition, based on the Device Smart Data Model, each sensor is assigned to a corresponding geoposition (`location`), a device type (`category`) and a reference to the road segment (`controlledAsset`) on which the sensor is installed. Moreover, instead of modeling each sensor measurement value as a separate NGSI-LD entity, e.g., based on the Device Measurement Smart Data Model [18], each sensor data model is also assigned an NGSI-LD property relating to the measured variable recorded by the sensor, including metadata for the time of measurement (`observedAt`) and the unit of measurement (`unitCode`).

With the help of the presented NGSI-LD-compliant data model, information can now be integrated into a holistic virtual representation of the road. To achieve that, data from the roadside sensors

```
1  "device_id": "airTemperatureSensor1",
2  "entity_name": "urn:ngsild:AirTemperatureSensor:airTemperatureSensor1",
3  "entity_type": "AirTemperatureSensor",
4  "timestamp": true,
5  "static_attributes":
6   [{"name": "location", "type": "GeoProperty",
7     "value": {"type": "Point","coordinates": [13.742511,51.113057]}},
8    {"name": "category", "type": "Property", "value": ["sensor"]},
9    {"name": "controlledAsset", "type": "Relationship",
10    "value": "urn:gsild:RoadSegment:roadSegment1",
11    "link": {"attributes": ["airTemperature"],
12             "name": "providedBy","type": "RoadSegment"}}],
13 "attributes":
14  [{"object_id": "temperature", "name": "airTemperature",
15    "type": "Property",
16    "metadata": {"unitCode": {"type": "Text", "value": "CEL"}}}]
```

Figure 4: Example configuration for provisioning an air temperature sensor IoT agent.

as well as other data sources have to be collected by the context broker. For this, the lower layer named **data acquisition** of our three-layer architecture is responsible. To connect IoT-devices to a FIWARE-based platform, the provided `IoT Agent JSON` within the FIWARE-framework can be used, although an additional MQTT (Message Queuing Telemetry Transport) broker is required. The IoT Agent specifies certain MQTT topics that an IoT device can use to provide measured values. The virtual representation of the roadside sensors according to the presented NGSI-LD data model can be fully configured via the IoT Agent during device provisioning. Assuming that the roadside sensors can provide their measurement data in a JSON-based format, such as "temperature": 20.0, via MQTT, Figure 4 shows an example configuration for registering an air temperature sensor via the IoT Agent. In this example, an `AirTemperatureSensor` entity (lines 2-3) is created with NGSI-LD properties for the `geoposition` of the sensor (lines 6-7), the `device type` (line 8), and an NGSI-LD relationship with the road segment (lines 9-12) for which the sensor is installed. In addition, the mapping rule for the device messages received via MQTT in JSON format is defined for an NGSI-LD property (line 13-16) so that the IoT agent can update the virtual representation of the sensor in the context broker accordingly for each received measurement value.

Moreover, a dynamic traffic sign is considered as an actuator within the road system as part of our case study. The IoT agents within the FIWARE framework natively support a forwarding model for command execution via the NGSI-LD API, whereby the available commands for the IoT device can be configured when provisioning the devices via the IoT agent. In addition to sensor and actuators, data from other sensors (e.g., road construction data) have to be integrated. For this, our context broker Scorpio offers rich support via so-called context producers.

4.2. Use Case Implementations

Generally, the application implementation of the described *operation use case* requires (i) the implementation of the computational model for the condition estimation of the risk of slipperiness as well as (ii) the realization of the condition monitoring itself, which automatically derives operational decisions regarding the switching of slipperiness warnings on traffic signs based on the virtual representation of roads and applies them in physical reality by controlling corresponding IoT devices (traffic signs). Thus, the technical implementation of this use case requires the integration of components for data analysis into a FIWARE-based platform. The same applies also for our *maintenance use case*, as a data-driven model also has to be calculated there.

Due to the large number of FIWARE components, various implementation approaches for the integration of data analysis are possible here. As part of our use case study, we took a closer look at the following solution approaches: (i) implementation as a stream processing pipeline in a big data framework, (ii) implementation as an individual NGSI-LD agent and (iii) implementation in FogFlow [19, 20]. From the considered approaches for a FIWARE-based implementation of calculation models of roads, FogFlow emerged as the most promising approach and we therefore used it within our prototype. FogFlow is a standard-based IoT fog computing framework that supports serverless computing and edge computing with advanced programming models [19, 20]. Similar to the other approaches, current context information of e.g., `RoadSegment` entities can be imported from Scorpio into the FogFlow system using the subscription/notification mechanism of the NGSI-LD API. When implementing the condition estimation as a FogFunction, the risk of slipperiness for a road segment e.g., is determined as soon as a `RoadSegment` entity is imported into the FogFlow system. Generally, a computational model for state estimation must be implemented accordingly as a FogFlow operator. By configuring several FogFunctions, each with its own operator, different computational models can be implemented and thus an individualization of the condition estimation with regard to the risk of slipperiness on road segments can be achieved. Moreover, FogFlow offers the option of assigning FogFunctions to specific road segments, e.g., by limiting the geographical region. In addition, by selecting the granularity of a FogFunction, the number of specific task instances can be configured up to an individual task instance per `RoadSegment` entity. Therefore, the same computational model in an operator can be used for different segments and individualization can be carried out within the task instance, e.g., through a segment specific parameterization.

Important to note, even if the FogFlow system holds its own virtual representation of the road segments within the IoT brokers, the management of historical context information is not natively supported by the FogFlow system. Therefore, historical data required for state estimation must either be managed independently by a task instance or alternatively retrieved from Scorpio via the *NGSI-LD Temporal API*.

5. Evaluation

To evaluate our developed digital twin road prototype, we used a cloud deployment using Amazon Web Services (AWS)[5]. To ensure the reproducibility of the entire test environment via *Infrastructure as Code*, the cloud deployment is carried out using Terraform [21]. The individual components such as the Scorpio Context Broker, the IoT Agents, and the Fog Flow System were each placed in a separate container, which supports the execution of the components within a scalable cloud environment. Due to space restrictions, we will mainly focus on the *operation use case* below.

In a series of experimental tests, we investigated the scaling behavior of our FIWARE-based prototype by examining the system behavior when integrating an increasing number of roads. For this purpose, we constructed a reference road and by replicating this reference road, the number of roads can be varied as desired. The reference road is subdivided into ten identical reference road segments and each road segment is equipped with four sensors and one traffic sign as actuator as required for the operation use case. Moreover, we simulated the values per sensor using jMeter [22] and we configured jMeter to produce every minute a value to generate a corresponding load in our experiments. The currently measured sensor values are published via MQTT. Then, the FogFlow system determines

---

[5] https://aws.amazon.com/de/

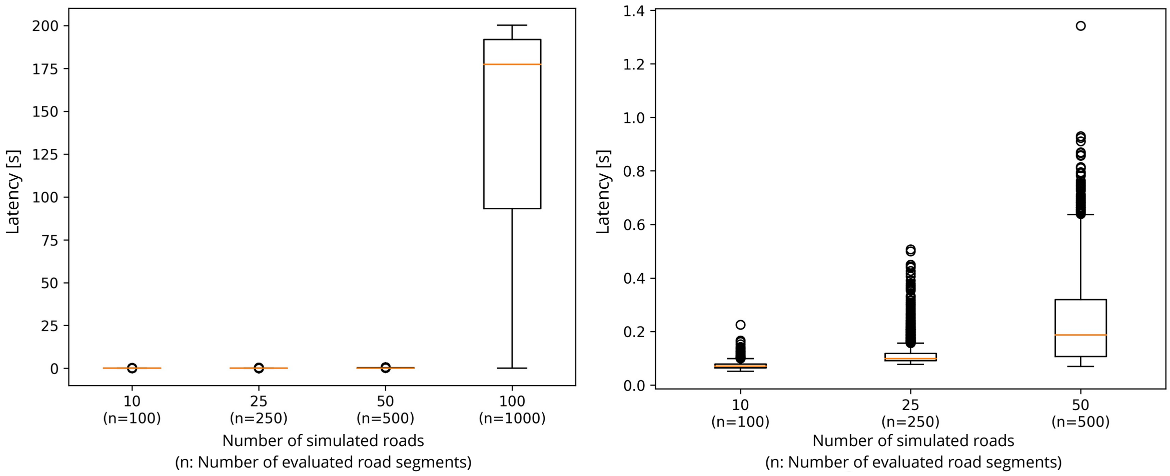

Figure 5: Latency of the calculation of the risk of slipperiness using one context broker.

the risk if slipperiness per segment and the corresponding commands are triggered on the traffic sign via the NGSI-LD agent.

Figure 5 shows initial results for this specific experiment, where the right-hand diagram is a zoom into the left-hand diagram. The diagrams show the latency behavior for the calculation of the risk of slipperiness depending on the number of roads. In this experiment, we deployed one Scorpio context broker and four FogFlow worker each on its own EC2-instance. As we can see in Figure 5, the scaling is very good up to 50 roads, but after that the latency increases disproportionately. In particular, this system configuration can process the current sensor values for up to 50 roads before the new sensor values arrive in the system and promptly issue a warning accordingly. When evaluating the system metrics, Scorpio showed an increasing CPU and RAM utilization depending on the number of roads, with a maximum of up to 100% when simulating 100 roads.

Therefore, the single Scorpio context appears to be a bottleneck. To evaluate this in more detail, we also carried out the same experiments with two Scorpio instances. Although the latency for state estimation for 100 simulated roads could be significantly reduced with a median of 31 seconds compared to the previous experiments, the latencies are still not in a comparable order of magnitude as for 10, 25 and 50 simulated roads with a median of less than 0.10 seconds in each case. In a further experiment, we increased the number of Scorpio Context Brokers to four. With this configuration, our FIWARE-based platform exhibits a similar system behavior for 100 roads as for the load of 10-50 roads, with a median of 0.11 seconds.

Therefore, it can be stated that load balancing is beneficial for a scalable behavior. In concrete terms for the digital twin road, this would mean, for example, that roads should be distributed to several independent broker instances within the FIWARE-based platform based on their geographical location.

6. Conclusion

In this paper, we present a case study of a digital twin for road systems using the open-source platform FIWARE. The goal of this case study was to conduct an initial evaluation of the application potential of FIWARE for the technical implementation of a comprehensive digital twin road system. For this, we

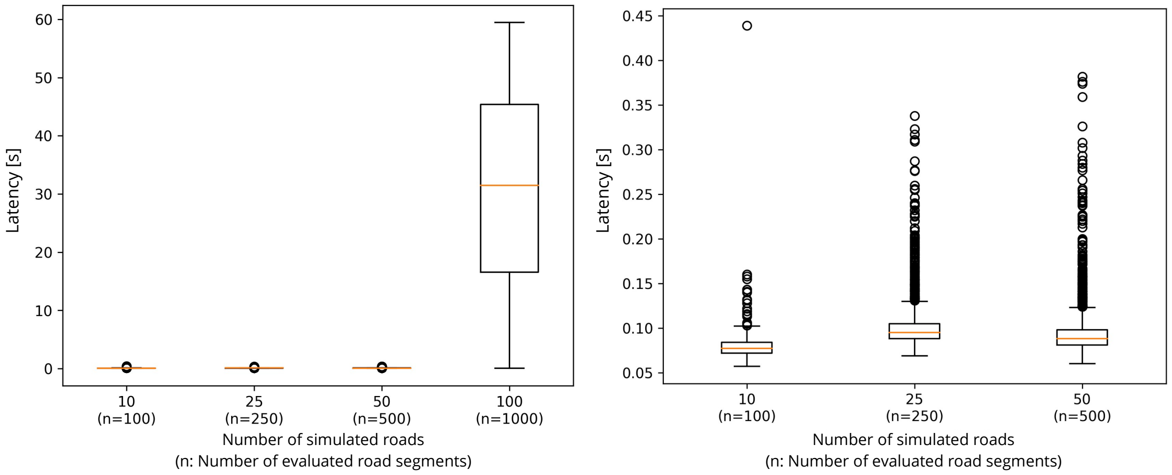

Figure 6: Latency of the calculation of the risk of slipperiness using two context broker.

presented two specific use cases and explained the implementation challenges for such a system: the distributed environment, the data heterogeneity, and the scalability issue. Then, we introduced our common system design implemented in FIWARE for both use cases and the central role of the Scorpio context broker, the used data model for road segments, and the configuration of sensors by example. To evaluate our developed prototype, we used a cloud deployment using AWS for the investigation of the scaling behavior when integrating an increasing number of roads. We observed, that with an increasing number of road segments, the latency of the calculation of the risk of slipperiness increased disproportionately and seems to be a bottleneck.

Returning to our research question formulated in Section 1, we can state the following: FIWARE can indeed be used to build a digital twin for road systems. However, attention must be given to load balancing when a large number of sensors and road segments are present by utilizing an appropriate number of independent context broker instances. In particular, the number of employed broker instances should be increased for increasing load to avoid high latencies.

Acknowledgments

The research work is supported by the German Research Foundation (SFB/TRR 339, Project ID 453596084).

Disclosure of Interests.

The authors declare no competing interests.

References

[1] M. Kaliske, R. Behnke, I. Wollny, Vision on a digital twin of the road-tire-vehicle system for future mobility, Tire Science and Technology 49 (2021) 2–18. doi:`10.2346/tire.21.190223`.

[2] O. El Marai, T. Taleb, J. Song, Roads infrastructure digital twin: A step toward smarter cities realization, IEEE Network 35 (2020) 136–143.

[3] R. Matchett, J. Wium, Digital twins for road infrastructure, in: 40th Annual Southern African Transport Conference, 2022. URL: https://repository.up.ac.za/handle/2263/87359.

[4] E. Ferko, A. Bucaioni, M. Behnam, Architecting digital twins, IEEE Access 10 (2022) 50335–50350. doi:https://doi.org/10.1109/ACCESS.2022.3172964.

[5] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, A. Nee, Enabling technologies and tools for digital twin, Journal of Manufacturing Systems 58 (2021) 3–21. doi:https://doi.org/10.1016/j.jmsy.2019.10.001.

[6] M. Bauer, FIWARE: standard-based open source components for cross-domain iot platforms, in: 8th IEEE World Forum on Internet of Things, WF-IoT 2022, Yokohama, Japan, October 26 - Nov. 11, 2022, IEEE, 2022, pp. 1–6.

[7] F. Cirillo, G. Solmaz, E. L. Berz, M. Bauer, B. Cheng, E. Kovacs, A standard-based open source iot platform: FIWARE, IEEE Internet Things Mag. 2 (2019) 12–18.

[8] ETSI, Context information management (cim); ngsi-ld api, https://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.05.01_60/gs_cim009v010501p.pdf, 2024.

[9] G. Privat, A. Medvedev, Guidelines for modelling with ngsi-ld, ETSI White Paper 42 (2021).

[10] Forschungsgesellschaft für Straßen- und Verkehrswesen (FGSV), Hinweise zur Erfassung und Nutzung von Umfelddaten in Streckenbeeinflussungsanlagen (unfortunately only available in German), ausgabe 2017 ed., FGSV Verlag GmbH, 2017.

[11] S. M. Piryonesi, T. E. El-Diraby, Data analytics in asset management: Cost-effective prediction of the pavement condition index, Journal of Infrastructure Systems 26 (2020). doi:https://doi.org/10.1061/(ASCE)IS.1943-555X.0000512.

[12] K. Chen, M. E. Torbaghan, M. Chu, L. Zhang, A. Garcia-Hernández, Identifying the most suitable machine learning approach for a road digital twin, Proceedings of the Institution of Civil Engineers - Smart Infrastructure and Construction 174 (2021) 88–101. doi:https://doi.org/10.1680/jsmic.22.00003.

[13] The FIWARE Foundation, Fiware, the open source platform for our smart digital future, https://www.fiware.org, 2024.

[14] ETSI, Context information management (cim); information model (mod0), https://www.etsi.org/deliver/etsi_gs/CIM/001_099/006/01.01.01_60/gs_CIM006v010101p.pdf, 2024.

[15] J. Hierro, FIWARE for Digital Twins, Position Paper, FIWARE Foundation, 2021. URL: https://www.fiware.org/wp-content/uploads/FF{_}PositionPaper{_}FIWARE4DigitalTwins.pdf.

[16] S. D. M. Initiative, Road smart data model, 2022. URL: https://github.com/smart-data-models/dataModel.Transportation/blob/078fd868ab7b59bc87d2a0fb83d8b6ea7b01b49f/Road/doc/spec.md.

[17] S. D. M. Initiative, Device smart data model, 2022. URL: https://github.com/smart-data-models/dataModel.Device/blob/9ce60e64de271a57931b18a340973c34b1a89672/Device/doc/spec.md.

[18] S. D. M. Initiative, Device measurement smart data model, 2022. URL: https://github.com/smart-data-models/dataModel.Device/blob/9ce60e64de271a57931b18a340973c34b1a89672/DeviceMeasurement/doc/spec.md.

[19] B. Cheng, G. Solmaz, F. Cirillo, E. Kovacs, K. Terasawa, A. Kitazawa, Fogflow: Easy programming of iot services over cloud and edges for smart cities, IEEE Internet Things J. 5 (2018) 696–707.

[20] FogFlow, Github fogflow, https://github.com/smartfog/fogflow, 2024.

[21] HashiCorp, Terraform, 2022. URL: https://www.terraform.io.

[22] Apache Software Foundation, Apache jmeter, 2022. URL: https://jmeter.apache.org.