IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 23 August 2023, accepted 21 September 2023, date of publication 25 September 2023,
date of current version 29 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3319079

RESEARCH ARTICLE

MaaS Blender: Integration of Mobility Simulators for Mobility as a Service Evaluation

YUKIHISA FUJITA[1], (Member, IEEE), MASAHIRO KUWAHARA[1], AND EIJI UEDA[2]
[1]InfoTech, Connected Advanced Development Division, Toyota Motor Corporation, Tokyo 100-0004, Japan
[2]Future Technology Laboratory Inc., Aichi 460-0007, Japan

Corresponding author: Yukihisa Fujita (yukihisa.fujita@toyota-tokyo.tech)

ABSTRACT With advances in information technology over the past decade, new mobility services have been developed and the concept of mobility as a service (MaaS) has become an achievable reality. There have been several attempts to implement MaaS, which is expected to be beneficial for users, practically. However, policy-makers have found estimating the effects of practical MaaS implementation challenging, as the integrated behavior of two or more mobility services is difficult to understand. In this study, we propose an event-based simulator integration mechanism to evaluate MaaS easily. The proposed mechanism attaches/detaches existing simulation models and/or mobility services via application programming interfaces (APIs). Moreover, it utilizes independent mechanisms to generate transportation demands and simulate user preferences. Two application scenarios are evaluated qualitatively, indicating that the proposed mechanism can successfully handle various scenarios such as private and small area MaaS and public and middle size area MaaS, and evaluate them from the viewpoints of transportation performance. The mechanism will improve user experience and service quality and enable realization and evaluation of practical MaaS.

INDEX TERMS Event drive architecture, mobility as a service, simulator integration, user and mobility behavior model.

I. INTRODUCTION

A. BACKGROUND

With the advancement of information technology, new types of mobility services (e.g., ride hailing, ride sharing) have been developed. Each service is typically operated by a servicer. In contrast, one typical version of the many forms of MaaS involves a MaaS operator (or a MaaS integrator) that integrates multiple mobility services as one single service [1], [2]. The MaaS operator acts as an intermediator between users and mobility service providers to provide users with a single endpoint for integrated mobility services. Such integration is expected to improve user experience and service quality.

MaaS has been implemented in several cases (e.g., [3]). However, standardized MaaS implementation methods have not been established owing to its complexity and operational costs. MaaS is typically complex because regional conditions, such as demands, user behavior trends, and existing regional transportation networks, affect its effectiveness. It is also expensive in terms of financial cost and time. Mobility services typically contribute to the infrastructure of daily life. Thus, evaluation or optimization of new designs by modifying real-world services frequently is not practical. Therefore, an effective but non-disruptive evaluation method for MaaS is required that considers several regional conditions without affecting the actual environment.

B. PURPOSE

To resolve the aforementioned problem, we propose a MaaS simulation tool to simulate various combinations of mobility services under several conditions. Moreover, to avoid replicating existing work, we propose an integration mechanism for existing mobility simulators. Several studies have been conducted on mobility simulators. SUMO [4] is a widely used simulator used to evaluate traffic flows on

The associate editor coordinating the review of this manuscript and approving it for publication was Jesus Felez.

large road networks. CityMOS [5] performs mobility service simulation from the perspective of urban design, and has been utilized for city design in Singapore. SUMO and CityMOS simulate various types of mobilities; however, they primarily focus on traffic flows, not combinations of mobility services, as in the case of MaaS. To evaluate the proposed MaaS design in terms of service quality using theses simulators, we need to construct new simulators for each mobility service. Müller et al. [6], [7] evaluated the effect of shared autonomous electric vehicles on traffic patterns in Vienna by adding new mobility features to the MATSim [8] simulator in a MATSim-specific manner. In contrast, in this study, we focus on combining existing simulators, without modifying them.

In this context, the major contributions of this study are the following:

- abstract event definitions of mobility services,
- API-based integration mechanism with the events.

In Section II, related works are analyzed from the perspective of mobility simulators and platform technologies as integration methods to clarify the research desideratum addressed by the proposed integration method. Then, we describe the requirements of the ideal MaaS simulator in Section III and the proposed integration mechanism satisfying them in Section IV. Subsequently, we evaluate the proposed mechanism in two applications in Section V and in terms of qualitative measures in Section VI. Finally, in Section VII, we summarize the study and discuss potential directions of future research.

II. RELATED WORK

Several tools have been proposed for mobility simulation. In addition to the previously mentioned SUMO [4] and CityMOS [5], tools such as VISSIM [9], AIMSUN [10], and MATSIM [8] can simulate mobility behaviors on macro-, meso-, and micro-scales. Although a new mobility service can be simulated based on these tools, its implementation and incorporation have to be performed in a specified manner in terms of programming languages, frameworks, and data formats. Thus, attaching/detaching existing simulation tools for specific mobility services used for other purposes is difficult. Moreover, these tools are typically used for the evaluation of traffic behaviors, e.g., traffic jams, and not to assess the performance of combinations of several mobility service. Hence, a new platform is required to attach/detach models easily and evaluate service performance of MaaS.

Some platforms contain common APIs and/or intermediary modules to attach/detach other modules and their specifications are publicly available, e.g., platforms such as "open architecture" or "open platforms." TrasMAPI [11] is such a platform that integrates different traffic simulations based on APIs. The purpose of TrasMAPI is to provide an environment to evaluate traffic models on several simulators. In other words, its purpose and architecture is for comparison between simulators, not combination. MOAST [12] is another platform that integrates real-time robotics control systems with simulators. To address problems arising from the differences between simulation environments and actual ones, integration and communication methods have been proposed to simulate actual environments accurately. However, the aforementioned studies typically focus on traffic behaviors or single mobility behaviors; hence, their modeling perspectives and required abstractions are different from MaaS evaluations. Hence, a mobility service abstraction method and an integration mechanism based on this abstraction are required for MaaS evaluation.

III. REQUIREMENTS

Although there are several case studies on MaaS implementation in specific areas [13], no standardized implementation methods exist for MaaS. Therefore, we hypothesize on the usage mode of a MaaS simulator to define its requirements.

To simulate MaaS, the MaaS simulator must contain several types of mobility services. To avoid replicating existing work, the MaaS simulator is not expected to implement all types of mobility services internally. It is required to cooperate with other existing mobility service simulators as external programs. This means existing simulators can be attached when the specific corresponding services are presented in simulation scenarios. Additionally, the utilization knowledge of the internal logic of mobility service simulators should be avoided during their attachment. Understanding the internal logic is a time-consuming task, similar to developing a new one, because mobility service simulators typically use their own operational algorithms and configurations. Therefore, the MaaS simulator is required to include an attaching/detaching mechanism for existing simulators without using their corresponding knowledge.

Second, the MaaS simulator is required to simulate various forms of MaaS, such as rural MaaS, MaaS for employees, and inter-city MaaS [14]. Mobility services included in these MaaS are also varied, e.g., public transportation such as trains, shared bicycles, on-demand buses, etc. These services are operated in a wide range, between in-factory regions and inter-city areas in size. Thus, the MaaS simulator is required to satisfy various transportation conditions, such as employees' movements in a factory, travelers' sightseeing in a rural area, and everyday commutes in a large and complex city. For example, factory workers need to travel to the factory every day because their movement demands depend on their contractual obligations, and not on available mobility services. Conversely, they decide commuting routes based on available mobility services. To handle such situations, demands have to be generated in simulation scenarios without information from attached mobility service simulators. Hence, the MaaS simulator is required to include mechanisms to consider such various scenario configurations independent of attached simulators. Based on the presented analysis above, we summarize the requirements of the MaaS simulator as follows:

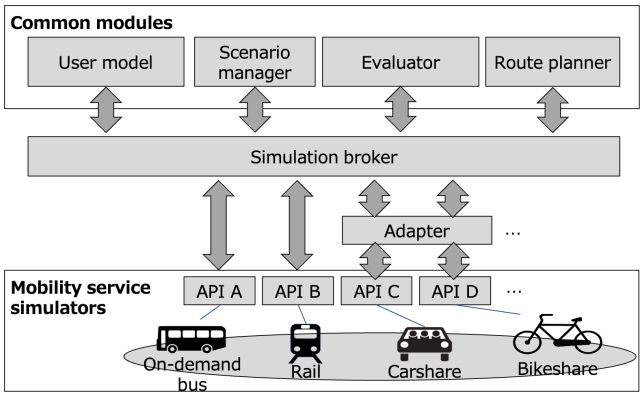

FIGURE 1. Conceptual architecture of the MaaS simulator constructed using the proposed integration mechanism.

A) attach and detach various types of mobility service simulators without additional knowledge about their internal logic,
B) manage various scenarios without considering attached simulators.

IV. INTEGRATION MECHANISM FOR MOBILITY SIMULATORS

A. FRAMEWORK

To satisfy the requirements described in the previous section, we propose a mobility simulator integration mechanism that consists of the following features:

1) Abstraction of mobility simulator behavior
2) Common scenario-management modules
3) API-based collaborations

To satisfy requirement A), we manage each mobility simulator based on its behavior, and not its internal logic; this enables us to handle various types of simulators. We model the inputs and outputs of mobility service simulators as events representing the abstract behavior of mobility service usage. This abstraction enables each mobility simulator to be handled without additional details, i.e., as a black box. To satisfy requirement B), we prepare common scenario management modules for demand generation, route planning, and route selection based on user preferences. The common modules are independent of the attached simulators to configure scenarios without additional information on attached mobility service simulators. Finally, to integrate the mobility service simulators and common modules, we propose API-based collaboration mechanisms. This enables integration with logic isolation.

Fig. 1 depicts the conceptual architecture of the MaaS simulator based on the proposed integration mechanism. In the figure, three categories of modules are illustrated—common modules, simulation brokers, and mobility service simulators. Similar to microservice patterns [15], simulation brokers connect common modules and mobility service simulators via APIs. Mobility service simulators are also connected via APIs. In the absence of the APIs required to function in this framework, we prepare adapters to convert the original interfaces of mobility service simulators to the required APIs.

TABLE 1. Abstraction mobility events transmitted between modules via APIs.

Subject	Name	Attributes
Scenario manager	new demand	user ID, event time preferred departure time (mobility service) origin/destination
User model	new reserve	user ID, event time preferred departure time origin/destination mobility service
	depart	user ID, event time, mobility service
Mobility service simulator	reserved	user ID, event time scheduled departure time scheduled arrival time origin/destination success, mobility ID
	departed	(user ID), event time, mobility ID departed location
	arrived	(user ID), event time, mobility ID arrived location

B. ABSTRACTION OF MOBILITY SIMULATOR BEHAVIOR

We model simulator behaviors as events to handle various types of simulators without additional knowledge regarding their internal logic and implementations. While utilizing existing simulators, particularly ones developed by third parties, complete knowledge about internal logics and implementations is unlikely. To avoid such time-consuming tasks, we use only inputs and outputs of the simulators. Additionally, we assume that simulators for mobility services exhibit similar behavior. Typically, such simulators require inputs, such as specific demands (i.e., origin/destination of user) and users' decisions, and output results such as users' movement results and evaluation criteria. We model such similar behavior as abstract events transmitted between modules described in Fig. 1. These events can be converted from/to simulators' outputs and inputs.

Table 1 lists the defined abstraction events. As we focus on not only mobility service performance but also MaaS usability, the events are assigned user identification (ID) numbers as common attributes to detect user-related activities. Conversely, mobility specific behaviors not associated to movement-related activities, such as fuel supply and maintenance, are not represented by events. Thus, such behaviors are expected to be processed internally by each mobility service simulator. All events correspond to one subjective module, and their conceptual meanings are different corresponding to each subject. For example, the two events, "depart" used by user model and "departed" used by mobility service simulators, are similar. In the proposed mechanism, actual movements are managed by each mobility service simulator, and the user model is expected to manage users' intentions, e.g., "I will go to location X using mobility service Y." Hence, events generated by user models

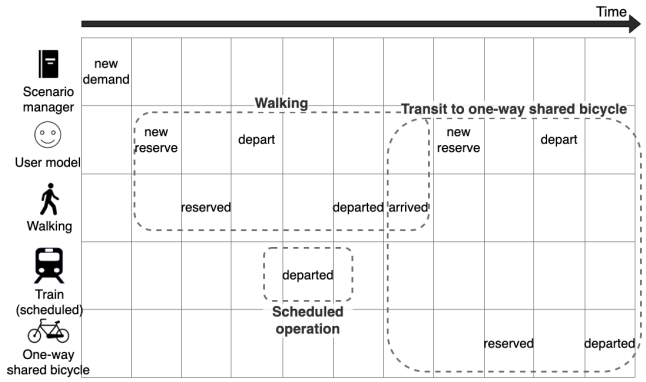

FIGURE 2. Example of event flow for a user undergoing movement via walking and a one-way shared bicycle.

are referred to as ''start of movement.'' Mobility service simulators manage actual moves and utilize events such as ''departed,'' which are operational in nature. Notably, a ''new demand'' event may correspond to a mobility service (e.g., bus, bicycle), even though the scenario manager is independent of mobility service simulators. The attribute mobility service is designed to simulate recorded demands when actual demands are introduced and selection results are obtained from the collected data.

To instantiate event usage, Fig. 2 depicts the event flow of one user. We assume that there are three mobility service simulators (walking, train, and one-way shared bicycle) and the user has a movement demand. The user decides to undergo movement by a combination of walking and using a one-way shared bicycle to satisfy the demand. First, the user triggers the ''new reserve'' event to confirm the availability of walking. This reservation is not an actual reservation in the real world, but merely a confirmation of availability (i.e., walkable). In the case of other mobility service simulators, reservations may correspond to actual reservations if this service is provided by the mobility service simulators. Then, the ''depart'' and ''departed'' events are triggered by the user and walking, respectively. After walking triggers the ''arrived'' event, the user triggers the ''new reserve'' event again to transition to a one-way shared bicycle. The triggering of these events represents the user's movement via multiple mobility services. In contrast, in Fig. 2, the train triggers the ''departed'' event irrespective of the events triggered by the user. Scheduled mobility services, such as the departure of the train, are executed as per schedule even in the absence of passengers. Managing events that are not triggers for other events directly is important to notify mobility service statuses to users. In this case, after the train triggers the ''departed'' event, other modules perform internal updates, such as changing future demands based on the scenario manager, and re-evaluating current route plans by the user model. Hence, some mobility service simulators triggers events as scheduled. Using these events, we represent behaviors of users and mobility services.

TABLE 2. List of APIs achieving event-driven simulation execution.

Name	Description	Parameter	Return Value
peek	Obtain start time of events	-	time
step	Process specified event	time	events
triggered	Notify event execution	event	-
plan	Obtain candidate routes	demand	routes

C. COMMON SCENARIO-MANAGEMENT MODULES

To simulate various scenarios without additional details about the attached simulators, the scenario-related modules are designed to be independent from the simulators. As scenario-related modules, we define the following user-side modules:

- scenario manager, which generates users' movement demands,
- route planner, which suggests candidate routes for the demands,
- user model, which selects one of the candidate routes.

To manage demands without considering additional details about mobility services, the scenario manager module incorporates the expected departure time and simple pair of origin and destination within a single demand. It also provides demands from pre-defined or randomly generated lists. When a new demand event is triggered by the scenario manager, the user model selects the preferred route from the candidates calculated by route planners. The user model can manage users as heterogeneous entities, and each user is considered to have a preference, e.g., routes with shortest travel time, smallest expense, or minimum walking distance. Other preference models are used when the model modules accept our APIs and events. Candidate routes are calculated by the route planners based on operation data obtained from each mobility service simulator. We adopt GTFS [16], GBFS [17] and GTFS-flex [18] as the standard data formats of operation data, and share them between the route planner and mobility service simulators. Using these three modules, various scenarios can be simulated without considering the internal logic of mobility services.

D. API-BASED COLLABORATIONS

To transmit events between common scenario management modules and mobility service simulators, we use APIs and a simulation broker module. Table 2 presents the APIs processing events transmitted between modules. For scenario executions, ''peek'' and ''step'' are defined in terms of the mode of identification and execution of the next event, and ''triggered'' represents an execution notification transmitted by the simulation broker to all other modules except the route planners. Additionally, ''plan'' is used to identify route candidates for specified demands; it is implemented solely in the route planner. We minimize the number of common APIs to reduce implementation costs. Instead of preparing numerous common APIs, we define abstraction mobility events to represent various activities in simulations.

To simulate scenarios using APIs, the simulation broker follows the following steps:

1) it collects the occurrence times of future events (i.e., scheduled triggering time) corresponding to each module
2) it determines the next event by calculating the earliest event among the collected events,
3) it notifies the owner module of the next event of the beginning of the event,
4) it receives an event completion intimation from the event owner module,
5) it transmits event processing results to all other modules.

Owing to the various types of modules triggering corresponding processes as events, the triggering times and orders need to be managed accurately. To this end, the simulation broker collects the occurrence times of all events from their respective modules via peek API. After deciding the events to be processed in the subsequent step, the simulation broker notifies the event owner module to process the event, which, in turn, returns the results to the simulation broker. Subsequently, the simulation broker transmits the event processing results to all other modules. Receiving the results may trigger processes for other modules, such as stepping times of simulation in mobility service simulators, generating new demands, and/or updating internal states.

One scenario is simulated by iterating the aforementioned flow until the simulation broker cannot obtain any event from the other modules. This is an event-driven module; i.e., each module triggers events in accordance with the given scenario and/or current status. Although some mobility service simulators may adopt a time-driven method, in which the internal process is managed after fixed time intervals (e.g., hourly, daily), we can convert them to an event-driven method easily using adapters to trigger events at a specific frequency.

V. EVALUATION OF APPLICATION CASES

A. OVERVIEW

To evaluate the proposed integration mechanism, the following two scenarios are simulated:

1) private and small area: corporate MaaS in a factory,
2) public and mid-size area: regional MaaS in a city.

Whereas corporate MaaS operates within a factory and focuses on movement efficiency of employees, regional MaaS operates in a mid-size area and focuses on benefit improvement for citizens. In other words, these two cases are similar in terms of combinations of mobility services but different in terms of service operations and evaluation criteria. Using these two scenarios, we evaluate the ability of the proposed integration mechanism to simulate various cases, particularly for events and APIs.

As both cases involve two or more mobility services, three mobility simulators are prepared to represent the different types of mobility services, as described in Table 3. The simulator for fixed routes & times operates mobilities as per a schedule defined by timetables and routes in GTFS. The one-way simulator focuses on one-way mobility services and manages their statuses (e.g., current location, battery status, and reservation) and facility information (e.g., location and availability of parking lots). The simulator assigns available mobility to users based on their requests. The on-demand simulator manages the statuses of mobilities in the same way as the one-way simulator; however, its input information is obtained from GTFS-Flex, and it exhibits matching logic between mobilities and users to represent sharing services. While matching problems are typically formulated as vehicle routing problems or its expansions [19], we utilize simple matching logic to determine sharing details if at least one mobility can reach the user's location within their expected departure time with a specific time window. The simulators described in Table 3 account for all major mobility service types.

TABLE 3. Prepared mobility service simulators and their representation services.

Name	Representation Services
Fixed routes & times simulator	Train, bus, LRT, BRT
One-way simulator	Shared-car, shared-bicycle
On-demand simulator	Taxi, on-demand bus, ride sharing

TABLE 4. Common scenario management module configurations for each scenario.

	Corporate MaaS	Regional MaaS
Scenario manager	demand generation based on collected data	random demand generation
User preference model	fastest arrival	fastest arrival & minimum walking
Route planner	shortest path without transits	multi-modal path planning

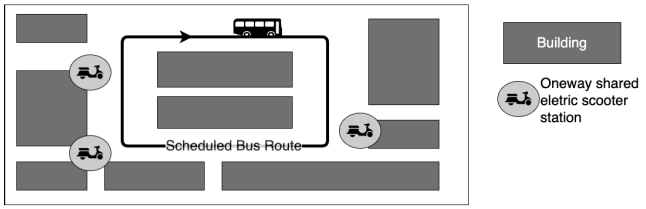

FIGURE 3. Abstract image of corporate MaaS in a large factory owned by Toyota Motor Kyushu Corporation. There are two types of mobility services: scheduled bus and one-way shared electric scooter.

In the case of common scenario management modules, Table 4 presents an overview of common scenario management module configurations for each scenario. As each scenario has different users and different purposes, different configurations are used for each scenario.

B. SCENARIO 1: PRIVATE AND SMALL AREA

1) SITUATION AND SCENARIO SETTINGS

In the scenario involving the private and small area (i.e., corporate MaaS scenario), MaaS is simulated in Toyota Motor Kyushu, Inc. The "Miyata Factory" is approximately 1km^2 in size, with free MaaS implemented for employees. Fig. 3 depicts a service image of the MaaS system as of February, 2022. The implemented MaaS comprises a

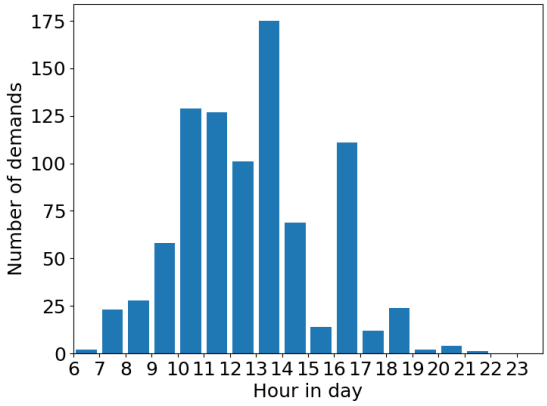

FIGURE 4. Number of movement demands in one day based on collected data in scenario 1.

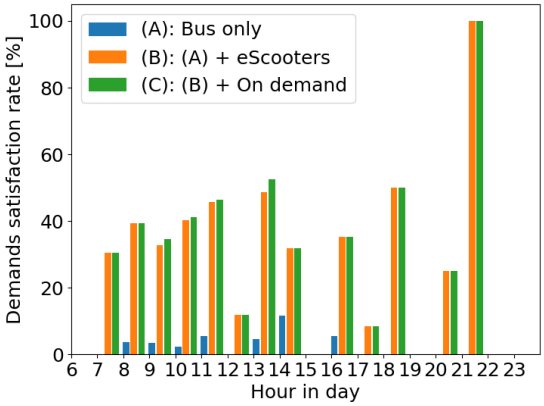

FIGURE 5. Demand satisfaction rate in scenario 1.

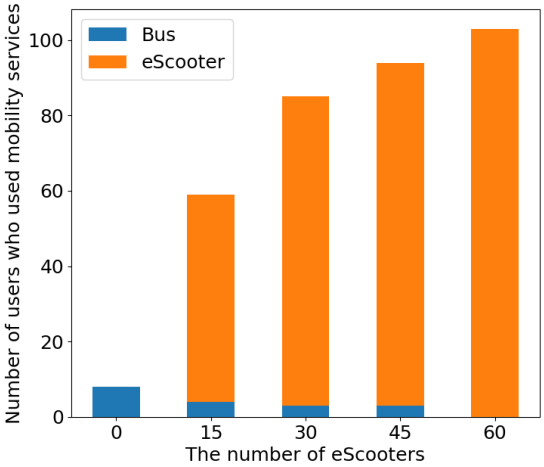

FIGURE 6. Number of users corresponding to each mobility service between 13:00 and 14:00 in scenario 1 with respect to the number of available eScooters.

one-way shared electric scooter service (henceforth referred to as eScooter) and fixed-route buses running as per fixed schedules. We consider an extended version of this MaaS as an application case, with an on-demand service in addition to scheduled bus and eScooter services. The number of buses and on-demand services are both set to 1 and their passenger capacities are 6 and 5, respectively. Additionally, 30 eScooters are used with 11 stations. Fixed routes & times simulator, on-demand simulator, and one-way simulator are used to represent fixed-route bus service, on-demand service, and eScooter service, respectively.

To simulate this situation, employees' demands regarding movement within the factory are estimated using Bluetooth sensors that detect employees' locations within the factory. Prospective movement demands are estimated following a method comparable to using Wifi packets [20]. The number of demands estimated based on the collected data in one day is depicted in Fig. 4. The number of movement demands are observed to vary with respect to time. Considering that the users in this case are factory employees, we adopt the fastest arrival selection as the user model. Candidate routes transmitted to users are calculated in terms of the shortest path based on the network generated by bus stops and eScooter stations.

The demand satisfaction rate is evaluated in terms of usability and the number of users for each mobility service is evaluated in terms of the cost performance of their MaaS. Because the implemented MaaS is free, employees use the services when they are available at appropriate times and locations. Hence, the demand satisfaction rate is observed to be nearly identical to usability. In the case of cost performance, the company pays all introduction and operation costs, and they discontinue the services in the absence of users. Although actual costs cannot be obtained via abstract simulation, we compare the relative values, such as the number of users in various cases corresponding to various MaaS configurations. The cost performance can be calculated in detail by adding cost information to such results. Therefore, in this scenario, the number of users corresponding to each mobility service represents the cost performance.

2) RESULTS

Fig. 5 presents the demand satisfaction rate for each mobility service combination: (A) bus only, (B) eScooters in addition to (A), and (C) on-demand transportation in addition to (B). We observe that eScooters lead to a significant increase in the demand satisfaction rate, whereas the other two services have limited effects. Almost 50% of all demands are satisfied during the peak time, between 13:00 and 14:00. One-way shared mobility services, e.g., eScooters, are typically highly beneficial for the movement of individuals over short distances. In contrast to the movement of families in a city which move as a group over long distance, employees move individually and they do require transportation over short distance in the factory. Hence, eScooters improve the demand satisfaction rate significantly.

Fig. 6 depicts the number of users corresponding to each mobility service with respect to the number of available eScooters, pertaining to the aforementioned case (B),

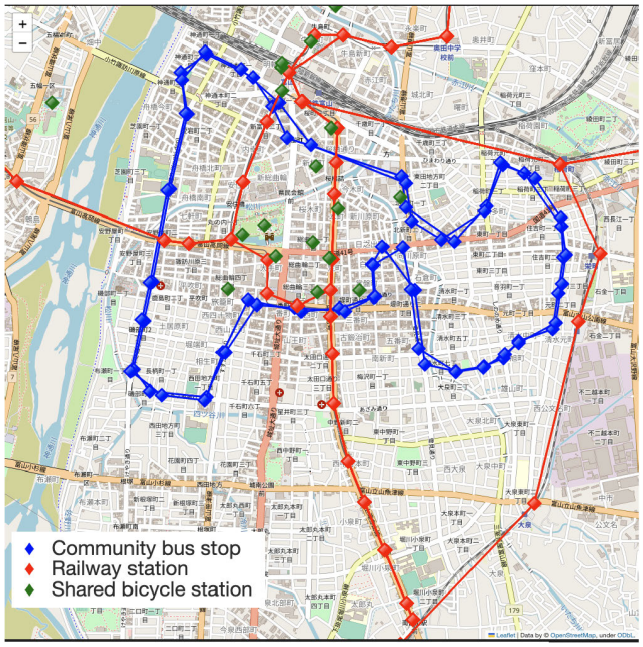

FIGURE 7. Transportation network used for evaluation in Toyama city.

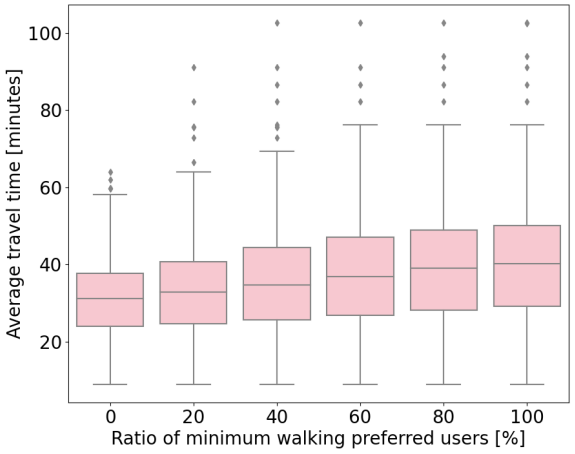

FIGURE 8. Variation in average travel time including waiting time with respect to the number of users with a preference for minimum walking time. The results exclude the case of walking only.

between 13:00 and 14:00. As the number of eScooters is increased, the total number of users increases, whereas the number of bus users decreases. Both bus and on-demand services require employees to wait until arrival of the respective vehicles. This has a negative impact on the calculation of candidate paths to the destinations, and employees prefer walking or using eScooters when they are available.

Moreover, by Fig. 6, the increase in the total number of users is not linear; it becomes saturated quickly owing to the limited number of employees. As noted above, optimal cost performance may be achieved using this result in combination with the introduction and operation costs of MaaS.

These results imply that the proposed MaaS simulator qualitatively evaluates the performance of various mobility service combinations effectively in the case of Scenario 1.

C. SCENARIO 2: PUBLIC AND MID-SIZE AREA

1) SITUATION AND SCENARIO CONFIGURATIONS

The south area of Toyama city is selected as a public and mid-size area for evaluation (i.e., regional MaaS scenario). Toyama city contains several types of public transportation and a private shared bicycle service is available in the south area (spanning approximately 9 km^2). Fig. 7 illustrates the current transportation network in Toyama city. We consider one railway, one community bus, and the shared bicycle service as available mobility services, along with their operational areas. As some of the transportation data of Toyama city is available in GTFS as public data, they are used to evaluate the proposed mechanisms. The other data are prepared manually based on public information as of February 2023. Fixed routes & times simulator is used to represent railway service and community bus service. One-way simulator is used to represent shared bicycle service.

In this scenario, citizens' movement activities are analyzed within the current transportation network, without considering precise data, such as detailed demands, users' preferences, and their actual behaviors. This is a typical exercise for the improvement of future transportation, and the available data are typically insufficient even for local municipalities. Hence, various configurations should be prepared to calculate the ranges of evaluation criteria. The total number of mobility service users and average travel time corresponding to various demand volumes are used as evaluation metrics.

Demands are randomly generated based on user preferences. During the evaluation of transportation services in public areas, it is difficult to have complete knowledge about user preferences. In this context, estimated preferences or various ranges of preferences may be used. In this case, we adopt the latter approach. Two user models, fastest arrival and minimum walking, are prepared, and evaluated by changing the ratio of users. Financial cost may be considered to be a significant factor in user preferences, in addition to travel time and walking time. Conversely, the aim of this simulation is not to contribute to Toyama's transportation services but to evaluate the proposed mechanisms. Hence, we consider only two preference models.

2) RESULTS

Fig. 8 depicts the average travel time, including waiting time for each case with demands including one of two preferences: minimum walking or fastest arrival. As depicted in Fig. 8, an increase in the number of demands with minimum walking preference increases the average travel time. Using mobility services is not always faster than walking because of the additional waiting time in the former case. Moreover, longer distances are sometimes involved compared to direct walking

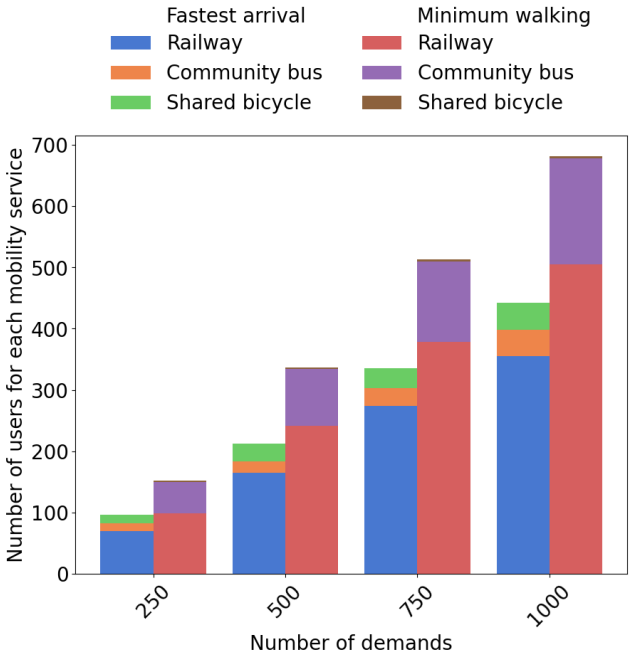

FIGURE 9. Number of mobility service users with 250–1000 total number of demands. The left bar represents the fastest arrival only case, and the right bar represents the minimum walking only case.


```
{
    "eventType": "DEPARTED",
    "source": "Railway",
    "time": 614.0,
    "details": {
        "userId": "U_3",
        "location": {
            "locationId": "10034",
            "lat": 36.68807,
            "lng": 137.232478
        },
        "mobilityId": "TRAM_1"
    }
},{
    "eventType": "ARRIVED",
    "source": "Railway",
    "time": 621.0,
    "details": {
        "userId": "U_3",
        "location": {
            "locationId": "10036",
            "lat": 36.670531,
            "lng": 137.220413
        },
        "mobilityId": "TRAM_1"
    }
}
```


FIGURE 10. Example of event log of scenario 2 (Sources and IDs are translated from Japanese to English for ease of comprehension).

routes. Therefore, an increase in such demands increases the average travel time. The number of such demands depend on the profiles of citizens. People who want to avoid walking, such as elderly people, people with disabilities, and people with children, prefer minimum-walking routes. Conversely, younger people prefer fastest arrival routes. As depicted in Fig. 8, different ratios correspond to different results. Hence, we consider both preferences when precise data cannot be used.

Fig. 9 depicts the number of mobility service users with 250–1000 demands. As the number of mobility service users increases with an increase in the number of total demands, the current transportation network is not completely saturated. Conversely, usage of the community bus and the shared bicycle service corresponds to completely different preferences. In the case of a community bus, the number of users in the minimum walking scenario is more than that in the fastest arrival scenario. As community buses are less frequent than other bus services or railways, it is unsuitable for fast movement. In the case of the shared bicycle service, the number of users is almost zero in the minimum walking only case, unlike that in the fastest arrival only case. The number of shared bicycle stations is limited in this scenario and they are located near the railway station. People can move faster via shared bicycles than railways owing to the availability of shortcuts within the small loop line area; however, this requires the users to walk to the shared bicycle stations. People with walking minimum preference avoid service transfer as transfer requires walking between stations. Hence, the number of users is almost zero in the minimum walking only case.

These results imply that the proposed MaaS simulator compares and evaluates various cases effectively using ranges of criteria values when precise data are not available.

VI. QUALITATIVE EVALUATION

A. CALCULABLE EVALUATION CRITERIA

In the proposed mechanism, the only outputs of simulations are event logs. The events comprise abstracted information of the behaviors of users and mobilities, and the various evaluation criteria are calculated based on them. Fig. 10 depicts an example of the events generated in scenario 2, highlighting a case in which user U_3 takes tram TRAM_1 (Railway). We can extract information from these events, such as the time of travel and movement distance, the number of transits during one trip, etc. From the extracted information, we may calculate user-centric criteria, such as travel time, user waiting time for specific mobility services, and user waiting time for entire trips, which are typically used in transportation analysis [21]. Moreover, by treating the events logs as spatial and temporal data, we can calculate the performances of combinations of mobility services in terms of the number of users availing specific transportation methods in specific time windows, failures of service due to overburdened capacities, and the contribution of each mobility services.

Additionally, the proposed mechanism accepts additional modules to collect events and interact with other modules during simulations. By receiving the event data from the simulation broker, modules deduce current statuses in scenarios, and verify various statuses or interact with other modules if required. This enables the calculation of other

TABLE 5. Feature comparison between the proposed mechanism and other simulators.

	Proposed mechanism	SUMO [4]	MaaSSim [23]
Evaluation target	MaaS performance	Traffic flow	Ride-sharing matching mechanisms
Model execution mechanism	APIs	Built-in process	Built-in process
Minimum resolution	User and mobility behavior Multi-modal routing candidates	Mobility behavior Internal state of mobility	User and driver behaviors Internal states of user and mobility Matching details
Typical criteria	Travel time User waiting time Demand satisfaction rate	Travel time Vehicle speed Emissions	User waiting time Vehicle kilometers per driver Revenue

criteria, such as the number of routes, candidate users, and users' decisions based on the candidates. These criteria clarify user-centric factors, which are important in MaaS [22]. In summary, by utilizing the event logs and event-driven mechanisms, we can calculate both typical criteria and MaaS specific criteria.

B. FEATURE COMPARISON

As the proposed mechanism utilizes existing simulators, it has certain advantages and disadvantages compared to other ordinal simulators. Table 5 presents a comparison between the proposed mechanism and other simulators. SUMO [4] is used as a traffic flow simulator and MAASSim [23] is used as a mobility service simulator to evaluate newer services such as ride-sharing.

Evidently, from Table 5, the purposes of the different simulations are different even though they all model mobility behaviors. The aim of this paper is to evaluate MaaS performance, which requires handling various mobility services. Hence, we adopt an API-based mechanism. Although this can analyze user and mobility behavior, it cannot analyze their internal states, e.g., remaining fuel amounts, driver status, and ride-sharing matching mechanism results. In the proposed mechanism, such states are handled by each mobility simulator internally, enabling the mechanism to operate without deep knowledge about these factors. There is a trade-off relationship between the ease-of-handling of various simulators and detailed resolutions of internal states. Thus, in cases in which internal states must be analyzed in detail to understand factors, such as the relationship between vehicle speed and emissions, and the influence of drivers preferences on matching results, the proposed mechanism cannot be used. Conversely, the purpose of the proposed mechanism is to evaluate MaaS, which is a combination of mobility services. This does not require detailed information on internal states; the focus instead is on integration of mobility services. Thus, in this case, we adopt an API-based mechanism satisfying the trade-off.

Moreover, the mechanism selection defines calculable evaluation criteria. As described in Section VI-A, the proposed mechanism calculates both typical and MaaS specific criteria effectively. Conversely, it cannot calculate criteria using internal states and purpose-specific criteria, e.g., emissions and kilometers per drive. This is because we prioritize achieving ease-of-mobility integration and mobility combination during performance evaluation. Despite the disadvantages of the proposed mechanism, we can calculate criteria required to evaluate MaaS as discussed in Section V. This implies that the proposed mechanism contains sufficient features to evaluate MaaS, even though it is not superior to other simulators.

VII. CONCLUSION

In this paper, we propose a simulator integration mechanism to evaluate MaaS, which is a combination of mobility services. The mechanism comprises the abstraction of behaviors of users and mobilities, followed by API-based interaction between simulators and proposed modules. Two application cases are considered, and the qualitative evaluation results imply that the proposed mechanism evaluate MaaS effectively in terms of transportation performance.

In future works, we intend to address the following issues:

1) evaluate more complex scenarios,
2) verify influence of differences in simulators' resolutions on performance,
3) introduce shared infrastructures.

Although we evaluated two scenarios in this paper, there are more complex scenarios such as inter-city MaaS. To simulate the inter-city MaaS, we need to consider both intra-city mobilities and inter-city mobilities. This means that micro-level mobilities and macro-level mobilities are mixed in one route. Therefore, we need to clarify that our proposed mechanism are capable for such scenarios. In the case of the second problem, the spatial and temporal resolutions of different simulators are often different. Although the proposed mechanism can accept them programmatically, integrity of simulation results is not guaranteed. Hence, we need to investigate such cases and develop a new mechanism to guarantee integrity. In the case of the third problem, various utilities have become shared owing to the development of a sharing economy, e.g., shared bicycle services, parking lots, and charging stations. This involves the use of the same infrastructure by two or more mobility services, leading to potential resource conflict. The proposed mechanism considers each mobility service to be isolated. Thus, we need to develop a shared infrastructure mechanism to simulate such cases.

REFERENCES

[1] M. Kamargianni and M. Matyas, "The business ecosystem of mobility-as-a-service," in *Proc. 96th Transp. Res. Board (TRB) Annu. Meeting*, Washington, DC, USA, Jan. 2017, pp. 1–14.
[2] J. Sochor, H. Arby, I. C. M. Karlsson, and S. Sarasini, "A topological approach to mobility as a service: A proposed tool for understanding requirements and effects, and for aiding the integration of societal goals," *Res. Transp. Bus. Manage.*, vol. 27, pp. 3–14, Jun. 2018.
[3] P. Jittrapirom, V. Marchau, R. van der Heijden, and H. Meurs, "Dynamic adaptive policymaking for implementing mobility-as—A service (MaaS)," *Res. Transp. Bus. Manage.*, vol. 27, pp. 46–55, Jun. 2018.
[4] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wiessner, "Microscopic traffic simulation using SUMO," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2575–2582.
[5] D. Zehe, S. Nair, A. Knoll, and D. Eckhoff, "Towards CityMoS: A coupled city-scale mobility simulation framework," in *Proc. 5th GI/ITG KuVS Fachgespräch Inter-Vehicle Commun. (FG-IVC)*, 2017, p. 3.
[6] J. Müller, M. Straub, G. Richter, S. Peer, and C. Rudloff, "MATSim model Vienna: Analyzing the socioeconomic impacts for different fleet sizes and pricing schemes of shared autonomous electric vehicles," in *Proc. Transp. Res. Board 100th Annu. Meeting*, Jan. 2021, pp. 1–22.
[7] J. Müller, M. Straub, G. Richter, and C. Rudloff, "Integration of different mobility behaviors and intermodal trips in MATSim," *Sustainability*, vol. 14, no. 1, p. 428, Dec. 2021.
[8] A. Horni, K. Nagel, and K. W. Axhausen, "Introducing MATSim," in *The Multi-Agent Transport Simulation MATSim*, A. Horni, K. Nagel, and K. W. Axhausen, Eds. London, U.K.: Ubiquity Press, Aug. 2016, pp. 3–8.
[9] M. Fellendorf and P. Vortisch, "Microscopic traffic flow simulator VISSIM," in *Fundamentals of Traffic Simulation* (International Series in Operations Research & Management Science), vol. 145, J. Barceló, Ed. New York, NY, USA: Springer, 2010, pp. 63–93.
[10] J. Casas, J. L. Ferrer, D. Garcia, J. Perarnau, and A. Torday, "Traffic simulation with Aimsun," in *Fundamentals of Traffic Simulation*, vol. 145, J. Barceló, Ed. New York, NY, USA: Springer, 2010, pp. 173–232.
[11] I. J. P. M. Timoteo, M. R. Araujo, R. J. F. Rossetti, and E. C. Oliveira, "TraSMAPI: An API oriented towards multi-agent systems real-time interaction with multiple traffic simulators," in *Proc. 13th Int. IEEE Conf. Intell. Transp. Syst.*, Funchal, Madeira Island, Portugal, Sep. 2010, pp. 1183–1188.
[12] S. Balakirsky, C. Scrapper, and E. Messina, "Mobility open architecture simulation and tools environment," in *Proc. Int. Conf. Integr. Knowl. Intensive Multi-Agent Syst.*, 2005, pp. 175–180.
[13] G. Smith, J. Sochor, and I. C. M. Karlsson, "Intermediary MaaS integrators: A case study on hopes and fears," *Transp. Res. A, Policy Pract.*, vol. 131, pp. 163–177, Jan. 2020.
[14] J. Sochor, "Piecing together the puzzle: Mobility as a service from the user and service design perspectives," *Int. Transp. Forum Discuss. Papers*, vols. 8–2021, no. 2021/08, pp. 1–28, Jan. 2021. [Online]. Available: https://www.oecd-ilibrary.org/transport/piecing-together-the-puzzle_1c7b4c5b-en
[15] C. Richardson, *Microservices Patterns: With Examples in Java*, 1st ed. Shelter Island, NY, USA: Manning Publications, Nov. 2018.
[16] Google LLC. *GTFS Static Overview | Static Transit*. Accessed: Mar. 24, 2022. [Online]. Available: https://developers.google.com/transit/gtfs
[17] NABSA. *General Bikeshare Feed Specification*. Accessed: Mar. 24, 2022. [Online]. Available: https://github.com/NABSA/gbfs
[18] GTFS-Flex Contributors. *GTFS-Flex*. Accessed: Dec. 10, 2022. [Online]. Available: https://github.com/MobilityData/gtfs-flex
[19] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Comput. Ind. Eng.*, vol. 99, pp. 300–313, Sep. 2016.
[20] Y. Gao and J.-D. Schmöcker, "Estimation of walking patterns in a touristic area with Wi-Fi packet sensors," *Transp. Res. C, Emerg. Technol.*, vol. 128, Jul. 2021, Art. no. 103219.
[21] J. De Vos, A. Ermagun, and F. A. Shaw, "Wait time, travel time and waiting during travel: Existing research and future directions," *Transp. Rev.*, vol. 43, pp. 1–6, Jun. 2023.
[22] P. Jittrapirom, V. Caiati, A.-M. Feneri, S. Ebrahimigharehbaghi, M. J. A. González, and J. Narayan, "Mobility as a service: A critical review of definitions, assessments of schemes, and key challenges," *Urban Planning*, vol. 2, no. 2, pp. 13–25, Jun. 2017.
[23] R. Kucharski and O. Cats, "Simulating two-sided mobility platforms with MaaSSim," *PLoS ONE*, vol. 17, no. 6, Jun. 2022, Art. no. e0269682.

YUKIHISA FUJITA (Member, IEEE) received the Ph.D. degree in information science from Nagoya University, in 2010.

Since 2020, he has been a Senior Researcher with Toyota Motor Corporation. He studies dynamics in transportation and city-scale activities from the viewpoints of human behavior and their ecosystem. Prior to Toyota Motor Corporation, he was a Researcher in a manufacturing company and he researched in RPA, information retrieval, and logistics operations. He was also a data scientist in a consulting firm and worked in several areas, such as telecommunication, retail, and public sector.

MASAHIRO KUWAHARA received the Master of Science degree in adaptive machine systems from Osaka University, in 2003, and the Ph.D. degree in urban transport from Kyoto University, in 2021. He has been a Senior Researcher with Toyota InfoTech and Toyota Motor Corporation, since 2006. His research interests include simulation of MaaS and analysis of several type of data like connected-car data and the IoT sensor data.

EIJI UEDA received the Ph.D. degree in science (physics) from Nagoya University, in 1998. Since 2000, he has been a Software Engineer with Future Technology Laboratories Inc.