Implementing a Mobility Platform to Connect Hubs in Rural Areas

E. Neidhardt

***Abstract*—**Mobility, for many people, is an important factor in the satisfaction of daily needs and many people are dependent on public transport. In rural areas with a low population density, it is difficult to provide public transportation with sufficient coverage and frequency. Therefore, the available public transport is unattractive. As a result, people use their own car, which is not desirable from a sustainable point of view and not possible for children and elderly people. Sometimes people organize themselves and volunteer transport services are created. These services are similar to demand-oriented taxis. However, these transport services are usually independent from each other and from the available line-based public transport, limiting both their usability and sustainability. We have developed a platform to improve usability and sustainability by connecting the different demand-oriented transport offerings with the line-based public transport. The system was implemented and tested in a rural area in Germany, but the SARS-CoV-2 pandemic limited real live operation.

***Keywords*—**Demand-oriented, HubChain, living lab, public transport.

I. Introduction

With conventional, line-based systems for public transport, it is often difficult to achieve a sufficient degree of flexibility. These rigid approaches are well applicable in large cities because there is a constant demand for them. In rural and suburban areas, however, their use is not practical. The demand is too low, and the use of line-based transport is very costly. As a result, few trips are offered. This leads to an unattractive system, which is even less used [2].

With demand-oriented transport approaches, an attempt is being made to offer a more cost-efficient alternative. Empty runs and unnecessary detours should be avoided and at the same time travel times should be shortened. The aim is to create an attractive mobility offer. The primary focus are people without a vehicle of their own. If car owners can be persuaded to change vehicles, this would be an additional bonus. Nevertheless, it is important that the system does not compete with the existing public transportation. Furthermore, a call taxi is also not desirable, as this would put you in competition with the taxi companies. Moreover, such a system would not be very sustainable. Our aim is to improve the attractiveness of public transport by coupling it with demand-oriented services.

The area of our research was the spring area around the river Elde in Mecklenburg-Western Pomerania. The area is a very rural area with sparse and elderly population. The system is implemented and tested in a living lab, which is research cooperation between civil society and science. This kind of cooperation emphasizes research in a hands-on environment [1].

The following sections are going to provide a brief overview of a previous living lab we conducted, as well as the area of operation. The main part of this paper is going to tap into the technical aspects of the implementation. Organizational and legal aspects are only briefly covered.

II. Previous Research

There have been many demand-oriented systems before. Many of these, also called Demand Responsive Systems, fall between private car and a regular, line-based bus service. In the UK, some of these systems emerged already in 1970 [3]. Many of these systems failed for various reasons. Some failed due to high-costs or low commitment by political authorities. Another reason is insufficient reliability [4].

We conducted a living lab with a similar approach, but different research questions before. In living lab Schorndorf, our goal was to improve sustainability of existing public transport by adding a demand-oriented approach. In times of low demand, the regular, line-based bus service was replaced with a demand-oriented bus service. The primary challenge of living lab Schorndorf was the acceptance of the customers and the legal and administrative challenges. Although the system was able to reduce the number of empty runs, the passenger numbers were also lower. Some passengers were rejecting the system for various reasons [5].

III. Area of Operation

The area of operation was the Elde region in Mecklenburg-Western Pomerania. The Elde region is a sparsely populated area. The average age is quite old. Public transportation is almost non-existent. This makes it exceedingly difficult for elderly people to fulfill their daily needs. Usually, they must rely on their children or neighbors to help them. Owning a personal car is therefore particularly important for many inhabitants.

Fig. 1 displays the area of operation around the town of Röbel. The organization and operation of the demand-oriented shuttles was conducted by the KOMOB institute. These shuttles were used to connect multiple small villages to the line-based public transport system of the local transportation company (MVVG). At the beginning of the project, route planning for

E. Neidhardt is with the German Aerospace Center Institute of Transportation Systems, Berlin, 12489 BA Germany (phone: +49 30 67055-9666; e-mail: eric.neidhardt@ dr.de).

these fleets was done manually by the volunteer drivers. Later they were connected to the HubChain platform which would then take care of the disposition [7]. Extensive real live operation was not possible due to the SARS-CoV-2 pandemic.

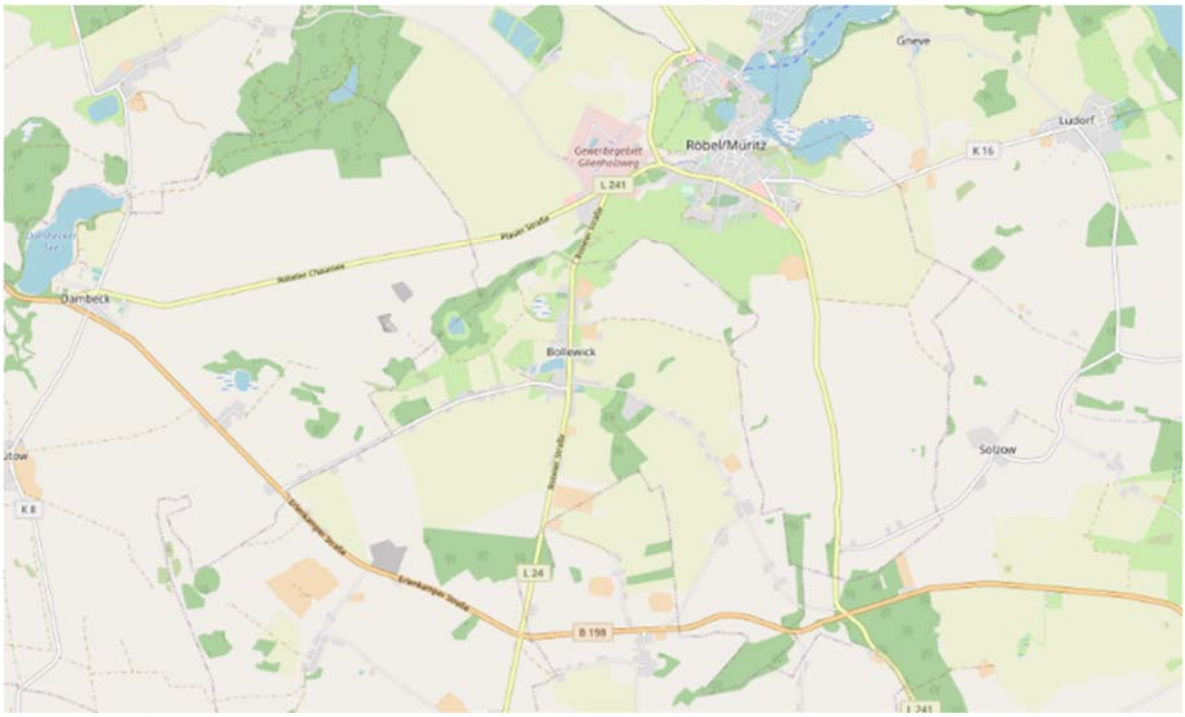

Fig. 1 Elde source area (Basemap and data by OpenStreetMap and OpenStreetMap Foundation)

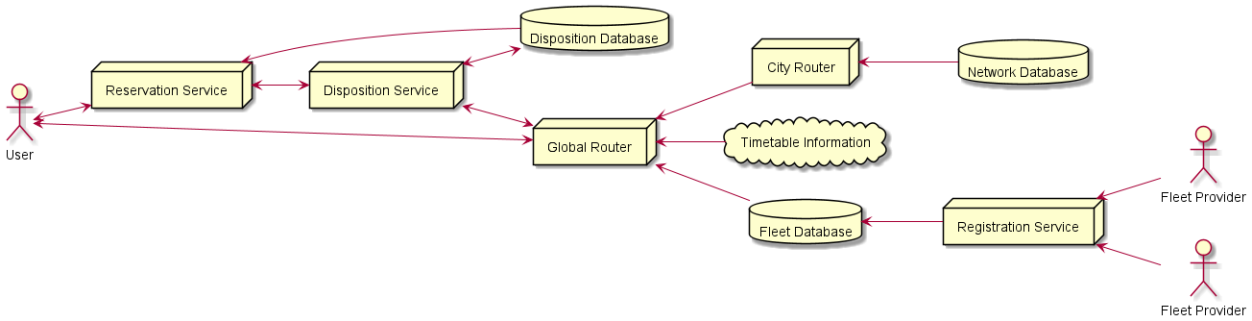

Fig. 2 HubChain system overview

IV. Implementation

This section will briefly describe the technical infrastructure and implementation of the backend services for the HubChain platform. Figs. 2-4 in this section are also used in [13].

The implementation of the HubChain platform is based on the DLR Traffic Data Platform and the KeepMoving Framework. KeepMoving is a service framework to collect process and utilize traffic data. Among other things, it provides services for intermodal routing and net adaption. Traffic data can be obtained from different sources, for example loop detector data or floating car data. It can be incorporated into the routing services. KeepMoving is able to work on OpenStreetMap data [6].

The implementation of the platform was conducted as a service-oriented architecture, with the goal to be as flexible as possible. This makes it easier to adjust the implementation to changing requirements. Furthermore, it would allow us to replace or repurpose individual components for upcoming projects. All services were implemented using the Java programming language. The services were connected using an Apache ActiveMQ messaging broker [8]. ActiveMQ implements the Java Message Service (JMS) API and JMS ObjectMessages are being used by the different service. External access was provided via SOAP and RESTful webservices.

Map data are usually obtained from OpenStreetMap. The SUMO tool osmGet.py is used to download OpenStreetMap data in the OSM XML format. Another SUMO tool (netconvert tool) is used to convert the downloaded OSM data into our internal format [9]. These data are stored in our network database. Both PostGIS and Oracle Spatial and Graph are supported. The imported data can later be manually refined by

incorporating other sources of information, like data from local authorities or the municipality. This is especially important if the imported map data lack completeness.

Timetable information from the local mobility provider is considered for the intermodal route calculation. Data are imported in the TRIAS format. The TRIAS (Travellors Realtime Information Advisory Standard) format was developed by the Association of German Transport Companies (VDV) and is supported by many German mobility providers [10].

Fig. 2 displays a brief overview of the system. As described before, the system is implemented as a set of multiple services. These services are communicating via a common message broker, which is omitted in this figure. Providers of demand-oriented mobility offers can register themselves at the system by using the registration service. They can input the constraints of their service, like area of operation, operation time, vehicle capacity and so on. Data regarding demand-oriented fleets is stored in the fleet db.

The city router is the primary routing engine for cars, busses, and pedestrians. Internally, there are different instances of the router, based on the mode of transportation. The city router is invoked by the global router, which acts as a broker for routing requests. The global router is responsible for calculating intermodal routes. It considers timetable information from the local mobility provider and all registered demand-oriented services. An especially important component is the disposition service. The disposition service is responsible for creating the traffic plan for the demand-oriented shuttles, based on user requests. The disposition service talks to the global router to obtain routes and travel time for the shuttles. The reservation service is mainly responsible for managing bookings and reservations.

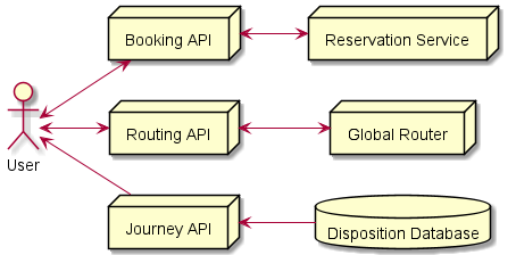

Fig. 3 HubChain User APIs

Fig. 3 provides an overview of the relevant APIs for the user. The routing and booking API are also displayed in Fig. 1. The booking API is responsible for managing booking. The use cases are reserve, cancellation and booking of trips. The routing API is responsible for providing routes to the user, based on his request. These routes may contain on-demand services, but do not have to. If a request can be satisfied by using conventional public transport, this result is preferred. The journey API is used to provide information on planned routes. It is mainly used to visualize the journey for the user. In the actual implementation, all APIs are encapsulated via a common interface, the ServiceHub. This interface is omitted in Fig. 3 for better presentation.

Fig. 4 displays the workflow of the primary use cases of the on-demand shuttle from the user's perspective. These use cases are planning and reservation of an intermodal trip, booking a reserved trip and cancelation. Before the user can ride the shuttle, the trip must be reserved and booked. We settled for a two-step booking process, consisting of a planning or reservation phase and the actual booking. After the users requests a route that would involve and on-demand component, the disposition service performs the necessary calculations for the trip. It tries to merge the request with already planned trips. If this is not possible, it creates a new trip. Trip information is returned to the user. In real world, multiple results would be returned to the user to provide him with options. Furthermore, it could be possible that the user request can be satisfied by only using line-based, regular services. In that case, no actual reservation is required and the user is informed that he can use regular public transportation. If the calculated route contains demand-oriented routes, a reservation is required. At this stage, the calculated route is already added to the disposition database but marked as preliminary. A generated booking number is returned, together with a request id. These IDs are used in further requests to confirm or cancel the trip. The user has 5 minutes to confirm his trip, otherwise the preliminary route is deleted. Because the preliminary route is already considered in further route calculations, it cannot be invalidated by another request. This has the downside, that with many requests in short time span and limited capabilities, the disposition may be cluttered with preliminary routes fast. However, because we concentrate on rural areas, this was not a problem during our tests.

Booking and confirmation of a preliminary route is straight forward. The reservation service verifies the provided information (booking number and request id) and updates a flag of the route accordingly. The process for cancelation is similar and works the same for preliminary routes and booked routes. However, cancelation of a request make is necessary to recalculate affected routes, because certain planned stops may no longer be necessary. The reservation service sends a notification to the disposition service and the disposition service performs the recalculation of affected routes.

V. Disposition Algorithm

This section describes the algorithm that was used by the disposition service. As explained before, the disposition is responsible for route planning. The service creates journeys for demand-oriented mobility offers from user requests. This section will not go into the details of the implementation and the algorithm. Details regarding the implementation are described in [11].

Dispositioning a demand-oriented shuttle based on user requests is a Dial-a-Ride problem. Solving a Dial-a-Ride can be done with multiple goals. Often it is desired to serve a maximum of requests with a given fleet size. The disposition service of the HubChain platform was implemented using an ant colony algorithm. In this approach agents are working to together to solve the traveling salesman problem [12]. The ant

colony algorithm mimics the behavior of real ants, hence the name.

A regular ant colony algorithm does not differentiate between different types of nodes. All nodes are handled equally. However, this is insufficient for our Dial-a-Ride problem. In our problem we have origin and destination nodes, based on the requested origin and destination. The algorithm must ensure that the destination nodes are not visited before the matching origin nodes. Therefore, adjustments to the algorithm were necessary. Details are described in [11].

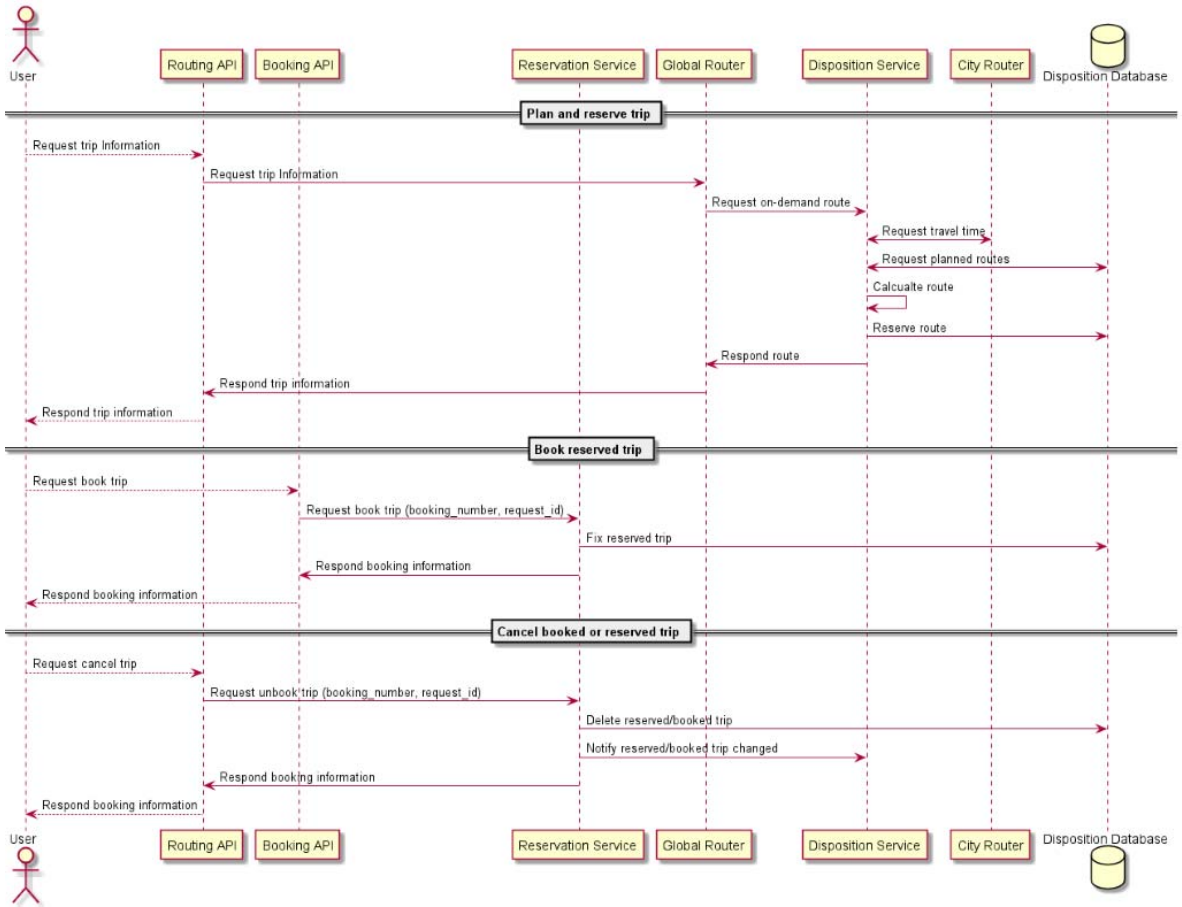

Fig. 4 Reserve, book, cancel on-demand shuttle trip

VI. Constraints and Request Clustering

The disposition service is responsible to consider many different constraints while planning the bus routes. Constraints have a strong impact on the clustering behavior, the capacity and the sustainability of the system. However, they also influence the attractiveness of the system from a user's perspective. Planning fewer but longer routes can be accomplished by clustering more request together. Planning fewer routes is more sustainable. However, this behavior also reduces the attractiveness of the system from a user perspective. It is important to find a suitable tradeoff for the route planning behavior.

Constraints are mainly divided into user-centric constraints, technical constraints and organizational constraints.

Technical constraints mainly focus on the used vehicle. Among other things, they include the maximum operation time and the passenger capacity. The capacity to transport heavy or unwieldy luggage is also a property of the vehicle.

Organizational constraints include the area and time of the operation, but also the presence of vehicle depots. Both organizational and technical constraints are fixed by the on-demand fleet provider and cannot be changed by the HubChain platform.

User centric constraints are important for the clustering behavior of the disposition service. The disposition service tries to cluster similar user requests together to reduce the number of stops and trips. The clustering part of the disposition service is the main difference to taxi service and important to increase the sustainability of the system.

While clustering requests, the actual planned pickup location as well as the departure or arrival time may differ from the original user request. However, there are limits to these changes, because otherwise the system would be too unattractive for the user. Two factors in user behavior are particularly important in order to assess the possible limits. The first factor is the maximum distance a user is willing to walk to the pickup or from the drop off location. The second factor is the maximum deviation from his desired departure or arrival time he is willing to accept. User centric constraints are set by

the provider of the HubChain platform. Determining optimal values is challenging and requires involving the customers and users. User centric constraints are the main way of adjusting the service. They should be monitored and adjusted to changing needs. In HubChain we settled for the properties in Table I.

TABLE I
USER CENTRIC CONSTRAINTS

Property	Value
How far in advance is a booking required	60 minutes
Wait time at bus stops	2 minutes
Maximum time shift for actual departure time	30 minutes
Maximum time shift for actual arrival time	30 minutes
How much longer a passenger's journey can take compared to taking the direct route (direct route factor)	5

VII. FUTURE WORK

Overall, implementation of the HubChain platform in the selected rural test area went successfully. The living lab provided some valuable input for further development. However, there were also problems. Unfortunately, due to the impact of the SARS-CoV-2 pandemic, real live testing could not be performed as extensively as desired. More testing and application in further projects are required to assess the software. This is going to be our focus in upcoming projects.

Currently, the disposition service does not incorporate the current, actual position of the vehicles. It is assumed that the dispositioned vehicles drive as planned. Because the area of operation is a very rural area, this is usually not a problem. Traffic jams are seldom and the route planning and booking is done with sufficient time in advance. Nevertheless, route planning must include some buffers to account for unexcepted delays. Should an exception occur, manual intervention is required.

The fleet registration service to add new on-demand fleets to the system is also not yet implemented. Adding new mobility offers and modifying the constraints of existing fleets currently require manual changes on the database, which is cumbersome and error prone.

ACKNOWLEDGMENT

This paper is supported by Federal Ministry for Economic Affairs and Energy, Germany, as a part of project HubChain, which focuses on the on-demand public transportation with electrical vehicles.

REFERENCES

[1] U. Schneidewind and K. Borschert. „Wissenschaft für Nachhaltigkeit - Herausforderung und Chance für das baden-württembergische Wissenschaftssystem“, Stuttgart, 2013

[2] J. Berg and J. Ihlström, “The Importance of Public Transport for Mobility and Everyday Activities among Rural Residents”, (S. Sciences, Ed.) Open Access Journal, vol. 8(2), 2019, pp. 1-13.

[3] P. Bakker, “Large scale Demand Responsive Transit Systems - A Local Suburban Transport Solution for the Next Millennium?” in Proceedings of European Transport Conference, Stream: Public Transport Planning and Management, 1999

[4] M. P. Enoch, S. Potter, G. P. Parkhurst, M. Smith, „Why do Demand Responsive Transport Systems fail?” in 85th annual meeting of the TRB, 2006

[5] E. Neidhardt and A. Sauerländer-Biebl, “Provisioning a demand-orientated bus system for public transportation”, 15th World Conference on Transport Research. Mumbai, 2018

[6] L. C. T. Tcheumadjeu, S. Ruppe, “Traffic Data Platform based on the Service Oriented Architecture (SOA)”, 12th World Conference on Transport Research (WCTR), 2010, Lissabon.

[7] Stadtwerke Osnabrück AG., „Hub Chain project“, https://www.hubchain.de/aktuelles/, 2021

[8] “Apache ActiveMQ”, http://activemq.apache.org, 2019

[9] “SUMO, Simulation of Urban Mobility”, https://sumo.dlr.de/index.html, 2019

[10] „Internet Protokoll basierte Kommunikationsdienste im ÖV - Das Projekt IP-KOM-ÖV“, https://www.vdv.de/ip-kom-oev.aspx, 2020

[11] Q. Tang and E. Neidhardt, “Simulation-based method of a dynamicalon-demand transportationproblem” in Proceedings of WTC 2019, 3rd World Transport Convention, Beijing, 2019

[12] Marco Dorigo and L. M. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem” in IEEE Transactions on Evolutionary Computation vol.1, 1997, pp. 53 - 66

[13] E. Neidhardt, “Eine Mobilitätsplattform für den bedarfsorientierten Verkehr”, unpublished