SUMO User Conference 2024
Conference paper
https://doi.org/10.52825/scp.v5i.1126

Published: 17 Jul. 2024

Using SUMO for Test Automation and Demonstration of Digitalized Railway Concepts

Integrating SUMO with the ”Train Dispatcher in the Cloud”

Arne Boockmeyer [1], Dirk Friedenberger[1], and Lukas Pirl[1]

[1]Hasso Plattner Institute, University of Potsdam, Potsdam, Germany

*Correspondence: Arne Boockmeyer, arne.boockmeyer@hpi.de

Abstract: The *Train Dispatcher in the Cloud* (ZLiC) is a cloud-based approach to digitalize the German *Zugleitbetrieb* (comparable to American track warrant control). The ZLiC aims to replace the train dispatcher with speech to text, natural-language understanding, a digital occupancy sheet, a prototype interlocking logic, and text to speech. For train conductors, who operate on the trains, the voice-based communication with the train dispatcher remains unchanged. Because the external interfaces of the ZLiC are either voice-based or graphic, understanding and testing the internal components from an integration level is a challenge. To address these challenges, we first injected the Simulation of Urban Mobility (SUMO) as a simulation environment. Since the ZLiC has been developed model-based, the integration requires minimal modifications. Afterward, we fetch the operation commands (e.g., registering trains, locating trains, drive requests for trains) between the components and send them to a SUMO instance for analyzing and visualizing the train operations in the railway network. Lastly, we insert additional planned trains to add simulated traffic. The reproducible operations enable test automation of the ZLiC while reusing the sophisticated models in SUMO. This prototype shows that SUMO can support the development of digitalized railway operating procedures.

Keywords: Train Dispatcher in the Cloud, SUMO, Visualization, Test Automation

1 Introduction

The more and more visible effects of climate change are moving political decisions towards more climate-friendly technologies and procedures. This shift also affects the energy industry, reducing fossil energy sources and expanding renewable energy sources. In turn, this affects, e.g., the Lusatia region in southern Brandenburg, Germany. This region has large deposits of coal and several coal-based energy plants. To reduce the national carbon footprint these coal-based energy plants should be shut down until 2038. However, this change also entails challenges for the Lusatia region, since the energy plants and all related industries are the main employers in that region.

To face the future problems of this region, new concepts for industries, for the reuse of the existing facilities, and for transportation are needed. One part of this is the future

usage of the existing railway network for public transport, which is currently still partly used for coal transportation.

Developing transportation concepts for transforming mining areas is the goal of the *FlexiDug* project. It is funded by the Federal Ministry for Digital and Transport (*Bundesministerium für Digitales und Verkehr*). Together with partners from academia and industry, we are developing a concept to enable railway-based public transport with low migration and operating costs. One part of this project is the development of the Train Dispatcher in the Cloud (German *Zugleiter in der Cloud*, abbreviated *ZLiC*) [1], a digitalized version of the German *Zugleitbetrieb* [2]. In the *Zugleitbetrieb*, which is comparable to the North-American *Track Warrant Control*, a train conductor on the train asks the central human train dispatcher for driving permission via voice communication. The train dispatcher verifies the request by checking the paper-based track occupancy sheet (German *Belegblatt*) and responds according to the occupancy status of the track. This prevents collisions on the track since there can not be two trains at the same time on the track. The *Zugleitbetrieb* is a technically simple operating procedure and is used on less frequently used railway networks.

The core idea of the ZLiC is the replacement of the human train dispatcher with a system based on cloud technologies. It communicates with staff on trains, manages track occupancy, and can control the railway infrastructure. This approach addresses the problem of the increasing shortage of skilled personnel. On the trains, the conductors use the same voice-based interface to communicate with the digitalized train dispatcher. The ZLiC also contains an interlocking logic to manage the infrastructure and a graphical occupancy sheet for tracing, documentation, and system introspection.

During the development, we were facing two issues, where the integration of a simulation tool, such as the Simulation of Urban Mobility (SUMO) [3], could support the development process of such a novel implementation of a railway operating procedure:

Visualization Performing dry runs, meaning executions without real trains and real infrastructures, lacks the possibility of visual output. Besides the communication and the graphical occupancy sheet, there are only textual command line outputs. For presentation purposes, visually moving trains would be more realistic, illustrative, and easier to understand.

Test Automation Having the voice-based interface for the train conductor makes it intricate to perform a test run since the input needs to be created manually. Besides this, for every train in the execution, a person needs to act as the train conductor. Large experiments with several interacting trains using the same interface are difficult to realize. It is possible though sending the right commands to the ZLiC, but a traffic simulation environment would increase the realism of the behaviour of the trains. This leads to advanced and more realistic test automation.

The remainder of this paper is structured as follows: The second section briefly explains the ZLiC and its architecture. The third section describes how we integrated SUMO to counteract the mentioned challenges in the area of demonstration and test automation. This also contains implementation details and limitations. The fourth chapter presents the current results of this integration. The fifth chapter presents ideas for future work with SUMO as part of the ZLiC and the last chapter concludes this paper.

2 The Train Dispatcher in the Cloud

As Pirl et al. describe in detail in [1], the system as shown in Figure 1 consists of functional components that are responsible for interlocking logic, the voice protocol,

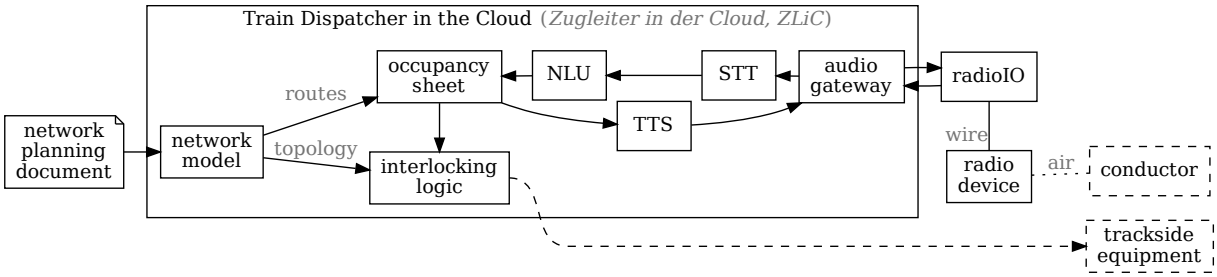

Figure 1. *The architecture of the ZLiC (solid box). It contains a distributed system consisting of several independent components. On the left side are the logic components, namely the digital occupancy sheet, and the interlocking logic. Both are initialized with a railway network model. On the right side are the voice processing modules, converting the voice-based input and output from and to commands. Related components (dashed) include radio devices and infrastructure components of the railway network. [1]*

speech input, and speech output. The audio gateway is used to receive voice commands from the train conductor, which are matched with predefined commands for the occupancy sheet using speech to text (STT) and natural-language understanding (NLU). The voice protocol with the train conductor is controlled via a state machine in the occupancy sheet component. To set the itineraries (meaning any connection between to stations) requested by the train conductors, the occupancy sheet translates them to a set of routes (meaning a connection between to following signals). This mapping is predefined. These routes are then sent as commands to the interlocking logic component, which can then control trackside equipment, such as signals and points[1]. The responses are sent to the conductor using text to speech (TTS) via the audio gateway. Communication with the conductor takes place via commercial off-the-shelf (COTS) radio devices. The *radioIO* component handles the transformation between the API of the ZLiC and voice radio hardware. Some components are supplied with a railway network model (containing the topology, details about the control-command and signaling infrastructure, and the predefined planned routes), which has been generated from a railway network planning document (for example in the PlanPro format). The occupancy sheet component uses this information to determine which itineraries and routes can be set, while the interlocking logic component uses the required topology to set the commanded route.

As also described in [1], an ontology-based approach has been chosen for the design and development. Ontologies have been developed for the individual concepts, such as a component model, domain-specific data models, and a generic artifact model. Pre-existing ontologies, such as for state machines [4], have been reused. The models are defined based on the ontologies and can be used to automatically generate the documentation, the infrastructure definition files for *Docker* or object files for *Kubernetes*, and parts of the implementation, such as the state machine for the occupancy sheet.

3 Integration of SUMO

The ontology-based approach enables the definition of components and data flows, as well as the messages that are exchanged between the components. Furthermore, the proxy design pattern [5] is used to decouple the implementations of the components

[1] The control of signals and points from a central interlocking logic goes beyond the scope of the *Zugleitbetrieb* but is useful for the SUMO simulation and possible future extension of the system.

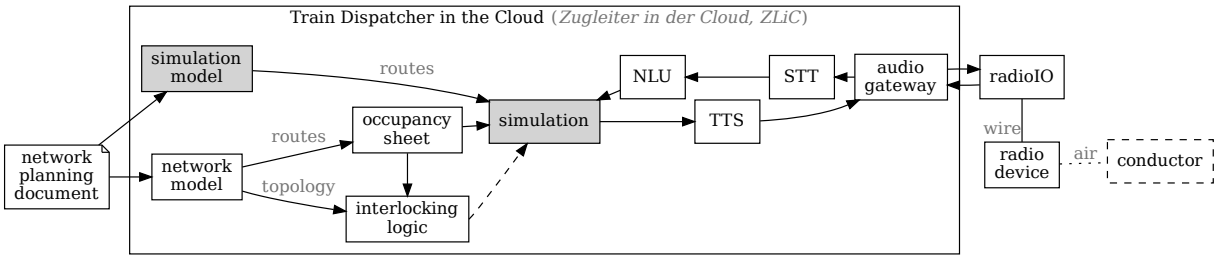

Figure 2. *This figure shows the architecture of the ZLiC with the injected simulation component. The simulation component is injected between the* digital occupancy sheet *on the logic side and the* NLU *respectively the* TTS *component on the audio side. The simulation gets the data model from the railway network.*

from the infrastructure. In this case, the proxy represents the interface for receiving and sending messages between the components. The chosen approach also eases exchange and adding of components. This has been utilized when integrating the simulation into the ZLiC system. Also the data flows have been redirected accordingly via the simulation component. As a result, the simulation can be switched on and off via a flag. There is no need to reprogram components and no need to include simulation-specific logic in the productive code. This can be referred to as an injection of the simulation component, as the other components are not affected by this injection due to the decoupling.

The customized component view is shown in Figure 2. On the one hand, various messages, such as the requests from the conductor or the occupancy sheet, are routed through the simulation component to update the state of the simulation itself. Commands from the simulation, e.g., from automatically operated trains in the simulation, are sent to the corresponding components, such as the occupancy sheet and the voice output. The simulation component also requires the list of itineraries and routes which is derived from the railway network planning document in the same way as the railway network model. The skeleton of the simulation components is generated based on the definitions of components, data flows, and messages. This consists of the proxy interface for sending and receiving messages as well as the *Dockerfile*. The implementation of the components can therefore be customized. In addition to the actual implementation of the interfaces, additional software can also be installed via the *Dockerfile*. In the case of the simulation component, this is *SUMO* and the control interface *TraCI*. This shows how easy it has been to integrate *SUMO* as a simulation component into the existing ZLiC system. With around 600 lines of code, the implementation effort has been low as well.

3.1 Phase 1: Voice-based Interface and Demonstration

After injecting the simulation component between the *NLU* component and the digital occupancy sheet, the first phase is to use the voice-based communication between the conductor on the train and the train dispatcher to control trains inside SUMO. Therefore we need to follow the process each train passes through in the ZLiC as shown in Figure 3. This means, we need to fetch and simulate the three different voice command chains:

Registration The process starts with the registration of a train. In reality, this means that a train is entering the railway network, that is managed by the ZLiC. Therefore, the train conductor asks, if the train dispatcher accepts the train. Since this is

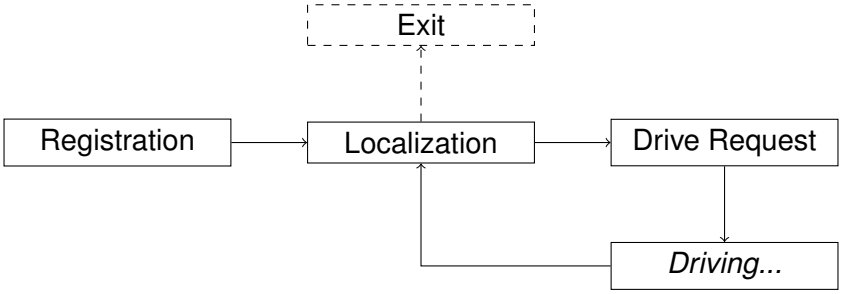

Figure 3. *The state machine for trains in the ZLiC system. The first step is the registration of the train when it arrives in the railway network that is managed by the ZLiC. Afterward, the train conductor localizes the train in the railway network. Now the conductor asks for permission to drive to another location in the railway network. When the train reaches its destination, the conductor localizes the train again. Afterward, the conductor can either request another permission or exit the railway network (leaving the network is not yet implemented at the time of writing).*

not connected with any location of the train, we can use this information only to prepare our simulation component for the train soon to be created.

Localization The localization of the train is the next step in the process. Now it is possible to create the train inside of SUMO. The localization message only has the name of the railway station, which needs to be translated to a SUMO route. With the route, the train can be created via TraCI. Since it has no driving permission yet, the speed of the train is set to zero and the following signal is red[2].

Driving Request Now, the train conductor can request a driving permit. This request contains the target destination. Therefore, the simulation component takes the mapping from itineraries to the routes as input. Each route has a corresponding route in SUMO. It also sets the first route of this list for the train. When this train reaches the end of a (SUMO) route, this simulation component sets the next (SUMO) route, until the train reaches the destination of the itinerary. The interlocking logic of the ZLiC manages the signals of each route on that itinerary via a second TraCI connection.

After reaching a destination, the train conductor needs to localize the train again. Since the train already exists, this has no effect on the simulation. The voice commands for each part are shown in Figure 4.

When the train leaves the ZLiC-managed railway network, it needs to be removed from the simulation. This however is not implemented in the ZLiC to the date of this paper.

This first phase leads to a more detailed output of the ZLiC since it only had the digital occupancy sheet, the voice interface, and command line outputs before. This leads to a better understanding and demonstrability of the ZLiC.

3.2 Phase 2: Automatic Train Operations and Test Automation

The second phase builds upon the first phase. It adds the capability for placing automatically operating trains. Each automatically operating train follows the same process as manually operated trains shown in Figure 3. Automatically operating trains also use the same interface to the digital occupancy sheet as manually operated trains (see phase 1).

[2] In our model, each route starts with a signal after the first edge of the route.

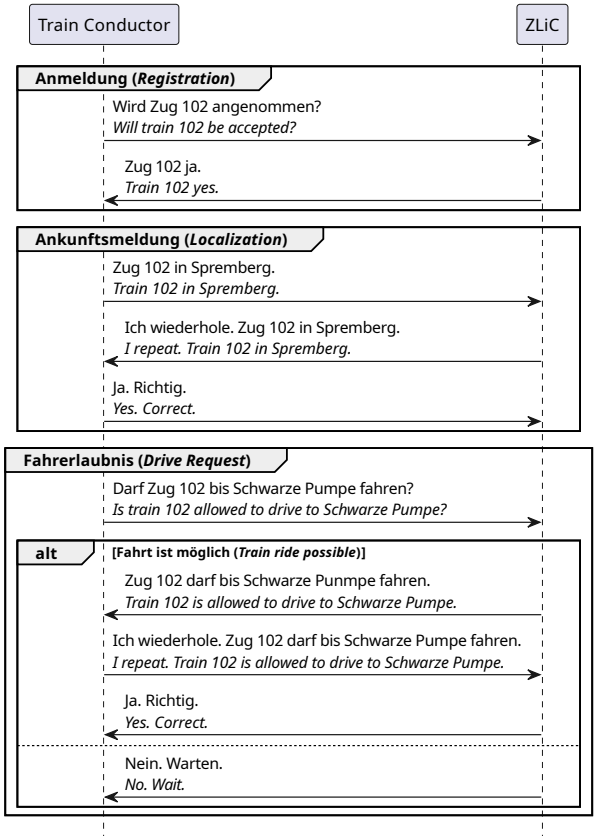

Figure 4. *The sequence diagram above shows the communication protocol of the three parts registration, localization, and drive request between the train conductor and the ZLiC.*

Each automatic train will be initialized with a timestamp after the train is created, the line (a connection between a start location and an end location, covering a single or multiple itineraries, depending if the train should stop at stations in between), and intermediate stops. When running through the process, each automatic train will also create the voice requests and answers to the voice output. For conductors which possibly manage another trains manually, the automatically operated trains appear as other train conductors on other trains in the same railway network.

This has the advantage, that tests with the ZLiC can cover multiple trains without requiring a human train conductor each. Another advantage of this is the possibility of test automation of the digital occupancy sheet and the interlocking logic since the automatic trains are using the same interface (meaning both components will not see any difference).

3.3 Implementation Details

This section contains details about the implementation of the simulation component.

SUMO Network Creation

Before running any of the simulations, the SUMO simulation configuration needs to be created. Therefore, we are using the *yaramo* model [6][3]. As depicted in Figure 5, the *yaramo* model can be created based on OpenRailwayMap[4] or digital network planning

[3]Publicly accessible on GitHub: https://github.com/simulate-digital-rail/yaramo
[4]https://www.openrailwaymap.org/

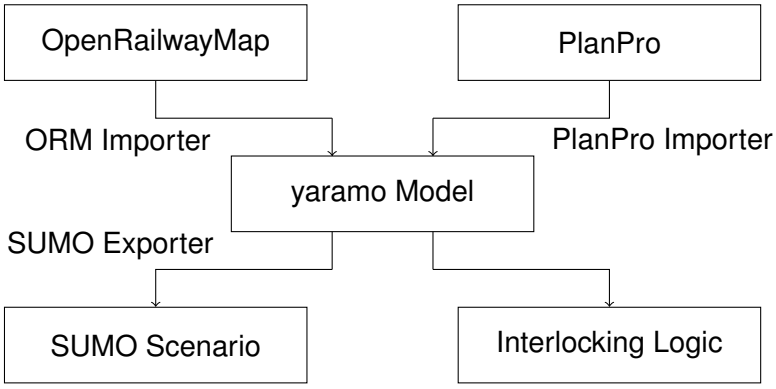

Figure 5. *The central* yaramo *model is the data basis for the simulation and the interlocking logic. Using the same model here has the advantage that both consuming tools are using the same model. The yaramo model can be created based on OpenRailwayMap or railway network planning documents in the PlanPro format. The importer and exporter are performing the data transformation.*

documents for railway networks in the PlanPro format[5]. The ZLiC needs a *yaramo* model as input to the interlocking logic. Since the *yaramo* model can be exported to a SUMO configuration, the same model as for the ZLiC can be used for the SUMO simulation. This has the advantage, that both simulation and interlocking logic use the same data model.

As explained in [6] in detail, the SUMO exporter first converts the *yaramo* model to the *SUMO PlainXML* format. Using *netconvert* the SUMO network will be created. At the end, a route file containing the SUMO routes will be attached to the configuration.

This SUMO simulation configuration is the input for the SUMO simulation. The configuration will be created when creating the image of the *Docker* container containing the simulation component.

Docker Image and TraCI Connection

We created a *Docker* image for the simulation component. This image is based upon our DDS[6] image for the data distribution and contains both, a SUMO installation and the TraCI library.

The SUMO instance needs two TraCI connections. The first connection is inside the container itself since the simulation component is connected via TraCI to SUMO. This TraCI connection is used to create and manage the trains inside SUMO. As shown in Figure 2, the interlocking logic needs a second connection to SUMO. This connection is for the management of the infrastructure. Therefore, the interlocking logic has the concepts of *infrastructure providers* (IPs), meaning these IPs are the connection to either real or simulated infrastructure components such as signals or points. All changes to the infrastructure (like switching a signal or moving a point) are communicated to all IPs. In the case of the SUMO infrastructure provider, TraCI forwards the signal changes to the SUMO simulation. As mentioned below in the limitations, this is limited to signals only.

[5] https://www.dbinfrago.com/web/schienennetz/dienstleistende/planpro
[6] Data Distribution Service, we are using Eclipse Cyclone DDS

3.4 Limitations

The current realization of the simulation component is mainly limited to two aspects:

Single Line Support So far, the ZLiC logic only supports a single straight line through the railway network. Having multiple lines would be closer to reality, but also increases the complexity of the digital occupancy sheet. Technically even interfering lines would be possible. Besides this, the model for the single line needs to be created by hand at the moment, meaning the test designer needs to collect the signal sequence (and with this the (SUMO) routes) manually.

Missing Support for Points Until now, SUMO has no support for points in railway networks. We are assuming, that the points are always moved to the correct position when driving a train through the network. However the interlocking logic is already able to calculate the correct point positions, but the SUMO infrastructure provider ignores them.

4 Results

We ran two experiments to demonstrate that our implementation serves the purposes of visualization and test automation. The experiments also show the ability to perform the two phases described in subsection 3.1 and subsection 3.2. This section gives an overview of these two experiments. Since both include voice commands, we recorded two videos[7,8] showing the experiments. As the ZLiC implements a German railway operating procedure, the voice commands are in German[9].

4.1 Experiment 1: Voice-based Interface and Demonstration

The first video[7] shows phase 1 with the voice-based interaction of a train conductor with the ZLiC and their effects on the SUMO simulation. In the first step, the train conductor registers his train with the number 102 to the ZLiC. As mentioned before, the conductor therefore asks for the registration and the ZLiC answers with a confirmation. Afterward, the conductor locates his train in the railway station *Spremberg*. According to the protocol, the conductor informs the ZLiC about his location, the ZLiC confirms the location by repeating the request and the train conductor confirms the repetition. Completing the second step, the simulation component creates the train at the location in SUMO. This is also shown in Figure 6.

After registering and locating the train, the train conductor requests driving permission to the destination *Schwarze Pumpe*. Therefore, the protocol, that is also shown in Figure 4, requires the request itself. Afterwards, the ZLiC checks if the requests can be granted. If yes, the ZLiC sends a positive response and the train conductor confirms this response again with a repetition. Finally, the ZLiC confirms the response of the train conductor. The interlocking logic manages the required signals via the second TraCI connection and the digital occupancy sheet shows the reservation. The train can depart now, meaning that SUMO sets a maximum speed and accelerates to that speed. This is also shown in Figure 7. When reaching the destination, the train conductor sends a new localization to the ZLiC. Only when the train has arrived completely the track can be released. Therefore the check of train completeness is an obligation

[7] https://osm.hpi.de/flexidug/sumo-user-conference-2024/video-1
[8] https://osm.hpi.de/flexidug/sumo-user-conference-2024/video-2
[9] A translation can be found in Figure 4.

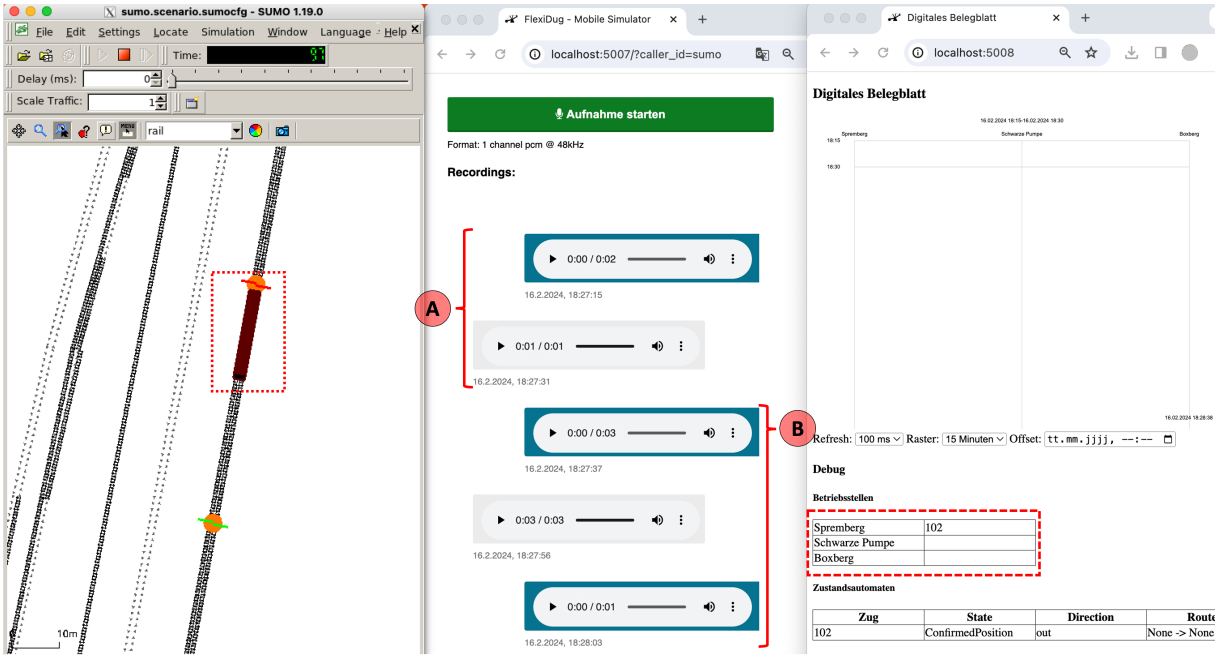

Figure 6. *This screenshot shows a train placed in SUMO on the left side (red dotted box). This train has been created and localized by the ZLiC. The communication between the ZLiC (grey voice outputs) and the user as the train conductor (blue voice outputs) of train* 102 *is shown in the middle. The voice messages marked with* ***A*** *are the registration commands and the voice messages marked with* ***B*** *are the localization commands. The digital occupancy sheet on the right does not show any train movements, since the train hasn't requested any so far. But the table below (red dashed lines) shows the train number in the list of the station.*

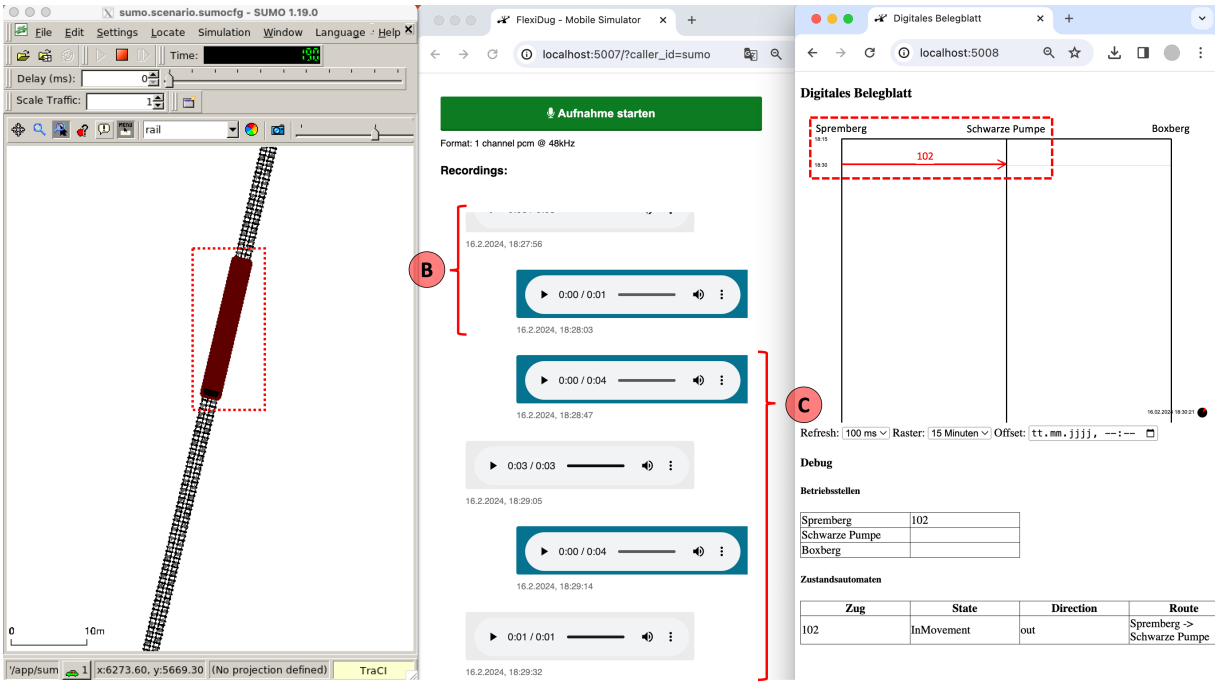

Figure 7. *The train conductor now requested permission to drive to the second station* Schwarze Pumpe*. The train starts moving in SUMO and the digital occupancy sheet (edited for readability in print) shows the reserved tracks (red dashed lines). The necessary voice commands for the drive request are marked with* ***C****.*

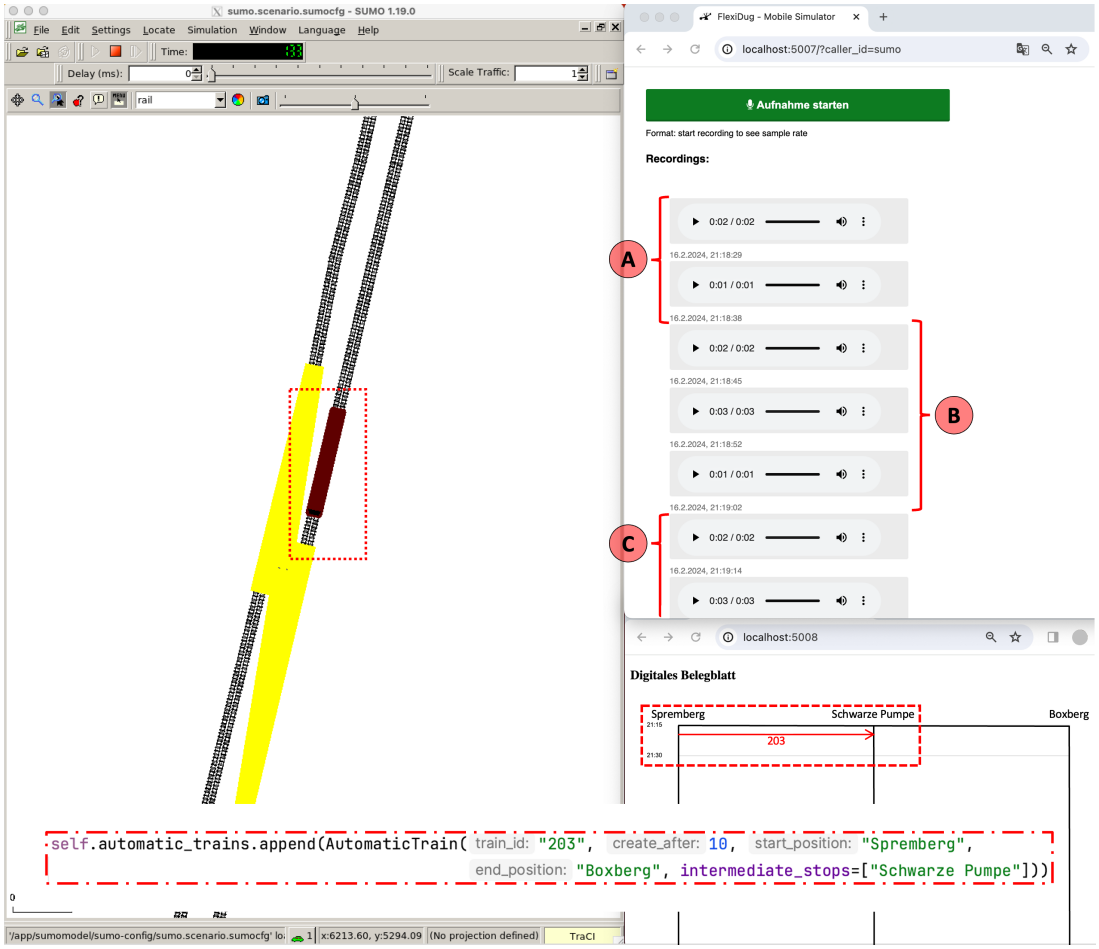

Figure 8. *This screenshot shows a part of the simulation with an automatically operated train. The communication on the right side also follows the three steps marked with **A** (registration), **B** (localization), and **C** (drive request). The digital occupancy sheet shows the reservation of the track (dashed lines) and the SUMO simulation on the left shows the moving train (dotted lines). The code in the bottom (red dotted/dashed lines) shows the configuration of the automatic train.*

for the train conductor to submit the new localization. The ZLiC releases the tracks in between and shows the operation in the digital occupancy sheet.

This experiment shows that the SUMO simulation component is able to fetch the communication between the train conductor and the ZLiC. It extracts the necessary details and sends them via TraCI the simulation. Since the simulation is displayed with the SUMO GUI, this closes the lack of a missing visualization and enhances the understanding of the system.

4.2 Experiment 2: Automatic Train Operations and Test Automation

The second video[8] shows the automatic train operations. Therefore, the test specification defines a leaving train with the number 203 at the timestamp 10. This train should drive from *Spremberg* to *Boxberg* with stopping in *Schwarze Pumpe*. As shown in the video and in Figure 8 the tester hears (and sees) the full voice interaction between the train conductor of the simulated train and the ZLiC.

The second experiment demonstrates that, together with the simulation component and SUMO, it is possible to run automated tests for the critical parts of the railway operating procedure implemented by the ZLiC. These are in particular the voice proto-

col, the digital occupancy sheet and the interlocking logic. The automatically operated trains depart as defined and lead to realistic inputs for the ZLiC logic. Since SUMO simulates realistic train movements, the communication points happen in a realistic order and time intervals. In addition, scenarios that occur in reality, such as the handling of delayed trains, can also be tested and thus the procedure can be stressed. It would also be possible to use this simulation as part of continuous testing processes during the development when using SUMO without a GUI.

5 Next Steps and Future Work

Continuing our research on the ZLiC and the simulation component, we are planning to extend the simulation component in the following ways:

Test Suite Creation The implementation of automatically operated trains leads towards support for test automation. However, having a larger test suit could automatically test multiple instances of different simulations at the same time and evaluate the performance of the ZLiC implementation. This could be connected with fault injection techniques to test the system under test in unusual situations.

Include Voice Interface Currently, the simulation component is inserted between the natural-language understanding library and the digital occupancy sheet. From the perspective of test automation, this leaves out the voice interface as part of the tests. Moving the automatic train operations of the simulation component before the voice-based interface would also include this part of the system.

Multi-line Support To realize more complex and more realistic scenarios, a multi-line support for ZLiC and the simulation component is necessary. Therefore, the data model of the simulation component needs to be extended and the visualization of the digital occupancy sheet as well.

6 Conclusion

In this paper, we show that we have been able to use SUMO to support the development, demonstration, and testing of newly developed railway operating procedures. Therefore we have used the Train Dispatcher in the Cloud (ZLiC) as an example. We have injected a simulation component in the model of the ZLiC system, nearly without modifying the other components of the system. It shares the same data model as the other logic components and is able to fetch the operations in the ZLiC system. These operations are mirrored in a SUMO instance, moving trains and managing the infrastructure. This supports the visualization and understanding of the operations of the ZLiC system as well as enables test automation procedures for the implementation. Our experiments prove that we are able to realize this concept with the current ZLiC implementation and SUMO as a simulation environment. In the near future, we are planning to extend the implementation of the simulation component by additional features to come closer to reality and add more evaluation features.

Data availability statement

The contributions of this work include the concept, model definitions, and software. As described in the subsequent section, all artifacts are publicly available.

Our experiments are based on data from OpenStreetMap[10], used as input to the generation of the *yaramo* model. In the overall FlexiDug project, we also use digital railway planning documents as input, but these are confidential data of our industry partners.

Underlying and related material

The implementation of the ZLiC, including all components and models, is available on GitLab[11]. The main repository, which also contains the simulation component, is the *Flexidug Ontology-based Systems Engineering* repository[12]. Required third-party components, including libraries and models, are also available publicly.

The links to videos of our experiments can be found in the footnotes [7,8].

Author contributions

Arne Boockmeyer: Conceptualization, Software, Validation, Writing - Original Draft
Dirk Friedenberger: Conceptualization, Software, Validation, Writing - Original Draft
Lukas Pirl: Conceptualization, Software, Validation, Writing - Original Draft

Competing interests

The authors declare that they have no competing interests.

Funding

This research has been funded by Federal Ministry for Digital and Transport (*Bundesministerium für Digitales und Verkehr*, BMDV) as part of the FlexiDug research project (project number 19FS2024D).

Acknowledgements

We want to acknowledge our colleagues in the FlexiDug consortium for their support of our work. Of particular note, we would like to acknowledge Prof. Dr.-Ing. Birgit Milius and Heiko Herholz from the Technische Universität Berlin (TU Berlin) for their expertise in the area of railway operating procedures and our professor Prof. Dr. Andreas Polze for his support during the project and this work.

References

[1] L. Pirl, H. Herholz, D. Friedenberger, A. Boockmeyer, A. Polze, and B. Milius, "Train dispatcher in the cloud — digitalising track warrant control for safe train operations in structurally transforming areas," *Transport Research Arena 2024*, to appear, 2024.

[2] DB Netz AG, *Richtlinie 436 (Ril 436) – Zug- und Rangierfahrten im Zugleitbetrieb durchführen*, (Regulations on *Zugleitbetrieb* by Deutsche Bahn).

[10] https://www.openstreetmap.org/
[11] https://gitlab.com/hpi-potsdam/osm/flexidug
[12] https://gitlab.com/hpi-potsdam/osm/flexidug/flexidug-obse

[3] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: An overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*, ThinkMind, 2011.

[4] G. F. Schneider, P. Pauwels, and S. Steiger, "Ontology-based modeling of control logic in building automation systems," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3350–3360, Dec. 2017, ISSN: 1551-3203, 1941-0050. DOI: 10.1109/TII.2017.2743221. [Online]. Available: https://ieeexplore.ieee.org/document/8015195/ (visited on 06/08/2023).

[5] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture—A System of Patterns. New York: Wiley & Sons, 1996.*

[6] A. Boockmeyer, J. Baumann, B. Schenkel, *et al.*, "Processing digital railway planning documents for early-stage simulations of railway networks," *Transport Research Arena 2024*, to appear, 2024.