Applying Microservice Principles to Simulation Tools

Richard Pump Arne Koschel Volker Ahlers
Department of Computer Science
University of Applied Sciences and Arts
Hannover, Germany
Email: {richard.pump | arne.koschel | volker.ahlers}@hs-hannover.de

***Abstract*—The usage of microservices promises a lot of benefits concerning scalability and maintainability, rewriting large monoliths is however not always possible. Especially in scientific projects, pure microservice architectures are not feasible in every project. We propose the utilization of microservice principles for the construction of microsimulations for urban transport. We present a prototypical architecture for the connection of MATSim and AnyLogic, two widely used simulation tools in the context of urban transport simulation. The proposed system combines the two tools into a singular tool supporting civil engineers in decision making on innovative urban transport concepts.**

Keywords–Microservices; Simulation; Urban Logistics

I. INTRODUCTION

Urban transport and logistics are evolving fields of research. Modern concepts ranging from crowd-sourced delivery platforms like Foodora, to drone based delivery are changing the transport of goods. At the same time, services like Uber shift personal transport away from public transport solutions towards crowd-sourced ride sharing. A modern civil engineer not only has to keep an overview over the ever evolving modern concepts, but also know their advantages and disadvantages, as well as their impact on different factors like CO2-emissions and noise pollution.

To support civil engineers, the *University of Applied Sciences and Arts Hannover* is working in cooperation with the *Leibniz University Hannover*, the *City of Hannover* and the *Technical University Braunschweig* to create a decision and support tool for urban logistics [1]. The support tool allows the simulation of novel ideas for urban transport, comparing the impact of different ideas and visualizing the results adequately for easy comprehension. This helps decision makers to test their ideas against the real world without committing significant resources.

The tool combines two simulation frameworks with data storage and a unified front end, improving the usability of complex software. To combine the simulation frameworks, we decided to use a microservice-based approach. Using microservice principles allows us to develop parts of the system independently, which conforms to the project's organizational structures. With multiple partners building the system in different physical locations, independence becomes paramount. However, a conventional application of microservices, 'splitting the monolith' as it is often called, is not possible, since the main goal of the project is the development of new software using existing frameworks, instead of building a new system.

This paper will present our current work on the design of the tool. In Section II we will present a short overview of works relevant to this paper. Section III gives a short overview of the goals and requirements of the decision support tool and a rough system sketch is shown in Section IV. The Sections V and VI contain the main contents of the paper, showing microsimulkations and discussion advantages and disadvantages of the presented approach. The paper ends with a conclusion in Section VII.

II. RELATED WORK

The usage of microservices is a widely discussed topic in current research. Sam Newman provides a thorough overview over the microservice paradigms, as well as their advantages in [2]. While giving general recommendations the work does not follow our specific use-case of combining two simulation tools. As we lack the time and resources, splitting up existing applications isn't feasible within the boundaries of the project. Our work differs by combining microservices with production monoliths into a conglomerate design, splitting where sensible and feasible but purposefully keeping monoliths where not, as opposed to replacing the entire application design with microservices.

In [3], Nicola Dragoni et al. provide a comprehensive overview about microservices, as well as some definitions. The article however focuses on the conceptual ideas behind microservices, instead of a single application. A history of issues is presented concerning past issues, current issues and issues most likely appearing in the future. In the conclusion, the authors provide an opinion, viewing microservices as evolutionary rather than revolutionary, which our application of microservice principles shares. We intend to apply microservice paradigms only on parts of our architecture, providing a next step in designing simulation tools, rather than completely revolutionizing simulation architecture.

The term microsimulation used in this paper differs from the microsimulation models described by Merz [4] and others. Microsimulation models are based on microinformation and model only small parts of the system to be simulated, while not defining an architectural pattern used in the development for those models. While the simulation scenarios described in our paper might fall into the category of microsimulation models, the modeling process and simulation-theoretic backgrounds are out of scope.

Lastly, we use the simulation frameworks MATSim and Anylogic. Horni, Nagel and Axhausen present MATSim in [5]. They describe an open source, Java-based simulation tool for agent-based transport simulation. The MATSim architecture is modular to allow for flexible extension of simulation models but does not utilize microservices. AnyLogic is mainly presented in the works of Grigoryev [6] and Borshchev [7].

 ISBN: 978-1-61208-702-3

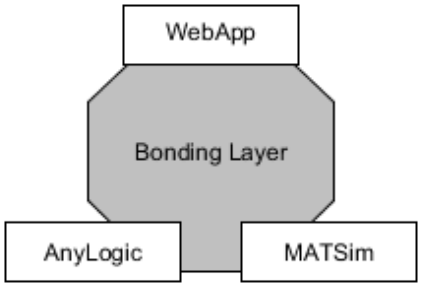

Figure 1. Rough system sketch of the decision support tool, showing the four major components.

AnyLogic is a proprietary, closed source, Java-based general purpose simulation tool. Again, microservices are not used in the simulation design.

Overall there are no works known to the authors that apply microservices to simulations.

III. REQUIREMENTS

The overall goal of the system is providing relevant information about novel logistic concepts to city planners. A city planner must be able to evaluate the impact of a logistic concept on his city, town or village.

The first step to make a decision is gathering information. The decision support and information system therefore has to inform the user about the different novel logistic concepts. Every concept has to be presented in an easily comprehensible way, e.g., by a short video. Not only concepts have to be presented, but also information about the city areas that can be used to evaluate the impact of the concept.

After informing the user about concept and city area, the software is to provide a configuration utility, allowing the user to modify parameters of the concept or the city area. The user should be able to save his configuration, as well as to load and edit old configurations, before starting the simulation.

The last step in using the information system is to evaluate the simulation results. The system has to present relevant information about the logistic concept stemming from the simulation in an easy to comprehend way. Also, comparisons with other simulated scenarios must be possible.

Not all of the information available within the tool is public, therefore an access control has to be implemented. The access control allows fine grained configuration of function and information access.

Furthermore the two simulation frameworks AnyLogic and MATSim have to be utilized to build the decision support tool. The project development team has experience in utilizing the two tools and switching to another framework is too costly.

IV. ROUGH SYSTEM SKETCH

Building upon the aforementioned requirements, the rough system sketch helps to recognize the architecture as a whole and is rather abstract. Figure 1 shows the four major components of the software.

The component ANYLOGIC is responsible for microscopic traffic simulation, modeling individual behavior using the simulation framework AnyLogic. For example, within the component ANYLOGIC, acceptance profiles for novel logistic solutions are simulated, giving a realistic representation of the individual person using logistics. Also, a precise modeling of the novel concept parts is achieved, e.g., the differently cooled compartments of e-grocery delivery vans are simulated. While giving a detailed view on the microscopic level of logistics, macroscopic events are not within scope of the ANYLOGIC component.

For the simulation of macroscopic events, the component MATSIM is used. Based on the equivalently named multi-agent transport simulation-framework, the component is responsible for the simulation of an entire city. For performance reasons, it models just the general movement of agents within the city, instead of concrete agent decisions. This allows the user to evaluate the impact of concepts on traffic flow and vice versa. Combining the MATSIM component with the ANYLOGIC component creates a holistic simulation for novel logistic concepts, modeling microscopic individual behavior and macroscopic events.

With results and analysis generated, results have to be presented within a graphical user interface, easily accessible from many locations. The component WEBAPP provides the graphical interface to access functions of the system, configure and run simulations, and compare the impacts of different novel logistic concepts. In contrast to the other two components, no external frameworks dictate architectural decisions, therefore the WEBAPP will be the only pure microservice-component.

The last component will be the bonding layer, which is a purely technical component, bridging the monolithic frameworks with the microservice-oriented WEBAPP. It is not a component in the traditional sense, since it does not group parts of the software which are responsible for a single piece of the domain logic. Also, the bonding layer contains elements that interact with the different simulation frameworks.

Generally, to evaluate a certain novel logistic concept, the user will select the concept within the WEBAPP, configure a simulation using one or both of the simulation frameworks, simulate her ideas and use the WEBAPP to evaluate the simulation results. The user can only choose between predefined logistic concepts.

In the following section, we will further explain the bonding layer.

V. THE BONDING LAYER

Utilizing two monolithic simulation frameworks results in problems when employing microservices. A completely clean, monotheistic architecture would require a reconstruction of the frameworks, which isn't feasible. We therefore propose to employ microservice ideas in the utilization of the tools, effectively encapsulating the rigidity of the frameworks within a flexible shell. This creates a flexible bonding layer, enabling rapid development.

Figure 2 shows the microservice-like approach to simulation using an external framework. Instead of developing a single, holistic simulation, we propose the development of small, interchangeable microsimulations. The microsimulation only implements the specifics of the logistic concept to be simulated and uses core functionalities or other microsimulations to further increase model accuracy.

 ISBN: 978-1-61208-702-3

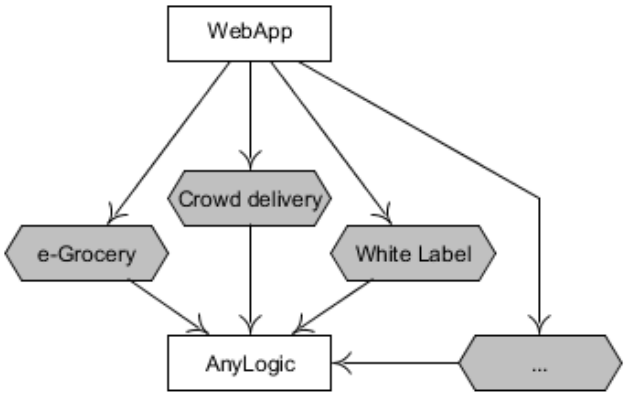

Figure 2. Microsimulations using Anylogic.

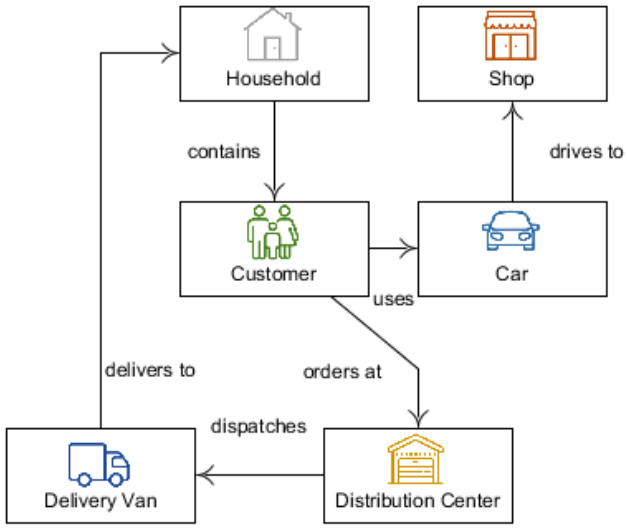

Figure 3. E-Grocery microsimulation using AnyLogic.

A. Microsimulations in AnyLogic

Traditional simulation software is often created by building a simulation framework that encompasses the technical aspects of the simulation. For example the Framework OMNet++, presented by Varga and Hornig in [8], provides an event-based simulation for networks in general. Further functionality is than added on top in the form of modules or add-ons to model specific types of scenarios. To model Internet traffic in OMNet++, an external library called INET is often used, which provides modeling of network stacks, switches, Ethernet cables, etc. The add-ons often model different domains, but no singular scenario. In our architecture we propose to build another layer on top of the simulation framework and the add-ons, with each module containing a specific, highly configurable scenario that is to be simulated.

For example, the microsimulation e-Grocery only implements the necessary supply chain for on-line grocery shopping. It uses the framework AnyLogic and its add-ons that provide GIS-support, agent-based simulation, database access, etc. Figure 3 shows the implemented agents for the simulation.

Each agent represents a different step of the digital or the conventional grocery shopping process. Following the classic agent-based simulation design, the agents all have specific behavior and interactions with other agents. The whole simulated process is a result of emergent behavior.

The CUSTOMER-agent models the customer in the e-grocery process. Depending on certain factors like age and income, the agent decides to either buy her groceries via the online store or the conventional local grocery store. If the online store is used, the agent places an order to the store and awaits delivery.

The STORE agent represents a local grocery store that provides a certain inventory and is opened during a specific time frame, depending on the store type. For the e-grocery model, a rather high abstraction for the shopping process is adequate, CUSTOMERS arrive via CAR and spend a predefined amount of time in the STORE before returning to their HOUSEHOLD.

The DISTRIBUTION CENTER however is part of the e-grocery concept. It receives orders from the CUSTOMERS and sorts the orders into delivery trips for the DELIVERY VANS to execute. Specific aspects of grocery delivery are considered during the trip planning phase. For example, if the order contains refrigerated goods the delivery window specified by the customer needs to be hit by the van, otherwise the goods might spoil. After planning trips, the DISTRIBUTION CENTER sends out the DELIVERY VANS to distribute the ordered goods.

The DELIVERY VAN is a simulation of a delivery van built for grocery delivery, containing multiple different temperature zones that can store different kinds of groceries and perishables. It receives a delivery plan from the DISTRIBUTION CENTER, is loaded with the ordered goods and proceeds to traverse a GIS map (provided by AnyLogic), stopping at the HOUSEHOLDS to deliver the cargo.

Each agent is very simple in implementation and the whole microsimulation can be replaced by another version within a month. Furthermore the agents are highly configurable. For example, the CUSTOMER follows a pre-defined schedule for daily activities, which can be configured by the user, changing delivery windows, ordered goods and traffic.

Overall we implement small specific scenarios for urban logistic in independent AnyLogic simulations, called microsimulations. Keeping scope small allows for rapid development and independent deployment.

B. Macroscopic Simulation

With AnyLogic-based simulations relatively small in scope, macroscopic events need to simulated by another framework. We use MATSim for this purpose. The aforementioned microsimulation allows evaluation of economical factors and provides a basis for the more abstract complete city model simulated by MATSim.

Figure 4 shows the general workflow of simulations using MATSim. To simulate traffic, MATSim creates a population of agents and optimizes each agent's day plan cost using a genetic algorithm, changing modes of transportation between activities. Optimization of agent plans ends after a pre-defined,

 ISBN: 978-1-61208-702-3

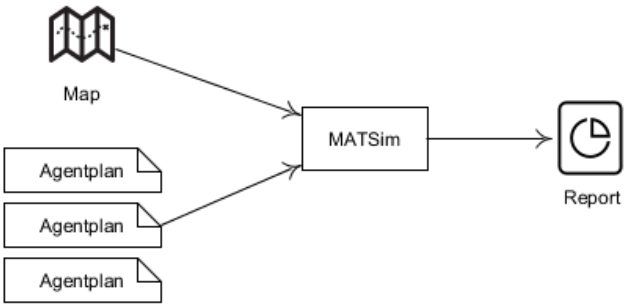

Figure 4. General Workflow using MATSim.

user-controllable amount of iterations. The simulation creates an output in form of reports, showing traffic flows, emissions and road utilization to be displayed by the WEBAPP.

VI. DISCUSSION

The usage of microservices often implies a completely new architecture and replacing the old system piece by piece with a microservice-conforming re-implementation. Microservices provide extreme advantages at scale and rapid development of new components. However, the necessary organizational structures for microservices cannot always be implemented and rewriting software is not always possible.

In our current research project, we can not rewrite extensive simulation frameworks to allow rapid implementation of new simulations. We therefore provided an adaptation of microservice principles to simulations, creating small scale microsimulations. This approach has a couple of advantages and disadvantages.

First, the usage of microsimulation relies on monolithic simulation frameworks. By modeling only small parts of the scenario, a rich framework is necessary to provide extensive functionalities like routing services, message channels and logging. A loose coupling between simulated scenario and framework is difficult to achieve. API changes to the frameworks also impact all microsimulations.

This approach also only works for simulations with a low degree of complexity. Very complex simulation models can rarely be written by a very small team in a reasonable time frame, without producing a complex to change system.

On the other hand, the usage of microsimulations allows rapid technical development of new simulations, reducing overall development time. The development of a simulation model often consists of a long phase gathering data about real life behavior and structure before beginning implementation. Reducing the necessary time for technical development frees up time for more detailed research, increasing simulation accuracy.

Furthermore, keeping simulation scenarios small allows for single person simulation development. Implementation of functionalities is reduced extremely if a rich framework can be used. By only modeling a single concept and using parameters for configuration scenarios, the necessary code base to be implemented by the developer is very small. This is a big advantage in research projects where the software is to be used as a tool in the project context.

VII. CONCLUSION

In this paper, we first presented our requirements for a web-application for city planners. The web tool aims to support city planners and decision makers in their evaluation of novel logistic concepts in regards to the impact of novel concepts on emissions, traffic flow and road utilization.

To achieve this goal, we presented a design for the decision support tool that incorporates two different simulation frameworks. The frameworks are combined using a bonding layer that bridges the microservice-paradigms and the organizational challenges in rewriting foreign software. The bonding layer applies the concept of small independent components to the simulations themselves by constraining the simulations to a single logistic concept, relying on the frameworks and other microsimulations to model more holistic views.

We also discussed advantages and disadvantages of the microsimulations. Most importantly, dependencies on simulation frameworks prevent microsimulations from adhering to a pure microservice approach. The impact of this disadvantage is however dependent on the software itself. In projects where software is produced as a prototype and not subject to further development and maintenance, the changes to used libraries do not impact the development process, as older versions of the library can easily be used. therefore, in a research project this trade-off might be considered, as some advantages from the microservice paradigm are immensely useful in small scale experimental work.

In further work, we plan to explore the applicability of microsimulations to the cloud context, as well as further evaluation of the proposed architecture.

ACKNOWLEDGMENT

This work was supported by the Federal Ministry of Education and Research of Germany (project USEfUL, grant no. 03SF0547). We would like to thank our colleagues from the Faculties for engineering and business information systems, as well as the colleagues from the other institutions and the City of Hannover.

REFERENCES

[1] Urbane Logistik Hannover (urban logistics Hannover). Retrieved April 2019. [Online]. Available: https://www.hannover.de/Urbane-Logistik-Hannover

[2] S. Newman, Building microservices: designing fine-grained systems. O'Reilly, 2015.

[3] N. Dragoni et al., "Microservices: yesterday, today, and tomorrow," in Present and Ulterior Software Engineering. Springer, 2017, pp. 195–216.

[4] J. Merz, "Microsimulation as an instrument to evaluate economic and social programmes," MPRA Paper 7236, 1993.

[5] A. Horni, K. Nagel, and K. W. Axhausen, The multi-agent transport simulation MATSim. Ubiquity Press London, 2016.

[6] I. Grigoryev, AnyLogic 6 in three days: a quick course in simulation modeling. AnyLogic North America, 2012.

[7] A. Borshchev, The big book of simulation modeling: multimethod modeling with AnyLogic 6. AnyLogic North America Chicago, 2013.

[8] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," in Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops. ICST, 2008, p. 60.

 ISBN: 978-1-61208-702-3