

What's new with open source MATSim dashboards and visualization

Dr. William (Billy) Charlton, Technische Universität Berlin
email: charlton@vsp.tu-berlin.de

Abstract

There are now many commercial tools in the MATSim ecosystem, but having open source visualizations and dashboards to complement the open source MATSim platform is also important. This paper/talk covers this year's improvements and features in SimWrapper, the fully open source MATSim output visualizer and dashboard tool from TU Berlin and DLR. This year brings cloud-native storage support, event viewer animations, large improvements to the freight, logistics, and public transport dashboard components, and a much-expanded default dashboard for standard MATSim runs.

1. Introduction

The open source SimWrapper data visualization platform is well-described in previous papers [1]. In a nutshell, SimWrapper is an open source, web based data visualization and dashboard platform that works with MATSim outputs and other transport models as well. Originally built for internal use at VSP TU Berlin, SimWrapper is now used in many contexts around the world. For an introduction to SimWrapper see the [online documentation](#) or browse some of the [public dashboard examples](#).

The remainder of this paper describes a subset of what is new and useful this year.

2. Comprehensive default dashboards

The [SimWrapper Contrib](#) in the main matsim-libs repository allows you to automatically create SimWrapper dashboards after simulation runs have finished. In addition to generating dashboard configuration files, it will also take care of executing all necessary post-process steps to generate required data files for the visualizations.

The default dashboards now include pages which dive into details on car trips, bicycle trips, public transport, air pollution, noise, traffic counts, travel time, and stuck agents. These can be displayed side-by-side for comparisons between scenarios. To use the contrib, include the maven dependency and then install the module by adding one line:

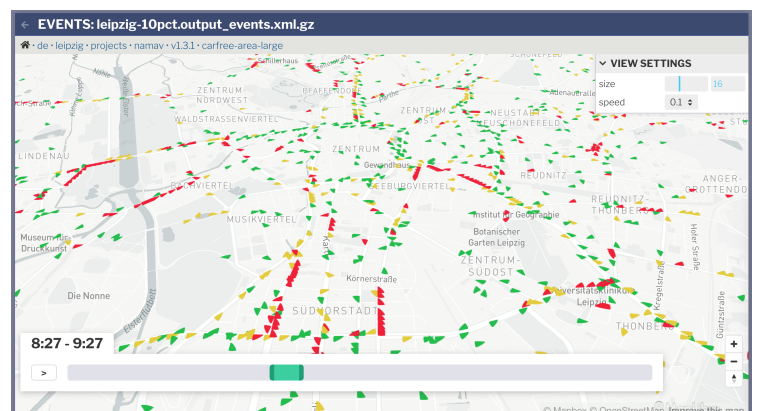
```
controler.addOverridingModule(new SimWrapperModule())
```

These default dashboards can be extended in your Java code to create visualizations specific to your use case. See the Berlin Open Scenario dashboard for a comprehensive example: <https://simwrapper.app/public/de/berlin/berlin-v6.4/output/berlin-v6.4-10pct/> and one screenshot following the paper contents.

3. Event viewer

The most exciting development this year is the new event file viewer.

Previous versions of SimWrapper included an event viewer that required extensive post-processing of the event file into a separate, smaller datafile. There were also size limits as the file needed to be ingested in its entirety by the browser tab, and large simulations would crash the tab.



Now, the event file can be streamed directly in the browser view. This new implementation leverages the use of many new technologies including the Rust language and Web Assembly (WASM) threads in the browser, allowing much more performance than standard JavaScript. We are excited for the future possibilities now that this set of technologies has been cracked!

4. Public transport

The PT viewer received some big updates this year. There is a new stop-to-stop output file which, when present, allows you to inspect station boards/alightings and link-level load factors. The details panel has a new incremental search, enabling selection of specific transit lines and/or routes. Legend colors and labels can also finally be defined in the view configuration.

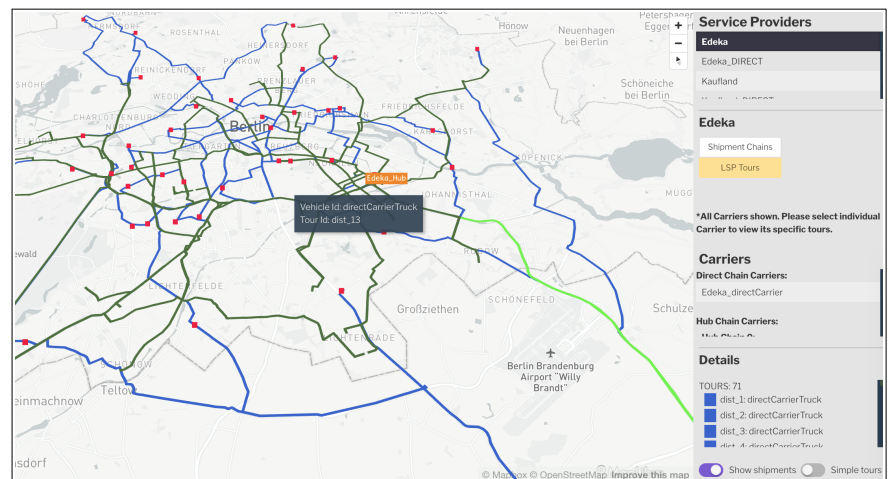


Separately there is also a new interactive flow map which shows general headway service levels between areas. Zooming out even aggregates the flows into general directions instead of specific routes.

5. Freight and logistics

A new logistics viewer supports viewing and debugging freight logistics from the MATSim [freight and logistics contrib](#).

Direct chains, hub chains, individual shipment chains and tours, plus hubs and distribution centers can all be visually inspected. Shipments can be viewed on links directly, or in a simplified "hops" view.



6. Cloud-native storage support

Since its inception, SimWrapper has supported server-based file storage via NGINX and other web servers as well as Subversion file servers, used at VSP TU Berlin. In addition, there is a Python package which allows one to run SimWrapper locally instead of using the main website at [simwrapper.app](#). This python tool can be installed with one Python command:

```
pip install simwrapper
```

New this year, the SimWrapper Python package was radically updated to include the latest build of SimWrapper with support for an advanced HDF5/OMX matrix viewer—a feature generally not used by MATSim researchers, but very helpful for non agent-based models like ActivitySim. That work also included extensive refactoring to enable cloud-based storage such as AWS S3 and Microsoft Azure.

Teams that run their simulations entirely on cloud infrastructure can now access outputs stored on S3 and Azure directly. This is done by leveraging the open source [RClone tool](#). RClone can mount remote blob storage from numerous providers and only transfers data sparsely when requested.

Thus for very large output files like MATSim outputs and OMX data files, it is an ideal match. The storage appears to be a local folder, and each file (and even subportions of files) can be accessed when needed as if they are regular files.

Current research shows that hosting SimWrapper in the same datacenter as the blob storage performs best, but even a local mount of blob storage works well with the right caching and multithreading tuning parameters. The details are available on the SimWrapper documentation site.

7. GMNS support - General Modeling Network Specification

Initial research in building a MATSim-native reader/writer of the upcoming "GMNS" file specification is underway. GMNS, the [General Modeling Network Specification](#), is a multiyear effort to produce an open, shareable, generic modeling network format, with the hope that it could do for modeling what GTFS did for transit routing data. GMNS is intended to work with multiple modeling and transport microsimulation platforms including MATSim.

This work is ongoing, but the latest findings and status can be presented in June.

8. Future challenges

The initial design of SimWrapper precluded any sort of background processing or server components beyond simple file storage. With that fully browser-based design came some great advantages but also quite a few disadvantages, too: browsers are not well-equipped to deal with the huge file sizes or processing tasks that large MATSim scenarios require.

Despite this, the fully browser-based approach allows quite a bit of flexibility—as long as users are aware of the limitations, especially on network and dataset sizes. Our development team has worked tirelessly to make working with large scenarios possible, if not always comfortable.

With this year's SimWrapper developments, including streaming of event files and cloud storage, it is possible that future improvements might include an optional server-based component. It is our hope that presenting this work will spark conversations at the MATSim user meeting that can help us steer development in a direction that is beneficial for all users of this free and open SimWrapper platform.

References

[1] SimWrapper: doi.org/10.1016/j.procs.2023.03.095(<http://doi.org/10.1016/j.procs.2023.03.095>

Example Screenshot of a typical dashboard panel (Noise, Berlin Open Scenario)

