How to prototype a client-side route planner for Helsinki with Routable Tiles and Linked Connections*

Julián Andrés Rojas[0000−0002−6645−1264], Harm Delva[0000−0001−8272−0754], Pieter Colpaert[0000−0001−6917−2167], and Ruben Verborgh[0000−0002−8596−222X]

IDLab, Dept. of Electronics and Information Systems, Ghent University – imec, Ghent, Belgium

Abstract. Route planning is key in application domains such as delivery services, tourism advice and ride sharing. Today's route planning as a service solutions do not cover all requirements of each use case, forcing application developers to build their own self-hosted route planners. This quickly becomes expensive to develop and maintain, especially when it requires integrating data from different sources. We demo a configurable route planner that takes advantage of strategically designed data publishing approaches and performs data integration and query execution on the client. For this demonstrator, we (i) publish a Linked Connections interface for the public transit data in Helsinki, including live updates; (ii) integrate Routable Tiles, a tiled Linked Data version of OpenStreetMap road network and (iii) implement a graphical user interface, on top of the Planner.js SDK we have built, to display the query results. By moving the data integration to the client, we provide higher flexibility for application developers to customize their solutions according to their needs. While the querying might be slow today, these preliminary results already hint at different data publishing strategies that may increase query evaluation performance on the client-side.

Keywords: Linked Data · Public Transport · Linked Connections · Routable Tiles · Route Planning.

1 Introduction

Route planning is a key feature for application domains like delivery services, tourism advice or ride sharing. However, each use case has its own specific needs. For example, delivery services require detailed data about road networks and building entrances. Tourism websites require data about public transit alternatives and points of interest. Ride sharing services need to check for intermodality with other services. Currently available *route planning as a service* solutions (e.g., Google Maps[1], CityMapper[2], etc.) do not offer a complete solution for

* Available online at https://julianrojas.org/papers/sem4tra2020-demo
[1] https://cloud.google.com/maps-platform/routes/
[2] https://citymapper.com/tools/1063/api-for-robots

every use case and, more importantly, do not allow extending their implementations to consider specific needs. This forces application developers to build their own self-hosted and custom tailored route planning solutions. Such self-hosted system quickly becomes expensive to develop and maintain, especially when it requires integrating heterogeneous data sources.

We consider this data integration problem an important bottleneck in setting up a route planner. We aim to lower the cost of prototyping a new route planner by automating the data integration from data sources published across the Web. To this end, we envision strategically designed Web APIs that allow route planning clients to access the relevant pieces of data they need to evaluate a particular query. The Linked Data principles [2] play a major role in the design of such Web APIs, as they facilitate data integration by reusing identifiers. Furthermore, we move data processing and query execution to the client side, allowing more flexibility for developers to tailor their implementations according to the specific needs of their use case. At its core is a Web engineering problem: how to design the Web APIs for automated querying?

This demo (available online[3]) illustrates our on-going work in automating data adoption in route planners. It introduces a new user interface built on top of an early release of our new Planner.js SDK[4]. We made it operational for the city of Helsinki, using data from Helsinki's public transit operator HSL[5] and OpenStreetMap, allowing browsers to calculate an attendee's way back to the hotel after the ICWE conference, if it would have taken place physically. Given a start URL provided by the users either at build or runtime, it can automatically find its way through a hypermedia structure to download the right fragments of the public transit schedules and road network data needed to answer route planning queries.

2 State of the Art

Route planning has been extensively studied throughout the years. Bast et al.[1] and Pajor [7] present an analysis of multiple route planning algorithms. Specialized software exists to calculate routes on road and transit networks. These can be categorized in: (i) software as a service, (ii) self-hosted server software, and (iii) client-side route planners. Software as a service that can be used for route planning includes Mapbox turn-by-turn[6], Google Maps or Navitia.io[7]. This approach is only customizable to the extent the service-provider allows customizing the route planning queries. Open-source tools to set up a route planner on your own server include Open Trip Planner[8], OSRM[9], or Itinero[10]. This approach

[3] http://193.190.127.152/plannerjs-demo
[4] https://github.com/openplannerteam/planner-example
[5] https://www.hsl.fi/en
[6] https://www.mapbox.com/use-cases/turn-by-turn-navigation/
[7] https://navitia.io
[8] http://opentripplanner.org
[9] http://project-osrm.org/
[10] https://itinero.tech

requires to set up robust server infrastructure and deal with data integration processes, increasing operation and maintenance costs for application developers.

A third approach is a client-side route planner. An early version of a Software Development Kit (SKD) called Planner.js is used in this demo. The SDK currently works for Belgian public transport data [4], and world-wide short road network queries with a republished version of OpenStreetMap (OSM) as Routable Tiles [3].

In previous work, we presented an evaluation of a hypermedia driven Web API for public transit time schedules [4]. We did back then not yet have a way to calculate road network routes, and thus also no way of combining footpath transfers with the public transit time schedules. For calculating how to traverse road networks and how to calculate transfers between public transit alternatives, we introduced Routable Tiles [3]. This is a translation of all the roads from OSM to tile set similar to the slippy map URL template[11].

3 Planner.js for Helsinki

To make a tweakable Planner.js operational in Helsinki, we had to take two steps: (i) republish the Public Transit data of Helsinki as Linked Connections; and (ii) make a specific end-user road network profile for Helsinki that can be used by Planner.js.

The Public Transit Network - We took the open datasets of Helsinki's public transit operator HSL, and translated these data from the General Transit Feed Specification (GTFS[12] and GTFS-RT[13]) into Linked Connections. We set up a server connected to these GTFS and GTFS-RT feeds, exposing the corresponding Linked Connections interface[14]. Additionally, the server exposes the Routes and Stops defined in the transit network.

The Road Network - For Helsinki's road network, we reused previously published Routable Tiles[15] derived from OSM. Also, we published a pedestrian road network profile[16] and configured Planner.js to use it to calculated road-based routes.

The majority of the effort for building this route planner for Helsinki, was dedicated mostly to develop the user interface. Converting and publishing the public transit data was simplified by using the Linked Connections Server[17]. Integrating road network data was handled automatically by Planner.js. This

[11] https://wiki.openstreetmap.org/wiki/Slippy_Map
[12] https://developers.google.com/transit/gtfs/reference
[13] https://developers.google.com/transit/gtfs-realtime/reference
[14] https://icwe.julianrojas.org/hsl/connections
[15] https://tiles.openplanner.team/planet/14/9326/4743
[16] https://hdelva.be/profile/pedestrian
[17] https://github.com/linkedconnections/linked-connections-server

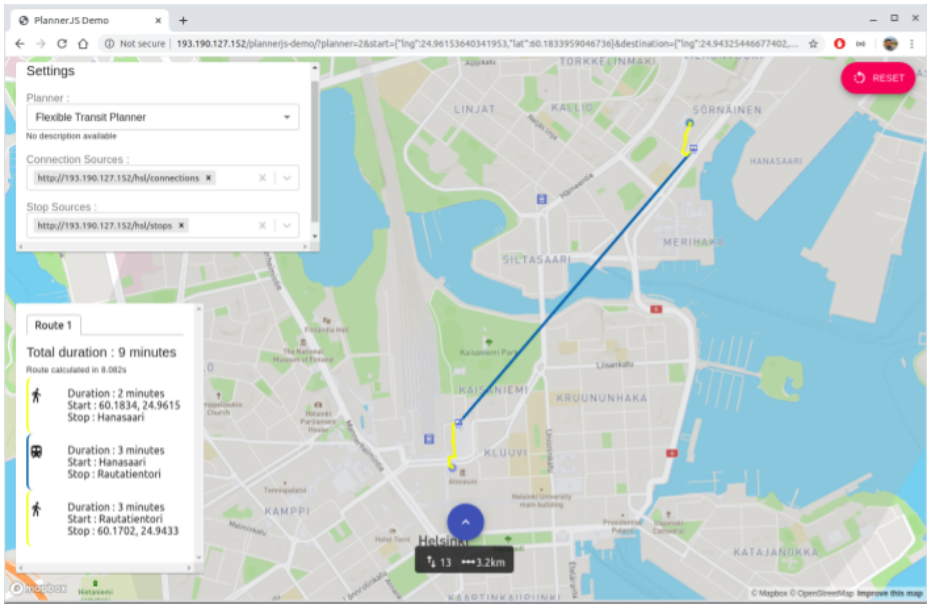

Fig. 1. Demo application showing Planner.js functional in the region of Helsinki

shows that once data has been published and made accessible through an appropriate Web API, application developers could dedicate most of their effort on improving their applications and providing a better service to their users, instead of on data integration tasks.

4 Demonstrator

TIn this demo, the users are able to choose between a *Flexible Road Planner* for road only route planning and a *Flexible Transit Planner* for road and transit route planning. Additionally to Helsinki's data, users are able to include more public transit datasources. This demo includes a list with our previously published sources from Belgium, but users may include their own. The demo is available online at https://icwe.julianrojas.org/plannerjs-demo).

5 Conclusion and Future Work

The benefits of this approach are: (i) full flexiblity for reusers: they can implement the algorithm they want, while always having the latest data; and (ii) cheaper to host transport datasets for data publishers.

The drawback of this approach is that it may perform more slowly than server-based solutions and it will need to download a high volume of data when the Linked Data APIs are not well fragmented for the specific query. However,

reusers do not need to maintain their own server and can keep their applications online using a simple file host. On-going work is making Planner.js work faster, by offering the client better preprocessed data to work with [6,5]. Clients will be able to pick the right source, also based on what source would give the fastest response to a certain query. Therefore, we will also be working on further automating the discovery of sources from a catalog instead of always mentioning the precise dataset.

We hope this early demo stimulates other researchers and open transport advocates to apply Linked Connections on the open data from their region. With the project Linked OpenStreetMap[18] we will gather all our projects on republishing OSM as Linked Data, and hope also other contributors will join in to make OSM queryable from the client-side. We also hope that HSL will start publishing persistent identifiers and global URIs for e.g., their stops, connections, trips, routes and lines, such that integration of their data with other datasets can be automated and remain valid over time.

References

1. Bast, H., Delling, D., Goldberg, A.V., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D., Werneck, R.F.: Route planning in transportation networks. CoRR **abs/1504.05140** (2015), http://arxiv.org/abs/1504.05140
2. Berners-Lee, T., Hendler, J., Lassila, O., et al.: The semantic web. Scientific American **284**(5), 28–37 (2001)
3. Colpaert, P., Abelshausen, B., Meléndez, J.A.R., Delva, H., Verborgh, R.: Republishing openstreetmap's roads as linked routable tiles. In: European Semantic Web Conference. pp. 13–17. Springer (2019), http://pieter.pm/demo-paper-routable-tiles/
4. Colpaert, P., Verborgh, R., Mannens, E.: Public transit route planning through lightweight linked data interfaces. In: International Conference on Web Engineering. pp. 403–411. Springer (2017), https://pietercolpaert.be/papers/icwe2017-lc/
5. Delva, H., Rojas Melendez, J.A., Abelshausen, B., Colpaert, P., Verborgh, R.: Client-side route planning: preprocessing the openstreetmap road network for routable tiles. In: Academic Track, State of the Map 2019. pp. 23–24. Ghent University (2019), https://hdelva.be/slides/sotm2019/
6. Delva, H., Rojas Melendez, J.A., Colpaert, P., Verborgh, R.: Decentralized publication and consumption of transfer footpaths. In: First International Workshop on Semantics for Transport. vol. 2447, pp. 1–7 (2019), https://hdelva.be/slides/sem4tra2019/#/
7. Pajor, T.: Algorithm Engineering for Realistic Journey Planning in Transportation Networks. Ph.D. thesis, Karlsruhe Institute of Technology (2013), https://d-nb.info/1058165240/34

[18] https://losm.org