A rule-based geospatial reasoning system for trip price calculations

Stefan Schenk
Development Team
taxiID
Medemblik, the Netherlands

March 19, 2018

Supervisor taxiID: Mewis Koeman
HvA tutor: Willem Brouwer

Abstract

A purely geometrical interpretation of user-defined locations would allow taxi-companies around the world to set up rules so that trip prices could be calculated without depending on distinct postal code systems. Geolocation datatypes provide part of the solution, but the benifits of geometrical definitions are lost when areas intersect. A presidence hierarchy of rules that are tied to reusable locations would eliminate these competing rule matches.

A solution is proposed to implement a microservice with a single responsibility of calculating trip prices that is accessible to existing systems and portals in which users can define the pricing rules. The company for which this system is realized requires customers to be able to migrate to the new system without downtime, while keeping the existing rules that determine the prices of taxi trips.

The core system manages user and company data, this complicates identity management in the microservice. A JSON Web Token would allow user identity to be stored in the payload of the token, thereby delegating authentication to the core system, maintaining the single responsibility of the microservice.

Preface

Before you lies the graduation report that displays all the accomplishments and research conducted during the final phase of my Bachelor Software Engineering study at the Amsterdam University of Applied Sciences, written to fulfill the graduation requirements. Allow me to briefly elaborate on the events that motivated me to reach this point in my career.

Before I began my study, I was a marine in the Royal Dutch Marine Corps. As their latin motto qua patet orbis ("As Far as the World Extends") suggests, I was sent to exotic places around the world. During these trips there was either lots of waiting, or lots of hard and dangerous work. A common activity during the waiting hours of the average marine was watching series, but I wasn't really into that. Before I was sent to Afghanistan, I bought a new laptop on which I installed visual studio and downloaded C++ tutorials, convinced that I would be learning how to program during my off-duty hours. My efforts, however sincere, didn't convince me that I was progressing my understanding of the matter very much. During one of my boat trips along the coast of West-Africa I tried once more, but this time I started experimenting with Javascript instead of C++, yielding more tangible results.

After five years I decided to quit the marines because I felt that I could accomplish greater things with a proper education, and went to an open house of the HvA, HBO-ICT. There I was confronted with my interest for logic and programming once more, and decided that I was going to become a Software Engineer. During my time at the HvA, I enjoyed the logic and elegance of algorithms and patterns in code very much, next to solving complex problems, and learning new things every day in general. I've learned some shallow concepts about machine learning in the Big Data course which interested me the most. Systems that could learn complex tasks for themselves.

Right before I started writing this thesis, I followed a minor Artificial Intelligence at the University of Amsterdam, and right before that I worked at taxiID as a vacation job. That's where I was offered an internship where I would develop a virtual assistant. This aligned very well with my minor and my interest in artificial intelligence in general. However, because of priorities within the company, the assignment changed to rebuilding a price calculation system from scratch that had to be used around the globe, qua patet orbis.

To many this may sound as a boring challenge. If you are not one of those people, you may think about the many ways in which this problem may be

solved. And this thesis will provide useful solutions for encoding locations and handling geospatial data in an intelligent and performant way. I was also challenged by the fact that my project had to supercede the existing system in effectiveness and efficiency. The moment I was introduced to my assignment reminded me of a chapter in a book called Clean Code, in which this exact pursuit is used as a common example for which the book would provide solutions; "Now the two teams are in a race. The tiger team must build a new system that does everything that the old system does. Not only that, they have to keep up with the changes that are continuously being made to the old system. Management will not replace the old system until the new system can do everything that the old system does." [1].

I would like to thank Dan Stefancu, Marco Strijker and Martin Zwaneveld for their insightful criticism that has led to the most useful lessons during my internship.

Stefan Schenk

Andijk, 01-03-2018

Contents

1 Introduction

Automatic fare estimations and calculations are one of many common features in taxi dispatch systems. A passenger books a ride, and a predicted price is displayed based on pickup and drop off locations. When the destination is reached, the system calculates the final price, with or without discounts, including taxes, and additional costs added by the driver. Taxi companies compete for customers, and prices are shifting regularly. This increases the demand for dispatch systems with easy and solid price managment features.

1.1 Context

This thesis is written during an assignment at taxiID, an Amsterdam based company providing end-to-end cloud solutions for taxi companies. Founded as a startup that successfully introduced smartphone taxi booking in The Netherlands, and offers a wide range of IT solutions to serve the taxi market, including a passenger app, a driver app, administrative panels, and track and trace hardware. taxiID solutions have proven to be a reliable set of tools for all size businesses. For independent taxi companies with 2 cars or a companies with large fleets, affordable solutions are available. taxiID's goal is to deliver affordable, time-saving solutions for taxi companies to allow for convenient planning and dispatching without requiring local installation. Tough based in Amsterdam, the development team is located in Medemblik, consisting of two mobile developers, two backend developers, a designer and two project managers. Clients are located across the globe, introducing challenges when developing applications that rely on clearly defined locations and infrastructures that vastly differ between countries.

1.2 Assignment

YourDriverApp (YDA) requires a pricing calculation functionality that is similar to the existing taxiID implementation. All functionalities within the current system align with the clients demands, but some features introduce difficulties, for example: region names are too vague for specific database queries. Some features could be abstracted so more possibilities can be implemented, some features are still unimplemented, and some features could be improved along the way. A system must be implemented in which group

admins can define pricing rules based on user defined locations and time schedules, that can be used for calculating a passengers trip price, or show prices of different products based on the trip the passenger is about to make. For example: a passenger may book a taxi ride from Utrecht to Schiphol using the passenger app. Available products are presented with their respective prices based on the distance and duration of the trip using the pricing rules that were created by the group admin of a taxi company. The system must be usable in countries with a poor postal code system. There should be a way for a group admin to describe locations in a way that are precise and consistent with reality, meaning that a defined location should be usable from outside of the system, or at least be interpretable. An example of this requirement would be: a taxi company that operates in Afghanistan. A passenger wants to be picked up on some road near the mountains. How would a group admin describe that location in order to define a price beforehand? The system should be accessible to other systems, meaning that applications that currently rely on the old system should be able to migrate to the new system. As the old system shouldn't be used for new applications, as it was not designed for this use case, the new system should. It should have a single responsibility, and should be atonomous in that regard.

1.3 Research Objective

Three main challenges that construct the assignment can be identified. Research must be done to attain the best possible way of mapping locations to pricing rules. What this means is that locations must be storable, comparable, and interpretable. The database must be able to store locations in an efficient manner, to which queries can efficiently be made in order to find out whether a pricing rule applies to a given ride. For this to be the case, the stored locations must be comparable to the location of the passenger, or the destination. The user must be able to reason about his pricing rules, from which an understanding of his defined locations logically follows.

Secondly, a system has to be developed that encapsulates the solution that is the result of the conducted research. It is helpful to extend the research of the problem to finding out how to incorporate the answers into a working system, where architecture has a major influence in the tools that are available. For example: if a solution to the main problem requires a database system capable of handling high quantities of geospatial queries, this requirement has to be satisfied in order to proceed in finding the final

solution.

Finally, a portal has to be created that enables users to define the pricing rules. The complexity of the portal depends on how straight forward the price calculation system is put together. The best way of making the systems capabilities available to the user through the UI in the portal, must be investigated.

1.4 Questions

From the description of the problem, one main important research question can be derived:

How can a generic location-based price calculation system be implemented that is usable around the globe?

Which varieties of address formats exist around the world?
Is it possible to redefine different address formats as a generic address?

- Gps
- Postal code
- Intersecting areas
- The hotel case
- The airport case

To what extent do address formats have an impact on performance of a price calculation?

2 Literature Review

3 Chapter 1 - proposed approach

4 Realization

This chapter describes the process and techniques used while implementing the solution.

4.1 Methods and Techniques

Before development could commence, requirements were described as user stories in Jira, a project management tool. The user stories were exported to a backlog, where the tasks were defined more precisely. The product owner prioritised tasks so that the most essential features were developed first.

To produce software that meets the expectations of the product owners, a sprint was planned every two weeks, joined by one of the product owners, and members of the development team that were responsible for the systems architecture, where issues or tasks were well-defined and prioritised.

5 Conclusion

6 Recommendations

7 References

8 Glossary

9 Appendices

10 Reflection

References

[1] R. C. Martin, Clean Code: A Handbook of Agile Software Craftsmanship. Pearson Education, August 2008, pp. 4, 5.