A rule-based geospatial reasoning system for trip price calculations

Stefan Schenk

Supervisor: Willem Brouwer

Advisor: Mewis Koeman

Department of Software Engineering
Amsterdam University of Applied Sciences

This dissertation is submitted for the degree of
Bachelor Software Engineering

May 2018

Todo list

Table of contents

Chapter 4

Trip Price Calculation System

4.1 Introduction

This chapter clarifies which information should comprise a price breakdown to reflect that of the legacy system, what logical flow of information is to be contrived, and how different pieces of information that are stored and processed restrict the time and space dimensions of a price rule without blurring the straightforwardness of the system.

4.2 Breakdown

To ensure a seamless transition from the legacy price calculation system to TPS, the response formats should be identical. Still an improvement, if profitable enough, could be taken into consideration. One requirement of the price breakdown states that the tax should be included, but as shown in Listing 4.1 the included tax is part of the breakdown. Is it by mistake or design?

```
[
  {
    "vehicleType": "saloon",
    "maxPassengers": "4",
    "price": {
      "currency": "EUR",
      "total": 850,
      "breakdown": {
        "route": 802,
        "tax": 48,
        "toll": 0,
        "parking": 0,
        "waiting": 0,
        "discount": 0
      }
    },
    "fixedPrice": "true"
  }
]
```

Listing 4.1 Legacy price breakdown

```
[
  {
    "vehicleType": "estate",
    "maxPassengers": 4,
    "isEstimated": false,
    "price": {
      "breakdown": {
        "route": 8300,
        "toll": 0,
        "parking": 0,
        "waiting": 0,
        "discount": -1650
      },
      "currency": "EUR",
      "total": 6650,
      "tax": {
        "amount": 400,
        "percentage": 6
      }
    }
  },
  ...
]
```

Listing 4.2 Improved price breakdown

Two possible solutions were proposed having VAT included in the price. The first solution extracts the tax element from the breakdown, so that the sum of the breakdown would add up to the total price where VAT is included in the price as shown in Listing 4.2. As demonstrated in Appendix B.2, a breakdown is easily constructed in four steps when VAT is included. Keep in mind that unlike the listings the prices in the proposal are not displayed in cents. The

second solution maintains the legacy format, but has to recalculate the prices without VAT. This could have downsides unlike the first approach:

1. If an error is detected in the calculation, it is hard to trace back which components contributed to the total VAT. This would be even harder when each component uses its own VAT percentage.
2. It takes extra steps to calculate the price of each component excluding VAT.
3. Rounding the individual components could result in a sum that is not equal to the total displayed in the breakdown.

The first proposal is chosen to be implemented, where the flag 'fixedPrice' is replaced by the 'isEstimated' flag to clearly reflect its purpose.

4.3 Timeframes

Next to the three dimensions of space, time will play a role in determining whether a rule has matched. The implementation of this concept should preferably offer enough freedom in the future, and should not be tailored toward one specific entity relation. Being able to reuse the timeframe entity improves maintainability of the system. The requirements state that the user must be able to define a start and end time, the days on which the times are active, and the start and end date of the timeframe. The timeframe is used to describe when rules or discounts are active. If for example a discount should be active during night of New Years Eve, between 23h and 5h, this description of a timeframe is already lacking.

kks32: Why?

4.3.1 Conventional Approach

The legacy system takes a straight forward approach of storing time in a relational database. The begin and end of a window are stored in a record that is related to a parent timeframe entity. The timeframe has many windows that could contain a timestamp. It either finds one or many time windows that contain the timeframe. This approach covers all possibilities imaginable.

4.3.2 Bitmap

4.4 Data Model

4.5 Logical Flow of a Price Calculation

- authentication - extracting companyId and daAppInstallId - extracting departure and destination coordinates - using directions service to aqquire distance and duration - converting to minutes and km - executing query - find daAppInstall model with matching companyId and daAppInstallId - find company country - find matching discounts ordered by priority - enabled - timeframes - departure - destination - find matching rules ordered by priority - enabled - timeframes - departure - destination - get pricing information of all products for highest priority rule

A tier price system, that calculates fixed prices based cascading thresholds, and a dynamic pricing system that calculates prices per distance unit and minute is a very specific problem that must be split up into sizable categories. The term 'distance unit' is used on purpose, as distances are measured using different metrics in various countries. Pricing rules should be constrained by time frames, making rules available only for some hours a day, or only on christmas for example. Rules should be specifyable per product as different vehicle types have different prices, but are included in the same pricing rules. Discounts may be calculated with the trip price, and VAT should be displayed in the price breakdown. Some additional requirements to the system may be added in later phases, as Scrum is used to manage work iterations (this fact is covered later in this chapter). 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 system should have a single responsibility, and should be autonomous in that regard.

References

[1] U. T. Inc. (2011) The uber story. [Online]. Available: https://www.uber.com/en-NL/our-story/

[2] J. R. Herring, "Implementation standard for geographic information - simple feature access - part 1: Common architecture," May 2011. [Online]. Available: http://portal.opengeospatial.org/files/?artifact_id=25355

[3] (2004) Geographic information – simple feature access – part 1: Common architecture. [Online]. Available: https://www.iso.org/standard/40114.html

[4] J. R. Herring, "Implementation standard for geographic information - simple feature access - part 2: Sql option," August 2018. [Online]. Available: http://portal.opengeospatial.org/files/?artifact_id=25354

[5] (2018) Postgis 2.4.5dev manual. [Online]. Available: https://postgis.net/docs/manual-2.4/using_postgis_dbmanagement.html#PostGIS_GeographyVSGeometry

[6] (2018) Mysql 5.7 reference manual - geometry class. [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/gis-class-geometry.html

[7] I. K. Center. (2018) Three-tier architectures. [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSAW57_8.5.5/com.ibm.websphere.nd.doc/ae/covr_3-tier.html

[8] R. Stephens, *Beginning Software Engineering*. John Wiley & Sons, 2015.

List of figures

List of tables