# Concerning DTO project

## The ascertaining part.

### The initial problem statement (as we understand before previous stage)

There are several breweries. There are distribution centers where beer is delivered in large quantities from factories using heavy trucks. The car is loaded at the plant, transports a batch of beer to the client and returns empty to the same plant.

The problem to be solved: during the back empty run, execute a third-party order, thereby receiving additional income from the use of the capacity of the used transport.

### What we achieved during the first stage (from June 03 to August 05)

In the first phase, we worked with a single plant in Houston, TX to clients in Texas (we switched to the Oregon brewery during last week of the project).

The fundamental possibility of solving the problem is shown by examples of third-party orders from the exchange.

Taking into account the conditions for costs, temporary restrictions on the operation of transport and the level of prices for spot transportation, the economic feasibility of this decision is shown.

Thus, the concept under consideration is proved in a draft version.

### Tools and data that we used during the first stage

To simulate the operation of the transport system, we used the RAST expert system (hereinafter referred to as the expert system or RAST).

To work with the Texas road network, we used data from the OSM open platform, converted into the format required for the operation of the expert system.

As a source of information about third-party orders we used data from truckstop.com exchange

A source data about the internal orders schedule was provided by DTO team.

### Unsolved fundamental problems of the first stage concerning the initial data for calculations

1. Incompleteness of data on the rules of transport (work and rest regime, penalties for violations of this regime)
2. Incomplete data on the transport itself
   1. total number of vehicles involved
   2. types of vehicles
   3. capacity characteristics
   4. operating costs depending on the type of vehicle
   5. compatibility with cargo of various types (we considered all vehicles as Vans)
3. Incomplete data on the possibility of disrupting business processes (is it possible to be late for the brewery for the next load)
4. There is no data on whether toll roads are used
5. There are questions about the correctness of the data provided by the exchange
6. The number of orders provided by the exchange is lower than expected

## Purposes of the next stages

### Fundamental questions for the next steps.

1. The most common understanding of what should be the “output” of the entire project as a whole
2. What should be the scale of the project, both in terms of geographic scope and the problems that the project solves
3. What is the estimated time frame for the project based on paragraphs 1 and 2
4. Number of project stages based on 1 - 3
5. What are the bottlenecks or risks of the project?
6. Estimated resources and financing (self-sufficiency is possible)
7. What is the intended architecture of the solution
8. Technologies used

### Proposed second stage.

We believe that the content of the Second stage should be determined by the unresolved issues of the First stage and analytical work on the elaboration of pp. 1-8 with the preparation and execution of the relevant documents.

### Summary.

During the next stage, we should determine the exact formulation of the task that we want to see at the end of the entire project (with the assumption that minimal modifications and risks of changing business processes are possible; it may be necessary to conduct a study of the effectiveness of different scenarios in a simplified form) and decide on such a data structure that would solve any of the above tasks. Then we should start automating the selected scenario from simple to complex.

Depending on the path chosen, the Catalyst stage will require at least 4 FTEs from the RAST team

# Application. Coverage of some issues related to further steps in the project.

## Optimization

There are several scenarios of optimization.

1. **Current scenario.** In it, we have an empty run, one driver in the vehicle and a temporary restriction on the movement of the vehicle. We thought that this limit could be 20 hours or 12 hours. In this case, the vehicle moves without breaking for rest or snack, i.e. does not comply with US (or state) law. Very different solutions are obtained depending on this parameter. We believe that any cargo can fit into a vehicle, i.e. we do not take into account either the type of equipment (by default, we believe that the goods that we unloaded are suitable for the type of body that the vehicle has), nor the weight and volume of the cargo, although unloading has the characteristics "Weight" and "Length" ... The calculation is done in iterations. For the 1st iteration, we take all empty runs and all orders that we managed to unload from the exchange and consider the modified assignment problem. For each next iteration, we throw out from the problem statement those external orders that we managed to compare with the empty run at the previous iteration and solve a new problem. We solve the problem until external orders are added to backhauls, but no more than 5 times. The key metric of the solution is the benefit received from the fulfillment of the order, which is calculated based on the cost of the order, the cost of a mileage and the operating hours of a particular vehicle. In our case, we believe that the prices are the same for all machines, only the cost of orders is different. In the current scenario, not completed yet:
   1. No matching external orders have been allocated to the same empty runs. We propose to solve it as follows: for our economic model, we believe that the cost of the mileage of all cars is the same, and all cars have the same characteristics, we can assign not a fixed route, but a line of movement of cars and select orders for it. Obviously, for all cars performing an empty run along this line, the benefit from one external order will be the same. Therefore, it is enough for us to select the maximum of orders for the line, and after the call has been made, distribute orders for specific machines.
   2. The shipper does not assign a cost for most of orders. In this case, the cost is taken based on the average market values using the RateMate service. For some orders, the cost is $ 1. This means that the shipper is ready to bargain with the carriers and will choose the one who will offer the best conditions (perhaps, not only the price affects). This parameter is required in the final solution.
   3. Different types of truck cabins are not connected. It is not clear what is the difference between them in terms of operating mode, since legislation imposes its own restrictions.
   4. Automatic pulling up of the economy is not enabled for cars belonging to different carriers. Prices for the movement of cars are set manually for each task
   5. If the shipper refuses to transport, the order placed by this shipper cannot be accepted for further consideration. When the pool of rejected orders reaches a certain size we need to calculate a new task, which, possibly, will contain a subset of orders from the previous unloading, for which calls have not yet been made
      1. The criteria for which it is necessary to do the recalculation are not defined
      2. The process of excluding certain orders from consideration (those for which a refusal came and those that have already been confirmed for transportation) is not automated
   6. Conditions are generated in a semi-manual mode using spreadsheets. This happened because from the very beginning there was no agreement on the final format of the input and output data. This format underwent several changes during the first phase of the project.
2. **Scenario with changing drivers.** In this scenario, a mechanism is assumed when at one of the points of the route a new driver gets into the car, and the old one leaves the workplace and goes to rest. In this case, we can reset the vehicle's working day length timer. This scenario involves either replacing the driver at one of the stopping points for loading / unloading the vehicle, or adding an additional operation to the route (the point at which the driver will change). If the driver change point is known before building the route, this problem can be solved using the modified assignment problem. If the driver's transfer point is determined from a set of known transfer points, but depends on the route configuration, then this is already a dynamic task and it requires additional serious efforts to solve (such a task will not be solved using the algorithms of assignment problems and will require a serious change in the solver code, since no widely used solver solves a dynamic problem). This task will require modification of the output to indicate where and when a driver change is required.
3. **Scenario 1, taking into account the driver's work and rest regime.** It is supposed to take into account the legislation related to the mode of work and rest with the help of stop windows (when the car cannot go). The stop window can be static (at a predetermined time, the machine stops where it is; such a scenario is currently implemented) or dynamic. Plus, restrictions are imposed related to the type of car cabins (some cars can rest in an “open field”, and some are required to call in specially designated places for rest, where there is a hotel). As in the second scenario, a dynamic task will require serious effort to refine the solver code. We have two scenarios for dynamic stop windows:
   1. There is a set of points on the map where the driver can rest. The program is obliged to plot the driver's route in such a way that, no later than a certain time, he gets to one of these points. Such a formulation would require a set of points where the driver can rest.
   2. The driver must choose whether he will rest before loading the external order or after. This may depend not only on the driver's desire, but also on the order parameter. This will require additional input to represent this moment.
4. **Scenario with one or two drivers in the vehicle.** A car with 2 drivers can go without stopping. While one driver is resting, the other is driving. Although here it is necessary to clarify the restrictions on the duration of the trip. Two drivers can only be in a certain type of car. This problem is not solved with the help of the assignment problem, while benefits are calculated for a specific car for a situation when there is one driver in it, or, accordingly, when there are two. In this case, the cost of moving the machine changes depending on this. If, nevertheless, there are restrictions on the time of movement of a car with 2 drivers, then scenario 3 will have to be connected. The decision for a car is made based on the calculation of all metrics for all orders and all cars with all variations (when in those cars in which it is possible there is one driver or two). This is already a purely optimization problem that cannot be solved by linear programming methods (using the assignment problem), since when trying to solve it using the assignment problem, an exhaustive search occurs (we are looking for where there should be one driver, and where two)
5. **Scenario adding more than one order to the empty run with a rigidly fixed internal schedule.** This problem is not solved with the assignment problem. This scenario has been partially implemented now.
6. **Floating Internal Schedule Scenario.** This is a full-scale task of planning for the next day. When solving it, you can get the maximum benefit with the delivery of external orders
7. **Scenario for calculating the minimum profitable price for external orders.**

All of the above scenarios are not mutually exclusive, but may complement each other. For the final selected scenario, the input and output data structure and workflow should be defined.

## Mapping and road network data and metrics based on this data (distance matrices for calculations)

Depending on the scenario you choose, you can use different mapping or metric data. But there are questions about the data from truckstop.com.

We have a city of departure and a city of destination, as well as the distance from the city of departure to the point of loading and from the city of destination to the point of unloading. This gives the distance between the loading and unloading points. It is unclear how truckstop.com calculates this distance, as this value is not additive. We can only assert that the distance between the points of loading and unloading will not be more than the sum of these three values ​​(two distances from cities to points plus the distance between cities).

Let me explain with an example. Suppose cities A and B are connected by a straight highway that extends to both city A and B. The distance between cities is 200 miles. 30 miles from city A is the loading point, 20 miles from city B is the unloading point. The maximum distance between the point of loading and unloading in this case will be 250 miles, and the minimum distance is 150. For a speed of 70 mph, the difference will be 1.5 hours. Plus 100 miles of run itself. For the HTN tariff, we get the difference in cost of 100 \* 1.18 + 1.5 \* 19.76 = $ 147.64, which can be a significant part of the benefit.

In other words, we need a clear explanation of why we may or may not rely on Truckstop.com data.

The following sources can serve as an alternative to these data:

1. Google maps. Below I will give an argument why you shouldn't use them. For the calculation, it is necessary to obtain a matrix of distances and travel times. Distance and travel time between one pair of points counts as one request to google maps api. To calculate the problem according to the first scenario, we need ~ 2 \* m \* n requests, where m is the number of machines for which the calculation is made, n is the number of external orders that we consider for this set of machines (we need to get all the distances from the starting point of the empty mileage to the start points of external orders and from the points of completion of external orders to the finish point of the empty run). For the sake of simplicity, let's assume that we count the tasks for each brewery separately. Suppose we have an average of 50 runs per day for one plant. For each task, we consider an average of 200 external orders. Thus, every day we will send 12 \* 2 \* 50 \* 200 = 240,000 requests to Google maps. Or ~ 7,000,000 monthly, which will cost $ 35,000 / month (https://cloud.google.com/maps-platform/pricing). It is also unclear what performance can be obtained with such volumes of requests.
2. Another alternative is to use your own OpenStreetMaps server. The quality of cartography there is practically not inferior to Google maps in the United States, but the maps themselves and the necessary cartographic data are free. Here the costs will be associated only with the preparation and installation of cartography on the server and their integration with the solver. Another major advantage OpenStreetMaps has over Google is that we have our own road graph editing tool available. We can change the fare on roads, change the speed limits, or delete roads altogether if we do not want to use them in our routes. The rough estimate for OSM implementation is 2 FTE for 2 months.

## Solver

We suggest using RAST to solve the optimization problem. RAST is not a completely ready-made tool for use in any of the listed scenarios, but the ability to adapt to any of the scenarios, including scenarios with dynamic tasks, is built into it, which sets it apart from OrTools or Gurobi. In addition, OrTools is poorly parallelized - only different instances of the process can run on different cores, which will not allow counting any large tasks without decomposition (decomposition will lead to a significant deterioration in the quality of calculations).

In any case, most likely, the computational kernel will have to be changed, and this is much easier to do with a tool that you created yourself than with a third-party code.

In the case of RAST, the mapping developed from OSM data is included in the RAST license. When developing an expert system based on third-party solutions, the cost of cartography is not included in the cost of work. During development stages, RAST is provided free of charge.

## Hierarchy of users.

It is necessary to clearly define what components of other systems the created system will interact with, as well as the hierarchy of users and the functionality available to each of them.

## Technology stack

It is proposed to use the Microsoft Azure platform to deploy core services. This platform has its own SQL-like database - MS SQL, its own web-server - IIS. This platform supports code written in C ++. Moreover, this platform allows you to run applications written for unix-like systems.

It is proposed to use C ++ as the main language for writing Backend code, since this language is designed specifically for computationally complex tasks (and this is what we will deal with; RAST is written in C ++) and has the ability to directly manage memory, unlike translated languages ​​such as Python, which increases code performance and reduces power consumption.

In addition, the utility for downloading data from truckstop.com is already written in C ++ and can be integrated as an IIS web server service.

**P.S.**

This document is a statement of thoughts and possibilities regarding the project and is not intended to impose one or another solution