

Analysis of Alternative Advisory Systems for Truck Dispatching

Aline Veronese da Silva, Bruno Grau Makowski, Ellen J. Bass, *Senior Member, IEEE*, Eric Danziger, Fernando Fialho Bassols, and Lia Buarque de Macedo Guimarães

Abstract—This paper describes a resource allocation system concept for assigning petroleum product requests to delivery trucks in an oil distribution terminal. Currently the activity is completed manually (with paper spreadsheets), without any formal documented procedures. Prior effort proposed two human-computer interfaces to aid the dispatchers. In this work, the dispatchers reviewed the designs, presented to them as static pictures. Using this feedback, the requirements were re-evaluated and a functioning prototype was designed. Usability tests were conducted with the two prior interfaces and the new functional prototype. The results showed that a combined interface, where the user can select the type of display is more appealing to users.

I. INTRODUCTION

THIS paper presents a dual-interface system to support decision-making in a resource allocation task. The task involves allocating petroleum products requests from gas stations to trucks from two trucking companies. Truck drivers travel to the petroleum product distribution center, fill their trucks with the appropriate products, and deliver the products. The operators who are responsible for making these allocations of requests to the trucks are called dispatchers and the process of making these allocations is called programming.

In prior work, requirements and two interface design concepts were developed [1]. The first interface was a matrix, created to display all possible driver/location assignments. The trucks were represented as rows and the requests as columns. The display's main function was displaying possible allocations using the intersections of rows and columns. The design concept was implemented in Excel, and supported sorting and user control of which variable (such a truck driver waiting time) was prioritized.

The second design concept was based on a map interface, which used geographic information to support decision making. This interface included an image of the states of Rio

Grande do Sul and Santa Catarina in Brazil, with the requests placed according to their physical location. The focus of this map interface was to help with routing, when a truck could accommodate more than one request. To ensure that the map was not cluttered, the truck specific information was presented in several columns in the right bottom side of the screen, visually separate from the map.

In the fall of 2006, these two display concepts were shown to dispatchers in Brazil. The dispatchers thought that the matrix was confusing and they felt that it overloaded them visually. The map was simpler, but they felt they already knew the geographic location for the client gas stations and therefore were not enthusiastic about the display.

The work described herein refined the design concepts in order to accommodate the dispatcher preferences. A task analysis for a portion of the dispatching task was completed. The requirements were refined. A functional prototype was developed, and compared to the first two interfaces in a usability study.

II. REQUIREMENTS ANALYSIS

To refine the requirements, the task to be supported had to be clearly defined. The dispatchers in Brazil have developed their own methodology for accomplishing the programming task. They balance several important decision variables.

The most important variable is the total amount of product (measured by volume) delivered by each trucking company. By contract, each firm is promised a ratio of the total deliveries per delivery period. While this ratio has to be maintained, it is more critical near the end of each contract period (because that is when management checks the ratio).

Another critical issue is the relationship with the client gas stations. If a request is not filled on the day requested, it is given high priority at the start of business on the next day. These requests have to be fulfilled in a timely manner. Also sometimes requests included delivery times. These requests mean that the driver needs enough time to fill and get to the station to meet the time constraint.

The dispatchers also consider the fairness in the truck driver's route assignments. They try to track which drivers have been getting good (those with shorter distances coupled with higher delivery fees and/or higher quality roads) or bad routes, to make sure no driver receives preferential treatment. The dispatchers try to equalize the drivers' salaries (based on delivery quantity and distance), so that the drivers will receive similar pay.

Manuscript received April 18, 2007. This work was supported in part by the U.S. Department of Education's Fund for the Improvement of Post Secondary Education (FIPSE) and Brazil's Coordenação de Pessoal de Ensino de Nível Superior (CAPES).

A. da Silva, B. Makowski, F. Bassols and L. Guimarães are with the Universidade Federal do Rio Grande do Sul, Porto Alegre – RS – Brazil CEP: 90035-190 (e-mail: aline@virginia.edu, bruno@virginia.edu, bassols@virginia.edu, lia@producao.ufrgs.br).

E. Danziger is with the University of Virginia, Charlottesville, VA 22904-4747 USA (e-mail: ecd3a@virginia.edu).

E. Bass, corresponding author, is with the University of Virginia, Charlottesville, VA 22904-4747 USA (phone: 434-243 5531; fax: 434-982-2972; e-mail: ejb4n@virginia.edu).

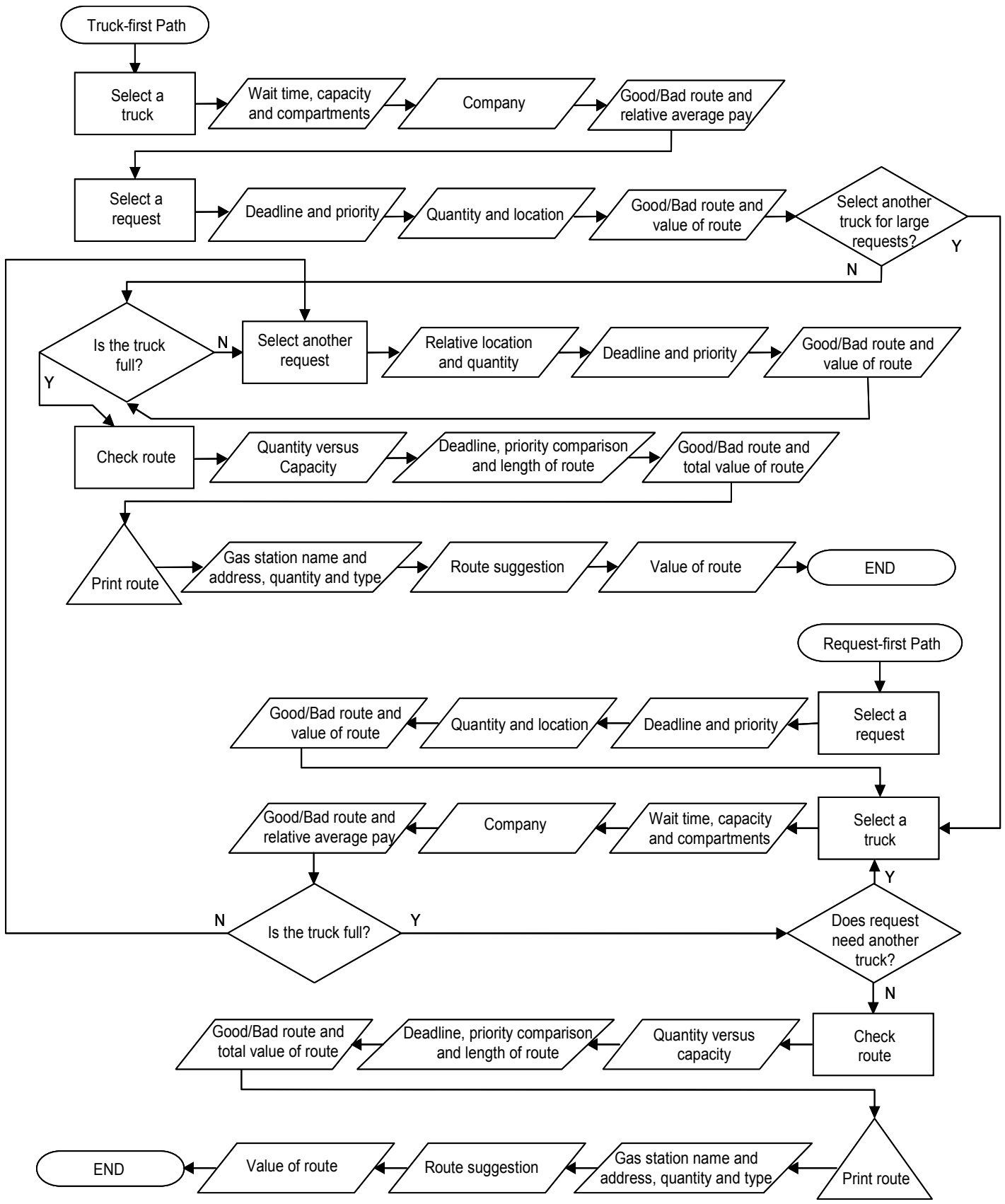


Figure 1 – Task analysis

In the dispatching task, dispatchers start with a truck (truck-first path) and assign requests or with a request (request-first path) and then assign a truck. Thus the task analysis included these two paths (Figure 1).

In the truck-first path, truck selection is first based on driver waiting time as the drivers come and wait in a queue in front of the dispatching window. Also important is the capacity of the truck, impacting which requests the truck can carry. The trucking company that the driver represents is also important due to the contract ratio. Finally, fairness data such as salary and route quality are assessed, to balance driver treatment.

Once a truck is selected, requests must be allocated to it. For requests, deadline and priority determine which should be dispatched first. The quantity of the request is then considered. For the dispatcher, it is best to handle as many requests as possible with each truck. Drivers also prefer multiple requests per trip as salary is higher for multi-request routes (since all requests use the distance from dispatching to calculate salary). The value and fairness of the request or route is also important, since this will be the part that balances the truck driver's salary and fairness. Since these variables are only to help the truck drivers, they are lower priority.

The request-first path has similar decisions to the truck-first path but in a different order. There are small differences in which variables are most closely considered at which point in the decision-making path.

Prior work [1] described functional requirements for the human-computer interface concepts. These were adapted in this work. The prior effort supported additional administrative duties, such as getting the requests from the clients and updating the list of truck drivers. This work only supports the assignment problem. That is, the requirements focus on supporting the two decision paths.

The system should show current unfilled requests and show a list of truck drivers, along with the information necessary for each. This includes quantity, priority or deadline, location, value, and fairness for the requests. For the trucks, wait time, capacity, company, fairness and salary information need to be displayed. These, along with sorting functions, a log of all actions, and assigning trucks to routes, were the functional requirements taken from the previous effort.

New requirements included how requests were displayed depending on what truck was selected. In the prior work, splitting requests across trucks was difficult to achieve. The new requirements were designed to ensure that a request will only be split into two or more trucks when it is necessary (but it would still be easy to do it). If a selected truck had a smaller capacity than an unallocated request, the request would be displayed to indicate that it was not optimal for that truck. Likewise if a large request was selected first, all the trucks with smaller capacity would be displayed differently.

The previous designs automatically created possible routes with secondary requests after the truck displayed

was assigned a request below its capacity. This requirement was changed to show secondary requests that will fit in the truck. If the user felt that another request was close or made a good route, he could select it.

The trucks have a unique login code used to track their location. Wait time and truck availability can be calculated from this information. Departure time of the trucks from the distribution center was selected for display because the estimated time of arrival is difficult to estimate.

III. FUNCTIONAL PROTOTYPE

The functional prototype (Figure 2) was designed as a web application. It is written in PHP, MySQL, Javascript, and HTML. PHP and MySQL allow for server-side processing and database manipulation. For speed, Javascript is used for retrieved data manipulation and human-computer interface actions that do not require database access. To support database interaction, Javascript makes calls to the server for PHP-generated database information. It updates the display without reloading the page. This asynchronous approach supports dynamic interaction with the system, similar to what is expected of a desktop program.

The database is designed as a set of seven tables. The client table includes the client name and billing information. Because a client can have multiple gas stations, another table of station information includes location information, address, and a client ID. Location is stored with latitude / longitude information. The request table includes a reference to a gas station, as well as including quantity, type, and deadline information. Truck data is stored in a table with driver and license information, company name, truck capacity, and number of compartments. Salary and route information is stored in a separate table for each truck. Truck status, including availability, is stored in a separate table.

The prototype includes functionality to create simulated scenarios. These scenarios can be small and pre-determined to accommodate testing between the three designs. These smaller scenarios use less than five trucks and less than ten requests. There is also a 'random' scenario, that creates larger, more realistic dispatching problems every time it executed.

The prototype includes a map display, based on the map design in [1]. The features of the previous year's display design were kept the same, although the map does not automatically generate routes as discussed previously.

The truck information is displayed almost the same way as in [1]. The rows were made larger and salary information was displayed relative to the mean with positive and negative numbers, to make the truck display easier to use.

The prototype also includes a more detailed display that can be viewed instead of the map. This was not a matrix, as in [1], but instead a list of requests. This list was more familiar to the dispatchers, since their current method includes a list.

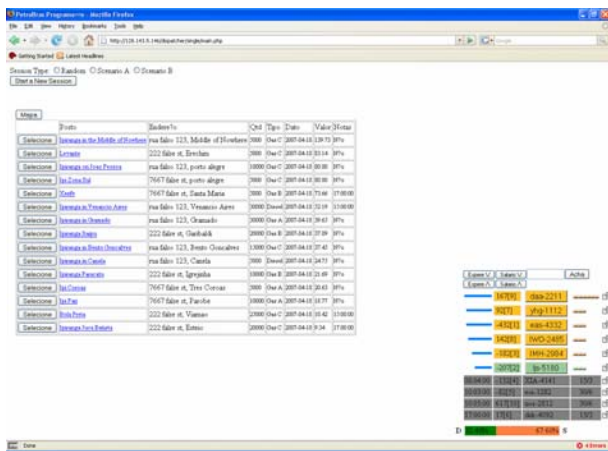


Figure 2 – Functional Prototype

For requests, the list display includes quantity, location, priority/deadline, and value information. Compared with the map display, which only shows quantity and location, this information is more helpful in decision-making.

The list was designed to work in concert with the map display. With the map displaying location information and the list displaying the rest, the two can be used together to make better decisions in the solution of the dispatching problem. A button at the top of the screen allows the user to switch between the two interfaces, without having to refresh the browser window or reselect a request or truck.

IV. EVALUATION

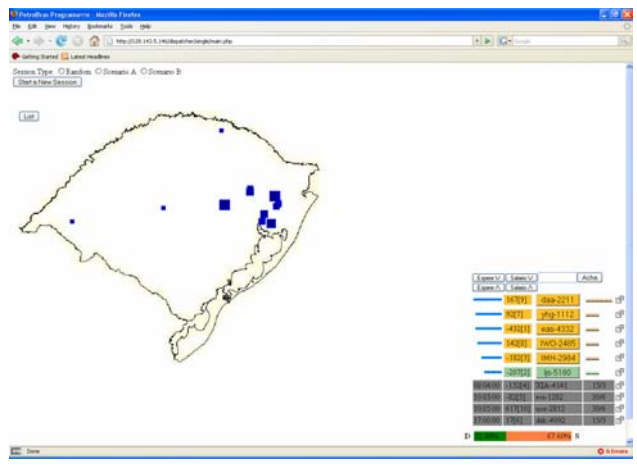
A. Usability study methods

Six undergraduate students participated in the usability study. According to [2], 2-4 users are sufficient to provide preliminary usability data. Six participants were used because they had no prior experience with the programming task. These participants would be similar to new employees.

Three different interfaces were tested. For the map interface, a series of screens were created and run in a Flash application, while for the matrix interface, a series of Excel spreadsheets were built. The functional prototype was the third tested interface.

Prior to testing, all subjects were briefly trained on the goals of the task. Also, prior to using each of the three interfaces, participants were shown how the information was laid out and how the interfaces worked.

Three programming scenarios simulated the most common situations in a programming task: distant routes, close routes, available and unavailable trucks. The order of scenarios was the same in all the tests. These scenarios were the same for all the tested interfaces. The first scenario, the easiest one, presented three requests in different locations and two available trucks, such that one route included two locations. A second scenario presented one large request (which required two trucks to deliver it) and a small request with three available trucks. The third



scenario included requests from five locations to be allocated to three different drivers.

For the map interface and the matrix interface, Eyegaze Response Interface Aid (ERICA), were used to collect the participant eye position data [3] (the functional prototype was a dual monitor application and was not supported by this data collection tool). ERICA's analysis software, GazeTracker [4], was used to determine how often participants looked at certain parts of the display and for how long. Mouse movement data were also collected.

A questionnaire (Figure 3) captured the participants' perceptions concerning the differences among interfaces. One type of questions asked his/her perception about a specific point, while other type asks him/her to rank the interfaces according a given criteria.

1) What system did you find easiest to use?
Put in order from 1 (easiest) to 3 (hardest)

Matrix interface

Map interface

Functional prototype

2) For the interface that you put "1" in the previous question, why did you think it was the easiest?

3) In what interface was the request information best displayed?

4) In what interface did you more easily find the request amount?

5) In what interface was the truck information best displayed?

6) In what interface did you more easily find the truck capacity?

7) In what interface did you find easily the good and bad routes?

8) Write down at least one good thing about each interface
(Matrix interface, Map Interface, Functional Prototype)

9) Write down at least one bad thing about each interface
(Matrix interface, Map Interface, Functional Prototype)

10) What interface distributes the information best?
Put in order from 1 (best distributed) to 3 (worst distributed)

Matrix interface

Map interface

Functional prototype

11) What would you change in the Matrix?

12) What would you change in the Map?

13) What would you change in the Functional Prototype?

Figure 3 – Questionnaire

The usability test included the following independent variables: (a) tested interfaces (matrix, map and functional prototype); (b) scenarios and (c) order that the interfaces were presented to the users (map first or matrix first) (Table 1).

The dependent variables included the information from ERICA and from the questionnaire. From the questionnaire, the results from the ranking questions (questions 1 and 10 in Figure 3) were obtained from the sum of each index rank, where 1 was assigned to the easiest or better interface and 3 to the hardest or worst one.

Table 1 – Test order description

	1	Participant 2	Participant 3	Participant 4	Participant 5	Participant 6
First tested Interface	Map	Matrix	Map	Matrix	Map	Matrix
Tested Second Interface	Matrix	Map	Matrix	Map	Matrix	Map
Third tested Interface	Func. prototype	Func. prototype	Func. prototype	Func. prototype	Func. prototype	Func. prototype

B. Results

All the six participants ranked the functional prototype as the easiest one to program (question 1). Five out of six participants ranked the matrix as the most difficult interface. The participants stated in question 2 that the functional prototype visualized information in different ways (map and table of requests). They also preferred the option to begin the programming process by both paths (the truck or the location).

Five participants preferred the functional prototype design for request information (question 3). Four participants also preferred the functional prototype for question 4, concerning request information. The customer information was considered best displayed in the functional prototype.

Concerning the truck information (question 5) and truck capacity display (question 6), the majority of participants indicated the functional prototype as the best. The map and the matrix interface were also cited once each for the truck information, but not for the truck capacity.

The participants listed good points about the map interface related to the routes and distance visualization and the system simplicity. The good points noted about the matrix interface regarded the visualization of driver availability: it is easy to see which requests only one driver could fill and to find the drivers available for each client. The possibility to visualize all the requests in the same row was positively regarded. Participants also preferred to program by the truck or by the location, a feature cited also for the functional prototype. With respect to the functional prototype, the good points concerned how well the information is displayed and how the buttons can be selected dynamically.

The most cited bad point about the map interface was that it is difficult to visualize the request amount. In addition, the user cannot see the good and bad routes. Also there is just one possible way to start the programming (request-first). The main weakness of the matrix interface was clutter: too much information displayed at once, making it difficult for novices to use.

With respect to the functional prototype, participants cited the apparent random distribution of information on the screen (its “ugly” appearance). In addition, in relation to the selection of the second request for a route, the system does not show the second best location once the first request is selected.

When asked to rank the interfaces that best displayed information, four participants agreed that the matrix interface is the best (even though this interface was considered cluttered).

Participants made several suggestions. For the map interface, the suggestions concerned improving the route information by adding the names of the cities on the map, the distance among the cities, the route quality, and the truck information. For the matrix interface, the suggestions involved the quantity of information displayed: to add a filter to choose which information to visualize and a legend for the meaning of the colors. Other points are related with the quality of displayed information: improve routes and client information, especially route rates. The suggestions for the functional prototype concerned the selection of the second location in a route construction and an improvement in the truck information display.

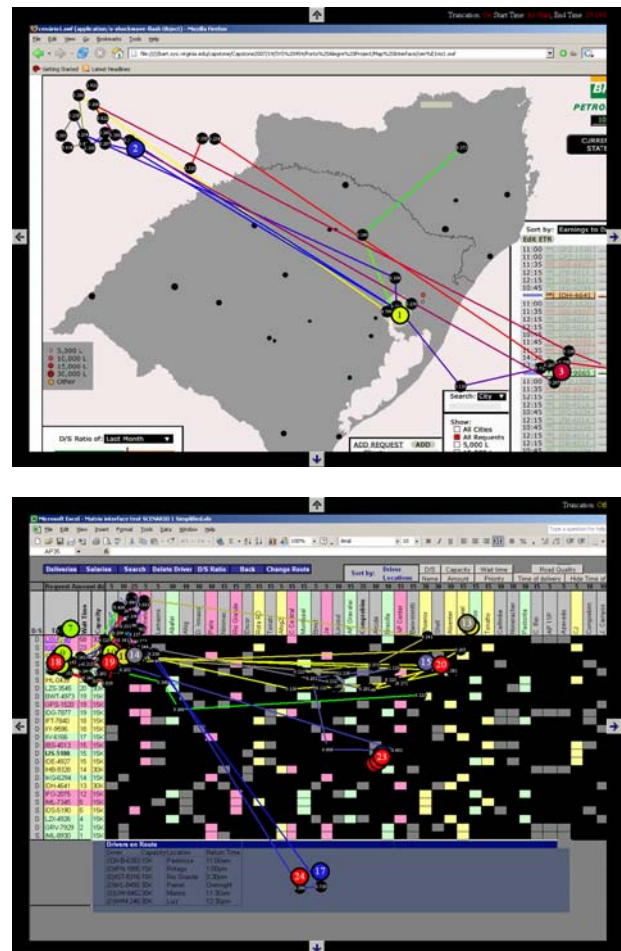


Figure 4 – Example of GazeTracker results for the map (top) and matrix (bottom)

For the matrix interface, the ERICA data (see bottom of Figure 4) indicated that much of the screen that was not used: 78% of the gaze frequency is concentrated in only two zones. These zones contained information about trucks and requests as well as where the “confirm” popups were placed. Not surprisingly, areas with frequent mouse clicks were viewed more often. Thus the matrix interface covers a lot of screen real estate but only a fraction is used.

The generated data for the map interface shows that 80% of the gaze frequency is concentrated on 8 different zones on the screen, especially those that include the route and truck information (Figure 5). In contrast with the matrix interface, the information is more spread on screen.

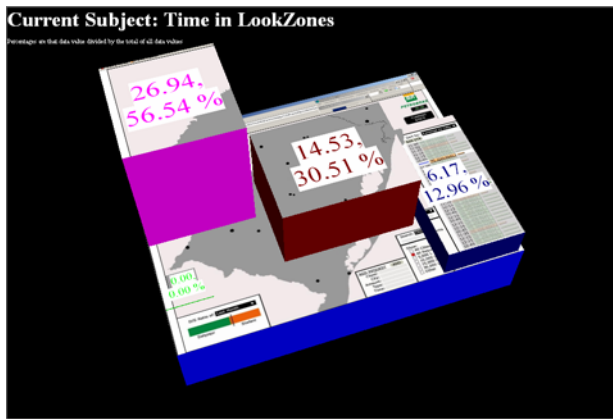


Figure 5 – Map interface gaze percentages

V. CONCLUSION

Prior effort proposed two human-computer interfaces to aid the dispatchers. The requirements were re-evaluated and a functioning prototype was designed. Usability tests were conducted with the two prior interfaces and the new functional prototype. The results indicate that an interface that combines different options of information display may be a good path to follow. The possibility of switching between both types of display makes the system easy to adapt to users with different levels of knowledge of the task. The combined interface is a good resource for users with different levels of knowledge about routes and the programming task and will be further improved if a 2-screen system is used, such that both screens can be seen/used simultaneously. Although the current dispatchers are experienced users, this task is being centralized in Rio de Janeiro, and thus having the map will be helpful for these new dispatchers who are not local to the area.

The results must be interpreted with caution, due to inherent biases in the testing methodology. Only the functional prototype was dynamic while the map and matrix interfaces that were tested consisted only of screenshots that simulated how the software could work. This method of testing could have made the functional prototype seem more flexible as compared to the other interfaces and, in this way, easier. In addition, the

functional prototype was always the last interface to be tested, because of the fact that it mix features from the both other interfaces. When the user tested the last interface, he/she was more used to the programming task and the test scenarios, thus potentially making the combination seem easier. Although the matrix interface seemed cluttered, it was ranked as having the best distribution of information by 4/6 subjects. Further testing will need to be done to tease out these different effects by making all of the alternate interfaces fully functional to enable running a more complete and balanced experimental design.

The simulation system developed here, with the ability to automatically create scenarios will now be able to be expanded upon and used to test these and similar concepts with the new centralized dispatching system in Rio de Janeiro. Such a centralized dispatching center will need to accommodate not just the southern two states of Brazil served by one distribution center, but to accommodate each of the different distribution centers throughout Brazil, likely with local requirements for each. This will serve as the next phase of our project.

ACKNOWLEDGMENT

The authors thank the manager, dispatchers, and truck drivers who provided project input and Lucimara Ballardin from Universidade Federal do Rio Grande do Sul who helped with the initial research in Brazil and with documentation.

REFERENCES

- [1] S. Guerlain, C. Pratsch, C. Cutrona, A. Welter, C. Valcarengh, L. Guimarães, “Resource Allocation Interface Design for an Oil Distribution Plant,” in *IEEE Systems and Information Engineering Design Symposium*, University of Virginia, Charlottesville, VA, 2006.
- [2] S. Lauesen, *User Interface Design: A Software Engineering Perspective*. Addison Wesley, 2005.
- [3] T.E. Hutchinson, K.P. White, Jr., W.N. Martin, K.C. Reichert, L.A. Frey, “Human- Computer Interaction Using Eye-Gaze Input,” *IEEE Transaction on Systems, Man, and Cybernetics*, 19(6), 1989, 1527-1534.
- [4] Eye Response Technologies (2006). Available: <http://www.eyeresponse.com/Analysis/>