2014

Data Exploration via Mind Maps in the space of Supply Chain Networks

Robert Hasty
Grand Valley State University

Follow this and additional works at: http://scholarworks.gvsu.edu/cistechlib

Recommended Citation

This Project is brought to you for free and open access by the School of Computing and Information Systems at ScholarWorks@GVSU. It has been accepted for inclusion in Technical Library by an authorized administrator of ScholarWorks@GVSU. For more information, please contact scholarworks@gvsu.edu.
Data Exploration
via
Mind Maps
in the space of
Supply Chain Networks
By
Josh Hasty
April, 2014
Data Exploration
via
Mind Maps
in the space of
Supply Chain Networks
By
Josh Hasty

A project submitted in partial fulfillment of the requirements for the degree of
Master of Science in
Computer Information Systems

at
Grand Valley State University
April, 2014
# Table of Contents

Abstract .........................................................................................................................................................4  
Introduction ...............................................................................................................................................4  
Background and Related Work ....................................................................................................................5  
Program Requirements .................................................................................................................................6  
Implementation ...........................................................................................................................................7  
Results, Evaluation, and Reflection .............................................................................................................8  
Conclusions and Future Work ......................................................................................................................9  
Bibliography ..............................................................................................................................................12  
Appendices ..................................................................................................................................................13
Abstract

Corporations associated with the supply chain industry depend heavily upon business intelligence to transform raw data into core metrics and key performance indicators. Because these types of measurements are critical in supporting strategic, day-to-day business decisions, both visibility and accessibility are of equal importance in order to facilitate effective management of transportation activity within the supply chain. Often times, analysts engaged in this industry have a need to regularly review these metrics not only individually but also simultaneously - and at various different hierarchical levels. It is through this type of analysis that analysts can begin to identify potential cause and effect relationships between metrics and take action in order to correct the issues that these relationships reveal. The purpose of this project is to create an application that transforms typical sets of transportation supply chain key performance indicators into an interactive mind map and explore the potential benefits of this result. The goal is to be able to use this output as a vehicle to quickly identify transportation related issues within the supply chain and expose relationships that are occasionally lost or difficult to identify within large analytical spreadsheets or amongst a group of individualized, detached reports.

Introduction

This project is about data exploration within the transportation space of the supply chain. At first glance, it is easy to underestimate the world of transportation. How challenging can it be to move products from point A to point B? But the reality is that transportation within the supply chain industry is anything but simple. There are many different methodologies used to transport goods and all of them can be quite costly to the companies needing to distribute their products which makes it particularly important for them to monitor those costs as closely as possible. Despite the fact that transportation rates can vary significantly based on factors such as geography, transportation demand, seasonality of products, and the needs of the transportation provider; transportation costs can still be reasonably controlled with proper visibility to the conditions within a company’s transportation network. This is often facilitated through access to and monitoring of core metrics and key performance indicators that illustrate how a company’s network is performing. The story that these measurements tell drive day-to-day business decisions enabling a shipper in controlling the fluctuation of transportation costs and maintaining a high level of service to their customers.

The primary goal of this project was to simplify the process of identifying transportation related issues (and their potential causes) by creating a business intelligence application that empowers the user through data exploration via mind maps. Using mind maps, the user should be able to easily navigate common data points at the various necessary levels in order to support decision making in a more efficient and effective manner. The motivation for this idea originated from the challenges that I have faced in my
recent experience as a report developer for a transportation management software company. In that experience, customer requests for reporting often require large spreadsheets full of aggregated data representing anywhere from 20-50 metrics, broken out at various different levels, and over a period of many weeks or months. (An example of this type of report is available via Appendix A.) The information is usually more than a user can consume in a single setting, and often requires additional analysis and manipulation to be performed in order determine what issues are prevalent within their network. Once the data is analyzed, there is often a follow up request for supporting data in order to determine the root cause of the issue(s) identified, which may also require additional manipulation and processing. This project is important because it can help minimize this extra work and provide the user with a more visual experience as they traverse points of interest within their data. The result of this project is something that I plan to present to my employer in the interest of utilizing as an internal tool within our organization and perhaps even implement into our company’s transportation management software as an add-on to the business intelligence module being operated by our customers today.

**Background and Related Work**

The idea for the application that I have designed, entitled “Supply Chain Metric Map”, was inspired by an application called TuneGlue. TuneGlue is a music relationship explorer that offers a unique take on a music recommendation system by displaying artist recommendations via an interactive mind map using nodes and connectors. (Screenshots of TuneGlue are available via Appendix B.) The application starts by prompting the user to enter an artist’s name which will become the focus of the application at the start. Then, the user can right click and select ‘Expand’ on the artist’s name thereby expanding the map by displaying other similar artist’s names around the original entry. This process can be repeated multiple times resulting in a large visual web of related artists that the user can investigate in order to discover new music. The experience with the primary functionality of TuneGlue was positive as it allowed me to navigate through related artists with ease and efficiency. Unfortunately, many of the other features appeared to be incomplete, such as the option to expand the release list of an artist, as they were not working properly. Still, after my experimentation with TuneGlue, I was left wondering if this same concept might be applicable to the supply chain industry. More specifically, to facilitate navigation through supply chain network information with the intent of identifying problematic issues within a transportation network.

During my research, I found it challenging to locate examples of applications that utilize mind maps within the supply chain industry, or any other industry, as a business intelligence tool in the same capacity employed in this project. Many of the mind maps that I did find relating to the supply chain industry were static concept maps designed to display relationships between various entities within the supply chain rather than ones that displayed actual measurements leading me to believe that the idea for the Supply Chain Metric Map may have potential as a unique solution for the business problem that this project is attempting to solve. (Examples of the static concept maps are available for reference via Appendix C.)
The most important discovery that I made during my research is regarding an application called SpicyNodes. SpicyNodes uses a tree structure, equivalent to that of a mind map, to organize information in a logical and often hierarchical manner. This type of visual display is very similar to what I had envisioned for the Supply Chain Metric Map. Thus, I was prompted to pursue further investigate what other types of domains have used SpicyNodes, in the interest of comparison and to confirm its place in within my project. (Screenshots of SpicyNodes examples are available via Appendix D). What I liked the best about SpicyNodes is that it provided the option to load data via an XML file, meaning that the mind map could change dynamically based on the contents of the file. This was key to the vision of my project was the determining factor for implementation into the “Supply Chain Metric Map”. Unfortunately, after reviewing the examples, it seemed that these too were all projects that were built off of static data with no need for the maps to change dynamically based on interaction with the user. However, the examples they provided did trigger ideas in terms of how to style nodes and organize information as well as provided me an overall feel for how to navigate mind map created by the SpicyNodes application. Based on these findings, by the time I completed my research I felt confident that SpicyNodes was the right technology for the presentation layer this project.

Program Requirements

In order to construct a prototype of the Supply Chain Metric Map, I called upon my ten years of experience within the transportation industry and shared the idea with a few of my co-workers to solicit their feedback. What transpired, was a plan for an ad-hoc type reporting tool to harvest common supply chain core metrics and key performance indicators (KPI’s) and display them in the form of an interactive mind map. After the last three months of development, the prototype I was able to create performs this task successfully allowing the user to traverse their network information in an original and unique way.

The application begins with a simple parameter page, prompting the user for selections that control the assembly of the map and the organization of the data displayed within it. (A detailed summary of available parameters is provided in Appendix G.) After the selections have been made, the user clicks ‘Generate Metric Map’ to create the map. Each map produced is made up of nodes and connectors that link related information in a hierarchical fashion beginning with a single node located at the center of the map. This node is called the “Center”, which is the core from which the information originates. The metrics selected are displayed at their highest aggregate level around the center providing the user with a quick visual of how the network is performing in those categories. The user can then click on any of these metrics to drill into the subsequent levels of information and obtain the metric values for the entities within the selected groups. To aid with the identification of issues, I also included a way for the user to set thresholds and based conditional formatting off of those thresholds in order to drive color changes within the map – “red” if the a threshold has been violated, and “green” if the metric is compliant. (Screenshots of navigation through the “Supply Chain Metric Map” are available via Appendix H).
Implementation

The primary technologies used to create this implementation of the Supply Chain Metric Map include:

- Oracle XE database
- SQL (via SQL Developer)
- Java (via Eclipse Kepler)
- Jboss 7.1 Runtime Server
- XML
- HTML
- SpicyNodes
- JavaScript, CSS, and UNIX

At the data layer, the Oracle XE database consists of five tables: LOAD, STOP, TENDER, ACCESSORIAL and METRIC. The first four contain all of the data elements necessary to calculate the metrics. The data contained in these four tables are representative of a three month sample from the datamart tables that reside in the transportation management system at my workplace. I used this data so that I would be able to demonstrate real world examples and manipulated the necessary information to protect any confidential information. The METRIC table contains the details of each metric including: name, description of the calculation, threshold, and the operator (greater than, less than) to indicate what constitutes a threshold violation. Built on top of these four tables are four views: LOAD_VIEW, STOP_VIEW, TENDER_VIEW, and ACCESSORIAL_VIEW. Each view contains an aggregate version of the table specified and includes a join back to the LOAD table which is the primary source of the fields needed for filtering. This is done to ensure that as data is filtered, it is consistently of the same set of loads across the four views. These aggregate views simply sum information which is what provides this application with the flexibility to traverse the various hierarchical levels. Any average or percentage calculations that need to be calculated are delayed until the filters and groups are selected so that they can all be applied to the queries simultaneously at run time. This is done to ensure that the calculations for each group are correct and to prevent the likelihood of summing/averaging percentages/averages which would result in incorrect information.

Above the data layer of the Supply Chain Metric Map is the application layer which is written in Java. The Java code is made up of 2 classes (Metric and Indicator), an HTML page to prompt the user for parameters, and a Java servlet to perform the parameter handling and application processing. Once the parameter selections have been made and the user clicks the ‘Generate Metric Map’ button, the servlet begins looping through each of the selected metrics and dynamically constructs the necessary queries based on the user’s selections. During this process, the application also needs to convert the results of each query into XML format in order for the information to be picked up by the presentation layer. DocumentBuilderFactory is the Java class that I used to build the XML document as it provided the necessary flexibility in order to position the elements and attributes within the XML file before passing it to SpicyNodes to be displayed as a mind map.
SpicyNodes is the technology used for the presentation layer of this project. In order to link the Java portion of the Supply Chain Metric Map to SpicyNodes, I needed to introduce the “SpicyConnector” into my project. The purpose of the SpicyConnector is to allow a SpicyNodes tree to be embedded within an HTML page. Since the nature of this project is to generate data dynamically based on user selections (rather than simply embed pre-existing SpicyNodes maps), this was my only option to get SpicyNodes to interact with Java. Within the embedding block for the SpicyConnector, I was able to set a path for the XML file produced by the Java code to be picked up and processed by SpicyNodes. This was the key to producing the interactive mind maps that I was looking to accomplish with this project. To top off the construction of the project, I used some additional HTML to display the thresholds stored in the METRIC table. This provides the user with a point of reference for the violated thresholds displayed within the map.

I chose to use the technologies specified earlier because this project is closely related to software developed at my place of employment. I wanted to keep the technologies similar enough so that if we were to want to pursue this application further at the end of the semester, we could easily do so. The technologies that I have a significant amount of experience with are Oracle and SQL. I have also worked with transportation data in a supply chain network frequently so knowing how to arrange the data was also one of my stronger areas of expertise. However, I have only had academic experiences with Java and XML so learning how to get Java to produce the XML file necessary to power the SpicyNodes presentation was challenging because I had to coordinate this effort with the dynamic creation of each of the queries in order for the XML layout to be constructed properly. And SpicyNodes was completely new to me at the beginning of this project. I spent a significant amount of time reading through the developer documentation in order to learn how to use the application, as well as integrate it with other technologies.

During the planning stages of this project, there were a few alternatives considered for the implementation. One of which was to use Visual Basic Studio and an SQL Server database to power the Supply Chain Metric Map application rather than the current Oracle XE and Java combination. This was quickly ruled out however since I wanted to keep this in line with the technologies being utilized at my workplace. Another alternative was to solely use JavaScript to interact with SpicyNodes instead of Java. But because I’ve had some previous academic experience coding with Java and XML, I thought that would be the most efficient route for me to take so that I would have more time to do as much as possible with the application. Regarding the mind maps, there were a few other mind mapping applications that were researched such as PearlTrees and MindMeister but neither of which provided the interactive experience and dynamic data capabilities for which I was looking - and found within SpicyNodes.

Results, Evaluation, and Reflection

On a personal level, I am very happy with the overall results of this project. When I first started development, I was apprehensive about the unknowns that I would encounter and concerned about running into roadblocks, particularly within the Java layer of the application. Because of this concern, I used an iterative and incremental development approach while working on this project. This process allowed me to
work through the items that I did not know how to do early on in the project so that I could create a very small scale model with the first iteration, and then build it out over the course of a second and third iteration of development. The main items that concerned me were converting results of database query to XML using Java, setting up the Java servlet to prompt and pass parameters into the application, and passing the XML file to SpicyNodes in order to dynamically generate a mind map. None of these items turned out to be a problem in the end, but were concerns due to my limited experience and the fact that they were all of vital importance to the project.

One challenge that I was not able to overcome was in regards to SpicyNodes. While the node map functionality of SpicyNodes works very well, I was unable to get some of the HTML elements that are supposedly supported to work with the SpicyConnector. All of the data was transferred as expected, but there were claims of functionality that allows you to embed HTML tags within a node which would have provided me with more options regarding the display of data within a node. I was able to get some of the supported HTML elements to work when creating a node map from directly within the API, but they were still inconsistent at best and not particularly useful to this project to be limited to the website’s API. The solution I finally settled with was to simply separate data elements within a single node with a tilde (~) to mark the separation of each item.

However, once I had reached a point where I could demo the project for my co-workers, I scheduled a second meeting with them to solicit feedback on the prototype. I did this as an exercise to help me validate the successfulness of the project and remarkably their comments were all very positive. In spite of some of the limitations I encountered with SpicyNodes, the demonstration revealed that the prototype accomplishes the goals that it was designed to accomplish as well as provides a new unique way of reviewing and analyzing transportation supply chain information. I am looking forward to pursuing this project further in hopes that we can implement it, or some derivative of it, into our application at my workplace.

Conclusions and Future Work

Even though the project went well, there are a number of items that I would like to address in future versions. The following list reflects those items that were attempted but fell short in terms of execution due to lack of time to research and/or develop:

**Bug - Single shipper limitation.** Currently, this project is using a 3 month sample set of data from a single shipper. If I were to load another shipper’s information into the database, the queries would likely return both data from both shippers. During development, I neglected to limit the queries to only the selected shipper and since there was only one shipper in the data, this was an item that was quickly taken for granted and easily overlooked but should also be easily corrected.

**Bug – Percent metrics regarding detention and lumper are not calculating correctly.** The dynamic generation of queries based on a single string can be very challenging. In order to calculate the detention and lumper totals correctly, I need to be able to pull the correct overall total cost for every entity
within each level. Unfortunately, I ran out of time at the end to do this properly so I had to remove these
calculations from the application. (Although I did leave them visible on the parameter page to fill it out a
little.)

**Refactor – Refactor java code to be more efficient and secure.** Before I go too much further
with this project, I would want a Java expert to review the code. Since I personally have not used Java a
significant amount, I would expect that there are many opportunities for improvement in both the areas of
performance and security. Especially if this application was to be utilized with our customer base, I would
want to convert the query strings that are being built to something that is more encrypted to reduce the
possibility of SQL injection.

**Enhancement – Opportunities for visual enhancement with styling.** There were several items
that I wanted to address with styling, some of which I attempted to implement with little success and others
I ran out of time to research and address. First, I would have liked to have more control over how each of
the nodes themselves were displayed. SpicyNodes provided a few different options to allow for this, but
any HTML that I included in the XML did not get processed at run time. And second, there is a
settings.xml file that SpicyNodes allows the user to create in order to better control the look and feel of the
node map. I could not find a sample of this file as a whole, and my one attempt to piece a file together
from the code provided within the developer’s reference of SpicyNodes was ineffective. While neither of
these were vital to the success of my project, both would have been nice to have in terms of visual effect.

The remaining items on this list are enhancements that were not attempted and were outside the
scope of the original project idea for the Supply Chain Metric Map application, but that I would like to
pursue in future iterations of this project:

**Enhancement – Add more selections to the ‘Center’ parameter.** Currently there are only two
active options for the ‘Center’ parameter of the application: shipper and carrier. I would also like to
incorporate options such as Location and Customer. These are two other key entities within the supply
chain that are necessary for analysts to report upon and monitor. To take this option a step further, giving
the user the option to group any of these entities (shipper, carrier, location, or customer) and assign a label
could also be useful. For example, let’s say a shipper is using 5 specific carriers to shuttle products a short
distance between two of the company’s facilities several times a day. An analyst may have reason to get a
better understanding of how these ‘shuttle’ carriers are performing as a whole. This option to group entities
allows the user to view the metrics as they relate to the overall aggregate group. This type of concept is
common in the industry and would add an additional layer of distinction to this project.

**Enhancement – Add links to retrieve additional information.** With regards to this
enhancement, it is important to prevent the project from becoming overly busy. That said, there are some
arguments that could be made for providing links to additional information from within the map. First, an
export of the data supporting the metric from any node would be necessary. Users reviewing the report are
often looking to this supporting data for validation and to help track down the root cause of an issue.
Second, a possible link to some related trending information in the form of line graphs may also be useful.
While the mind map can show weekly and monthly trends via nodes, it is often helpful to see these trends in the form of a graph to get a better visual what has been happening over a given period of time. This could also allow for viewing of multiple trending intervals displayed at the same time, rather than being limited a single option preselected from the parameter page.

In conclusion, this project has been a great learning experience. I was able to take a problem from within the transportation supply chain industry that we have been facing for the last couple of years and architect a prototype that can help us overcome that problem in a way that shows promise for future growth and development. The mixture of technologies was appropriate as was my level experience with each of them. This project has provided me with the opportunity to learn something completely new, SpicyNodes; increased confidence in a language that has often intimidated me in the past, Java; and a chance to put to good use the skills which I have been refining during my recent work experiences, SQL and Oracle. While this project was certainly challenging for me, I did not find it to be overwhelming and I’m looking forward to seeing what transpires from it in the future.
Bibliography
Appendices

Appendix A: Screenshots of large spreadsheets full of aggregated data that are commonly requested in the transportation supply chain industry.
Appendix B: Screenshots of TuneGlue (the application that inspired the “Supply Chain Metric Map”.

1. Begin by searching for a musical artist.

2. Click on the artist to reveal selections, and click ‘expand’.

3. Review the related artists revealed, and click on another artist to expand the list further.

4. Once expanded, the artists related to ‘Ringo Starr’ are revealed.
Appendix C: Examples of supply chain related mind maps. These are a few of the many examples I found that reflect entity relationships.

1. Supply Chain Management uploaded by wemindmap technology unknown.  
   <http://www.biggerplate.com/mindmaps/WpkJuEeqq/supply-chain-management>

2. Supply Chain Management by Uriel Buitrón using MindMeister.  
Appendix D: Examples of other SpicyNodes creations.

1. Exploring calendars through the ages.
   <http://www.spicynodes.org/a/fd15ee43638285acdad08ae5cc917755>.

2. SpicyNodes mash up with Google Maps.
Appendix E: Transportation related terms

**Shipper** – A company who has a product that needs to be moved.

**Carrier** – A trucking company who has trucks that can move products.

**Mode** – The transportation mode by which the product is moving.

**Truckload (TL)** – Full truckload – 53’ trailer - 26 pallets.

**Less Than Truckload (LTL)** – Not a full truckload – usually somewhere between 1 and 12 pallets.

**Intermodal (IM)** – Trailer or container moved by combination of truck and train.

**Equipment** – The type of equipment required to move the product.

**Van** – Standard semi-truck box trailer used to move products that require no temperature control.

**Ref** – Refrigerated semi-truck trailer used to move things that require temperature control.

**Flat** – Semi-truck trailer with no walls used to move things that do not require protection from weather.

**Load** – The movement freight from point A to point B.

**Stop** – Any location associated with a load at which a carrier stops to pick up or deliver freight.

**Tender** – Agreement between a shipper and a carrier contracting a carrier to move a load for the shipper. Contains information such as pick up location(s), delivery location(s) and rate.

**Accessorial Charge** – Any cost incurred during the transport of freight above and beyond the cost of the movement itself.

**Geography** – Geographical location associated with a load or stop.

**Lane** – Geography of the first pick up and last delivery of a load.

**Planner** – Person responsible for planning loads.

**Lumper** – Service fee for the unloading of a truck.

**Detention** – Fee for retaining a truck for longer than 2 hours at a pickup or delivery location.
Appendix F: Transportation Timeline

- Shippers create orders
- Orders made up of a list of products and geography
  - Order geography is made up of an origin and destination

- Planners plan orders into loads
  - Loads made up of geography, mode, and required equipment
- Load geography made up of origin, destination and sometimes intermediate stops.
- Mode can be either truckload (TL), less than truckload (LTL), or intermodal (IM)
- Equipment can be either Van (Dry), Ref (Refrigerated) or Flat (Flatbed)
- Planners tender loads to carriers

- Carriers respond to tenders with a response of ‘accept’ or ‘reject’
- Carriers transport loads
- Carriers charge the shipper the following fees
  - Rate (for the move itself)
  - Fuel Surcharge (for the fuel used to move the load)
  - Accessorial Charges (any additional charges incurred during transport)
Appendix G: Parameter Page for Supply Chain Metric Map

Supply Chain Metric Map

Date Range:
Start Date:  [ ]  End Date:  [ ]

Center:
- [ ] Shipper:  [ ]
- [ ] Carrier:  [ ]
- [ ] Customer:  [ ]
- [ ] Location:  [ ]

Levels: Groups:
1. [ ] Mode:  [ ]
2. [ ] Mode:  [ ]
3. [ ] Mode:  [ ]

Filters:
- [ ] --None--:  [ ]
- [ ] --None--:  [ ]
- [ ] --None--:  [ ]

Sorts:
- [ ] Total:  [ ]
- [ ] Total:  [ ]
- [ ] Total:  [ ]

Metrics:
- [ ] On Time Performance:  [ ]
- [ ] Accessorial Percent of Total:  [ ]
- [ ] Total Loads:  [ ]
- [ ] On Time Pickup:  [ ]
- [ ] Tender Percent Accepted:  [ ]
- [ ] Total Spend:  [ ]
- [ ] On Time Delivery:  [ ]
- [ ] Percent Stops w/ Lumper:  [ ]
- [ ] Total Lumper Charges:  [ ]
- [ ] Percent Multistop:  [ ]
- [ ] Percent Stops w/ Detention:  [ ]
- [ ] Total Detention Charges:  [ ]

Generate Metric Map:

Date Range: Dates to filter selected loads based on pickup date.
Center: The source from which the all information in the metric map is based. Check the box next to the desired entity and use the combo box to select value for that entity.
Levels: Susequent levels of the metric map that expand from the center. The aggregate of the metric is always level 0. Each additional level can be turn on or off by checking the check box.
Groups: The focus of the selected level. Will divided into as many parts available within the data. (i.e. Equipment = Van, Ref, Flat).
Filters: Preset filters to help limit the amount of data being returned. Used for Groups with large number of members. (i.e. Carrier, Geography).
Sorts: Preset sorts to help organize the data within the map. Total is used to sort by the highest number of records measured. Metic is used to sort by the calculated metric value. Group is used to sort by the value of the selected group (i.e. Month, Week).
Metric: Available metrics to be viewed on the metric map.
Generate Metric Map: Button that initiates creation of the map based on selected parameters.
Appendix H: Screenshots for Supply Chain Metric Map

When run with the following parameters:

- **Start Date:** 01/01/2013, **End Date:** 03/31/2013,
- **Center:** Shipper – Hasty Shipper
- **Level:** 1, **Groups:** Mode, **Filters:** None, **Sorts:** Total
- **Level:** 2, **Groups:** Equipment, **Filters:** None, **Sorts:** Total
- **Level:** 3, **Groups:** Carrier, **Filters:** Top 10, **Sorts:** Total
- **Metrics:** On Time Performance, On Time Pickup, On Time Delivery

Produces the following metric map:
Click on “On Time Pickup” to expand the map into the next level, where metrics are divided up by mode:

Click on “On Time Pickup – TL” to expand the map into the next level, where metrics are divided up by equipment:
Click on “On Time Pickup – VAN” to expand the map into the next level, where metrics are divided up by carrier:

Change out the parameters in order to explore data throughout the network.
Appendix I: Calculations used in Project

Total Loads = Total Number of Loads
Total Cost = Total Cost of Total Loads
On-Time Performance * = Total On-Time Stops / Total Measurable Stops
On-Time Pickup * = Total On-Time Pickups / Total Measurable Pickups
On-Time Delivery * = Total On-Time Deliveries / Total Measurable Deliveries
Percent Multi-stop Loads = Total Multi-stop Loads / Total Loads
Accessorial Percent of Total Cost = Total Accessorial Cost / Total Cost
Percent Tender Acceptance = Total Accepted Tenders / Total Tenders
Total Detention = Total Cost of Detention Incurred
Total Lumper = Total Cost of Lumper Incurred

* On-Time is calculated as being within 12 hours of the appointment time.