**Selecting the Challenge**

The Vehicle Survey Code Challenge was selected primarily as it demonstrates a number of important data manipulation techniques as well as basic and input and output processing requirements while not requiring any external resources. Additionally it lends itself particularly to TDD as each requirement statement can be stated as one or more test cases. This allows for safe refactoring as the data model develops during coding.

**Design Planning**

There are four parts to the project: Loading the test data, Processing the data into Vehicles passing the counter, Accumulating the Vehicles activity based on requirements, and Saving the results. There are some important assumptions to be made about the test data and these needed to be checked before proceeding: 1) There are no partially recorded counts, e.g. all ABAB and AA groups are complete, and 2) There are no overlapping counts, e.g. AABABA where the first and last A are for the same vehicle. An analysis of the test data confirmed that this is the case. A more robust solution would accommodate these cases either by excluding them or by attempting to re-construct the overlapping counts.

**Design Approach**

The tests included in VehicleSurveyTest.java were created to match the requirements where each test was created prior to coding the application in VehicleSurvey.java. Initially there was the need to make sure that all the test data lines were loaded correctly. This was validated by simply matching the input to the actual number lines in that file. The next step was to make sure that the correct number of vehicles could be determined as well as their direction of travel. Initially the check values were determined mathematically from the input test data. As the process of creating actual Vehicle records proceeded, these source for these values was refactored to use those counts instead of the raw data.

From there, the required reporting periods were identified by the number of rows returned for any one day’s vehicles. The code to support this translates each of the periods into the number of periods in a day and how many minutes there are in each period. If required, other periods could be added by extending this list.

Once daily counts could be determined, it was possible to write tests for summary counts across there different period breakdowns to confirm that all processing generated the same control values.

Averages were achieved by compressing all days into a single day and dividing by the total number of days processed.

**Calculations**

The requirements for values other than simple counts where based upon the information provided.

* Peak Volume times are based on the Standard Deviation of all counts reported. Any count greater than 2 standard deviations is considered to be peak.
* Speeds are calculated based on the 2.5 meter average wheel base and the number of milliseconds between successive A counts. As this works out to approximately 60 km / hour on average, this is probably correct.
* Distance between vehicles is calculated for each vehicle by the number of milliseconds between it’s first A count and the previous vehicle’s first A count taking into account the end of the day and based on the vehicles calculated speed. At 60 km/hour a minimum safe distance would be about 120 meters (2 seconds stopping time). The minimum numbers at peak times are between 136 and 300 so this seams reasonable.