Chapter 3

Proposed Solution and Analysis

3.1 High Level Architectural Diagram

In simplest terms, the system, named “Routed” (Refer Fig 3.1), which we aim to develop, is a bus scheduler to optimize the transport system currently in place. With the current scenario of crowded buses and the delay and discomfort caused by the crowd as well as the traffic, this system hopes to alleviate the aforementioned problems.

The schedule manager has to input the data records for buses i.e., the tickets sold for the previous weeks into the “Routed” system and simply wait to receive the output - an optimized bus schedule for the coming week and the number of buses per route – as the system goes about working on the input. The system essentially groups the data according to various grouping criteria and then uses the input data to predict travel patterns so as to come up with an “intelligent” solution i.e., the optimized schedule which is given to the schedule manager and a count of the number of buses along each route given to the bus regulator who can then decide which bus should make trips on which route.

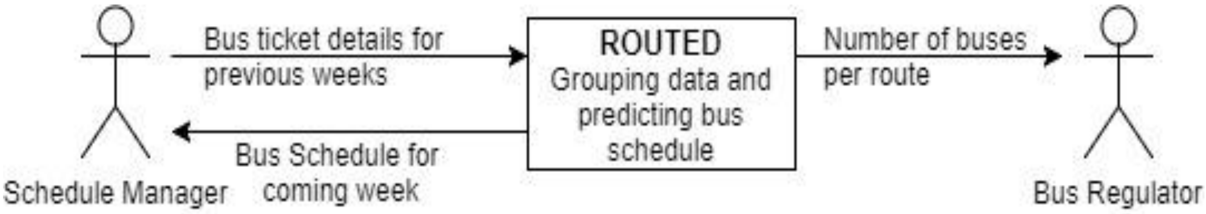
Bus ticket details for previous weeks
ROUTED
Grouping data and predicting bus schedule
Number of buses per route
Bus Schedule for coming week
Schedule Manager
Bus Regulator

Fig 3.1 High Level Architectural Diagram

3.1 Low Level Architectural Diagram

The “Routed” System is aimed at optimizing bus schedules on specific routes so as to alleviate traffic problems and the delay caused because of the same. The entire system can be broken down into multiple modules with each module responsible for certain tasks. The system is divided into the following modules (Refer Fig 3.2):

1) **<u>Data Cleansing and Validation:</u>**
Input: Bus records from previous weeks (Fed by Schedule Manager)
Output: Cleaned and validated dataset
Function: To check the input data records for attributes, data range and format of data.

2) **<u>Clustering Data Records:</u>**
Input: Cleaned Dataset
Output: Data records on each route with attribute list
Function: To group data records into clusters based on the route corresponding to that data record. For example, data clusters will be labelled as Route X, Route Y and so on with the records in that cluster being either a sub-route or the route itself.

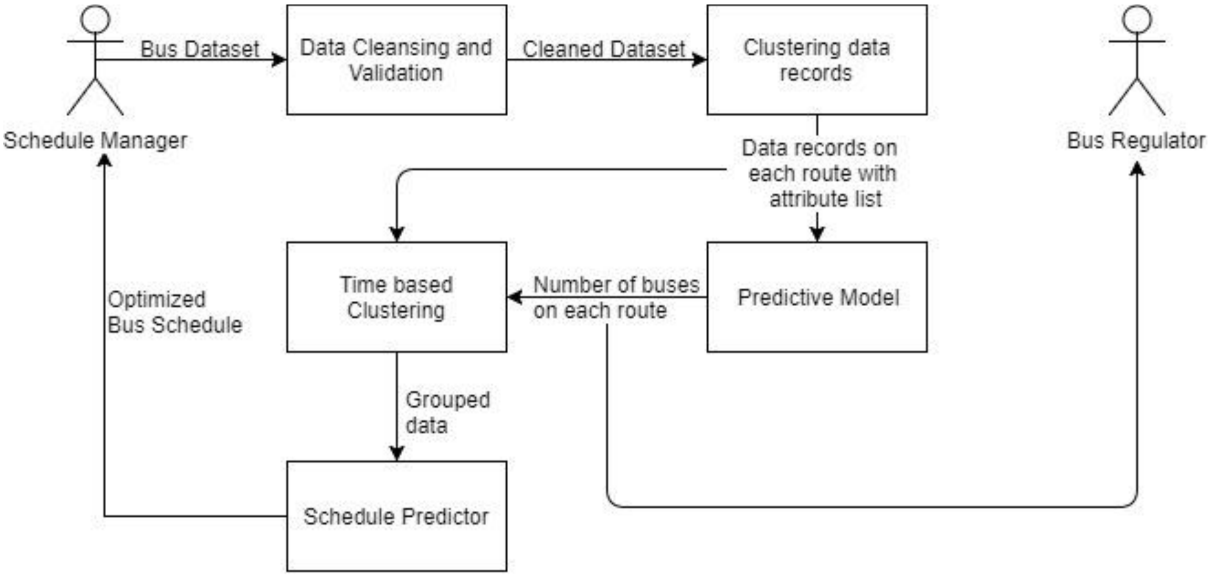

Fig 3.2 Low-Level Architectural Diagram

3) **<u>Predictive Model:</u>**
Input: Data records on each route with attribute list
Output: Number of buses on each route (Sent to Bus Regulator)
Function: To determine the number of buses on a route based on the traffic and frequency of passengers on that route. This information can be used by the Bus Regulator to decide which buses would run on which route.

4) **<u>Time based Clustering:</u>**
Input: Number of buses on each route and Data records on each route with attribute list
Output: Grouped data
Function: To further form clusters inside the existing cluster based on the timestamps to identify time slots where there is greater need for buses as compared to other time slots.

5) **<u>Schedule Predictor:</u>**
Input: Grouped data from Time based Clustering
Output: Optimized Bus Schedule (Sent to Schedule Manager)
Function: To prepare an optimized bus schedule based on the clusters within clusters generated using modules 2 and 4. The module uses information like peak time of boarding and the number of journeys on a particular route.

3.2 Use Case Diagram

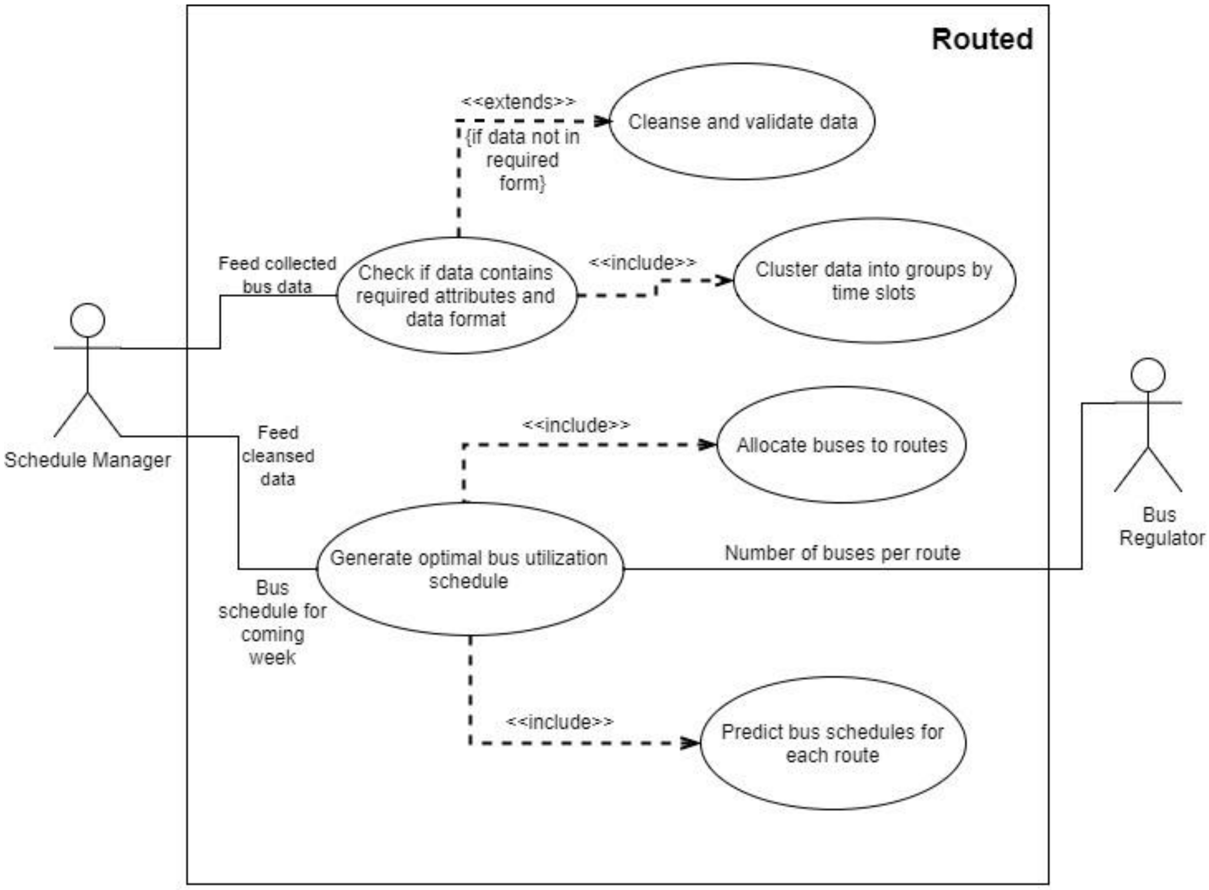
Routed
<<extends>>
Cleanse and validate data
{if data not in required form}
Feed collected bus data
Check if data contains required attributes and data format
<<include>>
Cluster data into groups by time slots
Schedule Manager
Feed cleansed data
<<include>>
Allocate buses to routes
Bus Regulator
Generate optimal bus utilization schedule
Number of buses per route
Bus schedule for coming week
<<include>>
Predict bus schedules for each route

Fig 3.3 Use Case Diagram for the "Routed" System