CS411 Red Team

Lab II

1. Specific Requirements
   1. Functional Requirements
      1. **Database** (**Odean Maye)**

The database shall contain tables that store values that correspond to each particular component of the system.

* + - 1. The database must contain a train table and it shall store the following information in fields :
         1. A train’s unique identifying number.
         2. A train’s active/inactive parameters in the form of time values.
      2. The database must contain a user table and it shall store the following information in fields:
         1. An interface user’s unique identifying number.
         2. An interface user’s username.
         3. An interface user’s password.
         4. An interface user’s access level.

The user access levels shall be defined as :

Admin level user.

Transit Authority Admin level user.

Business and Attraction level user.

Rider level user.

* + - 1. The database must contain a stops table with required fields as defined by the Google Transit Feed Specifications at <https://developers.google.com/transit/gtfs/reference#stops_fields>
         1. A stop’s unique identifier which must have the field name “stop\_id”.
         2. A stop’s name which must have the field name “stop\_name”.
         3. A stop’s latitude which must have the field name “stop\_lat”.
         4. A stop’s longitude which must have the field name “stop\_lon”.
      2. The database must contain an attractions table and it shall store the following information in fields.
         1. The attraction’s latitude.
         2. The attraction’s longitude.
         3. The attraction’s category.
         4. The attraction’s description.
         5. The attraction’s start date.
         6. The attraction’s start time.
         7. The attraction’s end date.
         8. The attraction’s end time.
         9. The attractions unique identifying number.
         10. The attraction name.
         11. The associated business id.
         12. Timestamp of record.
      3. The database must contain an adverts table and it shall store the following information in fields:
         1. The event’s latitude.
         2. The event’s longitude.
         3. The event’s category.
         4. The event’s description.
         5. The event’s start date.
         6. The event’s start time.
         7. The event’s end date.
         8. The event’s end time.
         9. The event’s image.
         10. The event’s unique identifying number.
         11. The event’s name.
         12. The associated business id.
         13. Timestamp of record.
      4. The database must contain an occupancy table and it shall store the following information in fields.
         1. The associated stop id.
         2. The number of departing riders.
         3. The number of boarding riders.
         4. The associated train id.
         5. Timestamp of record.
      5. The database must contain a GPS table and it shall store the following information in fields:
         1. The associated train id.
         2. The latitude of the train.
         3. The longitude of the train.
         4. Timestamp of record.
    1. **Decision Engine** (**Nathan Lutz**)
       1. **Ridership Trend Report**:

Provide an interface for the Web Application Engine to request a ridership report on the past.

Provide the ability to identify a date range to include:

Date(MM-DD-YYYY)

Time range (HH:MM:SS)

Provide the ability to identify a stop ID as an integer

Specified must be used to query the “Occupancy” table for number of departures and arrivals.

Provide output to Ridership Trend Report function in the form of non-negative integers.

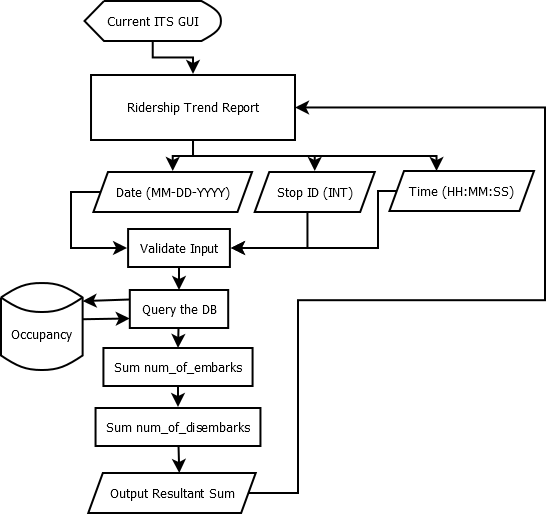


Figure 1. Ridership Trend Report Logic Flow

* + - 1. **Ridership Trend Analysis:**

Provide an interface for the Web Application Engine to request a ridership trend report on future dates.

Provide the ability to identify a date range as specified in 3.1.2.1.i

Provide the ability to identify a stop ID as specified in 3.1.2.1.ii

Specified must be used to query the Current ITS database for number of departures and arrivals from past dates during the specified time range.

Provide the ability to model the values of departures and arrivals for the future date range.

These queried values will be averaged using a “moving average” technique.

Provide output to Ridership Trend Report function in the form of non-negative integers.

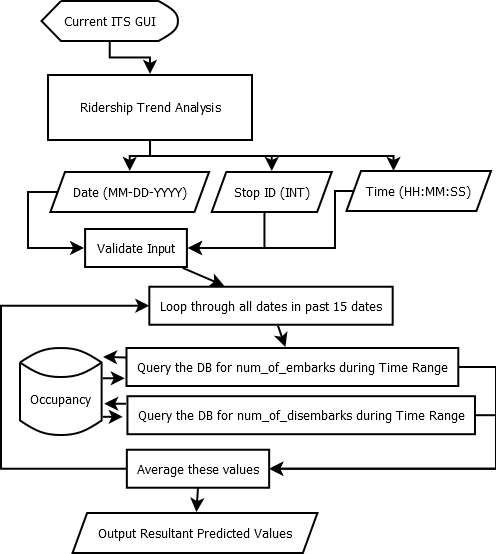


Figure . Ridership Trend Analysis

* + - 1. **Delay Impact Calculator**

Provide an interface for the Web Application Engine to request a Delay Impact report. Provide the ability to query the Current ITS database for the most recent simulated GPS location value of active trains.

Provide the ability to identify a GPS coordinate to include:

* Precede South latitudes and West longitudes with a minus sign.
* Latitudes range from -90 to 90.
* Longitudes range from -180 to 180.

Provide the ability to query the Current ITS database for past simulated arrival times at the station during the date and time range.

Provide the ability to compare those values to the HRT schedule, and return the average variance.

Provide the ability to query the Current ITS database for any active alerts and their severity level.

Specified must be used in its calculation of delay.

Provide the ability to compare the expected value of time-to-arrival from the calculated variance and current GPS position to the HRT schedule.

Provide output to (Delay Impact Module) function in the form of a time value.

* + - 1. **Ontime Performance Reporting**

Provide an interface for the Web Application Engine to request a Delay Impact report.

Provide the ability to identify a date range as specified in 3.1.2.1.i

Provide the ability to identify a stop ID as specified in 3.1.2.1.ii

Specified must be used to query the Current ITS database “GPS” and “STOPS” table for past simulated arrival times at the station.

Provide the ability to compare those values to the HRT schedule.

Provide output to Train Data Report function the average variance, in the form of a time value.

* + 1. **Test Harness (Akeem Edwards)**

The Test Harness will be used to demonstrate the current ITS prototype. This will be a standalone application that will maintain communication with the web application engine. The following functional requirements must be met:

* + - 1. **GPS Data Control**
         1. Provide the ability to access the virtual stops in the database to determine what GPS coordinates to which stops are assigned (3.1.1.4).
         2. Provide support GPS coordinate parameters:

Input parameters for single GPS coordinate are floating point values for latitude and longitude (3.1.2.2.1.1.1).

Input parameter for GPS coordinate set in an array structure to represent a virtual route.

Provide the ability to assign a GPS coordinate to each virtual train active:

Each coordinate must translate to the correct virtual route.

Each coordinate must be updated in half a minute intervals.

* + - 1. **Ridership Data control**

This software component responsible for the creation and management of virtual riders’ entities is the Ridership data control.

* + - * 1. Provide the ability to generate virtual riders at each stop.
        2. Must utilize mathematical probability distributions to define:

The amount of arrivals on each train.

The amount of departures on each train.

Must utilize probability distributions to estimate amount of virtual riders generated on days with events.

* + - 1. **Train control** 
         1. Each virtual train must provide the location coordinate abilities defined:

Return current GPS coordinate.

Return a GPS coordinate not associated with the virtual route to simulate sensor failure.

Provide the option to not return the current GPS coordinate assigned to simulate train outage.

Each virtual train must provide the ability to return amount of riders on board(3.1.1.1.4)

* + - 1. **GUI**
         1. Provide a GUI with the ability to view each virtual trains properties as defined:

The current amount of riders aboard (3.1.3.3.2)

The current location assigned(3.1.3.3.1)

* + - * 1. Provide a GUI with the ability to change each virtual train properties as defined:

The current amount of riders onboard

The location assigned (3.1.3.3.1)

* + - * 1. Provide a GUI with the ability to change ridership data at each stop.

list structure to represent a virtual route.

Provide the ability to assign a GPS coordinate to each virtual train active:

Each coordinate must translate to the correct virtual route.

Each coordinate must be updated in half a minute intervals.

* + - 1. **Ridership Data control**

This software component responsible for the creation and management of virtual riders’ entities is the Ridership data control.

* + - * 1. Provide the ability to generate virtual riders at each stop.
        2. Must utilize mathematical probability distributions to define:

The amount of arrivals on each train.

The amount of departures on each train.

Must utilize probability distributions to estimate amount of virtual riders generated on days with events.

* + - 1. **Train control** 
         1. Each virtual train must provide the location coordinate abilities defined:

Return current GPS coordinate.

Return a GPS coordinate not associated with the virtual route to simulate sensor failure.

Provide the option to not return the current GPS coordinate assigned to simulate train outage.

Each virtual train must provide the ability to return amount of riders on board(3.1.1.1.4)

* + - 1. **GUI**
         1. Provide a GUI with the ability to view each virtual trains properties as defined:

The current amount of riders aboard (3.1.3.3.2)

The current location assigned(3.1.3.3.1)

* + - * 1. Provide a GUI with the ability to change each virtual train properties as defined:

The current amount of riders onboard

The location assigned (3.1.3.3.1)

* + - * 1. Provide a GUI with the ability to change ridership data at each stop.
    1. **Web Application Engine** 
       1. **Alert Module (Chris Coykendall)**
          1. SQL query the Events database to obtain a list of ongoing outages and exceptions to normal vehicle operations (during the initial page load, select only ongoing Event alert records which have null end timestamps.)
          2. SQL query the Events database to create or modify alerts for outages and exceptions to normal vehicle operations. (Requirement 3.1.1.4)
          3. Provide an input form viewable only in the HRT GUI which enables HRT to create a new alert with the following fields:  
               
             Title  
             Begin  
             End (Estimated if future)  
             Description  
             Course of Action  
             Submit
          4. Upon pressing the Submit button, an Event for the alert must be created via SQL query to the Events database with an initialized end time stamp of null. (Requirement 3.1.4.1.3, 3.1.4.1.2)
          5. Must provide an additional Close button next to events that are ongoing in the HRT GUI view, which will set the end timestamps for the particular event to the current date/time via SQL query in the Events database. (Requirement 3.1.1.4.2)
       2. **Feedback Module (Chris Coykendall)**
          1. Must provide an input form which accepts the following fields:   
               
             Name  
             E-mail  
             Subject  
             Message  
             Submit Button  
             Reset Button
          2. Upon pressing Submit, an email must be sent to 411red@cs.odu.edu with the field contents of the input form and display a confirmation message. (Requirement 3.1.4.2.1)
       3. **System Overview Module (Chris Coykendall)**
          1. Utilize the DB Interface to obtain the stop positions, vehicle positions in latitude/longitude, and coarse (vacant/full) ridership for vehicles in operation. (Requirement 3.1.4.11.1)
          2. Overlay graphical markers for stop positions and vehicle positions onto a Google Maps satellite view of The Tide rail system, using their GPS coordinates to position the markers. (Requirement 3.1.4.3.1)
       4. **Google Maps Web Form (Chris Coykendall)** 
          1. A form from the Google Maps API must be obtained which will accept a current location and destination from the user, and direct them to Google Maps upon submission.
       5. **Calendar Event Module (Chris Coykendall)**
          1. The Events database must be queried to obtain a list of ongoing Events (ongoing is defined as Events with end timestamps of null.) (Requirement 3.1.1.4)
          2. Must provide an input form which accepts the following fields in the Business GUI:   
               
             Title  
             Start Date  
             End Date  
             Message  
             Submit Button  
             Reset Button
          3. Upon pressing the Submit button, the Calendar Event Module must SQL query the Events database to create the new Event. (Requirement 3.1.1.4)
       6. **Ridership Trend Report** **(Brian Dunn)**

The Ridership Trend Report will provide ridership information for display on all three GUI Frameworks (Requirement 3.1.1.4). The following functions shall be provided:

* + - * 1. Customization of data for the different authentication levels through the GUI Framework. (Requirement 3.1.1.4)
        2. Selection of a date for which to view hourly report.
        3. Selection of a range of dates for which to display summarized report averaging data over the timespan.
        4. Presentation of table with ridership information: time, number of departures, and number of arrivals.
        5. DE Interface to retrieve real-time data for reports. (Requirement 3.1.4.11.2)
      1. **Train Data Report (Brian Dunn)**

The Train Data Report provide on-timer performance data for display on the HRT GUI (Requirement 3.1.1.4.2). The following functions shall be provided:

* + - * 1. Selection of a date for which to view hourly report.
        2. Selection of a range of dates for which to display summarized report averaging data over the timespan.
        3. Presentation of table with train performance information: time, on-time percentage, and delay time.
        4. DE Interface to retrieve real-time data for reports. (Requirement 3.1.4.11.2)
      1. **Business Ad Campaign Module** **(Brian Dunn)**

The Business Ad Campaign Module will allow the Businesses GUI and HRT GUI to create and modify advertisement campaigns. The following functions shall be provided:

* + - * 1. Submission of new advertisement campaigns through the Business GUI. (Requirement 3.1.1.4.3)
        2. Editing of current advertisement campaigns through the Business GUI. (Requirement 3.1.1.4.3)
        3. Modification of advertisement campaigns through the HRT GUI. (Requirement 3.1.1.4.2)
        4. Submission of required image file or text to display in advertisement.
        5. DB Interface to modify and retrieve real-time data for advertisements. (Requirement 3.1.4.1.11.1)
      1. **User Management (CJ Deaver)**
         1. Provide a web based management module to control user accounts with the following capabilities:

Include a registration application to allow individual users to insert name, desired username, email address, and password.

Username request should be validated and suggestions returned for alternates when selected name already exists.

Incorporate a method for a user to retrieve their username.

The capability to allow user to reset the password.

Contain an application to allow administrators to perform administrative tasks to the user accounts.

User information update screen allowing users access to change personal information is also required.

Have an ability to create user groups and manage members.

* + - 1. **Authentication** **(CJ Deaver)**
         1. Provide a methodology for security control utilized throughout the application. The following are the minimum requirements:

A one-factor authentication mechanism for securely authorizing application access.

Token generation or other means of identifying users.

Access control mechanism controlling account access within the application.

Security time-out feature to log users out after a set amount of time

Logging capability recording the following:

Login time

Login location

Authentication success or failure

Page requested

* + - 1. **Data Integration** **(CJ Deaver)**
         1. Provide an interface for connecting the web application to the different data sources.
         2. DB Interface

Capability to open and close data stream connections

Transfer queries and results between the database and the Web Application Engine

Contain the ability to prevent SQL injection attacks.

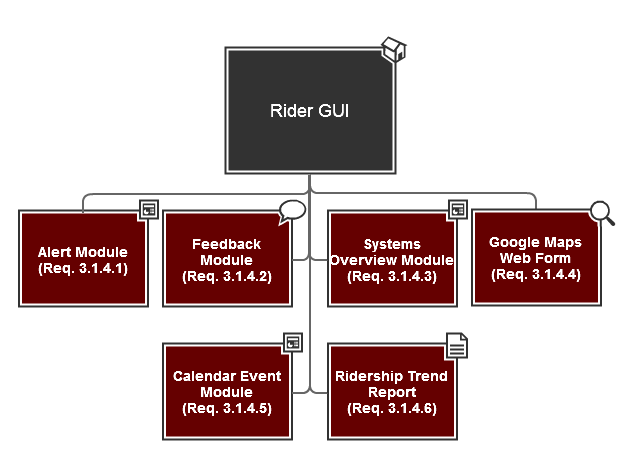
* + - * 1. DE Interface

Send and receive data between the decision engine and the Web Application Engine.

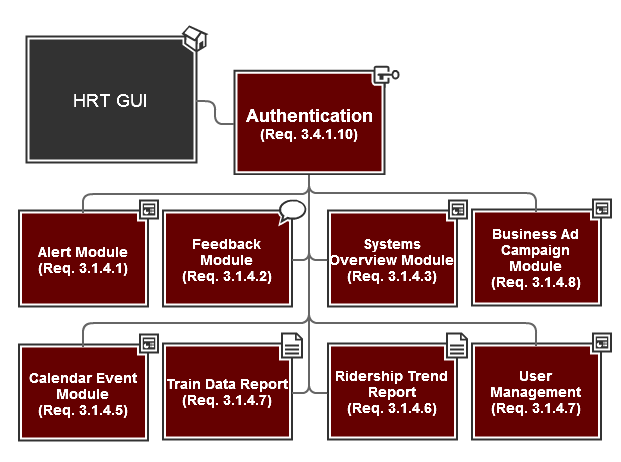
* + - * 1. Test Harness Interface

Receive data from the test harness and pass to the database or decision engine as applicable to replicate real-world data.

* + - 1. GUI Framework **(Brian Dunn)**
         1. Rider GUI



* + - * 1. HRT GUI



* + - * 1. Business GUI

