ETFOMM Reference Manual Volume 2, API Functions

Version 1.3

Prepared by:
New Global Systems for Intelligent Transportation Management
75 Cavalier Blvd Suite 221
Florence, KY 41042

May 2017

TRAFVU and ***TRAFED*** are trademarks of ITT Industries, Inc., Systems Division.

Abstract

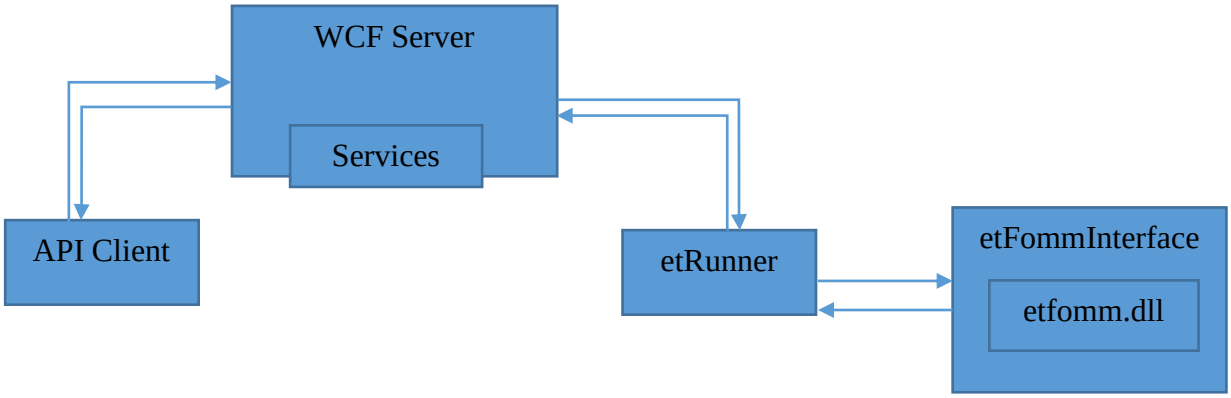
ETFOMM is an open-source traffic simulation that was developed to be used within TSIS and to be compatible with TRAFED and TRAFVU. Therefore, it was developed to read the same input records that CORSIM reads. It can be used in the same manner that CORSIM can be used. In addition, a 64-bit version of ETFOMM can be integrated into etRunner and used without TSIS. When integrated into etRunner, ETFOMM has several API functions that can be called to extend the functionality of the simulation.

Volume 1 of the Reference Manual provides the descriptions of the Record Types that are used by both CORSIM and ETFOMM. It was derived from the CORSIM Reference Manual. It provides the details required by users to manually edit the TRF file inputs to ETFOMM.

Volume 2 describes the API functions that etRunner can use to communicate with ETFOMM.

Contents

Architecture and Data Flow Path



Introduction

This document describes the functions used to implement the communications between etRunner and ETFOMM. ETFOMM is built as a DLL that is loaded by etFommInterface.

Functions Required to Execute a Simulation

When TSIS executes a simulation using CORSIM it calls several functions that are exported from CORSIM.

Prior to starting the simulation TSIS calls SETINPUTNAME to tell CORSIM which input file to use, calls SETIOFLAGS to define optional settings for outputs and calls SETOUTPUTNAME to define the name of the output file.

To perform the simulation it calls STARTUP once to initialize arrays and read inputs, calls RUNTOEQUILIBRIUM once to perform the initialization phase, calls SIMULATE to perform one timestep of simulation repeatedly until the simulation is finished, and calls SHUTDOWN to close files and deallocate memory.

For ETFOMM to be compatible with TSIS it has to export functions with the same names and functionality. etRunner uses those functions to perform a simulation.

SETINPUTNAME (FNAME)

FNAME is a structure that includes a character string specifying the name of the input file and an integer defining the length of the string.

SETIOFLAGS (TSFLAG, TIFLAG, LFLAG, CFLAG)

Inputs are integers [0,1] that are used as logical flags.

TSFLAG – write timestep data to the animation files

TIFLAG – write time interval data to the animation files

LFLAG – write to the output file (not used by ETFOMM)

CFLAG – write a CSV file equivalent to the output file (not used by ETFOMM)

SETOUTPUTNAME (FNAME)

FNAME is a structure that includes a character string specifying the name of the output file and an integer defining the length of the string.

GET_ETFOMM_MESSAGE_COUNT

Returns the number of messages currently in the message buffer.

GET_ETFOMM_MESSAGE

Returns the next message currently in the message buffer.

STARTUP

Starts timers and reads all inputs from the input file specified by SETINPUTNAME().

API_STARTUP

Starts the same timers as STARTUP, but does not read input data. All inputs must be passed in through function calls from etRunner.

RUNTOEQUILIBRIUM

Performs simulation without data collection until the initialization phase is completed. Does not produce animation files or MOEs.

SIMULATE

Performs one time step of simulation with data collection. Produces animation files and MOEs if the user chose to do so.

SHUTDOWN

Closes all files opened during the simulation and deallocates all dynamically allocated arrays.

Functions for Passing Data into ETFOMM

These functions have a single argument that is an integer.

SET_NUMBER_OF_FREEWAYLINKS

Set the number of freeway links in the network. Allocates freeway link arrays to the number of freeway links.

SET_NUMBER_OF_STREETLINKS

Set the number of street links in the network. Allocates street link arrays to the number of street links.

SET_NUMBER_OF_ENTRYNODES

Set the number of entrynodes. Allocates entrynode arrays to the number of entrynodes.

SET_NUMBER_OF_FTC_SIGNALS

Set the number of fixed-time control signals. Allocates fixed-time control signal arrays to the number of fixed-time control signals.

SET_NUMBER_OF_AC_SIGNALS

Set the number of actuated control signals. Allocates actuated control signal arrays to the number of actuated control signals.

SET_NUMBER_OF_RAMPMETERS

Set the number of rampmeters. Allocates rampmeter arrays to the number of rampmeters.

SET_NUMBER_OF_FREEWAY_DETECTORS

Set the number of freeway detectors. Allocates freeway detector arrays to the number of freeway detectors.

SET_NUMBER_OF_STREET_DETECTORS

Set the number of street detectors. Allocates street detector arrays to the number of street detectors.

SET_NUMBER_OF_BUSROUTES

Set the number of bus routes. Allocates bus route arrays to the number of bus routes.

SET_NUMBER_OF_EVENTS

Set the number of blockage events. Allocates event arrays to the number of blockage events.

SET_NUMBER_OF_PARKING_ZONES

Set the number of parking zones. Allocates parking arrays to the number of parking zones.

SET_NUMBER_OF_INCIDENTS

Set the number of freeway incidents. Allocates incident arrays to the number of incidents.

SET_NUMBER_OF_DIVERSIONS

Set the number of freeway diversions. Allocates diversion arrays to the number of diversions.

These functions have a single argument that is a user-defined type containing data specific to the function.

SET_RUN_INPUTS

Define run number for multiple runs, random number seeds and a flag to write MOEs.

SET_NETWORK_INPUTS

Define time period durations, time interval and a flag to skip initialization.

SET_FREEWAY_NETWORK_INPUTS

Define inputs specific to freeway networks: lag to accelerate or decelerate, friction coefficient, default HOV utilization percentage, car following factors, desired speed distribution table, percentage of cooperative drivers, lane change duration, and multiplier for discretionary lane changes.

SET_STREET_NETWORK_INPUTS

Define inputs specific to surface street networks: acceptable gaps at signs, amber deceleration table, bus station dwell multiplier, car following factors, desired speed distribution table, percentage of cooperative drivers, lane change duration, left turn jumper and lagger percentages, maximum left and right turning speeds, probability of joining spillback, speed at which a vehicle is considered stopped, speed for vehicles going through a yield sign, driver familiarity with path, multiplier distribution for startup lost time and queue discharge headway, pedestrian distributions, multiplier distribution for short term events.

DEFINE_VEHICLE_TYPES

Define properties of vehicle types: length, headway factor, average occupancy, emergency deceleration, distribution of types within fleets for freeway and street networks.

DEFINE_FREEWAYLINKS

Define all of the properties of all of the freeway links: upstream node, downstream node, length, number of full lanes, lane alignments, type of link, auxiliary lanes, added or dropped lanes, mean freeflow speed, car following multiplier, exit percentage, detectors, grade, tilt, curvature, pavement type, lane widths, vehicle-type exclusions, anticipatory lane change inputs, vehicle-type exit multipliers, mean startup time from a rampmeter, HOV lane inputs, through receiving link, offramp receiving link, shoulder width, barriers and datastation settings.

DEFINE_STREETLINKS

Define all of the properties of all of the street links: upstream node, downstream node, length, number of full lanes, receiving links, number of left and right pocket lanes, length of left and right pocket lanes, mean freeflow speed, lane alignments, car following multiplier, sight distance, right turn on red code, pedestrian code, grade, vehicle-type exclusions, turn percentages, vehicle-type specific turn multipliers, and the width of the intersection at the upstream end of the link.

DEFINE_ENTRYNODES

Define global settings applicable to all entrynodes: type of distribution, erlang parameter, and minimum separation between vehicles. Define all inputs for each entrynode: flowrate, truck percentage, carpool percentage, percentage of HOV lane violators, lane distribution.

DEFINE_FTC_SIGNALS

Define operating parameters for every fixed-time control signal: approach links, number of intervals, duration of intervals, offset, and signal codes for each interval.

DEFINE_AC_SIGNALS

Define operating parameters for every actuated control signal: approach links, phase inputs, detectors.

DEFINE_RAMPMETERS

Define operating parameters for every rampmeter: type of metering control, onset time, headway for clock-time metering, capacity for demand/capacity metering, speed thresholds for speed control metering, detectors used for metering, detector update interval and two-per-green setting.

DEFINE_FREEWAY_DETECTORS

Define operating parameters for every freeway detector: ID, link, lanes covered, location on the link, length of the sensing area, carryover time, delay time and operation code.

DEFINE_STREET_DETECTORS

Define operating parameters for every street detector: ID, link, lanes covered, location on the link, length of the sensing area, carryover time, delay time and operation code.

DEFINE_BUSROUTES

Define operating parameters for every bus route: route number, offset, headway, links in route and stations along the route.

DEFINE_BUSSTATIONS

Bus station arrays are dimensioned to 99 stations. Define operating parameters for every bus station: link, location on link, blocking code, capacity, mean dwell time and bypass percentage.

DEFINE_EVENTS

Define operating parameters for every blockage event: link, lane affected, start time, end time and type of event.

DEFINE_PARKING_ZONES

Define operating parameters for parking zones: frequency and mean duration of parking maneuvers, began and end location of parking zone.

DEFINE_INCIDENTS

Define operating parameters for freeway incidents: link, began and end location, begin and end time, rubbernecking factor, and type of incident.

DEFINE_DIVERSIONS

Define operating parameters for freeway diversions: link, location of warning sign, begin and end time, diversion route path ID, percentage of vehicles that will obey the diversion order and the speed at which vehicles should exit the freeway.

DEFINE_NODE_COORDINATES

Define node X- and Y-coordinates.

DEFINE_CONDITIONAL_TURNPCTS

Define conditional turn percentages for street links: left, thru, right and diagonal exit percentages for vehicles that entered the link via a left turn, left, thru, right and diagonal exit percentages for vehicles that entered the link via a thru movement, left, thru, right and diagonal exit percentages for vehicles that entered the link via a right turn, and left, thru, right and diagonal exit percentages for vehicles that entered the link via a diagonal movement.

DEFINE_INTERSECTION_DATA

Define detailed intersection data, including lane locations and length of paths that vehicles take when performing a turn through the intersection.

These functions have no arguments. They are called after all links have been defined.

PROCESS_FREEWAYINPUTS

Called after passing in freeway link inputs.

PROCESS_STREETINPUTS

Called after passing in street link inputs.

Functions for Getting Data from ETFOMM

These functions have no arguments. They return an integer value through the function call.

GET_NUMBER_OF_FREEWAYLINKS
GET_NUMBER_OF_STREETLINKS
GET_NUMBER_OF_ENTRYNODES
GET_NUMBER_OF_FTC_SIGNALS
GET_NUMBER_OF_AC_SIGNALS
GET_NUMBER_OF_RAMPMETERS
GET_NUMBER_OF_FREEWAY_DETECTORS
GET_NUMBER_OF_STREET_DETECTORS
GET_NUMBER_OF_BUSROUTES
GET_NUMBER_OF_VEHICLE_TYPES
GET_NUMBER_OF_EVENTS
GET_NUMBER_OF_PARKING_ZONES
GET_NUMBER_OF_INCIDENTS
GET_NUMBER_OF_DIVERSIONS
GETCPUTIME
GETELAPSEDTIME

These functions have a single argument that is a user-defined type containing data specific to the function. In each case, the data description is the same as the data passed into ETFOMM. The data returned by these functions represents the data that was either passed in by functions described above, or read from an input file.

GET_RUN_INPUTS
GET_NETWORK_INPUTS
GET_FREEWAY_NETWORK_INPUTS
GET_STREET_NETWORK_INPUTS
GET_VEHICLE_TYPES
GET_FREEWAY_LINKS
GET_STREET_LINKS
GET_ENTRYNODES
GET_FTC_SIGNALS
GET_AC_SIGNALS
GET_RAMPMETERS
GET_FREEWAY_DETECTORS
GET_STREET_DETECTORS
GET_BUSROUTES
GET_BUSSTATIONS
GET_EVENTS
GET_PARKING_ZONES
GET_INCIDENTS
GET_DIVERSIONS
GET_NODE_COORDINATES
GET_CONDITIONAL_TURNPERCENTAGES
GET_INTERSECTION_DATA

These two functions have a single argument that is a user-defined type containing data specific to detectors. In each case, the data description is the same as the data passed into ETFOMM. The data returned by these functions represents the data that was either passed in by functions described above, or read from an input file, plus the current values stored in the detectors.

GET_FREEWAY_DETECTOR_DATA
GET_STREET_DETECTOR_DATA

These functions are used to get all available data for every vehicle in either the freeway or street network.

GET_FVEHICLE_STRUCT_SIZE

Returns the current size of the freeway vehicle arrays.

GET_FVEHICLE_STRUCT

Returns a user-defined struct containing data for all freeway vehicles.

GET_SVEHICLE_STRUCT_SIZE

Returns the current size of the street vehicle arrays.

GET_SVEHICLE_STRUCT

Returns a user-defined struct containing data for all street vehicles.

These functions are used to get data for specific signals in the street network. NODE and N are integers. PHASES is an array of two integers.

GET_INTERVAL(NODE)

Returns the current interval of the fixed-time control signal at node NODE.

GET_DURATION(NODE, N)

Returns the currently specified duration of interval N of the fixed-time control signal at node NODE.

GET_PHASES(NODE, PHASES)

Returns the currently active phases of the actuated control signal at node NODE.

Functions for Dynamically Manipulating the Simulation

These functions can be used to change the simulation as it progresses. Changes are effective immediately.

CHANGE_DURATION(NODE, N, T)

Change the duration of interval N for the fixed-time control signal at node NODE to T seconds. All inputs are integers.

INCREMENT_INTERVAL(NODE)

Increment the current interval for the fixed-time control signal at node NODE. All inputs are integers.

CHANGE_ENTRYVOLUMES(INPUTS)

Change entry volumes as specified in the user-defined struct INPUTS.

CHANGE_TURNPERCENTAGES(USN, DSN, LPCT, TPCT, RPCT, DPCT)

Change the left, thru, right and diagonal turn percentages for the link from upstream node USN to downstream node DSN. All inputs are integers.

ADD_EVENT(EVENT)

Add a blockage event for a street as specified in the user-defined struct EVENT.

CLOSE_LANE(USN, DSN, LANE)

Close lane LANE on the link from upstream node USN to downstream node DSN. Vehicles will avoid the lane.

REOPEN_LANE(USN, DSN, LANE)

Reopen lane LANE on the link from upstream node USN to downstream node DSN. Vehicles will begin using the lane.

These functions can be used to manipulate vehicle data. All vehicles are represented in the structs.

SET_FVEHICLE_STRUCT(FVDATA)

The size of the freeway struct must be defined by using GET_FVEHICLE_STRUCT_SIZE prior to calling this function.

SET_SVEHICLE_STRUCT(SVDATA)

The size of the street struct must be defined by using GET_SVEHICLE_STRUCT_SIZE prior to calling this function.

These functions can be used with actuated signals.

GET_CONTROLLER_ID(NODE, CONTROLLER_ID)

Returns the controller id for the actuated signal at node NODE.

GET_LOCAL_CYCLE_TIMER(NODE, LOCAL_CYCLE_TIMER)

Returns the value of the local cycle timer for node NODE.

GET_CYCLE_LENGTH(NODE, CYCLE_LENGTH)

Returns the value of the cycle length for the actuated signal at node NODE.

GET_OFFSET(NODE, OFFSET)

Returns the value of the offset for the actuated signal at node NODE.

SET_NEW_CYCLE_LENGTH(NODE, CYCLELENGTH)

Changes the new cycle length of the actuated signal at node NODE.

GET_NEW_CYCLE_LENGTH(NODE, CYCLELENGTH)

Returns the new cycle length of the actuated signal at node NODE.

SET_NEW_OFFSET(NODE, OFFSET)

Changes the new offset of the actuated signal at node NODE.

GET_NEW_OFFSET(NODE, OFFSET)

Returns the new offset of the actuated signal at node NODE.

SET_NEW_SPLITS(NODE, SPLITS)

Changes the new splits of the actuated signal at node NODE.

GET_NEW_SPLITS(NODE, SPLITS)

Returns the splits of the actuated signal at node NODE.

GET_MIN_SPLITS(NODE, SPLITS)

Returns the minimum splits of the actuated signal at node NODE.

SET_TRANSITION_METHOD(NODE, METHOD, MAXPCTADD, MAXPCTSUB)

Changes the transition method of the actuated signal at node NODE. Sets the maximum percentages for add and subtract methods.

GET_TRANSITION_METHOD(NODE, METHOD, MAXPCTADD, MAXPCTSUB)

Returns the transition method of the actuated signal at node NODE, and the maximum percentages for add and subtract methods.

SET_ETFOMM_PHASE_STATES(NODE, GP, YP)

Sets the current phase states of the actuated signal at node NODE. GP is an 8-bit integer that represents the phases that are green. YP is an 8-bit integer that represents the phases that are yellow. Any phase that is not green and is not yellow is red.

GET_ETFOMM_PHASE_STATES(NODE, GP, YP)

Returns the current phase states of the actuated signal at node NODE. GP is an 8-bit integer that represents the phases that are green. YP is an 8-bit integer that represents the phases that are yellow. Any phase that is not green and is not yellow is red.

SET_ETFOMM_MOE_DATA(NODE, MAXOUTS, MIN_GREENS, TIMES_STARTED)

Passes in MOE values for the actuated signal at node NODE.

SET_PHASES(NODE, PHASES)

Sets the currently active phases for the signal at node NODE. PHASES is an array of two integers.

SET_EXTERNAL_ACTUATED_CONTROL(NODE, FLAG)

Sets a flag to indicate that the actuated signal at node NODE is controlled externally. ETFOMM will operate the signal if FLAG is FALSE, and will not operate the signal if FLAG is TRUE.