

# TrafficModeler: A Graphical Tool for Programming Microscopic Traffic Simulators through High-level Abstractions

Leontios G. Papaleontiou and Marios D. Dikaiakos

*Department of Computer Science, University of Cyprus 1678 Nicosia, Cyprus*

## Abstract

*In this paper, we present TrafficModeler, an open-source, graphical tool for the rapid high-level modeling and generation of vehicular traffic. TrafficModeler supports a variety of traffic definition models representing a wide range of traffic patterns. A set of traffic generation algorithms are implemented to convert high-level models to output compatible with SUMO, a popular open-source microscopic traffic simulator. TrafficModeler drastically reduces the time and effort required to generate traffic for SUMO. Furthermore, it can be easily extended to support other traffic simulators and to incorporate new types of traffic.*

## 1. Introduction

Traffic simulators are computer programs that simulate the movement of vehicles in a network of streets. Scientists and engineers use traffic simulators to conduct research and design in congestion relief, road network infrastructure modification, environmental studies, public transportation systems analysis and vehicular networking. As the number of vehicles, the size and the complexity of road networks that need to be simulated grow, it is becoming increasingly difficult to define the traffic that a simulator needs to calculate. Despite the existence of a number of commercial traffic generation and simulation tools [1], [2], these tools are proprietary, expensive, and usually offer little room for expansion and/or modification to their internal algorithms. To address this problem, we designed, developed, and released *TrafficModeler*, an open-source traffic modeling and generation tool [3]. *TrafficModeler* simplifies the process of traffic definition by supporting a set of high-level traffic specification abstractions via an easy-to-use graphical user interface. The traffic definition modeling support incorporated in *TrafficModeler* is structured in such a way so as to combine both everyday traffic and traffic caused by special circumstances.

*TrafficModeler* generates traffic for *SUMO* (Simulation of Urban MObility) an established, open-source space-continuous, time-discrete, microscopic traffic simulator, which was developed by the Institute of Transportation Research at the German Aerospace Centre [4], [5]. *SUMO* represents the road network using a directed graph encoded in XML format. The traffic simulated by *SUMO* is also defined in XML files containing definitions of *routes* and *trips*, which correspond to single-hop and multi-hop vehicular trajectories, respectively. *SUMO* supports the “junction turning ratio-based” and the “district-based” traffic definition models. A user of *SUMO* is required to specify the type and route of every single vehicle in the simulation. This makes it extremely difficult, time-consuming and error prone to specify traffic for a large number of vehicles. *TrafficModeler* addresses this problem by enabling users to specify vehicular traffic using high-level abstractions and by translating these abstractions to the low-level input specification understood by *SUMO*.

The remainder of this paper is organized as follows: Section 2 introduces our high-level modeling approach. Section 3 discusses the translation of demographic-based abstractions into low-level trip specifications. Section 4 gives an overview of the system design and implementation, and Section 5 concludes with a summary of results and future work.

## 2. Traffic Definition Abstractions

To provide *TrafficModeler* with a high-level traffic definition functionality, we have defined and implemented a number of traffic-modeling abstractions, such as: traffic flow elements, population activities that induce particular traffic patterns, emergency road conditions, and vehicle types. Instances of these abstractions are represented and serialized according to the *Traffic Definition Language (TDL)*, an XML-based schema (a complete definition of TDL is given in [6]). In

this section we briefly describe the main abstractions supported by the current version of TrafficModeler.

**Layering:** TrafficModeler introduces and implements a *layered* traffic definition model describing traffic via a set of traffic layers placed over a road network. Layers can be thought of as transparent sheets stacked one on top of another. The traffic generated from each layer is combined to generate the final result. Each layer defines traffic in a specific way according to its type. Currently, TrafficModeler supports three types of traffic layers: i) *User-defined traffic*: this layer consists of traffic generated from traffic patterns defined directly by the user; ii) *Activity-based traffic*: this layer consists of traffic generated indirectly, according to demographic data that characterize the population of a map; iii) *Random traffic layer*: this layer consists of traffic generated according to some random distribution.

Layering offers some important advantages: First, it simplifies model development. This is especially important when modeling the traffic of large city-areas or when creating models that are very complex; different aspects of such models logical or otherwise may be represented on different layers. Secondly, layering gives the user the ability to enable or disable a traffic layer. Consequently, some traffic elements can easily be left out when creating the traffic that the model defines, so that their effects can become more obvious. Another reason in favor of the layered approach is the ability to combine layers from different sources. When working with large models, it is sometimes difficult for a single user to create the whole model. By splitting the model into layers, multiple users may simultaneously work on different layers and later combine them to form the final model.

**The Traffic Definition Element Abstraction:** Traffic is defined inside layers using *traffic definition elements*. These are objects that represent specific traffic patterns. Each element is associated with a set of *attributes* that define how traffic will be generated for that element. These attributes can be time-related (e.g. time of vehicle departure), location-related (e.g. vehicle source area) or other (e.g. number of emitted vehicles). Furthermore, each traffic definition element is associated with a traffic generation algorithm, which generates traffic based on the element's attributes.

**User-defined traffic layer:** The user-defined traffic layer can be used to define traffic directly, according to the following traffic definition elements:

*Street-to-street flow element:* This element represents a flow of vehicles starting from one street and ending to another. The flow is characterized by the number of vehicles that will be emitted and the duration of

time during which the emission will take place. The attributes of this element are: the streets linked by the traffic flow; the number of vehicles to be emitted; the time at which the vehicle emission starts, and the time at which the vehicle emission stops.

*Area flow element:* Represents a flow of vehicles going from one area to another. An area is defined as a two-dimensional shape that is placed over a part of the road network; the exact source and destination streets of the vehicles are not specified explicitly but are calculated internally by TrafficModeler.

*Hotspot element:* This element represents an area that is a popular starting and/or destination point of vehicles (outgoing and incoming hotspots, respectively). An outgoing hotspot is defined by the following attributes: the area from which the vehicles depart (the hotspot); the number of vehicles to be emitted; the time at which the vehicle emission starts and the time at which the vehicle emission stops. An incoming hotspot represents traffic that starts from any part of the road network and ends at the hotspot. TrafficModeler supports also a hotspot that is a combination of the above two hotspots, i.e. vehicles start from any part of the road network, arrive at the hotspot, remain there for a period of time and then return to where they initially started from. A screen-shot of TrafficModeler's support for defining hotspot elements is depicted in Figure 1.

*Accident element:* The accident element represents the closing of some or all the lanes of a street for a period of time, due to accidents, illegal parking, demonstrations etc. The element's attributes are: the street that will be closed; the lanes of the street that will be affected; the time at which the accident happens, and the time at which the street reopens.

For all elements, the emission of the vehicles is uniformly distributed in time. This means that, on average, the same number of vehicles will be emitted every second. Likewise, the spatial distributions of the emitted vehicles for the area flow and hotspot traffic elements are uniform.

**Activity-based traffic layer:** The activity-based layer models traffic by creating virtual populations based on demographic data [7], [8]. This model generates everyday activities for a given population and calculates the traffic resulting from these activities [9]. Assuming that most of the traffic during a day occurs due to transportation between home, school and work, the definition of the activity-based traffic layer focuses on these main activities for a virtual population. We have introduced two types of traffic elements for the activity-based traffic layer: the traffic area element and the school element.

The *traffic area element* is a two-dimensional shape

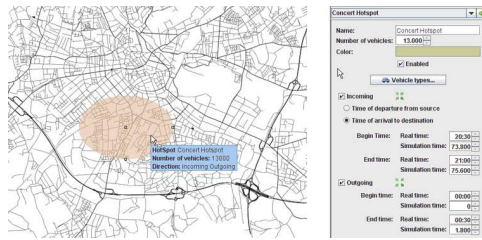


Figure 1. TrafficModeler: graphical representation and property panel of the hotspot flow element.

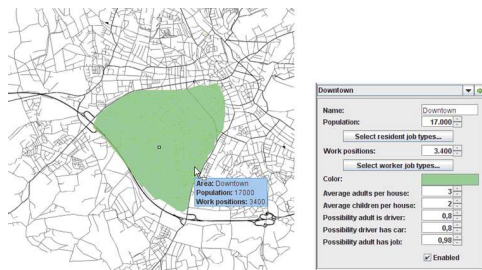


Figure 2. TrafficModeler: graphical representation and property panel of the traffic area element.

which is placed over the road network to define an area with specific demographic properties (see Figure 2). This element has the following attributes: area shape (two dimensional polygon); number of families and residents of the area; average number of adults per family; average number of children per family; total number of available working positions; probability that an adult has a job; probability that an adult is a driver; probability that a driver has a car; types and probability distribution of the jobs of residents; types and probability distribution of job-posts in the area. The last two attributes have been included in the model because a significant amount of traffic occurs due to people going to or returning from work. Thus, it is important to model different kinds of jobs with different starting and ending times, as well as locations.

To manage traffic induced by workers going to work, TrafficModeler supports the abstraction of *job type*. A job type is characterized by its starting and ending time. Job types affect both the temporal as well as the spatial distribution of traffic caused by the daily commute to and from work. The support of job types gives TrafficModeler enough flexibility to define a wide range of traffic areas. For example, an industrial area can be defined as an area that has no residents and a large number of worker positions with the characteristics of industrial job types.

The second traffic element that can be placed on an activity-based traffic layer is the “*school*” element.

Schools are usually places of high traffic activity during their opening and closing times. TrafficModeler supports four types of schools: kindergarten, elementary, middle and high. Furthermore, a school is characterized by its capacity, i.e. the number of children attending it. The type, location and capacity of a school affect the generated traffic since these are the factors that determine which school a child is attending.

### 3. Traffic Generation

TrafficModeler translates the traffic-definition abstractions of a traffic model into a set of vehicle trips. To this end, a traffic generation module loops over the set of enabled traffic layers and generates traffic for each layer, using special algorithms tailored to each traffic-definition abstraction. The basic approach taken by these algorithms is to: i) identify those parts of the road network that are pertinent to the corresponding abstraction (e.g., the street sections that fall into the area of a hotspot); ii) produce a fleet of vehicles according to a given statistical model (e.g., 20% trucks, 10% buses, 30% old cars, 40% default); iii) choose the starting points and emission times of those vehicles by applying a random selection of streets that satisfy the semantics of the abstraction (e.g. vehicles moving towards an incoming hotspot will leave from randomly selected streets, just in time to arrive to their destination); iv) choose accordingly the ending points of vehicle trips, and v) generate the trips in a format understood by SUMO (or any other traffic simulator). The details of these algorithms are omitted due to a lack of space; the interested reader can check [6].

The generation of activity-based traffic is more complex, as it requires the creation of a virtual population based on given demographic data, the generation of every-day activities for this population, and, finally, the generation of the vehicle traffic that results from those activities [9]. All activity-based layers are merged together into a single layer before running the traffic generation algorithm: full knowledge of all the traffic areas and schools defined on all layers is required to generate traffic for a single traffic area. The algorithm for generating activity-based traffic is divided into 4 main steps: i) distribution of available work positions to the areas of the map; ii) virtual population synthesis according to the demographic properties of the map, as specified by the user; iii) children transportation assignment: arranging the transportation of children to and from their schools; iv) traffic generation based on the daily activities of the virtual population.

For the virtual population synthesis, TrafficModeler creates a number of virtual families in each traffic area

so that the area’s population reaches a desired target, specified in the corresponding population model. The characteristics of each family (number of adults and children, age and sex of family members) follow known statistical distributions [8]. Vehicles are assigned to virtual families, and job types and work positions are assigned to adults, according to statistical models derived from the population model at hand. Finally, each child is assigned to a school closest to home; in this process, TrafficModeler takes into account the child’s age, the type of the school (kindergarten, elementary, middle, high) and the capacity of each school.

The children-transportation assignment is important as school travel introduces strong spatial and temporal linkages between the travel patterns of parents and children. Empirical results indicate that the characteristics of children, like age and gender, and the employment and work flexibility characteristics of the parents, have a strong impact on the decisions made regarding the transportation of children [10]. TrafficModeler arranges the transportation of children by their parents to and from their schools by assigning the children to adults that are able to transport them and belong to the same virtual family. The decision of whether an adult is able and available to transport a child to and/or from its school is based on several factors, such as vehicle possession and employment conditions.

The vehicle trips are generated independently for each adult in each virtual family, in two steps: first, we generate trips for going to work and taking the children to school, and second, we generate trips for leaving work and picking up children from school. Trip generation takes into account whether an adult has a job or not and whether he/she is responsible for taking or picking up children from school. To estimate the departure times of vehicles, we take into account parameters such as the distance of the trip, the expected averaged speed, etc. In some cases we need to create trips that pass through intermediate destinations, for example a trip from home to school and then to work.

#### 4. Software Design and Implementation

For the design and implementation of TrafficModeler we followed a bottom-up, object-oriented approach and used the Java programming language. The system comprises various components (see Figure 3), with most important being: i) The *road network importer*, which is responsible for reading and parsing the SUMO-formatted road network from an XML file, creating an internal representation of the network’s

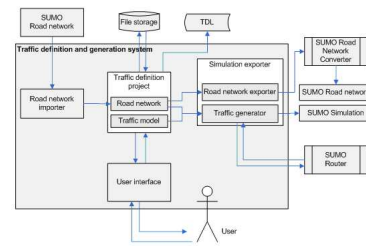


Figure 3. TrafficModeler: System Architecture.

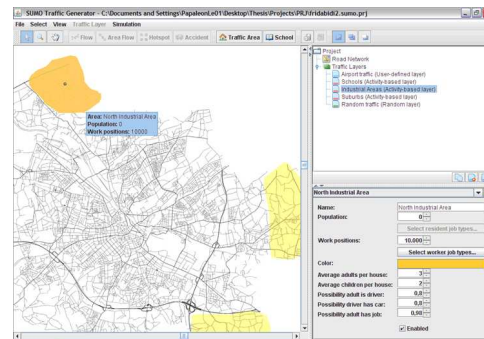


Figure 4. TrafficModeler: Graphical User Interface.

nodes and edges, and preparing the structures needed for drawing the road network on screen; ii) The *traffic definition project*, which manages the main data structure that contains all information regarding a road network and the traffic defined over it; iii) The *simulation exporter*, which regenerates the road network following modifications done by the user and generates traffic based on the traffic definition model, and iv) the *graphical user interface*, which is responsible for displaying the road network and providing graphical tools for defining traffic abstractions and demographic models.

The system introduces the concept of a *traffic definition project* for the definition of traffic on a road network. A project is a data structure that acts as an abstract container for all the information regarding the road network and the traffic defined over it. When the user wants to define traffic on a road network, a new project is created and stored on disk by the system. The user is also allowed to conduct road network manipulation operations, such as temporary and permanent road-network cropping, intersection removal, and street removal. The user may subsequently save, open and close a project from disk. Every project created by the system is stored in a single autonomous file. Furthermore, the project can be exported by the system

in XML format using the TDL language mentioned earlier.

TrafficModeler operates as a single-user integrated development environment whose main purpose is the definition and generation of traffic for SUMO. The system loads and displays a road network and lets the user define traffic on it using graphical tools based on the traffic definition models described in the previous sections. It then generates traffic by compiling traffic definition models in a format that is compatible with the SUMO traffic simulator. The system can also be used for some basic road network manipulation such as temporary or permanent map cropping or intersection and street removal.

The system's main window is comprised of 4 parts (see Figure 4): i) The road network display is used to draw the road network. The traffic definition layers created by the user are placed on top of the road network as transparent sheets; ii) The tool bar is positioned on the top part of the main window and contains the graphical tools that can be used to manipulate the road network and to define new traffic elements; iii) The traffic project explorer is used to display and allow the management of the project's traffic definition layers. Finally, the property panel is used to display the attributes of the currently selected traffic definition element.

## 5. Conclusion and Future Work

In this paper, we introduced TrafficModeler, a graphical tool for the modeling and generation of vehicular traffic. Thanks to its support for high-level traffic definition abstractions and its advanced graphical-user interface, TrafficModeler significantly reduces the time and effort required to define realistic traffic patterns, to configure traffic simulations, and to investigate the effects that road-network and area-demographic changes can have on vehicular traffic. To support its high-level traffic definition abstractions, TrafficModeler implements a number of powerful algorithms that generate vehicle trips from demographic and other models. TrafficModeler's output is compatible with the input of SUMO, a popular open-source traffic simulator. Other traffic simulators can be easily supported as well, since the definition and generation of traffic are implemented as separate software components inside TrafficModeler. Thus, by modifying the traffic generator module, one can easily export the traffic to a format compatible to another traffic simulator.

The system has been released to the scientific community as an open-source software through Sourceforge, and has been used already for the study of ve-

hicular ad-hoc network protocols and services [11]. A number of improvements for TrafficModeler are under way: in particular, we are investigating the support of abstractions for public transportation, multi-modal trips, large parking and shopping areas. We are also investigating the integration of alternative temporal and spatial distribution models for vehicle emission.

## References

- [1] "Vissim microscopic traffic flow simulation for traffic and transit movements," <http://www.ptvamerica.com/vissim.html> (accessed Nov 2006).
- [2] L. Smith, R. J. Beckman, and K. Baggerly, "TRANSIMS: Transportation analysis and simulation system," Los Alamos National Laboratory, Tech. Rep. LA-UR-00-1725, 2001.
- [3] Leontios Papaleontiou, "TrafficModeler," <http://sourceforge.net/projects/trafficmodeler>.
- [4] Centre for Applied Informatics (ZAIK), Institute of Transport Research, German Aerospace Centre, "Sumo - simulation of urban mobility," <http://sumo.sourceforge.net/> (last accessed September 2008).
- [5] D. Krajzewicz, G. Hertkorn, C. Rössel, and P. Wagner, "Sumo (simulation of urban mobility); an open-source traffic simulation," in *4th Middle East Symposium on Simulation and Modelling (MESM2002)*. SCS European Publishing House, September 2002, pp. 183–187.
- [6] L. G. Papaleontiou, "High-Level Traffic Modelling and Generation," Master's thesis, Dept. of Computer Science, University of Cyprus, March 2008.
- [7] M. Birkin, A. Turner, and B. Wu, "A Synthetic Demographic Model of the UK Population: Methods, Progress and Problems," in *Second International Conference on e-Social Science*, June 2006.
- [8] R. J. Beckman, K. A. Baggerly, and M. D. McKay, "Creating synthetic baseline populations," *Transportation research. Part A, Policy and practice*, vol. 30, no. 6, pp. 415–429, 1996.
- [9] M. G. McNally, "An Activity-Based Micro-Simulation Model for Travel Demand Forecasting," Center for Activity Systems Analysis, Tech. Rep. UCI-ITS-AS-WP-96-1, 1996, <http://repositories.cdlib.org/itsirvine/casa/UCI-ITS-AS-WP-96-1>.
- [10] A. K. Yarlagaadda and S. Srinivasan, "Modeling children's school travel mode and parental escort decisions," *Transportation*, vol. 35, no. 2, pp. 201–218, March 2008.
- [11] M. D. Dikaiakos, A. Florides, T. Nadeem, and L. Iftode, "Location-aware Services over Vehicular Ad-Hoc Networks using Car-to-Car Communication," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 8, pp. 1590–1602, 2008.