Smartest

Simulation Modelling Applied to Road Transport European Scheme Tests

http://www.its.leeds.ac.uk/smartest

Simulation Report

Jaime Barceló, Eric Bernauer, Laurent Breheret, Gianni Canepari, Carlo Di Taranto, Jaime Ferrer, Ken Fox, Jean-François Gabard and Ronghui Liu

SMARTEST Project Deliverable D6

Submission Date: May, 99

Circulation Status: P - Public

The “SMARTEST” Project

Contract N^{o}: RO-97-SC.1059

Project part funded by the European Commission under the Transport RTD Programme of the 4th Framework Programme

Smartest

Simulation Modelling Applied to Road Transport European Scheme Tests

http://www.its.leeds.ac.uk/smartest

Simulation Report

DOCUMENT CONTROL INFORMATION

Title	:	Simulation Report
Author(s)	:	Jaime Barceló, Eric Bernauer, Laurent Breheret, Gianni Canepari, Carlo Di Taranto, Jaime Ferrer, Ken Fox, Jean-François Gabard and Ronghui Liu.
Reference Number	:	SMARTEST/D6
Version	:	1.0
Date	:	31 May 1999
Distribution	:	ITS(3), CTS(2), SODIT(2), CERT(2), UPC(2), MIZAR(2), Transek(2), Softeco(2), DGVII(5), HIPERTRANS
Availability	:	Public
File	:	d:\smartest\d8.doc
Authorised by	:	Ken Fox
Signature	:	

TABLE OF CONTENTS

1 INTRODUCTION

This document is the sixth deliverable of the SMARTEST project. The SMARTEST project directly addresses task 7.3/17 in the second call for proposals in the Transport RTD, Road Transport Traffic, Transport and Information Management area. The project is directed toward modelling and simulation of dynamic traffic management problems caused by incidents, heavy traffic, accidents, road works and events. It covers incident management, intersection control, motorway flow control, dynamic route guidance and regional traffic information. The project's objectives are to:

1. review existing micro-simulation models, so that gaps can be identified
2. investigate how the SMARTEST models can best be enhanced to fill the identified gaps, thus advancing the State-of-the-Art
3. incorporate the findings of the study into a best practice manual for the use of micro-simulation in modelling road transport and to disseminate these findings throughout Europe.

This document responds to the second objective of the SMARTEST project: investigate how the SMARTEST models can best be enhanced to fill the identified gaps, and it is the result of Workpackage 4, Modelling.

The objective of Workpackage 4 is to develop the new modelling features and improvements that were identified as gaps in Workpackage 2, Review of Tools, and then prioritised in Workpackage 3, Model update specifications, according to the specifications established in this Workpackage.

Each partner concerned with the development of simulation models has enhanced their models according to the requirement specifications produced in Workpackage 3. Each one has selected a set of improvements that have been implemented and calibrated or verified within their models.

There are four member of the consortium that have enhanced their micro-simulation packages, each one addressed in one chapter of this document: AIMSUN2 (UPC), DRACULA (ITS), NEMIS (Mizar) and SITRA-B+ (CERT).

2 AIMSUN2

2.1 INTRODUCTION

In order to comply with the Model Update Specifications proposed in Workpackage 3, the following functions have been developed or enhanced in AIMSUN2:

- *Incident Management*
- *Adaptive Traffic Signals*
- *Ramp Metering*
- *Variable Message Signs*
- *Dynamic Route Guidance*
- *Results Analysis Tool*

Improvements to the *incident generation* model include deterministic and random incident generation. Deterministic incidents may be defined either through the user's interface or by means of an incidents log file, while random incidents can be generated according to a random distribution that is varied according to certain section characteristics.

The *adaptive traffic signals* improvements consist of a new and more flexible definition of the traffic control plans and the development of a new interfacing protocol between AIMSUN2 and any external traffic control or management application. This link has been implemented by the use of the GETRAM Extensions Module.

Through this interfacing protocol it is possible not only to control any traffic signal but also any *ramp metering* or *Variable Message Sign.*

Regarding *VMS* and *Dynamic Route Guidance Systems*, a better behavioural model that emulates the influence that routing information may have on the drivers has been implemented. To achieve a better characterisation of the behaviour of drivers, several former global parameters have been transformed into local or individual parameters (e.g. compliance level and speed acceptance parameters).

A new *Result Analysis Tool* has been developed. Its main functions are to define and conduct simulation experiments, to perform results analysis, to make output data representation and to provide statistical tools for model calibration and validation. Within the SMARTEST project two of these functions are implemented: experimentation and output data representation modules.

Most of the new models included in AIMSUN2 are based or make use of the *GETRAM Extension Module*, a set of Dynamic Link Libraries (DLL) through which any user is able to either implement or communicate any control or management strategy to AIMSUN2.

2.2 GETRAM EXTENSIONS

2.2.1 Introduction

The current trend in the development of Advanced Transport Telematic Applications, either real-time adaptive, or based on other specific approaches, is far from being standardised. To try to incorporate them in a microscopic traffic simulator in a specific fixed way would therefore be of little use. If any specific ATT application were included in a micro-simulator as an in-built function, it is likely that it would not be suitable for simulating other similar applications.

This is true whenever we address the problem of simulating traffic management and control systems such as, for example:

- adaptive signal control systems (SCOOT, SCATS, SPOT/UTOPIA, PRODYN, BALANCE etc),

- vehicle actuated control,
- public transport priority systems,
- Advanced traffic management systems (using VMS, traffic calming strategies, ramp metering policies, etc),
- Vehicle guidance systems,
- Public transport scheduling and control systems,
- applications aimed at estimating and controlling the environmental impacts of pollutant emissions, and energy consumption.

The main question then is: How can these Advanced Transport Telematic Applications be properly evaluated and tested by simulation?

To evaluate and test any of these systems a micro-simulator must be capable of incorporating in its model the traffic devices that are used by the system: e.g. detectors, traffic signals, VMS, etc. It must also emulate their functions: e.g. provide the specific traffic measurements at the required time intervals, increase the phase timing in a given amount of time, implement a traffic calming strategy (slow down the speed on a road section, recommend an alternative route, etc). How can such evaluations be done by simulation without explicit in-built modelling of the specific Advanced Telematic Application?

The approach taken in GETRAM/AIMSUN2 consists of considering the Advanced Telematic Application to be tested as an *External Application* that can communicate with AIMSUN2. An ad hoc version of AIMSUN2 including a DLL has been developed. This library gives AIMSUN2 the ability to communicate with almost any of the above-mentioned external applications.

Using the Tedi and AIMSUN2 functions, vehicles, traffic signals, detectors, ramp-meters and VMS and can be modelled and their attributes defined. The process of information exchange between AIMSUN2 and the external application is shown in Figure 1:

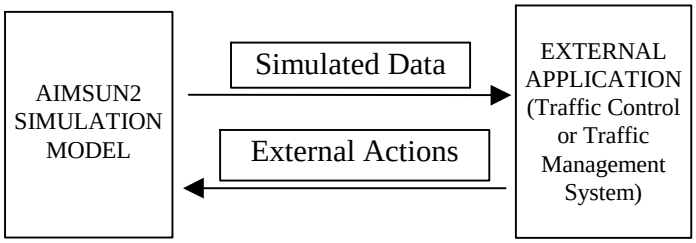


Figure 1: Process of information exchange

The AIMSUN2 model of the road network emulates the traffic providing the external application with the required "*Simulated Data*", which may be vehicle data, detector data, statistical data or control data. The external application (user provided) decides which control, management or other actions have to be applied on the road network and sends the corresponding information to the simulation model which then emulates their operation through the corresponding model components such as vehicles, traffic signals, VMS, etc.

2.2.2 Implementation

The GETRAM Extensions are implemented using DLL's (Dynamic Link Libraries). There are two modules: on one side there is the executable program which corresponds to the simulation logic and on the other side we have a DLL (or a set of DLL's) which corresponds to the control and management logic (or policy).

The DLL has to have four functions defined:

1. GetExtInit(): It is called when AIMSUN2 starts the simulation and can be used to initialise whatever GETRAM Extension needs.
2. GetExtManage(float time, float timeSta, float timTrans, float acicle): This is called in every simulation step at the beginning of the cycle, and can be used to request detector measures, vehicle information and interact with junctions, meterings and VMS in order to implement the control and management policy. This function receives four parameters in relation to time: absolute time of simulation, time of simulation in stationary period, duration of warm-up period, duration of each simulation step.
3. GetExtPostManage(float time, float timeSta, float timTrans, float acicle): This is called in every simulation step at the end of the cycle, and can be used to request detector measures, vehicle information and interact with junctions, meterings and VMS in order to implement the control and management policy. This function receives four parameters in relation to time: absolute time of simulation, time of simulation in stationary period, duration of warm-up period, duration of each simulation step.
4. GetExtFinish(): It is called when AIMSUN2 finish the simulation and can be used to finish whatever GETRAM Extension needs.

The next figure shows graphically how AIMSUN2 and a GETRAM Extension DLL interact.

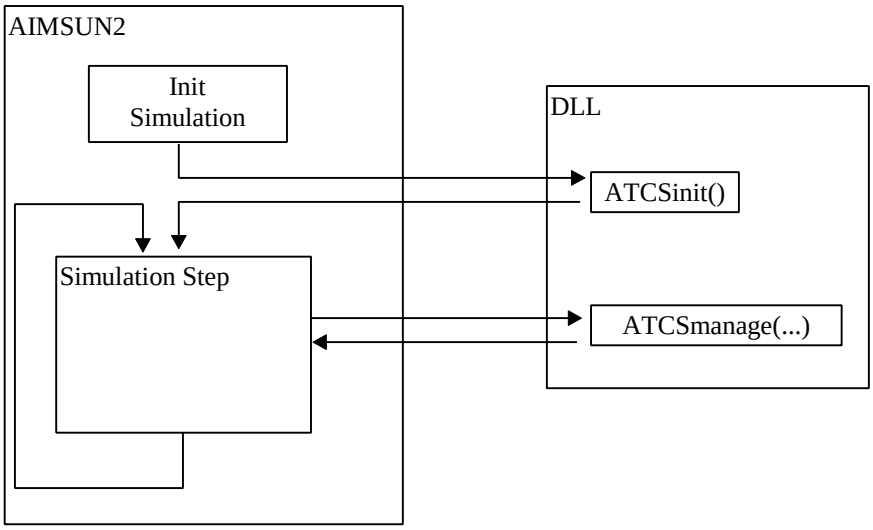


Figure 2: Scheme how AIMSUN2 and GETRAM Extension interact

2.2.3 GETRAM Extensions Functions

The functions provided by the simulator that can be called by the DDL to perform the interaction between AIMSUN2 and the GETRAM Extension can be grouped into different sets, depending on the type of information they are related to: junction control, ramp-metering, VMS, detectors, vehicles or statistics.

Functions relative to control junctions:

- Read the Number of junctions
- Read the Identifier and Name of a junction
- Read the number of Signal Groups of a junction
- Read the Number of Phases of a junction

- Read Time Duration of a phase of a junction
- Read the Current Phase of a junction
- Disable the fixed control plan of a junction
- Enable the fixed control plan of a junction
- Change directly the Phase
- Change the Current Duration of Phase
- Change directly the State of a Signal Group

Functions relative to ramp-metering

- Read Number of meterings
- Read the Section Identifier and Name of a metering
- Read the Type of metering
- Read the Control Parameters of a Green Metering
- Change the Control Parameters of a Green Metering
- Read the Control Parameters of a Flow Metering
- Change the Control Parameters of a Flow Metering
- Read the Control Parameters of a Delay Metering
- Change the Control Parameters of a Delay Metering
- Disable the fixed control plan of a metering
- Enable the fixed control plan of a metering
- Change the State of a metering

Functions relative to VMS

- Read the Number and Identifier of VMS
- Read the number of messages in a VMS
- Read the message of a VMS
- Read the Current Active Message
- Activate a Message in a VMS

Functions relative to detector measures

- Read Number of detectors
- Read Name of a detector
- Read the Detection Interval
- Read the Presence of a detector in the Last Cycle
- Read the Occupied Time of a detector in the Last Cycle
- Read the Counter measure of a detector in the Last Cycle
- Read the Average Speed of a detector in the Last Cycle
- Read the Number of Occupied Intervals of a detector in the Last Cycle
- Read the Initial Time of an Occupied Interval of a detector in the Last Cycle
- Read the Final Time of an Occupied Interval of a detector in the Last Cycle
- Read the SCOOT Occupancy of a detector in the Last Cycle
- Read Counter Aggregated in the Last Detection Interval
- Read Speed Aggregated in the Last Detection Interval
- Read Occupancy Aggregated in the Last Detection Interval
- Read Presence Aggregated in the Last Detection Interval

Functions relative to vehicles information

- Read Number of Vehicles in a Section
- Read Number of Vehicles in a Junction

- Read the information of a Vehicle in a Section
- Read the information of a Vehicle in a Junction

Functions relative to vehicle entrance

- Enter a Vehicle in flows and turning proportions traffic definition
- Enter a Vehicle in O/D matrix traffic definition
- Put a Vehicle in flows and turning proportions traffic definition
- Put a Vehicle in O/D matrix traffic definition

Functions relative to vehicle tracking

- Modify the Speed of a Vehicle Tracked
- Modify the Lane of a Vehicle Tracked
- Modify the Next Section of a Vehicle Tracked
- Remove a Vehicle Tracked
- Read the information of a Vehicle Tracked

Functions relative to statistical data

- Read global statistics data for a section
- Read periodical statistics data for a section
- Read periodical statistics data for the system
- Read global statistics data for the system

2.2.4 Building and enabling the Getram Extension DLL.

The DLL can be built from the supplied files using a C++ compiler. These files are: GetExt_common.h, GetExt _common.cxx, AKIProxie.h, AKIProxie.cxx, CIProxie.h, CIProxie.cxx, GetExt.h, GetExt.cxx. The user can only modify the file GetExt.cxx, fill in the routines GetExtIni(), GetExtManage(…), GetExtPostManage(…) and GetExtFinish() and add some other files in order to implement the policy. After building the DLL with the C++ compiler, it has to be placed in the same directory where AIMSUN2 is located.

To enable the GETRAM Extension, it is necessary to load it before loading a control plan. Through the AIMSUN2 Interface the user can load or unload a set of DLL's. The Save option can be used to save DLL's which have been loaded, then using the Initial button reload the DLL's and add them automatically.

2.3 INCIDENT MANAGEMENT

2.3.1 Introduction

In AIMSUN2 the microscopic traffic simulation of the road network emulates the traffic detector measurements used by an incident detection algorithm. The incident detection algorithm is then a second component of the simulation model, a component that would be supplied by each user according to their current or foreseen practice.

The proposed open platform requires an interface to integrate the two components (traffic simulation model and incident detection model) which could consist of an exchange of detector measurements according to the degree of aggregation and format required by the user.

A third component of the common simulation, according to this approach as open platform, is the traffic management/incident management component, whose integration with the two other components is illustrated in the diagram of Figure 3.

The interfacing between AIMSUN2 and the incident detection and management system is achieved through the GETRAM Extension module, described in the previous section, through which the user is able to implement and communicate any external application to AIMSUN2.

The simulation model emulates traffic flows at the network, and generates incidents according to the specified patterns. The emulation of the detector measurements defines for the detection algorithms an input equivalent to the input that the real detectors supply. This procedure enables the estimation of the detection time taking into account that the simulator knows which is the exact time at which the incident was generated.

The incident detection module communicates to the traffic management and incident response systems the occurrence of the incident and its location. The specific management and response actions, such as motorist information using variable message panels, access control using ramp metering policies, speed control on the main road sections, etc., are decided by the traffic management module and communicated to the simulation model which implements them. The subsequent simulation experiments enable the assessment and evaluation of the impact of the proposed actions.

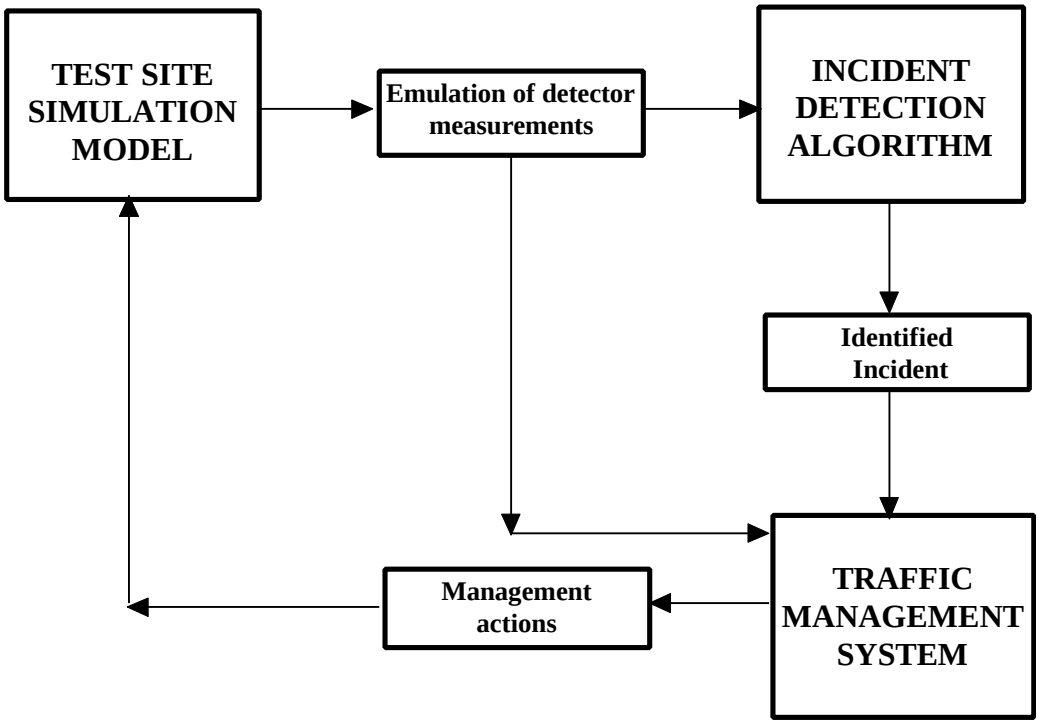


Figure 3: Data flows between the simulator and the detection module

Traffic volume, occupancy, space mean speed and density are data collected by the AIMSUN2 traffic sensors. The detection process can be either based on the direct observations or on more complete information from treated data, e.g., first and second order statistics of the data. Traffic data may include travel time and routing information, e.g., turning movements or tracking of vehicle paths through the test site. Data are sampled at regular intervals. Traffic data are statistically treated and processed.

2.3.2 Inputs

Deterministic Incident Data

- Incident location: identifier of the section where the incident occurs.
- Time at which incident takes place

- Number of lanes blocked by the incident
- Length of incident: part of the lane that becomes useless for drivers
- Duration of Incident

Random Incident Data

For each section

- Time Interval between incident occurrences (mean)
- Duration of Incident (mean and deviation)
- Length of incident (mean and deviation)

Interface description

The above data needs to be supplied to the model. This can be done through the AIMSUN2 Graphical User Interface and the Tedi graphical network editor.

An incident could be generated anywhere on the simulated road network, either in a deterministic way or stochastic. The user, dialoguing with the simulation model through an Incident Generation window may define deterministic incidents.

For deterministic Incidents the user may, through the graphical interface, select the section, the lane in the section, the position on the lane, the lane length blocked by the incident, the time at which the incident will occur, and the incident duration. Depending on the severity of the incident the user may decide whether one lane or more than one lane are going to be blocked by the incident.

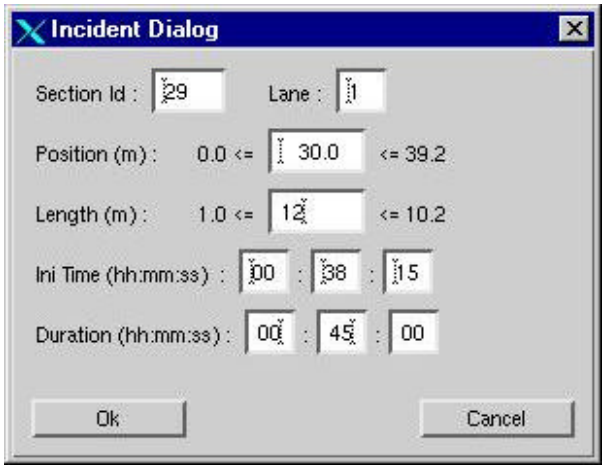


Figure 4: Deterministic Incidents Dialog

The user may define as many deterministic incidents as desired, either at the beginning of the simulation or at any time during a simulation run. A set of incidents for a given model may be stored in a Log File, which can be re-used in future simulation experiments. This feature allows the use of the same set of pre-established traffic incidents for different simulation experiments, in order to evaluate how different alternatives of incident management would behave with the same traffic conditions. The Incidents Log File can be either directly edited as a text file or input through the graphical interface.

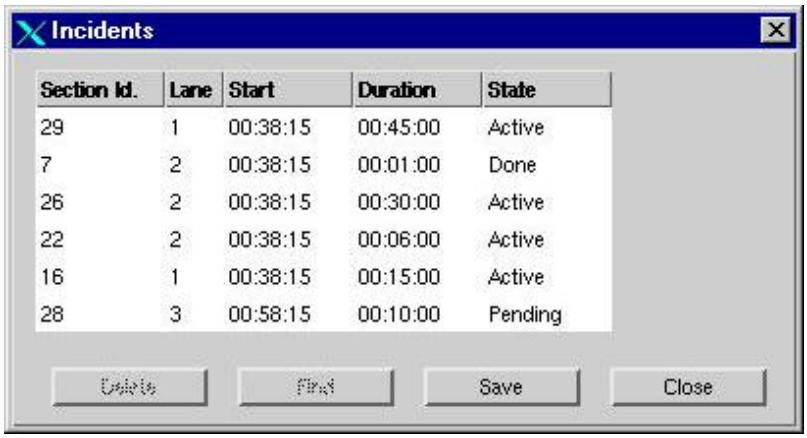


Figure 5: Incidents log dialog

For stochastic incidents, the user may define, using the Tedi editor, the additional parameters required for each section. In this respect, the Section dialog has been modified in order to include the dialog boxes for the time Interval between incidents, their duration (mean and deviation) and length (mean and deviation). A section where these parameters have been defined are assumed to have a probability of incidents occurring greater than zero. Otherwise no incidents are generated for the section.

2.3.3 Processing

The incident detection and management model is able to reproduce the appearance of the incident, the behaviour of traffic at the vicinity of the incident, the on-road equipment installed to detect incidents (e.g. traffic detectors), and the management and control actions implemented to alleviate the impact of the incident.

Figure 6 shows the new Simulation Process including the Incident generation model.

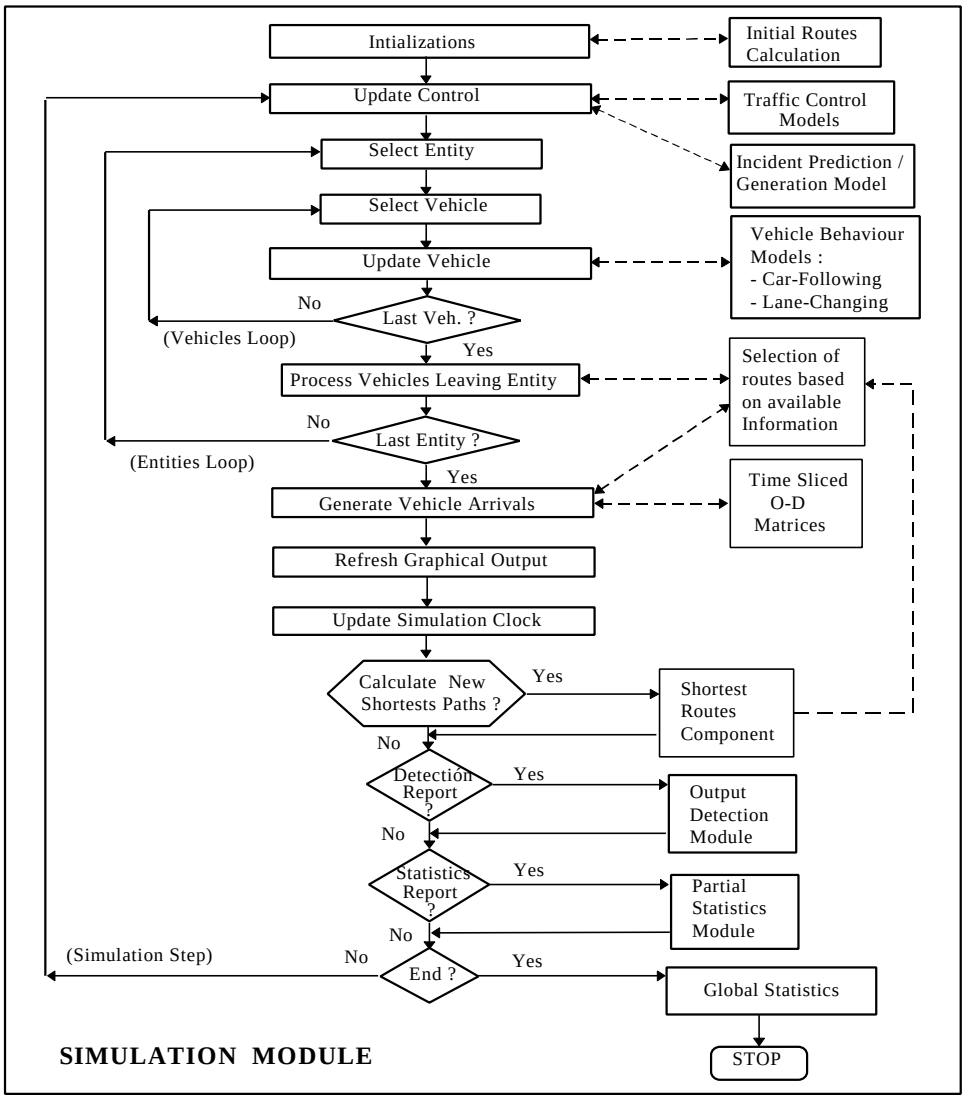


Figure 6: Simulation Process

Modelling the Incident

The model is capable of generating incidents anywhere on the simulated road network and then reproduces the dynamics of the queue and congestion building processes. If an incident prediction or incident warning system is taken into account, then incidents should be created on each road section according to the corresponding probability model for that section.

The simulation process deals with the probabilistic incident generation as a scheduled event for the sections. That means that at the beginning of each simulation step, at the same time that the simulator control module updates the other scheduled events, like those related to the traffic control signal changes, it will also check whether incidents will occur at the sections according the corresponding probability distribution. The fields that compose an Incident Event are: Time at which incident will take place, Duration of Incident, Number of lanes blocked by the incident and Length of incident.

Therefore, Traffic Incident Generation will follow an Event Scheduling simulation approach. At the beginning of simulation, the first Incident Event will be generated for every section in the network, according to the specified probability distribution.

Incidents are sorted in the simulation Event List by time of occurrence. At every simulation step, the Event List is checked to see whether or not a traffic incident is due to occur in the current

simulation step. If so, the incident is generated and the corresponding event is removed from the Event List. Then, next incident event is scheduled for that section, according to the incident generation parameters defined.

The incident is implemented in the simulation by the generation of a dummy vehicle, which is located at the incident position, and will be stopped there during the duration of the incident. Therefore, other vehicles will be affected by the incident by following normal vehicle behaviour models (car following and lane changing). To check whether an incident has finished it is done when updating a vehicle. If it is an incident dummy vehicle and the incident duration time has expired, the dummy vehicle may be removed from the network, so the blockage may be finished.

Emulation of Detector Measurements

The simulator produces detection output data periodically, provided that there are any detectors defined in the network. The data produced depends on the measuring capabilities of the detectors. It may be Count (number of vehicles per interval), Occupancy (percentage of time the detector is pressed) and Speed (mean speed for vehicles crossing the detector). These data may be stored into files or directly accessed by the Incident Management System, through the GETRAM Extension Module.

Modelling of Management Actions

The types of management actions that are modelled include modifications on the speed limits, recommendations of alternative routes or just information about the presence of an incident. As impact for these messages, the modeller can define the following actions:

- Modifications of the speed limit of any section. This is used to model both, the variable speed limit signs and the warnings for incidents or congestion ahead.
- Input flow modifications, which is only applicable to the input sections. The modeller can specify an increment or decrement (in percentage) in the flow rate.
- Re-routing actions. Depending on the type of simulation, based on turning proportions or in O/D matrices, they are turning proportions modifications for any section in the network, modification of next turning movement for drivers in certain section going to specific destinations, or modifications of destination centroids.

2.3.4 Outputs

If an animated graphical display is used to show the simulation progressing then it should be possible to view the following aspects related to the incident in the display:

- Incident or lane blockage
- Effect of the incident in the traffic behaviour
- Management actions decided by traffic management system
- Effect of the management actions in the traffic behaviour

On the other hand, the Statistical Output produced by AIMSUN2 may be used to check the presence of an incident and to assess the impact that the Management actions are having on the traffic behaviour. Among others, the mean flows, density, average speed, delay time, stop time and queue length corresponding to the sections involved in the incident can be used to assess the impact of the incident and the management policy.

2.3.5 Calibration Results

To check that the incident model is working as specified, scenarios are to be generated which allow the following checks to be made:

- Whether incidents, 1) are generated at the correct place and time, and 2) last for the expected time period.
- Whether drivers, 1) behave properly in the presence of the incident, and 2) react to the management actions.

The model can be validated using the following data, which is provided by AIMSUN2 as Statistical Output:

- reduction of capacity due to the incident
- queue length produced by incident
- new distribution of flows in the incident surroundings

The Barcelona Test-Site does not include any Incident detection and management system and for that reason no validation data is available for this function. Therefore, only verification tests are being made, using a simple example scenario.

The example model consists of a freeway section in which there are two off-ramps. Initially all vehicles are assigned to a destination which means going straight ahead on the freeway. Several detectors are located in the freeway in order to detect any possible incident. Two VMS are also located some distance upstream of the on-ramps. The purpose is to detect incidents downstream the freeway and to inform drivers to take some of the on-ramps to avoid the congestion. A simple Incident Detection algorithm has been implemented using the GETRAM Extensions Module. A couple of incidents have been generated in the freeway, detected by the Incident Detection module and messages are sent to the VMS routing vehicles through the off-ramps. Then the incidents disappear and the congestion is cleared up. The new situation is identified by the external module and new messages are displayed on the VMS in order to re-route vehicles again through the freeway.

2.4 ADAPTIVE TRAFFIC SIGNALS

2.4.1 Introduction

The approach used in AIMSUN2 to model Adaptive Traffic Signals is by means of the GETRAM Extensions Module, explained in section 2.2. In this way, AIMSUN2 traffic signals are adaptive if there is an Adaptive External Traffic Control System interfaced to AIMSUN2 that is running during the simulation run and it takes control of the signals.

For the intersection control, AIMSUN2 uses a phase-based approach in which the cycle of the junction is divided into phases where each one has a particular set of signal groups with right of way at the same time.

All the turning movements that are controlled by the same traffic signal and have right of way simultaneously can be grouped in one signal group. Then, a sequence of phases is defined for the whole junction. Each phase has a set of signal groups associated with it.

During the simulation of a scenario, AIMSUN2 executes a fixed control plan taking into account the phase modelling for each junction. However, this fixed control definition can be variable along the simulation period. The user may specify different fixed plans that will be activated during the simulation at a specified time.

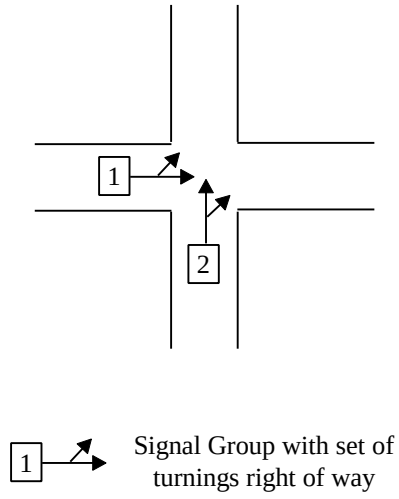


Figure 7: example of a simple junction, with signal groups.

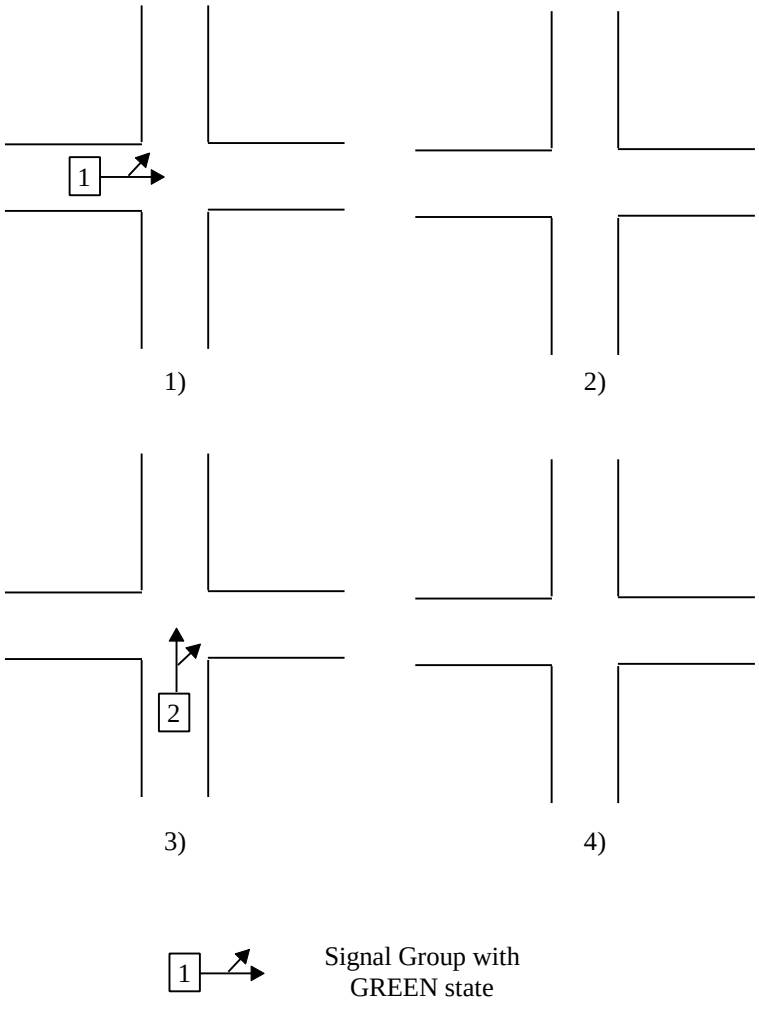


Figure 8: Example of rights of way sequence for the junction in figure 7.

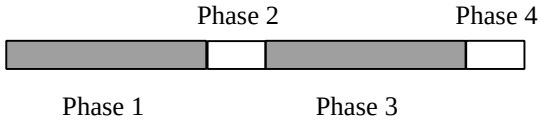


Figure 9: Phase modelling for the junction in figure 8.

On the other hand, the External Adaptive Traffic Control System can modify this execution by means of different actions, such as changing the duration of a phase or directly jumping from one phase to another. This is done through the GETRAM Extensions Module. The functions relative to traffic control in junctions are listed in section 2.2.3.

During simulation the traffic control plan structure cannot be modified (i.e. the definition of signal groups), but it is possible to change the allocation of signal groups to phases or the duration of any phase.

2.4.2 Inputs

The inputs for the Traffic Signal Control of a Junction are the Signal Groups definition and the Control data. Each Signal Group has set of turning movements (represented by the 'Section From' and 'Section To' identifiers) associated with it.

The Control data may be composed of a set of Control plans. For each Control Plan, the following information is needed:

- Control: name of the control plan.
- Interval: Starting time, i.e. time at which the control plan will become active.
- Yellow: duration of yellow (or amber) time, which will be used for all the traffic signals.
- Offset: time offset for the plan, i.e. the time origin for all time settings.

For each control plan there is the type of control (Uncontrolled, Fixed or Adaptive), the time offset (time at which the first phase of the junction starts) and the phase's information. For each phase there is the following data:

- Phase number (from 1 to n)
- Minimum, initial and maximum duration of the phase, in seconds. When simulating with fixed control all three values would be the same. When simulating with an External Adaptive Control these parameters provide the initial duration of each phase and the range of feasible variation for the phase duration.

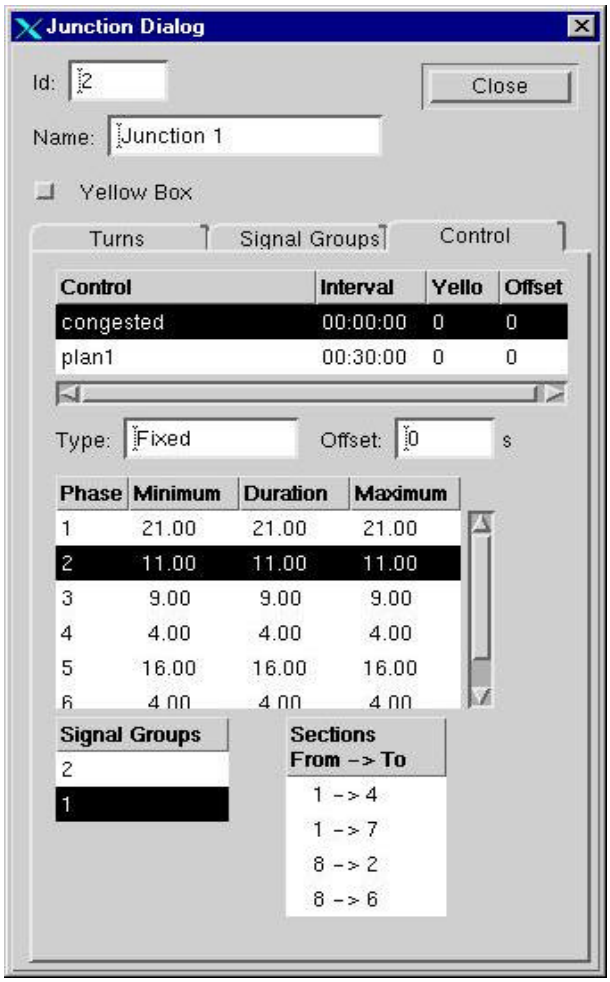


Figure 10: Junction-Control dialog

2.4.3 Processing

The Traffic Signal Control modelling is implemented using an Event Scheduling approach. At the beginning of the simulation the Control State is initialised for all signalised junctions and the first phase-changing events are scheduled.

During simulation, the control events list is revised at the beginning of every simulation step to check whether there is any change of phase due to occur during the current step in order to update the Control State.

Signal Control Initialisation

At beginning of simulation the following procedure is run:

Getram Extension Initialisation (adaptive signals)
For *each Signal Controlled Junction of the network* ***do***
Calculate the active phase (according to the Plan and Junction offsets)
Set Green State to Signal Groups belonging to the phase
Set Red State to other Signal Groups
Locate red-light dummy vehicles
Schedule next Event change-of-phase
Enddo

Signal Control Updating

At the beginning of every simulation step, the following procedure is run:

Getram Extension Manage (adaptive signals)
For *each change-of-phase Event occurred during last simulation step* ***do***
    For *each signal group belonging to the previous phase*
        and not belong to the new phase ***do***
        Set Red or Yellow State
        Locate red-light dummy vehicle
    Enddo
    For *each signal group belonging to the new phase*
        and not belong to the previous phase ***do***
        Set Green State
        Remove red-light dummy vehicle
    Enddo
    Schedule next Event change-of-phase
Enddo

2.4.4 Outputs

The outputs produced by the Traffic Control Model are the changing of the traffic signals from green to red and from red to amber or green, in accordance with the fixed or variable control plan and the external adaptive control system, if any. The user, through the AIMSUN2 Graphical Interface, can observe these changes.

By using the GETRAM Extension Module it is possible to have direct access to the Signals State, current phase and duration, and to any other traffic control data (signal groups, phases and timings).

The Statistical Output produced by AIMSUN2 may be used to obtain some measures of performance in the controlled junctions. For instance:

- for each controlled junction
  - for each input section of the junction
    - the mean flow that has crossed the junction
    - the mean speed observed on the section per time slice and over the simulation period
    - the mean delay time per vehicle per time slice and over the simulation period
    - the mean and maximum queue length value observed on the section per time slice and over the simulation period

2.4.5 Calibration results

To check that the adaptive signal model is working as specified, scenarios are to be generated which allow the following to be checked:

- Whether traffic controllers execute the phase sequence according to their specified default settings and,
- Whether they receive and react correctly to the impulse and plan type messages sent by the external strategy.

We do not consider here tests that are related to the external strategy itself or to the chosen communication process.

The Barcelona test-site does not include any Adaptive Traffic Control System and for that reason no validation data is available for this function. Therefore, only verification tests are being made, using a simple example scenario.

In this case the example model consists on an urban network composed by two signalised intersections. An external control system consisting on a simple adaptive traffic control policy that gives priority to public transport has been implemented through the GETRAM Extensions Module. A default fixed traffic control plan is used until the external system detects the arrival of a bus to the first junction. Then it changes to adaptive mode and priority is given to the bus, changing phases in both intersections until the bus has passed through the second one, at which moment the control is turned back to fixed mode.

2.5 RAMP METERING

2.5.1 Introduction

AIMSUN2 incorporates ramp-metering control. This type of control is used to limit the input flow to certain roads or freeways in order to maintain certain smooth traffic conditions. The objective is to make sure that the entrance demand never surpasses the capacity of the main road. Ramp metering objects are located at the downstream end of a section approaching a node type juncture and affect all the lanes of the section. Figure 11 shows a ramp-metering layout.

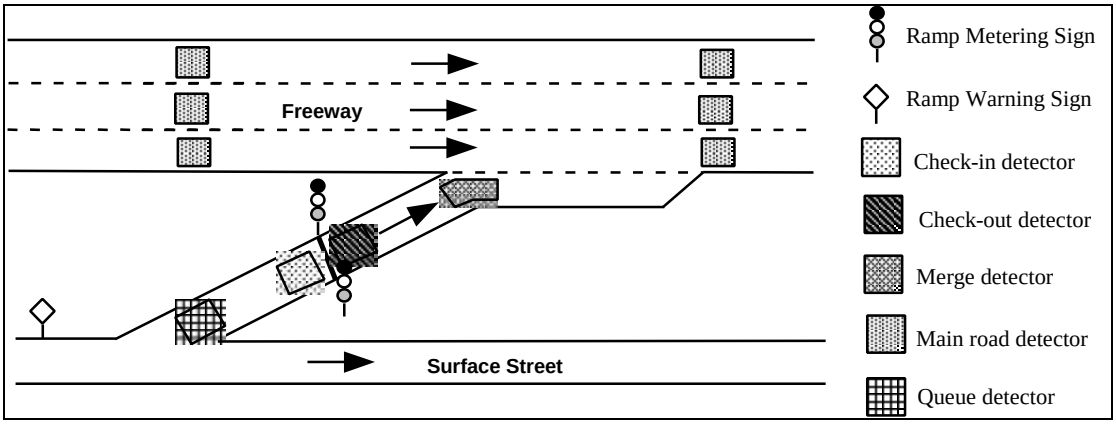


Figure 11: General layout of traffic-responsive entrance ramp metering system

AIMSUN2 considers three types of ramp metering depending on the implementation and the parameters that characterise it: green time metering and flow metering. Also there is the possibility of using the same Ramp metering model to emulate other types of access control in which the stopping time may be a given random distribution. This is delay metering.

Ramp metering objects may be located at any point of a section. Ramp metering control can be fixed, variable or adaptive. In the fixed control, the same control plan is used for the whole simulation period. In the variable control, a set of different control plans can be used at different times of simulation. Last, the adaptive control is achieved through the interfacing of AIMSUN2 to an external traffic control system. This is done through the GETRAM Extension Module, whose functions relative to meterings are listed in section 2.2.3.

Green Time Metering

Parameters are green time and cycle time. The ramp metering is modelled as a traffic signal that turns red and green on a cyclic basis. If it is a fixed traffic control, only a constant green time is used. In the case of simulation with some external Adaptive Traffic Control System, there would be a minimum and maximum value for the acceptable range of green time variation. The rest of the cycle time, the traffic signal will be red. Vehicles will stop at a red signal and cross at a green signal.

Flow Metering

Parameters are platoon length and flow (veh/h). The meter is automatically regulated in order to permit the entrance of certain maximum number of vehicles per hour. In this case the ramp-metering objective is to let a certain number of vehicles per hour to cross the meter. Each time the meter is opened to release vehicles, it is done in a way that platoons of a given length can pass. This can be done in two ways, either by counting the vehicles crossing the meter or by allocating a green time as a function of the platoon length. On average, a certain number of vehicles per hour will be released. In the case of simulation with some external Adaptive Traffic Control System, there would be minimum and maximum values for the acceptable range of flow variation.

Delay Metering

Parameters are the mean delay time and the standard deviation. This type of metering may be used to model the stop of vehicles due to some control facility, such as tolls, customs, checkpoints or any other type of individual control. It is assumed that every vehicle will have to stop at the control point (i.e. the ramp metering stop line) for a certain amount of time. This time is a random variable distributed according to a given probability distribution, e.g. a normal distribution with a given mean and standard deviation.

2.5.2 Inputs

Metering data

Metering data is input through the Tedi network editor just by clicking on the network display on a section at the desired position. Other ramp metering data is then entered through a dialog window, which appears when the ramp metering is selected

- Metering Identifier
- Link or section controlled by the metering
- Ramp metering sign location

Control data

- Type of Ramp Metering
- List of Control Plans. For each one:
  - Name or identifier of plan
  - Time at which control plan starts

  - Offset
- Control Mode (fixed or adaptive)
- For Green Time Metering
  - Cycle length time (fixed or minimum and maximum values)
  - Offset time
  - Green time (fixed or minimum and maximum values)
- For Flow Metering
  - Entrance flow (fixed or minimum and maximum values)
  - Platoon length
- For Delay Metering
  - Mean Delay Time
  - Standard Deviation

2.5.3 Processing

The ramp metering model is able to reproduce the metering control process, the behaviour of traffic at the presence of the ramp metering, and the vehicle detectors used (vehicle detection model is not described here).

The vehicle stop at the ramp metering line may be achieved by putting a dummy vehicle at the stop line which will be stopped while the ramp metering is closed and will be removed when it is opened.

The Metering Control modelling is implemented using an Event Scheduling approach. At beginning of simulation, the metering state is initialised for all controlled meterings and events corresponding to first changes of state are scheduled.

During simulation, the events list is revised at the beginning of every simulation step to check whether there is any change of state due to occur during the current step in order to update the metering state.

Control Metering Initialisation

At beginning of simulation the following procedure is run:

Getram Extension Initialisation (adaptive meterings)
For *each Flow Metering of the network* ***do***
    Calculate Cycle and Green times according to Flow and Platoon length
Enddo

For *each Green and Flow Metering of the network* ***do***
    Calculate the initial state ((taking into account the Plan and metering offsets)
    If *Initial State is Green* ***then***
        Set Green State to metering
    Else
        Set Red State to metering
        Locate red-light dummy vehicle
    Endif
    Schedule next Event change-of-state
Enddo

For *each Delay Metering of the network* ***do***
    Set Red State to metering
    Locate red-light dummy vehicle
Enddo

Control Metering Updating

At the beginning of every simulation step, the following procedure is run:

Getram Extension Manage (adaptive signals)
For *each change-of-state Event occurred during last simulation step* ***do***
    If *change-of-state is Green to Red* ***then***
        Set Red State to metering
        Locate red-light dummy vehicle
    Else
        Set Green State to metering
        Remove red-light dummy vehicle
    Endif
    Schedule next Event change-of-phase
Enddo

For *each Delay Metering* ***do***
    If *a vehicle has just arrived to stop line* ***then***
        Sample Delay Time
        Schedule next Event change-of-state
    Endif
Enddo

2.5.4 Outputs

The outputs produced by the Ramp Metering Control Model are the changing of the traffic signals from green to red and from red to green, in accordance with the fixed or variable control plan and the external adaptive control system, if any. The user, through the AIMSUN2 Graphical Interface, can observe these changes.

Using the GETRAM Extension Module it is possible to have direct access to the current ramp metering state, and to other traffic metering control data.

The Statistical Output produced by AIMSUN2 may be used to obtain some measures of performance in the metered sections. For instance:

- for each metered section
  - the mean number of vehicles arriving to the metering
  - the mean of vehicles crossing the metering
  - the mean speed observed on the section per time slice and over the simulation period
  - the mean delay time per vehicle per time slice and over the simulation period
  - the mean and maximum queue length value produced by the metering on the section per time slice and over the simulation period

2.5.5 Calibration results

To check that the ramp-metering model is working as specified, scenarios are to be generated which allow the following checks to be made:

Whether the ramp meter

- allows the entrance of the expected mean flow
- produces the correct platoon lengths
- is opened and closed in a cyclic way with the expected duration's

Whether drivers:

- behave properly in the presence of the ramp metering
- reduce speed and stop when ramp meter is closed
- accelerate and cross the meter when it is opened

The model has to be validated using the following data:

- capacity of the ramp metering according to the control policy and the traffic demand
- queue length produced by ramp metering
- average waiting time at ramp metering

The Barcelona Ring-Roads test-site model includes Ramp Metering Control in the area known as Ronda de Dalt. However, due to unexpected circumstances, the Municipality of Barcelona has not been able to provide the required data for calibration purposes yet.

An example model for verification purposes has also been developed. It consists of a two-lane freeway section in which there is a one-lane on-ramp with a ramp meter. A simple ramp metering policy has been implemented using GETRAM Extensions. It uses detection data coming from three simulated detectors. Two are located in the freeway, upstream and downstream the on-ramp and a third one is located at the on-ramp. The ramp metering policy tries to ensure that the flow downstream on the freeway increases no more than a safety maximum flow by controlling the on-ramp entrance flow.

2.6 VARIABLE MESSAGE SIGNS

2.6.1 Introduction

Information to drivers is considered as a possible result of the actuation of a Traffic Management System on a network containing Variable Message Signs (VMS) equipment. Messages may inform the drivers about the presence of incidents, congestion or suggest alternative routes. They can even be used to make some prohibitions. AIMSUN2 takes into account the modelling of VMS and their influence on the driver's behaviour.

Each VMS has a set of acceptable messages, and each message has a list of Actions associated with it, which represent the influence the message has on the driver's behaviour. Upon activating a message, the associated actions are implemented. The types of message that can be modelled include modifications to the speed limits, recommendations of alternative routes and information on congestion or incidents.

An Action represents the impact that a message has on the driver's behaviour. Different types of actions are considered depending on whether the simulation is run using the Traffic Result option (Input flows and turning proportion) or the Route Based O/D Matrix simulation mode.

A Traffic Management System that displays messages on the Variable Message Signs can be interfaced to AIMSUN2 through the GETRAM Extensions Module. The functions related to VMS are listed in section 2.2.3.

Actions for a Traffic Result based Simulation

When the simulation is done using the Traffic Result option (Input flows and turning proportion) three types of actions can be defined: modifications of the speed limit, modifications of the input flow and modifications of the turning proportions.

1. *Modifications of the Speed Limit*: a new speed limit for a set of sections can be defined.

2. *Modification of the Input Flow*: an increment or decrement of the input flow can be defined as a percentage of the current flow. Input flow modifications may only affect to input sections, where traffic is generated and injected into the network.
3. *Modifications of the Turning Proportions*: the user can define an increment or decrement to the proportion of vehicles that having entered a section through an entrance will follow a turning. This is defined as a percentage of increment or reduction over the current turning probability.

Actions for a Route Based Simulation

When the simulation is Route based (using OD matrices and route choice models) two types of actions can be defined: modifications of the speed limit and Re-routing actions, which can be either modifications to the destination centroid or modifications to the next turn to make.

1. *Modifications of the Speed Limit*: works in the same way as in the Result Based mode.
2. *Re-routing Actions*: Re-routing means the possibility of altering the vehicle's path. This effect is accomplished by defining the next turn and/or defining a new destination.
   - The first type of re-routing action is the modification of the destination centroid. The user may define a set of pairs composed of the previous destination centroid and the new destination centroid.
   - The second type of re-routing action is the modification of the next turning. The user may choose among All or Selected Destinations.

   The re-routing effect can be defined by each vehicle type independently or be the same for all vehicle types. There is a *Compliance* parameter (δ) which gives the compliance level of the action, i.e. the percentage of vehicles accepting the recommendation. It can be Compulsory, Warning or Information. Compulsory means $\delta=1$, which implies that the re-routing will be followed by everybody (i.e. an obligation). Information means $\delta=0$, where the action's success will depend on the driver's behaviour (Guidance acceptance λ, a vehicle attribute). In the Warning option the user may define δ ($0<\delta<1$), which is the level of acceptance, i.e. is an advice.

2.6.2 Inputs

The required input data for each VMS is the following:

- VMS name (a string of characters).
- Position where it is located in the section. It is measured as the distance from the entrance point of the section to the VMS.
- List of feasible messages for this VMS. Only messages included in this list may be accepted for displaying by this VMS.
- List of all Actions available for this network.

2.6.3 Processing

VMS Initialisation

At beginning of simulation the following procedure is run:

For *each VMS of the network* ***do***
    Read Initial Message of VMS
    Implement Actions associated to the message
Enddo

Getram Extension Initialisation (externally setting the initial state)

VMS Updating

During simulation new messages can be displayed on the VMS's. They may be activated in two different ways:

1. Directly through the user interface, any message from the messages list box can be activated.
2. Through the GETRAM Extension Module, any external system may activate a message on any VMS of the network, by sending the corresponding command, which consists of the VMS identifier and the message text.

In both cases, it will cause the message to be displayed and the associated actions to be implemented.

Actions Implementation

1. *Modifications of the Speed Limit*: When a message containing this action is displayed in a VMS, it will automatically modify the Speed limit for the selected sections and therefore all vehicles driving along these sections will be affected.
2. *Modification of the Input Flow*: the input is increased or decreased by the corresponding percentage.
3. *Modifications of the Turning Proportions:* When an increment or decrement of a turning proportion is defined, the original percentage is changed and it will be taken out from, or added to, the other turns of the section for which increments or decrement are defined. Therefore when an increment is defined for some turn(s), it is necessary to define a corresponding decrement for some other turn(s) of the same section. The procedure is as follows:
    - Calculate the modifications of the percentages in all the affected turns.
    - Adjust the resulting percentages of the affected turns in order to keep the same original sum of proportions.
4. *Re-routing Actions*:
    - Modification of the destination centroid: Vehicles going to Previous destination will change to New destination as soon as they enter any of the sections affected by this action, taking into account the compliance level.
    - Modification of the next turning: All destinations means that all vehicles entering the affected sections will take as their next turning the one specified, regardless of their destination. Selected destinations means that only vehicles going to those destination will be affected by the new turn. In both situations the compliance level is taken into account and it may cause the original destination to be lost if there is no path to it via the specified next turn.

2.6.4 Outputs

The outputs produced by the Variable Message Sign Model are the activation of the different messages, in accordance with the user's declarations, performed using either the AIMSUN2 Graphical User Interface or through the GETRAM Extension Module.

It is also possible to have direct access to the Current State of any VMS during simulation using the GETRAM Extension.

The Statistical Output produced by AIMSUN2 may be used to obtain some measures of the influence of displayed messages on the driver's behaviour. For instance:

- Network data
  - travel time
  - average speed on whole network
  - total distance travelled
  - total delay
- section data:
  - travel time (mean value and standard deviation)
  - average speed
  - number of vehicle entered during the simulation
  - number of vehicle exited during the simulation
  - number of stopped vehicles
  - average delay
  - average occupancy (average number of vehicles / carriageway capacity)
- VMS Section data:
  - number of vehicles which crossed the link
  - number of vehicles which received VMS suggestions (suggestions concern vehicle destinations)
  - number of vehicles which followed the recommendations (according to driver behaviour model)
  - number of vehicles that modified their route according to VMS suggestions

Finally, the animated graphical display is used to show the simulation progress, and the following information is included in the displays:

- On the VMS panel
  - VMS identifier
  - VMS Status and Message
  - Actions related to the messages

2.6.5 Calibration Results

Although the Barcelona Ring-Roads test-site model includes Variable Message Signs, no data are to be provided by the Municipality of Barcelona, as they do not appear to be gathering this kind of information. Therefore only a verification of this model seems feasible.

The same model presented in section 2.3.5 used for the Incident Detection model verification has been used for the Variable Messages Sign verification.

2.7 DYNAMIC ROUTE GUIDANCE

2.7.1 Introduction

We consider here Individual Route Guidance as a function of Traffic Management, whose purpose is to operate on the individual basis, guiding a specific subset of vehicles, that are supposed to be equipped, towards their destinations.

Route Guidance is only implemented in AIMSUN2 whenever the simulation is based on O/D matrices and shortest paths, which is called the Route Based simulation model. In this model, vehicles are fed into the network according to the demand data defined by an O/D matrix and they drive along the network following a given path in order to reach their destination.

2.7.2 Inputs

The input data required for this model is the following:

Shortest Routes Calculation

- origin and destination centroids
- connections between centroids and network sections or nodes
- traffic demand: O/D matrix
- speed limit for every section
- turning speed for every turning
- capacity for every section
- capacity weight, a parameter to control the influence that the section capacity has in the cost in relation with the travel time
- time interval for recalculation of shortest paths
- estimation of travel times for every interval for all sections and turnings

Route Choice

- route choice model
- parameters of the route choice model:
  - k, number of different alternative paths to keep
  - Binomial: p, the probability of success
  - Logit: θ, shape or scale factor parameter
- percentage of guided vehicles for each vehicle type
- compliance rate (%,defined per vehicle type)
- vehicle destination centroid

2.7.3 Processing

Shortest Routes Component

During the simulation, the computation of shortest routes is determineded at certain time steps. This is usually in a periodic manner, with a period that depends on the length of the section and on the level of congestion.

The simulator needs to store shortest routes from the beginning of every section to all destinations for each vehicle type at each time interval. One needs to keep all previously generated routes as long as there are vehicles using them. For each destination and instant in time, the routes are stored as a tree that makes it possible to determine how to reach the destination from any section of the network. We also attach to this tree a field that counts the number of vehicles using it. When this counter is empty, the tree may be deleted.

The procedure that we use to compute the shortest routes to a destination (either a centroid node or a section) uses a network where an arc, connecting two nodes, models a section. A special arc connecting the beginning of the turning to its end models a turning movement. The computation of shortest routes uses a label setting method, where the labels are associated with an arc. The network is constructed only once, before the start of the simulation.

The shortest route routine is a variation of Dijkstra's label setting algorithm. It gives the shortest routes from the start of every section to all destinations. The cost labels are attached to sections instead of nodes, as is usual. The arc candidate list is stored as a heap data structure. During each iteration of the algorithm, the section with minimum value is removed from the heap and the heap is restored by using efficient operations. As a new section is reached, one adds it to the heap in the correct position.

Cost Functions

Two types of section cost functions are used for calculating the shortest path trees, depending on whether or not there are simulated data available to be used for. These are the Initial Cost Function and the Current Cost Function. In both cases, the cost function represents section travel time in seconds, including the penalty of the turning movement, if it exists.

The *Initial Cost Function* is applied at the beginning of the simulation when there is no simulated data gathered to calculate the travel times. In this case, the cost of each section is calculated as a function of the travel time in free flow conditions and the capacity of the section.

The initial cost of each section, *IniCost(s),* is calculated as follows:

$$IniCost(s) = TravelTFF(s) + TravelTFF(s) \times j \times \left(1 - \frac{Capacity(s)}{MaxCapacity}\right)$$

where:

TravelTFF(s) is the travel time, in seconds, of section *s* in free flow conditions. It is calculated as *Length(s)/SpeedLimit(s)*.
Capacity(s) is the capacity of section *s*, in vehicles per hour.
MaxCapacity is the maximum capacity of any section in the network.
j : Capacity weight. It is a user-defined parameter that allows the user to control the influence that the section capacity has in the cost in relation with the travel time.

The *Current Cost Function* can only be applied when there is some simulated travel time data available, and therefore it cannot be used at the beginning of the simulation but only when the simulation has already started and some statistical data has been gathered.

The current cost for each section, *CurrCost(s),* is the mean travel time, in seconds, for all simulated vehicles that have crossed the section during the last statistical gathering period (*TravelTime(s)).* As there may be situations in which any vehicle has not crossed a section, the following algorithm is applied to calculate *CurrCost(s):*

```
if (Flow(s) > 0) then
        CurrCost(s) = TravelTime(s)
else
        if (there is any vehicle stopped) then
                CurrCost(s) = Maximum (AvgTimeIn, IniCost(s))
        else
                CurrCost(s) = IniCost(s)
        endif
endif
```

According to this algorithm, when some vehicle has crossed the section during the last statistical period (*Flow(s)* > 0), the cost is the simulated mean travel time. In the case that no vehicle has crossed the section we distinguish the case of a totally congested section from the case of an empty section. In the first case, the cost is calculated as the maximum between the Initial Cost and the average waiting time for the vehicles in front of the queue in the section (*AvgTimeIn*). In the second case, the cost is taken as the initial cost.

Fixed Routes Mode

In the Fixed Routes Mode, shortest path trees are calculated from every section to every destination centroid at the beginning of the simulation. Then, during the simulation, vehicles are generated at origin centroids and assigned to the shortest route to their destination centroid. There is no need for a Route Choice Model as there are no alternative routes. No new routes are

recomputed during simulation; therefore all vehicles always follow the shortest path and no decisions about changing to another path can be made during the trip.

Variable Routes Mode

In the Variable Routes Mode the simulation process includes an initial computation of shortest routes going from every section to every destination, a shortest route component which calculates periodically the new shortest routes according to the new travel times provided by the simulator, and a route selection model.

The simulation procedure can be characterised as follows:

1. Calculate initial shortest routes, taking as costs the estimated travel times for each section (i.e. length of section / speed limit).
2. Simulate for a period (e.g. 5 minutes) using available routes information and obtain new average travel times as a result of the simulation.
3. Recalculate shortest routes, taking into account the new travel times.
4. Add the new information calculated in 3 to the knowledge of the drivers.
5. Go to step 2.

At the beginning of the simulation, shortest path trees are calculated from every section to each destination centroid, taking as section costs the Initial Cost Function. During simulation, new routes are recomputed every time interval taking as section costs the simulated travel times obtained for each section during the last interval, this is the Current Cost Function explained before. Figure 12 illustrates when are the shortest paths (SP) calculated along the simulation period and what cost functions are used.

The user may define the time interval for recalculation of paths and the maximum number of path trees that they wish to maintain during the simulation. When the maximum number of path trees (K) is reached, the oldest paths will be removed as soon as no vehicle is following them. It is assumed that vehicles only choose among the most recent K path trees, therefore, the oldest ones will become obsolete and disused.

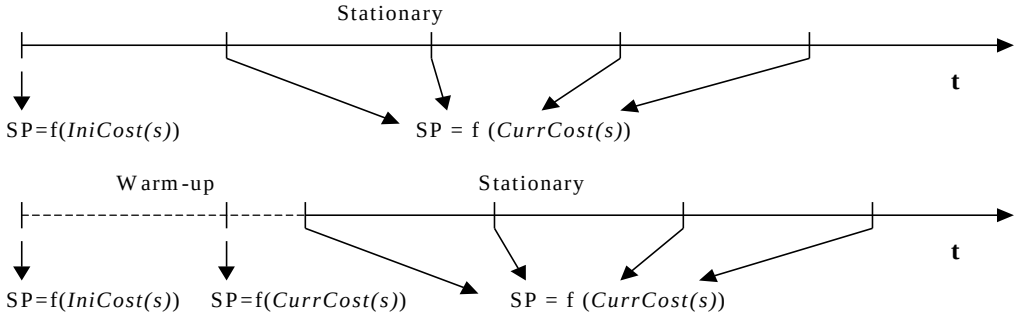


Figure 12: Calculation of Shortest Paths in a Variable Routes Mode

Static versus Dynamic Route assignment Models

Vehicles are initially assigned to a route from a set of available routes on a probabilistic way. Apart from the initial assignment of route, which is made at the vehicle's depart time, there is the possibility of making a route reassignment during the trip. This is called the Dynamic route choice model, as opposite to the Static one.

In the Dynamic route choice model a guided vehicle can make a new decision about what route to follow at any time along their trip, whenever there are new shortest routes available. In the Static model, a vehicle will always follow its initially selected route until reaching the destination, although new shortest route could be available during the trip. Note that in the

Dynamic model only guided vehicles can make a decision to change to a new shortest route during the trip, as it is supposed that information is only available for equipped vehicles. Regarding this, there is a parameter for each vehicle type that gives the percentage of guided vehicles.

The behaviour of the driver in response to information acquisition may be modelled in different ways using any of the following route choice models.

Route Choice Models

Currently there are two Choice Models implemented, which are used either when assigning the initial path for a vehicle at the beginning of its trip or when having to decide whether or not to change path en-route in the dynamic modelling.

Binomial Model

A Binomial (k-1, p) distribution is taken to find the probability of selecting each path. Parameter k is the number of available paths and p is the "success" probability. This model does not consider the travel costs in the decision process, but only the time at which the path was calculated. Selecting a small p will mean that oldest paths will be more likely used while selecting high values of p, the most recent paths will be more frequently taken.

Multinomial Logit Model

We assume that the utility U_k^{rs} of route k between origin r and destination s is given by:

$$U_k^{rs} = -qt_k^{rs} + e_k^{rs}$$

Where:

q is a shape or scale factor parameter

t_k^{rs} is the expected travel time on route k from r to s, calculated as the sum of the current costs of all the sections composing the path (CurrCost(s) function as explained above), and

e_k^{rs} is a random term

The underlying modelling hypothesis is that random terms e_k^{rs} are independent identically distributed GUMBEL variates. Under these conditions the probability of choosing route k amongst all alternative routes from r to s is given by the logistic distribution:

$$P_k^{rs} = \frac{e^{-qt_k^{rs}}}{\sum_l e^{-qt_l^{rs}}} = \frac{1}{1+\sum_{l\neq k} e^{-q(t_l^{rs}-t_k^{rs})}}$$

The scale factor q plays a twofold role making independent of the measurement units the decision based on differences between utilities, and influencing the standard error of the distribution of expected travel times:

$$Var(t_k^{rs}) = \frac{p^2}{6q^2}$$

that is:

$q < 1$ high perception of the variance, in other words a trend to utilise many alternative routes

$q > 1$ alternative choices are concentrated in very few routes

The parameter, or scale factor q in AIMSUN2 is a user defined parameter that can be used to adjust the effect that small changes in the travel times may have on the driver's decisions.

2.7.4 Outputs

The outputs produced by the Route Guidance Model are the following:

- for every time interval of recalculation (see Figure 13)
  - k alternative shortest path from every section to every destination centroid
  - estimated cost (travel time) for every path

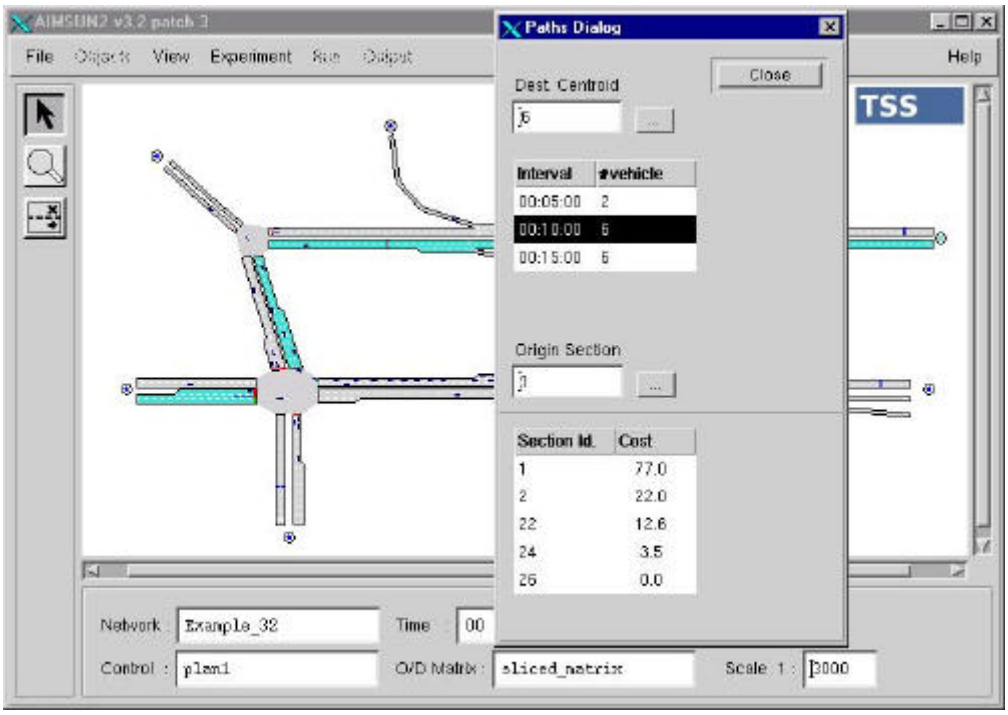


Figure 13: Paths Dialogue window

- for every origin, destination or OD pair
  - traffic flow
  - mean speed
  - travel and delay time
  - stop time and number of stops
  - distance travelled

2.7.5 Calibration Results

To check that the dynamic route guidance model is working as specified, two alternative scenarios of the Barcelona model are to be generated:

- Basic scenario (without Route Guidance)
- Basic scenario with Individual Route Guidance System

Different alternatives can be considered in the second scenario varying the value of percentage of equipped vehicles and the compliance rates.

The influence of the route guidance model can be assessed using the following data, which is provided by AIMSUN2 as Statistical Output:

- number of vehicles per origin-destination pair
- travel time for each O-D pair

2.8 RESULTS ANALYSIS TOOL

2.8.1 Introduction

A simulation model does not provide a unique solution to a given problem, it just tries to emulate the behaviour of a complex system in which randomness is involved. Each run of a simulation program, called a replication, produces a possible behaviour of the modelled system, which is a point in a sample of feasible results of the model. The final result is obtained through the statistical processing of the simulation results coming from different replications. Therefore, a simulation study requires the run of a number of replications of the same model, using different random seeds.

For that purpose, a more flexible mechanism for storing simulation outputs has been included. The user may decide to store the simulation outputs (statistics and detection) either as ASCII files or as a database, the latter using an ODBC format. In both cases the user may select where to locate these data, which makes it possible to store the results of different runs of the same model.

The idea of Experiment has been included in AIMSUN2. An experiment consists of a set of replications of the same scenario, composed of the traffic network, traffic demand, traffic control plan and a set of global modelling parameters. The user can define the number of replications to perform and the seed for each replication. Then the whole experiment can be run in Batch mode and the results of each replication are stored.

A graphical representation of simulation results is provided. Through the AIMSUN2 graphical interface the user can get time plots of different traffic variables, and also colour the network with a range of colours representing different values of the traffic parameters.

Apart from the experiment definition and storing module, the Results Analysis Tool can be completed with the addition of two further components:

- Statistical tools for result analysis: mean and variance estimation, calculation of confidence intervals.
- Statistical tools for model validation: hypothesis test, regression analysis.

2.8.2 Inputs

Experimentation Module

The required input data for the experimentation module is the following:

- Scenario to simulate:
  - traffic network
  - traffic demand data (either traffic flows and turning proportions or OD matrix)
  - traffic control plan
- a set of global modelling parameters (simulation step, reaction time, car-following and lane-changing parameters)
- Number of replications to run
- For each replication, random seed to use
- Location where to store simulation results

2.8.3 Processing

Experimentation Module

To run an experiment, the following procedure is applied:

Load scenario (traffic network, demand data and control plan)

Load or define modelling parameters
Define experiment (number of replications and random seed)
Define Output location (a database or a directory)
for *each replication* ***do***
Run simulation (in Batch mode)
Store simulation output
endfor

2.8.4 Outputs

Experimentation Module

The outputs provided by this module are both the simulation statistical output data and the simulated detection data. This data can be stored either in a database or in ASCII files. In the first case, the different results from each replication are stored using different primary keys, while in the second case the results from each replication are stored in different subdirectories.

The main statistics provided by AIMSUN2 for each replication is the following:

- Mean Flow
- Density
- Mean Speed
- Travel Time and Delay Time
- Stop Time and Number of Stops
- Mean and Maximum Queue Length
- Total Travel
- Fuel Consumed and Pollution Emitted

These data can be shown as time plots (see Figure 14) or displayed on the network as a range of colours representing different values.

Figure 14: Statistics Graphics. Time Series Plot. Flow, Density, Speed and Delay Time

The detection data provided for each replication is the following:

- Count: number of vehicles that have passed through the detector during the interval (vehicles)
- Occupancy: percentage of time that the detector has been pressed during the interval (%)
- Speed: mean speed of the vehicles when crossing the detector (kilometres/hour)
- Density: calculated using the measured count and speed (vehicles/kilometres).

2.8.5 Calibration Results

As the Experimentation Module does not involve a new model, it only needs a verification process rather than a proper calibration. By verification we mean that we check whether the module is performing as expected according to the program design.

Therefore it consists of verifying that the output data of each replication run is produced and stored properly, either in the database or in ASCII files and that the graphical representation of data works correctly.

For that purpose a test set has been used on the experimentation module with satisfactory results.

3 DRACULA

3.1 INTRODUCTION

In order to comply with the Model Update Specifications proposed in SMARTEST Deliverable 4, the following five models have been improved within DRACULA:

- Roundabouts,
- PT Services,
- Adaptive Traffic Signals,
- PT Priority,
- Detectors

and one new model has been added:

- Traffic Calming.

Improved validation of the model has also taken place. Improvements in the PT services model include a new bus stop model and the development of guided bus and tram operations. New roundabout and traffic calming models have also been developed, which have been calibrated and validated using data collected in Leeds.

The adaptive traffic signals improvements concentrated on linking DRACULA to a BALANCE UTC system that is due to be installed in Leeds and Sheffield. The installed BALANCE system will use the TCP/IP communications protocol to link up its various components. With this in mind a DRACULA interface that also uses TCP/IP has been developed. The improvements in the detector model in DRACULA concentrated on providing the BALANCE system with the on-street information it required. As well as the usual loop detector data this also included both public transport and emergency vehicle location information.

3.2 ROUNDABOUTS

3.2.1 Introduction

DRACULA makes the following assumptions about each roundabout:

- the roundabout is circular,
- the roundabout is modelled as a continuous link, with a given number of lanes, with entry and exit points at positions along it,
- vehicles attempt to travel at a desired circulation speed when on the roundabout,
- the usual car following rule is used on the roundabout,
- a new lane changing rule is used on the roundabout.

3.2.2 Inputs

The *roundabout diameter* D (defined as the diameter of the circle round the middle of the roundabout, this is equivalent to (IR+ICR) in Figure 15, where IR is the radius of the central island and ICR is the outer radius of the roundabout).

The desired circulation speed of vehicles travelling around the roundabout (A default value of 7.5 m/s is suggested). When each vehicle is generated it is given a scale factor to use when calculating its desired speed on a link. This scale factor is based on a random selection from a distribution with a given mean and variance. Each vehicle type has a given mean and variance to use.

Number of lanes on the roundabout.

Safe gap time for entering roundabout (a default value of 3.0s is suggested). This is the minimum size of gap which must exist before a vehicle will move on to the roundabout. For each vehicle type this can be modified by a scale factor. As with the desired speed this scale factor is based on a random selection from a distribution with a given mean and variance according to the vehicle type.

Which lanes on each input arm should be used for each exit from the roundabout.

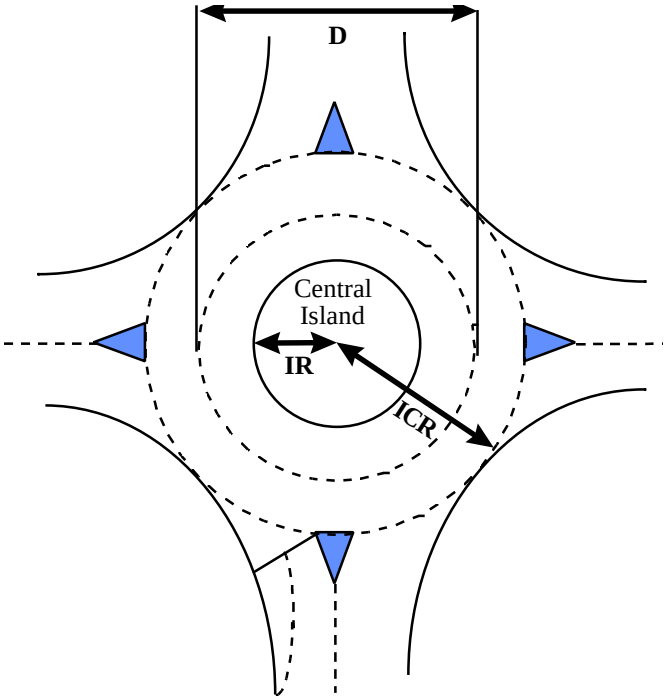


Figure 15: Roundabout geometry (UK)

3.2.3 Processing

The roundabout model uses three regimes. Firstly on approaching the roundabout vehicles have to get into an appropriate lane. When vehicles arrive at the roundabout they have to determine whether there is a suitable gap to allow them to enter the roundabout. Finally, when travelling on the roundabout vehicles have to choose an appropriate lane to allow them to leave the roundabout at the desired exit. The rules used for each of these regimes are given below in pseudo code.

Approaching roundabout

- Choose lane appropriate to exit
- Reduce speed as if approaching give way junction

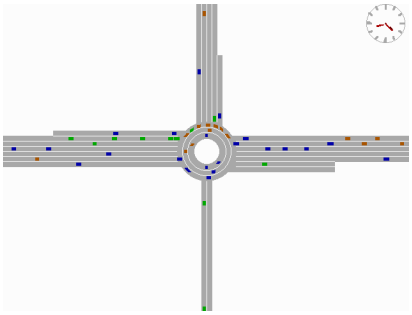
Gap search - to move on to roundabout

loop through vehicles on roundabout approaching from right (UK)
    if vehicle NOT exiting at this exit *then*
        calculate time to arrival at entrance
        if time to arrival > safe gap time *then*
            move on to roundabout in exit lane
        end if
    end if
until vehicle found OR distance scanned>50m

Movement on roundabout

- Use usual car following rule
- Use new roundabout lane changing rule
    if next exit is desired exit and not in outer lane *then*
        move to outer lane
    end if
    if lane ahead blocked by queue from non-desired exit *then*
        move into non-blocked lane
    end if

3.2.4 Outputs



The main additional output is a graphical representation of the roundabout as the program runs. (See Figure 16)

Figure 16: Roundabout output

3.2.5 Calibration Results

Not yet available.

3.3 PUBLIC TRANSPORT SERVICES

3.3.1 Introduction

The main improvements to the Public Transport service models within DRACULA have been:

- a new public transport service model
- a new public transport stop model
- a new reserved lane model
- guided bus operation

3.3.2 Inputs

Public transport services

The public transport services in the model are described by their:

- service number;
- vehicle type (bus or guided bus);
- service frequency (veh/hr);
- service route which includes:
    - Origin zone number
    - Destination zone number
    - Number of nodes (excluding the zones) en route
    - List of nodes en route
    - Number of public transport stops en route
    - List of public transport-stop identification numbers for all the stops en-route.

Only the departure time of the service (via a fixed hourly service frequency) is modelled. The bus schedule (in terms of route timing points) is not represented in the current version.

Public transport stops

A public transport stop might be used by all or just some of the public transport services (or routes) that go past it. Therefore a method of defining which services use which stops has been devised.

Two types of public service stops are modelled: ordinary bus stops or bus laybys. An ordinary bus-stop is a single sign on the road side and buses stop alongside the sign on the road to pick up or put down passengers, thereby blocking upstream traffic in that lane. A bus-layby, however, provides a space for the bus to pull into at the bus-stop and thus allows following traffic to pass. A bus-layby in the model is represented as a special type of bus-stop.

Each bus stop is defined by the following data:

- A unique public transport stop identifying number
- Upstream node of the link on which the stop lies
- Downstream node of the link on which the stop lies
- Position of the stop, measured from the entry of the link
- Side of the road where the stop lies, 0=curb side, 1=median side
- Type of the stop, 0=ordinary, 1=layby
- Layby length (metre), 0 for ordinary bus stop
- Average pedestrian flow (ped/hr) to the bus stop

Reserved public transport lane

The lane reservation in the model is specified by:

- the link identification;
- location on the link (near or offside lane);
- type(s) of vehicles reserved for the lane;
- "set backs" at the beginning and end of the link;
- start and end time of the reservation, measured from the start of the simulation.

Guided bus operation

Guided buses are represented in the model as a distinct type of vehicle. The distinction is made both in terms of vehicle characteristics and the traffic regulations governing their movement on the streets.

The guideway is represented as separate links, dedicated to the guided-bus vehicle type. The guideway can only be entered at the start of the link and exited at the end, unlike reserved lanes (see above).

3.3.3 Processing

Public transport service

The public transport vehicles enter the network at regular service frequency. They follow the pre-defined fixed route through the network as other traffic except when using reserved public transport lanes where provided, stopping at public transport stops for the service.

Public transport stops

There are two elements required to model public transport vehicle motion in the vicinity of public transport stops. Firstly, when approaching the stop, public transport vehicles need to move into the lane where they can access the stop. The public transport vehicles begin to attempt this manoeuvre in the link before the link with the stop. Secondly, if there are passengers waiting at the stop then the public transport vehicle has to stop at the stop for sufficient time to pick up all the passengers. Pseudo code for both these elements is given below.

PT vehicle motion approaching stop

if next link has a stop on it *then*
    try to move into lane which leads to lane with stop
    if fails *then*
        move into link
        look for gaps to move into lane with stop
    end if
end if

PT stop dwell time

if a bus is *NOT* already waiting at a bus stop *then*
    if number of passengers > 0 *then*
        stop time = a N + b
    end if
end if

where
    N = number of passengers waiting (passengers arrive at the stop at a fixed rate as given in the input section. The stop time is extended if more passengers arrive during the stop.
    a = time taken for 1 passenger to board (4s default).
    b = time for doors to open and close (5s default).

Reserved public transport lane

The following pseudo code describes the movement of public transport vehicles as they approach and move into a reserved lane.

if a reserved public transport lane is in the next link *then*
    try to change to the lane in the current link which leads to the reserved lane
    if failed to change lane *then*
        stay in lane until the next link
    end if
end if

Once in the link with the reserved lane the following logic applies

if there is a reserved public transport lane in the current link *then*
    try and move into the reserved lane
end if
if the reserved lane permits the public transport vehicle's next turn *then*
    stay in the reserved lane
else
    move off the reserved lane into a lane that allows the turn, when near the junction
end if

Guided bus operation

The operational distinction between a guideway and a reserved lane which this implementation incorporates is that a bus may join the guideway only at dedicated points on the route whilst a bus may "drift" into and out of a reserved lane anywhere along its extent. A lane can be specified as reserved for one particular type of vehicle or a combination of vehicle types.

3.3.4 Outputs

The additional outputs include the public transport service route specified summary statistics, which include the total travel time, distance, average speed, fuel consumption, pollutant

emissions for the service route. As the user's request, each public transport service vehicle's link-by-link travel times are also output.

3.3.5 Calibration Results

A test network in Leeds has been used to calibrate and validate the bus stop model.

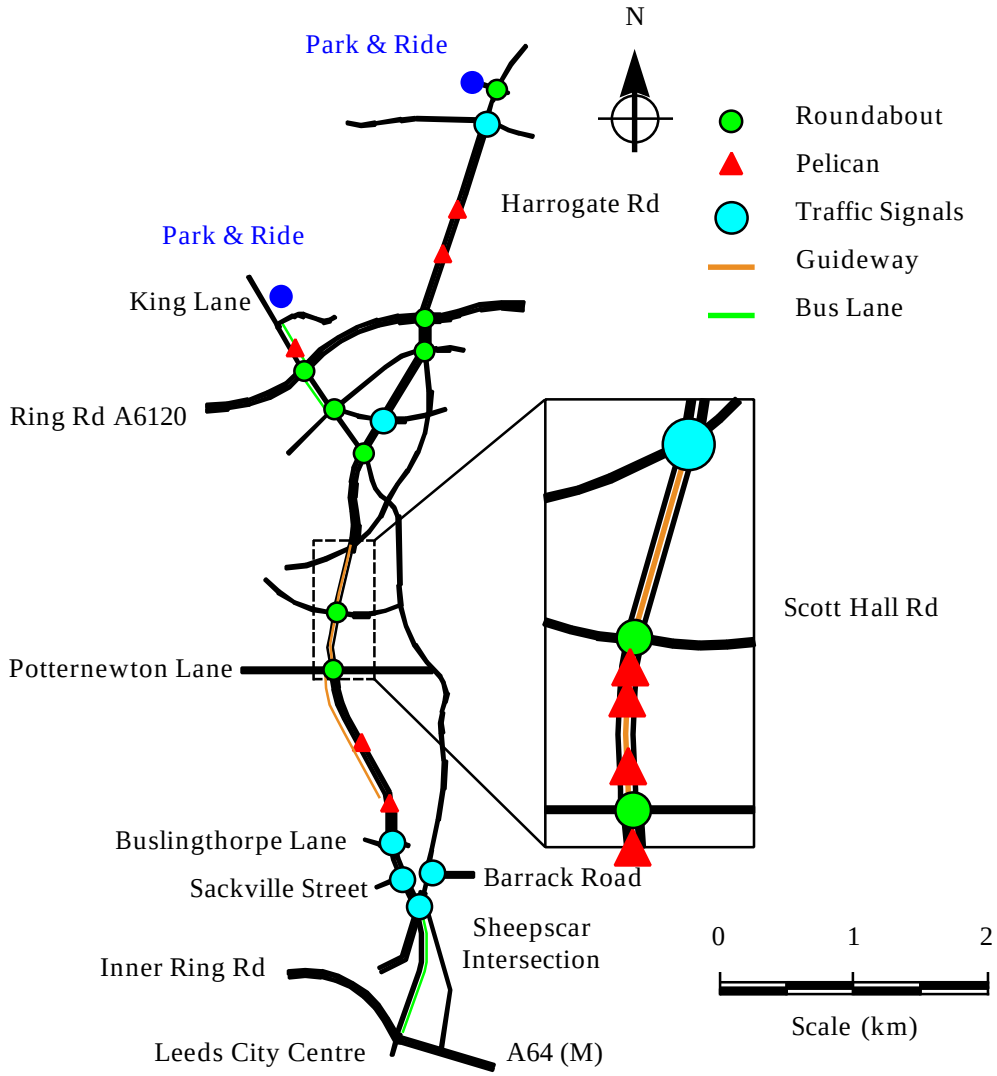


Figure 17: The Leeds test network

Data has been collected, using moving observers, on the journey times of buses between five bus stops and dwell times of buses at these stops for buses travelling down the Scott Hall Road between Potternewton Lane and Sackville Street during the morning peak period (See Figure 17). A summary of this data is presented in Table 1 and Table 2. The mean value, the number of observations (N) and the standard deviations (s.d.) are given. The upper and lower limits of the confidence interval, at the 95% confidence level, between which it is expected that the mean value will lie are also given in the tables.

Stops	Mean (s)	N	s.d.	Lower (s)	Upper (s)	DRACULA (s)
1-2	21.88	33	4.285	20.42	23.34	24.9
2-3	36.31	16	5.654	33.54	39.08	38.9
3-4	40.87	30	15.900	35.18	46.56	34.6
4-5	39.92	49	12.670	36.37	43.47	44.4

Table 1: Bus journey times between stops during the morning peak (08:00-09:00)

Stop ID	Mean (s)	N	DRACULA (s)
1	25.9	31	25.7
2	19.8	12	19.7
3	38.4	25	40.7
4	22.6	21	23.9
5	11.6	23	13.5

Table 2: Dwell times at stops during the morning peak (08:00-09:00)

The final column in the tables shows the value output from DRACULA for these times. Mean values from five simulation runs were calculated. As can be seen there is good agreement between the observed and the modelled journey times and bus stop dwell times.

3.4 ADAPTIVE TRAFFIC SIGNALS

3.4.1 Introduction

It is becoming increasingly common to link micro-simulation models to real urban traffic control (UTC) systems and to then let the two systems interact. The UTC systems can obtain data from the simulated network, such as vehicle detections, and use this information to perform control actions in the simulated network. This approach has considerable merits. It negates the need to produce a model to replicate the effects of the UTC system. It also allows accurate simulations to be performed without the modeller having to know precise details of the how the UTC system works. This can avoid commercially sensitive information having to be revealed to the modeller.

Within the SMARTEST project DRACULA has been linked to the BALANCE UTC system. BALANCE (Friedrich et.al., 1995, Toomey et.al., 1998) is a distributed UTC system which has been developed at the Technical University of Munich. It underwent field trials in Munich within the EC funded DGXIII LLAMD project, and has since been tested in three other European cities, namely London, Glasgow and Belfast. It is due to be used in Leeds and Sheffield in the near future.

Within a BALANCE system, decisions about signal settings at individual junctions are made by Micro-BALANCE outstations at each junction. Strategic control decisions at the network level, which can override or weight decisions at the junction level, are made by a centralised Macro BALANCE computer. BALANCE uses standard TCP/IP communications protocols to communicate with signal controllers on-street and between its system components.

An interface between DRACULA and BALANCE was therefore developed which used the same TCP/IP communication protocols as used by the BALANCE system on-street. The interface was tested by simulating the operation of a single junction, namely the A30/Stanwell Road junction in South West London.

By using a multi-tasking operating system, such as Windows 95/98/NT, it is possible to run all the micro-BALANCE tasks on a single computer, if required, rather than use a separate computer for each one, as would be the case in the real world. A flexible approach was developed in the project to allow the micro and macro BALANCE tasks to be spread across a computer network as available. Similarly it should be possible to treat DRACULA as just another task and run it on any of the PCs in the computer network. In practice this caused problems if DRACULA and all the BALANCE tasks were chosen to run on a single computer. It proved difficult to display the animated graphical outputs of DRACULA simultaneously with the outputs from BALANCE. A further problem was encountered when considering how to link the standard communications routines used by BALANCE into DRACULA. The DRACULA software is currently DOS based and is incompatible with the Windows based communications routines. It therefore proved

impossible to link the communications routines with DRACULA directly. This problem was overcome by writing a small Windows program, called SPRUCE, to handle the communications which was run every simulation step within DRACULA.

The design for linking DRACULA to BALANCE was based around a further INTERFACE task which sat between DRACULA and the BALANCE tasks. The INTERFACE task acted as a server. It handled communications between all the tasks, ensured all the tasks were synchronised, and translated all the data going between the tasks into the required formats. The INTERFACE task also performed some of the duties normally carried out by on-street signal controllers, such as checking that minimum green periods had elapsed before changing signal phases.

3.4.2 Inputs

DRACULA Inputs

DRACULA required the signal settings to be input every second. This was provided as a list of the current signal aspects (Red, Amber or Green) for all the possible turns at the signalised junction.

INTERFACE Inputs

The INTERFACE program received detector data from SPRUCE / DRACULA and signal force bits from BALANCE. The force bits simply specify which stage is to be run next.

BALANCE Inputs

BALANCE received detector data bits from the INTERFACE program. The detector data was in the format of SCOOT nibbles (see Section 3.6).

3.4.3 Processing

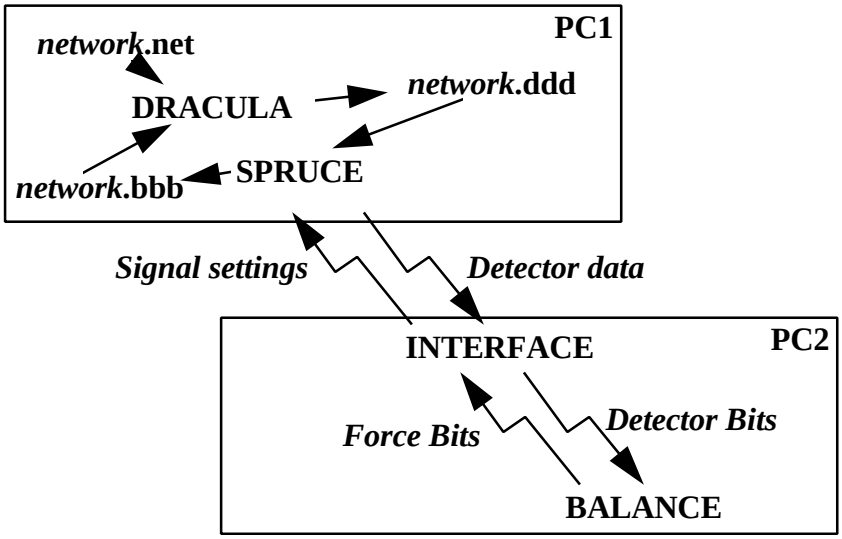


Figure 18: DRACULA - BALANCE Data Flows

Introduction

The initial proposal required that DRACULA should communicate with BALANCE via an INTERFACE program. This interface is based on a function which would need to be used by BALANCE to send out signal settings and receive detector data. This function was written using Microsoft Visual C++ and used TCP/IP communications protocols. It was supplied via a DLL

called ChatDll.dll and contained the exportable function 'XDataOnTCPIP' which takes four arguments as shown below

int XDataOnTCPIP(char* RemoteIP, int RemotePort, char* Msg2Send, char* Msg2Take);

This function just uses TCP/IP to send and/or receive message strings between two computers on a network. The function was to be called every second, by DRACULA, BALANCE and the INTERFACE program to transfer data between them.

To use the function it required:
i) BALANCE to be adapted to use the interface function
ii) the translation of the data going to (detector bits) and from (force bits) BALANCE into something DRACULA could understand. This was performed by the INTERFACE program.

DRACULA processing

As mentioned in the Introduction, it proved impossible to incorporate the XDataOnTCPIP function directly within DRACULA. Instead a simple program called SPRUCE.EXE was written which did incorporate the function. This program was run at each simulation step by DRACULA using the standard *system* function. At each simulation step, DRACULA would write detector data to a file called network.ddd and read signal settings from a file called network.bbb. The SPRUCE program would then be called and it would read the data in the network.ddd file and transmit it to the INTERFACE program and receive the signal settings back from the INTERFACE program at the same time. SPRUCE would then write the signal settings to the network.bbb file. (see Figure 18)

INTERFACE processing

The INTERFACE program performed the following functions:
- Translation of the BALANCE force bits into appropriate signal settings.
- Translation of the DRACULA detector data into the stream of detector bits required by BALANCE
- Synchronisation of the tasks

BALANCE processing

BALANCE uses the detector data it receives to optimise the signal settings. The detector bits are received and the signal force bits transmitted using the XDataOnTCPIP function.

3.4.4 6.4.4 Outputs

DRACULA outputs

Every second, DRACULA produced a list of SCOOT nibbles for all of the detectors in the network.

INTERFACE outputs

Every second the INTERFACE program sends detector bits to BALANCE and signal aspects to DRACULA.

BALANCE outputs

Every second BALANCE outputs stage force bits messages, which are full 16-bit UTC control bit pattern, but only using the stage force bits. To cater for the multiple node

architecture an item was added to the message to define which controller the following control bits relate to as in the following table:-

Byte No.	Data Content
1	Number of controllers (1 - 20 say)
2	1st controller id (0 - 19)
3	Control bit states 0 - 7, where 0 = bit inactive & 1 = bit active
4	Control bit states 8 - 15, where 0 = bit inactive & 1 = bit active
5	.....
?	Nth controller id (0 - 19)
?	Control bit states 0 - 7, where 0 = bit inactive & 1 = bit active
?	Control bit states 8 - 15, where 0 = bit inactive & 1 = bit active

3.4.5 Calibration Results

A single four arm junction is SW London was used to test the operation of the DRACULA / BALANCE interface. A careful check was made to ensure that the signal plans being recommended by BALANCE were being implemented in DRACULA.

3.5 PUBLIC TRANSPORT PRIORITY

3.5.1 Introduction

Apart from providing public transport with special reserved lanes, public transport is also given priority at signalised intersections. When a public transport vehicle is detected at time t_0 and predicted to arrive at the stopline at time t_a, one of two actions may be performed:

- *Extension*, which extends the bus green period in order to allow the bus to exit;
- *Recall*, which terminates the bus red stage earlier in order to reduce the bus waiting time.

3.5.2 Inputs

The inputs data for public transport signal priority include to install selective vehicle detectors (selected to detect public transport vehicles only) and specify a simulation control parameter to switch the priority scheme on.

3.5.3 Processing

Figure 19 shows schematically the signal priority in a space-time diagram. The signals for the bus link are shown on the top, with t_r and t_{ra} representing the start and end time of the red aspect. ra denotes red/amber. $t_{ext}=t_r+E_{max}$, where E_{max} is the user specified maximum allowed extension (in second). The distance from the detector to the stopline is d. Three bus trajectories from the detector to the stopline are drawn in dashed lines.

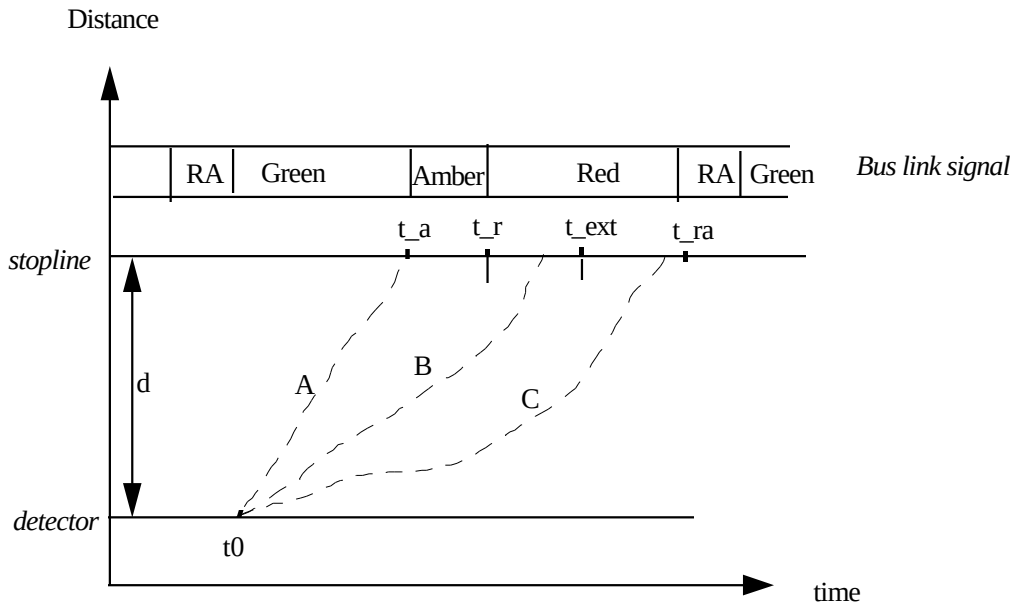


Figure 19: Space-time representation of bus signal priority

If a bus is predicted to arrive at the stopline just after the start of the red signal (case B in Figure 19), the bus green aspect will be extended by just enough time to allow the bus to exit. The amount extended depends on the predicted bus arrival time, subject to a user-defined maximum (E_{max}) and to minimum greens for the subsequent stages affected.

If a bus is predicted to arrive during the red, but an extension is not appropriate (i.e. requires more than the maximum permitted extension, case C above), then the duration of the bus red aspect may be reduced by a constant amount of 5 seconds. The length of other stages remains unchanged, so the length of the current cycle is decreased temporarily.

Otherwise the signals will not be changed (case A in Figure 19).

3.5.4 Outputs

No additional outputs have been produced.

3.5.5 Calibration Results

The operation of model has been checked using a test network in Leeds. It was not possible to validate the model as no scheme using bus priority has yet been adopted on the test site.

3.6 DETECTORS

3.6.1 Introduction

Detectors in DRACULA have been modified to allow them to output the data produced by SCOOT detectors, which are common in the UK. The data consists of quarter second occupancy bits which are sent out every second as blocks of four bits.

3.6.2 Inputs

Each detector in the network is defined by the following data:

- A unique integer identifier
- An 8 bit bit-mask to identify which of the eight DRACULA vehicle types the detector can detect
- Upstream node of the link on which the detector lies

- Downstream node of the link on which the detector lies
- Position of the detector on the link
- The lane in which the detector lies

3.6.3 Processing

The front and rear ends of a vehicle are compared to the location of a detector in the current lane the vehicle is travelling in, a detection is triggered if a vehicle passed or is stopped on a detector. The exact timing, in quarter second intervals, when the vehicle passed the detector is extrapolated based on the current speed the vehicle. This is because the simulation time increment is one second.

3.6.4 Outputs

At every simulation time step the program loops through all detectors in the network and outputs all detections. The detector data is in the form of SCOOT nibbles. This is quarter second occupancy data that is sent every second. For each detector four bits of information are sent every second. The information is passed using bytes (i.e. 8 bits), so two detectors worth of data are sent with each byte. The SCOOT nibbles are created left to right, so the leftmost bit is for the first quarter second, the rightmost bit for the last quarter second.

3.6.5 Calibration Results

The network used for the calibration test of Adaptive Traffic Signals (see Section 3.4.5) was also used to check the correct operation of the detector module.

3.7 TRAFFIC CALMING

3.7.1 Introduction

Traffic calming is represented as a special speed-controlled region in a link.

3.7.2 Inputs

A traffic claming region is specified by:

- Upstream node of the link the traffic calming region locates
- Downstream node of the link
- Starting position of the region, measured from the link entry
- End position of the region, measured from either the link entry to from the stopline;
- Maximum speed over the region.

3.7.3 Processing

When approaching a traffic calming region:

If the current speed is more than the maximum speed of the region, *then*

Decelerate at a normal deceleration to the maximum speed of the region

else

Move at the car-following speed

end if

3.7.4 Outputs

No additional outputs have been produced.

3.7.5 Calibration Results

A simple model including a traffic calmed section was built to check the correct operation of the new model.

3.8 REFERENCES

Friedrich, B., Sachse, T., Hoops, M., Jendryschik, W. and Reichert, G. (1995) "BALANCE and VARIA Methods for Traffic Adaptive Control", Proceedings of the Second World Congress on Intelligent Transportation Systems, Yokohama, Nov 9th-11th 1995, pp 2356-61.

Toomey, C., Clark, M. and Friedrich, B. (1998) "Tipping the BALANCE: A European Trial of Advanced UTC", Traffic Technology International, Dec 97/Jan 98, pp51-54.

4 NEMIS

4.1 INTRODUCTION

In the context of WorkPackage 2 the main missing features of the state-of-the-art micro-simulation models were identified (32 tools were analysed) based primarily on the results of a User Requirements survey (ref. Deliverable 3, Chapter 4).

Gaps identified in the SMARTEST models were addressed (ref Deliverable 4, Chapter 2). Based on the users' requirements, the feasibility of implementation and developers' interests, the final plan for improving the models within the lifetime of the project was determined.

According to the users' requirements, all partners agreed that particular attention has to be paid to the "Transport Telematic Functions" along with several "traffic objects and phenomena" such as *incidents, public transport* and *roundabouts.*

Hence, in order to comply with the Model Update Specifications proposed in WorkPackage 3, the main efforts have been spent to provide the micro-simulator NEMIS with an improved and standardised interface suitable for the simulation in real time of the following external Transport Telematic Applications:

- *Adaptive Traffic Signals*
- *Public Transport Priority*
- *Variable Message Signs*
- *Dynamic Route Guidance*

A standard interface, based on a TCP/IP communication protocol was adopted to connect the computer where the model runs to the network where the external strategies operate, is described with further details in the next chapter ("the new interface").

Furthermore, two models have been improved in the new release of the micro-simulator:

- *Public Transport Services*
- *Vehicle Detectors*

All the enhanced models take care of transferability aspects and will be tested according to the verification tests described in the Model Update Specifications chapter of Deliverable 4 on two different test sites: Torino and Genova.

With more details:

- Public Transport Services model has been enhanced with the introduction of layby PT stops (see Appendix A of Deliverable 4).
- Vehicle detector improvement concerns performance, error rate and breakdown occurrence.
- The validation of the standard interface mainly deals with operational aspects. Stress conditions will be generated connecting the model to a real network of SPOT traffic control units.

Further validation activities are envisaged concerning the following items:

- calibration of the car following rule according to data collected from the field
- calibration of driver compliance to VMS and DRG indications against the information made available by surveys conducted in the test-site by other specific projects.

4.2 The NEW INTERFACE

The calibration and setting up of a UTC system on street can be extremely time-consuming and difficult unless extensive off-line tests are performed in a controlled environment.

The NEMIS software package was designed specifically as a tool for testing urban traffic control strategies prior to or in parallel with on-street testing.

NEMIS already supports an interface with external road-side processors (e.g. SPOT or SCOOT OTU) to test the effectiveness of urban traffic control systems as well as to screen strategies and tune system parameters before field installation.

The interface package needs to be installed on a MS-DOS PC connected by a serial line with the NEMIS computer and by another serial line with a SCOOT system or to the SPOT MFOs[1] of the UTOPIA Network (Figure 20).

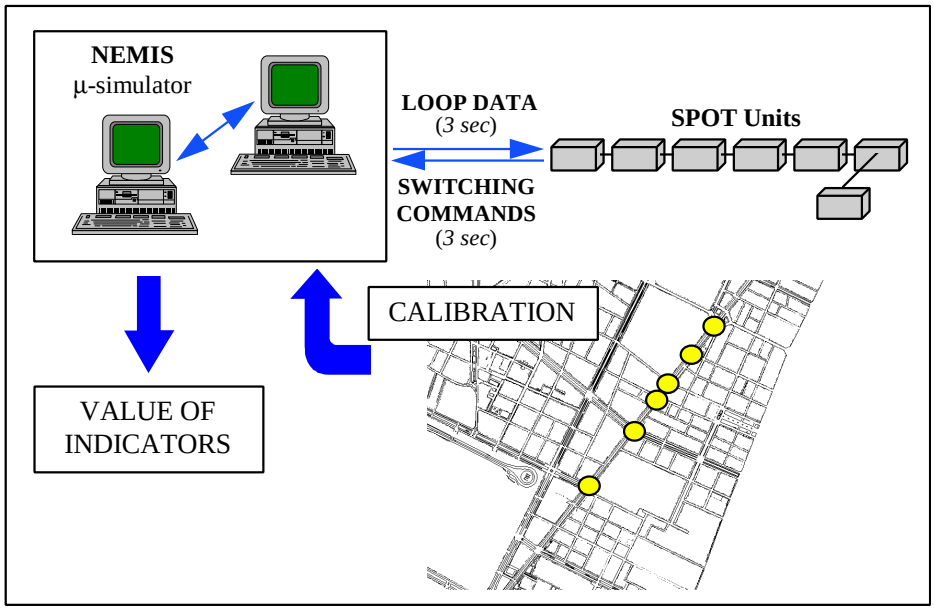


Figure 20: Use of NEMIS as a system evaluation tool before field installation

The existing interface communication protocol operates using several serial connections (RS232) between the NEMIS computer, the MS-DOS PC where the interface package runs and the real system. This solution presents the following limits:

- limited communication capabilities (limited speed of serial connections)
- non-standard communication protocol (proprietary protocol "ad hoc" to interface NEMIS with SPOT and SCOOT)
- limited simulation capabilities (dedicated hardware is needed for the intersection controller)

The new interface overcomes these limits by using a standard protocol based on TCP/IP in order to connect NEMIS with a LAN/WAN where the intersection controllers implementing external Traffic Control Strategies (e.g. MFOs SPOT) operate.

The new NEMIS interface package provides users with a simplified and high efficiency micro-simulator suitable for investigating the impact of Advanced Transport Telematics functions

[1] MFO = Multi Functional Outstation

(adaptive and co-ordinated traffic signals, public transport priority, VMS and Dynamic Route Guidance) on big network areas.

The main objective of implemented modifications (as shown in Figure 21) is to have a new interface package based on standard communication protocol TCP/IP and suitable to directly interface NEMIS with several external control strategies embedded in UTOPIA SPOT units. In fact, the new interface package is able to manage messages written directly in the UTOPIA format (For more details see Section 4.5).

It is also possible to use NEMIS in order to test any "ad-hoc" external control strategy. If this is the case, the external control strategy developed needs to exchange appropriated messages with NEMIS using the proposed format for each message (see section 4.5.3 for further details).

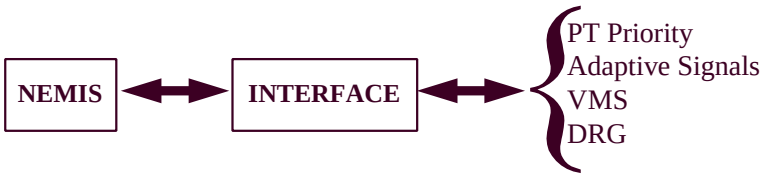


Figure 21: Target of the new interface package

Starting from the physical approach that is shown in Figure 21, the new interface package has been developed according to the architecture that is proposed in Figure 22.

Figure 22: Implementation of the new interface package

The new interface package is composed of two parts:

- TCP MANAGER — tools that manages the communications between NEMIS and the network where the external control strategies are located
- LOG FILE SYSTEM — A set of circular files[2] where the messages that need to be exchanged between NEMIS and the TCP MANAGER are stored.

The behaviour of the new interface package can be easily described by means of the following steps:

- The TCP MANAGER receives from the external control functions that reside within SPOT units or within others users developed packages, all the messages containing the elaborated control strategy. The communications between the TCP MANAGER and the external control functions are based on the TCP/IP standard protocol.

[2] Circular file forms and behaviours are described with great details in chapter 4.5: "adaptive traffic signals and public transport priority"

- When the TCP MANAGER receives a new message, the message itself is processed and then written onto the appropriated file into the Log File System.
- The TCP MANAGER also reads the circular files HIPRY and LOPRY (two command files that belong to the Log File System, see chapter 4.5) where NEMIS write the messages needed by the external control functions.
- When a new message is written by NEMIS into a command file of the Log File System, the TCP MANAGER, processes the message and then sends it out towards the appropriate SPOT unit or to the appropriate external control function.

The flow chart in Figure 23, shows in a schematic way, the behaviour of the two main tasks of the TCP MANAGER and the existing interaction between NEMIS, the TCP MANAGER and the whole simulation network.

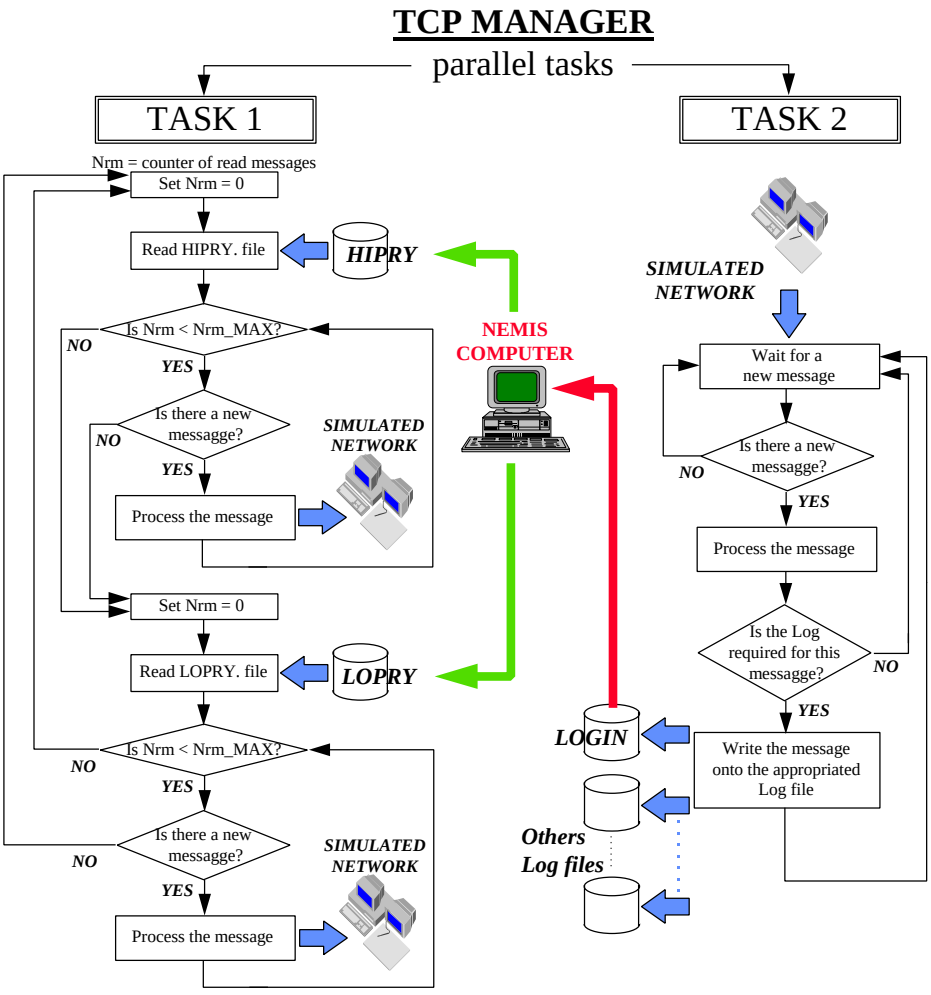


Figure 23: TCP MANAGER behaviour

The messages are written onto the Log File System using the LFS management functions provided together with the interface package. The same LFS management functions have to be used to read the messages previously written onto the Log File System by NEMIS itself and/or by any other external operating tools (see section 4.5.3 for further details).

Comparing the new interface with the old one, we can highlight the following advantages of the new interface:

- it uses a standard communication protocol
- the communication speed is limited only by the communication network features (speeds greater then 10Mb/s can be achieved by using optical fibre)
- the simulation does not need dedicated hardware (a standard PC network can be used)

The interface package is described in greater detail in section 4.5.

4.3 PUBLIC TRANSPORT SERVICES

4.3.1 Introduction

The micro-simulator NEMIS supports a detailed model for Public Transport management.

The main aspects of this model are the following:

1. PT vehicles are generated at the terminus with a random headway depending on the nominal frequency and variation defined for the service. Nominal parameters must be specified during the network coding process. During the simulation, as a PT vehicle is introduced into the network, an extraction from the distribution is made to evaluate the generation time for the next vehicle of the service. Each service has an independent random process as it has its own generation seed for the random extractions.
2. Time spent by a vehicle at a stop is randomly extracted from a distribution that changes according to the service and stop. An average stop time and a standard deviation must be specified for each stop. Another stop characteristic (that must be specified during the network coding process) is the distance of the stop from the previous one (or from the beginning of the link) in metres.
3. A PT vehicle that is moving on a link where no stops are located or that has already done all the stops on the link, can move as a private vehicle, so it could change lane and overtake other vehicles.
   If the PT vehicle has still stops to do on the current link, it stays in the lane where the next stop is, following the vehicle in front.
4. It is possible to define a separate set of traffic signals for PT vehicles that use reserved lanes. During the coding process, traffic signal for PT vehicles must be described using the same syntax as used for the “private” ones (See NEMIS User Manual, Sec. 4.4).
5. Statistics about PT are reported in the SMTP.DAT file.

Point 2 of the previous list, highlights the characteristics of the PT stop as it is implemented in NEMIS. It clearly appears that there is no information regarding the kind of the stop itself. With reference to Deliverable 4 (Appendix A - Sec.1 “Public Transport Services”), four types of PT stops must be provided by the micro-simulator in order to model the various types of public transport stop found in road network throughout Europe. Figure 24 shows these four main types of public transport stop.

The NEMIS model does not take into account the possibility of having kerbside parking at the bus stop, furthermore no models for passenger generation are provided (time spent by a vehicle at a stop is randomly extracted from a distribution that changes according to the service and stop.). From this point of view, the behaviour of the drivers that follow a PT vehicle is the same for both a typical bus stop and bus stop boarder. If there is a lane available to overtake the bus and if the gap is suitable to change lane, the driver can change lane, overtaking the PT vehicle. If no

overtaking lane is available, private vehicles can only stay in the lane where the PT vehicle is, following the vehicle itself.

Similarly, in the case of central tram boarder, the drivers that follow the PT vehicle must slow down their speed and look for a suitable gap to change lane and overtake the tram. In this case, some attention has to be paid to the passengers that are leaving the tram and that could cross the lane where private traffic is flowing. For the layby stops, when the PT vehicle reaches the stop, it leaves the lane on the carriageway and private traffic can flow normally on the lane. Later, the PT vehicle must look for a suitable gap in order to leave the stop, and, if this is the case, the private vehicles should give priority to the PT vehicle that is leaving the stop

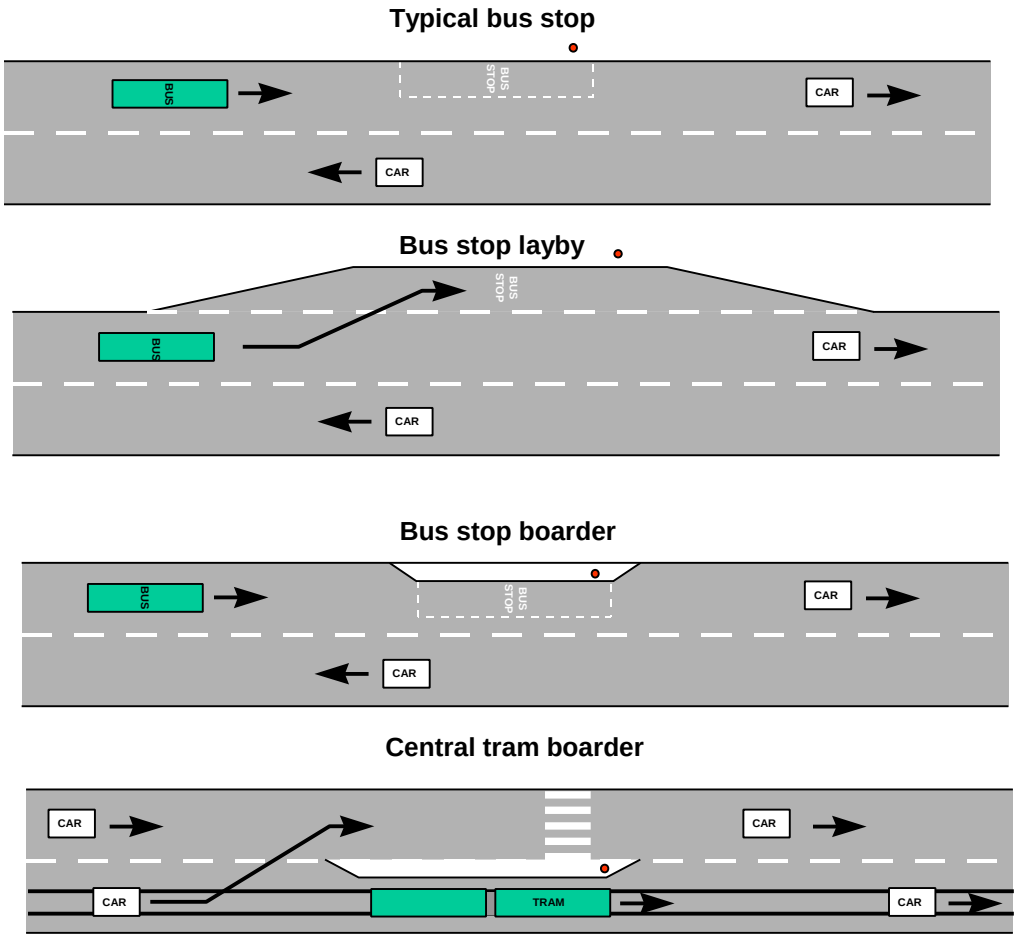


Figure 24: The four different types of public transport stop driving on the left

The micro-simulator NEMIS has been enhanced with the introduction of two different kinds of PT stops:

- Typical Bus Stop : suitable for simulating typical bus stops, bus stop boarders and central tram boarder (this last assumption is true if we do not take care of pedestrians that can cross the lane next to the stop, and that can slow or stop the incoming private vehicles)
- Bus Stop Layby : suitable to simulate bus stop laybys.

4.3.2 Inputs

Public Transport Stop

- position: distance of the stop from the previous one (or from the beginning of the link) in metres
- kind of the PT stop (normal, layby)
- number of PT vehicles that can occupy a layby stop simultaneously (only for layby)
- average stop time
- standard deviation

Interface description

The above data needs to be supplied to the model through input text file. The user must create a file to contain data representing use of the network by public transport. Data is entered in this file in accordance with specific format rules given in section 4.5.1 of the NEMIS User Manual. Due to the modifications introduced, the input record that describes a stop is now:

SENSO x,y,CORSIA z,F(pos,kind,num,tm,dv)

where: pos: position
kind: normal, layby
num: number of PT vehicles that can occupy the layby stop simultaneously[3]
tm: average time
dv: standard deviation

4.3.3 Processing

The Public Transport stop model is able to reproduce the presence of a PT stop in the simulated network and the behaviour of traffic within the region close to the stop.

When a PT vehicle enters a link, its behaviour depends on the presence of possible stops within the link.

If no stops are forecast for the PT vehicle on the link it has entered, it can move freely as if it were a private vehicle, using all the lanes available on the link and changing lanes to overtake any vehicles that proceed slowly ahead of it.

If the PT vehicle must stop on the link, it must change lane to the one where the stop is located, so the lane changing procedure for PT vehicle is actuated. When the PT vehicle has achieved the correct lane, it proceeds, following the vehicle in front until the stop has been reached. In order to avoid the situation where a PT vehicle stops in the right position but in the wrong lane during the lane changing process, vehicles proceeding on a parallel lane should give priority to the PT vehicle when it has shown its intention to change lane.

Before the stop is reached, a random extraction of the stop time will be produced, using the average stop time and the standard deviation produced as input for the model.

[3] For normal PT stops, this term is ignored (assumed = 1)

When the PT vehicle reaches the stop, its behaviour and the behaviour of any following vehicles, depends on the kind of stop. As said in the input section, two kinds of stop are provided: normal (grouping the typical bus stop, the bus stop boarder and the central tram boarder) and layby.

If the PT stop is of the normal kind, the stopped PT vehicle blocks the lane during all the stop time so that the following vehicles must change lanes to overtake the stopped PT vehicle. For the central tram border stop, the vehicles that pass the stationary tram on the inside lane should give priority to any passengers that have left the tram and that will cross the lane using the pedestrian crossing which is always provided for this purpose. To model this feature requires the development of a passenger model or, alternatively, the determination of the number of passengers that have left the PT vehicles and want to cross the inside lane. This can be modelled on the basis of a probabilistic distribution based on the stop time and on the average width of the inside lane.

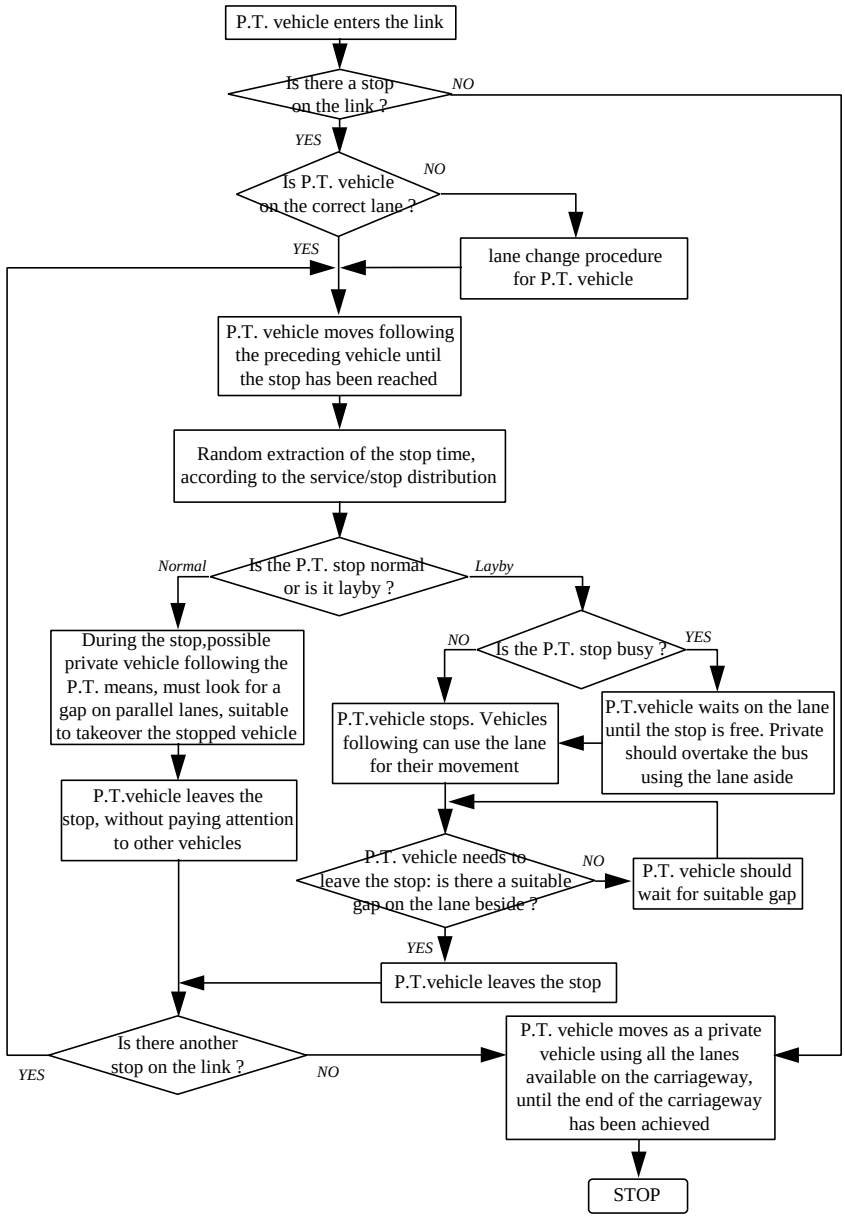


Figure 25: Public transport stops processing

In the case of a layby PT stop, the stopped PT vehicle leaves the lane where it was proceeding so that, during all the stop time, the private vehicles following the PT vehicle can proceed normally, without changing lane to overtake the stopped PT vehicle.

Of course, the layby PT stop can serve different services so that, when the PT vehicle reaches the stop, a PT vehicle of a different service might already occupy it. In this case, the dimension of the layby area (expressed in number of buses that can occupy the layby area simultaneously) determines whether the stop can be occupied immediately or whether a waiting period is required until the stop is free. If there is no space available in the layby area, the PT vehicle should stop on the lane waiting for a slot and thus block the lane to all following vehicles that must now change lane to overtake the stopped PT vehicle.

Furthermore, although the PT vehicle has priority during its lane changing movements, when it needs to leave a layby stop area, it must look for a suitable gap between incoming vehicles on the closest lane before moving. In any case, approaching private vehicles should give priority to the PT vehicle.

Figure 25 shows all these procedures in a schematic way.

4.3.4 Outputs

In the SMTP.DAT file statistics about PT are reported.

Each recorded record contains information about of reaching time, status, entrance time, exit time and stop time of a PT vehicle on a link of the network.

In particular, details are reported about:

col.	information
1	vehicle code
2	service code
3	length of the link which data are related (length is reduced by 20m for exit links)
4	link code
5	entrance lane
6	entrance clock
7	exit clock (negative if vehicle has reached a terminus or an exit node)
8	clock when the vehicles stopped near the intersection or terminus (if no stop has been made the same clock as the previous column is reported)
9,13,17	position of the stop on the link
10,14,18	status of the stop (0=free; n=occupied by n PT vehicles)
11,15,19	clock of the beginning of the stop
12,16,20	clock of restarting

4.3.5 Calibration Results

To check that the PT stop is working as specified, scenarios are to be generated which allow the following check to be made:

- whether PT stop is a layby kind, and there is no conflict between services for the use of the stop (the stop is always available), the traffic correctly operates closest to the stop. When the

PT vehicle leaves the stop it waits for a suitable gap (the private traffic should facilitate the exit of the PT vehicle from the stop).

- whenever the PT stop is a layby kind, and a conflict occurs between services for the use of the stop (the stop is busy and there is a PT vehicle stopped on the lane that is waiting to stop), the traffic behaves correctly and the vehicles following the stopped bus overtake it using other available lanes.

The enhanced PT stop model has been calibrated and evaluated on both the Torino and Genova test sites.

4.4 DETECTORS

4.4.1 Introduction

In the context of the models specifications (See Del.4 - Appendix A Sec 7), a definition has been carried out for a general detector, that is a device that provides measurements of variables that have to be selected by the users. In this case the detector technology does not matter and the interest for micro-simulation models lies in the data that can be measured and exchanged.

Another definition has been carried out in order to classify detectors into two different classes: passive detectors (take variable measurements and do not exchange data with vehicles) and active detectors (take variable measurements or receive information from vehicles and can also send information (Dynamic Route Guidance practice) to vehicles).

In NEMIS, the capability to simulate detectors is fundamental for a correct application of Adaptive Traffic Signals and PT Priority strategies[4], so there is a strong interest in the various types of passive detectors.

In the current state of the art models, the detectors are placed within the network during the set up procedure and they detect the passage of a vehicle when they reach the point where detectors are placed. No technical fault or breakdown capabilities are provided for the sensors.

Furthermore, in some cases, the users could want to test the robustness of the applied strategies when a breakdown occurs in one station of detectors or when a detector counts too much or too little. In this new enhanced release of NEMIS, it is possible to define a bias percentage for the sensor counts during the loading procedure.

Hence, the bias percentage is a static parameter that cannot be varied during the simulation time. Different simulation issues should be executed with different bias percentages for the same set of sensors in order to compare the behaviour of the external control strategies with different operative scenarios.

4.4.2 Inputs

Detector

For each controlled intersection

- controlled intersection identifier
- number of detectors

[4] Detectors for the PT locator are already provided by the simulator.

For each detector

- name of the detector in the UTOPIA terminology[5] [4 number]:
  downstream/upstream intersection identifier
  carriageway identifier
  section identifier
  flags to switch between incoming/outgoing section
- origin and destination node identifier [2 number] of the carriageway where detector is placed in the NEMIS network
- position of the detector [1 number]
  1 = detector on the main carriageway
  2 = detector on the secondary carriageway
- identifier of the controlled intersection (in the UTOPIA terminology) to which the detector belongs (needed to know the final addressee to which the message containing the detector counts will be sent)
- bias percentage for detectors counts [1 number]
  0 : detector broken
  0 < bias < 1 : detector counts too little
  1 : detector OK
  bias >1 : detector counts too much

Interface description

The above data needs to be supplied to the model through an input text file. The user must create a file "NEMSPOT.DAT" (that will be read by the procedure INIZSPOT during the initialisation phase of the simulation).

This file contains all the data required for the initialisation of the data structure used to manage the information that NEMIS and SPOT need to exchange. Data is entered in this file in accordance with specific format rules given in the following:

row	***description***	***example***
1	<*c*>Number of connected SPOT units	1
2	<*n1*> SPOT unit identifier <*n2*> number of detectors <*n3*> number of stages	29,7,4
3 to <*n3*>+3	description of detectors <*d1*> UTOPIA upstream/downstream intersection ID <*d2*> UTOPIA carriageway ID <*d3*> UTOPIA section ID <*d4*> flag for UTOPIA incoming/outgoing section <*d5*> NEMIS origin node ID <*d6*> NEMIS destination node ID <*d7*> position of detector (main/secondary cway)	28,1,1,1,78,110,1,20,29,1 30,1,1,1,110,109,1,20,29,1 127,1,1,1,110,83,1,40,29,1 127,1,1,0,83,110,1,50,29,1 127,2,1,0,15,110,1,25,29,1 30,1,1,0,109,110,1,20,30,1 28,1,1,0,110,78,1,20,28,1

[5] The main objective of implemented modifications is to have a new interface package based on standard communication protocol TCP/IP and suitable to directly interface NEMIS with several external control strategies embedded in UTOPIA SPOT units. Anyway, it is also possible to use NEMIS in order to test any "ad-hoc developed" external control strategy. If this is the case, the external control strategy developed needs to exchange appropriated messages with NEMIS using for each message the proposed format (see section 4.5.3 for further details).

<*d8*> position of detector (in metres) within the cway
<*d9*> UTOPIA intersection ID to which detector belongs
<*d10*> bias percentage

..... section regarding the description of the stages (described in 4.5.2)

4.4.3 Processing

The detectors are placed across the carriageway (one detector for each carriageway) in the position loaded from the file NEMSPOT.DAT during the initialisation procedure.

Their behaviour could be summarised as follows:

- the routine that manage the vehicle movement moves the vehicle
- if there is a detector on the carriageway and if the examined vehicles has overtaken the detector, the detector counts the vehicle
- if a bias percentage not equal to 1 is defined for the detector itself, the count is update multiplying the last count for the bias factor
- the routine that manage the vehicle movement moves next vehicle

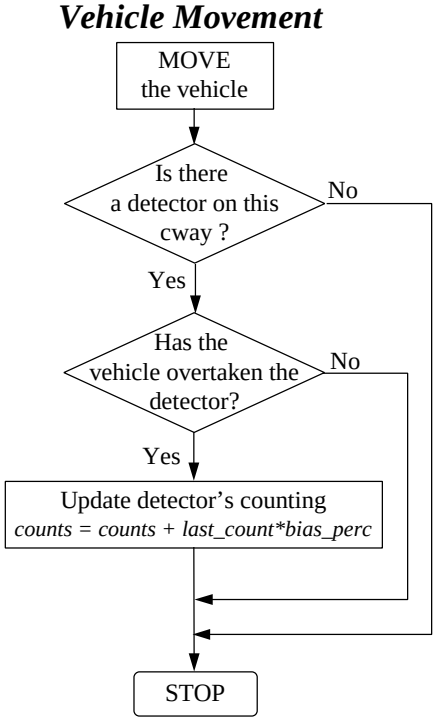


Figure 26: Detectors processing

Figure 26 shows a schematic diagram of the detectors processing procedure.

4.4.4 Outputs

The outputs of the detectors processing are the traffic counts that will be directly use by the simulator to produce the input messages required by the external control strategies. No specific output is provided for detectors evaluation[6].

[6] The messages exchange between NEMIS and the external control strategies are stored onto the appropriated file of the Log File System and can be easily accessed using the library provided with the micro-simulator. (See section 4.5.3 for further details)

4.4.5 Calibration Results

No calibration is required. The enhancement regards only the introduction of the bias percentage for counts, in order to allow the test of the robustness of proposed external control strategies.

4.5 ADAPTIVE TRAFFIC SIGNALS and PUBLIC TRANSPORT PRIORITY

4.5.1 Introduction

Adaptive control strategies for private traffic and public transport priority services, are a key part of urban traffic control systems.

Taking into consideration the growing interest shown by users for simulating the main telematic functions for traffic management with micro-simulation, the next generation of micro-simulators must provide users with the capability of simulating these fundamental functions.

It is also clear that the state of the art technology proposed for the implementation of adaptive control and public transport priority strategies is far from being standardised.

Hence, it is a good idea to separate the micro-simulator itself from the software module implementing the control strategies.

This last consideration gave birth to the idea of providing users of the NEMIS micro-simulator with a tool able to interface, directly and in an easy way with the external control strategies embedded in the SPOT unit (local multifunctional unit of the UTOPIA integrated system). The adopted approach lets NEMIS, in the easiest possible way, have the capability of evaluating the impact of particular adaptive control and public transport priority strategies (those implemented by SPOT unit). It also allows the simulation of control strategies developed ad-hoc by the user. This last concept will be explained with further details in subsequent sections, and the structure of required messages and the Library for the Log File System management will be analysed.

There follow a brief introduction of the UTOPIA integrated system, and of the adaptive control and public transport priority strategies implemented by the SPOT units at the local level.

UTOPIA (Urban Traffic OPtimisation by Integrated Automation) is a system designed to improve urban travel conditions by the application of fully automated control principles. The concept is based on a specific system architecture and control strategy. It was conceived as an innovative response to two fundamental requirements of wide-area traffic control systems:

- significant improvements in private vehicle mobility in all traffic conditions
- the assignment of absolute priority to selected public transport vehicles at traffic signal intersections

UTOPIA control strategies aim to reduce significantly the total time lost by private vehicles during their trips within the controlled area, subject to the constraint that public vehicles requiring priority shall not be stopped at intersections with traffic signals.

Field trials have shown that systems based on the UTOPIA concept can meet both the above requirements simultaneously[7].

The UTOPIA architecture is hierarchical and decentralised: optimal control strategies are determined at the higher level on the basis of area traffic prediction, while traffic signal control is

[7] UTOPIA - Technical Reference Manual

actuated at the lower level according to the actual traffic conditions encountered at the intersections.

UTOPIA control strategies overcome the unmanageable complexity of the area optimal control problem by a process of decomposition into several simpler, interrelated sub-problems. Control problem decomposition, the choice of suitable functions for the resulting sub-problems and the introduction of rules for sub-problem interrelation allow a hierarchical and decentralised architectural solution to the original control problem and provide control actuation that is close to optimal. Decomposition is performed by following a topological rule: firstly, the area is subdivided into 'overlapping' zones, where each zone is centred on a controlled intersection and includes neighbouring intersections. Then an optimal control problem is defined for each zone.

The zone control problem concerns the traffic control to be actuated at the central intersection only, but it provides a consistent interrelation with the control of the neighbouring intersections and takes into account traffic information and traffic control data concerning the whole zone. The function to be optimised consists of the terms relating to the traffic observed on the incoming link of the central intersection and those that implement the following two fundamental interaction principles:

- a *strong interaction principle* is implemented by taking into account the time lost at the downstream intersections by vehicles leaving the central one;

- a *look-ahead principle* is implemented by taking into account traffic forecasts defined on the whole optimisation horizon (120 sec) for all the central intersection's incoming links.

In order to guarantee stability and robustness at the network level, interactions are provided with a higher level, where an area optimal control problem is defined on the basis of the area traffic macroscopic model.

The final result is a series of problems that can be classified as belonging to two different classes: the 'intersection level' and the 'area level'. At both levels the problem formulation is based on the definition of two fundamental modules: the *state observer* and the *controller.*

Control problems are solved by a network of interconnected control units which apply suitable Open Loop Feedback Optimal (OLFO) techniques.

The new interface approach developed for NEMIS is based on the following components:

- the LogFile System — set of circular files that are used to store the messages exchanged between NEMIS, the TCP Manager and the SPOT units' network

- the Library for the Log File System management — Library that contains the FORTRAN functions used by NEMIS in order to manage the access to the circular files set, and the operation of reading/writing messages onto circular files

- the TCP MANAGER — a front-end TCP that, manages the communication interface of the whole UTOPIA system and the data exchange between NEMIS and the local controller unit (SPOT) of the simulated network

4.5.2 Inputs

Adaptive Traffic Signals Data

- number of controlled intersection

For each controlled intersection

- controlled intersection identifier
- number of detectors
- number of stages

For each detector[8]

- name of the detector in the UTOPIA terminology [4 number]:
- origin and destination node identifier [2 number] of the carriageway where detector is placed in the NEMIS network
- position of the detector [1 number] (main/secondary carriageway)
- identifier of the controlled intersection (in the UTOPIA terminology) to which the detector belongs
- bias percentage for detectors counts [1 number]

For each stage

- row of the SEM6 matrix where the stage is described [4 number] [9]
- offset of the traffic signal commutation with respect to the start time of the SPOT stage[10] [4 number]

Public Transport Priority

- name / number of the service
- vehicle type
- links on the service route
- position, kind, average stop time and standard deviation of the stops
- departure and frequency details

Interface Description

The above data needs to be supplied to the model through an input text file. The user must create a file "NEMSPOT.DAT" (that will be read by the procedure INIZSPOT during the initialisation phase of the simulation). This file provides information regarding the interface with the SPOT units and contains data representing the use of the network by public transport (See NEMIS Manual).

4.5.3 Processing

Adaptive control and PT priority are external strategies, i.e. external tasks embedded within the local SPOT unit of the UTOPIA System.

[8] See Detectors, Sec 4.4.2 for further details.
[9] In NEMIS, each intersection can have at most 4 arms.
[10] Each SPOT stage includes the transition period needed for switch from the stage before to the actual stage

In this section, instead of a description of the strategies themselves, the new interface provided, that allow users to implement the information exchange between the external modules where the control strategies reside and the NEMIS micro simulator is described.

This section contains the following four subsection:

- The Circular files
- The exchanged messages and their format
- The Log File System Management Library
- The simulation process

The circular files

Figure 27 shows the interface between the SPOT units (unit where the software module that implement the external adaptive control and public transport priority strategies resides) and the micro-simulator NEMIS.

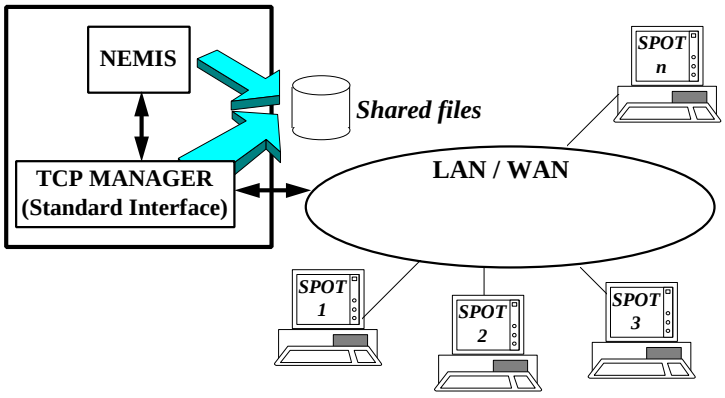


Figure 27: Scheme of the new interface

The communication takes place by means of messages logged by the TCP Manager onto a set of circular files: the Log File System. The term circular files refers to a set of logical structures that allow sequential accesses, in which the first record logically follows the last one. Circular files are used to store records of the messages that are exchanged between the various elements of the whole simulation system. They contain all the messages produced by NEMIS, by the local SPOT units and by possibly other tools of the UTOPIA system (users interface tools).

Within circular files, records are sorted on the basis of the date and time of the logged messages.

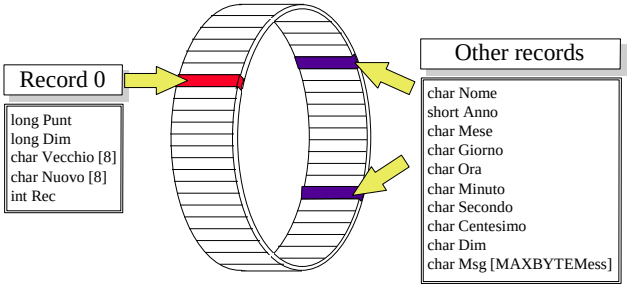


Figure 28: Circular file elements

Within the Log File System, NEMIS and the TCP manager exchange information using the three following circular files:

- HIPRY. HIgh PRioritY messages — contains messages 78 (PT forecasts) written by NEMIS.
  The TCP Manager reads the records in this file with a user defined time period (default = 0.5 sec)
- LOPRY. LOw PRioritY messages — contains messages 93 (detector counts) written by NEMIS
  the TCP Manager reads the records written in this file with a user defined time period (default = 0.5 sec)
- LOGIN. LOG INput messages — contains messages 91 ("planned" signal plan) and messages 111 (synchronisation messages) written from the TCP Manager.
  NEMIS reads this file searching for synchronisation messages (MSG 111) and "planned" signals plan messages (MSG 91)

The exchanged messages and their format

The messages needed to implement the simulation of adaptive control and public transport priority strategies are the following:

- MESSAGE 78: PUBLIC TRANSPORT FORECASTS (for PT PRIORITY)

  Communicates to the external control strategy, the arrival time forecast for the PT vehicle. It can also contain data related to the travel time between detector and stop line. It is logged onto the circular file "HIPRY." of the Log File System

Source :	NEMIS.	
Destination :	SPOT unit.	
Format :	Forecast of arrival time	3 byte
	Service Code	1 byte
	Vehicle Code	1 byte
	Number of offsets	1 byte
	For each offset:	
	Offset	2 byte

MESSAGE 93: PRIVATE TRAFFIC DETECTORS (for ADAPTIVE TRAFFIC SIGNALS)

Communicates data related to the traffic counts of the detectors directly connected to the micro.
It is logged in the circular file "LOPRY." of the Log File System

Source :	NEMIS.	
Destination :	SPOT unit.	
Format :	Time	3 bytes (MSB - MSB - LSB)
	Number of connected stations	1 byte
	For each station:	
	detector code	2 byte
	detector status	1 byte
	counts	1 byte
	occupation time	1 byte
Unit of measure :	The counts term is the number of detected vehicles.	
	The occupation time is in 18^{th} of second.	

- MESSAGE 91: STAGE PLANNED (for ADAPTIVE TRAFFIC SIGNALS / PT PRIORITY)

  This message is sent out to the upstream intersection (that needs to know the future strategy planned by the downstream intersections in order to achieve the strong interaction principle) and to NEMIS. NEMIS can then vary the SEM6 matrix (matrix for the traffic signal description) taking care of the adjustment in the control strategy.
  It is logged in the circular file "LOGIN." of the Log File System

Source :	SPOT units.	
Destination :	NEMIS and adjacent SPOT units.	
Format :	Time	3 byte
	Green time	1 byte
	Queue clearing start time	1 byte
	horizontal queue forecast	2 byte
	vertical queue forecast	2 byte
	Number of transmitted stages	1 byte
	First stage to be actuated	1 byte
	Carriageway code	1 byte
	For each stage:	
	offset for stage start	1 byte
	Look-Ahead cost coefficient	1 byte
	Nominal queue	2 byte

- MESSAGE 111: SYNCHRONISATION MESSAGE (for ADAPTIVE TRAFFIC SIGNALS)

  This message is sent out to the micro-simulator NEMIS to start a new simulation period.

  Every simulation period lasts 3 seconds (1 STEP) and the next simulation period starts when a new message 111 is received from the TCP MANAGER.
  It is logged in the circular file "LOGIN." of the Log File System

Source :	TCP MANAGER.
Destination :	NEMIS.
Format :	void

All the messages in UTOPIA format are preceded by a 4 byte header that contains the following information:

Message Code	1 byte
Origin Micro	1 byte
Destination Micro	1 byte
Step/Cost	1 byte

Many run-time messages contain, after the header of the message itself, the send time of the message (using 3 bytes). This time is expressed in seconds from midnight.
If there are no different indications (message 93) the time bytes are ordered as follows:

LSB - MSB - MSB.

Figure 29 summarises the above information.

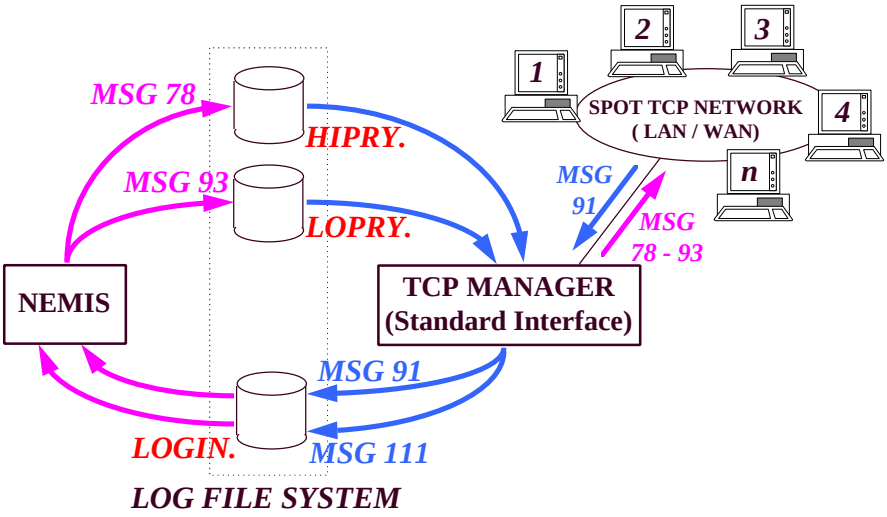


Figure 29: Description of the Log File System and exchanged Messages

The Log File System Management Library

The ***L****og* ***F****ile* ***S****ystem* ***LIB****rary* manages the Log File System files. When a message has to be logged on a file of the Log File System, the function of the LFS LIBrary control, the consistency of the recording date and time.

The system date / time variation are also managed by the library in order to avoid Log files corruption.
The main functions of the LFS LIBrary are the following:

- LFSOpen
- LFSClose
- LFSLock
- LFSUnLock
- LFSInfo
- LFSRead
- LFSWrite

All these functions are described in tables below

function	***LFSOpen***

#include	<lfslib.inc>
Syntax	RetOpen = LFSOpen (ProcessName, FileName, Path,MatFile[REFERENCE])

Type	**Parameter**	**Description**
char	ProcessName	Process Identifier
char*	FileName	Name of circular file, without path
char*	Path	full path for Log File System directory
array	MatFile	array of MaxMat structures StrMat

Description
The LFSOpen function opens the circular file named *FileName*, into the directory *Path*, for the process *ProcessName*. All the files are opened with the path specified in the environment LFSLIB. If the environment LFSLIB has not been set, the files are opened using only *FileName*. Many processes operating in multi-tasking on the same machine can open the same circular file at the same time: if this is the case, the file will be opened by all processes as a separated stream so that each process is the owner of the circular files that has opened. The information related to the opened circular files, are stored in an internal structure of the LFS LIBrary that can be read using the function LFSInfo. As soon as the file is opened, the processes can access to the file, in order to read / write a record, using only their handle (this is the value returned by the function).

Return Value
The function LFSOpen returns an integer greater or equal to zero if the opening procedure has been successfully completed: in this case the returned value is the handle that will to be used to reference the file. If the returned value is lower than zero there has been an error. -1 : a generic error occurs opening the circular file. The *OPEN* function does not succeed -3 : the file cannot be opened because the number of files already opened is equal to the maximum number of opened files that can be managed simultaneously by the library. N : handle of the opened file

Table 4.5-A: LFSOpen function

function	***LFSClose***

#include	<lfslib.inc>
Syntax	RetClose = LFSClose (ProcessName, Path, MatFile[REFERENCE])

Type	**Parameter**	**Description**
char	ProcessName	Process Identifier
char*	Path	full path for Log File System directory
array	MatFile	array of MaxMat structures StrMat

Description
The LFSClose function closes all the files opened by the process identified by ProcessName

Return Value
The function returns the number of files that has been closed for the process identified by ProcessName

Table 4.5-B: LFSClose function

function	***LFSLock***

#include	<lfslib.inc>
Syntax	RetLock = LFSLock (Handle, position, nbyte, MatFile[REFERENCE])

Type	**Parameter**	**Description**
integer	handle	Process Identifier
long int	position	record of circular file that needs to be locked
integer	nbyte	dimension [bytes] of the record to be locked
array	MatFile	array of MaxMat structures StrMat

Description
The LFSLock function allows the logical locking of the record identified by the field *position* that occupies *nbyte* bytes in the circular file.

Return Value
The function LFSLock returns the following values: 0 : the Locking operation succeed -2 : during the locking operation, one of the following errors occurs: ***EACCESS*** = (IOS=6410) Locking Violation (file already Locked/UNLocked) The program tried to lock a file that was already locked or tried to unlock a file that was already unlocked ***EBADF*** = Invalid file handle ***EINVAL*** = Invalid argument given to the function ***EGENERIC*** = (IOS=6409) LOCKING illegal on sequential file. A Locking statement specified a unit that was not opened with ACCESS='DIRECT'.handle of the opened file -3 : ***EDEADLOCK*** = (IOS=6411) Locking Violation (Value returned when the LOCK or RCLK flag is specified and the file cannot be locked after 10 attempts)

Table 4.5-C: LFSLock function

function	***LFSUnLock***

#include	<lfslib.inc>
Syntax	RetUnLock = LFSUnLock (Handle, position, nbyte, MatFile[REFERENCE])

Type	**Parameter**	**Description**
integer	handle	Process Identifier
long int	position	record of circular file that needs to be locked
integer	nbyte	dimension [bytes] of the record to be locked
array	MatFile	array of MaxMat structures StrMat

Description
The LFSUnLock function release the logical locking previously generated for the record identified by the field *position* that occupies *nbyte* bytes in the circular file.

Return Value
The function LFSUnLock returns the same values of the LFSLock function.

Table 4.5-D: LFSUnLock function

function	***LFSInfo***

#include	<lfslib.inc>
Syntax	RetInfo = LFSInfo (Handle, FileName[REF], Dim[REF], Last[REF], Young[REF], Old[REF], MatFile[REFERENCE]);

Type	**Parameter**	**Description**
int	Handle	Handle of circular file
char*	FileName	Name of circular file, without path
long*	Dim	File Dimension (in records)
long*	First	Index of the first record
char*	Young	Storage date and time of the youngest record
char*	Old	Storage date and time of the oldest record
array	MatFile	array of MaxMat structures StrMat

Description
The LFSInfo function provides some information on circular files: the name of the file (*FileName*) without its path, its dimension in number of records (*Dim*), the first free record of the file (*First*), the oldest record written in the file (*Old*) that correspond to the first record that will be covered during future write operations. The LFSInfo function returns the storage date and time of the two extreme records, the oldest and the newest ones.

Return Value
The function returns zero in case of success. The returned value is equal to -1 if the handle passed to the function (Handle) does not corresponds to any opened file or if it is outside of the available handles

Table 4.5-E: LFSInfo function

function	***LFSRead***

#include	<lfslib.inc>
Syntax	RetRead = LFSRead (Handle, Punt[REF], Buffer[REF], MatFile[REF])

Type	**Parameter**	**Description**
int	Handle	Handle of circular file
long*	Punt	Index of the record to be read
char*	Buffer	Read message
array	MatFile	array of MaxMat structures StrMat

Description

The LFSRead function allows the reading of a message stored on a circular file. After each read operation, the function updates the index of the record to be read (*Punt*); furthermore, the function returns a message only if the index *Punt* passed to the function does not corresponds to the next record to be written on the circular file.
The LFSRead function allows easy management of the reading procedure for a circular file, starting from the oldest message to end with the last message written.

The LFSRead function uses the handle (*Handle*) in order to read the file in transparent way, in other words, several different processes can read the same circular file at the same time.

Return Value

The function LFSRead returns the following values:

-1 : the passed handle, does not corresponds to any opened file, or it is outside of the available handle range

0 : the process has tried to read the message corresponding to the first record free of the circular file.

N : the reading procedure succeed. *N* is the number of significant bytes of the message read

Table 4.5-F: The LFSRead function

function	***LFSWrite***

#include	<lfslib.inc>
Syntax	RetWrite = LFSWrite (Handle, Buffer[REF], Lungh, MatFile[REF], NonHoMaiScritto[REF])

Type	**Parameter**	**Description**
int	Handle	Handle of circular file
char*	Buffer	Message to be written
int	Length	Significant length (in byte) of Buffer
array	MatFile	array of MaxMat structures StrMat
logic	NonHoMaiScritto	flag = .TRUE. for the first write operation of the process

Description
The LFSWrite function allows the writing of the message *Buffer* (that is long *Length* in byte) to a circular file. The message is written to the first free record of the circular file. The other information related to the record (date and time, process identifier) is provided directly by the function. The LFSWrite function operates the logical locking of the resource, so that only one process can modify a circular file at the same time. When a process tries to write a circular file that is locked, it waits for a fixed time-out. Then, if the lock is removed before the end of the time-out, the write operation is complete, otherwise the write operation ends without writing.

Return Value
The function LFSWrite returns the following values: -1 : the passed handle, does not corresponds to any opened file, or it is outside of the available handle range -2 : the function LFSLock returns an error -3 : the LFSWrite ends for time-out -4 : the date and time of the written record are older with respect to the date and time of the last message stored. N : the writing procedure fully succeed. *N* is the number of significant bytes of the message written

Table 4.5-G: The LFSWrite function

The Simulation Process

The flow chart in Figure 30 (the same as in the introduction), shows in schematic way, the behaviour of the two main tasks of the TCP MANAGER and the interaction between NEMIS, the TCP MANAGER and the whole simulation network.

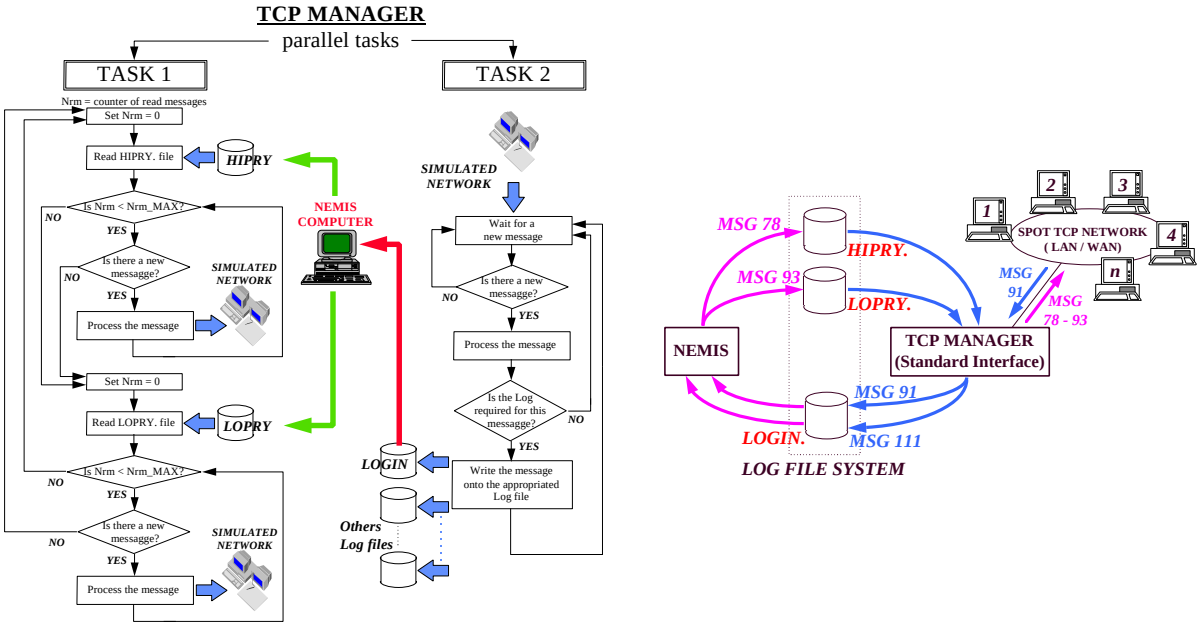


Figure 30: TCP MANAGER behaviour, Log File System and exchanged Messages

The behaviour of the communication between the TCP MANAGER and NEMIS, can be summarised as follows:

- The TCP MANAGER receives from the external control functions that reside within SPOT units or within other user developed packages, all the messages containing the elaborated control strategy (Message 91). The communications between the TCP MANAGER and the external control functions are based on TCP/IP standard protocol.
- When the TCP MANAGER receives a new message, the message itself is processed and then written into the appropriate file in the Log File System.
- The TCP MANAGER also reads the circular files HIPRY and LOPRY where NEMIS writes the messages needed by the external control functions (Messages 78 and 93).
- When a new message is written by NEMIS into a command file of the Log File System, the TCP MANAGER, processes the message and then sends it out towards the appropriate SPOT unit or to the appropriate external control function.

Figure 31 shows a screen-shot of the TCP Manager user interface.

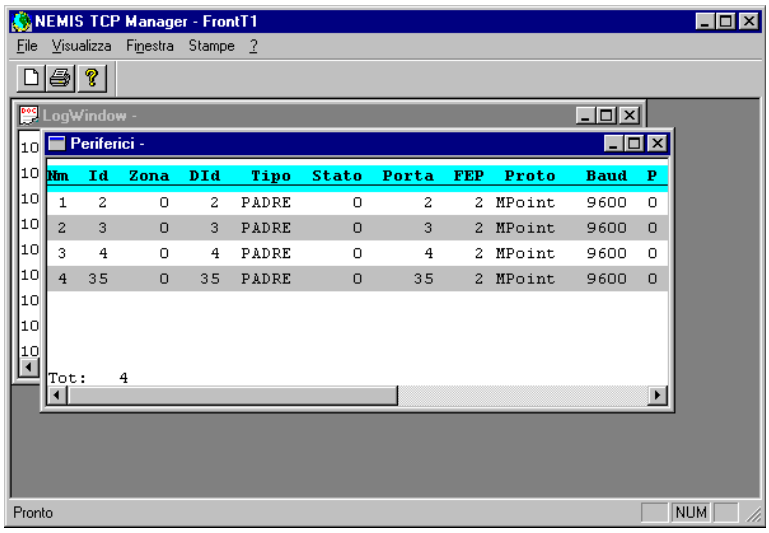


Figure 31: User interface of the NEMIS TCP Manager

The way in which the simulation process changes when NEMIS is linked with external control strategies can be described as follows:

1. NEMIS simulates for the three seconds and then:
   prepares the messages 93 (traffic counts for external adaptive control strategy)
   prepares the messages 78 (PT vehicle forecasts for external PT priority strategy)
2. NEMIS writes onto the Log File System the messages previously prepared
   all the messages 93 are written onto "LOPRY." file
   all the messages 78 are written onto "HIPRY." file
3. At the end of the simulation period, NEMIS reads the circular file "LOGIN." starting from the last message read and looking for messages 91 coming from the SPOT local units and for message 111 coming from the TCP Manager.
4. When a message 91 is detected the SEM6 matrix (containing the traffic signal information) is updated with the new planned plan information.

5. When a message 111 is found, the new simulation period (that lasts three seconds) starts and the simulation process returns to the step 2.

Figure 32 shows a flow chart for the simulation process.

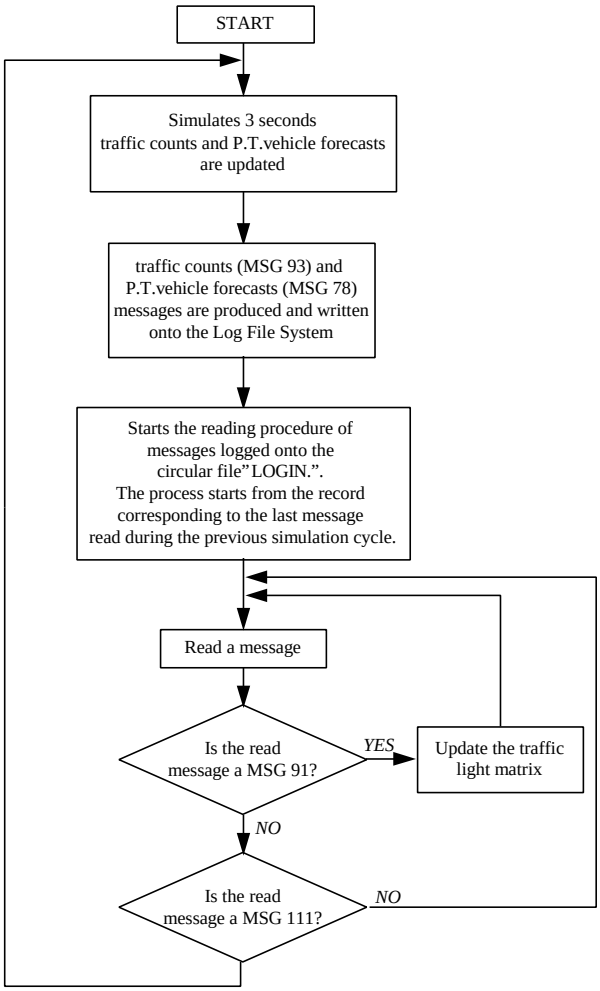


Figure 32: NEMIS simulation process

4.5.4 Outputs

All the messages exchanged between NEMIS and the TCP MANAGER are logged in the Log File System. Therefore together with the standard output provided by NEMIS for private traffic and public transport analysis (See NEMIS Manual), it is possible to use all the UTOPIA analysis tools to evaluate the impact of the external control strategies on traffic mobility and on public transport priority. Figure 33 shows some examples.

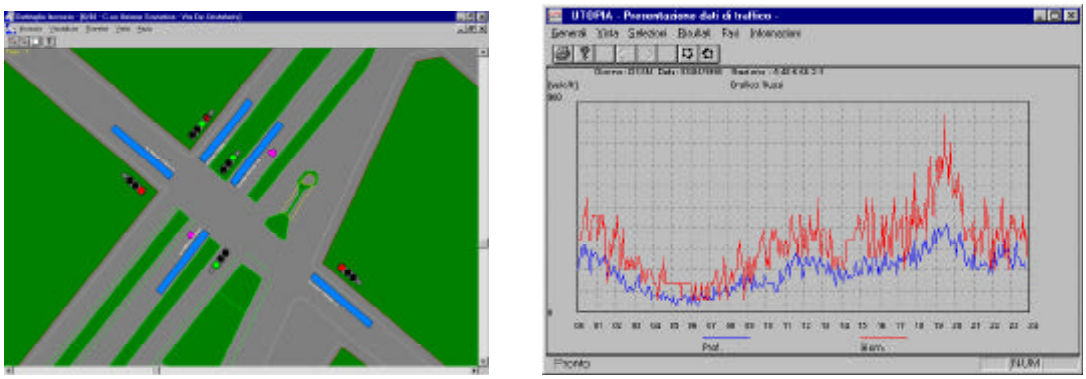


Figure 33: Example of UTOPIA analysis tools

4.5.5 Calibration Results

Due to the fact that we do not consider here tests that are related to the external control strategies themselves, and the data structure of NEMIS have been already validated and calibrated, the only tests should regard the communication process and the Log File System management strategies.

Tests have been made to verify the consistency of the newly developed LFS library (messages are read / written correctly onto the Log File System) and the communication protocol in NEMIS.

Also the TCP MANAGER behaviour has been tested.

4.6 VARIABLE MESSAGE SIGNS

4.6.1 Introduction

The main objective of the use of Variable Message Signs for traffic guidance is to support drivers by dynamic and collective information about suitable directions to reach their destinations.

Within the extensive concept of Collective Traffic Information, the purpose of variable message signs is to provide drivers with information of general interest concerning current and foreseen problems in the network - such as roadworks and limitations to traffic circulation. This information does not necessarily include suggestions about the route to follow.

Focusing on Collective Traffic Guidance applications, a significant influence on traffic behaviour is achieved by placing the signs at strategic points of the road network, in such a way as to intercept the main traffic flows. Then the Traffic Guidance applications provide additional information on the *causes of the diversions* when the directions suggested differ from those "normally" chosen by traffic on the basis of the network knowledge, or from those suggested by static signs.

Micro-simulation models performing the "verification" (operational tests and impact analyses) of guidance strategies should have the following characteristics:

- drivers behavioural model review — introduction of stochastic processes suitable for representing drivers compliance with panel information
- data structure definition — introduction of new data structure to model VMS panels and their information
- VMS loading and updating procedures — introduction of new procedures related to the new data structures loading and updating operations
- definition of the interaction between VMS panels and control function
- definition of data that need to be collected during the simulation in order to assess control strategy effects

NEMIS already supports a model for the simulation of VMS effects on traffic behaviour. In this case the model will be further calibrated focusing attention on the parameters characterising the control strategy operations.

The following section will describe with further details the model itself and will also consider its calibration.

4.6.2 Inputs

Road Network Model

nodes (n)	origins, destinations and intersections between arteries
arcs (i)	oriented links between two adjacent nodes on a traffic arterial
micro-destinations (d)	traffic flow destinations. Can be related to nodes or arcs and represent main destinations within the network
macro-destinations (D)	micro-destination logical aggregations, are related to zones directly addressable by the VMS panels.
turns (ij)	consecutive arcs that constitute the possible turning movements for each node

Traffic Model

traffic flow (F_i)	traffic flow on the arc_i
percentage rate (a_{ijd})	percentage of traffic flow on arc_i that wants to reach destination *d* through arc_j . This parameters determines the turning percentages of the node *n* (defined by the *turn ij*) related to the destination *d.*

Control Variables

target percentage rates (a^o_{ijd})	for each *turn ij* and for each destination *d*, are updated with a short time period user defined
reference percentage rates (a^r_{ijd})	for each turn *ij* and for each destination *d*, are updated with a brief time period user defined
reference traffic flows (F_{id})	percentage of traffic flow on the arc *i* that wants to reach destination *d*

VMS panel description

- Number of VMS panel

For each panel

- panel code
- number of addressed macro-destinations
- code of the carriageway where the panel is located

For each macro-destination

- macro-destination code
- list of addressable micro-destinations
- turning movements allowed to reach the macro-destination

Driver behaviour description

- compliance rate for each private vehicle class

Interface description

Of the above data, used by the VMS external control strategy, only the data related to the VMS panel description and the drivers behavioural model needs to be supplied to the model via an input text file. Data related to the description of the network and of the traffic are provided during the loading process of the network, while data related to the control strategy are evaluated and updated by internal procedures.

The user must create a file "VMS.DAT" that will be read by the micro-simulator NEMIS during the setting up phase.

This file contains all the data required for the initialisation of the data structure used to manage the VMS control strategy simulation. Data is entered in this file in accordance with specific format rules given in the following:

row 1: <number of VMS panels on the network (*N_VMS*)>
For each panel:
first row : <VMS panel code [*1-N_VMS*]>,
<number of macro-destinations of the panel [*N_ZONE*]>
second row : <origin node of the carriageway where the panel is located>
<destination node>
For each macro-destination
first row : <macro-destination code [*1-N_ZONE*]>,
<number of addressable micro-destinations>
<turning movement allowed to reach the micro-destination>
second row : <list of micro-destinations separated by commas>

Drivers compliance rate specified for each class of private vehicle.

4.6.3 Processing

It would seem from reading the previous sections that the VMS model is based on the aggregation of micro-destinations in macro-destinations. In fact, the destinations addressed by the VMS are selected in such a way to meet the interest of main traffic flows crossing the sites where the signs are placed. Therefore VMS control strategies must be able to model and elaborate traffic diversions towards destinations that in general correspond to groups of elements of the road network and that can be defined as "*macro-destinations*".

The destination of the driver corresponds to a particular point (or limited zone) of the network. For micro-simulation purposes this destination is modelled in terms of nodes. Consequently, driver destinations can be defined as "*micro-destinations*".

Therefore a correspondence between macro and micro destinations is defined. This is needed both to implement the model of the interaction between drivers and VMS (the driver needs to identify the possible macro-destination which corresponds or includes his micro-destination) and to fix the area addressable by the guidance strategy by means of each VMS.

Defined:

$\mathbf{d_i}$ the generic micro-destination
$\mathbf{D_j}$ the generic macro-destination

The following table shows an example of the correspondence between the macro and micro destinations of a VMS_1 and a VMS_2.

	d_1	d_2	d_3	d_4	d_5	**D_1**
	d_6	d_7	d_8	d_9	d_{10}	
D_2	d_{11}	d_{12}	d_{13}	d_{14}	d_{15}	
	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}	
	d_{21}	d_{22}	d_{23}	d_{24}	d_{25}	

				D_1		
	d_1	d_2	d_3	d_4	d_5	
	d_6	d_7	d_8	d_9	d_{10}	
D_2	d_{11}	d_{12}	d_{13}	d_{14}	d_{15}	
	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}	
	d_{21}	d_{22}	d_{23}	d_{24}	d_{25}	

VMS_1 Macro-destination D_1 = aggregation of micro-destinations (d_4 , d_5 , d_9 , d_{10} , d_{14})
VMS_1 Macro-destination D_2 = aggregation of micro-destinations (d_{11} , d_{12} , d_{13} , d_{16} , d_{17} , d_{18})
VMS_2 Macro-destination D_1 = aggregation of micro-destinations (d_9 , d_{10} , d_{14})
VMS_2 Macro-destination D_2 = aggregation of micro-destinations (d_{11} , d_{12} , d_{13} , d_{16} , d_{17} , d_{18})

Table 3 : Macro-destinations (D) and micro-destinations (d)

Table 3 shows that the same macro destination can be addressed by different VMS, but due to the different VMS positions the common macro-destination could correspond to a different aggregation of micro-destinations.

The VMS control strategy can be subdivided into the following modules:

- control function
- actuation module
- driver behavioural model

The *control function* runs every 5 simulation minutes and performs the following actions:

- on the basis of the observation of the density of critical links in network the turning percentages a_{ijd} for each carriageway and for each destination in the network are evaluated, starting from their nominal value a^r_{ijd}
- for each VMS panel, the turning percentages a_{ijd} are modified to turning percentages a_{ijD} related to the macro-destinations, using the carriageway flows for macro-destinations F_{iD} that are obtained by grouping the carriageway flows for micro-destinations F_{id}
- the turning percentages a_{ijD} are used to evaluate the time of permanence of the message on the VMS panel
- for each panel, for each turn and for each macro-destination the indicators of the diverted flow I_{id} are evaluated

The *actuation module* operates every second and maintains the suggested turns on the VMS panel for the time evaluated by the control function

The *driver behavioural models* operates updating the status of the vehicle every simulation step (1 sec) on the basis of the following item:

- status of the traffic signal at the end of the link and/or right way precedence rules
- desired turn at the end of the link (defined on the basis of pre assignment results (BASSOT) that aims to minimise the travel time in the network)
- particular control strategies operating on each vehicle (i.e. route guidance)
- movements allowed on the link (depending by the position of the vehicle on the link)
- car following rules

- The modifications fn driver behavioural models necessary to simulate the presence of a VMS panel on the link, are taken into account by the simulator during the assignment of the turn at the end of the link for the examined vehicle.
- When a VMS panel is placed on a link, the desired turn at the end of the link will be determined by taking into account the information shown by the panel, as well as the equilibrium assignment.
- The assigned turn depends on the destination **d** of the vehicle, and the correspondence between this destination and the macro-destination **D** addressed by the panel
- The message shown will be accepted/rejected in a stochastic way but will adhere to a mean compliance rate (user defined)

Figure 34 shows in schematic way the driver behavioural model adopted

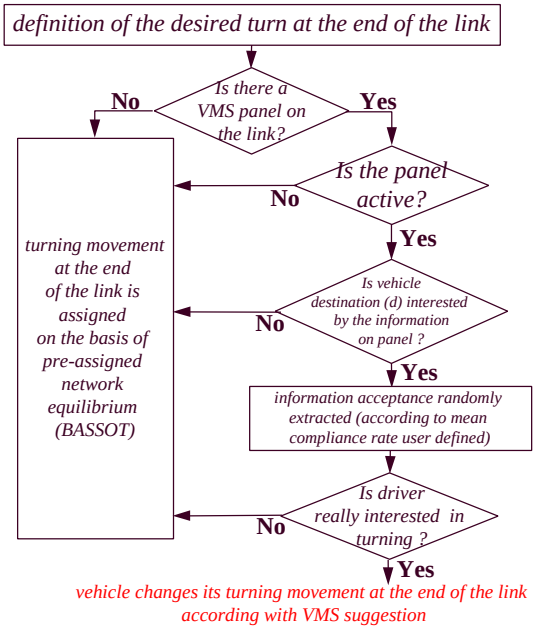


Figure 34: Driver behaviour model

4.6.4 Outputs

The standard outputs of NEMIS allow the gathering of data related to the whole network as well as single carriageways. In particular, the most significant outputs are:

Global data: global travel time
average speed in the whole network
global travelled distance
global delay

Carriageway data travel time (average and standard deviation)
number of vehicles that enter the link during the simulation period
number of vehicles that exit from the link during the simulation period
number of vehicles stopped on the link during the simulation period
average delay
average occupation of the link (ratio between average number of vehicles present on the link and its capacity)

A significant indicator for the VMS action is the average travel time origin-destination. This value can be more or less significant depending on the number of vehicles that travelled on the route. If the O/D flow is not sufficiently great, only main O/D pairs can be considered or it is possible to evaluate an average time O/D for the whole network using the flows as weighting factors.

Another indicator is the status of each carriageway (traffic density on the carriageway), that allows the identification of local congestion, and the management action of the control strategy.

Furthermore, due to the fact that panels diverts flows, statistics related to traffic diversion are saved when the VMS control strategy operates. For each carriageway where the panel is located, data are gathered regarding:

- number of vehicles that have travelled along the link
- number of vehicles interested in the indication shown by the panel (because this indication influences their destination)
- number of vehicles that comply with the panel indication (depending on the behavioural model)
- number of vehicles that change their turning movement after a panel indication

4.6.5 Calibration Results

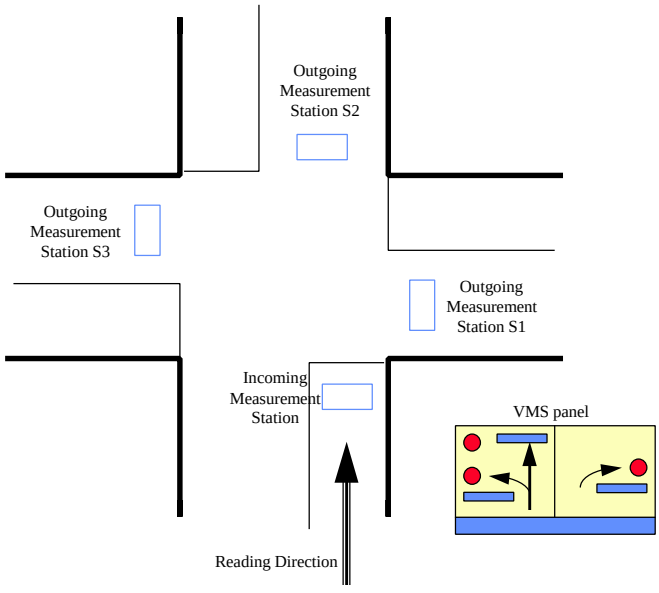
For Collective Traffic Guidance Strategies, user compliance is very important. Compliance rates are being further validated by field trials on the basis of two different methods:

- indirect method:

Compliance rates are computed on the basis of re-routed traffic flows.
An ON/OFF approach can be used to gather traffic data in the two different operative conditions: system in operation (ON) and system not in operation (OFF).
During OFF measurements, the VMS must not be visible.
Carriageways downstream of the link where the VMS is located must be equipped with traffic detectors in order to gather traffic counts in both ON and OFF conditions (see the following scheme). These values will be used to compute turning percentages.

- direct method:

Compliance rates are computed by direct interview of drivers downstream of the VMS.



4.7 DYNAMIC ROUTE GUIDANCE

4.7.1 Introduction

There is an interest in Individual Route Guidance systems simulation because of the general opinion that this kind of system will soon become an important instrument for Traffic Management.

There is interest in both in the possible impact of different systems features, architecture and penetration rates, and in the feasibility of integration of schemes involving IRG, UTC, VMS and other traffic and transport control systems.

Individual guidance information is provided to the driver by means of acoustic, optical or combined technologies. The best solution has not been fixed yet and depends both on the type of information to be communicated and on safety issues.

Individual guidance is provided according to static route definitions or dynamic route calculation. The dynamic solution is performed based on current and foreseen traffic conditions and is more related to Traffic Management concepts.

On the basis of the Individual Route Guidance systems (in the following simply referred to as IRG) classification provided in Deliverable 4 - Annex A Sec 9, NEMIS can simulate dynamic[11] autonomous[12] and infrastructured[13] systems. Also dual-mode[14] systems can be simulated.

All the simulations of infrastructured systems are based on Short Range Communication systems: communication performed by means of Infrared or Radio Beacons.

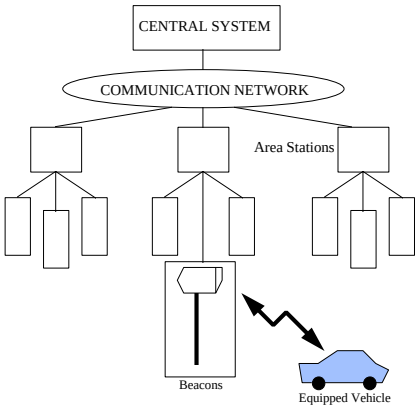


Figure 35: Infrastructured Route Guidance general architecture

[11] Travel time, traffic density and congestion are the parameters dynamically updated in the context of the dynamic IRG system.

[12] Decisions are taken according to the current vehicle position referred to the digitised territorial map, following pre-defined routes or taking into account possible dynamic traffic information (congestion, incidents, flows or travel times according to the system) provided by broadcasting systems.

[13] The system operates based on two-way communication between on-board equipment and roadside infrastructure (such as infrared beacons) connected to a centre. The ultimate "route choice" is performed on-board the vehicle according to the driver destination, while the "route calculation" is performed at the central level where dynamic traffic data are processed to update the network status estimate and to consequently optimise the routes for the possible O/D pairs.

[14] Dual mode is a combination of autonomous and infrastructured systems. The vehicle is able to perform the route choice on-board, based on the local database and on the traffic information transmitted by the broadcasting systems. When it crosses the area of the road side equipment it exchanges data and performs as in the case of the infrastructured solution.

In the adopted architecture, the global map is located at the infrastructure level. Small streets are not modelled. Positioning is performed on-board using autonomous equipment (dead reckoning and map matching functions), and dynamically via beacons. Traffic data (from vehicles and other sources) are centralised and refreshed with a sample period of few minutes. They are used together with historical data to compute the optimal routes.

Mono-routing and multi-routing criteria are used to define routes. In the mono-routing concept only one route is suggested to all the equipped vehicles going to the same destination. In the multi-routing concept the flow is split into several paths according to the possible (significant) alternatives.

Individual route guidance modelling involves the following aspects:

- the development of the module which performs the strategy for route calculation according to the optimal criteria adopted
- the development of the module which performs the route choice for the single vehicle
- the representation of the communication infrastructures (if any) which are located in the network
- the extension of the driver behaviour model to include the interaction with the on-board equipment
- the development of the data filtering module that acts as the interface between the network/traffic model and the guidance strategy module.
- the development of the scheduler which defines the timings of exchange of information (if any) between vehicles and infrastructure
- the extension of the traffic model to include the new typology of equipped-vehicles, the related generation procedure and the connection with route choice activities.

The main part of these models, already supported by NEMIS, have been revised and calibrated.

4.7.2 Inputs

Road Network Model[15]	
nodes (n)	origins, destinations and intersections between arterials
arcs (i)	oriented links between two adjacent nodes on a traffic arterial
micro-destinations (d)	traffic flow destinations. Can be related to nodes or arcs and represent main destinations within the network
turns (ij)	consecutive arcs that constitute the possible turning movements for each node

Traffic Model	
traffic flow (F_i)	traffic flow on the arc_i
percentage rate (a_{ijd})	percentage of traffic flow on arc_i that wants to reach destination *d* through arc_j . This parameter determines the turning percentages of the node *n* (defined by the *turn ij*) related to the destination *d.*

[15] The Dynamic Route Guidance system operates on the whole network described for the micro-simulator. The micro-provided simulator does not supply a network model ad-hoc for Route Guidance system.

Control Variables

target percentage rates (a^{o}_{ijd})	for each *turn ij* and for each destination *d*, are updated with a short time period user defined
reference percentage rates (a^{r}_{ijd})	for each turn *ij* and for each destination *d*, are updated with a brief time period user defined
reference traffic flows (F_{id})	percentage of traffic flow on the arc *i* that wants to reach destination *d*

Infrastructure description (IR beacon)

- Number of beacon

For each beacon

- code of the node where the beacon is located

Driver behaviour description

- compliance rate for each private vehicle class (normally assumed equal to 1)

RG Vehicle Generation

- Penetration rate

Interface description

Of the above data, used by the DRG external control strategy, only the data related to the location of the DRG system infrastructures (IRED beacons) needs to be supplied to the model through an input text file. Data related to the description of the network and of the traffic are provided during the loading process of the network, while data related to the control strategy are evaluated and updated by internal procedures. The user must create a file "MPA.DAT" that will be read by the micro-simulator NEMIS during the setting up phase.

This file contains all the data required for the initialisation of the data structure used to manage all the DRG control strategies supplied. Data is entered in this file in accordance with specific format rules given in the following:

row 1: <number of beacons in the network>
For each beacon:
< node of the network where the beacon is located>

The penetration rate (number of routing vehicles that will be generated) must be specified within the DIGIT input file (See NEMIS User Manual).

Also the kind of routing strategy adopted must be specified within the DIGIT inputs file, choosing a strategy between those proposed in

Code	*DRG Strategy*
NO	no routing
SI	RG with fully equipped network
MP	Multi-Path algorithm
MR	Time sharing Mono-routing
MB	Mono-routing with "max-beta"
MT	Minimum Time
DM	Dual Mode RG

Table 4 : Available Routing algorithms

4.7.3 Processing

It must be underlined that the results in this field are only expected from the Dynamic IRG solutions, i.e. those normally referred to as DRG.

First of all we introduce the concept of a routing vehicle as used by NEMIS: a routing vehicle has the capability to elaborate information and take decisions. A routing vehicle knows (as does a normal one) its final destination and, while no information is received from external control strategies, it behaves as a normal vehicle, trying to achieve its final destination. As soon as IRG information is received from an infrastructure of the network (such as IRED beacon), the routing vehicle calculates the best route for its destination. While no further information is received, the routing vehicle follows the best route for its destination calculated during the last elaboration. It follows that the route taken by a routing vehicle depends on the elaboration of all the available information and so, due to the fact that information is provided by the external control strategy, on the routing strategy adopted.

Roadside Infrastructure Model, Driver/Vehicle model, RG Vehicle generation and the management of the RG data flows are integrated within the network/traffic model due to their direct correspondence with models that are already important components in the micro-simulation model.

NEMIS supports the following DRG control strategies:

- *Multi Path Fully equipped Network.* Each time that a routing vehicle approaches an intersection, it receives the information regarding the next turning movement (based on its destination). This approach can be compared to the assumption of a fully equipped network, where each intersection is equipped with a beacon.
- *Multi-Path Algorithm (MPA).* It is assumed that only some intersections within the network are equipped with an IRED beacon, so that a routing vehicle can receive the information needed to choose the route (desired turning percentages) only when it is approaching an equipped intersection. The effective route choice is performed on-board. This approach (which is more realistic than the previous one) supposes the presence of a communication system able to manage the exchange of a great amount of data between routing vehicles and beacons.
- *Time sharing Mono Routing.* Together with the assumption that only some intersections within the network are equipped with an IRED beacon, here it is assumed that only one path is suggested to each routing vehicle. The choice of the route to be suggested is performed on a time period greater than the time needed to perform the evaluation of turning percentages. For each route, starting from the desired turning percentages, an attribution percentage is evaluated (β); then, this attribution percentage is converted into the time period during which the corresponding route is suggested to the vehicles. It follows that, at any time, only one

route is suggested to all the routing vehicles that have the same desired destination; different suggestions can be provided at different moments.

- *Mono Routing "max-beta"*. Similar to the preceding solution for the route calculation method, it is different because during all the time period needed to perform the evaluation of turning percentages only one route (those maximising the attribution percentage β) is suggested to all the routing vehicles that have the same destination.
- *Minimum Time*. This algorithm is not properly a DRG approach. It is based on the evaluation of the shortest path between beacons and all reachable destinations. The minimum path is then communicated to all equipped vehicles crossing the intersection where beacons are located. It is assumed that the current travel time on each link[16]as well as incidents and congestion phenomena are known.
- *Dual-Mode Route Guidance.* As the preceding solution, this last algorithm is based on minimum path evaluation. It simulates the behaviour of two different strategies operating at the same time within the simulated network. *RDS/TMC* technology transmits to the whole network, information related to local congestion (in terms of link impedance), with 20 minutes of delay. *Beacons* transmit minimum paths to reach all destinations, evaluated on the basis of the current network status, with 10 minutes of delay. Equipped vehicles normally follow the minimum path autonomously evaluated on the basis of nominal travel time and of all information coming from the RDS/TMC system; then, when they cross an intersection where a beacon is placed, they receive all the information on optimal path to reach their destination.

The control strategy applied is common to all the above solutions, and it is based on the calibration of the link density to a nominal value. Also the calculation of the impedance and cost functions, as the calculations of the desired turning percentages is common to all solutions.

Figure 36 shows in a schematic way the behaviour of DRG strategies

[16] In NEMIS this times are evaluated on the basis of the time employed to travel the link by preceding vehicles

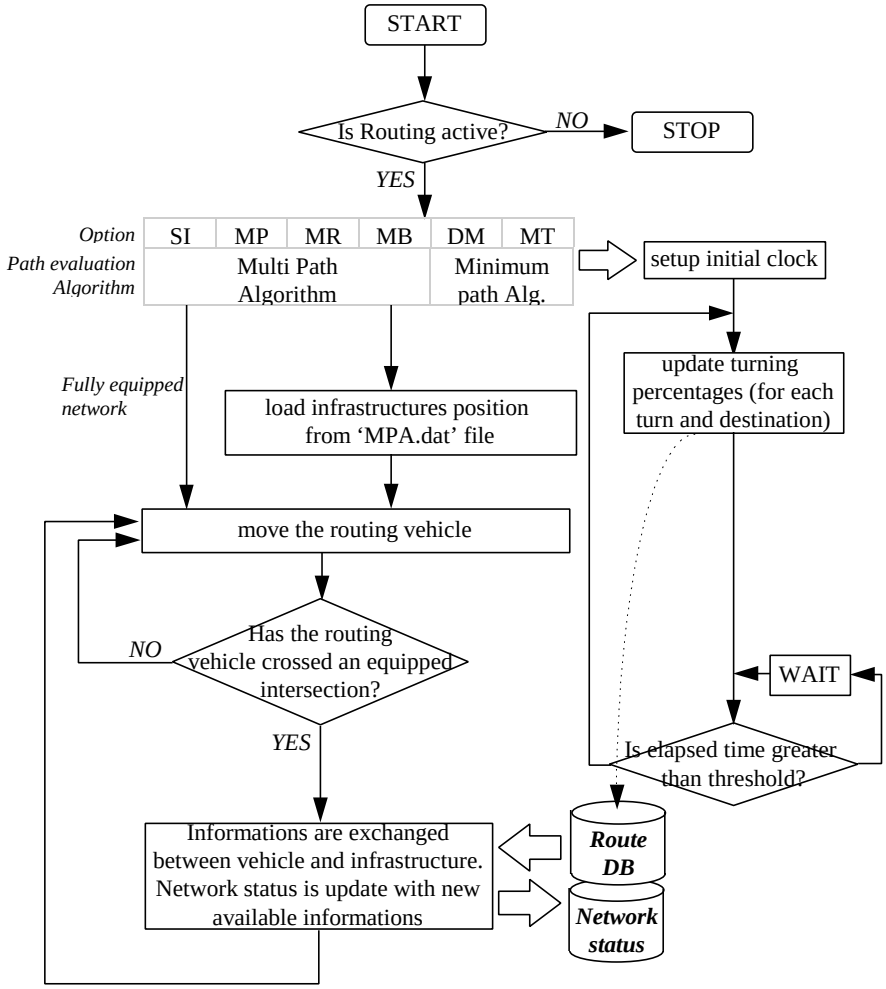


Figure 36: DRG schematic behaviour

4.7.4 Outputs

The standard output of NEMIS allows the gathering of data related the whole network as well as to single carriageways.

All the global data gathered for private vehicles (See NEMIS User Manual) are also gathered for routing vehicles, so that comparisons can be easily performed.

4.7.5 Calibration Results

To check the effect of RG strategies, scenarios have been generated which allow the following scenarios tobe simulated:

- Basic scenario (without control actions):
  - general traffic data (congestion detection)
- Basic scenario with Individual Route Guidance System ON:
  - different penetration rates
  - different control strategies
- Comparison between different scenarios

For each scenario, the following must be verified:

- effectiveness of the actuated guidance strategy
- roadside infrastructure status (re-routing percentage for each destination)
- number of vehicles which crossed the equipped nodes
- number of vehicles receiving route suggestion
- number of vehicles that complied with the suggestion
- number of vehicles that modified their turning movement according to route suggestions

The effectiveness of the enhanced RG strategies has been evaluated on the Torino test site.

5 SITRA-B+

5.1 INTRODUCTION

In accordance with the Update Specifications proposed in Project Deliverable D4, the following functions have been developed or enhanced in SITRA-B+ :

- *Public Transport Services*
- *Roundabout*
- *Parking Management*
- *Adaptive Traffic Signals*
- *Public Transport Priority*
- *Variable Message Signs*
- *Incident Management*

The improvements to the *Public Transport Services* modelling consist of a better definition of routes, schedules and stops. Bus routes are described as a fixed series of links from an origin to a destination, schedules definition is frequency-based with possible random deviations, and a new type of bus stop is created : the bus stop lay-by, including new behaviour rules for pulling into or out at the bus stop.

A complete *Roundabout* model was implemented in SITRA-B+ ; this new development addresses both driver and vehicle behaviour models and graphical user interface functions ; simple rules were defined in order to deal with lane changing decisions both approaching and driving in the roundabout, and new behaviour parameters were introduced for the gap acceptance model. Video data from a test site in Toulouse has been used for the validation of this roundabout model.

The *Parking Management* model improvements mainly deal with street parking management : street parking (along the roadside) are no longer considered as destination or origin nodes, but as intermediary destination nodes with a given stopping probability ; a series of parking spaces at precise locations is attached to each street parking node (which is itself attached to a given lane) ; mean and standard deviations of parking duration are parameters which can be selected by the user for each street parking set.

The *Adaptive Traffic Signals* improvements consist in the implementation of the new traffic signals description and management protocols presented in Deliverable D4 « Update Specifications » ; it thus increases the range of UTC strategies able to be linked with SITRA-B+, such as the possibility to alternatively run fixed time plans and to interrupt them by adaptive sequences.

The development of new specialised detectors dedicated to public transport vehicles now allows us to consider a wider set of *Public Transport Priority* strategies that can be tested with the microscopic traffic simulator ; formatted messages are generated and stored in data files ready to be used by the external PT priority strategies.

As far as *Variable Message Signs* are concerned, a new class has been created for VMS modelling, and dynamic route guidance purposes were associated to this new object. Guidance controls to be displayed on the VMS are calculated and sent to SITRA-B+ by an external strategy, together with modified routes and compliance rate for concerned vehicles.

Finally, concerning *Incident Management* features, the possibility to generate scheduled incidents (location, occurrence time, duration) was added in SITRA-B+ ; this new feature is particularly well suited for testing UTC strategies robustness and ability to react to unpredictable events.

5.2 *PUBLIC TRANSPORT SERVICES*

5.2.1 Introduction

In the former version of SITRA-B+, it was possible to associate only one bus stop with a given bus route, and vehicles were generated according to a deterministic period (without random variations) ; stopping time was also constant, and only typical bus stops were modelled, thus causing systematic blockage for following traffic.
The new developments provide a more complete and more realistic description of *Public Transport Services*, both for route schedule, bus stop layout and pulling out behaviour.
Route schedules are still given by starting and ending time and a theoretical frequency, but a random parameter (standard deviation of the time period) is added in order to model the usual irregularities ; these parameters are used by the bus generation module ; a null value for the time period standard deviation would mean that the generation node is a terminus.
There is no longer any limitation on the number of bus stops per route ; each of them is attached to a given link and to a given route, and other parameters are the position on the link, the mean and standard deviation of stop time, and the layout parameter (*typical* or *lay-by*).
In the case of a bus stop lay-by, a pulling into and a pulling out behaviour model were implemented ; the pulling out model allows the following traffic to stop for a few seconds before the end of the bus stop time, and the pulling out manoeuvre takes place as soon as the lane is cleared along the bus stop location ; the animated graphical display enables the proper behaviour of the model to be checked.
A set of outputs is available at the end of a run, which can be used for example to analyse the effect of a given UTC strategy on the journey time and regularity.

5.2.2 Inputs

Three text files are used for *Public Transport Services* description in SITRA-B+ :

Route description data

File *vehicle_route.rel* contains the following route description data :
- route identifier
- list of network links along the route

Route schedule description data

File *vehicle_schedule.rel* contains the following schedule description data :
- route identifier
- vehicle modality (which refers to vehicle type and equipment level)
- origin node
- destination node
- starting time
- ending time
- mean value of time period
- standard deviation value of time period (to be applied by the generation module at the origin node)

Bus stop description data

File *bus_stop.rel* contains the following input parameters :
- bus stop identifier
- route identifier

- link identifier
- bus stop position on the link (downstream point)
- mean value of stop time
- standard deviation value of stop time
- bus stop configuration (*typical* or *lay-by*)

5.2.3 Processing

Three main functions were developed in order to improve *Public Transport Services* management in SITRA-B+ : vehicle departure and stop time generation, pulling into model and pulling out model for bus stops lay-by.

Vehicle departure and stop time generation

For each PT route, a table containing the future departure times is generated during the initialisation phase : each theoretical departure time is altered with a truncated Gaussian noise value, whose standard deviation value is given in file *vehicle_schedule.rel*.
A similar procedure is applied for stop times : each time a new bus is generated at an input node of the network, a list of stops is created, including the stop time value which is calculated as a truncated Gaussian value taking into account the mean and standard deviation values given in file *bus_stop.rel* ; as different values can be assigned to each bus stop, this allows the effect of disturbances generated by different levels of passenger demand to be evaluated.

Pulling into algorithm (bus stop lay-by)

When a bus reaches the stop position (the stop is supposed to be « reached » when the distance between the front of the bus and the stop position is less than a given threshold), it becomes « transparent » for the following traffic (case of a bus stop lay-by), which means that it is no longer considered as the preceding vehicle by the following car ; this procedure, which avoids adding supplementary lanes in the network description, offers a modelling capacity almost as complete as the one obtained with an explicit bus stop layout description. The graphical display of SITRA-B+ was modified in order to visualise the bus stop lay-by configuration (see next paragraph).

Pulling out algorithm (bus stop lay-by)

In order to warn upstream traffic before the pulling out manoeuvre, and so initiating the creation of a gap, the following procedure is implemented : at a given number of seconds before the end of the scheduled stop time (fixed parameter of the model, not to be changed by the user), the bus looses its « transparency » : incoming traffic therefore decreases speed, which naturally leads to a gap creation in fluid conditions. Then, when the stop time has elapsed, bus leaves its stop after having checked the presence of an acceptable gap on its lane ; if there is no acceptable gap (case of congested traffic situation), it waits until the queue is cleared.

5.2.4 Outputs

A sequence of screen copies from the animated graphical display of SITRA-B+ (UNIX Version) is shown in Figure 37, illustrating the various steps of a bus departure from a bus stop lay-by :

(a) : bus stopped ; incoming traffic running freely
(b) : incoming traffic being warned of bus departure
(c) : bus leaving the stop
(d) : a few seconds after bus departure

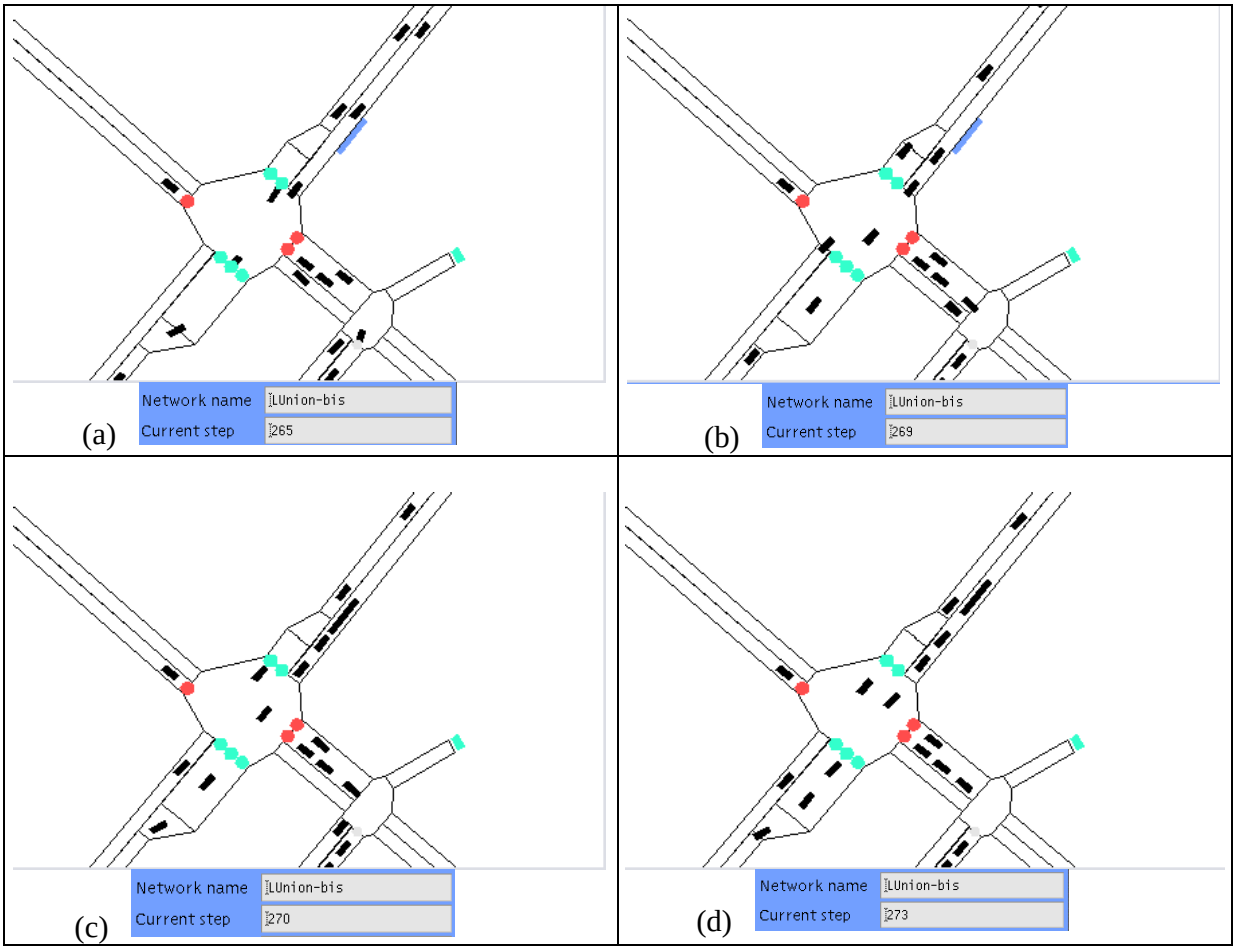


Figure 37 : The bus layby in operation

Output file *PT_route_output.rel* contains the following summary performance indicators for each PT service route :

- time slice hour
- route number
- number of generated vehicles
- number of vehicles which completed the route
- for all vehicles which have completed the route :
    - average speed
    - average journey time
    - minimum journey time
    - maximum journey time
    - average number of stops
    - total travelled distance

Bus stops summary data are stored in file *bus_stop_output.rel*, which contains, for each bus stop of each bus route, per time slice :

- time slice hour
- bus stop identifier
- number of observed intervals between buses
- mean time period between buses
- standard deviation of time period between buses
- number of stops

- mean stop time
- standard deviation of stop time

Moreover, it is always possible to get detailed indicators for each generated bus from file *vehicle_output.rel* [17] which includes :

- generation time and location
- route
- average speed and travel time
- number of stops and total stop time
- total stop time at bus stops (along the route)

5.2.5 Calibration results

Data available from Toulouse area have been used to validate the model ; validation data includes average journey time and bus stop data such as mean and standard deviation of time period between buses.

5.3 *ROUNDABOUT*

5.3.1 Introduction

The roundabout simulation model implemented in SITRA-B+ addresses « classical » or « conventional » roundabouts, as they are described in paragraph 5.2 of Deliverable D4 « Update Specifications ».
Topological description of roundabouts makes use of the existing basic network description structures of SITRA-B+ (links, intersections, link- and intersection-lanes), with a specific development related to the animated graphical display, which now allows curved links to be represented ; new data fields were introduced to distinguish between new link categories or shape and priority rules.
Three new behaviour models were introduced in order to take into account driver behaviour at different levels : in the approaching phase (lane choice and gap acceptance models), and inside the roundabout (lane changing model). In order for the gap acceptance model to work, a new stochastic parameter was added to the ones attached to each vehicle : the *aggressiveness* parameter ; the way it is used by this model is explained in section 5.3.3.
Special attention has been made to the validation of those various elementary models : this will be done by exploiting video recordings of a roundabout located close to the CERT offices in Toulouse.

5.3.2 Inputs

Three existing input data files were modified, and a new one was added for roundabout modelling :

Roundabout description data

New file *roundab.rel* contains the global description of the roundabouts, which means the number and the list of links and intersections belonging to a given roundabout. These data are mainly used for producing statistical results. The format of this file is :

- roundabout_number : integer value
- roundabout name : string
- list of links : list of links belonging to the roundabout (link numbers) ; both RDB_ENTRANCE and ROUNDABOUT type links have to be entered (see next file description)

[17] this file contains detailed information on each generated vehicle.

- list of intersections : list of intersection belonging to the roundabout (intersection numbers)

Link description data

File *link.rel* now contains the following supplementary parameters :

- link type ; possible values are :
  - RDB_ENTRANCE : for a roundabout entrance link
  - ROUNDABOUT : for a roundabout link
  - ORDINARY : for other links
- link length parameter : length parameter may be used for all types of links. It allows, for example, a series of bends to be represented by a straight link but using the real road length. If this parameter equals 0 then link length is computed from the connection point co-ordinates of the link lanes. In other cases, the following treatment is applied :

  Suppose that L represents link length computed from connection point co-ordinates :
  If *L < link length parameter < 2×L*
  Use link length parameter as link length
  Else
  Use L as link length
  Endif

- link radius : link radius parameter is used by the graphical interface to represent a curved link as a circular arc. If the link radius equals 0 then the link is supposed to be straight. The link radius parameter sign indicates the orientation in which the circular arc should be displayed from the upstream connection point to the downstream connection point :

  If link radius parameter sign >0
  The circular arc is displayed clockwise
  Else
  The circular arc is displayed counterclockwise
  Endif

  The following treatment is applied to the link radius value :

  If abs(link radius value) >= L/2
  Parameter is used to draw the curved link
  Else
  The link is supposed to be straight
  Endif

  In order to obtain the most appropriate display of two adjacent curved links, the user should pay attention to the choice of the link radius value, the radius value of the farside border of the farside lane of the link. Indeed, it is the common part of two adjacent curved links.

Priority rule description data

The SITRA-B+ microscopic description of a network is based on the use of *lanes* connected by *connection points* ; the priority rule to be used when changing lane (from a link lane to an intersection lane or vice-versa, not in the case of lateral lane changing) is thus given by the

connection point nature ; a new type of connection point was introduced in file *connection.rel* : the LEFT_PRIORITY one.
Figure 38 shows an example of a roundabout layout : it includes three roundabout entrance links, 4 roundabout links, 3 outputs (« ordinary » links) and 4 intersections with their associated intersection lanes ; red dots show the location of LEFT_PRIORITY connection points.

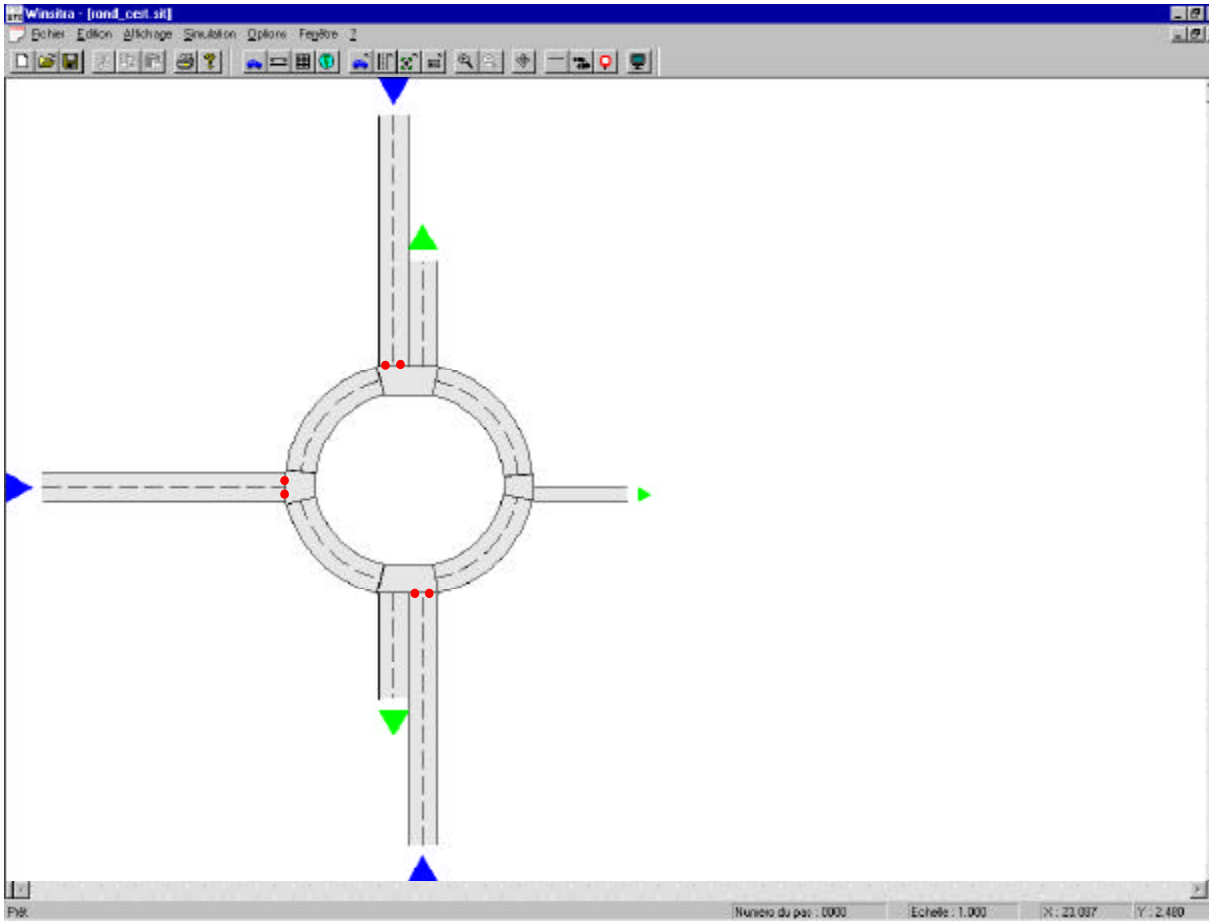


Figure 38 : Roundabout layout

Behaviour model data

The user of the SITRA-B+ simulation tool can adjust two sets of parameters so that the simulated behaviour matches the observed one.
The first one concerns aggressiveness modelling, which is a new individual stochastic parameter attached to each generated car in the network ; the proposed scale goes from 0 to 10, and the user can specify its mean and standard deviation values by vehicle category ; this parameter is then used to derive other attributes of the gap acceptance model.
The second set of parameters allows the user to choose the maximum and minimum values of the gap acceptance time (see also next paragraph for a more precise definition) ; those values are also given by vehicle category, and default values will be proposed to the user.
All those new input data, which have to be entered in file *vehicle_type.rel*, are thus the following ones :

- mean value of aggressiveness (0 to 10)
- standard deviation value of aggressiveness
- minimum gap acceptance time
- maximum gap acceptance time

5.3.3 Processing

New procedures have been implemented in SITRA-B+ to model the behaviour of drivers approaching and driving inside roundabouts ; they have been added to the model in such a way that they do not interfere with existing procedures, which for example apply conflict rules inside intersections.

Lane choice model

The first behaviour model is the lane choice model for vehicles entering the roundabout ; the choice depends on the position of the exiting link, which implies that the vehicle destination and route have to be known ; the following algorithm is implemented in procedures *next_lane()* and *test_rdb_path()* for each vehicle entering a new link :

If *the vehicle is entering a RDB_ENTRANCE type link*
If the vehicle route follows only one ROUNDABOUT type link
If the vehicle micro-route follows the farside lanes of those two links
Compute a new micro-route for the vehicle, following the nearside lanes
Endif
Endif
If the vehicle route follows more than two ROUNDABOUT type links
If the vehicle micro-route follows the nearside lanes of the RDB_ENTRANCE type link and of the first ROUNDABOUT type link
Compute a new micro-route for the vehicle, following the farside lanes
Endif
Endif
Endif

This first approach assumes simple roundabout layouts, typically four-link roundabouts and two-lane links ; the results gained from this model will allow to consider later more complex layouts.

Gap acceptance model

Figure 39 illustrates the general principle of the gap acceptance model.

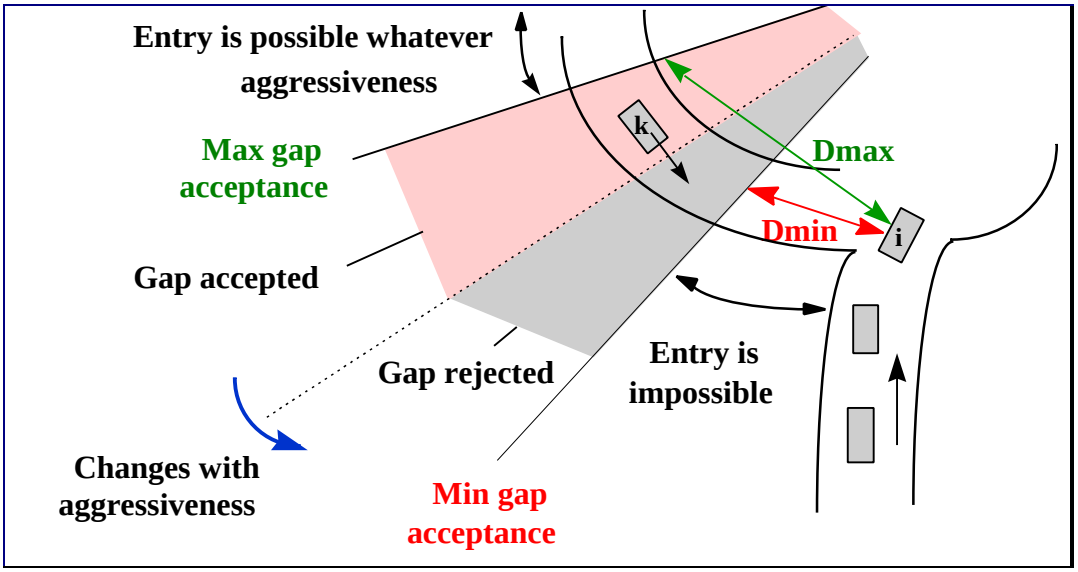


Figure 39 : Roundabout gap acceptance model

For a better understanding, the gap areas have been represented as distances and geometrical areas. They must however be transposed into time co-ordinates, where the previously defined parameters *minimum gap acceptance time* and *maximum gap acceptance time* (see paragraph 5.3.2) would be the transposed values of Dmin and Dmax in Figure 39. These distances are of course strongly dependent of the speed of approaching vehicle **k**.
The dotted line shows the initial value of the gap requested by entering vehicle **i** ; this value is derived from the aggressiveness parameter attached to this vehicle when it was generated in the network ; if vehicle **i** does not succeed in entering the roundabout with this initial gap acceptance value, and thus spends time queuing at the roundabout entrance, this value is reduced at a fixed rate until the minimum gap acceptance value is reached.
For each vehicle approaching a roundabout, a « decision distance » is calculated, from where it has to decide whether or not to enter the roundabout, and so to apply the gap acceptance model ; this distance corresponds to the stopping distance of the approaching vehicle.
If the vehicle is not authorised to enter the roundabout, a « virtual stop » is generated at the LEFT_PRIORITY connection point located at the roundabout entrance ; then, at each time step, this vehicle keeps checking the gaps until entrance is possible.

Lane changing model (inside the roundabout)

This model mainly concerns the lane changing from the farside to the nearside lane, for vehicles which have to drive more than a half circle ; the proposed algorithm simply detects when the vehicle enters the last ROUNDABOUT link of its route, and computes a new micro-route following the nearside lane, as explained previously in the lane choice model ; nevertheless, if there is a path towards the desired exit using the farside lane and if the nearside lane is congested, the vehicle can keep to the farside lane until the exit.

5.3.4 Outputs

The animated graphical display of SITRA-B+ allows the simulated behaviour of vehicles approaching and driving in the roundabout to be checked, and a comparison to be made with the real behaviour, using for example video recordings of a roundabout.

Concerning text outputs, new file *roundab_output.rel* is added, which puts together the following results per roundabout :

- time slice hour
- roundabout number
- roundabout name
- number of trips having entered the roundabout
- number of trips having left the roundabout
- average speed on the roundabout
- number of lane changing in the roundabout

Results relating to link (specially entry links) can be found in existing file *link_output.rel*, where the following four supplementary fields were introduced:

- mean value of gap acceptance times, observed from vehicles when entering the roundabout
- standard deviation value of gap acceptance times
- mean value of queuing time
- standard deviation value of queuing time

Finally, new file *rdb_trip_output.rel* was added, which contains results per origin-destination pair of the roundabout:

- time slice hour
- input link number (trip origin)
- output link number (trip destination)
- vehicle type
- number of vehicles having achieved the trip
- mean value of travel time
- standard deviation value of travel time

5.3.5 Calibration results

The chosen area is a Grade-Separated Interchange with One Bridge and two Roundabouts (see the terminology adopted in model update specifications for roundabout). One of these roundabouts has 3 entries (each with two lanes) and 3 exits (2 with two lanes and 1 with one lane). Furthermore it has 1 segregated lane that allows a part of traffic to go from an entry to the first exit. This roundabout is represented in Figure 40.

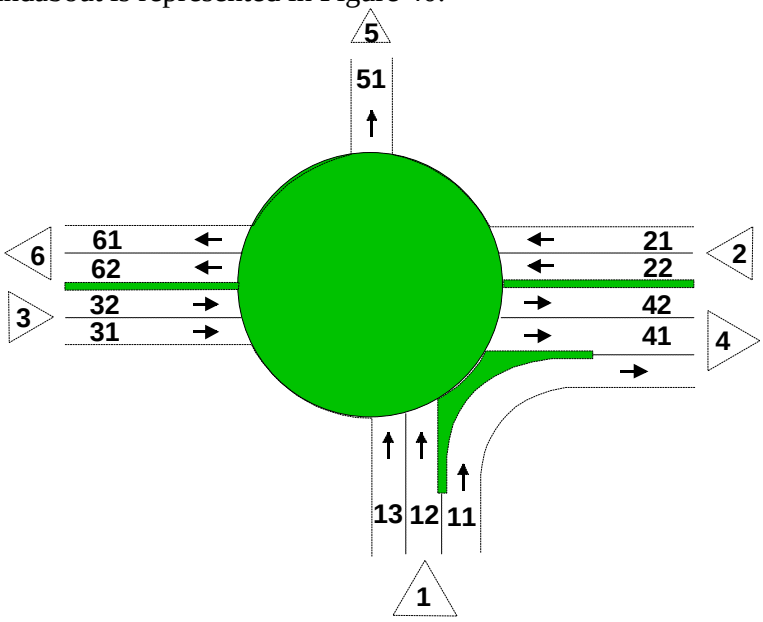


Figure 40 : The test roundabout

Entries and exits are represented with triangles. Entry and exit numbers are represented in each associated triangle. Lane direction is represented with a black arrow. Lane number is indicated on each lane.

Data collection has been performed by video from a point where all entries/exits are visible and with a video recording. Video data analysis has determined :

- traffic flow for each entry/exit
- for each Origin/Destination pair
  - traffic flow
  - average travel time
  - lane choice at roundabout entrance
  - lane changing inside the roundabout
- average travel time inside the roundabout
- average gap acceptance time for each entry
- driver behaviour near each entry

A sample of these data has been used to tune the roundabout model to represent real traffic conditions (lane choice, lane changing and gap acceptance). The remaining data has been used in simulation to check that the roundabout model developed in SITRA-B+ performs close to real conditions.

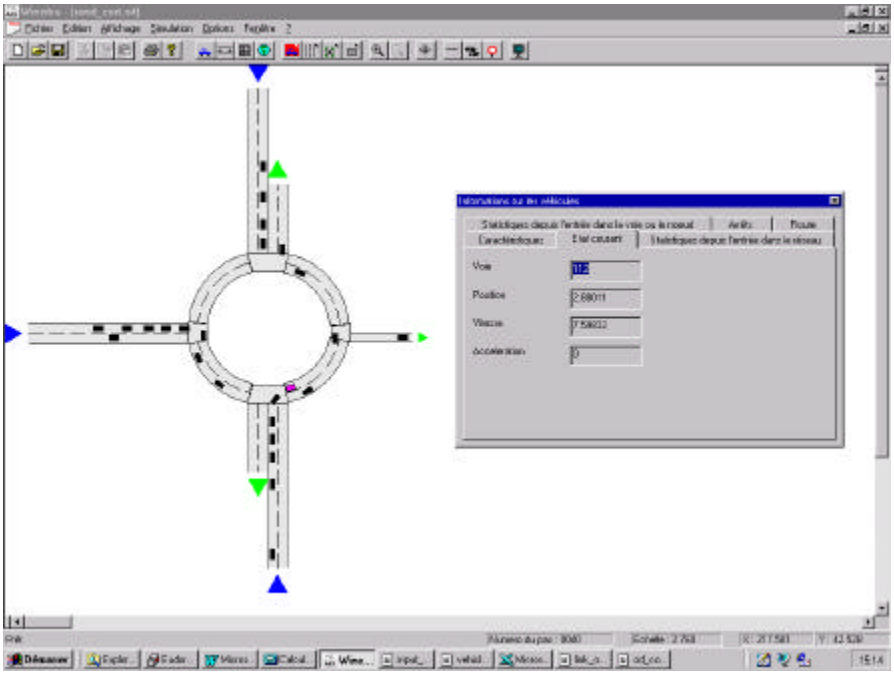


Figure 41 : The test roundabout in SITRA-B+

Table 5 shows the initial results obtained with the roundabout model of SITRA-B+, in order to check its correctness. In particular this testing relates to the gap acceptance model. Two different sets of values for minimum and maximum gap acceptance time were applied to the same demand on the test roundabout (5 minute time slice).
Those results show the significant sensitivity of the model to the gap acceptance time values, especially on the queuing time of vehicles approaching the roundabout on entry 2, without affecting the driving behaviour inside the roundabout (no significant difference on average speed and number of lane changing).

	Case 1	Case 2
min/max gap acceptance time (seconds)	1.0/3.0	2.0/5.0
mean value of observed gap acceptance times (entry 2)	1.77	3.14
standard deviation value of observed gap acceptance times (entry 2)	0.35	0.55
mean value of queuing time, in seconds (entry 2)	4.71	14.51
standard deviation value of queuing time, in seconds (entry 2)	9.87	31.80
number of vehicles having entered the roundabout	171	168
number of vehicles having exited the roundabout	164	163
average speed in the roundabout, in km/h	25.5	24.9
number of lane changing	42	43

Table 5 : How the gap acceptance time affects queueing time

Further data analysis gave the following results:

Traffic flow for each input (veh/h)

Input number	Simulated flow	Observed flow	Difference
1	1164	1296	10,2%
2	1068	1200	11,0%
3	468	612	23,5%
Total	2700	3108	13,1%

Traffic flow for each output (veh/h)

Output number	Simulated flow	Observed flow	Difference
4	216	240	10,0%
5	948	1440	34,2%
6	1080	1428	24,4%
Total	2244	3108	27,8%

Traffic flow for each O/D (veh/h)

O/D number	Simulated flow	Observed flow	Difference
1 - 5	384	576	33,33%
1 - 6	612	720	15,00%
2 - 5	372	516	27,91%
2 - 6	432	684	36,84%
3 - 4	216	240	10,00%
3 - 5	192	348	44,83%
3 - 6	24	24	0,00%

The results obtained from the simulation of this roundabout show differences in traffic flows from 0% to 44%. It is noticeable that the simulated flows at all entries are always less than the observed flows at the same points. This is due to the fact that the queues at entries were building (video is recorded at peak hour and queues are long at all entries) and the simulated roundabout is not able to let out as many vehicles as in real life. The "aggressiveness" and the "gap acceptance" parameters have to be adjusted in SITRA-B+ to consider the behaviour of drivers who are used to crossing this particular roundabout. The video tape observation also shows a more complex lane choice and lane changing behaviours than the simulated one which contributes to limit the capacity.
However, travel times for vehicles crossing the roundabout, as well as the general behaviour of drivers within the roundabout were satisfactory.

5.4 *PARKING MANAGEMENT*

5.4.1 Introduction

This feature, which already existed in the preceding version of SITRA-B+ in a simplified form, has been enhanced to offer a more realistic representation of on-street parking. The car park type which is considered here is the « along the roadside » one (see Deliverable D4 page 55).
Two main improvements were done. The first one concerns the integration of the use of the street parking area by vehicles in relation with their trip inside the simulated network : street parking is no longer considered as elementary origin or destination nodes, but as « intermediary » nodes in the vehicle route description. This allows a more precise description of the street parking itself : parking spaces are clearly positioned along the street, and a procedure similar to the one developed for bus stop lay-by is used to model vehicle manoeuvres to occupy or free the parking space.
The user can assign mean and standard deviation values of parking duration to each street parking area, and the desired parking time of a given vehicle is a stochastic value drawn from the associated Gaussian law. The effective parking time can be greater during congested traffic

situations. In order to be able to model various behaviours related to the parking manoeuvre, a « pulling in » duration can be specified by the user (a default value is proposed).

5.4.2 Inputs

Two existing input data files were modified :

Route data

The following data fields were added to file *route.rel* in order to specify one or more parking stops for a given route :

- number of parking nodes along the route
- list of the node numbers
- list of attached stopping probabilities

Street parking data

The new format of file *street_park.rel* is the following one :

- node identifier : integer value
- link identifier : integer value
- vehicle type : name of the vehicle category using the parking spaces
- starting point : position of upstream extremity of the parking area
- ending point : position of downstream extremity of parking area
- mean value of stopping time : real value , in seconds
- standard deviation value of stopping time : real value , in seconds
- mean value of pulling in time : integer value, in seconds

The abscissa of starting and ending points must be compatible with the link length (checked by the program) ; the number of parking spaces is automatically calculated from those abscissa and from the length of the vehicle type.

5.4.3 Processing

Two main steps can be distinguished in parking modelling : *getting in the car park* and *getting out of the car park.*

Getting in the car park

Each time a vehicle enters a new link, the following algorithm is applied :

If *the link is associated to a street parking node*
    If *this node is the next node in the list of parking nodes of the vehicle route*
        Decide if vehicle will stop (using stopping probability attached to the node)
        If *vehicle is going to stop*
            Calculate (if necessary) a new micro-route using the nearside lane of the link
            If *no free parking space is available*
                Renounce to decision to park and continue the trip
            Else
                Choose randomly a parking space among the free spaces
                When the chosen space is reached, leave the vehicle stopped on the lane during pulling in time value, then put it in the parking space
            Endif
        Endif
    Endif
Endif

The procedure used for parked vehicles is the same as that used for the bus stop lay-by (« transparency » indicator).

Getting out of the car park

The same algorithm as the one used for buses to pull out from a bus stop lay-by is used (see section 5.2.3. As soon as a space is freed, it is added to the list of free parking spaces for incoming vehicles.

5.4.4 Outputs

In addition to the animated graphical display of SITRA-B+, which can be of great help in the verification phase, file *park_output.rel* was expanded ; it now contains the following indicators per street park :

- time slice hour
- street park identifier
- average occupancy (percentage)
- maximum occupancy
- mean parking time per vehicle
- standard deviation of parking time
- number of unsuccessful parking attempts

5.4.5 Calibration results

No validation data being available for this function, only verification tests are being made , using scenarios similar to the ones described in section 6.7 of Deliverable D4 (page 118).

5.5 *ADAPTIVE TRAFFIC SIGNALS*

5.5.1 Introduction

In accordance with the Update Specifications of Adaptive Traffic Signals presented in Deliverable D4 (see page 99 and following), the new developments made in SITRA-B+ concern traffic signal modelling and traffic controller description and operation.
Concerning the modelling part, a new data structure now has the « colour » class as the basic class for traffic signal description, and, if necessary, to it is possible to associate a dedicated behaviour to each colour ; the derived classes then lead to the new controller and fixed time plan representations.
Concerning traffic controller operations, the implemented procedures enable the full exploitation of the new plan description (impulse based) and thus increase the range of strategies to be linked with SITRA-B+ (it is of course assumed that adaptive strategies are external entities).

5.5.2 Inputs

The organisation of the input data related to traffic signals and controller operation was modified in order to deal with the new specifications. Two sets of input files can be distinguished : the first one (four files) describes the relations between the traffic signals and the network, the traffic controllers structure and their fixed time operation, and the second one is related to the data exchange process with external adaptive strategies (two files).

Colour data

New file *colours.rel* contains the following data fields :

- colour identifier
- behaviour identifier

The behaviour identifier is used by the simulation program to process the first upstream vehicle on the lane facing a traffic signal displaying this colour : for example, with the « blinking amber arrow », a give-way rule similar to the one used at roundabout entrances has to be applied, for the associated turning movement.

Traffic signal data

Existing file *traffic_signal.rel* has been modified ; its new structure is the following one :

- traffic signal identifier
- intersection identifier (node number)
- list of connection point numbers (which are controlled by this signal)
- list of « transient » colours supported by the signal, and, for each of them :
    - duration
    - preceding and following « stable » colour identifiers

Traffic controller data

New file *traffic_controller.rel* has been created, which contains the description of stage sequences at a given intersection, according to the specifications proposed in Deliverable D4 ; it enables to link impulses with all the traffic signals belonging to the intersection ; its structure is the following one :

- traffic controller identifier
- intersection identifier
- type of strategy : internal (fixed time plan) or external
- list of stage impulse identifiers
- for each traffic signal belonging to the intersection, and for each impulse :
    - next « stable » colour identifier
    - associated time-lag

Initial signal plan data

New file *initial_signal_plan.rel* contains the time data describing the fixed-time plan to be applied at the beginning of the simulation run (and during the whole run if no external strategy is used) ; its contents is :

- cycle time
- for each traffic controller :
    - traffic controller identifier
    - list of impulse times

External strategy data

Two files are used by the external UTC strategy to send the new traffic signal settings to SITRA-B+ ; file *external_signal_plan.rel* allows the implementation of a new fixed time plan, and file *impulse.rel* is used when the external strategy is taking full control of traffic signals on an intersection.

File *external_signal_plan.rel* contents is the following one :

- plan identifier
- cycle time
- synchronisation times : real time value associated to plan reference time
- for each traffic controller :
    - traffic controller identifier
    - list of impulse times

File *impulse.rel* contents is :

- controller identifier
- impulse identifier
- impulse time

5.5.3 Processing

An event-driven type approach is used to implement the new traffic signals management in SITRA-B+ ; two types of events are associated with controllers and traffic signals : impulse occurring time and colour changing time ; at each simulation step, intersection controllers are investigated, and the following procedure is applied to each of them :

For *each traffic signal* ***do***
    If *time == next colour changing time*
        Process colour changing
    Endif
Enddo
If *time == next impulse occurring time* ***then***
    Calculate next impulse time (case where a fixed time plan is running)
    For *each traffic signal* ***do***
        Process colour changing (if required)
        Calculate next colour changing times and next colour values
    Enddo
Endif

The data fields contained in file *traffic_controller.rel* are used to determine the next colour changing times and colour values when an impulse is processed ; transient colours changing times are also automatically calculated, using file *traffic_signal.rel* data.
The advantage of this approach is that the same procedure is used both with fixed time plans and external adaptive strategies ; the only difference concerns the next impulse time calculation, which holds only when a fixed time plan is running.
As there is no input data file to allow the direct initialisation of the traffic signal states, a start-up procedure was introduced to achieve this task, simply running a blank cycle before starting the simulation itself.

5.5.4 Outputs

The traffic signal colour changes can be observed on the graphical user interface of SITRA-B+ ; the PC version also allows the current parameters associated with a traffic signal or with an intersection controller to be displayed.
In addition to the usual traffic level indicators edited by SITRA-B+ in file *intersection_output.rel* for each intersection, a new output file was added - *controller_output.rel* - which collects statistics on stage duration per traffic controller ; its contents are as follows :

- mean, minimum and maximum cycle time values per time slice
- for each traffic signal, mean, minimum and maximum green time values per time slice

These last data are of course especially significant in the case of adaptive strategies.

5.5.5 Calibration results

Validation tests here amount to verification tests, in order to check that simulated controllers correctly execute the phase sequences associated with the current fixed time plan, or controlled by the impulse type orders sent by the external strategy. There is no special need in this case of field trial measurements.

5.6 *PUBLIC TRANSPORT PRIORITY*

5.6.1 Introduction

The main developments which were undertaken to improve *Public Transport Priority* management with SITRA-B+ are related to the modelling of Public Transport vehicle localisation procedures and to the implementation of a new detector type ; as in the case of Adaptive Signals, the strategy producing or altering the traffic signal settings is considered as an external strategy, which can in this case receive new types of messages produced by the bus localisation procedure.

5.6.2 Inputs

Two input files are modified in order to implement the new *Public Transport Priority* functions :

PT vehicle localisation system

The following data fields are added to file *modality.rel* :

- localisation system existence : true or false

If the preceding value equals true :

- localisation system type : absolute or relative
- localisation model parameters :
    - mean and standard deviation of the absolute positioning error, in metres (case of an absolute positioning system, such as GPS)
    - mean and standard deviation of the localisation drift, in metres per kilometre (case of a relative positioning system, such as an odometer based system)
- communication process mode : asynchronous or synchronous

Localisation beacons

A new type of beacon is introduced : the « localisation beacon » ; this is simply done by adding a new type of detector in file *sensor.rel* : the LOC_BEACON type (the existing types are LOOP - for flow and occupancy measurements - and BEACON - used for dynamic route guidance).

5.6.3 Processing

The choice of an « asynchronous » communication mode in file *modality.rel* for a given vehicle category means that the vehicle position will be known by the external strategy only when it reaches a LOC_BEACON type detector : there is thus no need of position calculation in this case.

When a « synchronous » communication mode is chosen, the vehicle is supposed to transmit its position at regular time intervals : functions *give_position_abs* or *give_position_rel* are thus activated, depending on the localisation system type :

- function *give_position_abs* : a random value, derived from the localisation model parameters given in file *modality.rel* is added to the true vehicle position
- function *give_position_rel* : a random value of the odometer drift is calculated using the corresponding parameters and attached to the vehicle when it is generated in the network ; this value is then multiplied by the distance travelled by the vehicle since the last LOC_BEACON crossing, and added to the true vehicle position ; LOC_BEACON are thus in this case « resetting » beacons for the travelled distance.

5.6.4 Outputs

A new icon is introduced to represent localisation beacons on the network links ; most outputs are appearing in text files ; those related to PT services or traffic signals performance can be found in output files referenced in sections 5.2.4 (*Public Transport Services*) and 5.5.4 (*Adaptive*

Traffic Signals), and a supplementary output file is added in order to keep trace of the localisation message sent by the vehicle : file *localisation_output.rel*, whose contents is :

- vehicle identifier
- message time
- link number
- real vehicle position in the link
- estimated (transmitted) vehicle position in the link

5.6.5 Calibration results

As no bus priority experimentation is planned to occur on the selected test sites, only verification tests can be held ; file *localisation_output.rel* allows the correctness of the simulated localisation procedures to be checked.

5.7 *VARIABLE MESSAGE SIGNS*

5.7.1 Introduction

The modelling capabilities of SITRA-B+ concerning dynamic information and guidance systems have been extended in order to be able to deal with Variable Message Signs. As for on-board route guidance systems, already supported by SITRA-B+, guidance messages are assumed to be composed by an external strategy, together with the new proposed routes and the estimated compliance rate.
All vehicles passing by the VMS location see and read the message, which displays a route guidance type message (concerned destination, advised route) ; VMS are modelled as a new kind of beacon, and a new function is added at the vehicle level in order to first identify the concerned vehicles (going towards the same destination but using a route different from the one advised), and then divert them, taking into account the obedience coefficient proposed by the external strategy.
VMS locations are displayed on the graphical user interface, together with associated messages ; new output text files are introduced to check the effects of the collective route guidance strategies.

5.7.2 Inputs

Three input files were modified or created in order to model VMS guidance strategies messages : the first two ones are related to the VMS itself, and the second one to the description of the messages sent by the external strategy.

VMS description

There is a strong similarity between a BEACON object and a VMS object (they are located on the roadside, and act on driver behaviour when drivers see the message), file *sensor.rel* is still used to enter the *Variable Message Signs* objects ; the new VMS type is then added, and the meaning of the attached data fields is as follows :

- VMS identifier
- lane identifier
- position : distance from the beginning of the lane
- range : defines the distance from which the VMS can be read
- type : VMS

A new data field is also added in file *modality.rel*, in order to know if a given vehicle category is globally influenced by the VMS (for example, buses are usually not influenced).

VMS messages

File *VMS_commands.rel* is created ; this file, which is sent by the external strategy to SITRA-B+, contains the following information :

- VMS identifier
- advised destination identifier : string
- advised destination number : node number
- advised route identifier : string
- advised route description : list of links
- compliance rate : real number (0.0 to 1.0)

Advised destination and route identifier are strings which are mainly used at the graphical interface level, in order to display the message contents, while destination number and route description are directly used by the simulation model for selecting those vehicles which can be influenced by the message and their diversion ; the string associated with the advised route description is however also used by the simulation model, in order to find out if a relevent vehicle passing by the VMS is already using or not the advised route : this string is in fact the name of a significant link belonging to the advised route, which can be also found in file *link.rel*.

5.7.3 Processing

Each time a vehicle belonging to a category which is able to react to a VMS message (see file *modality.rel*) passes by a VMS location, the following procedure is applied :

If *the vehicle destination is the advised one*
    If *the initial vehicle route does not use the advised one*
        Generate a random number between 0.0 and 1.0
        If *this number is less or equal to compliance rate*
            Assign the vehicle to the advised route
        Endif
    Endif
Endif

5.7.4 Outputs

VMS locations are shown on the Graphical User Interface using special icons, and the current message can be displayed by clicking on it.
Two output files were added to present VMS results. The first one is used to check the VMS effects on the drivers, and the second one to assess the time benefits or loss of diverted vehicles, compared to non diverted ones.
The contents of file *VMS_ouputs.rel* is :

- VMS identifier
- total number of vehicles having passed by the VMS location
- total number of vehicles concerned by the advised destination
- total number of vehicles likely to divert (concerned by the advised destination with an initial route different from the advised one)
- total number of diverted vehicles

File *diversion_output.rel* gathers mean travel time values encountered by vehicles having passed by VMS locations, with the same destination as the advised one ; those travel times are taken from the VMS to the destination, and divided up into two categories : those following the advised route, and those following all other routes. In order to be able to perform a detailed analysis of

the performance of the strategy, travel times are also classified by time periods (between message change) :

- VMS identifier
- advised destination identifier
- advised route identifier
- message start time
- message end time
- mean travel time on advised route, encountered by vehicle having passed by the VMS location during the period
- mean travel time on other routes, encountered by vehicle having passed by the VMS location during the period

5.7.5 Calibration results

A scenario conducted on the Toulouse test site (see Deliverable 4, page 22), using 10 urban VMS, was used to check these new VMS dedicated functions ; the strategy computes guidance recommendations (to turn right, left or to go straight on at the next intersection for a given destination) for all signs.

5.8 *INCIDENT MANAGEMENT*

5.8.1 Introduction

In SITRA-B+, the new developments related to *Incident Management* only deal with incident generation ; we do not address strictly speaking incident management strategies, but rather consider how UTC strategies can react to unpredictable events such as incidents.
Incidents are modelled by stopping vehicles at given times and at given locations, which remain stopped during a given duration.
Scenarios implying incidents are deterministic, which means that incident location, time of occurrence and duration are pre-defined by the user using a special input file.

5.8.2 Inputs

File *incident.rel* contains the following deterministic incident data :

- incident identifier
- lane number
- incident location on the lane
- starting time
- duration

5.8.3 Processing

At the beginning of the simulation, file *incident.rel* is read and incident starting times are put in an ordered list of scheduled incident generation events.
At the beginning of each time step, current time is compared to the next scheduled incident generation time ; if this time is reached, the first upstream vehicle driving on the concerned lane and able to stop at the incident location is looked for, and a "virtual stop" is attached to it; the vehicle is linked to the processed incident of the incident list and, at incident ending time, the "virtual stop" is reset.

5.8.4 Outputs

Concerning numerical results, the contents of files *lane_output.rel* and *link_output.rel* (average and maximum occupancy, average speed and travel time, number of stops and stop time) are used to evaluate the consequences of incidents and the way UTC strategies react to them.

5.8.5 Calibration results

Scenarios have been generated to process verification tests . It was be particularly interesting to check the lane changing behaviour of incoming vehicles, in the case when not all the lanes belonging to the same link are blocked by incidents.

6 CONCLUSIONS

This document fulfils to the second objective of the SMARTEST project: investigate how the SMARTEST models can best be enhanced to fill the identified gaps, and it is the result of Workpackage 4, Modelling.

Four members of the consortium have enhanced their micro-simulation packages: AIMSUN2 (UPC), DRACULA (ITS), NEMIS (Mizar) and SITRA-B+ (CERT). Each one has selected a set of improvements that have been implemented and calibrated or verified within their models. The following table is a summary of the new features implemented by each partner:

Feature	**AIMSUN2**	**DRACULA**	**NEMIS**	**SITRA-B+**
Roundabouts		✓		✓
Public Transport Services		✓	✓	✓
Traffic Calming		✓		
Incident Management	✓			✓
Ramp Metering	✓			
Adaptive Traffic Signals	✓	✓	✓	✓
Public Transport Priority		✓	✓	✓
VMS	✓		✓	✓
Dynamic Route Guidance	✓		✓	
Parking Management				✓
Detectors		✓	✓	
Results Analysis Tool	✓			

This distribution covers many of the gaps identified in Workpackage 2 and allows most of the new features to be compared against at least one alternative implementation in another SMARTEST tool.

Each developer has verified that the new features incorporated within their models functions well. When real data has been available, a calibration process has also been done. However, in some situations the very detailed and specific real data required has not been available from the test-sites and only a verification process has been feasible.