PERIODICA POLYTECHNICA SER. TRANSP. ENG. VOL. 33, NO. 1–2, PP. 103–124 (2005)

BUILDING A TRAFFIC SIMULATION SOFTWARE AND A NEW CONCEPT FOR INTEGRATION OF BEST SIMULATORS

Gábor SZŰCS

Department of Information and Knowledge Management
Budapest University of Technology and Economics
International McLeod Institute of Simulation Sciences Hungarian Center
H–1521 Budapest, Hungary
Phone: (36 1) 463-1417, Fax: (36 1) 463-4035
e-mail: szucs@itm.bme.hu

Received: March 1, 2004

Abstract

Applications by simulation are known worldwide, and more and more complex systems can be analysed by this tool because of rapid development of computers. The traffic modelling, as application field is in similar state, there are currently at least 50 traffic simulators in Europe with varying qualities of functionality. Rather than developing one more traffic simulator, it would be more useful to develop an open framework, which would be used to integrate the best features of the existing simulators. Additionally, users are interested in the simulation of specific events, for example how can a stopped vehicle in a lane affect to the rest of the network, etc. This publication presents more detailed requirements for generic framework modules for entering the road network data, visualization, analysis, and high performance simulation and shows a developed framework with these requirements.

Keywords: traffic simulation, open framework, UML, OO Simulators.

1. Introduction

The reader is guided through the structure of the document. After the introduction, section 2 presents Object Oriented Tools for simulators, in the part 3 there are shown different types of traffic models and tools. In section 4 the new concept is presented and part 5 shows a little problem as an example, solved by this new open framework.

In some EU countries the governments and local authorities have identified the use of traffic simulation tools as a way of enabling transport policies that can reduce pollution in cities, improve traffic safety on the roads, and ensure that safety requirements are traceable through the decision-making process. Traffic experts, engineers needed toolkits for modelling and simulation of urban road traffic operation, for scale-up, design of new functionality, and to improve the operation of existing systems. They have identified computer modelling and simulation of traffic operation and control as a strategic technology. The users of traffic simulators did not want yet another traffic simulator; they demanded an open framework to integrate the best features of the existing simulators and traffic control systems, and to support the interchange of data between these systems.

The motivation behind this was that experts needed to share their experiences, and to create a knowledge base from which the best and most relevant existing technology could be considered for inclusion in a new framework. An international project (OSSA) was to develop this standard open road traffic simulation framework. OSSA (Open Framework for the Simulation of Transport Strategies and Assessment) was a project in the Competitive and Sustainable Growth Programme of the Commission of the European Communities DG TREN. OSSA was not developing a new simulation model, OSSA was defining and demonstrating a new framework.

Experts had many requirements: some were only partially met by the existing tools and methods available to transport planners, traffic engineers and city policy-makers. The OSSA framework now provides interconnectivity between traffic simulators, traffic control systems, and data sources. This system offers new capabilities at the tactical level of implementing and evaluating policies relevant to road networks, road traffic, transport planning and vehicle emissions.

The implementation of the two-way relationship (see *Fig. 1*) between issues of broad policy in traffic and transportation and the tools by which the effects or impacts of such policies was important.

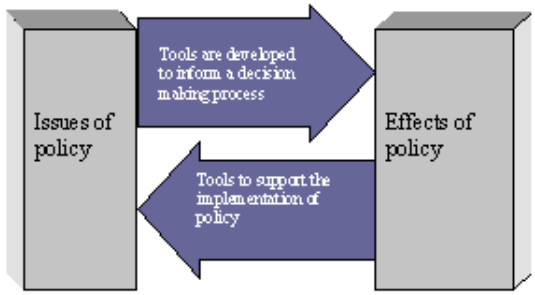


Fig. 1. Two-way relationship between issues of policy and the tools

The first type of tools – they are developed to inform a decision making process – underlies the development of strategic policies. For example:

- Reduction of emissions policy requires measurement and modelling.
- Increased safety requires accident data and modelling.

Tactical level policies are means of achieving strategic level policies. They require tools to support the implementation of the policy. For example:

- Influencing mode-share in favour of public transport, using tools such as public transport priority on the road and at intersections, and better and timelier information on schedules and service operation.
- Minimizing delay in a network, using such tools as adaptive traffic control systems and better and timelier information on traffic conditions and route guidance.

2. Technologies for Developing an Open Framework

2.1. Object-Oriented Tools for General Simulation Purpose

There are plenty of different object-oriented tools (based on object-oriented languages) or libraries used for general simulation purpose. For planning and developing a general framework described above for traffic simulation, it is important to have an overview of these tools. In this section Awesime, C++SIM, Maisie, ModSim, MOOSE, PROSIt, Ptolemy, SIM++, SimPack, and Simex are presented. All of these tools, except ModSim, use the general-purpose object-oriented computer programming language C++ as the simulation language. Of course, this is not a complete list, and a lot of other tools are available, but they can be seen as enabling a framework for simulation.

Awesime

Awesime is a process-oriented simulation library, and also a library for parallel programming in a shared address space (a multiprocessor computer). Entities are implemented using threads, and interactions between entities are made through method calls and the synchronization is solved by a special kind of semaphore [21].

C++SIM

C++SIM is very similar to Awesime, but distributed simulations cannot be built. It was inspired by the SIMULA language and adds SIMULA functionalities to C++. One original point for this tool is that 'non causal' classes can handle events, such as interrupts [27].

Maisie and MOOSE

MOOSE stands for Maisie Object-Oriented Simulation Environment. It is a process-oriented library with more structured interactions between entities: it uses message exchange, and the synchronization goes through a conditional queue of messages. The conditional recovery is a Boolean expression that can be evaluated using data from the state of the entities and from the messages. Finally, a parallel version of a simulation can easily be created due to the inheritance mechanism [5], [7].

ModSim II

ModSim (a commercial tool) is a simulation language developed by the US Army for interoperability purposes between their different simulation tools. It is similar to Awesime or C++SIM, but it uses a specific object language based on Modula-2. However, there are some restrictions on the object-oriented features: for example, a method can be overridden by another method only if it takes the same parameters. Activities (through threads) are created with a specific method, and the synchronization is made with special events [6].

PROSIT

PROSIT is a process-oriented library created with parallelism in mind. Activities are implemented using active objects which can easily be distributed on a network of workstations. A PROSIT simulation can be thought of as a collection of concurrently active objects interacting, via service calls, in the simulated time. If two active objects have to synchronise, they use a blocking service. The advantages of this kind of architecture are the capabilities provided for distributed and parallel simulations and the openness to various modelling formalisms [35].

Ptolemy I and II

Ptolemy I uses C++ as a simulation language and Ptolemy II is based on Java. Ptolemy is an event-oriented library for modelling reactive and embedded systems but it can also be used to simulate more general systems. The main objective of Ptolemy's developers was to offer a homogeneous environment that can simulate discrete event models but also continuous-time models with differential equations. In Ptolemy, a simulation is seen as a set of processes that have to be executed: The series of executions relies on criteria evaluated by a scheduler [9].

SIM++

SIM++ is (a commercial product developed by Jade) a process-oriented library with a different synchronization mechanism. When a SIM++ object wants to synchronize, it has to wait for an event. In this case, it is a really simple mechanism because an object can only wait for the next event, other tools allow waiting for an event from a specific class. SIM++ can be used to make distributed simulation [4].

Simex

Simex is quite different from the previous tools. It is a process and event-oriented library with several activities in the same entity. Synchronization and interaction are made through events. This tool was originally developed for micro-population simulations with an emphasis on human populations and diseases spread. It can, however, be used to simulate more general models and it offers specific classes e.g. event management classes [1].

SimPack

SimPack is a general simulation toolkit for creating discrete event or continuous simulations. It is event-oriented and uses a specific structure named *token* for synchronization. Its key point is that it supports a wide variety of event scheduling and continuous-time simulation models: declarative, functional, constraint models and multi-models [12].

Table 1 summarizes the different features of these object-oriented environments (part of the table is taken from [35]).

Table 1. Summary of object-oriented environments

Environment	Type	Active objects	Synchronization	Interaction
Awesime	Process	Yes	Semaphore	Method call
C++SIM	Process	Yes	Semaphore	Method call
ModSim II	Process	Yes	Semaphore	Method call
MOOSE	Process	Yes	Conditional waiting of events	Event
PROSIT	Process	Yes	Blocking services request	Service request
Ptolemy	Event	No	Particles exchange	Particles exchange
SIM++	Process	Yes	Conditional waiting of events	Event
Simex	Event & Process	Yes	Events	Event
SimPack	Event	No	Token exchange	Token exchange

2.2. Technologies for Developing OSSA

All object-oriented environments enumerated in the previous section are based on an object-oriented language and a simulation kernel. The earliest languages for OOS were object languages specific for simulation, for example ModSim. However, more recent tools use classical object languages like C++ (PROSIT, Sim++) or Java (Ptolemy) as a simulation language.

Object-Oriented Simulation with the known advantages was a good choice for a simulation framework. In order to plan this framework, developers knew and applied some object oriented technologies. One of these technologies is the Object-Oriented Analysis and Design (OOA&D), which is a set of techniques, languages, and tools covering all the phases of a software project except the final implementation phase, for which an ordinary programming language, preferably itself object-oriented, is employed.

There is a common language for object-oriented planning as well, the Unified Modelling Language (UML) [18, 16], which has been accepted by the Object Modelling Group (OMG) as the standard modelling language [30], and which was used in OSSA project. Clearly, complex systems usually modelled with UML have a complex structure and exhibit a complex behaviour, see *Fig. 2*. In order to deal with such complexity, the UML allows to model the various aspects of a system, such as the static structure, the dynamic behaviour, the relationships among the various components, and so on. These aspects all belong to the same system, so the aspects of the OSSA system can be seen as different views of it. The UML reflects this complexity in the presence of a number of different diagrams to represent the various aspects of the system being modelled.

3. Traffic Models and Tools

3.1. Types of Traffic Models and Tools

Traffic simulation models may be classified according to the level of detail with which they represent the system to be studied:

- Microscopic (high fidelity)
- Mesoscopic (mixed fidelity)
- Macroscopic (low fidelity)

A microscopic model describes both the system entities and their interactions at a high level of detail. For example, a lane-change maneuver at this level could invoke the car-following law for the subject vehicle with respect to its current leader, then with respect to its putative leader and its putative follower in the target lane, as well as representing other detailed driver decision processes. The duration of the lane-change maneuver can also be calculated.

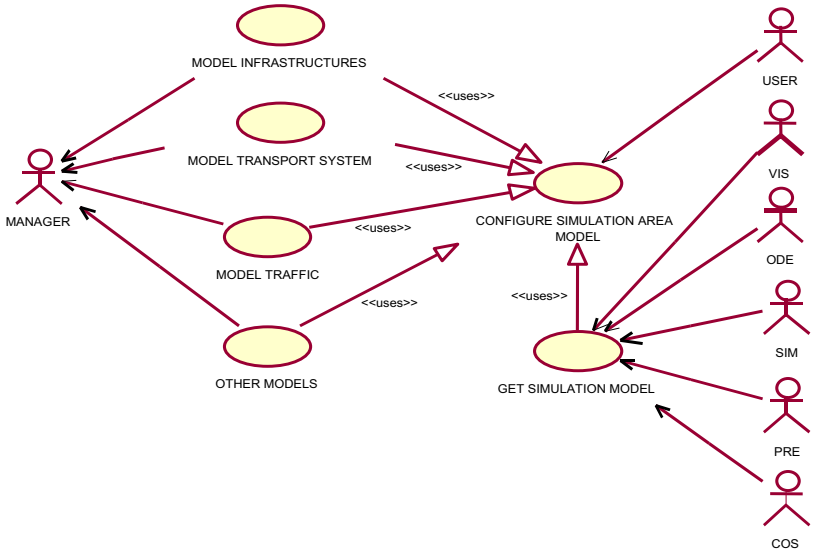


Fig. 2. UML for configuring the simulation conditions in OSSA system

A mesoscopic model generally represents most entities at a high level of detail but describes their activities and interactions at a much lower level of detail than would a microscopic model. For example, the lane-change maneuver could be represented for individual vehicles as an instantaneous event with the decision based, say, on relative lane densities, rather than detailed vehicle interactions.

A macroscopic model describes entities and their activities and interactions at a low level of detail. For example, the traffic stream may be represented in some aggregate manner such as a statistical histogram or by scalar values of flow rate, density and speed. Lane change maneuvers would probably not be represented at all; the model may assert that the traffic stream is properly allocated to lanes or employ an approximation to this end.

3.2. Transport Modelling in the OSSA Context

OSSA project is concentrating on road traffic, and thus the modelling of transport that uses roads is most relevant. Transport modelling in the OSSA context relates to:

- Traffic Flow Models
- Transport Planning Models and Tools
- General Traffic Engineering Models and Tools

The review begins with an overview of Traffic Flow Models, which, historically, were the first to be developed.

3.2.1. Traffic Flow Models

Traffic flow modelling methodologies seek to describe in a precise mathematical way the infrastructure of the road network (the static component) and the interactions of the vehicles and their operators (the mobile components) within the infrastructure. These theories provide an indispensable basis for the models and tools used in the design and operation of streets and highways. The scientific study of traffic flow began in the 1930's with the probability theory to the description of road traffic [8], with the study of models relating volume and speed [19], with the investigation of performance of traffic at intersections [20]. The theoretical developments were based on a variety of approaches, such as car-following, traffic wave theory (hydrodynamic analogy), queuing theory [29]. Most traffic models today use simulation for part or for all of these analyses.

Traffic flow analysis is one of the few areas, where macroscopic (or continuous flow) simulation is widely used. Most of the well-known macroscopic applications in this area originate in the late 1960's or the early 1970's. The UK-developed TRANSYT program [10] is an example of macroscopic simulation of urban arterial signal control coordination. The US programs FREQ and FREFLO [11], and the corresponding German analysis tool are related to motorway applications. The traditional traffic flow descriptions are based on continuous speed and distance variables.

3.2.2. Transport Planning

In the Transport Planning systems the planners may define various transport systems by combining transport mode and means of transport. The systems provide facilities to enable co-operation and decision-making between all relevant authorities (e.g. Ministries, local authorities, police forces etc.) to define optimum traffic management strategies. The transport planning is used in CONTRAM [38], a tool for analysis of street networks with signalized and non-signalized intersections. This tool is widely used for the analysis of street networks in large geographic areas. Another tool, EMME/2 [15] can be used for a wide variety of purposes, including the implementation of trip generation models and mode choice models. An efficient procedure balances a matrix in order to satisfy totals for origins and destinations, and EMME/2 is especially useful for implementing trip distribution models. It may be used to implement trip chaining models, such as park-and-ride, and for many other applications [17, 28].

SATURN (Simulation and Assignment of Traffic to Urban Road Networks) is a suite of flexible network analysis programs developed at the Institute for Transport

Studies, University of Leeds [33, 39]. Public transport is assigned to the network based on the transport system, line routes and timetables. TRIPS is used for modelling transport systems and covers all stages of the traditional four-stage-modelling process with advanced functionality for modelling trip production, distribution, modal split and network assignment [37]. TransCAD is a Geographic Information System (GIS) designed specifically for use by transportation professionals to store, display, manage, and analyse transportation data [36]. It combines GIS and transportation modelling capabilities in a single integrated platform, and provides a powerful engine with special extensions for transportation, mapping and visualization tools designed for transportation applications, travel demand forecasting and modules for routing. VISUM is a Transport Planning Tool (from PTV) that combines all relevant aspects of private and public transport planning [40], and there is another similar tool, VISSIM from PTV, that is OSSA-compliant.

3.2.3. General Traffic Engineering Tools

General Traffic Engineering Tools are used for the design and modelling of small areas of the road network, and usually for individual junction/intersection priorities and signalling. There are some tools, e.g. ARCADY [3], PICADY [32] and OSCADY developed by the TRL (Transport Research Laboratory [38]) for simulating roundabouts, priority junctions and traffic signals, respectively. Traffic assignment programs indicate the effects of a scheme on congestion and estimate resulting changes in routes, though they take no account of safety issues. These programs are capable of estimating both queues and delays, and numbers of accidents for isolated junctions. However, they only look at a specific junction and can take no account of possible interactions between different junctions. To take full account of changes across a whole network they must be used iteratively with an assignment program ([14]). OSCADY [31] (Optimized Signal Capacity and DelaY) has been developed into junction design and predicts capacities, queue lengths and delays at isolated signal-controlled junctions. By minimal data entry OSCADY provides the traffic engineer with an invaluable aid to designing new signalized junctions as well as assessing the effects of modifying existing designs. The programme can be applied to numerous junctions including T-Junctions, four-armed crossroads and staggered junctions. Up to five lanes it can be modelled on any approach. Up to three sets of signal timings it can be modelled, triggered by time of day or by critical queue length.

TRANSYT (Traffic Network Study Tool) is applicable to traffic control area modelling. It is one of the most widely used signal-timing programs. The Transportation and Road Research Laboratory in England developed the original version of TRANSYT in 1968. TRANSYT is a macroscopic, deterministic simulation and optimization model. The model requires the link flows and link turning proportions as inputs and assumes them to be constant for the entire simulation period. The TRANSYT program simulates traffic conditions for the duration for one complete

cycle length, and these conditions are assumed to be representative of all other cycles. The program optimizes splits and offsets given at a set cycle length and carries out a series of iterations between its traffic simulation module and the signal setting optimization module. For the initial signal settings, the traffic module estimates the performance index by simulating traffic as it reaches each intersection in the network. To minimize vehicular delay, the program searches for the optimum signal settings with a two-stage procedure. In the first stage the optimum cycle length is found through a search through the user-specified minimum and maximum cycle lengths. In the second stage, the cycle length producing the lowest performance index is investigated further by a hill-climbing procedure to determine the optimum offsets and phase splits. Although there are no guarantees that the search will find the global minimum, the program heuristic has provided a very practical trade off between accuracy and efficiency.

3.2.4. Aims in the Beginning of the Development

There are many aims that could be declared, improvements in road safety, higher priority to pedestrians and cyclists, reductions in car commuting, reductions in road freight, improved public transport, signal plans for bus priority, park and ride, energy saving, better use of existing infrastructure, intervention in demand e.g. restrictions on parking, road user charging, incident detection and monitoring, reducing emissions and (air and noise) pollution. The first of these is that the issues of safety and pollution are likely to be critical elements of future assessment strategies, and merit a high profile within the framework. The second is that public transport must also be incorporated, and the OSSA framework can reach these many aims, because into this open framework a lot of traffic simulators or modules can be plugged.

4. OSSA as a Concept for Integration

4.1. General Purpose Middleware Tools Relevant to OSSA

The trend in the implementation of new technology information systems is to create distributed applications. This makes the systems flexible, open and easy to manage. In OSSA the communication standards are very important, because it enables the intercommunication among different processes running on the same or on different computers interconnected by a network (LAN, WAN, Internet, etc.). Depending on the characteristics of the applications, computers and platforms to be used, the network protocols supporting the interconnection, the services to be provided, etc, there are different approaches and methodologies for implementing the distributed systems. It was possible to select a method that has no support for objects such as Sockets or Remote Procedure Calls (RPC), or to use a method that supports objects implicitly, like DCOM and CORBA, but the object approach was a relevant issue in

a simulation framework, so in OSSA the DCOM was preferred and the framework communication was based on DCOM.

4.2. The Structure of the OSSA System

One of the largest lack of the traffic application tools is the different formats of the traffic & transport systems, policies and strategies. This leads to an inconvenient situation, in which users need to afford additional work (when possible) to combine the results of the different tools (sometimes on different platforms) they are using. This reduces also the flexibility of traffic engineers and decision makers to freely choose among tools from different vendors because of the lack of a standard communication protocol for exchange data or combine results. In addition, traffic and transport simulation tools usually need inputs (mainly those addressing real or faster than real time simulation) from the live traffic and transport systems throughout their specific infrastructures for data collection. The simple idea behind the OSSA Framework (*Fig. 3*) is building a horizontal software framework to allow easy integration of heterogeneous data collection devices on the one side and software applications on the other side. The OSSA Framework allows traffic engineers and policy makers to freely choose simulation tools coming from different vendors, as long as they are 'OSSA-compliant.'

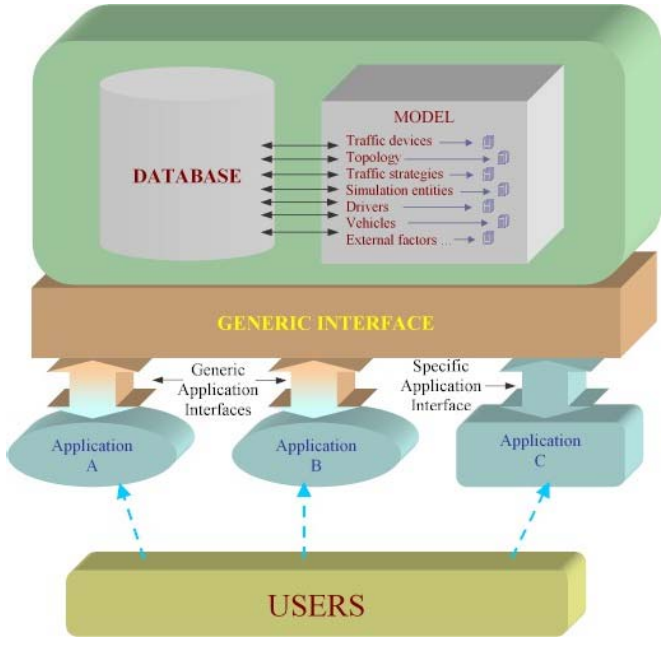


Fig. 3. OSSA framework context diagram

The OSSA, by enabling the interconnection and interoperability of a number of existing/future traffic and transport simulation related modules, involve the abstraction of all the elements, entities, actors represented and involved in this application area. This is possible by using OO Analysis & Design techniques and UML, and all these above mentioned elements come into the standard model. After modelling it is possible to have instances of them in a database that will store all the information concerning the area to be simulated. To access these models and database, generic interfaces for storing and retrieving information were to be defined. These interfaces should be simple and public pieces of software for enabling the communication with the elements defined in the OSSA Framework. In this way all the applications including an OSSA-compliant interface, will have access to the elements of the OSSA Framework.

The OSSA framework has been developed in three years and some simulators and modules are connected into it. Two kinds of simulators – a normal one and a real time simulator running on parallel computers – are worked out for OSSA system and several accessory modules are developed to be plugged in and to use this system. In the next part it is demonstrated how useful for problem solving these OSSA-compliant modules are. First an Urban Traffic Control System module called SCOOT is shown for calculating percentage congestion, then the visualizer and the analyzer are presented.

4.3. Urban Traffic Control System

In the traffic analysis it is important to obtain the optimum traffic signal settings according to the actual traffic. The prediction of traffic delays and stops for the particular pattern of traffic flows which exists in a network requires a traffic model and various optimization routines that can predict traffic movement for some time into the future. An external adaptive Traffic Control System, such as SCOOT is able to solve this. SCOOT (by TRL, Siemens and Peek Traffic, UK [34, 22, 13]) is one of the best known and most widely implemented traffic responsive systems based on a central computer and communications to individual traffic controllers. It has continuous communication between each controller and the central computer, data pass to and from each controller in every second. This is a centralized UTC system, the central processor performs an adaptive control on the traffic lights, and local controllers do not perform any regulating action, but can act independently if it is required.

SCOOT gives an estimation of traffic demand for the next cycle of the traffic lights and further ahead for cycle time changes. The optimization routines predict the effect of small changes to the current settings of the signals. Having considered separately the Split, Offset and Cycle time of the signals the best timings are chosen for use on the street. The optimizing theory model is based on data collected from detectors being present on every link in the network for which optimum timings need to be calculated. These data are processed and used to establish the flow profiles

– one for each link, divided into four-second intervals. The traffic predicted to be crossing the down stream stop-line in each interval is stored in these profiles. *Fig.4* shows three very different profiles. Profile A can be seen to show a platoon of traffic in the early part of the cycle with a trail (which might be turning traffic) of much smaller volume in the second half. Profile B is of similar average volume, but no platoon is obvious. In the Profile C two distinct platoons can be seen showing double cycling profile.

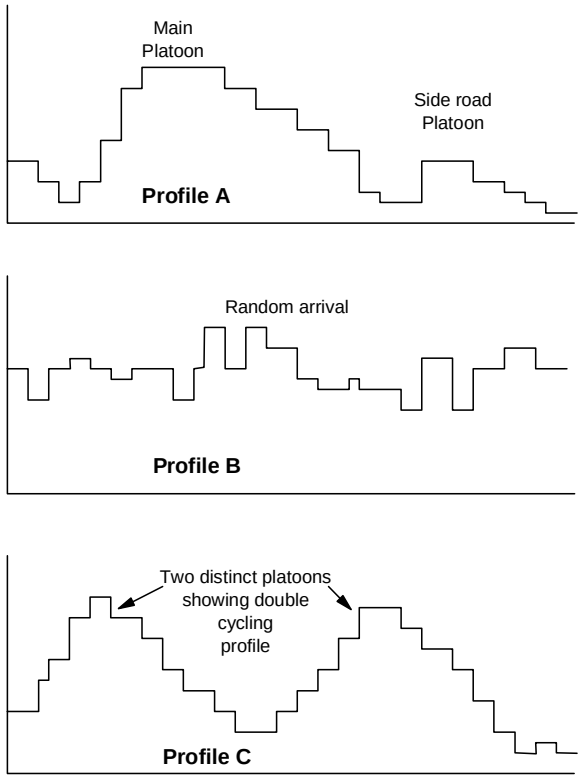


Fig. 4. Example flow profiles

All the traffic control stop-lines are considered in the optimizing model. The use of Red-Green information at the stop-lines and traffic volume profiles allows the prediction of the queues that will form for a given green time. The model contains a representation of traffic demand (Flow Profiles), stages, stops, and delays. In *Fig.5* the VEGA diagram shows the arrivals (the red and green bars on the top of the figure) and the queue build up and discharge at the stop-line (e.g. the actual queue at the bottom of the figure).

For the optimizing the percentage congestion (ratio of the congestion time and cycle time) should be determined. Congestion is directly measured by the

detector. If the detector is placed beyond the normal end of the queue in the street, it is never covered by stationary traffic, except of course when congestion occurs. If the detector detects traffic standing for the whole of the interval (for example four seconds, denote the length of the interval in seconds by i) then this is recorded, and the number of intervals of congestion (n) in any cycle is also stored (c is the cycle time in seconds). The percentage congestion is calculated from:

$$\text{Percentage\_Congestion} = \frac{n * i * 100}{c} \tag{1}$$

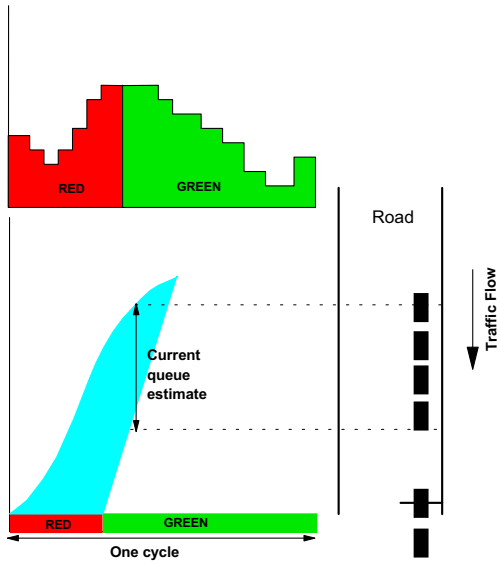


Fig. 5. VEGA diagram with arrrivals and queues

The purpose of the optimizers is to predict the effect of different signal timings on the street and to select the most appropriate set taking into account the percentage of congestion. SCOOT has three optimization procedures by which it adjusts signal timings: the Split Optimizer, the Offset Optimizer and the Cycle Time Optimizer. Each optimizer estimates the effect of a small incremental change in signal timings on the overall performance of the region's traffic signal network in order to minimize disturbance to traffic movement.

4.4. Standard Network Topology

In order to create a standard network topology (*Fig. 6*) in the OSSA we had to consider the structure of the existing transport-planning systems. The greatest

number of transport-planning models use zones as areas in which traffic begins (origin) or ends (destination) a journey. Within a zone the traffic is usually generated at a centroid and the traffic flow is distributed via various nodes onto a number of links. Some simulators generate traffic also along individual links. This will require a method to disperse zonal based O/D trip data onto the area of each zone. Thus the base objects of a network topology include: nodes, links and zones. Each object has a number of attributes depending on the simulation model used. The infrastructure includes objects (stop lines, traffic lights, yield signs, etc.) that are placed on zones, links or nodes.

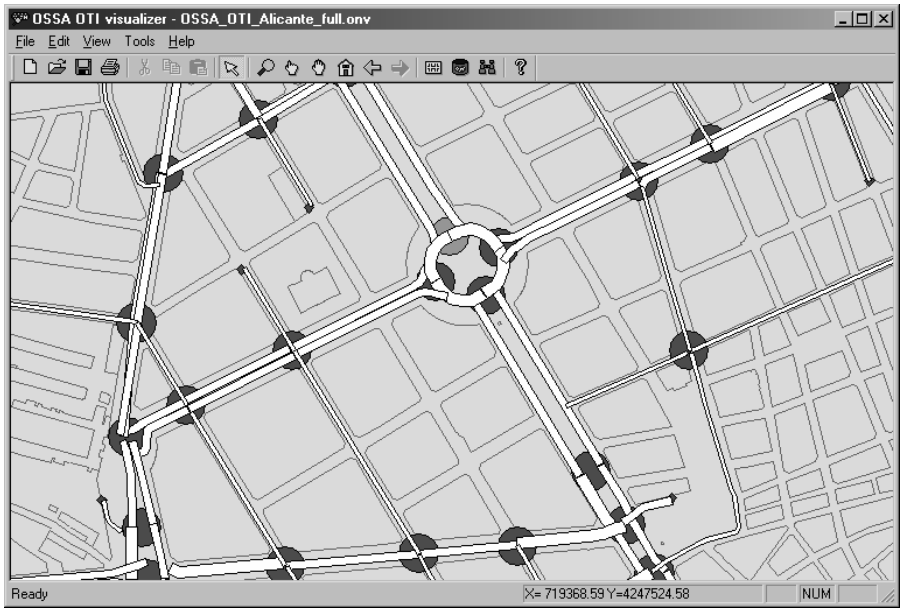


Fig. 6. General network topology

4.5. Interconnections

The OSSA makes a distributed model possible, where the different applications can run on different computers. Distribution means that a process can find, and have access to objects across the network. As with every distributed system which works with online data, the efficiency of the interface will be crucial to the online capability of the visualization tool. Some of the data, such as the topology, vehicle models and objects need to be transferred just once to the visualiser. These data will be kept locally and they remain unchanged, while the simulation is in progress. Also vehicles, which are currently at stop, do not need new information about their

state. For moving vehicles the description of their state will be kept as slim as possible because of sufficient simulation.

4.6. Analysis Tools for Inter-Model Communication

In the development of the analysis module the relevant state of the art theoretical and practical results were to be incorporated [2, 24, 23, 26]. In the analyser the input data were collected continuously during the simulation run and these data were aggregated and pre-evaluated by a fast algorithm. After the simulation the data were post-evaluated in an overall analysis, and the information obtained from evaluating the simulation results is provided in form of end-results of the simulation experiment. For the end users it is of great importance to provide a measure of the reliability of the results, for this reason the analyser gets the simulation results as inputs and determines the confidence intervals or error margins i.e. the reliability of the simulation results. Another task for the analyser is the termination of the simulation run in accordance with the required level of statistical accuracy [25].

5. Example Problem Solved by OSSA

Let us consider an example problem to illustrate the description above and solve the difficulty by OSSA Framework. We have to introduce a system for bus priority at the intersection of two main roads of a given city. We must evaluate the impact on the transversal road to the bus line direction in peak hours (considering also the environmental impact). In order to take a decision we need to use a simulation environment to evaluate the traffic behaviour. At the given city there exists a UTC with the necessary infrastructures for traffic data collection, and there is also a GIS application where the traffic infrastructures of the city are represented.

5.1. First Step to Simulate

We can simplify the problem by considering that only four entities will be involved in the simulation process: the intersection, cars, buses and traffic strategies. Although we have reduced the scope to these four entities, each one is complex and it is necessary to define a model of each one by considering all their aspects. For example:

- For the intersection: Topological parameters, number of lanes, stop lines, horizontal and vertical signals, etc.
- For the cars: Type of car, fuel consumption, emissions, etc.
- For the bus: Timetable, type of vehicle, fuel consumption, emissions, etc.
- For the traffic strategy: Traffic parameters, traffic plans, etc.

After the model is declared it is necessary to define how the logical representation of the real elements in the area of study will be created. Otherwise it should provide the OSSA Framework with geographical coordinates of the different elements at the intersection and other topology information, which are stored at the GIS system.

The next step is to work out the simulation for taking decisions. We will need a UTC for introducing the traffic plans in the intersection or statistical information about the traffic flows at peak ours, a simulation engine enabling bus priority simulation, an analysis tool for evaluating the environmental impact of the traffic and a visualizer for seeing the evolution of the area at a given time. By the OSSA GUI these 4 modules can be plugged into the framework of the whole system, as it can be seen in the *Fig. 7*.

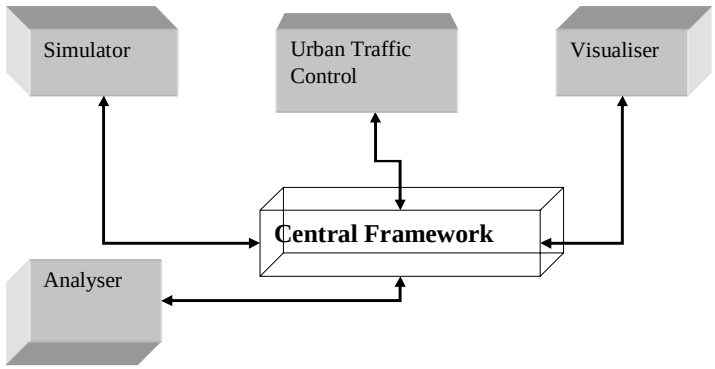


Fig. 7. Four modules are plugged into the framework

5.2. Simulation of the Model of the Problem

During the simulation the simulation engine has retrieved information about the area by means of an interface to the OSSA Framework and has stored their outputs, e.g. in terms of number of stopped vehicles at the transversal road. Regarding the analyser of the environmental impact, it will take its input (i.e. number of stopped and running vehicles) from the framework through an interface. The visualizer has received the topology of the intersection and the number of cars running. The first one (the data of the intersection) has been retrieved from the OSSA Framework database; the second one from the traffic flow information stored in the same database by the simulation engine. All these data are obtained through the OSSA interfaces.

Fig. 8 shows very detailed information of vehicle and occupancy on the links of the road network, which is important to the traffic engineer and to the simulation modeller.

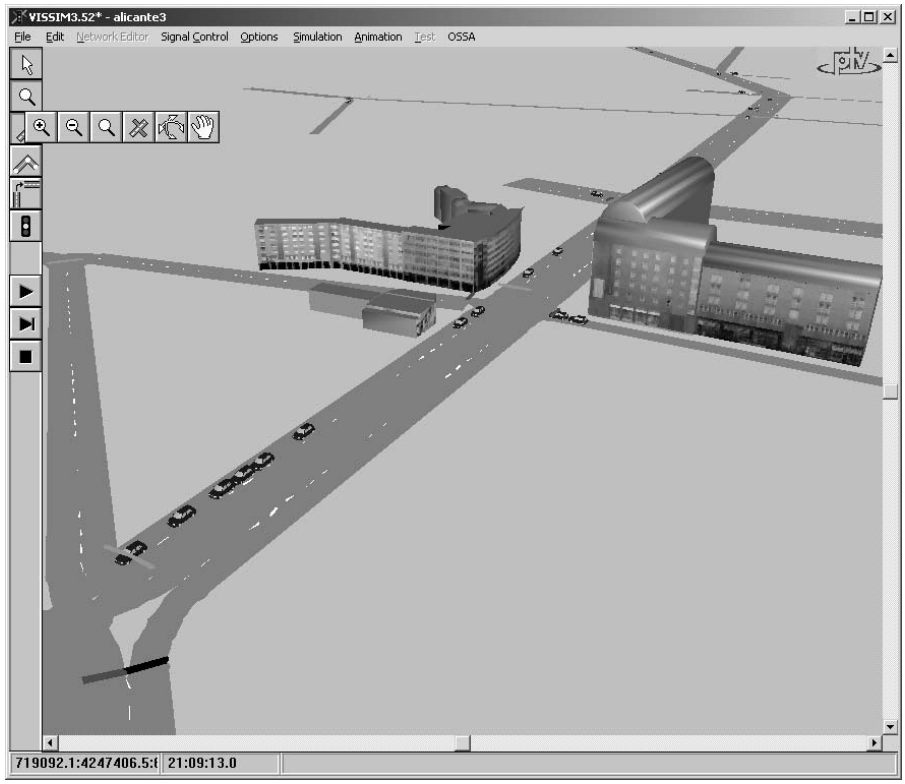


Fig. 8. The visualizer module shows the actual state of the traffic during the simulation

ANA Offline Evaluator 2.40.0

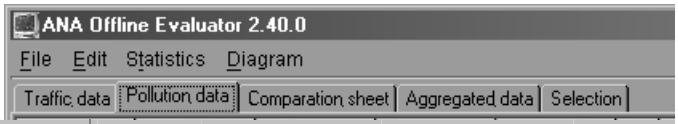
File Edit Statistics Histogram

Traffic data | Pollution data | Comparation sheet | Aggregated data | Selection

Date/Time	Link ID	ID	Height	Pos. X	Pos. Y	Speed	Acceleration
16:23	1013	125	980	152	351	42	1.3
16:23	1064	20	1050	132	453	49	2.1
16:23	1091	175	950	823	598	37	-1.4
16:23	1124	65	990	421	447	28	3.1

Fig. 9. The table of the results in the analyser module

After the simulation the analyser shows the results (see *Fig. 9* and *10*), from which a decision can be taken. Comparing the data of the original situation and the results of the introduction of the bus line, we can evaluate the impact of the transversal road to the bus line direction in peak hours. This impact is negligible, and the environmental impact is so little, that this introduction has no environmental pollution.



Link ID	Pollutant	Start Time	End Time	Amount	Amount / meter	Unit
1024	Carbon-Dioxide	16:23	16:33	855	7.1	g
1024	Carbon-Monoxide	16:23	16:33	3847	32	g
1024	Nitrogen-Oxide	16:23	16:33	3405	28.3	g

a) Without Bus Line

Link ID	Pollutant	Start Time	End Time	Amount	Amount / meter	Unit
1024	Carbon-Dioxide	16:23	16:33	893	7.4	g
1024	Carbon-Monoxide	16:23	16:33	3851	32.1	g
1024	Nitrogen-Oxide	16:23	16:33	3589	29.9	g

b) With Bus Line

Fig. 10. The results of the example with and without bus line

We can take a general conclusion: it is easy to inter-connect directly the simulation engine with the OSSA-compliant analyser and with the visualizer, since a lot of different interfaces are implemented in the OSSA Framework. On the other side there is a well-done interface between the simulator and the framework for configuring the area of study. In addition, if the decision-maker decides to use other visualizer or add another analysis tool, this will not impact the working mode of the simulator.

6. Summary

The conclusion is that the technology of the OSSA Framework guarantees the interoperability and interconnectivity of different applications. Object-Oriented Models are the most appropriate for guaranteeing the mentioned functionalities, enabling the existing application to run in their own platforms without needing migration to other ones. This generic tool is good for comparing data derived from different tools, from the live traffic control system's detectors. One of the key aspects of OSSA is to implement the interfaces for enabling the interconnection of the different tools throughout the common framework. The OSSA system is equally used by traffic control system operators, traffic planners, policymakers and

possibly the road users for different purposes. There are some advantages of the system, the most important ones are the following:

The OSSA Framework

- is openly available in order to plug new simulators and modules into the OSSA Framework in the future, so any organization can use and apply the framework.
- is platform and technology independent, because the modules can be written in different popular or less popular programming languages.
- facilitates the integration of different simulation related modules by means of standard interfaces and the robust and thorough object model reflecting the main relevant parameters in traffic simulation.

These advantages are of use to the professional users in the local & metropolitan authorities and the public transport authorities, because they, applying traffic modelling and simulation tools in their daily work, have realized that each of the existing tools has inadequacy and limitations. The OSSA Framework suits the user expectations of the next generation of simulation-based tools – which are flexible and customizable with facilities for interconnecting the different applications and tools – since the motivation for the research was strongly user-focussed. These users apply this system for decision-making, and the solutions can be obtained easily from OSSA. Beginning with a thorough survey of existing techniques and tools, it is confirmed that there is no equivalent to the OSSA Framework available in the state of the art.

Acknowledgement

The author would like to thank the leader of the OSSA project, Antonio Marqués for the possibility of the working in an EU project. Special thanks to Dr. J. Berényi and Prof. A. Jávor for leading the Hungarian participant of the project.

References

[1] ALTMAN, M. – WEE, B. – ONG, M. – ONG, C., *SIMEX Reference Manual*, University of Minnesota Academic Health Center, January 1995.
[2] AMBRÓZY, A. – JÁVOR, A., *Evaluation of Measured Data*, Műszaki Könyvkiadó, Budapest, 1976. (in Hungarian).
[3] ARCADY: http://www.trlsoftware.co.uk/productARCADY.htm
[4] BAEZNER, D. – LOMOW, E., A Tutorial Introduction to Object-Oriented Simulation and Sim++, *Proceedings of Winter Simulation Conference*, 1991, pp. 157–163.
[5] BAGRODIA, R. – LIAO, W., MAISIE: A Language and Optimizing Environment for Distributed Simulation, *SCS Multiconference on Distributed Simulation*, D. Nicol, editor, **22** (2), San Diego CA, January 1990, pp. 205–210.
[6] BELANGER, R. – RICE, S., ModSim User's Manual, CACI Product Company, La Jolla, CA. 1988.

[7] BENSLEY, E. H. – GIDDINGS, V. T. – LEIVENT, J. I. – WATRO, R. J., A Performance-Based Comparison of Object-Oriented Simulation Tools, In *Proceedings of Object-Oriented Simulation*, 1992.
[8] BIERLAIRE, M. – TOINT, P., MEUSE: an Origin-Destination Matrix Estimator that Exploits Structure, *Transportation Research B*, **25B** No. 1 (1995), pp. 47–60.
[9] BUCK, J. – HA, S. – LEE, A. – MESSERSCHMIDT, D., Ptolemy: A Framework for Simulating and Prototyping Heterogeneous Systems, Technical Report, University of California Berkeley, August 1992.
[10] BYRNE, A. – DE LASKI, A. – COURAGE, K. – WALLACE, C., Handbook of Computer Models for Traffic Operations Analysis, Technology Sharing Report FHWA-TS-82-213. Washington, D.C., 1982.
[11] COSMOS Project: ttp://www.cordis.lu/telematics /tap_transport/research/projects/cosmos.html
[12] CUBERT, R. – FISHWICK, P., *Sim++*, University of Florida, July 1992.
[13] DAY, I., SCOOT, A Presentation to the Transportation Research Board, *TRB Committee A3A18 Traffic Signal Systems, TRB Mid-Year Meeting and Adaptive Traffic Signal Control Workshop*, July 12–14, 1998.
[14] DETR Traffic Advisory Leaflet: 08/99, Urban Safety Management: Using SafeNET Department of the Environment, Transport and the Regions, 1999.
[15] EMME/2: http://www.inro.ca/
[16] ERIKSSON, H. E. – PENKER, M., *UML Toolkit*, John Wiley and Sons, 1997.
[17] EYKAMP, C. – SUTTON, J., GIS and Transportation Modeling , *PLANetworks*, Issue 1, Spring 1997, p. 5.
[18] FOWLER, M. – SCOTT, K., *UML Distilled: Applying the Standard Object Modeling Technique*, Addison Wesley Object Technology Series, (ISBN: 0-201-32563-2), 1997.
[19] GREENSHIELDS, B. D., A Study in Highway Capacity, *Highway Research Board, Proceedings*, **14**, (1935), p. 458.
[20] GREENSHIELDS, B. D. – SCHAPIRO, D. – ERICKSON, E. L., Traffic Performance at Urban Intersections, Bureau of Highway Traffic, Technical Report No. 1. Yale University Press, New Haven, CT., 1947.
[21] GRUNWALD, G., User's Guide to AWESILE-2, A Widely Extensible Simulation Environment. Technical Report CU-CS-552-91, University of Colorado at Boulder, 1991.
[22] HUNT, P. B. – ROBERTSON, D. I. et al, SCOOT – A Traffic Responsive Method of Coordinating Signals, Department of Transport, TRRL Report, LR 1014, Crowthorne, 1981.
[23] JÁVOR, A. – BENKŐ, M. – BUZÁSY, GY. – FARKAS, A. – SZŰCS, G., Traffic Simulation with the AI Controlled CASSANDRA Simulation System, *2nd IMACS Symposium on Mathematical Modelling*, Vienna, Austria, February 5–7, 1997, pp. 835–840.
[24] JÁVOR, A. – SZŰCS, G., Intelligent Demons with Hill Climbing Strategy for Optimizing Simulation Models, *Summer Computer Simulation Conference*, Reno, Neveda, July 19–22, 1998, pp. 99–104.
[25] JÁVOR, A. – SZŰCS, G., A Tool Set for Evaluating and Controlling Traffic Simulation, *EUROSIM 2001 Congress 'Shaping Future with Simulation'*, June 26-29, Delft, The Netherlands, 2001.
[26] KLEIJNEN, J. P. C., *Statistical Tools for Simulation Practitioners*, Marcel Dekker, Inc. 1987.
[27] LITTLE, M. – MCCUE, D., Construction and Use of a Simulation Package in C++, Technical Report 437, University of Newcastle upon Tyne, July 1993.
[28] LUSSIER, R. – WU, J. H., Development of a Data Exchange Protocol between EMME/2 and ARC/INFO, *17th Annual ESRI User Conference*, July 8-11, San Diego, California, 1997.
[29] MIDDELHAM, F. – WANG, T. C. – KOEIJVOETS, R. – TAALE, H., FLEXSYT-II- manual (part 1 and 2), Transport Research Centre (AVV), Rotterdam. 1994.
[30] OMG: http://www.omg.org/
[31] OSCADY: http://www.trl.co.uk
[32] PICADY: http://www.trlsoftware.co.uk/productPICADY.htm
[33] SATURN: http://www.its.leeds.ac.uk/software/saturn

[34] SCOOT: http://www.scoot-utc.com/
[35] SIEGEL, G., Prosit: An Environment for Discrete-Event Simulation Programming. Ph.D Thesis, University of Nice-Sophia Antipolis, September 1997.
[36] TransCAD: http://www.caliper.com/
[37] TRIPS: http://www.trips.co.uk/
[38] TRL: CONTRAM5 – An Enhanced Traffic Assignment Model, Transport and Road Research Laboratory Research Report 249, N B Taylor. TRL Crowthorne, UK. 1990.
[39] Van VLIET, D. – HALL, M., SATURN User Manual (on line), 1998. http://www.its.leeds.ac.uk/software/saturn/
[40] VISUM: http://www.english.ptv.de/produkte/visum.htm