The University of Michigan

IVHS

Intelligent Vehicle-Highway Systems

Modelling a System Architecture

Arvind Gupta and Bernard A. Galler

Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, Michigan 48109

May 1991

IVHS Technical Report 91 - 04
For Release on May 1, 1992

UNIVERSITY OF MICHIGAN
TRANSPORTATION RESEARCH INSTITUTE • (313) 936-1066
2901 Baxter Road, Ann Arbor, MI 48109-2150, and

COLLEGE OF ENGINEERING • (313) 764-4333
4112 EECS, Ann Arbor, MI 48109-2122

Modelling a System Architecture

Arvind Gupta and Bernard A. Galler

Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, Michigan 48109

May 1991

IVHS Technical Report 91 - 04
For Release on May 1, 1992

Modelling a System Architecture

Arvind Gupta and Bernard A. Galler

Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122

Abstract

The IVHS group at the University of Michigan, in an attempt to understand better the process of specifying a system architecture for an ATIS system, carried out the entire specification process for a small architecture. Then the question of modelling that architecture (or any ATIS system architecture) was considered, and the NetMod modelling system at the University of Michigan was selected to see how well it would serve for that purpose. This report describes the results, including the shortcomings of NetMod for that kind of problem, and the changes that are expected which will help the situation.

Introduction

The selection of a system architecture for any Advanced Traveller Information System (ATIS), is a complex process. There are many constraints, such as the need for compatibility of the in-vehicle unit - at some level of detail - with ATIS systems across the country. On the other hand, there are many choices, ranging from which objectives to include to which technologies to consider. To make the matter even more complex, the *process* of selecting an architecture and its implementation is important, if only to make sure that all the selected objectives and constraints have been considered, and all available technologies considered.

In order to better understand the *process*, a group of University of Michigan

researchers held a retreat, in which the process was carried out in its entirety, albeit for a small and manageable set of objectives, constraints and technologies. We believe that we gained a much better understanding of the process and its implications. The result of the day's activity was a system architecture, plus a feasible configuration of technologies, for a stated set of objectives and constraints. The activity of that day, and its lessons, will be described in another report.

After arriving at the results of that day, a request was made to explore the various parameters of that system architecture, i.e., to model it in a computer system. A convenient network modelling system, NetMod, had been developed in the Department of Electrical Engineering and Computer Science of the University of Michigan under the supervision of Prof. Toby J. Teorey, and it was decided that we should attempt to model the system architecture in NetMod. At the very least we would find out how useful NetMod would be for our ATIS purposes, and hopefully, we would be successful in modelling our candidate architecture. The report which follows presents the parameters of the component architecture, the results of the modelling effort, and an evaluation of NetMod from our ATIS perspective.

Objective of Simulation

The objective of this particular simulation is to study the traffic flow of communication between the Vehicle and the Traffic Advisory Center (TAC), and from the Roadside to the Vehicle. The parameters to be monitored are time delays, request-reply bottlenecks, throughput, and accuracy (i.e. the number of packets/messages actually received by each destination).

Overview of the System Design

The system has been designed [RG] so that the TAC sends state-of-the-system messages to the Vehicle. The Vehicle uses the information received from the TAC to compute its route and then sends back to the TAC various vehicle statistics, such as time taken to traverse a particular link etc., requested by the TAC. There is also one-way communication from the Roadside to the Vehicle, which provides the Vehicle with any changes to the global information - Yellow Pages and the "Micro-Map", contained in the Vehicle.

System Components

The primary subsystems are the Vehicle, TAC and the Roadside, each having the following components:

Vehicle - 1) RF Receiver;
2) Collision Sensor;
3) User Input and Display Unit;
4) Processor;
5) Probe Data Response;
6) Navigation Locator;
7) Localcast Receiver of Micr0-Map; and,
8) RF Transmitter.

TAC - 1) RF Receiver;
2) Mainframe Computer; and,
3) RF Transmitter.

Roadside - 1) Computer Memory Micro-Map; and,
2) RF Transmitter.

Components in the Vehicle

The Vehicle has the following components, as shown in Figure 1A, to communicate to and from TAC :

1) *RF Receiver*: This receives the messages sent from the TAC.

   The types of messages received are:
   a) Local Yellow Pages Data;
   b) Link Times;
   c) Request for MayDay Verification;
   d) Request for Vehicle Location;
   e) Request for Time taken to Traverse a particular link.

2) *Collision Sensor*: This sends an automatic **MayDay** signal to the TAC on sensing a collision.

The following message is sent directly to the TAC via the RF Transmitter:
a) MayDay.

3) *User Input and Display Unit*: This displays requested information to the user. It also provides the user with an interface for input requests.

It receives the following messages from the Processor:
a) Routing Information;
b) Yellow Pages Information;
c) Request for User-provided Input;
d) Travel Time.

This unit send the following messages to the Processor:
a) Request for Routing Information;
b) Request for Yellow Pages Information;
c) User-provided input;
d) Request for Travel Time Information.

4) *Processor*: This unit processes information requested by the TAC and the Driver. For example, it would plan the route that the driver had requested or present the requested yellow pages information to the driver.

It receives the following messages from the User Input Display:
a) Request for Routing Information;
b) Request for Yellow Pages Information;
c) User-provided input;
d) Request for Travel Time Information.

It sends the following messages to the User Input Display:
a) Routing Information;
b) Yellow Pages Information;
c) Request for User-provided Input;
d) Travel Time.

It receives the following message from Probe Data Response:
a) Link Traversal Time.

It sends the following message to the Probe Data Response:

a) Request for Link Traversal Time.

It receives the following message from the TAC via the RF Receiver:
a) Request for MayDay Verification.

It sends the following message to the TAC via the RF Transmitter:
a) Mayday Verification.

It receives the following message from the Localcast Transmitter of Micro-Map changes:
a) Local micro-map changes.

It sends the following message to the Localcast Receiver for Micro-Map information:
a) Request for local micro-map changes.

It receives the following message from the Navigation Locator:
a) Vehicle Location.

It sends the following message to the Navigation Locator:
a) Request for Vehicle Location.

5) *Probe Data Response*: This handles the request from the TAC and from the Processor for information regarding the link traversal time by the vehicle.

It receives the following message from the Processor:
a) Request for Link Traversal Time.

It sends the following message to the Processor:
a) Link Traversal Time.

It receives the following message from the TAC via the RF Receiver:
a) Request for Link Traversal Time.

It sends the following message to the TAC via the RF Transmitter:
a) Link Traversal Time.

6) *Navigation Locator*: This handles a request from the Vehicle or the TAC

to get a fix on the location of the Vehicle.

It receives the following message from the Processor:
a) Request for Vehicle Location.

It sends the following message to the Processor:
a) Vehicle Location.

It receives the following message from the TAC via the RF Receiver:
a) Request for Vehicle Location.

It sends the following message to the TAC via the RF Transmitter:
a) Vehicle Location.

7) *Localcast Receiver of Micro-Map*: This unit receives the micro-map information sent by the Roadside Transmitter.

It receives the following message from the Processor:
a) Request for local micro-map changes.

It sends the following message to the Processor:
a) Local micro-map changes.

8) *RF Transmitter*: This unit transmits the information processed in the vehicle to the TAC:

The following messages are sent to the TAC:
a) Vehicle Location;
b) MayDay Verification;
c) Link Traversal Time;
d) MayDay.

Components at the TAC

The TAC has the following components, as shown in Figure 1B, to communicate to and from a Vehicle.

1) *RF Receiver*: This receives the information transmitted from the

vehicle.

The message are:
a) Vehicle Location;
b) MayDay Verification;
c) Link Traversal Time;
d) MayDay.

2) *Mainframe Computer*: This generates travel times. It also calls 911 when it receives a **MayDay** message.

The messages received are:
a) Vehicle Location;
b) MayDay Verification;
c) Link Traversal Time;
d) MayDay.

The messages sent to RF Transmitter to be transmitted to the Vehicle are:
a) Local Yellow Pages Data;
b) Link Times;
c) Request for MayDay Verification;
d) Request for Vehicle Location;
e) Request for Time taken to Traverse a particular link.

Message sent to 911 is:
a) Report of MayDay.

3) *RF Transmitter*: This transmits the following messages to a Vehicle:
a) Local Yellow Pages Data;
b) Link Times;
c) Request for MayDay Verification;
d) Request for Vehicle Location;
e) Request for Time taken to Traverse a particular link.

Components at the Roadside

The Roadside has the following components, as shown in Figure 1C, to

update a Vehicle with local changes to the area map.

1) *Computer Memory Micro-Map*: This holds the changes in the yellow pages information and changes to the local area micro-map.

   It sends the following messages to the RF Transmitter, to be transmitted to a vehicle:
   a) Changes in the Yellow Pages Information;
   b) Changes in the local area micro-map.

2) RF Transmitter: This periodically broadcasts the following messages to the Vehicles in its area:
   a) Changes in the Yellow Pages Information;
   b) Changes in the local area micro-map.

Modelling Tool

We decided to use NetMod as the modelling tool to model our system and do our simulation study. We decided to use NetMod for various reasons, the major ones being:

1) NetMod was designed and developed at the University of Michigan by Prof. Toby Teorey's group and is readily available. In addition, since the model designer is still at the University, one can get advice from him, and there is a possibility that we could get a custom-made Designer Tool to meet our special needs, if any.

2) This helps us to get a first crack at designing and simulating the system, in order to get better insight into our simulation, and to determine some of the initial specifications of certain system parameters, etc.

About NetMod

"The Network Modelling Tool (NetMod) uses simple analytical models to provide designers of large interconnected local area networks with an in-depth analysis of the potential performance of these systems" [Net]. NetMod can analyze an existing or proposed network in terms of its basic

performance characteristics, for example, component utilization, throughput, and packet delay times. Network operating system details are beyond the scope of existing analytical model and this report.

Modelling Tools Used

Each component (Processor etc.) is modelled as a WORKSTATION (computer) that is interconnected with another component via a Communication Media (We have used Ethernets in our Model). We can specify paths (route that a particular message/packet will take) from one component to another using a Specify Path command. For each Ethernet we can specify:

1) *Submodel speed (bps)*: The speed at which data can flow through the communication channel (media);
2) *Length of coax cable end-to-end*;
3) *Length of twisted pair end-to-end*;
4) *Number of repeaters between ends*;
5) *End-to-end delay (in microseconds)*.

For each component (workstation) one can specify:

1) User Category: It defines the type of traffic that the workstation will produce;
2) Machine and Card type (for example, the machine is a SUN and comes with an Ethernet card);
3) Number of Workstations: The number of components of a particular type;
4) Application Category (i.e. Request for Link Times etc.)
5) Time Used (hours per day): Percentage of processor (CPU in each workstation) time spent on a particular application.
6) Total User Time (hours per day): Total time that a particular component is used.

For each application one can specify:

1) *Traffic Rate* (bps): The number of bits of the message produced per second (Percent of message/traffic in bits produced per second);
2) Packet Size (bits): The size of the packet/message produced.

Interconnection between Sub-Models

For the purpose of this simulation the interconnections (Communication Media) between the Roadside and the Vehicle, and between the Vehicle and the TAC, are Ethernets. These interconnections are shown in Figure 1A, where all the components of the Vehicle are shown, and the TAC and the Roadside are shown as submodels connected by Ethernets to their respective connection (communication) points (components) in the Vehicle.

Types of Output Statistics

There are two types of output statistics: intermediate and final. Intermediate output displays the number of packets per second being processed by the LAN, interconnect device, or source device under steady state output. Final statistics are shown in terms of LAN and router utilization, throughput in pps (packets per second), average packet delay at each LAN or interconnect device in milliseconds, and implicitly the average end-to-end delay, which is calculated as the sum of the individual average packet delay times. Figures 2A, 2B, and 2C show the network activity without showing the queuing delays, while Figures 3A, 3B, and 3C show the network activity with queuing delays, for each of the three System Components - Vehicle, TAC and Roadside.

Note: These statistics correspond to the example parameters input for the various atomic components (Processor, Navigator etc) in the Vehicle, TAC and Roadside. These parameters are listed in Table 1.

Drawbacks of NetMod

NetMod is a simple modelling tool that does parametric analysis. What we need for the Intelligent Vehicle-Highway System (IVHS) simulation is a network simulation tool that does actual simulation. NEST is a network simulation tool developed at Columbia University, and work to integrate NetMod and NEST is currently being done at the University of Michigan. NetMod does not allow us to study the dynamic request-reply sequence that would model the real system. That is, it does not allow us to simulate the

actual life-time of an individual message being sent from the client to the server, and the processing of that message by the server. It is a static model in that utilization of different applications are given a priori, and it is quite possible that the message received by the server may be a request for a service A, and the response may be for a service B, because all that can be simulated is the percentage of the time in which a server is sending messages. We cannot even see the request-reply sequence, because only the type of output that we can see from running a simulation is as shown in Figures 2 and 3. We also cannot model our system hierarchy exactly the way we choose. For example, a Submodel cannot be a parent of a node in the interconnecting model.

The model places the additional constraint that the simulation has to be modelled for an 8-hour work day. The percent of use per hour in this report has been normalized to account for this requirement. Appendix 1 gives the formulae used to calculate the various system parameters shown in Table 1.

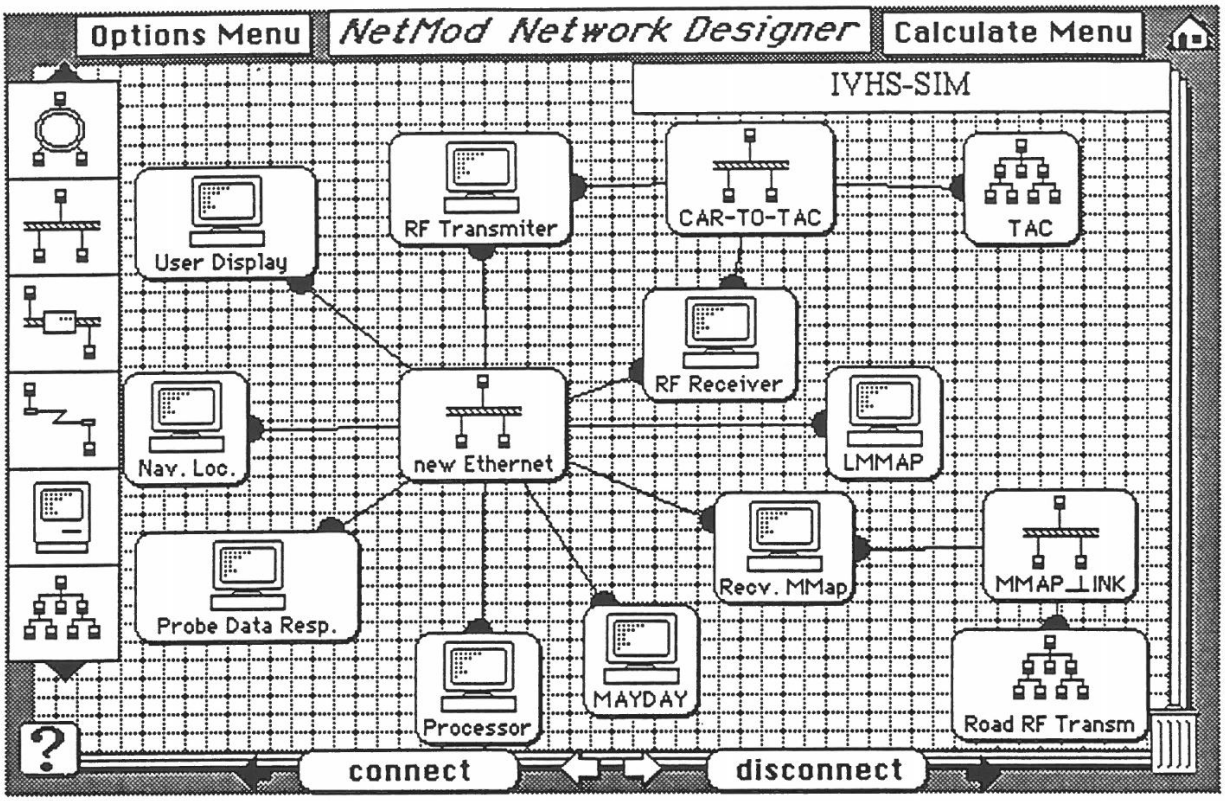

Figure 1A: Components in the Car with Interconnections to TAC & Roadside

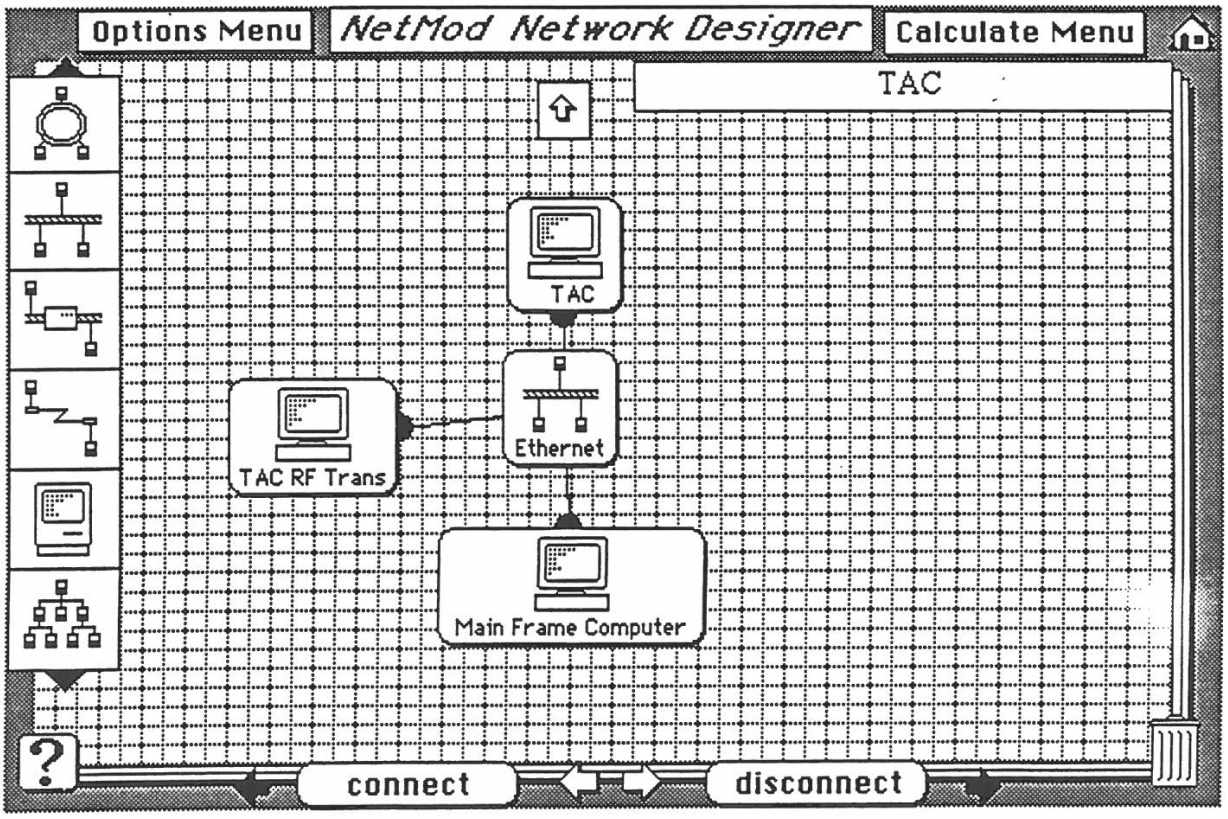

Figure 1B: Components in the TAC

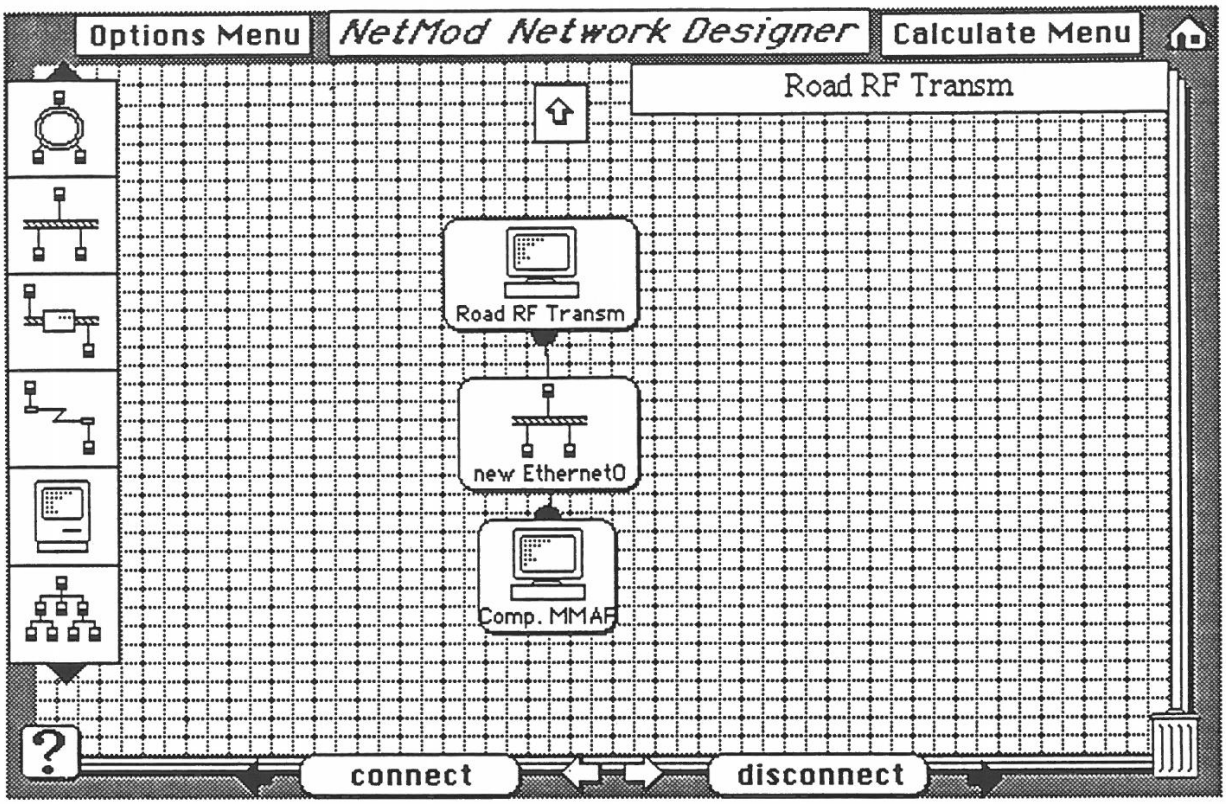

Figure 1C: Components at the Roadside

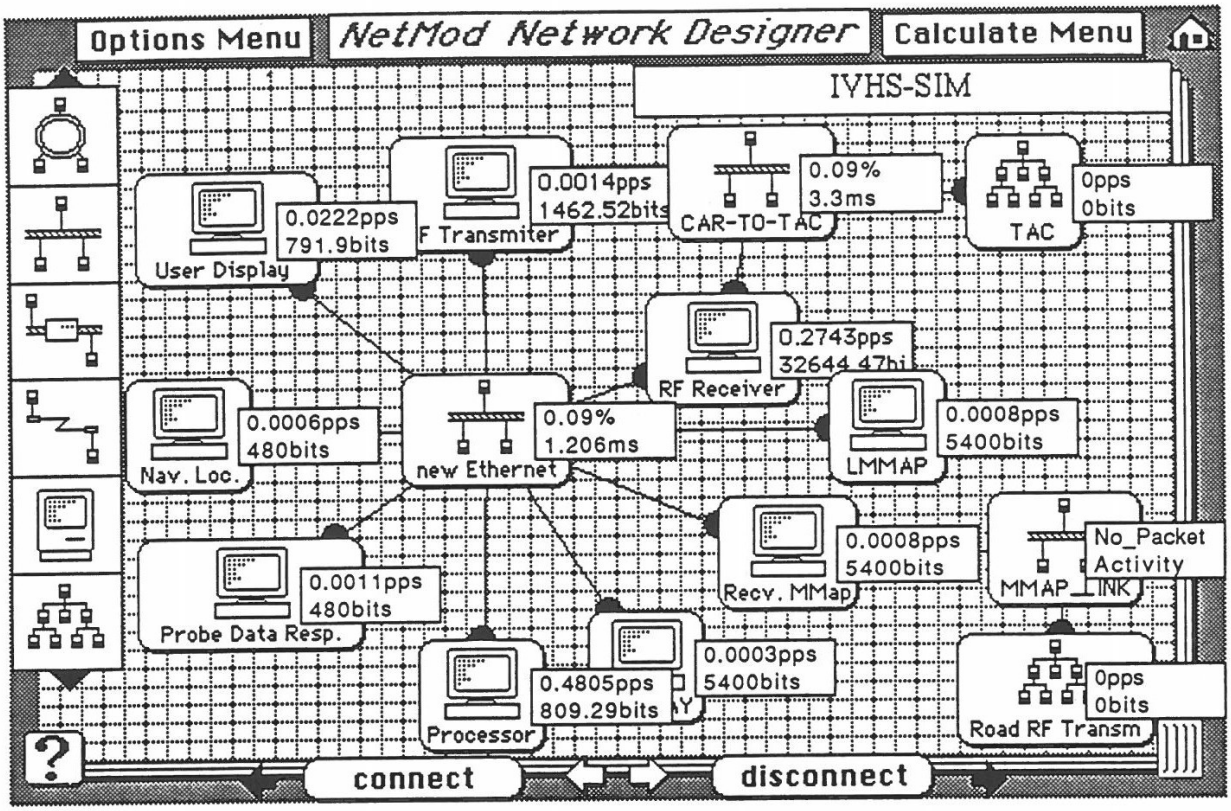

Figure 2A: Network Activity in the Car

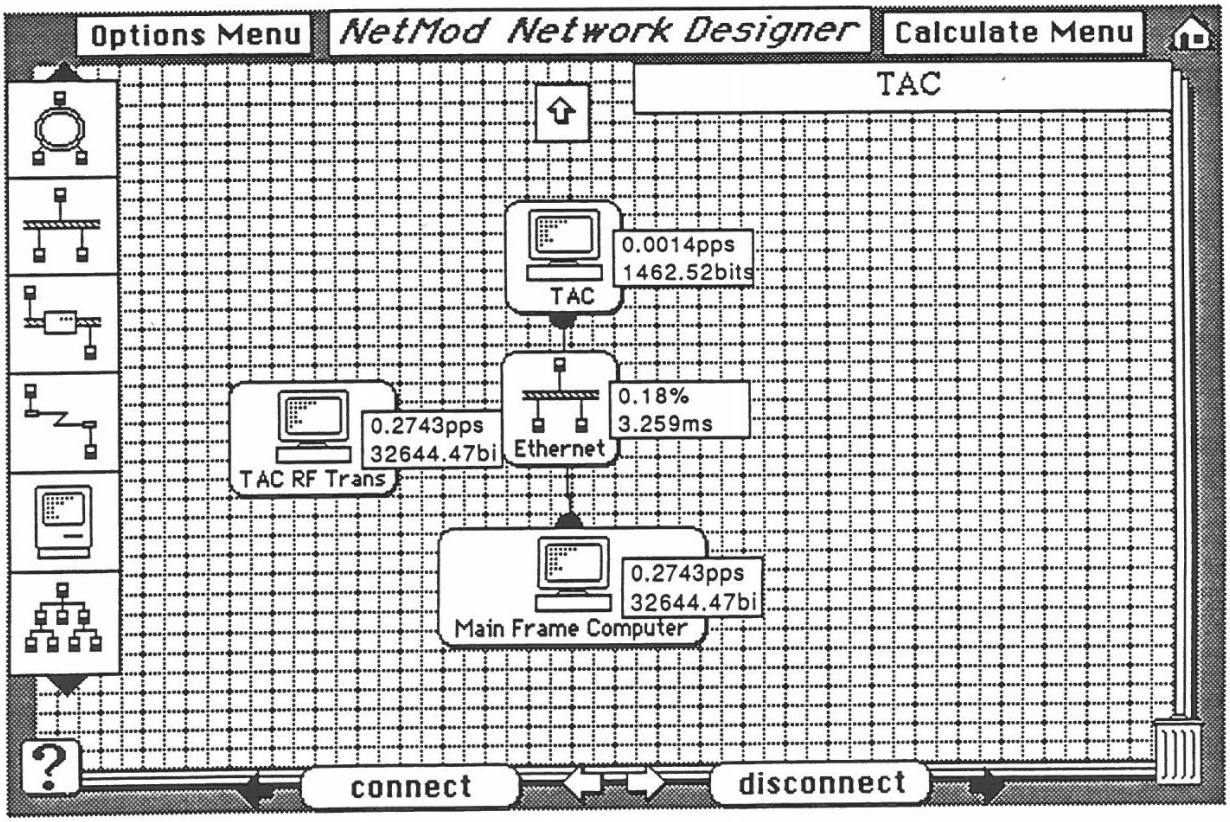

Figure 2B: Network Activity in the TAC

Network Activity

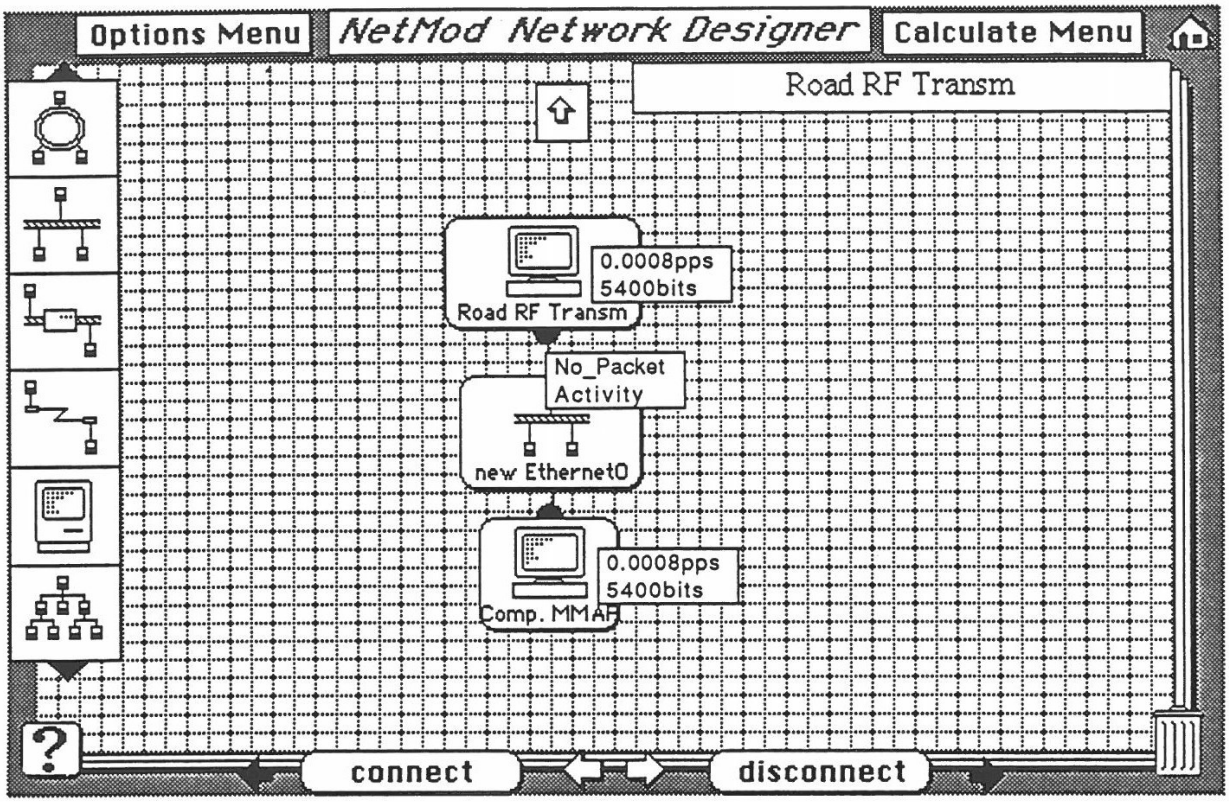

Figure 2C: Network Activity at the Roadside

Network Activity Showing Queueing Delays

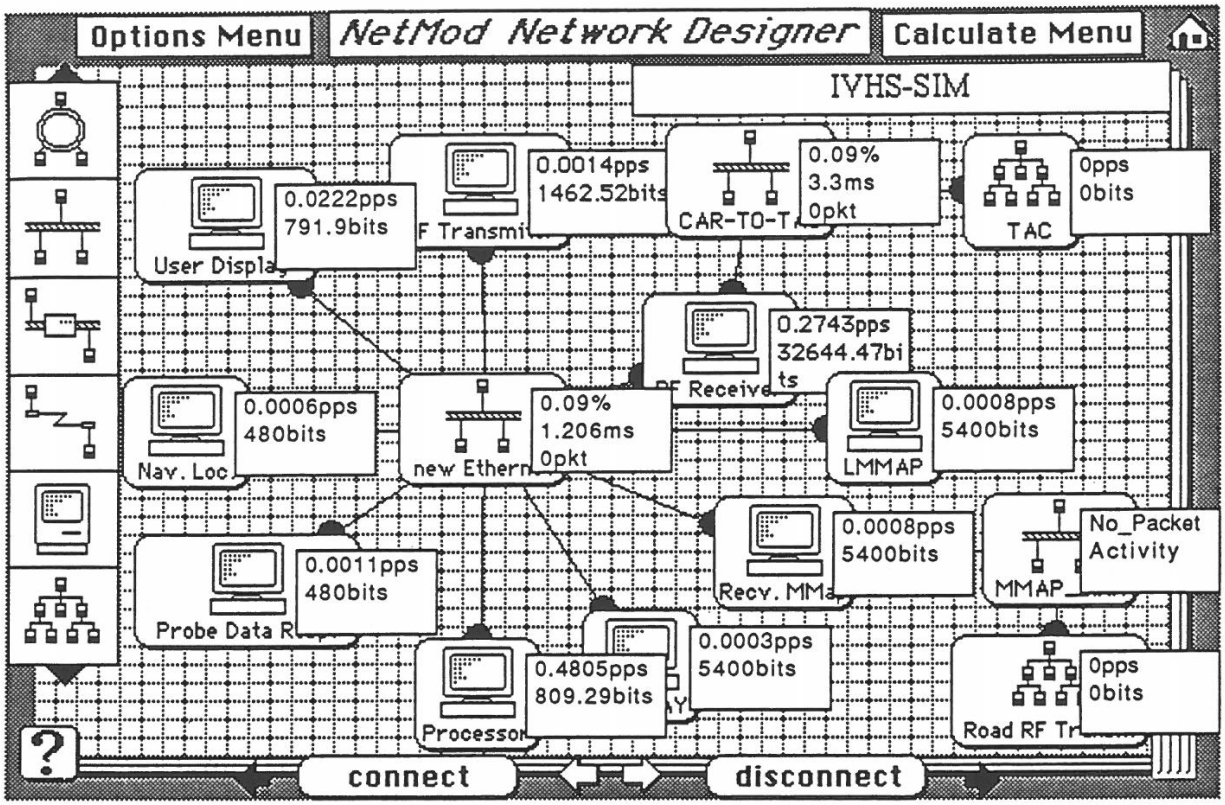

Figure 3A: Network Activity in the Car

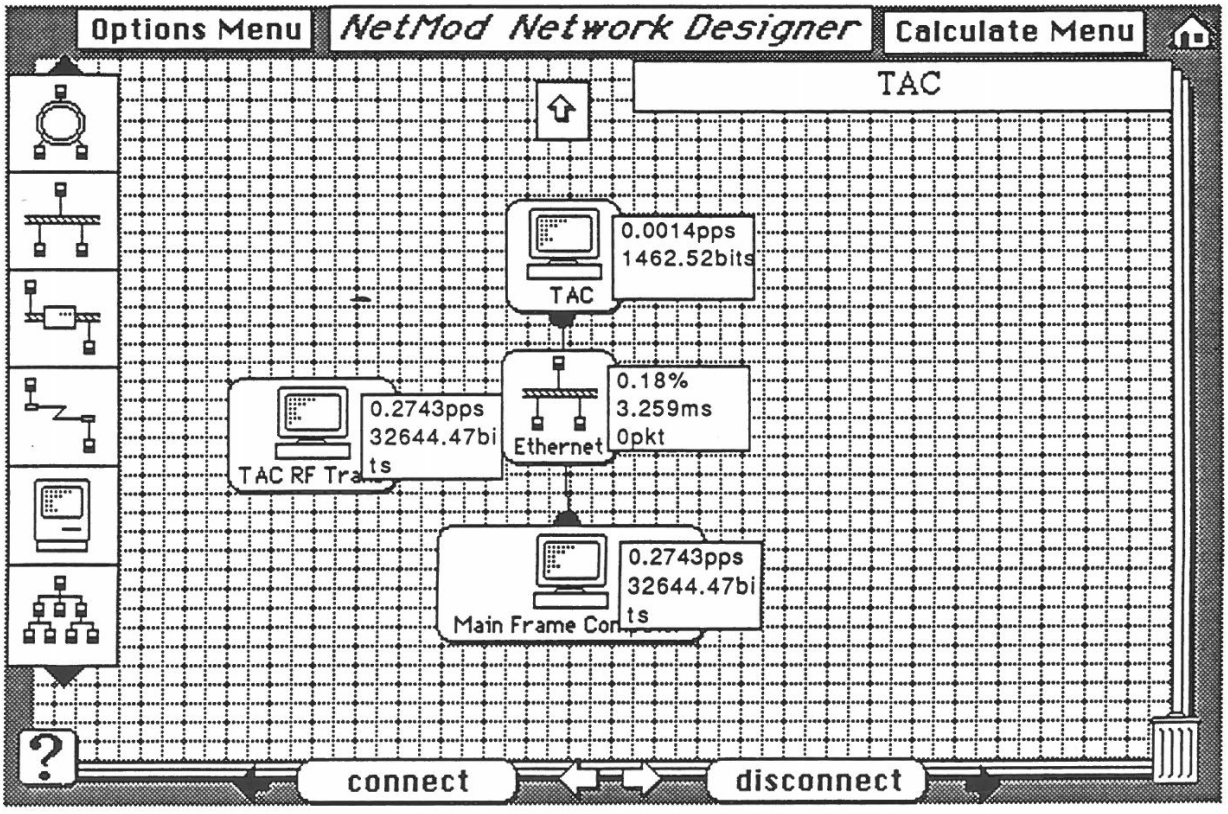

Figure 3B: Network Activity in the TAC

Network Activity Showing Queueing Delays

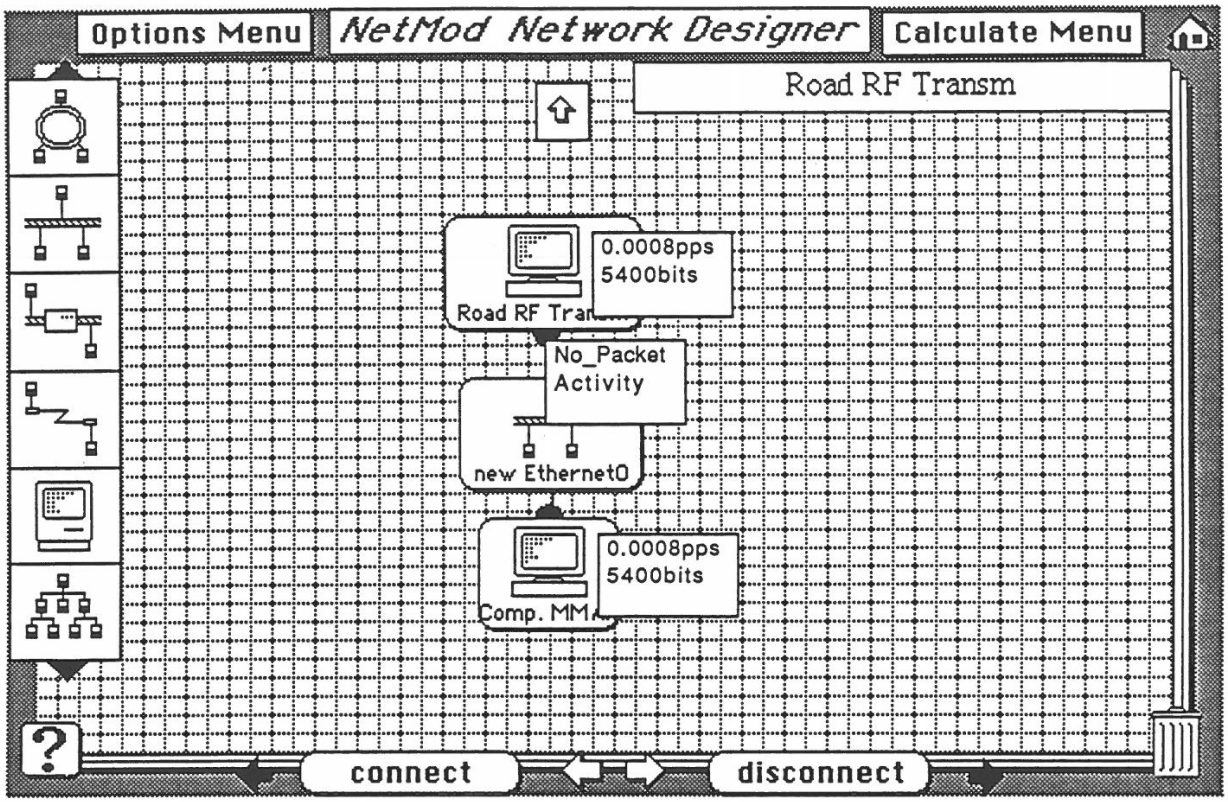

Figure 3C: Network Activity at the Roadside

References

[RG] Rubin A. and Galler B. - A First Design of Phase II of DART. *IVHS Technical Report-90-8*, The University of Michigan, Ann Arbor.

[Net] NetMod User's Guide, Version 0.16. *Center for Information Technology Integration*, The University of Michigan, Ann Arbor.

Bachmann, D.W., Segal, M.E., Srinivasan, M.M., Teorey, T.J. - NetMod: A Design Tool for Large-Scale Heterogeneous Campus Networks. *IEEE Journal on Selected Areas in Communication*, Vol. 9, No. 1, January 1991.

Table 1 : Example System Parameters for Simulation

	A	B	C	D	E	F
1		Application	Frequency/hr	% of Processor	Packet/minute	Packet Size
2				Time (Normalized)		(bits)
3						
4	TAC	Send Yellow Pages Data	2	0.267	90	5400
5		Send Link Times	12	1.6	43694	2621640
6		Request for MayDay Verf.	1	0.133	4	240
7		Request for Vehicle Locn.	1	0.133	4	240
8		Req. for Time per Link	2	0.267	4	240
9						
10	TAC Transmitter	Send Yellow Pages Data	2	0.267	90	5400
11		Send Link Times	12	1.6	43694	2621640
12		Request for MayDay Verf.	1	0.133	4	240
13		Request for Vehicle Locn.	1	0.133	4	240
14		Req. for Time per Link	2	0.267	4	240
15						
16	Vehicle Receiver	Send Yellow Pages Data	2	0.267	90	5400
17		Send Link Times	12	1.6	43694	2621640
18		Request for MayDay Verf.	1	0.133	4	240
19		Request for Vehicle Locn.	1	0.133	4	240
20		Req. for Time per Link	2	0.267	4	240
21						
22	Collision Sensor	MayDay	1	0.133	90	5400
23						
24	User Display	Request for Route	1	0.133	40	2400
25		Req. for Yellow Pages	2	0.267	4	240
26		Request for Travel Time	2	0.267	4	240
27		User Provided Input	1	0.133	8	480
28						
29	Processor	Route To User	1	0.133	90	5400
30		Yellow Pages Information	2	0.267	90	5400
31		Req. for User Provided Input	1	0.133	4	240
32		Travel Time	2	0.267	8	480
33		Request for Link Trav. Time	2	0.267	4	240

Table 1 : Example System Parameters for Simulation

	A	B	C	D	E	F
34		Application	Frequency/hr	% of Processor	Packet/minute	Packet Size
35				Time (Normalized)		(bits)
36						
37		MayDay Verification	1	0.133	8	480
38		Req. for Micro-Map Changes	3	0.4	90	5400
39		Request for Car Location	1	0.133	4	480
40						
41	Probe Data Resp.	Link Time	4	0.533	8	480
42						
43	Navigator	Navigation Location	2	0.267	8	480
44						
45	Roadside	Micro-Map Changes	3	0.4	90	5400
46						
47	Car Transmitter	Car Loaction	1	0.133	8	480
48		Link time	2	0.267	8	480
49		MayDay Verification	1	0.133	8	480
50		MayDay	1	0.133	90	90

Appendix 1

We start off by deciding how frequently a certain task is to occur in an hour. Given that, we normalize the Percentage of Processor (CPU) time for an eight-hour day using the following formula:

% of Processor Time = (8 * Frequency/hour)/60

Here we assume that given the frequency of a particular activity per hour, it takes one minute to complete one whole activity. We have normalized the activity to reflect an eight-hour day, because the modelling tool is configured to work for an eight-hour day. Further, we cannot specify the frequency/hour parameter (shown in Table 1), in the modelling tool, but it is a simulation-specific parameter that the user has to take into account when calculating the Percent of Processor time, given our assumption.

Given the size of the message in bits (assuming that the message is a packet), we have calculated the processor speed to be fast enough to produce one packet per minute.

Processor Speed = Packet Size/60 packets per minute.

Given the processor speed, we can calculate the packet size and also drop the assumption that each activity takes a minute. For example, if we want the same frequency of messages, we can either decrease the Processor Speed or increase the size of the Message (packet), and accordingly increase or decrease the Percentage of Processor Time.

Assume that we want to send a message 2 times an hour, and the message size is 5400 bits. If we assume that each message preparation should take exactly one minute, then processor speed has to be 90 bits/second. Now if we drop the assumption that message preparation time has to be one minute, and want it to be 2 minutes and assume further that the frequency of the message is the same, then to match our specifications, either our Processor speed has to be slowed by half, i.e., Processor Speed goes down to 45 bits/second, or the size of the message has to double to 10800 bits.

An alternative way to look at this problem is that given the Processor Speed, Message Size and frequency per hour, we can calculate the Percent of Processor Time using the following formulae:

Time in Minutes to Generate 1 message = Packet Size/(Processor Speed*60)

where, **Packet Size** is the Size of the message in bits; and
Processor Speed is the speed of the Processor in bits/second.

Time in Minutes to generate X messages = Time in Minutes to Generate 1 Message * X

where, **X** is the Frequency of message per hour.

% of Processor Time = (Time in Minutes to Generate X messages * 8)/60

Glossary of Terms

Collision Sensor: In the Vehicle, sends an automatic **MAYDAY** signal to the TAC on sensing a collision.

Computer Memory Micro-Map: At the Roadside, holds the changes in the yellow pages information and changes to the local area micro-map.

Localcast Receiver of Micro-Map: In the Vehicle, receives information regarding changes in the micro-map sent by the Roadside.

Mainframe Computer: At the TAC, the central computer that monitors the state-of-the-system and processes information from and sends information to the Vehicle.

Navigation Locator: In the Vehicle, handles requests from the TAC and from the driver of the Vehicle, for information regarding the location of the Vehicle.

Probe Data Response: In the Vehicle, handles requests from the TAC and from the driver of the Vehicle, for information regarding the link traversal time by the Vehicle.

Processor: In the Vehicle, processes information requested by the TAC and the driver of the Vehicle.

RF Receiver:

1) In the Vehicle, receives information sent from the TAC;
2) In the TAC, receives information sent from a Vehicle.

RF Transmitter:

1) In the Vehicle, transmits information processed in the vehicle

to the TAC;
2) At the TAC, transmits state-of-the-system messages to a Vehicle;
3) At the Roadside, broadcasts any local changes to the yellow pages information or changes in the Micro-Map to a Vehicle.

TAC: Traffic Advisory Center, sends state-of-the-system messages to vehicles.

User Input and Display Unit: In the Vehicle, displays requested information to the user, with an interface for input requests.