CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Towards a Fault Tolerant AHS Design Part II: Design and Verification of Communication Protocols

D.N. Godbole, J. Lygeros, E. Singh
A. Deshpande, A.E. Lindsey

California PATH Research Report
UCB-ITS-PRR-96-15

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

Platoon-based cooperative maneuvers for system-level redundancy

June 1996

ISSN 1055-1425

Towards a Fault Tolerant AHS Design*
Part II: Design and Verification of Communication Protocols

D.N. Godbole, J. Lygeros, E. Singh, A. Deshpande, A.E. Lindsey

Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, CA 94720
{godbole,lygeros,ekta,akash,lind@eecs.berkeley.edu}

Abstract

We present the design and verification of inter-vehicle communication protocols for degraded modes of operation on the Automated Highway System (AHS). We consider various hardware and sensor faults that can develop on the automated vehicle in an AHS and design discrete event supervisory controllers to stop the faulty vehicle or take it out of the highway in a safe manner. The protocols are verified for logical correctness by using automatic formal verification tools. This work presents an important step towards analyzing the safety of the AHS.

Keywords

Automated Highway System Design, Fault Detection and Fault Tolerant Control, Safety, Verification, Coordination Layer

*Research supported by the California PATH program, Institute of Transportation Studies, University of California, Berkeley under MOU 135

Executive Summary

The goal of MOU 135 project is to design an extended AHS control architecture to maintain safe operation while minimizing the impacts of abnormal conditions on the highway throughput. In [Varaiya1993] a hierarchical controller was proposed for platooning under normal conditions of operation, i.e., when environment is benign and the operation of the vehicle and infrastructure is faultless. In a companion report [Lygeros *et al.*1995], the normal mode control hierarchy is extended to deal with a wide range of faults and adverse environmental conditions. The extended architecture of [Lygeros *et al.*1995] proposes extended information structure, fault classification and coordinated response strategies.

In this report, we focus on the design and verification of inter-vehicle communication protocols for the coordination layer for degraded mode of operation. The fault tolerant control architecture [Lygeros *et al.*1995] requires additional control laws for each layers of the control hierarchy as the normal mode design is insufficient to deal with all cases of interest. We consider various hardware and sensor faults that can develop on the automated vehicle in an AHS and design discrete event supervisory controllers to stop the faulty vehicle or take it out of the highway in a safe manner. We assume that the faults are irreversible so that once a fault occurs, the vehicle must exit the highway system.

The design is inteded to increase the system autonomy by making it robust against multiple simultaneous faults. In [Hitchcock1994] it was proposed that an AHS will be safe if up to 3 simultaneous faults in any neighborhood can not result in a catastrophe, i.e., a high speed collision. This safety specification relates to the hybrid system formed by the interaction between the discrete (coordination layer) and continuous (regulation layer) controllers. Unfortunately, hybrid system verification tools for such a complex system do not exist at this time. The report aim for a more modest goal. The coordination protocols are verified for logical correctness by abstracting the regulation and physical layers as discrete event systems. However, by means of counterexamples, the safety criterion mentioned above is shown to be too demanding for any design to satisfy.

The extended coordination layer is divided into two levels: a strategic planning level called *coordination supervisor* and an execution level called *coordination maneuver*. Once a fault is detected the capability and performance monitors of [Lygeros *et al.*1995] uniquely classify it into one of the fault classes. Depending on the class or subclass the fault belongs to, the coordination supervisor selects an appropriate strategy. A coordination strategy consists of executing atomic maneuvers in the predefined sequence. The atomic maneuvers depend upon certain capabilities of the neighbors and are executed in coordination with the neighbors.

In the normal mode architecture of [Varaiya1993], mutual exclusion between maneuvers is achieved by imposing the constraint that each platoon leader can be engaged in at most one maneuver at a time. Any other request received during this time is discarder. This normal mode constraint can prove to be unsafe in the presence of faults, as the faulty vehicle must now wait for a normal mode maneuver to finish before it gets assistance from the neighbors. In our design, the constraint that every platoon can engage in only one maneuver at a time is still imposed but, the degraded mode maneuvers are given higher priority than normal mode maneuvers.

The report develops a detailed logical design of the coordination layer strategies and inter-vehicle communication protocols for atomic maneuvers, along with the priority structure for the fault classes based on the severity of a fault. The verification of the proposed design is carried out in two steps. The automatic verification tool, COSPAN, is used to verify logical correctness of the degraded mode design in case of a single fault on the entire highway. The single fault verification is extended to

cover multiple simultaneous faults with nonintersecting fault neighborhoods. The verification task is simplified by taking into account the coordination structure imposed by the protocols. We formally define protocol and fault neighborhoods around a faulty vehicle to identify neighboring vehicles that coordinate with the faulty vehicle and help isolate the fault from the rest of the highway respectively. It is shown that for arbitrary collection of faults, the degraded mode design will not deadlock.

In summary, the fault tolerant controller design described in this report and the companion report [Lygeros *et al.*1995], improves the robustness of automatic vehicle control system to handle adverse environmental conditions and vehicle faults on the AHS. The coordination layer design is formally verified to be logically correct (i.e., does not deadlock and either brings the faulty vehicle to a stop or takes it to the nearest exit safely) under certain assumptions on environment, regulation and physical layers. It is shown that the original safety specification [Hitchcock1994], that up to three faults in a neighborhood do not result in a catastrophe, can not be met. Consequently, refinements to the safety specification are proposed in the report. This design provides a major step towards increasing the autonomy and reliability of automated highway travel.

Contents

List of Figures

1 Introduction

Automated Highway Systems (AHS) are developing into one of the major alternatives to solve the transportation problems of the future. The objective of an AHS is to reduce congestion and increase the throughput and safety of the highway by adding automation to the vehicles and the roadside, without having to build new roads. The AHS design proposed by researchers in the California PATH project builds on the concept of platooning. The basic unit in the system is the "platoon," a group of tightly spaced vehicles (with separations of 1 to 2 meters) moving at relatively high speeds. Platoons are isolated from one another by large gaps, of the order of 30 meters. Packing the vehicles tightly yields significant improvement in highway capacity (estimated to be up to 4 times the current value). Preliminary studies [Varaiya1993] also show that this can be done without an adverse effect on passenger safety. In order to implement such a scheme automatic control of the vehicles is necessary, as human drivers cannot reliably produce the inputs necessary for platooning. The control and coordination of a large number of vehicles poses a formidable problem.

In [Varaiya1993] a hierarchical controller was proposed for platooning under normal conditions of operation, i.e., when environmental conditions are benign and the operation of the vehicles is faultless. The controller is hierarchically organized into four layers and is distributed between the vehicle and the roadside. The backbone of the hierarchical controller is the design and verification of five basic maneuvers: join, split, lane change, entry and exit [Hsu *et al.*1994, Godbole *et al.*1995] that form an alphabet for describing the behavior of an automated vehicle on an AHS. The normal mode controller design is nearing completion by designing feedback control laws for each maneuver and design of roadside controllers to optimize the flow of traffic. In [Lygeros *et al.*1995] the normal mode controller hierarchy was extended to deal with a wide range of faults and adverse environmental conditions. The work presented here forms the backbone for this extended architecture. The extended design of [Lygeros *et al.*1995] defines a set of new maneuvers used for safely removing the faulty vehicles from the system. The maneuvers aim at producing minimal degradation of the highway throughput. The functional description of these maneuvers is outlined in [Lygeros *et al.*1995]. In this paper, we formally specify each of the new maneuvers and mathematically prove that the design is logically correct.

1.1 Problem Formulation

The extended architecture of [Lygeros *et al.*1995] requires additional control laws for each layer of the control hierarchy as the current normal mode design is insufficient to deal with all cases of interest. In this paper we design a set of communication protocols to safely execute the extended coordination layer maneuvers in cooperation with the neighbors of the faulty vehicle. We assume that faults are irreversible so that, once a fault occurs, the vehicle must exit the highway system. According to the extended architecture of [Lygeros *et al.*1995], in case of severe faults (such as steering failure) the vehicle has to stop and wait to be towed while for less severe faults it may be possible for the vehicle to exit the highway on its own[1]. In either case, special coordination and regulation layer strategies are required. In addition, for the former case, link layer intervention may be needed to maintain acceptable system performance. In this paper we present a formal way of specifying the extended coordination layer strategies. We also present verification results that indicate that the proposed strategies produce the desired effect.

[1] If a breakdown lane is available, then the faulty vehicle can stop in the breakdown lane and wait for assistance. The control laws designed in this paper can be easily adapted to take a breakdown lane into account

Recall that the goal of the design of extended architecture is to maintain safe operation while minimizing the impact of faults on the highway throughput. A safety requirement can be formalized using the following definitions, proposed in [Hitchcock1994]:

Definition 1 *A* **Catastrophe** *is a high speed collision. A possible catastrophe is a situation that can give rise to a high speed collision without an additional fault developing in the system.*

Definition 2 *A* **Hazard** *is a situation in which one additional fault leads to a possible catastrophe.*

Using the above definitions the following safety specification was proposed in [Hitchcock1994]:

Definition 3 *The AHS is safe if a hazard does not arise without two simultaneous faults of system components in a small neighborhood in space and time. Thus, the system is safe if up to 3 faults in any neighborhood cannot result in a catastrophe.*

Our analysis will indicate that this safety criterion is too demanding for any design to satisfy. We will show, by means of counterexamples, that some refinement of this criterion is needed and propose two possible techniques for the refinement.

Unfortunately, we have no tools that allow us to verify a safety claim such as the one given above. This specification relates to the hybrid system formed by the interaction between the discrete (coordination) and continuous (regulation) controllers. Hybrid systems are, in general, difficult to analyze.

We aim for a more modest verification goal. We will show, by means of automatic verification, that the protocol design produces no deadlocks and is fair. To prove this claim we will make use of a finite state description of the regulation layer controllers and the environment. The verification would be complete if we could show that this finite state description is indeed an abstraction of the regulation layer behavior. However, this task is not trivial, as indicated by the results in [Godbole *et al.*1994].

Similar problems are encountered when trying to verify claims about the improvement in throughput produced by our design. It should be noted that a *simple* design to satisfy safety requirements could be to stop any faulty vehicle as soon as possible and wait for a tow truck to get it out of the highway. Our design is a lot more complicated and aims at keeping the faulty vehicle moving (without compromising its safety) and letting it get out of the highway on its own, whenever possible. We are willing to deal with the *higher degree of complexity* because, intuitively, it seems that our approach would lead to less severe disruption of the flow of traffic. Proving such a claim is not easy, however, as it involves the ill-understood interaction between the link and coordination layer models. The increased complexity of the design also gives rise to increased complexity of verification of the safety of the overall hybrid system. We refer the reader to [Lygeros *et al.*1995] for a more thorough discussion of optimality and design tradeoffs for the extended controller.

1.2 Outline

The paper is arranged in four sections. In Section 2 we describe in detail the protocol design and show how it fits in the framework for emergency maneuver initiation and fault classification presented in [Lygeros *et al.*1995]. In Section 3 we describe how the design was modeled formally using finite state machines and verified using automatic verification tools. We explore the implications of the verification results and relate them to the original goals set for the design. This indicates directions in which our work can be extended or modified; these are summarized in the concluding section.

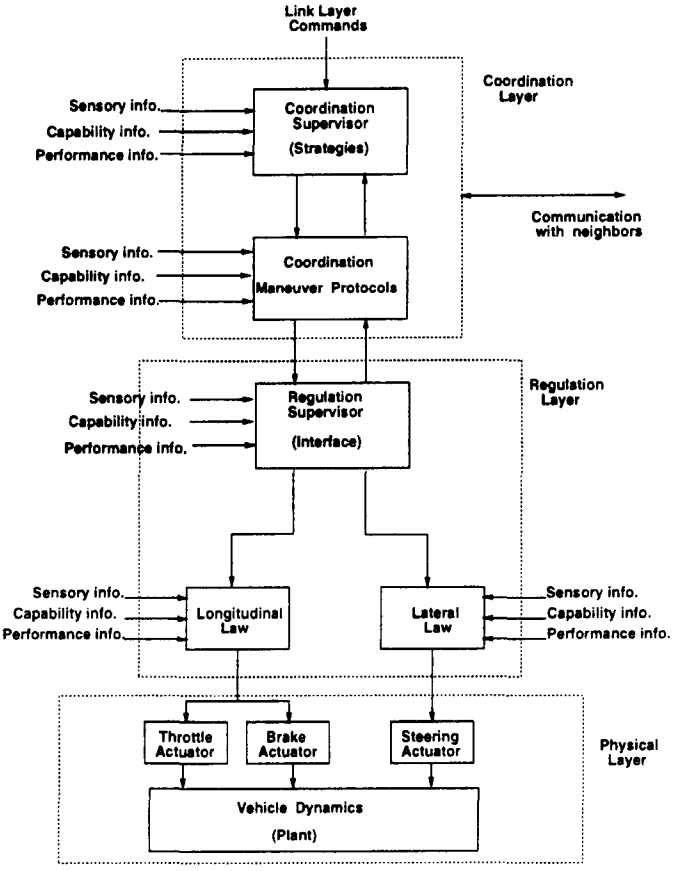

Figure 1: Structure of the controller

2 Design of Coordination Protocols

The fault classification of [Lygeros *et al.*1995] considers vehicle as well as infrastructure faults including adverse weather conditions. As this paper deals with extended coordination layer protocols, we consider only vehicle faults.

2.1 Structure of the controller

According to the framework of [Lygeros *et al.*1995], the extended coordination layer controller is divided into two levels (Figure 1): a strategic planning level called *coordination supervisor* and an execution level called *coordination maneuver*. Once a fault is detected the capability and performance monitors of [Lygeros *et al.*1995] uniquely classify it into one of the fault classes. Depending on the class or subclass the fault belongs to, the coordination supervisor selects an appropriate strategy. The extended architecture design of [Lygeros *et al.*1995] outlines new strategies and maneuvers to be used for each fault class. The objective of the design is to guarantee safe operation of vehicles on the AHS in the presence of faults. This must be achieved without large negative effect on overall system capacity.

For safety reasons, the faulty vehicle should be taken off the AHS. The degraded mode design of [Lygeros *et al.*1995] consists of new strategies and maneuvers that either stop the faulty

vehicle or take it to the nearest exit. Execution of the strategies depends upon certain capabilities of the neighbors. The control strategies are executed by following a sequence of maneuvers. In this paper, the individual maneuvers at the coordination layer are called *atomic* maneuvers and the coordination strategies are called *compound* maneuvers. The coordinating maneuver protocols and supervisor strategies are modeled using finite state machines (FSM). We now describe these coordination layer controllers in greater detail. The FSM diagrams representing the protocols are given in Figures 9–15. The protocols unambiguously define the strategies and maneuvers. This FSM description is in turn used to verify their behavior in Section 3.

2.2 Coordination supervisor strategies (compound maneuvers)

The extended coordination layer strategies can be categorized into two classes. For the most severe faults, strategies are required to stop the vehicle on the highway. Three strategies are defined for this purpose in [Lygeros *et al.*1995], namely, *Gentle Stop*, *Crash Stop*, and *Aided Stop*. Once the vehicle comes to rest, the link layer employs specific control laws to ease congestion, divert traffic away from the incident, assist emergency vehicles, and get the queued vehicles out. In this case, the performance of an entire section of the highway is degraded from the normal mode performance.

For faults in the class "vehicle needs assistance to get out", and "Vehicle needs no assistance to get out", coordination layer strategies are designed so as to get cooperation from the neighbors to get the faulty vehicle out of the highway, *without stopping the traffic* on the highway. Three such strategies are defined in [Lygeros *et al.*1995], namely, *Take Immediate Exit*, *Take Immediate Exit - Escorted* and *Take Immediate Exit - Normal*.

Gentle Stop and Crash Stop Strategies: The goal of these control strategies is to bring the faulty vehicle to a complete stop on the highway. These strategies in turn make use of *gentle stop* and *crash stop* maneuvers. The names of the strategies indicate the severity of braking used to execute the maneuver. They do not need any assistance from the neighboring vehicles. (see Figures 16, 18 for the FSM diagrams.)

Aided Stop Strategy: The faulty vehicle with a "brakes off" failure is aided by the vehicle immediately ahead of it in the same platoon to come to a stop. If the faulty vehicle is a leader, it uses the *front dock* maneuver to become a follower before executing *aided stop* maneuver (see Figure 11). The faulty vehicle does not accelerate, using the engine friction to slow down, while the assisting vehicle applies gentle deceleration to let the faulty vehicle collide from behind. Then the assisting vehicle uses its brakes to bring the combined mass of the two vehicles to a halt.

Take Immediate Exit (TIE): The *Take Immediate Exit* strategy (Figure 9) is used by the faulty vehicle to get out of the AHS as soon as possible. The strategy consists of up to two *forced split* maneuvers to make the faulty vehicle a free agent. This is followed by a sequence of *emergency lane change* maneuvers until the vehicle reaches the rightmost automated lane from where it takes the next exit.

Take Immediate Exit - Escorted (TIE-E): This strategy is used by a faulty vehicle that has lost the capability to be a platoon leader but can still be a follower. In this case, the faulty vehicle leaves the system as part of a two vehicle platoon in which the faulty vehicle is the follower. This requires up to two *forced split* maneuvers (if the faulty vehicle is a follower) or a *front dock* and possibly a *forced split* maneuver (if the faulty vehicle is the leader of its

platoon). The leader of this new platoon (called the *escorting vehicle*) escorts the faulty vehicle out of the AHS by executing sequence of *emergency lane change* maneuvers of the two vehicle platoon to take an exit immediately. Once out of the AHS, the escorting vehicle drops off the faulty vehicle in a special turnout called "dormitory" and re-enters the AHS at the next entrance (see Figure 10).

Take Immediate Exit - Normal (TIE-N): This strategy is similar to the TIE strategy except the faulty vehicle uses normal lane change and split protocols of [Hsu *et al.*1994] instead of *emergency lane change*. The FSM diagram for TIE-N is similar to Figure 10, except the state ELC is replaced by LC.

Normal: This is the normal mode strategy defined by the normal mode AHS architecture of [Varaiya1993].

2.3 Atomic Maneuvers

The following atomic maneuvers are designed to execute the strategies. The finite state machines (Appendix A) represent the inter-vehicle coordination required to carry out these maneuvers.

Forced Split: This maneuver is similar to the split maneuver of [Hsu *et al.*1994]. It is used by a faulty vehicle to become a free agent. If the faulty vehicle is a follower it requests the leader of the platoon to initiate a *forced split*. The leader breaks the platoon at any desired location (see Figure 13). We use the following terminology to identify the location of forced split: FS_0 indicates that the faulty vehicle will be the leader of the new platoon. Similarly, the value of n in FS_n indicates the location of the new platoon leader relative to the faulty vehicle, with positive values corresponding to vehicles in front of it.

Emergency Lane Change: This maneuver is used by a free agent with reduced capabilities. The faulty vehicle requests the leader of the platoon in the adjacent lane to create and maintain a gap so that the faulty vehicle can change lane into it. A special case of *emergency lane change* is also defined for a platoon of vehicles to change lane into a gap (used for TIE-E) (Figure 14).

Front Dock: This maneuver (Figure 15) is initiated by a platoon leader that wants to join with the vehicle in front but, because of a fault, it cannot execute the *join* maneuver of [Hsu *et al.*1994] which requires accelerating to close the gap and then decelerating to match the speed. The initiating vehicle requests the leader of the preceding platoon for a *front dock*. The leader of the preceding platoon orders the last vehicle in its platoon (itself in the case of a free agent) to front-dock with the initiator. This vehicle will then decelerate to close the gap between itself and the initiator. In the end, the initiator becomes the first follower of the new platoon. Thus the leader of this new platoon joins with the trailing platoon by decelerating.

Aided Stop: This maneuver is initiated by a follower that has developed a brakes off failure. The faulty vehicle uses its engine to decelerate and asks the leader to assist in bringing it to a stop on the highway. The responding vehicle (the vehicle immediately ahead of the faulty vehicle) applies *gentle* braking and lets the faulty vehicle collide with it from behind. The responding vehicle then uses its brakes to bring the combined mass of both the vehicles to a stop (see Figure 12).

Gentle Stop: This maneuver is used by a faulty vehicle that is ordered to stop and can do so by using its brakes. The fault is not severe enough to require the vehicle to use maximum

emergency braking. The vehicle will use gentle braking in order to minimize the disturbances to following vehicles (see Figure 17).

Crash Stop: This maneuver is similar to *Gentle Stop* except the severity of the fault requires the faulty vehicle to apply maximum emergency braking (see Figure 19).

Queue Buildup and Queue Management Whenever a faulty vehicle is stopped on the highway, vehicles in the same lane immediately behind the faulty vehicle will form a queue of stopped vehicles. The extended link layer controller is designed to stop the queue buildup by diverting traffic upstream of the stopped vehicle from the blocked lane to the other lanes. If the emergency vehicle (e.g. tow truck) has not appeared until the queue buildup has stopped, we use the following strategy to get the queued vehicles moving again. In our strategy, the queue is dissipated in Last In First Out (LIFO) fashion. We use the *queue buildup* maneuver to keep track of the current leader of the queue. With a LIFO strategy, the queue leader is the last vehicle of the queue. When the queue buildup stops, there is a large gap behind the last vehicle of the queue. This gap is created by the link layer strategies of diverting the traffic upstream of the stopped vehicle. This is the proper stage for *queue dissipation*.[2] Queue dissipation will be carried out in some fixed platoon sizes. The link layer orders creation of gap in the adjacent lane upstream of the accident. This gap will travel towards the queued up vehicles. At this time, a platoon of appropriate size (depending on the size of the gap in the adjacent lane) breaks up from the queue and backs up into the gap behind the queue. This platoon will stop its backward motion when it creates a sufficient gap in between its front vehicle and the last vehicle of the remaining queue. The platoon is now ready to accelerate up to the speed of the next lane and change lane (using *emergency lane change for the platoon*) into the gap in the next lane that is approaching it. The backup distance depends on the speed of the approaching gap and the constraints on acceleration and jerk (see Figures 20, 21).

2.4 Priorities and mutual exclusion

In the normal mode architecture of [Varaiya1993], mutual exclusion between maneuvers was achieved by imposing the constraint that each platoon leader can be engaged in at most one maneuver at a time. Any other request received during this time is discarded. This constraint assures basic level of safety while carrying out the maneuvers. For example if two platoons are executing a *join* maneuver, the above constraint will not allow initiation of a split maneuver within the preceding platoon which could give rise to a crash. This normal mode constraint can prove to be unsafe in the presence of faults, as the faulty vehicle must now wait for a normal mode maneuver to finish before it gets assistance from the neighbors. In our design, the constraint that every platoon can engage in only one maneuver at a time is still imposed but, the degraded mode maneuvers are given a higher priority than normal mode maneuvers.

As a maneuver can be utilized by two different strategies, we assign priorities to different strategies and allow a higher priority maneuver request to preempt and abort an ongoing maneuver representing a lower priority strategy. The state *Busy?* in all the FSM diagrams implements this priority scheme. The normal mode strategy has the lowest priority. As the normal mode strategy is not safety critical, if a normal mode maneuver gets aborted, the vehicle still follows the default normal mode strategy. On the other hand, if a degraded mode maneuver gets aborted, the faulty vehicle has to choose another strategy to safely exit the highway. Thus for a complete specification of

[2] [Lygeros *et al.*1995] contains details of the degraded mode link layer design for queue dissipation.

the extended coordination layer, we need a priority assignment (possibly dynamic) and an algorithm to determine a new strategy after a maneuver gets aborted. These problems are addressed in Section 3.4.2.

Let us assume, at this time, that any vehicle can develop only one fault at a time and multiple faults on the highway are isolated by sufficiently large distance (to be made precise in Section 3). This ensures that at the time of occurrence of a fault, all neighbors of the faulty vehicle are executing normal mode strategy.

3 Formal Specification and Verification

3.1 Verification Scope

We show that the sequence of actions commanded by the above design results in the faulty vehicle either stopping or exiting the system and all other vehicles returning to their normal operation. The verification considered here only deals with the discrete event aspects of the problem. The questions that we answer are related to the logical correctness of the protocols, i.e. fairness, and absence of deadlocks.

Important questions concerning the safety of the design and the resulting system performance cannot be formulated in a discrete event framework. These questions relate to the hybrid nature of the system; they are discussed further in the companion paper [Lygeros *et al.*1995].

3.2 Verification Framework

Examination of the degraded mode protocols reveals that platoons in the neighborhood of a faulty vehicle can be classified based on the messages they transmit or receive during the execution of the chosen strategy.

Faulty platoon: This platoon contains the faulty vehicle which can transmit any degraded mode message. The faulty vehicle could be a leader or a follower either in the middle or at the end of the platoon.

Assisting platoon: This platoon accepts all normal or degraded mode messages from other platoons. The assisting vehicles are directly involved in executing the degraded mode maneuvers.

Isolating platoon: This platoon accepts all normal mode messages and only the "Abort" degraded mode message. The isolating platoon may be engaged in a normal mode (or a lower priority degraded mode) maneuver with either the faulty or the assisting platoon. At the onset of a fault, this maneuver is aborted and then the isolating platoon is no longer directly involved in assisting the faulty vehicle.

Exterior platoon: This platoon accepts only normal mode messages.

Figure 2 shows such a classification of platoons.

For the purpose of verification, we extract a discrete representation of the highway by imposing a grid structure on it (see Figure 3). Each cell in the grid can be occupied by at most one platoon. The cell is labeled 'F' if it is occupied by a faulty platoon, 'A' if it is occupied by an assisting

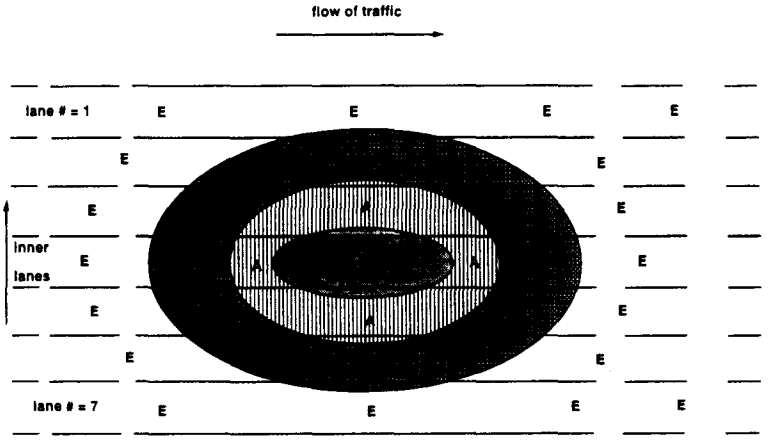

Figure 2: Highway Snapshot

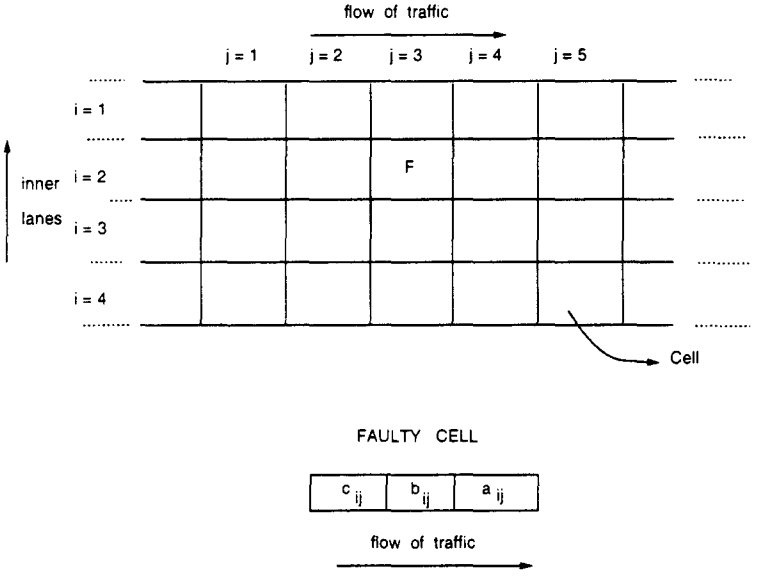

Figure 3: "Grid"

platoon, 'I' if it is occupied by an isolating platoon, 'E' if it is occupied by an exterior platoon, 'O' if it is empty, and 'S' if it is occupied by a platoon that has come to a stop on the highway. Given a set of protocols being followed by platoons, these adjacency relations between platoons are implicitly defined by these protocols. This relationship is consistent since each platoon can engage in at most one protocol at a time.

In this verification framework, using the grid and the cell labels, execution of each degraded mode protocol can be graphically represented as a set of sequences of grid snapshots. To illustrate this, in Figure 4, we depict a series of grid snapshots for the Force Split and Emergency Lane Change protocols.

By observing the evolution of the label changes, in the series of grid snapshots for each protocol we define two kinds of neighborhoods for the faulty cell.

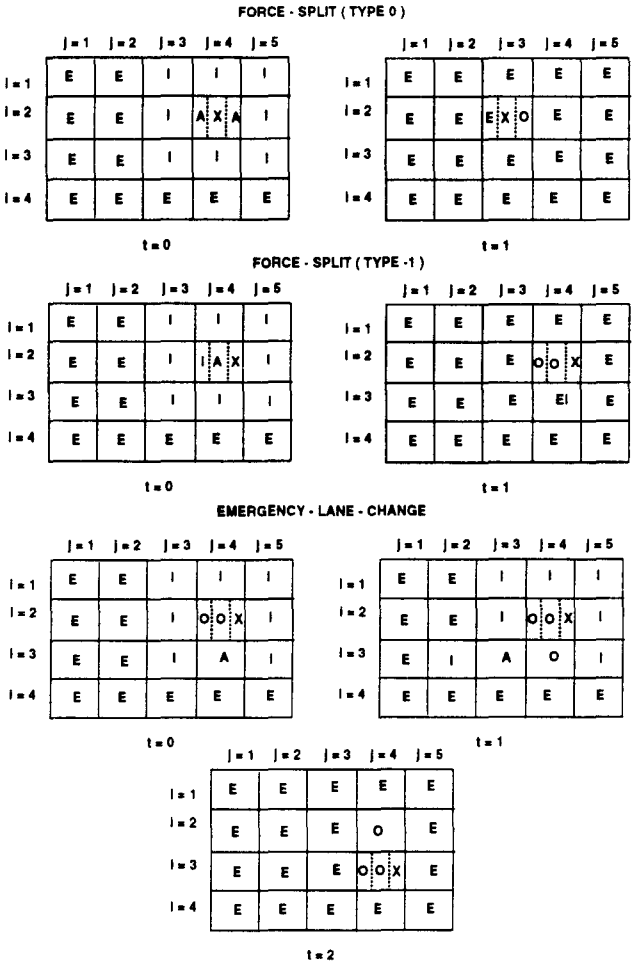

Figure 4: Series Of "Grid Snapshots"

Definition 4 *The* **Protocol Neighborhood** *of a faulty cell consists of the faulty cell and the Assisting cells in a grid snapshot. The* **Fault Neighborhood** *of a faulty cell consists of the faulty cell, the Assisting cells, and the Isolating cells in a grid snapshot.*

Thus far, the grid size is undefined. These observations of the neighborhood of the faulty cell highlight that the only cells of importance to us, are the 'F', 'A', and 'I' cells. By taking the union, over all the DM protocols proposed in Section 2, of all the 'F', 'A', and 'I' cells, we get a maximal grid size of 18 cells. The grid now becomes a 4x4 matrix of cells with the faulty cell further subdivided into three sub-cells.

Using the above grid size, however, we face a potential problem when dealing with scenarios where, if we anchor one of the grid cells to the faulty vehicle, a sequence of maneuvers (e.g., two forced splits) may cause 'A' or 'I' cells to fall out of the grid snapshot. In order to accommodate these scenarios in our framework, we define a new transformation which establishes equivalence between grid snapshots.

Definition 5 *During the execution of a maneuver, a grid snapshot is said to be* **e-transformed** *to a new grid snapshot, if and only if one or more columns of 'E' cells are added to, or removed from it.*

This e - transformation or the lateral displacement of the grid with respect to the faulty cell results in equivalent grid snapshots, and is used to test for trace equivalence during verification. This grid size is now optimal because all possible traces for any given "DM" protocol can be represented in it with the F cell appropriately located in the grid, i.e., for each DM protocol there exists a location for F cell in the 4×4 grid such that all the A and I cells are accommodated in this grid.

In order to use the grid snapshots for the specification and verification of the degraded mode protocols, we need to impose two key requirements. The first deals with initial conditions. The initial conditions for a given DM protocol consist of the position of the faulty cell in the grid, position of the faulty vehicle in the platoon, and the labels on the remaining cells along with the status of their involvement in another maneuver if any. Different initial conditions lead to a different series of grid snapshots for the same protocol. This is important as each series of grid snapshots generates a corresponding trace during verification. We make use of different initial conditions in our verification framework to get rid of the ambiguity in allocation of platoons to the grid cells as mentioned above. Thus, we verify the correctness of protocols for all possible initial grid snapshots which are consistent with the given situation.

The second requirement we need to impose has to do with concatenation. By looking at the multilayered design of the coordination layer protocols, we propose a hierarchical verification strategy, wherein by verifying the simpler and smaller "atomic" maneuvers, we will be able to verify the more complex and larger, "compound" maneuvers. To do this, we define a new operator, called *concatenation*, in our framework.

Definition 6 *Atomic maneuvers may be* **concatenated** *if and only if the initial grid snapshot of one atomic maneuver is equivalent to the final grid snapshot of the previous atomic maneuver.*

Note that in doing so, we consider two grid snapshots *equivalent* if one of the following is true:

- They are related by an e-transformation
- They are related by replacing I or E cells by O cells and vice versa.

3.3 Single Fault: COSPAN Verification

With this framework, we now proceed with the automatic verification of the protocols for individual maneuvers and strategies of the extended coordination layer. In this part of the verification, we assume that there is only one fault on the entire highway. We will prove that the individual coordination protocols are logically correct.

The full DM controller is fairly large, involving a machine with about 5×10^{20} states. To simplify the verification task we propose a strategy involving one creative step, and one methodological approach. The creative step is to structure the controller in a hierarchy with modular components. In our exercise, the DM controller is organized in four sublayers. The lowest sublayer is expressed as a set of individual protocol machine components which are invoked by supervisory machines (see Figure 5). The components and sublayers are then formally specified and verified individually. A good structure ensures that the sublayers and components are small and readily understood, and the interaction between them is minimized. The advantages of the proposed hierarchical verification strategy are as follows:

1. Smaller and more tractable state space.

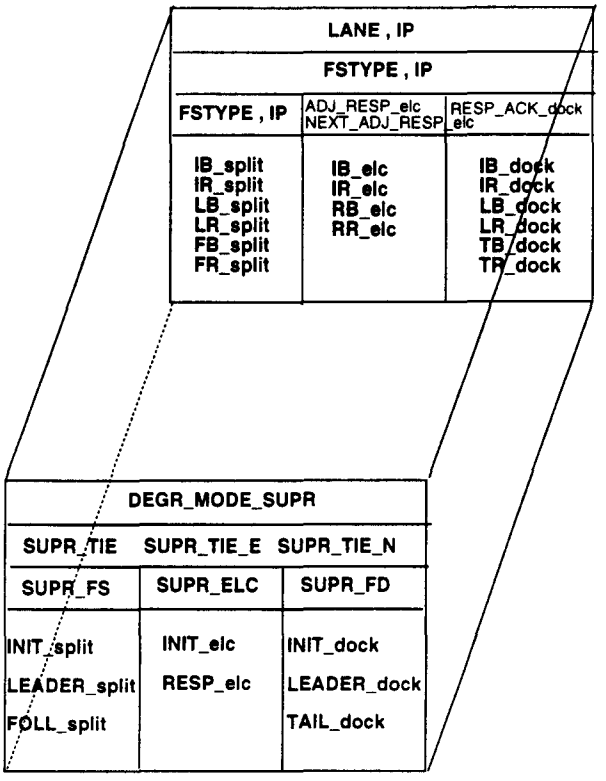

Figure 5: The Four Sublayers of The Degraded Mode Controller

2. Faster and easier verification of the atomic maneuvers, along with easier debugging and error detection.

3. The use of concatenation allows us to build and verify new compound maneuvers by executing the verified atomic maneuvers in a predetermined sequence.

The methodological approach is used to convert the informal FSM into the formal FSM, and the subsequent translation into COSPAN. The DM protocols are informally specified in Section 2, with the FSM contained in the appendix. The FSM are "informal", since their states and transitions refer to actions and conditions that may depend on the Regulation layer, on information from sensors on-board the vehicle, and on information from roadside monitors. These are not part of the protocol machines themselves. The requirement that the system to be verified be *closed* forces us to build additional finite state machines to account for all the interactions between the controller and its environment.

3.3.1 Formal FSM Representation of the Protocols

In order to convert the informal FSM into formal FSM, the references to "environment" actions and conditions need to be represented by separate FSM. In addition, we use the fact that the protocols for the normal mode of operation have been verified [Hsu *et al.*1994], to abstract their behavior into smaller FSM.

Figure 5 shows a compact representation of all the machines that are needed for closed verification of three of the DM strategies; TIE, TIE-E and TIE-N. The panel on the left consists

of the informal state machines for the maneuvers and the supervisors introduced for verification. The lower layer contains informal FSM for three atomic maneuvers FS, ELC and FD. For example, **INIT-split**, **LEADER-split** and **FOLL-split** are the informal state machines for the initiator, leader and responding follower in the FS protocol as seen in Figure 13. The second level corresponds to three verification supervisors; **SUPR-FS**, **SUPR-ELC**, and **SUPR-FD** are introduced for the verification of the corresponding atomic maneuvers. The next higher level, includes the informal FSM for the three compound maneuvers. The DM supervisor **DEGR-MODE-SUPR** in the highest level, receives a command from the coordination layer supervisor to start the DM of operation. It considers the faulty vehicle's current lane number and position within its platoon, selects the corresponding compound maneuver, and instructs the corresponding supervisor, **SUPR-TIE**, **SUPR-TIE-E** or **SUPR-TIE-N**, to start the maneuver. In response, these supervisors for the compound maneuvers invoke the supervisors for the atomic maneuvers: **SUPR-FS**, **SUPR-ELC**, and **SUPR-FD**.

These 16 FSM constitute part of[3] each vehicle's DM co-ordination layer. The **DEGR-MODE-SUPR** is running at all times, whereas the FSM in the lower sublayers are invoked as needed, depending on the role (initiator or responder) a vehicle plays in a particular DM protocol. A FSM which is not invoked remains in its IDLE state.

The supervisor machines embody the mutual exclusion rule that, a platoon may engage in at most one maneuver at a time. This leads to modular verification, as the specification and verification of each maneuver is done separately. The one-maneuver-at-a-time rule is implemented by two devices: First, by having a single BUSY flag for each vehicle: **IB** for the **INIT** vehicle (Figure 22), and **RB**, **LB**, or **TB** for the various types of responders. This flag is set to "normal" mode busy, if the vehicle is engaged in a normal mode protocol, "DM" busy, if it is engaged in a DM protocol, or not-busy otherwise. Second, contention among simultaneous requests for normal and DM protocols is resolved by awarding highest priority to the DM of operation. A vehicle in DM commandeers its responders, be they in normal mode busy or not-busy mode, and forces them to set their flags to "DM" busy. Note that currently we are considering a single fault scenario, wherein only a single "DM" protocol is triggered.

The panel on the right of Figure 5 lists another collection of FSM, arranged in four sublayers corresponding to the ones on the left panel. They specify the "environment" within which the DM controller operates. They are of three types:

1. Interface between co-ordination layer and the Link and Regulation layers.
2. Interface to the Sensors (on-board and roadside)
3. Initial "state" information of the vehicle: lane number, position, and busy status.

The environment FSM "close" the system around the coordination layer so that verification is possible.

3.3.2 Protocol Processes

Due to space limitations, we describe only the Emergency Lane Change protocol in detail.

Emergency-Lane-Change protocol The ten processes in ELC are shown in Figure 6. The

[3] A complete figure will include the three strategies (GS,CS,AS) used for stopping the vehicle as well. We have not included them in this picture or this discussion to keep the figure and the paper at a reasonable size.

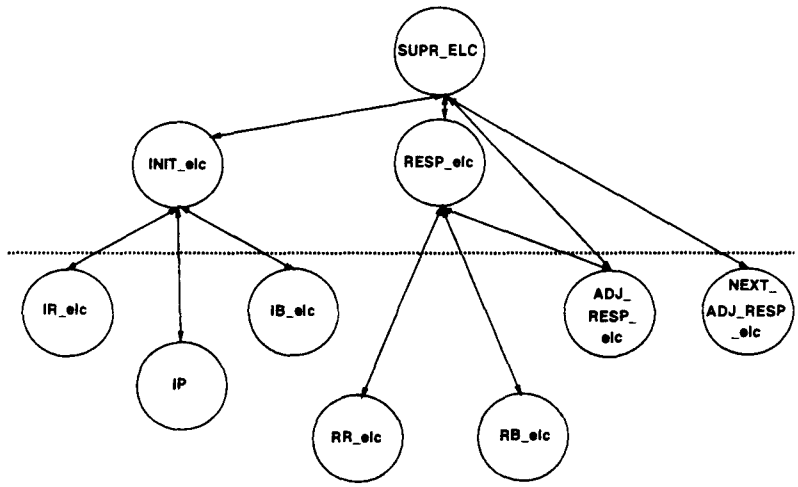

Figure 6: Emergency Lane Change Maneuver Processes

three processes above the horizontal line are the protocol processes, while the ones below represent the environment. Double arrows indicate information links. In terms of COSPAN this means, for example, that the state transition predicates of **INIT-elc** involve only the outputs of **IR-elc**, **IB-elc**, and **IP**. Sample environment processes are specified in Figure 22.

The **INIT-elc** must coordinate with the **RESP-elc** before it can change lane. A distance sensor detects a neighboring vehicle within a certain range. **ADJ-RESP-elc** and **NEXT-ADJ-RESP-elc** model this sensor's response. **IR-elc** and **RR-elc** are the interface with the Regulation layer, when it is implementing the change lane feedback law, for **INIT-elc** and **RESP-elc** respectively. **IB-elc** and **RB-elc** indicate the status of the busy flag and the mode of operation, for **INIT-elc** and **RESP-elc** respectively. **IP** depicts the current position of **INIT-elc**, at all times during the execution of any "DM" maneuver. The three protocol processes are specified in Figure 23 and 24.

SUPR-ELC is normally in the IDLE state. It initiates the ELC protocol when **SUPR-TIE.** = start-ELC (i.e., when the vehicle's **SUPR-TIE** machine selects ELC). In verifying the ELC protocol by itself this predicate is replaced by "true", so that this maneuver is repeatedly initiated. The **INIT-elc** protocol machine responds to the condition **SUPR-ELC.** = start. It responds affirmatively after checking various conditions, especially the busy status. **INIT-elc** in turn communicates and commands the **RESP-elc** and the maneuver proceeds as depicted.

3.3.3 Supervisory Processes

We only present the Take-Immediate-Exit supervisor in detail. **SUPR-TIE** consists of two protocol processes: **SUPR-FS**, **SUPR-ELC**, and three environment processes: **IP**, **LANE**, and **FSTYPE**. The supervisor for TIE is shown in Figure 25.

3.3.4 Verification Results

The formal FSM described in the previous section were transcribed into the language used by the verification software, COSPAN [Har'El and Kurshan1987]. The design was verified by using a set of monitors. The monitors themselves are finite state machines whose transitions depend on the states

and transitions of the design FSM. The monitor FSM generate their own language and COSPAN proves that the language generated by the design is contained in the language generated by the monitors. The debugging of the design is facilitated by the detailed error trace generated by COSPAN upon failure of the task.

FS
Monitor TASK 1 checks for the logical correctness of the FS protocol, by verifying the following behaviors:

- FS is verified to be deadlock free and fair by checking that the protocol always completes and that no vehicle gets an abort from its regulation layer forever.
- Once the atomic maneuver is properly completed the position of the faulty vehicle in the grid is verified to be the correct one.

Monitor TASK 2 is designed to check the consistency between the initial and final position of the vehicles in a platoon. This is necessary for concatenation of atomic maneuvers.

FD
Monitor TASK 1 checks for the logical correctness of the FD protocol, by verifying the following behaviors:

- FD is verified to be deadlock free and fair by checking that the protocol always complete and the regulation layer does not abort the request forever.
- Once the atomic maneuver is properly completed the position of the faulty vehicle in the grid is verified to be the correct one.

Monitor TASK 2 is designed to check the consistency between the initial and final position of the vehicles in a platoon. This is necessary for concatenation of atomic maneuvers.

Figure 26 contains three monitors used for verifying the ELC atomic maneuver. By defining accepting conditions in terms of desired states which occur infinitely often (*cyset* notation in COSPAN), or edges which represent desired behavior (*recur* edges notation in COSPAN), the correctness of the protocol can be verified.

ELC
Monitor TASK 1 checks for the logical correctness of the ELC protocol, by verifying the following behaviors:

- **INIT-elc** is called upon to perform ELC, if and only if the faulty vehicle is a FREE AGENT.
- **INIT-elc** never gets a negative acknowledgement, due to the **RESP-elc** being "DM" busy, because we only consider a single fault scenario.
- ELC maneuver always completes (liveness of the protocol).
- The fairness of the protocol is checked, by verifying that no vehicle gets an abort from its regulation layer forever. Eventually, it gets a success from it.

Monitor TASK 2 checks that on completion of the ELC maneuver, the **LANE** machine reflects the correct current lane position of the vehicle, i.e., lane number or exit from the highway.

Monitor TASK 3 checks that no crashes occur between the **INIT-elc** and the **RESP-elc** vehicles, by ensuring that **INIT-elc** moves into the adjacent lane, only after receiving confirmation from the **RESP-elc** vehicle (i.e., **RESP-elc** has finished decelerating, a gap has been created and is being maintained for **INIT-elc**). If there is no **RESP-elc** vehicle in the adjacent lane, **INIT-elc** coordinates and receives confirmation from the vehicle, if any, in the next-to-adjacent lane, and thus ensures that they do not crash, by moving into the available gap simultaneously.

The verification of compound maneuvers is also carried out by constructing monitors. TIE-N is a concatenation of two normal mode maneuvers namely split and lane change. Since all normal mode maneuvers have been previously verified we expected TIE-N to verify successfully. We verified TIE, using three monitors as follows:

TIE
Monitor TASK 1 checks logical correctness of the Take-Immediate-Exit protocol, by verifying that

- The various initial conditions necessary for performing FS and ELC are satisfied. For example, it is verified that an FS_0 is issued only when the faulty vehicle is a follower and an FS_-1 is issued only when the faulty vehicle is a leader.
- INIT should never get a nack, due to the RESP being DM busy, because we are considering a single fault scenario.
- The fairness of the protocol is checked, by verifying that no vehicle gets an abort from its regulation layer forever. Eventually, it gets a success from it.
- The entire compound maneuver is constructed by a concatenation of atomic maneuvers, executed in predetermined sequence and under compatible initial conditions.

Monitor TASK 2 checks that on completion of each ELC maneuver, the **LANE** machine reflects the correct current lane position of the vehicle.

Monitor TASK 3 checks that on completion of each FS maneuver, the **IP** machine reflects the correct current position of the vehicle.

A similar procedure is followed for Take-Immediate-Exit-Escorted and Take-Immediate-Exit-Normal supervisors. However, TIE-E involves one more protocol process, **SUPR-FD**. In testing the higher level supervisors: **SUPR-TIE**, **SUPR-TIE-E**, and **SUPR-TIE-N**, we use a monitor FSM to check that concatenations of the "atomic" maneuvers occur, based on valid "e-transformations" of their "grid snapshots".

The following properties are verified for the three stop maneuvers:

Aided Stop
Monitor TASK 1 checks for the logical correctness of the AS protocol, by verifying the following behaviors:

Maneuver Verified	Initial States	States Reached	Edges Traversed	CPU Time (Sec)	Result
Gentle Stop	4	270	364	0.016	Task Performed
Crash Stop	4	108	130	0.016	Task Performed
Aided Stop	1536	108138	176378	47.3667	Task Performed
Queue Management	128	11168	14496	2.43	Task Performed
TIE	32	50568	79958	27.5	Task Performed
TIE-E	256	466808	515344	294.68	Task Performed

Table 1: Verification Results using Sun-Sparc 20

- AS maneuver is deadlock free and always completes, i.e., liveness.
- The fairness of the protocol is checked, by verifying that no vehicle gets an abort from its regulation layer forever. Eventually, a success occurs.

Monitor TASK 2 checks whether Various initial conditions necessary for performing ASTOP are satisfied. For example, it is verified that an FS_-1 is issued only when the faulty vehicle is a follower other than the last vehicle.

Gentle Stop
Monitor TASK 1 checks for the logical correctness of the GS protocol, by verifying that the GS maneuver is deadlock free and always completes, i.e., liveness.

Crash Stop
Monitor TASK 1 checks for the logical correctness of the CS protocol, by verifying that the CS maneuver is deadlock free and always completes, i.e., liveness.

Queue-buildup
Monitor TASK 1 checks logical correctness of the Queue-Buildup (QB) protocol, by verifying that the QB maneuver is deadlock free and always completes, i.e., liveness.

Queue-management
Monitor TASK 1 checks logical correctness of the Queue-Management (QM) protocol by verifying that the QM maneuver either completes or aborts. In the case of abort the maneuver reduces to QB.

The statistical results of COSPAN verification are summarized in Table 3.3.4 During the verification process, some errors, in message passing and event synchronization between vehicles were found in the original design of the protocols. The necessary modifications were made in the design, and the redesigned protocol were reverified. This process was iterated until no bugs could be found and the COSPAN output was, "Task Performed".

3.4 Multiple Faults

The definitions of neighborhood, assisting and isolating vehicles used in the above verification are intuitive and unambiguous in the case where there is only one faulty vehicle on the entire highway. We would like to be able to extend these ideas (and hence the results of our verification) to the case of multiple faulty vehicles that do not interact. In order to do this we need to make the notion of interaction more precise:

Definition 7 *The* **duration** T^j **of a fault** f^j *is a union of intervals*

$$T^j = [t_0^j, t_1^j] \cup [t_1^j, t_2^j] \cup \ldots [t_{n-1}^j, t_n^j]$$

such that:

- t_i^j *are transitions times in the degraded mode protocols for fault* f^j
- *The neighborhood of the faulty vehicle (defined as above) remains the same for the entire duration of the interval* $[t_i^j, t_{i+1}^j]$.

Remarks:

- t_i^j need not be all the transition times for the protocols, as a number of events may happen without the neighborhood changing.
- We allow $t_i^j = t_{i+1}^j$, i.e. and instantaneous change of neighborhood. In fact this phenomenon will be quite common when dealing with isolating vehicles.
- t_n^j may be infinite. This situation will be encountered when the faulty vehicle has to stop and wait for a tow truck. In this case the coordination layer degraded mode protocols terminate in finite time, but the faulty vehicle remains on the highway. As we are not concerned with the operation of the link layer (in this report), we will not keep track of the time after the faulty vehicle stops.
- The verification described in the previous section shows that the protocols terminate in a finite number of steps, i.e. that n is finite. In the case where the faulty vehicle exits the highway, we would eventually like to show that the actual time t_n^j is finite. This proof will involve the interface between the coordination and regulation layers. We hope to be able to show this by means of automatic verification, under minimal and realistic assumptions about the regulation layer (see [Lygeros and Godbole1994] for related ideas for normal mode operation).

Using the above definition we can now define the extent of a fault. Assume that the entire highway has been divided into cells like the ones discussed above and let l be the number of lanes. Then the position of the faulty vehicle f^j at time t can be parametrized by a vector $F^j(t) \in \mathcal{Z} \times \{1, \ldots, l\}$ where the first entry denotes the longitudinal and the second entry denotes the lateral position of the vehicle. Let N_i^j denote the fault neighborhood of faulty vehicle j in the interval $[t_i^j, t_{i+1}^j]$, defined relative to the position of the faulty vehicle as before. Then $F^j(t) + N_i^j$ where $t \in [t_i^j, t_{i+1}^j]$ will give the position of the fault neighborhood at time t in absolute coordinates. Here the addition denotes the usual addition in $\mathbb{R}^2$. Note that the exact shape of N_i^j will depend on the position of the faulty vehicle $F^j(t)$ (in particular the lane number), the type of fault f^j and the value of i (how far along in the degraded mode protocol we are).

Note that the involvement of isolating I cells (vehicles) in this fault clearing process is only in the beginning of any atomic maneuver. In particular, an isolating vehicle may be involved in a normal mode maneuver with either an assisting vehicle or the faulty vehicle before the fault occurs. Due to the higher priority assigned to the degraded mode maneuvers, the maneuver of the isolating vehicle gets aborted. From this time onwards, the isolating vehicle is free to engage in any other maneuver with the external vehicles in its neighborhood. Therefore, our definition of N_i^j includes I cells only at the very beginning of any atomic maneuver.

Definition 8 *The* **extent of fault** f^j *is the set:*

$$\mathcal{E}^j = \bigcup_{i=0}^{n-1} \left(\bigcup_{t=t_i^j}^{t_{i+1}^j} (F^j(t) + N_i^j) \times t \right) \quad \subset \mathcal{Z} \times \{1, \ldots, l\} \times \mathbb{R}$$

The definition can be better understood pictorially. Figure 7 shows the extents for two different faults, f^1 and f^2. Faulty vehicle 1 exits the highway at time t_n^1, after a finite number of maneuvers, therefore its extent is a bounded set. Faulty vehicle 2, on the other hand, comes to a stop at time t_n^2 and therefore its extent is unbounded. It is apparent that it is very hard to unambiguously define the extent of the fault globally, as we have tried to do above. The reason is that the definition of the neighborhood is only relevant locally, as it depends on the protocol currently being executed. Even so the only cases that are of any interest from our point of view are the ones where the extents of two or more faults intersect. For this to happen the faulty vehicles need to be in close proximity to one another, in which case the local definition of neighborhood is adequate. In other words, the fact that the definition of fault extent is applicable only locally will not affect the subsequent arguments, as the cases we will be concerned with will deal with the local problem of intersecting extents.

The detailed protocols have already been verified automatically for the case of a single fault on the entire highway (Section 3.3). In this section we state and prove a sequence of claims about more general fault conditions. In particular, we show that the results proved for the single fault case also apply to the case of countably many faults with nonintersecting extent and we discuss extensions for the situation where multiple faults with intersecting extents occur. In the process we obtain some insight on the safety demands made on our design and we propose ways of refining these demands.

3.4.1 Multiple Noninteracting Faults

The definitions given in Section 3.2 as well as the results of the COSPAN verification allow us to draw certain conclusions about the performance of the system. We state these conclusions explicitly in a sequence of claims. The proofs of these claims will typically be a simple recapitulation of the automatic verification results in a more "conversational" manner.

All our claims are based on a set of underlying assumptions.

1. Fault detection and emergency maneuver initiation is taken care of by the extended degraded modes architecture. This is a reasonable assumption and will be fundamental in stating our claims, as the verification results presented here are concerned only with the execution of the emergency maneuvers and not their initiation. Extensive work has already been done on fault detection, while [Lygeros *et al.*1995] presents a possible emergency procedure initiation scheme.

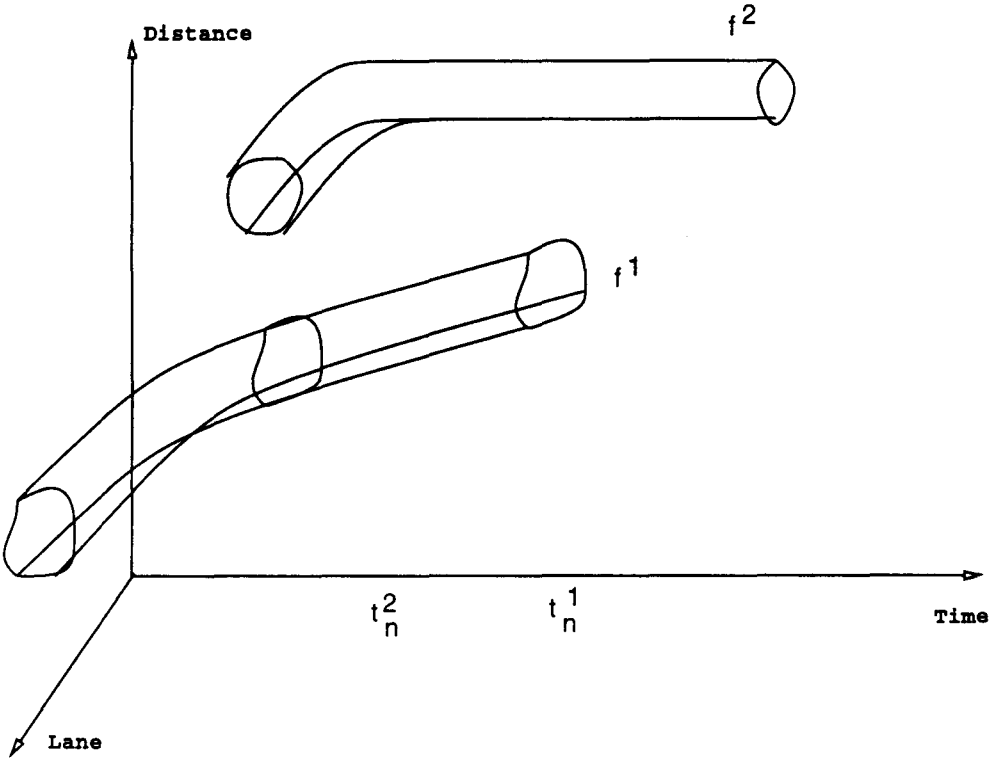

Figure 7: Extent of a Fault

2. The normal mode protocols have been properly designed and verified. This is a valid assumption based on the results of [Hsu *et al.*1994].

3. The regulation layer behavior is adequately summarized by the corresponding environment machine used in our verification. This is a strong assumption but we will not attempt to relax it at this stage, as equally strong theoretical results in hybrid systems are needed before we can do it.

Finally, we need to make some additional assumptions in order to state and prove claims about the behavior of our design when multiple faults are present in close proximity to one another. We will make these assumptions explicit in Section 3.4.2.

Claim 1 *A single fault on the entire highway system results in the faulty vehicle either stopping or exiting the highway and the remaining vehicles returning to their normal operation.*

Proof: From the assumption 1 above, the fault detection and degraded mode strategy initiation works properly. From the automatic verification results presented in previous section, we conclude that if the faulty vehicle starts a particular degraded mode strategy, then it will complete the strategy (i.e., the protocols for every atomic and compound maneuver will terminate successfully). By keeping track of the label changes of the grid cells, we conclude that all the surrounding vehicles return to normal mode in the end.

To complete the proof we need to show that vehicles not included in the fault neighborhood are not affected. Two external (E) vehicles can obviously engage in any normal mode maneuver.

Similarly, after the initial phase (as mentioned in previous section), two I cells can also engage in a normal mode maneuver with each other, or with the E vehicles. If the I or E cells, on the boundary of the neighborhood, request a maneuver with the F or A cells, they will receive a *Busy* signal, which is a normal mode message. Therefore, by assumption 2 above (that normal mode protocols are designed properly), we conclude that the fault neighborhood appears as a single E cell to the neighboring external vehicles. □

The proof of Claim 1 gives rise to two immediate corollaries:

Corollary 1 *The interaction across the boundary of the fault neighborhood is always Normal, i.e. consistent with the normal mode protocols.*

Corollary 2 *The degraded mode at the coordination layer terminates after a finite number of maneuvers.*

Claim 2 *Consider a countable*[4] *collection of faults* f^j, $j = 1, \ldots$ *whose extends do not intersect, i.e.,*

$$\mathcal{E}^i \cap \mathcal{E}^j = \emptyset \quad \textit{for all} \quad i \neq j$$

Then all the faulty vehicles either stop or exit the highway and all other vehicles return to normal mode of operation.

Proof: The results of COSPAN verification can be interpreted as follows:

The faulty vehicle can complete its protocols under the following two cases; either the A cells are empty or the vehicles in them assist the faulty vehicle. As long as the extent of a fault does not intersect with that of any other fault, the faulty vehicle will *always* get assistance from the A cells. The extents of two faults may be close enough that an I cell for one fault will be an A cell for another. This is allowed because if the I vehicle (for fault f1) is involved with an assisting maneuver with another faulty vehicle (f2), it cannot be involved in any maneuver with an A or F vehicle for faulty vehicle f1 before the fault f1 takes place.[5] This is a result of the mutual exclusion property of the normal mode protocols.

Therefore, the protocols for f1 are in no way obstructed by such a fault f2. Thus, this claim is proved by combining the results from Claim 1, and Corollary 1 along with the verification results of previous section. □

As the above claim involves only pairwise faults, we can eliminate the ambiguity in global definition of the fault location $F^j(t)$ by anchoring the origin of the coordinate frame at one of the faulty cell locations.

3.4.2 Multiple Interacting Faults

It remains to determine the performance of our protocols in the case when multiple faults occur in close proximity to one another. The most significant difference between this case and the ones

[4] It should be noted that for a realistic, physical highway, countably many faults is effectively the same as finitely many faults.

[5] It is important to note that all the vehicles in the fault neighborhood that *can* receive an "Abort" message due to the fault are labeled as I cells. Thus, in particular, an I platoon may or may not be involved in a maneuver with the A or F cells at the occurrence of a fault

considered above is that we can no longer assume that the degraded mode maneuver has top priority and will never be preempted. The reason is that there may be more than one degraded maneuver going on at the same time and a vehicle may be needed as an assisting vehicle by two faulty vehicles. This implies two major changes in our scheme:

- We need to add some form of fault priority to determine which maneuver will be initiated first and when a maneuver should be aborted.
- Our control scheme will need to be extended to allow for on-line changes in the fault handling strategy, in case the original plan has to be aborted.

Depending on the severity, we assign following priorities to the faults, making use of the classification scheme of [Lygeros *et al.*1995]: faults in class A have the highest priority followed by faults in class B, C and D in that order. The priorities within each class are as follows: A2 and A3 have equal priority which is higher than that of A1. In class B, B2 has higher priority than all other subclasses, which have equal priority. Every subclass in class C has the same priority. A static map between fault class and strategy to be used was described in [Lygeros *et al.*1995] and is also shown in Figure 8. This static map along with the fault priorities also defines priority structure among the extended coordination layer strategies. In case of more than one faults on the same vehicle, the strategy corresponding to the fault with higher priority is followed. In general, if the neighborhoods for two faults intersect, then the maneuver corresponding to the lower priority fault is preempted by the higher priority one. If the low priority fault occurs later, then its maneuver requests are not entertained until the high priority maneuvers are completed. If two strategies have the same priority and their protocol neighborhoods intersect, then the strategy that gets initialized second must wait until the first strategy is completed. This means that a static map between faults and strategies will have to be modified as frequently certain strategies will get aborted by others. Therefore, we introduce the tower of successive strategies in Figure 8. For example, if a TIE for faulty vehicle f1 is aborted by the assisting vehicle, then car f1 should follow GS and so on.

We can show that under this extended scheme our protocols will again perform satisfactorily.

Claim 3 *For an arbitrary collection of faults f_j the degraded mode protocols will not deadlock*

Proof: In the case of one fault on each vehicle, multiple faults can have intersecting neighborhoods only if they share the same assisting (A) vehicles. As we move up the tower of alternative strategies, the protocol neighborhood tends to become smaller. For the topmost strategies (CS and Engine Stop) the protocol neighborhood consists of just the faulty vehicle itself. Therefore, the chosen protocols can always be completed.

In case of multiple faults on the same vehicle, the above proof still holds. The only problem is that this scheme may be infeasible in real life, as the design of strategies is based upon reduction in individual capabilities. Multiple faults on the same vehicle will result in loss of many capabilities of the vehicle which might make many of the strategies unusable. [Lygeros *et al.*1995] describes a systematic way of keeping track of the capabilities of the vehicle. Based on the remaining capability of the vehicle, the best possible strategy among the possible ones can be chosen. □

It should be noted that this claim is not very informative. For one thing the proposed control scheme is definitely not optimum from the point of view of performance degradation, as it may cause vehicles with relatively minor faults to stop on the highway. Minor changes in the

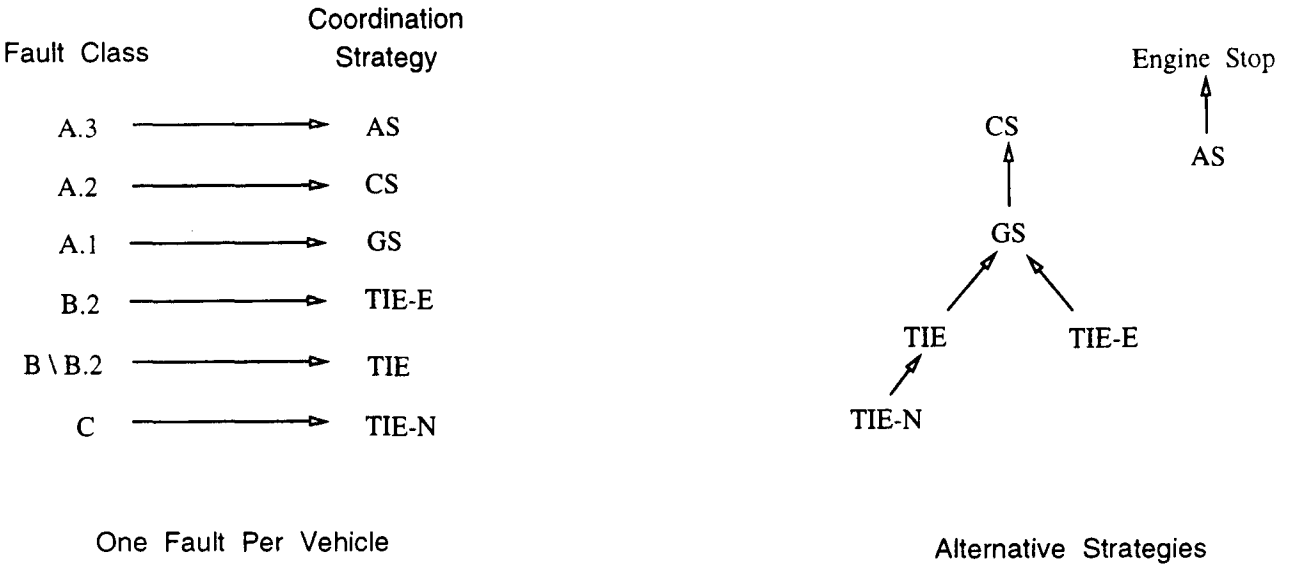

Figure 8: Fault Priorities and Alternate Strategies

scheme may lead to significant performance improvement. For example, if vehicle f1 following TIE-N gets an abort from the assisting vehicle because it is assisting another faulty vehicle f2 following TIE-E, then following the tower of Figure 8 this minor fault in f1 will give rise to a stopped vehicle in the middle of the road (note that the next higher strategy TIE is also of lower priority than TIE-E thereby vehicle f1 will follow GS). This would have been avoided if vehicle f1 continues its journey on the road and requests assistance for TIE-N after few seconds. The neighborhood of f1 can change due to factors such as difference in speeds across the lanes thereby possibly providing another assisting vehicle which can help get the vehicle f1 off the road on its own. But the performance of such a scheme is impossible to analyze discretely. Further tuning of this design, especially of the priorities and alternate strategies of Figure 8 should be performed after comparing the performance degradation of various schemes in AHS simulation such as SmartPath [Eskafi *et al.*1994]. Another factor that might lead to performance degradation is the aborting of normal mode maneuvers by faulty vehicles. This is unavoidable in case of many safety critical faults (those in classes A and B) as safety must take precedence over performance. But in case of minor faults (those in class C), the situation can be improved. For example if a TIE-N strategy requires aborting a join maneuver which is *very close* to completion and if the fault can be *safely* serviced after *some* time, then it may be a better choice to complete the join maneuver before starting TIE-E. There are two issues involved in such an optimization; first, we need to develop a decision algorithm that decides whether to abort a maneuver so as to minimize performance degradation. This will be based on some kind of simulation analysis. Secondly, this complicated algorithm must be proved to be safe. We intent to implement these control strategies on the AHS simulator SmartPath and tune the above priority scheme so as to optimize the performance while maintaining the safety of the design.

More importantly the above claim says nothing about the safety of the design. In fact not only is there no guarantee that the vehicles will not crash, but there may even be situations when it is impossible even to find a feasible action. We can show that:

Claim 4 *For any design there exists a combination of three faults that will lead to a catastrophe.*

Proof: We will provide two examples to prove the claim.

Example 1: Consider the following combination of three faults on a single vehicle. Brakes off, steering off, and communication off. With this combination, the three basic functions of a vehicle cannot be performed. If the vehicle cannot brake, steer and ask for help, it can be involved in a high speed collision.

Example 2: Consider two consecutive free agents moving in the same lane. If the front vehicle has a brakes on failure and the rear vehicle has a brakes off failure, then an accident is imminent.

The first example indicates that a catastrophe is possible with three faults, no matter what restrictions we impose on their distribution (any neighborhood definition will have to include at least the faulty vehicle). The second example indicates that in fact two faults between two vehicles are sufficient for a crash. □.

It is clear that the original safety objectives set for our design can not be met. We propose to refine them in one of two ways:

1. Use fault classification in terms of severity to restrict the allowable combinations of faults, for example "up to 3 faults in a neighborhood, not more than one from class A".
2. Introduce probabilities to fault occurrence, use the design to obtain probabilities of catastrophes and require that the probability of a catastrophe is smaller than a certain threshold.

The second approach seems preferable as it is less ad-hoc, it allows us to compare different designs by a common norm and it captures the idea that the hardware design should be such that severe faults should be less likely. Such a probabilistic analysis was done by means of monte carlo simulations by Hitchcock [Hitchcock1994]. We would like to use analytical methods as much as possible, and use simulation to verify our predictions. It should also be noted that the above analysis does not take into account redundant hardware/equipment used for obtaining the same functional capability. Thus the "faults" described here are to be treated as "loss of functionality". Redundancy considerations should also help in deriving a more realistic specification.

4 Conclusions and Future Directions

We have presented the design and verification of inter-vehicle communication protocols for degraded modes of operation on the Automated Highway System. We considered various hardware and sensor faults that can develop on the automated vehicle in an AHS and designed discrete event supervisory controllers to stop the faulty vehicle or take it out of the highway in a safe manner. The protocols were verified for logical correctness by using automatic formal verification tools. This work presents an important step towards analyzing the safety of the AHS.

The fault tolerant controller design presented by us improves robustness of automatic vehicle control system to handle adverse environmental conditions and vehicle faults on the highway. We formally verified that under certain assumptions on environment, regulation and physical layer responses, our design does not deadlock and either brings the faulty vehicle to a stop or takes it to the exit safely.

We showed that the original safety specification proposed in [Hitchcock1994], that up to three faults in a neighborhood do not lead to a catastrophe, cannot be met. We proposed a refinement of this specification. We have also implemented our scheme on the highway simulator SmartPath [Eskafi *et al.*1994] and are using it to get performance estimates.

It should be noted that the control of automated highways involves both continuous parameter and discrete parameter dynamics and it must be addressed in the hybrid systems framework. This is a topic for future research.

Acknowledgment: The authors would like to thank Mireille Broucke, Jonathan Frankel, John Haddon, Roberto Horowitz, Perry Li, Antonia Lindsey, Anuj Puri, Shankar Sastry and Pravin Varaiya for helpful discussions providing insight into this problem.

References

[Eskafi *et al.*1994] Farokh Eskafi, Delnaz Khorramabadi, and Pravin Varaiya. SmartPath: An automated highway system simulator. PATH Technical Report UCB-ITS-94-4. Institute of Transportation Studies. University of California, Berkeley, 1994.

[Godbole *et al.*1994] Datta N. Godbole, John Lygeros, and Shankar Sastry. Hierarchical hybrid control: An IVHS case study. In *IEEE Control and Decision Conference*, pages 1592–1597, 1994.

[Godbole *et al.*1995] Datta N. Godbole, Farokh Eskafi, Ekta Singh, and Pravin Varaiya. Design of an entry and exit maneuvers for AHS. In *American Control Conference*, pages 3576–3580, 1995.

[Har'El and Kurshan1987] Z. Har'El and R.P. Kurshan. *Cospan User's Guide*. AT&T Bell Laboratories, 1987.

[Hitchcock1994] A. Hitchcock. A specification of an automated freeway with vehicle-borne intelligence. PATH Technical Report UCB-ITS-PRR-92-18, Institute of Transportation Studies, University of California, Berkeley, 1994.

[Hsu *et al.*1994] Ann Hsu, Farokh Eskafi, Sonia Sachs, and Pravin Varaiya. Protocol design for an automated highway system. *Discrete Event Dynamic Systems*, 2(1):183–206, 1994.

[Lygeros and Godbole1994] John Lygeros and Datta N. Godbole. An interface between continuous and discrete-event controllers for vehicle automation. In *American Control Conference*, pages 801–805, 1994.

[Lygeros *et al.*1995] John Lygeros, Datta N. Godbole, and Mireille E. Broucke. Design of an extended architecture for degraded modes of operation of IVHS. In *American Control Conference*, pages 3592–3596, 1995.

[Varaiya1993] Pravin Varaiya. Smart cars on smart roads: problems of control. *IEEE Transactions on Automatic Control*, AC-38(2):195–207, 1993.

A Coordination Layer Protocols

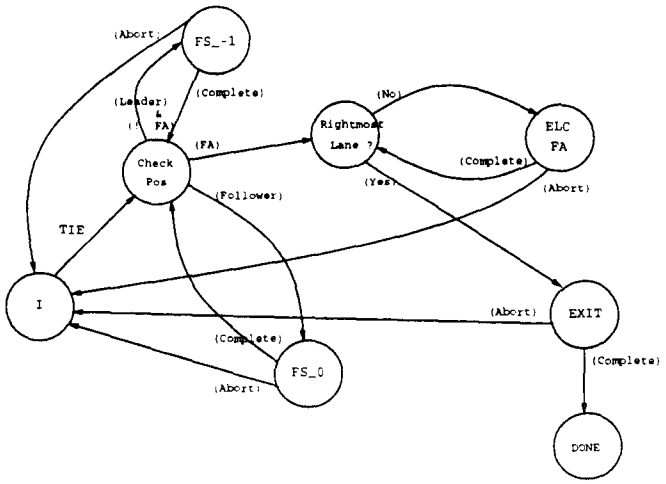

Figure 9: State Machine for Take Immediate Exit

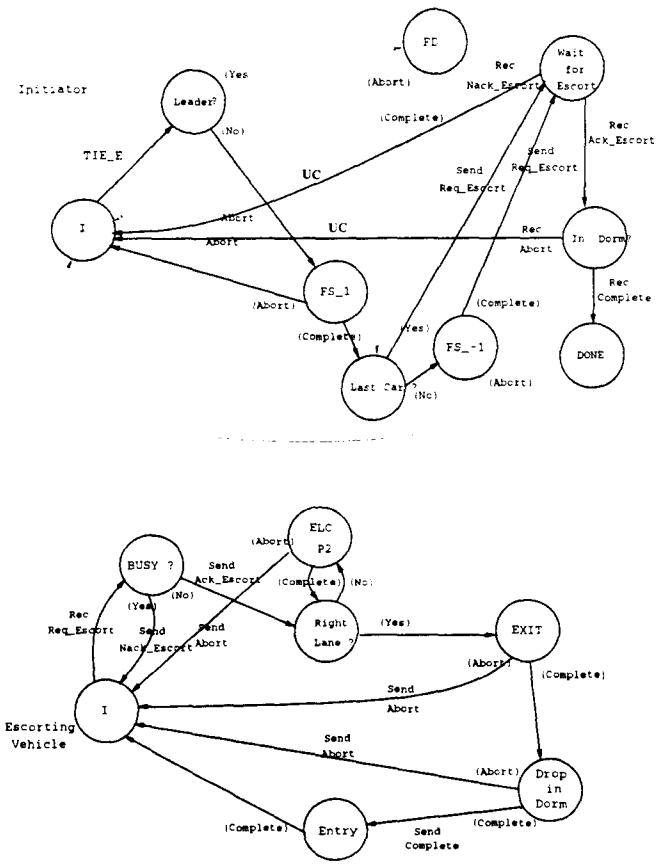

Figure 10: State Machine for Take Immediate Exit - Escorted

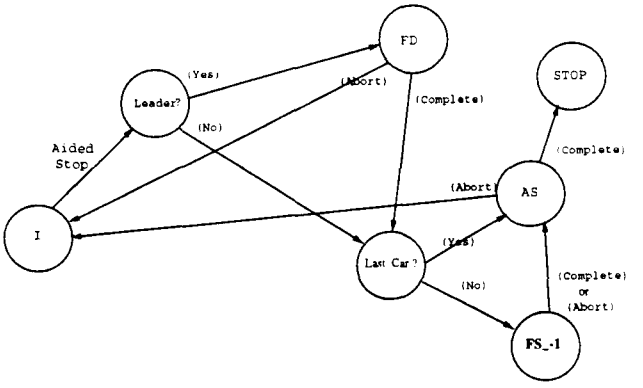

Figure 11: State Machine for Aided Stop

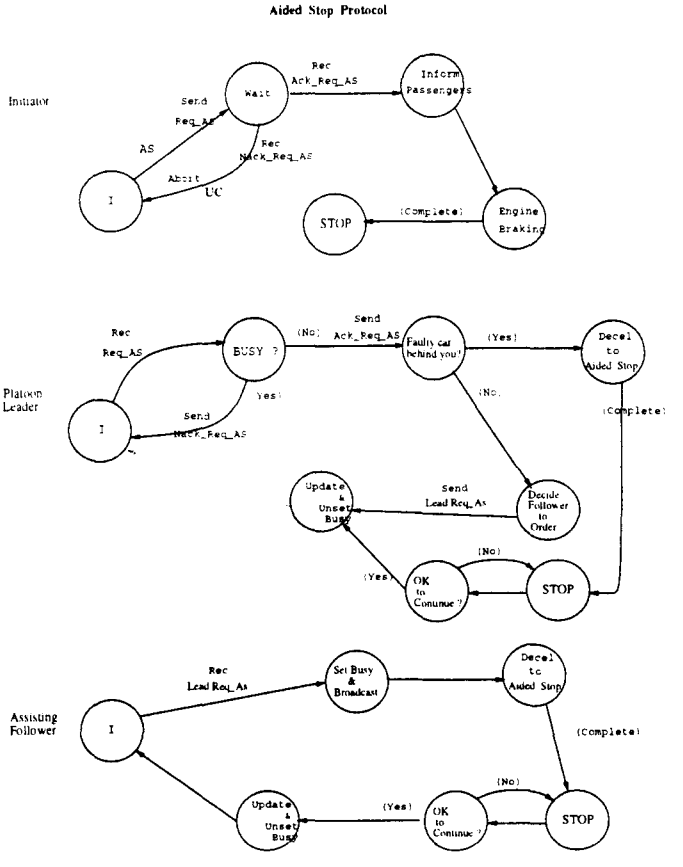

Figure 12: Aided Stop atomic maneuver protocol

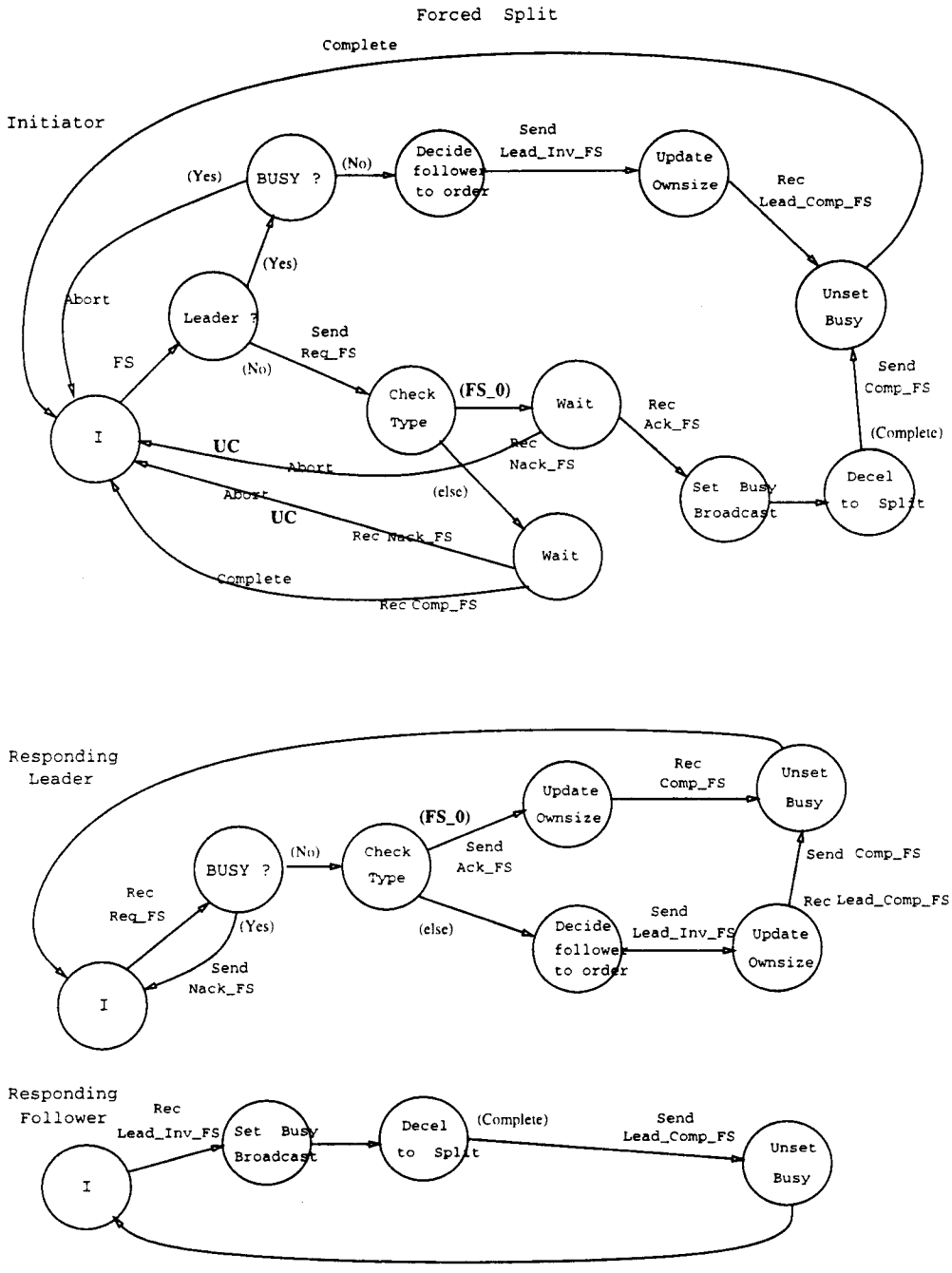

Figure 13: Forced Split maneuver protocol

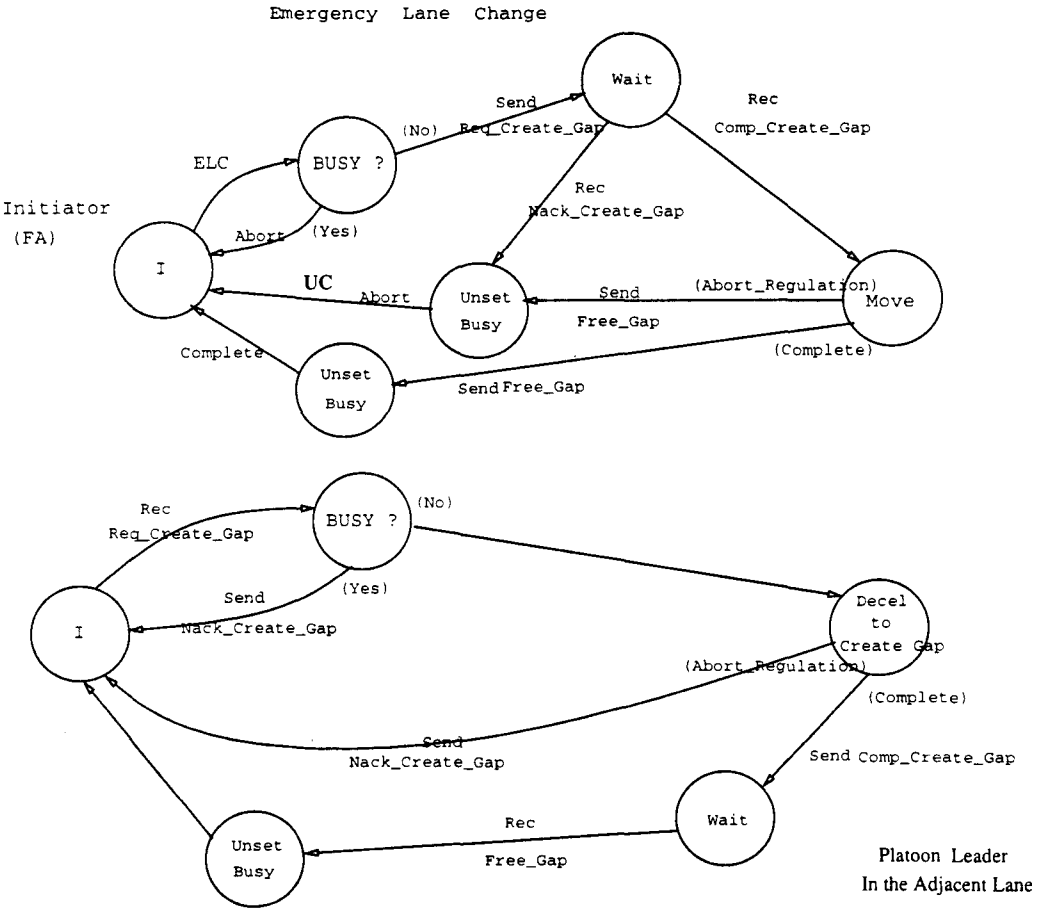

Figure 14: Emergency Lane Change maneuver protocol

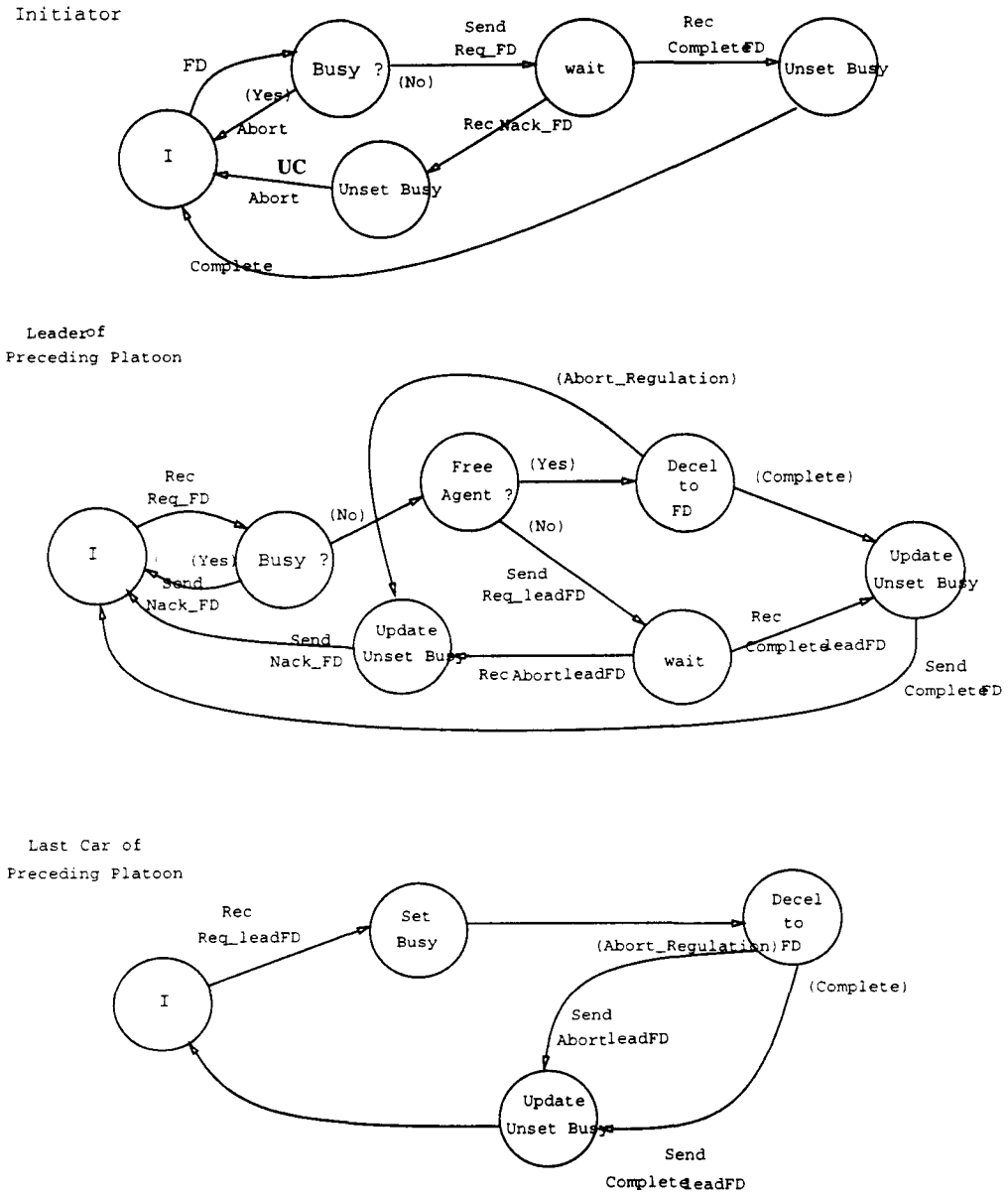

Figure 15: Front Dock maneuver protocol

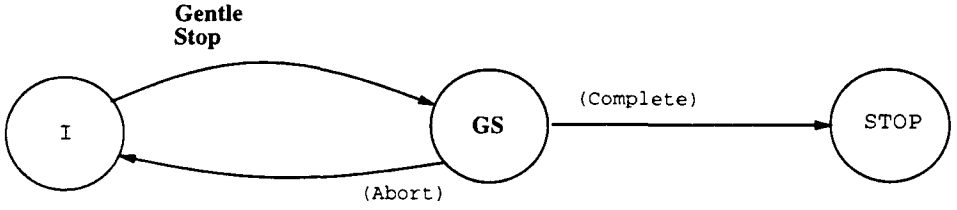

Figure 16: Gentle Stop Strategy

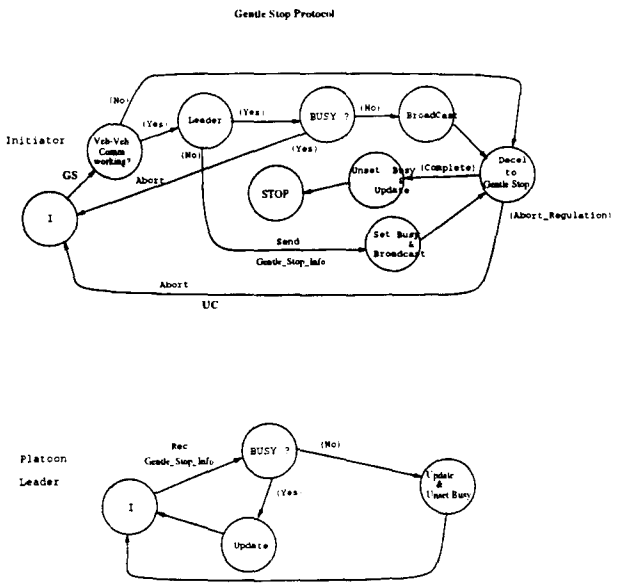

Figure 17: Gentle Stop maneuver protocol

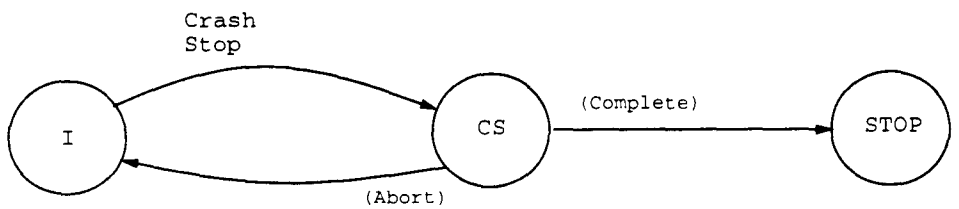

Figure 18: Crash Stop Strategy

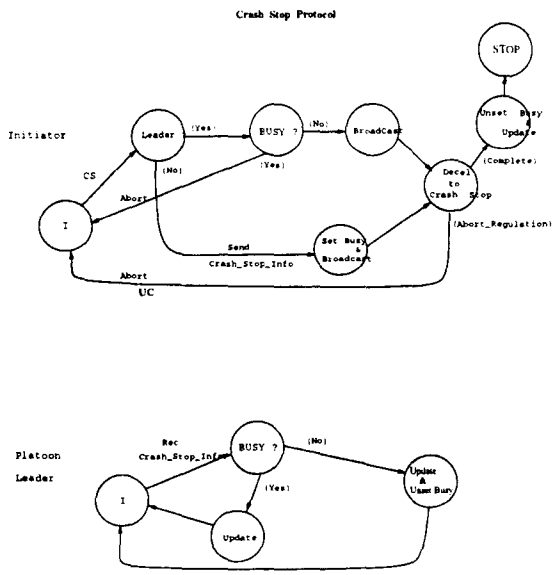

Figure 19: Crash Stop maneuver protocol

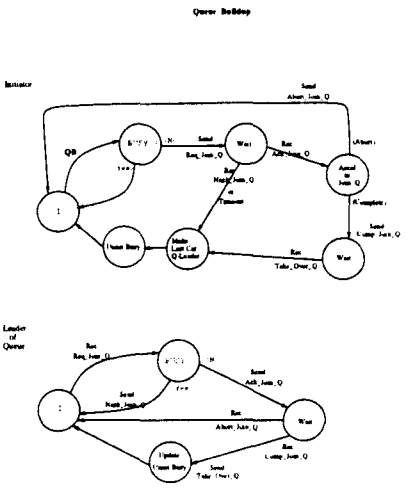

Figure 20: Queue Buildup

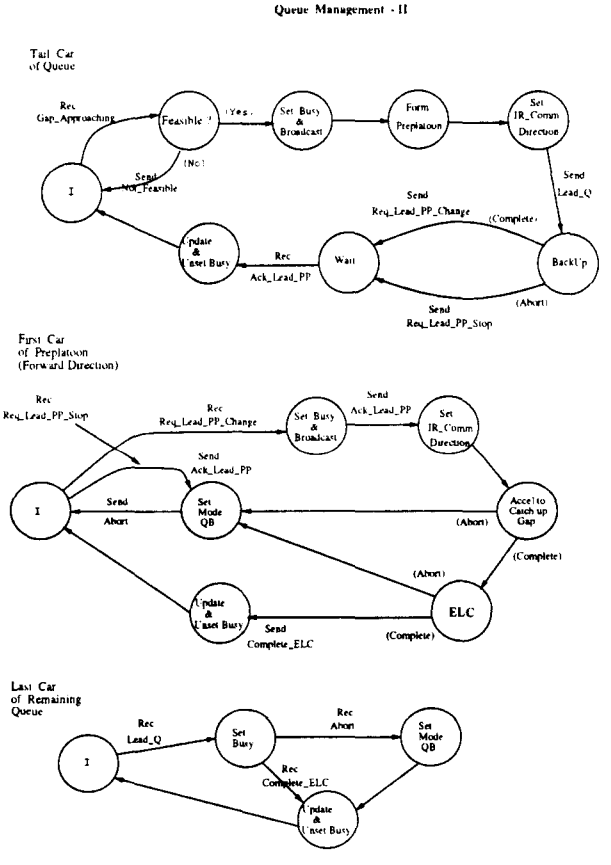

Figure 21: Queue Dissipation using Backup and Catchup maneuvers

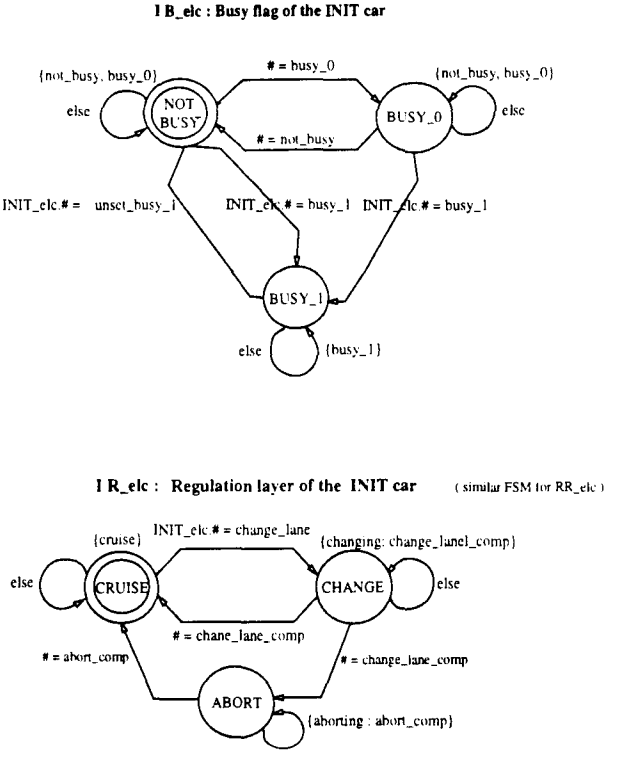

Figure 22: Emergency Lane Change Environment Processes

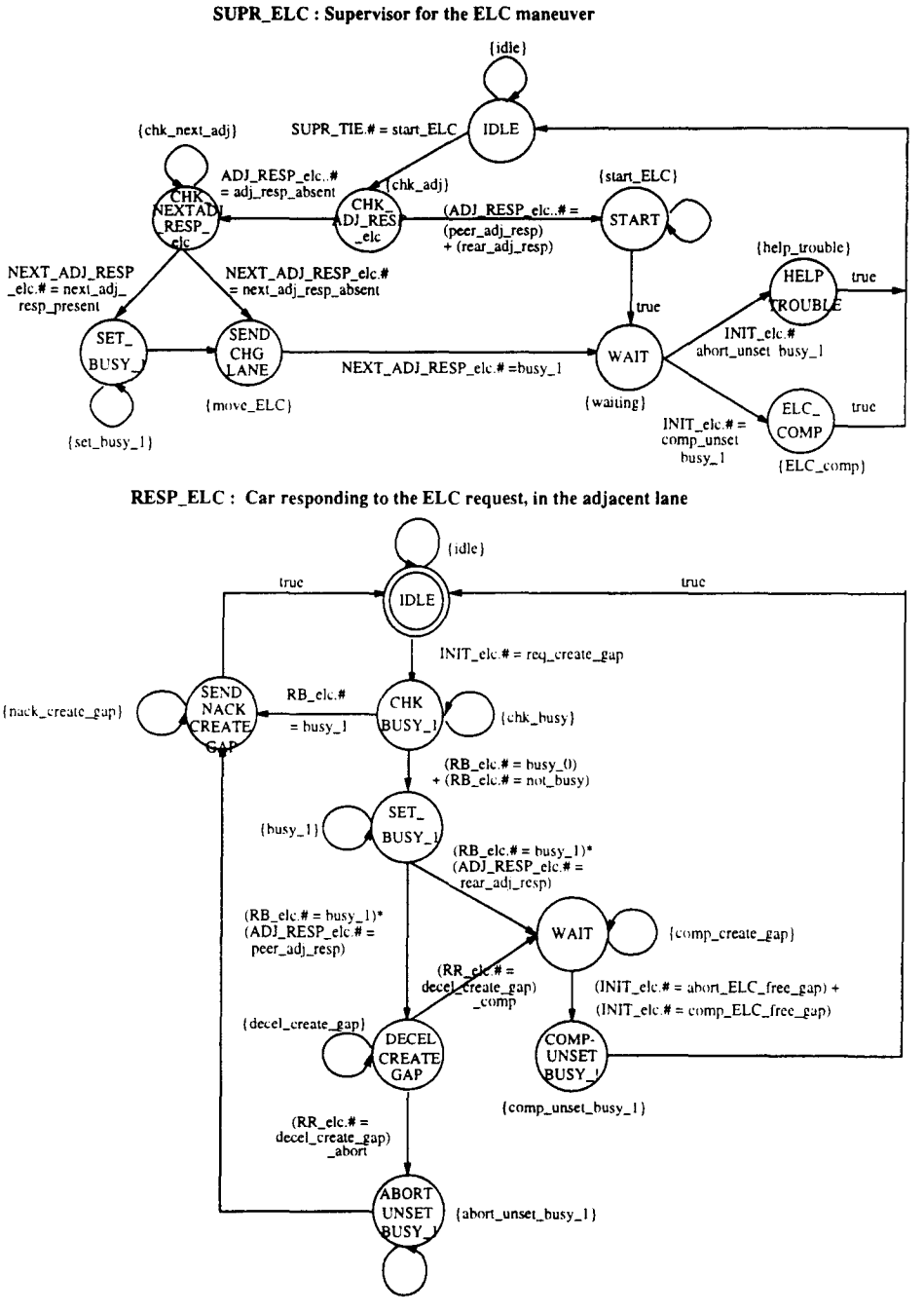

Figure 23: Emergency Lane Change Protocol Processes

INIT_ELC : Initiator of the Emergency_Lane_Change maneuver

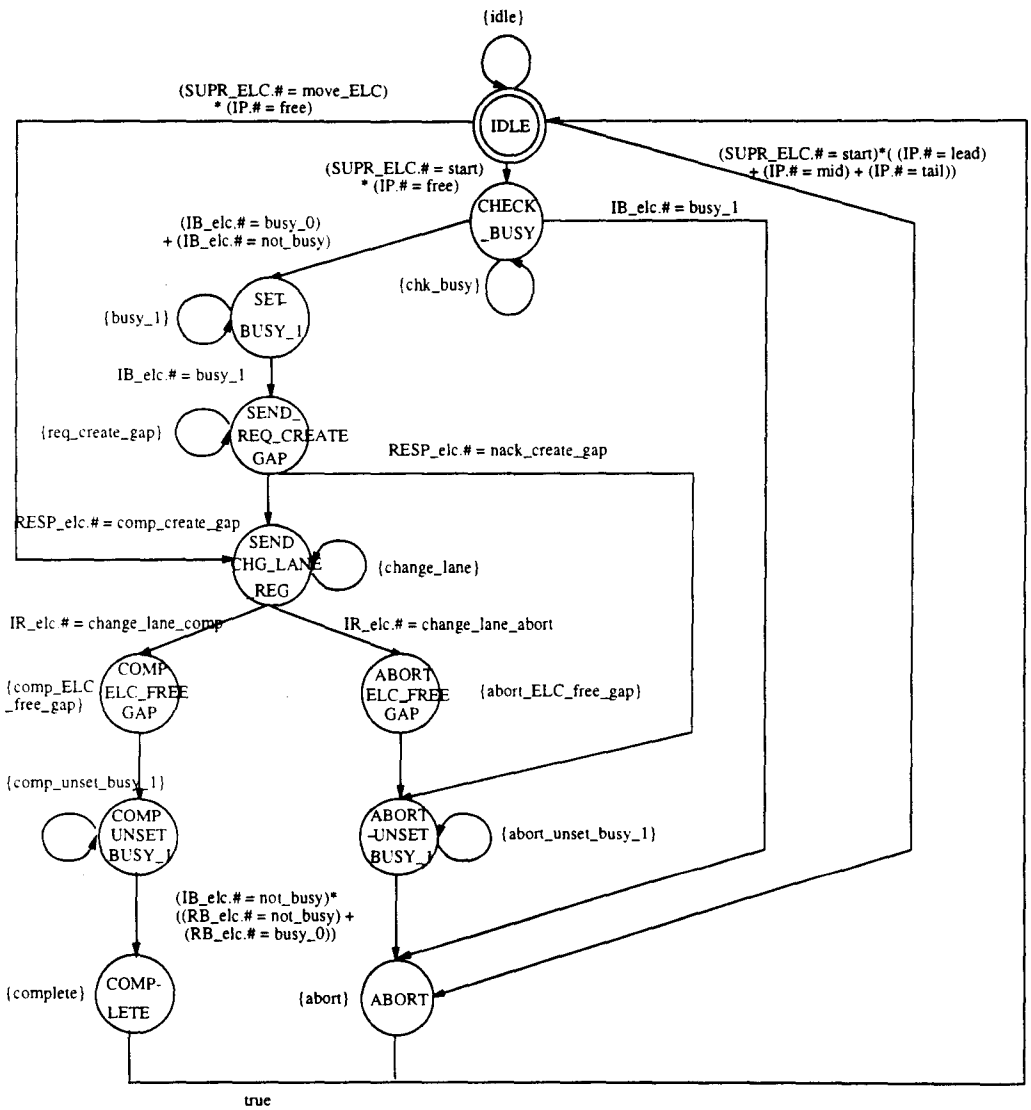

Figure 24: Emergency Lane Change Protocol Processes

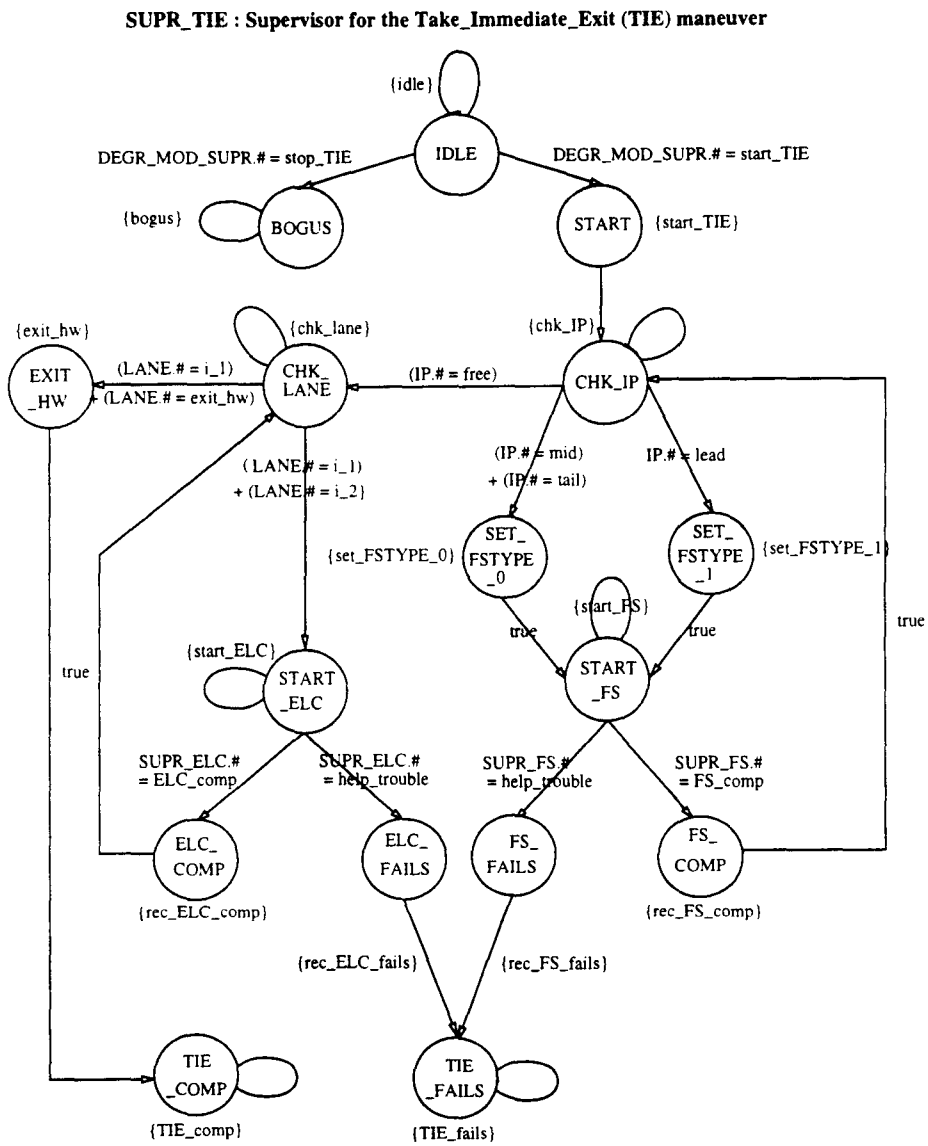

Figure 25: Take Immediate Exit Supervisor Processes

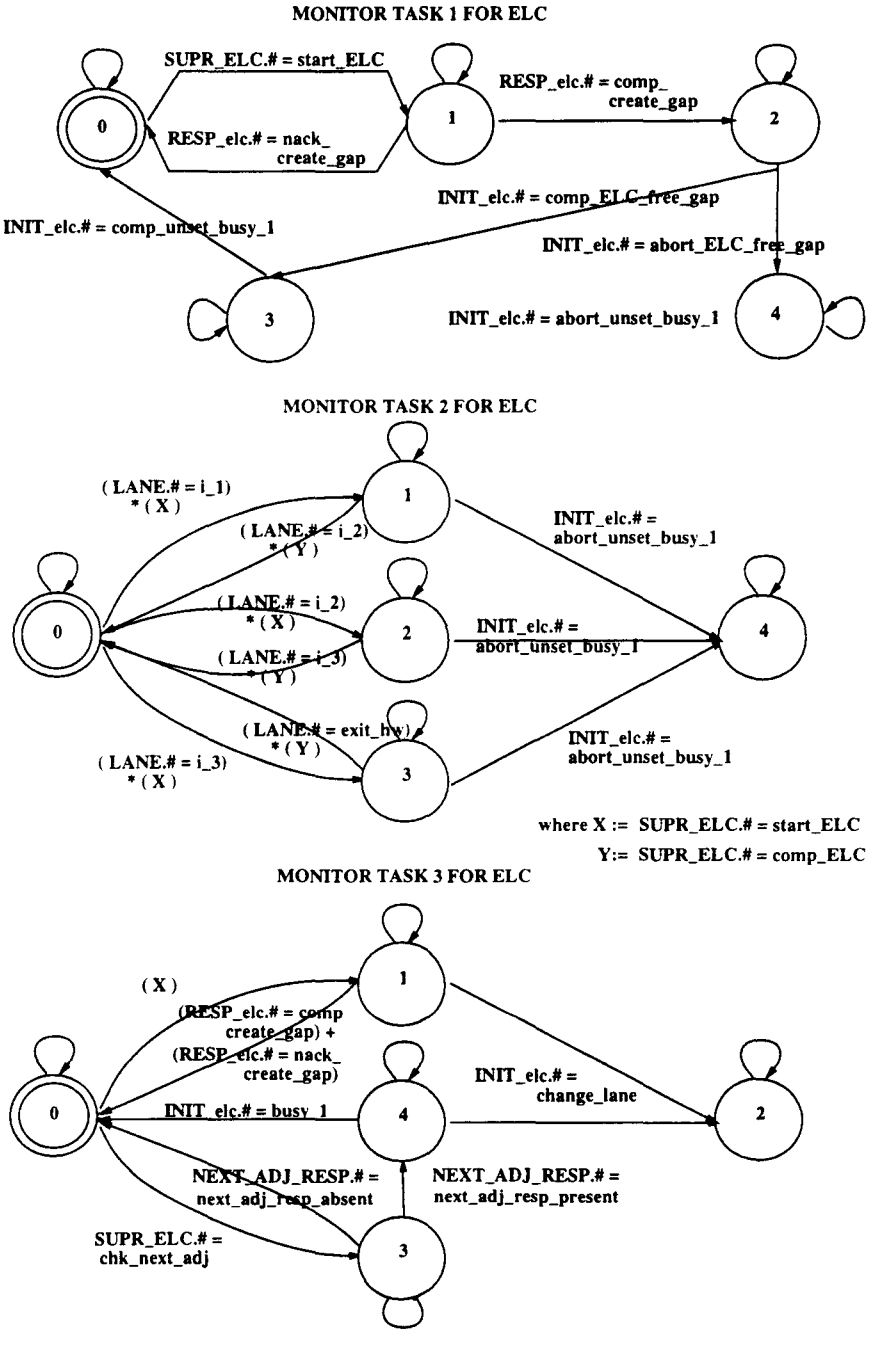

Figure 26: Monitors For Emergency Lane Change Protocol