Formal Language Methods Based Modeling for Traffic Incident Management

Neveen Shlayan (corresponding author)
Transportation Research Center
University of Nevada, Las Vegas
4054 Maryland Pkwy
Las Vegas, Nevada 89104
shlayann@unlv.nevada.edu
Phone: 702-524-5129

Pushkin Kachroo, PhD
Transportation Research Center
University of Nevada, Las Vegas
4054 Maryland Pkwy
Las Vegas, Nevada 89104
pushkin@unlv.edu
Phone: 702-895-5258

Adrain Reed Gibby, PhD
Nevada DOT
1263 South Stewart Street
Carson City, NV 89712
agibby@dot.state.nv.us
Phone: 775-888-7192

Fred Ohene, PhD
Regional Transportation Commission of Southern Nevada
600 S. Grand Central Pkwy.
Las Vegas, NV 89106
ohenef@rtcsnv.com
Phone: 702-676-1500

4513 words + 6 figures + 2 tables

July 31, 2009
Abstract

Traffic Incident Management is a multi-jurisdictional process. Complications with communication, compatibility, coordination, institutional responsibilities and legal issues are inherent in a traffic incident management system. Increasing delay in incident clearance due to various conflicts has vital economical, safety, environmental and social impacts. Therefore, a thorough and rigorous modeling of the system is necessary to better understand its properties and systematically solve issues that might arise. This paper proposes the use of formal language theory for modeling, analyzing, and implementing the traffic incident management process. This theory has been used very effectively for hardware and software systems. Using formal languages, allows us to perform debugging on a traffic incident management system covering all possibilities for inefficiencies and problems for which we can find solutions. This paper demonstrates how to use formal language methodology to model the traffic incident management system through a case study in the Las Vegas area.
INTRODUCTION
Traffic incidents are non-recurring events in traffic flow operations that often cause delay due to congestion and safety hazards (8). These incidents account for approximately one third of all delay on the US highways caused by traffic congestion and are responsible for nearly 60 percent of delay triggered by weather, construction, and special events (13). The operating capacity of a typical highway is reduced by 63 and 77 percent during one lane and two lane obstructions, respectively, on a three lane freeway segment (3). Incidents, such as a disabled passenger car parked on the shoulder of the roadway reduce the available capacity by up to 17 percent (3). In addition, crash reduction or avoidance can be significant as illustrated by an evaluation conducted by Minnesota Department of Transportation. This evaluation in 2004 reported that 68 percent of the monetary benefits of a traffic incident management program was a reduction in crashes. The impact of traffic incidents stretches beyond safety degradation and traffic congestion. Human productivity loss and fuel waste are definite economical outcomes (17). In 2005, congestion costs were estimated to be $78.2 billion in 437 U.S. urban areas where 52 to 58 percent of the total motorist delay is due to traffic incidents (3).

When an incident occurs, various public agencies, such as ones related to medical, law enforcement, fire, and other public emergency agencies are usually among the first to respond. In addition, private agencies such as towing companies and hazardous materials contractors are most likely to be involved (13). On one hand, the existence of specialized entities delivers high quality work in handling tasks at the incident scene. On the other hand, this also raises challenges since each of these agencies has uniquely different priorities.
and views (13). Moreover, every agency has a separate communication system through which dispatchers communicate information about the incident to their agents. Such independency in communications leads to further misunderstanding among the various agents present at the scene leading to even more delays in the incident clearance process. Carson, et. al. (4), conducted a comprehensive evaluation of an incident response team program for the Washington State Department of Transportation determining its effectiveness. The study claims a 20.6 minute reduction in average duration of incidents from 1994 to 1995 resulting in $20,600 to $61,800 savings per incident (4). Therefore, an organized traffic incident management process promoting integration and bonding of multiagency operations and communications at the incident scene is necessary. A well planned incident management system, through formal and informal processes, improves efficiency and coordination between the multi-jurisdictional responses reducing incident clearance times and vehicle delays (17). However, such coordination is faced with many obstacles that are inherent in the system, such as uncertainty, sudden events, resource shortage, faulty information, and disruption of infrastructure support (5). In addition to support systems, incident management is currently formulated and implemented conventionally based on manual methods relying on personal experiences of the personnel from within the incident management field which has its shortcomings (7). The current conventional approach does not provide timely information on traffic networks, and it does not allow for conflict detection, or alternative incident management scenario evaluation due to time constraints (7). Furthermore, personal experiences are likely limited; they also vary from person to person’s experiences which is the main factor in the decision making process. This often leads to further conflicts and difficulties and, thus adversely impacts the traffic operations.

In this paper, using formal languages methodology for modeling, analyzing, and implementing a unique Incident Management process is proposed. Using formal language theory, allows us to perform rigorous debugging on existing and future incident management systems covering all possibilities for inefficiencies and problems for which we can find solutions. The modeling approach introduced herein provides the flexibility to reflect on any Incident Management process depending on various variables involved for a certain urban region. Through formal methods modeling, customized software tools can be developed for a specific region enhancing the Incident Management process significantly. In section 2, a literature review on previous work for Incident Management modeling will be discussed. In section 3, the IM process in the Las Vegas area is described. An overview of the used modeling method and approach are presented in section 4. A demonstration on how formal methods are used to model the IM process through a case study is presented in section 5 followed by further analysis in section 6. Then, conclusions are in section 7.

LITERATURE REVIEW

Effective traffic incident management systems consist of three main aspects, multiagency communications and control, decision making, and sharing of limited resources. For a model to be successful these three aspects have to be addressed, otherwise complications in the incident management system may be overlooked. In this section, some proposed approaches to improve Incident Management processes are discussed.

Incident prediction models based on analysis of incident patterns, frequency, and duration were proposed by Konduri, et. al. (10) and used for improving the freeway management system by assessing various IM strategies. Such models would be very useful in incident management systems; however, methods based on static data are not sufficient to comply with the required short term actions necessary in the most effective incident management systems (18).

An agent based approach for monitoring, analyzing, and supporting Incident Management processes by error detection and providing support for such errors is proposed in (6). Temporal Trace Language (TTL) was used as a tool for formal representations of system’s properties. The author’s approach is adequate; however, the scope of this modeling is error detection for improving techniques in current incident management support systems that detect contradictory and unreliable information. This approach does not address broader issues in incident management such as the overall interaction and harmony between the involved
agencies, limited and shared resources, or liveness properties of the incident management system as a whole.

Mingwei in (7) proposes a real-time evaluation and decision support system (REDSS) for IM which detects traffic incidents, estimates impacts of incidents, formulates guidance scenarios, and monitors and evaluates scenario implementations. REDSS integrates a series of information analysis and processing technologies such as data fusion, expert systems, data warehousing and data mining (7). However, REDSS has not been validated.

Chen in (5), the author recognizes the constraints on responder’s capabilities to analyze coordination problems due to the requirement of rapidness in decision making. Therefore, a life cycle approach is introduced providing a broad and systematic view of activities relating to emergency response management.

An inter-vehicular communication system design is proposed in (16). This system provides the ability to quickly discover and transmit real time multimedia information from an incident location to the approaching first responders (16). Kim in (9) introduces a conceptual model explaining the efficiency of decision-making of the Critical Incident Management Systems (CIMS).

Researchers have demonstrated numerous attempts in improving the incident management process (7); however, in the history of IM, such attempts have been mainly focused on supporting systems. Such systems are mainly used to assist participating agency in assigning tasks and making decisions. Clearly, these systems do not integrate the three aspects necessary for a successful incident management modeling as well as implementation of strategies. A successful IM necessitate a broad and integrated response to incidents (14).

In this paper, an incident management representation is proposed which provides the ability to account for all three aspects of IM integrating them into one systematic model. Moreover, specifying properties for the system and verifying them before implementation. The proposed model provides the ability to validate and verify any existing incident management process including supporting systems; most importantly, it provides a method to verify the interaction between such systems as well as multiagency processes.

INCIDENT MANAGEMENT IN THE LAS VEGAS AREA

Until recently, Las Vegas (map depicted in Figure 1) has been one of the fastest growing cities in the United States. Consequently, highway capacity investment has not been able to keep pace with the growth in traffic; therefore, major roadways are experiencing substantial congestion during off-peak periods as well as peak periods. Users cost per hour for a closure on I-15 was recently estimated at $240,000 and can go up to $750,000 during the afternoon peak period (8). The report produced by Iteris (1) identified the existing institutional relationships which include operational agreements between various agencies for the Las Vegas area. Furthermore, it showed responsibilities of various organizations during an incident management process. Emergency responders in Las Vegas include but are not limited to the following agencies: Department of Safety - Nevada Highway Patrol (NHP), Las Vegas Metropolitan Police Department (LVMPD), Regional Transportation Commission of Southern Nevada (RTC), Freeway Arterial Transportation System (FAST), Clark County Office of Emergency Management and Homeland Security, Clark County Fire Department (CCFD), and Coroner’s Office.

A local traffic incident management (TIM) Coalition has been formed where various emergency responder agencies meet and discuss regional issues involving traffic incidents in the hopes of resolving any boundaries resulting in an improved communications, enhanced coordination, and an efficient incident management process in the Las Vegas area (1). FAST center operates the freeway and arterial traffic signal systems. FAST also supports incident management through traffic control (1). According to the Incident Management Strategies Draft Report (8), incident management is the key motivation for the existence of FAST in Las Vegas. Specifically, FAST provides data and tools to identify incidents and assists with remote monitoring of the incidents.

Las Vegas has witnessed drastic improvement in the incident management process as a result of FAST efforts in detecting and monitoring incident occurrences, and the TIM team attempts to resolve any
miscommunication issues between local agencies. However, crash data, presented in tables 1(a) and 1(b), from 2003 through 2008 obtained from the Computer Aided Dispatch (CAD) Center of LVMPD and a year worth of data from NHP were analyzed; as depicted in Figures 2(a) and 2(b). It was found that the average management and clearance times of incidents need improvement. Thus, a systematic solution through qualitative modeling is necessary for finding all possible sources of inefficiencies.

**MODEL APPROACH**

The purpose of formal languages theory is to bring order to complex system anarchy (15). Formal languages are characterized by predefined rules such as formal notations in mathematics, logic, and computer science (2, 15). A finite automaton is a string processor that assists in defining certain formal languages by accepting or rejecting a sequence of symbols (15); Applications that require pattern recognition techniques have fundamental interest in finite automata (2). A deterministic finite automaton consists of a finite number of states or conditions in which a system can exist and only one of these states can be an “initial” state. Additionally,
TABLE 1 Average Arrival, Management, and Clearance times for incidents that occurred on the I15 and arterials in the Las Vegas area

**(a) LVMPD data**

<table>
<thead>
<tr>
<th>Year</th>
<th>AVG Arrival Time</th>
<th>AVG Management</th>
<th>AVG Clearance Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2003</td>
<td>0:18:04</td>
<td>1:05:16</td>
<td>1:23:29</td>
</tr>
<tr>
<td>2005</td>
<td>0:25:08</td>
<td>1:10:29</td>
<td>1:35:37</td>
</tr>
<tr>
<td>2006</td>
<td>0:25:09</td>
<td>1:13:12</td>
<td>1:38:21</td>
</tr>
<tr>
<td>2008</td>
<td>0:19:47</td>
<td>1:43:21</td>
<td>1:46:00</td>
</tr>
</tbody>
</table>

**(b) NHP data**

<table>
<thead>
<tr>
<th>Month</th>
<th>AVG Arrival Time</th>
<th>AVG Management</th>
<th>AVG Clearance Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jul-08</td>
<td>0:11:30</td>
<td>1:12:04</td>
<td>1:30:01</td>
</tr>
<tr>
<td>Aug-08</td>
<td>0:11:10</td>
<td>1:13:22</td>
<td>1:30:27</td>
</tr>
<tr>
<td>Sep-08</td>
<td>0:11:06</td>
<td>1:13:18</td>
<td>1:31:35</td>
</tr>
<tr>
<td>Oct-08</td>
<td>0:10:53</td>
<td>1:16:39</td>
<td>1:33:47</td>
</tr>
<tr>
<td>Nov-08</td>
<td>0:10:43</td>
<td>1:08:20</td>
<td>1:25:59</td>
</tr>
<tr>
<td>Dec-08</td>
<td>0:13:21</td>
<td>1:10:58</td>
<td>1:31:33</td>
</tr>
<tr>
<td>Jan-09</td>
<td>0:11:22</td>
<td>1:13:12</td>
<td>1:28:49</td>
</tr>
<tr>
<td>Feb-09</td>
<td>0:12:12</td>
<td>1:08:40</td>
<td>1:27:29</td>
</tr>
<tr>
<td>Mar-09</td>
<td>0:11:42</td>
<td>1:07:50</td>
<td>1:24:54</td>
</tr>
<tr>
<td>Apr-09</td>
<td>0:12:38</td>
<td>1:08:42</td>
<td>1:27:07</td>
</tr>
<tr>
<td>May-09</td>
<td>0:11:22</td>
<td>1:07:06</td>
<td>1:23:33</td>
</tr>
<tr>
<td>Jun-09</td>
<td>0:11:52</td>
<td>1:07:18</td>
<td>1:26:40</td>
</tr>
</tbody>
</table>

such an automaton must contain at least one or more “terminal” or “accepting” states. Transitioning may be performed through two different actions, either switching to another state or remaining in the current state (2). Execution of state transition depends on the current state and the action or inaction identified by a symbol.

Using finite automata, a simple example of an incident may be modeled in a pictorial form called a state diagram, as depicted in Figure 3. Glossary for FSP actions is introduced before the introduction. State “$S_0$” represents a pre-accident situation which might imply traffic is in a free-flow state. The symbol “accident” represents an occurrence of an incident which causes the system to switch to state “$S_1$” implying an incident scene. Once the system is in state “$S_1$” only two transitions are possible represented by the symbols “call_911” and “call_failed”; first of which causes the system to switch to state “$S_2$” implying that the incident is in the management process. The second symbol causes the system to remain in the same state implying that no advancements can be made unless an emergency responder is informed. Once the system reaches the management state, it can switch states when congestion is cleared (“congtn_clrd”) going back to free traffic flow in pre accident conditions (state “$S_0$”). It remains in the management state “$S_2$” if congestion is not cleared (“stillcongested”).

Modeling the evolution of an incident scene using finite automata is methodically appropriate in terms of a sequence of events. Furthermore, many transitioning possibilities can be considered, depending on various conditions, which adds flexibility in modeling any Incident Management system. Every Incident Management process, however, is an interaction between multiple processes occurring concurrently. Thus, concurrency is an aspect that must be addressed in the Incident Management model.
Shared actions in Labeled Transition Systems (LTS) provide the ability to model concurrent finite state machine processes. They are described textually as finite state processes and displayed and analyzed by the LTS analysis tool (11). The LTS analysis tool provides the possibility to structure complex systems as sets of simple activities represented as sequential processes using Finite State Processes (FSP) (11). Processes can overlap or run concurrently reflecting real world situations as in the Incident Management process.

Finite state processes (FSP) have a predefined language for their description; actions can be described using the action operator “→”. For instance, \((x → P)\) describes a process that initially engages in the action “x” and then behaves as described by process P (11). In order to model choice, the choice operator
“|” is used, for instance \((x \rightarrow P|y \rightarrow Q)\) describes a process that may engage in either action “x”, which leads the system to behave as described by process P, or action “y” leading the system to behave as described by process Q (II). Figure 1 demonstrates an FSP process illustrating the incident model described in Figure 3.

Program 1 An FSP model for an incident occurrence process

PRE_ACCIDENT = (accident->ACCIDENT),
ACCIDENT = (call_911->CLRNSinPROCESS|callfailed->ACCIDENT),
CLRNSinPROCESS = (congtn_clrd->PRE_ACCIDENT|stillcongested->CLRNSinPROCESS).

Concurrency can be modeled by using the parallel operator “||”. For example, \((P||Q)\) represents the concurrent execution of the processes P and Q (II). Parallel processes have the capability to interact via shared actions which are executed at the same time by all participant processes (II).

Since FSP provides the ability to modeling parallel processes as well as methods to interacting them, the incident model can be expanded to include an emergency response agency where the occurrence of an incident and the agency’s operation are running in parallel and have a shared action “call_911”. Figure 4 illustrates the model of LVMPD based on the actual agency in Las Vegas.

As demonstrated in Figure 4 there exists within LVMPD several processes that are executed in parallel and interacting through shared actions. LVMPD has three main separate entities that function concurrently, call takers, dispatchers, and officers. When an accident occurs and 911 is dialed “call911” a 911-operator (CALLTAKER) from LVMPD will answer the call and has three options for transferring the call to either police, fire, or medical agency. The call will be transferred to the requested agency. Freeway incidents are under NHP’s jurisdiction. Therefore, the call will be transferred to NHP if incident location is freeway. If police is requested, LVMPD dispatch will receive the call, accomplished by the shared action “transf police”, then, it will acquire information about the incident, verify redundancy of call, send an officer to the scene (which immediately starts the process of the officer through the shared action “officer order”), contact fire, medical, or both depending on the severity of the incident, then goes back to initial state making dispatch available to accept new calls. As 911-operators and dispatchers are available to receive new calls, an officer driving to the incident scene, could be faced with various circumstances such as traffic congestion, blocked streets, faulty information about the actual incident severity. These are all examples of possible scenarios which can be considered in the model. The officer’s task is not done until the system returns to normal conditions. The following acknowledgement is achieved via the shared action “congtnclrd” between the LVMPD model and the incident model described in Figure 3. Figures 4(a), 4(b), and 4(c) demonstrate the state diagram for the LVMPD model in Figure 4.

Executing the three processes concurrently produces 520 different states which becomes pictorially
Program 2 LVMPD model integrating three concurrent processes, Calltaker, Dispatch, and Officer

//CALL TAKER PROCESS
CALLTAKER_LVMPD = (call_911->MFP_QUESTION|nocall_911->CALLTAKER_LVMPD),
MFP_QUESTION = (medical->MED_TRNSFR|fire->FIRE_TRNSFR|police->POLICE_TRNSFR),
MED_TRNSFR = (med_busy->MED_TRNSFR|transfr_med->CALLTAKER_LVMPD),
FIRE_TRNSFR = (fire_busy->FIRE_TRNSFR|transfr_fire->CALLTAKER_LVMPD),
POLICE_TRNSFR = (police_busy->POLICE_TRNSFR|transfr_police->CALLTAKER_LVMPD).

//DISPATCH PROCESS
DISPATCH_LVMPD = (transfr_police->INFO_LVMPD|nocall_lvmpd->DISPATCH_LVMPD),
INFO_LVMPD = (getinfo_lvmpd->RDNCHECK_LVMPD),
RDNCHECK_LVMPD = (rdnlvmpd_call->DISPATCH_LVMPD|newlvmpd_call->ASSN_OFFICER),
ASSN_OFFICER = (officer_order->OTHER_ER|officer_unavl->ASSN_OFFICER),
OTHER_ER = (other_er->DISPATCH_LVMPD|fm_req->CALL_FM|m_req->CALL_M|f_req->CALL_F),
CALL_FM = (fm_busy->CALL_FM|called_fm->DISPATCH_LVMPD),
CALL_M = (m_busy->CALL_M|called_m->DISPATCH_LVMPD),
CALL_F = (f_busy->CALL_F|called_f->DISPATCH_LVMPD),
POLICE_BLOCKOPT = (another_route->GOTO_SCENE|no_other_route->STUCK_LVMPD),

//LVMPD OFFICER MISSION PROCESS
OFFICER_LVMPD = (officer_order->OFFICER_LVMPD|officer_order->GOTO_SCENE),
GOTO_SCENE = (officer_drive->STREET_CON),
STREET_CON = (trfjam_lvmpd->STREET_CON|blocked_police->POLICE_BLOCKOPT|arrive_police->SCENE_LVMPD),
SCENE_LVMPD = (notneeded_er->TOWORNOT|fm_needed->GET_FM|m_needed->GET_M|f_needed->GET_F),
GET_FM = (fm_busy->GET_FM|called_fm->TOWORNOT),
GET_M = (m_busy->GET_M|called_m->TOWORNOT),
GET_F = (f_busy->GET_F|called_f->TOWORNOT),
TOWORNOT = (tow_needed->CONTACT_TOW|tow_notneeded->ER ARRIVAL),
CONTACT_TOW = (tow_notavl->CONTACT_TOW|tow_informed->ER ARRIVAL),
ER ARRIVAL = (aller_arrived->ETOVERRIDE|towarrive_loc->ERTASK_COMPLETION),
ERTASK_COMPLETION = (towdone_goback->TRAFFIC_MGT|ertogether_complete->ERTASK_COMPLETION),
TRAFFIC_MGT = (congestion_notclear->TRAFFIC_MGT|congestion_clear->OFFICER_LVMPD).
progress property is not satisfied for “cngtn clrd”. Safety property is verified by specifying a set of actions the system must satisfy at all times. This specification is executed concurrently with the system’s model for analysis. A case study is presented in section 5. An existing IM model is presented and analyzed using LTS tools.

**CASE STUDY**

After attending a meeting held by the local traffic incident management (TIM) team, where representatives from various emergency responder agencies gather in order to discuss regional issues involving traffic incidents, it was perceptible that a certain incident (a rollover) that occurred in Las Vegas was the center of discussion during the meeting. In the rollover, towing services were needed and a private towing company was contacted with some information about what kind of equipment was needed and the location of the rollover. However, the tow truck arrived 30 minutes late. Upon arriving, the wrong vehicle was towed, then it was discovered that different equipment was required to tow the right vehicle. After that, the officer and the tow company were discussing whose responsibility it is to clean-up the scene. This process delayed the scene clearance by two hours. Clearly, such complications are a result of decisions that are based on personal experiences and that a systematic way to discover and solve possible disruptions does not exist. These inefficiencies are inherent in the present system.

In order to model the Incident Management process in the rollover incident, the incident and LVMPD models presented in Figure 1 and Figure 4 are used. A model describing the tow company operation is presented in Figure 3 taking into consideration the issues discussed in this specific case.

**Program 3**  
Tow company model integrating two concurrent processes Dispatch and Driver

```plaintext
//DISPATCH_PROCESS
DISPATCH_TOW = {tow_informed->INFO|nocall->DISPATCH_TOW},
INFO = {get_info->RDN_CHECK},
RDN_CHECK = {rdn_call->DISPATCH_TOW|new_call->GIVE_ORDER},
GIVE_ORDER = {driver_order->DISPATCH_TOW|driver_unavl->GIVE_ORDER}.

//TOW MISSION PROCESS
DRIVER_TOW = {no_order->DRIVER_TOW|driver_order->GET_EQPT},
GET_EQPT = {eqpt_unavl->GET_EQPT|eqpt_avl->READYtoDRIVE},
READYtoDRIVE = {drive_loc->TRAFFIC_SITUATION},
TRAFFIC_SITUATION = {traffic_jam->TRAFFIC_SITUATION|blocked->BLOCKED_OPT|towarrive_loc->EVAL_LOC},
BLOCKED_OPT = {alt_route->READYtoDRIVE|no_alt_route->STUCK},
EVAL_LOC = {wrong_eqpt->GET_EQPT|truck_notneeded->RESOURCE_WASTE|right_eqpt->WAITtoTOW},
RESOURCE_WASTE = {drive_back->DRIVER_TOW},
WAITtoTOW = {waittotow->WAITtoTOW|cantow->CANTOW},
CANTOW = {another_tow->CANTOW|towdone_goback->DRIVER_TOW}.

//CONCURRENT PROCESS
||TOW_COMPANY = (DISPATCH_TOW || DRIVER_TOW).
```

The towing company has two concurrent processes: 1) dispatching which receives calls and information from customers and 2) delivering (‘driver’) the proper equipment to the scene. State diagrams for the towing company model are depicted in Figures 5(a), 5(b), and 6.

The state diagram representation of the towing company which includes the two processes “Dispatching” and “Delivering” become too complicated to represent pictorially as illustrated in Figure 6. After the models of the incident scene, LVMPD, and Towing Company are obtained, they are executed in parallel by the process described by the FSP in Figure 4.

Safety and liveness properties for the system are verified. Safety property is verified by the process illustrated in Figure 5 which indicates that at every state the system is not going into a situation where the
Program 4 IM process as a concurrent execution of three processes tow company, LVMPD, and Incident scene

\[ ||ER\_MNGMNT = (TOW\_COMPANY \mid LVMPD \mid PRE\_ACCIDENT) \]

process is blocked. The analysis of the safety checks for the complete Incident Management model and the towing company model are depicted in Figure 5.

Program 5 System verification for safety property

Safety property specification

property ACCIDENT\_RESOLVED = (accident→call\_911→congn\_clrd→ACCIDENT\_RESOLVED).

\[ ||ER\_MNGMNT = (TOW\_COMPANY \mid LVMPD \mid PRE\_ACCIDENT\mid ACCIDENT\_RESOLVED) \]

System verification for safety property

Trace to property violation in LVMPD.Officer\_LVMPD:

accident

call\_911

police

transf\_police

getinfo\_lvmpd

newlvmpd\_call

officer\_order

officer\_drive

blocked\_police

no\_other\_route

Trace to property violation in DRIVER\_TOW:

call\_rc

get\_info

new\_call

driver\_order

eqpt\_avl

drive\_loc

blocked

no\_alt\_route

Figure 5 implies that there exists a trace where the system does not comply with the safety requirement. Thus, the system is not safe and requires improvement in the specified trace. In this case study, the system is not safe when the officer reaches a blocked state preventing the arrival to the incident scene. Other safety checks may be specified for the Incident Management system or the individual agencies.

Liveness property is specified by the process illustrated in Figure 6 which provides that the system will eventually reach a certain state; the desired action in this case would be congestion clearance “cgn\_clrd”. The analysis of the liveness check is demonstrated in Figure 7.

Program 6 Liveness property specification

progress LVMPD\_MISSION\_ACCOMPLISHED = congtn\_clrd

The liveness check indicates that the system will not reach the desired state if it reaches one of the listed terminal states. The analysis in Figure 7 recognizes the set of terminal states where progress property is violated. It also provides the trace to terminal states. Therefore, the system is not “alive” and requires improvement in the indicated actions. Even though, the issues in the rollover incident were taken into consideration in modeling the IM system, analysis of the model has identified a trace which leads to the
Program 7 System verification for liveness property

Progress Check...
-- States: 14 Transitions: 107 Memory used: 4022K
Finding trace to cycle...
Finding trace in cycle...
Progress violation: LVMPD_MISSION_ACCOMPLISHED
TOW_MISSION_ACCOMPLISHED
Trace to terminal set of states:
accident
call_911
police
transfr_police
getinfo_lvmpd
newlvmpd_call
officer_order
officer_drive
blocked_police
no_other_route
Cycle in terminal set:
nocall
Actions in terminal set:
no_order, nocall, nocall_911, nocall_lvmpd,
notaller_arrived, stillcongested
Progress Check in: 62ms

action “no other route”. This action is also recognized to be a member of the terminal set whose members avert the system’s progression.

Ultimately, every Incident Management process should be live, implying it will always eventually reach a desired terminal state where the incident is cleared such as congestion. Ideally, every Incident Management process should be perfectly safe, signifying that the system is always safe. Safety can take various measures, according to which specifications are executed. For instance, a certain Incident Management system may be considered safe if delay does not exceed a certain amount or if only certain routes are allowed.

FURTHER ANALYSIS
This approach studied in this paper can be expanded to include all resources for every process within the agencies as well as to model all agencies that might be involved in the Incident Management process. In addition, this model can be enhanced by including real time information within the states representing traffic conditions or other continuous, random activities. Finally, real data and statistics can be incorporated to support predictions and estimations.

CONCLUSIONS
In this study, using formal language methods for modeling the Incident Management process was demonstrated. Formal languages methodology provides the ability to perform rigorous debugging and analysis through which an increasing robustness of the Incident Management system can be achieved upon implementation. Furthermore, this approach allowed conducting analyses of concurrent execution of processes with liveness and safety properties specifications. Thus, this method offers flexibility in modeling various Incident Management systems accounting for every possible existing scenario leading to development of customized systems resulting in a more successful Incident Management.
REFERENCES


(a) Call taker process

(b) Dispatch process

(c) Officer process

FIGURE 4 State diagrams for the LVMPD model
FIGURE 5 State diagrams for the tow company model
FIGURE 6  Concurrent state diagrams dispatch and driver processes