Freeway Incident Management Expert System Design

Edmond Chin-Ping Chang and Kunhuang Huarng

Nonrecurring incidents may cause unexpected congestion on freeways, even when surveillance, communication, and control (SC&C) systems are in operation. A knowledge-based expert system has been developed for microcomputers to assist in urban freeway corridor incident management. Overall study activities include literature review, conceptual design, prototype system development, program documentation, and user interface design of the expert system. This paper documents the expert system being developed, which includes a graphics user interface, decision-making rules, and a knowledge inference mechanism to automate freeway incident management applications. The benefits of using this expert system are also summarized.

Freeway and arterial incidents often occur unexpectedly and cause undesirable traffic congestion and regional mobility loss, even when computerized freeway surveillance, communications, and control (SC&C) systems are in operation. Automatic incident detection should apply information observed from freeway detector stations. The most commonly used method is the comparative method (California-type algorithm) in which traffic operational characteristics between consecutive detector stations are continuously monitored and closely evaluated.

A microcomputer-based, knowledge-based expert system, Incident Management Expert System (IMES), has been developed to assist with control operations by improving urban freeway corridor incident management. An effort was made to summarize, extract, and select the information needed during the decision-making process to implement urban freeway incident management strategies. This paper documents the development of a microcomputer-based expert system design for assisting in freeway incident management. In the following sections the incident management process, microcomputer system design, Microsoft Windows software interface features, and user-definable elements are described that allow for flexibility in future system expansions.

IMES has been developed in the Microsoft Windows environment, which provides a user-friendly interface that makes IMES easy to learn and use. IMES uses the unique features of Windows to provide a graphics user interface and visual programming through a rule editor. The rule editor allows users to maintain a flexible rule base in the expert system without requiring extensive programming knowledge and previous experience. The IMES system separates inputs and outputs into data files. Whenever inputs and outputs are changed, IMES is modified to reflect the changes. The inference engine, developed in C-Language Integration Production System (CLIPS) 5.1, provides object-oriented features to facilitate software reuse, encapsulation, and data abstraction. These advantages make IMES a reusable and easily maintained system. IMES is a stand-alone program, running on an MS DOS-based IBM/XT/AT/386 or compatible microcomputers, with or without a math coprocessor.

Texas Transportation Institute, Texas A&M University System, College Station, Tex. 77843-3135.

INCIDENT MANAGEMENT PROCESS

In this section the basic information requirements and control responses needed to make proper decisions during urban corridor freeway incident management are discussed. The following conceptual design describes the freeway incident management process. The information analysis covers information type, quantity, and quality of data, or overall information needed in highway system analysis. The decision-making process was identified though a step-by-step analysis after an alarm sounds indicating the potential occurrence of an incident. The analysis focused on the different types of control decisions and responses available to control operators and field personnel. The entire process emphasized identification of data requirements and information flow to make timely decisions, such as selection of the proper incident response (*1*).

Decision-Making Process

Figure 1 is a step-by-step flowchart representing the typical decision-making process normally followed by control center operators when they respond to an identified freeway incident (*2*). As indicated, the decision-making process should include five steps, including incident detection, confirmation, prediction, management, and response (*3,4*). This expert system was designed to provide assistance to speed response.

Incident Detection

There may actually be different levels of information requirements or alarm status, through combinations of video images or audio signals, that can notify control center operators that an "abnormal" operating condition has occurred in the freeway surveillance environment. This incident condition may include field equipment failure, a drastic change in traffic conditions, or a remark about scheduled special operations. Depending on the nature of the freeway incidents and needed management responses, the status of a potential freeway in-

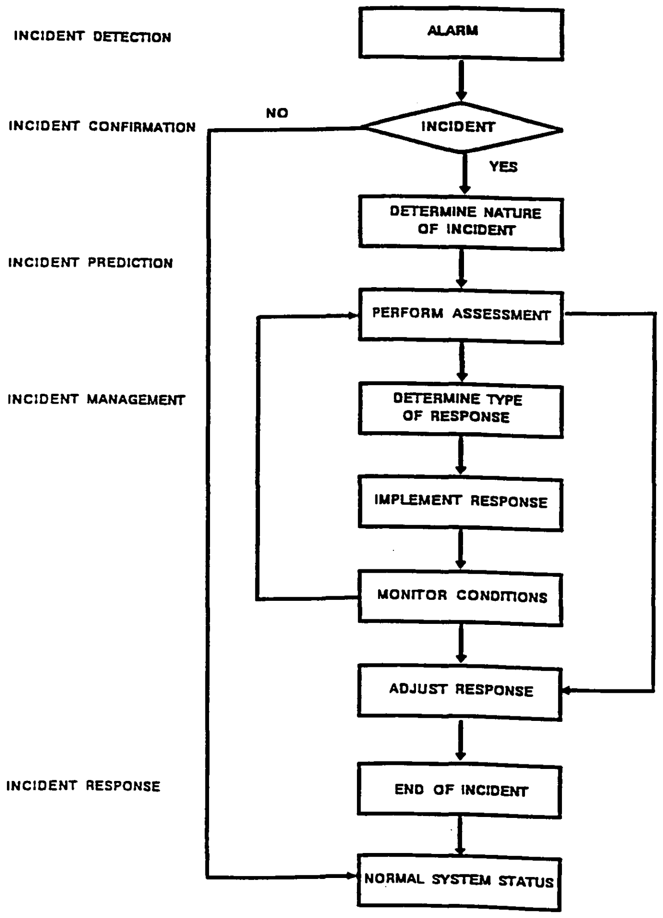


FIGURE 1 Incident management decision making.

cident can be properly determined. On the basis of traffic control requirements of the freeway surveillance and control system, a freeway incident alarm may sound in response to four possible operating conditions:

1. An automatic incident detection algorithm;
2. A call from the field by a service patrol, police, and so forth;
3. The observation of traffic flows; or
4. A combination of visual and automatic techniques.

Incident detection by electronic surveillance serves to monitor real-time traffic data through vehicular detectors installed at critical locations along the freeway. When a delay-related incident occurs, freeway capacity is reduced at the point of incident occurrence. If capacity is reduced to an amount less than the existing demand and traffic occupancy is greater than a predetermined value, an incident has likely occurred. Similarly, incidents can be detected through logic by evaluating variations in traffic flow characteristics. Some controlled experiments have been conducted using operating speed as the determining variable. However, most electronic surveillance systems can also use occupancy data for incident detection.

For example, in Los Angeles, changes in either the lane occupancy or the percentage of time that vehicles spend over a particular detector location will provide an indication of congestion when an incident has occurred. Normally, computer software can calculate the difference in the measurements between adjacent detector stations. In some cases, mainlane vehicle detectors may be spaced at half-mile intervals. The incident alert condition can be signaled automatically by the computer through algorithmic analysis when a relative percent change between the present occupancy and that of the preceding samples for the downstream detectors exceeds a certain threshold value.

As additional traffic information immediately upstream of the incident is obtained, control operators can make decisions to activate appropriate responses. The advantage of detector-based surveillance is that it can continuously monitor the network at a relatively low operating cost with minimal human supervision. The information can be used for other traffic control tasks, such as establishing metering rates for traffic-responsive ramp metering systems. The main disadvantage of the system is that the nature of the incident cannot be readily identified, and some other type of surveillance is often required to determine what type of response is needed.

Incident Confirmation

When an incident alarm goes off, it is necessary to identify all the possible triggering factors of the incident and confirm its occurrence through other means. In particular, the freeway traffic management system should act automatically to

1. Determine whether an operational failure in the surveillance, communication, and control system has led to the alarm;
2. Identify the reasonableness of the incident alarm and point out the locations of the incident; and
3. Establish a level of confidence in the alarm by confirming the incident through other field identification techniques.

Incident Identification

Given that a freeway incident has already occurred and has been confirmed in the field, it is necessary to determine the nature of the incident before any further control action can be taken.

With a number of unknown factors, the overall incident identification process should take into account

1. Location of the incident: freeway mainlane, shoulder, median, on-ramp, off-ramp, or interconnecting service road;
2. Type of incident: accident, stalled vehicle, cargo spill, or environmental condition; and
3. Severity of the incident: number and size of vehicles involved; number of lanes blocked; property damage only, injury, or fatality; type of cargo involved; and exploration potential.

Incident Assessment

Next the control center operator must assess the overall operating condition of the freeway corridor and the nature of the incident. It is important to identify the available design elements involved for timely decision making. Comprehensive incident assessment must consider the following information:

1. The capabilities of the organization in terms of equipment availability, status, and location; personnel availability; and operating procedures (who has agreed to do what);
2. The likely duration of the incident acquired from historical experience, computed from an incident prediction algorithm, or assessed from similar incidents;
3. The potential impact on traffic flow and route, time of day, and traffic volumes; and
4. The status of the primary and diversion routes for a potential freeway diversion and for releasing traffic information.

Incident Response

It is noted that the control response to be taken depends highly on locally established practices and operating procedures. If the control response is multijurisdictional, there is the potential for conflict among different operating agencies. Historically, operators contacted the police or highway patrol, who determined the need for a response. To establish a proper incident management system, it is important to develop a relationship of mutual trust among all responsible participating agencies. Incident assessment can lead to the determination of the type of control responses required for different incident conditions.

The incident response involves immediate decisions relating to

1. Personnel and equipment: who is at the scene, who else should be sent to the scene, and who to inform;
2. Real-time motorist information: signs, Highway Advisory Radio (HAR), radio, TV broadcasts;
3. Off-site traffic control for diversion; and
4. Available traffic control strategy.

In the United States, freeway management agencies have used various coordination schemes among the different levels of freeway agencies, highway patrols, and local police to manage freeway corridor traffic. For example, in Chicago, service patrols take care of disabled vehicles without calling the police except in the event of an accident. In Los Angeles and Long Island, the police must be present to remove a disabled vehicle. Either way, a response should be implemented, and conditions monitored and assessed. Incident response is adjusted as needed on the basis of feedback from freeway monitoring systems.

Condition Analysis

Condition analysis addresses the control decisions needed and determines the types of responses available to control operators and field personnel. Condition analyses should allow the operators to assess continuously the basic data elements that describe the nature and extent of the freeway incident.

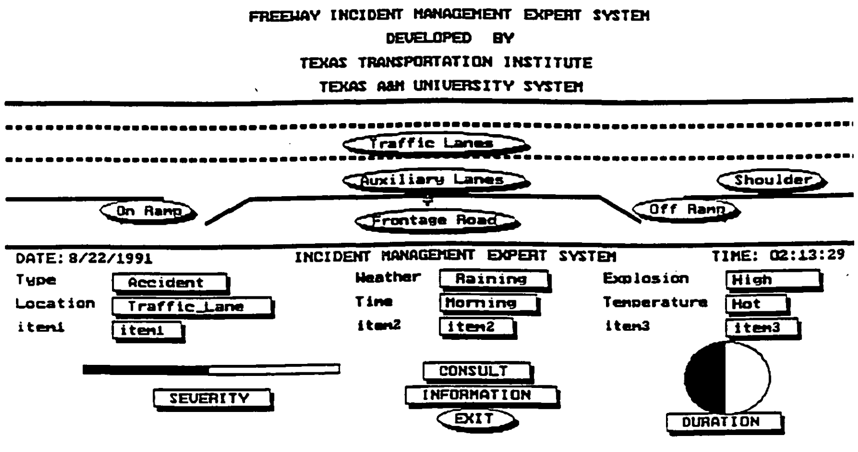
The condition analysis focuses on identifying the overall system data requirements that can feed the information flowchart developed during the freeway incident management process. The data elements mainly include type of data, amount of data, form of the data base input, source of data, and how the data are acquired.

The realistic availability and suitability of basic data elements depend on the freeway management system design. It is important to investigate basic data needs, the system information process, and communication requirements while planning traffic control strategies. Design considerations must be taken into account during the planning, design, and development stages of the computerized freeway corridor traffic management system.

MICROCOMPUTER SYSTEM

IMES is a microcomputer-based expert system environment developed by the Texas Transportation Institute at Texas A&M University. The IMES system provides an intelligent, user-friendly expert system framework by applying several state-of-the-art computer programming techniques. The system components include a graphics user interface, a mouse-supporting function, a rule base, a menu selection file, a response file, and the CLIPS expert system building tool.

As shown in Figure 2, there are three display components in the graphics user interface. The upper portion displays


FREEWAY INCIDENT MANAGEMENT EXPERT SYSTEM
DEVELOPED BY
TEXAS TRANSPORTATION INSTITUTE
TEXAS A&M UNIVERSITY SYSTEM

DATE: 8/22/1991 INCIDENT MANAGEMENT EXPERT SYSTEM TIME: 02:13:29

Type	Accident	Weather	Raining	Explosion	High
Location	Traffic_Lane	Time	Morning	Temperature	Hot
item1	item1	item2	item2	item3	item3

FIGURE 2 System initial screen.

default and help information. The middle portion displays and marks various portions of a freeway from which users can easily identify the locations of incidents. When users click on any marked area, the corresponding help information is displayed on the upper portion of the screen to provide explanations about this area of the freeway. The lower portion displays the menu selections for describing the incidents. When users select any menu item, IMES asserts the item as a fact in the fact base.

The mouse allows users to point at any place on the screen directly and select menu items. Conventional keyboard inputs require typing, which may involve several problems: typing is boring and tedious, typographical errors are inevitable, and users cannot feel control. Unlike keyboard input, control with the mouse creates a convenient user input medium.

The expert system is the heart of IMES. It serves as a consultant that helps users make appropriate decisions according to type of incident. The rule base of the expert system automates the process of incident management: it fires the corresponding rules and generates responses appropriate to manage certain incidents without the user having to go through the process of obtaining them. CLIPS, an expert system building tool, contains the reasoning mechanism or inference engine that performs forward-chaining to formulate responses as advice to users.

Built-in flexibility has been implemented through maximum system expansion capability. Users can change menu items using a text editor, the details of which are described later in this paper. Similarly, users can change the responses by modifying a text file edited by a common word processor. All these modifications do not affect the contents of IMES. IMES reads these files as inputs and displays items correspondingly, so that system expansion can be performed without recompilation.

Users can easily maintain the rule base in IMES. By applying Windows features, IMES provides a rule editor, allowing users to modify rules without having knowledge of CLIPS and programming. This process is described in the section headed "Windows Environment."

System Architecture

The basic system architecture is shown in Figure 3. Users can access a text editor, the IMES main screen, and a rule editor via Windows. COND.TXT and RESPONSE.DAT are text files. The text editor is used to maintain or expand these files for menu selections and responses. The rule editor, supporting visual programming, provides a convenient way to maintain and expand the rule base. The IMES main screen is a graphics user interface, displaying menu selections, responses, and help information. Users interact with IMES through the IMES main screen.

Since flexibility is one of the IMES design concerns, menu selections, responses, and the rule base are separated from IMES. These components are stored in COND.TXT, RESPONSE.DAT, and IMES.CLP, respectively. When IMES is invoked, it reads these files and displays menu selections according to the items read from COND.TXT. The response items that IMES can provide are read from RESPONSE.DAT. The rule base in IMES is read from

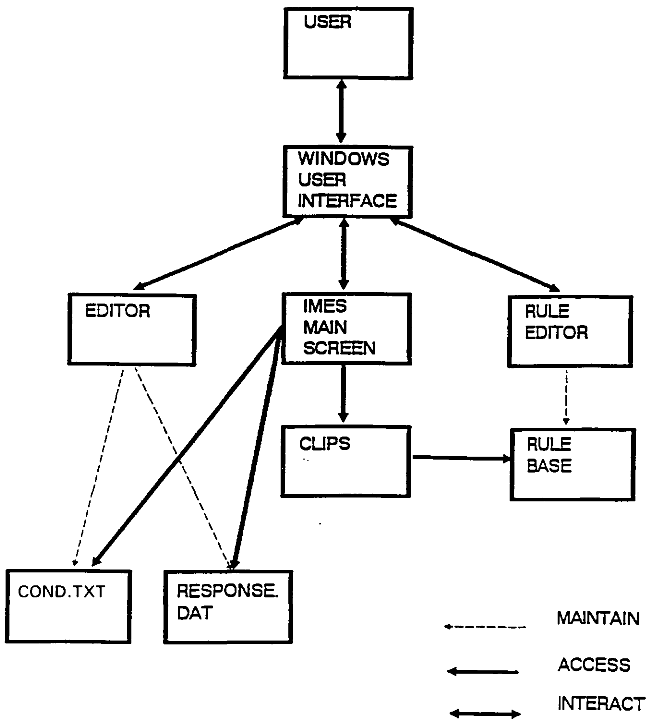


FIGURE 3 System architecture of IMES.

IMES.CLP, which is the expert system built in CLIPS. IMES invokes CLIPS to read the rule base. IMES is then ready to take users' selections and generate appropriate responses for managing certain incidents.

Since these components are separated, they can be maintained individually and easily. COND.TXT and RESPONSE.DAT can be maintained via a text editor. IMES.CLP can be maintained via the rule editor provided by IMES. As stated earlier, any change to these components requires no recompilation of IMES. As a result, no programming experience or knowledge is required to maintain these components.

System Configuration

IMES runs on an IBM PC or IBM-PC-compatible machine with an EGA, VGA, or Hercules graphics adapter. IMES runs well with or without a mouse. When IMES runs without a mouse, keyboard input is effective. To run IMES properly, all the programs or files such as COND.TXT, RESPONSE.DAT, IMES.EXE, and IMES.CLP should be included.

CLIPS

CLIPS is an expert system building tool developed and maintained by the National Aeronautics and Space Administration (*5,6*). CLIPS was developed in the C programming language and can be integrated or embedded within conventional C programs.

Rule-Based Expert System Building Tool

CLIPS is a rule-based expert system building tool with a forward-chaining inference engine. Facts and rules are the underlying knowledge representation scheme. A fact is an essential data element. Each fact represents interface information constituted by one or more items. The whole set of facts is called the fact base.

A rule, the major way of representing knowledge, consists of a collection of preconditions and postconditions. The preconditions of a rule list the conditions to be matched with facts, whereas the postconditions are actions. Once the preconditions of a rule have been matched with the facts, the postconditions of the rule are executed. The whole set of rules in an expert system is called the rule base. CLIPS provides an inference mechanism called an inference engine to match the preconditions of rules and to execute the corresponding postconditions.

Once rules have been created and facts have been prepared, CLIPS is ready to run. Unlike conventional programming, CLIPS need not specify the sequence of operations explicitly. The execution cycle in CLIPS is described as follows:

1. CLIPS examines rules to see if the preconditions of the rules are matched with the facts.
2. All rules whose preconditions are met are activated and put into the agenda. The top rule in the agenda is selected and fired. When the rule is fired, the postconditions of the rule are executed.
3. After the execution, if the fact base has been changed, the cycle returns to Step 1; otherwise, it returns to Step 2 until the agenda is empty.

CLIPS 5.1 is highly portable; it can be used in various machines and software environments, such as IBM PC MS-DOS, Macintosh, and VAX VMS.

Object-Oriented Programming

The latest version of CLIPS, version 5.1, supports object-oriented programming development, which provides several features to enhance software quality (*7*): use of the common domain problem, software stability, and software reuse. In object-oriented programming, the major concepts are class and object. A class is defined as a group of similar instances, and an object is defined as an instance of a class. The concept of class expresses the commonalty of the domain problem. Each class or object consists of several attributes called slots to store values. Each slot comprises several attributes called properties to describe the slot.

Each subclass or object can inherit from one or more than one parent class. The useful features of the parent classes are broadcast automatically to the subclass or object. In other words, the features of the parent classes can be reused without redefinition.

Communication among objects is accomplished via message passing schemes. The message is sent to the designated object to modify slots of the object. If the data of the object are encapsulated, the contents of the object cannot be changed without message sending. Unintended modification is impossible. Since the modification of contents of the object must be specified, encapsulation facilitates program debugging.

On the basis of encapsulation, object-oriented programming supports data abstraction, the purpose of which is to define a data type by the methods that can be applied to the object of the data type (*8*). The state of the object can be accessed by its methods. Communication among objects or classes can be done only through message sending, and a message usually contains the information to be changed. When an object or class receives a message, the state of the object or class is changed correspondingly. Communication with messages is considered a mechanism for handling software complexity (*7*).

IMES uses CLIPS Object-Oriented Language (COOL) to model incidents. Whenever an incident occurs, it can be declared as an instance, that is, an object, of the incident class. The object inherits all the attributes (slots), such as incident type, location, time, and so forth, and properties, such as allowed words, of the incident class without redefinition. The object is encapsulated because contents of the object cannot be accessed without message sending. Therefore, when IMES uses COOL, it can easily manage multiple incidents at the same time.

For the same reason, the responses can also be declared as a response class. When IMES provides suggestions for managing each incident, those suggestions can be an instance, or object, of the response class.

Operating Procedure

The basic procedure of interacting with IMES is described as follows. Users can select menu items from the initial screen. There are different kinds of menu selections. Help information is invoked by clicking items such as On Ramp, Traffic Lanes, Auxiliary Lanes, Frontage Roads, Off Ramp, and Shoulders. Help information is displayed on the top half of the screen. The graphics user interface provides another kind of menu. When users select any item such as Type, Location, Weather, Time, Explosion, or Temperature, some related information will pop up for selection. For example, when users select the item Location, the pop-up menu will display Traffic_Lane, On_Ramp, Off_Ramp, Shoulder, Aux_Lane, and Unknown from which the users may select. When clicked, the selected information is asserted to the fact base of the expert system in IMES for later inference. The initial display then returns.

Similarly, the menus Severity and Duration are for users to input the degree of incident severity and duration. The graphics display changes according to users' input. After the inquiry has been made, the display will then return to the initial screen.

The procedure for manipulating IMES is described further in Figure 4. When users invoke IMES, the IMES main screen shows up. From the main screen, users can choose the data selections from the graphics user interface, system help information, or default system information. After users choose the data selections, they can select Consult to request suggestions or provide necessary responses from the expert system based on the data selections. After users select Consult, suggestions from IMES are given as shown in Figure 5. The

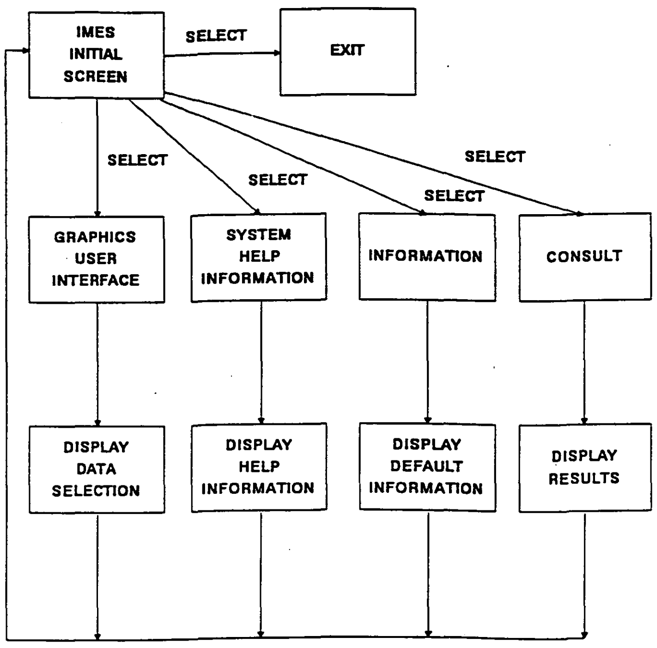


FIGURE 4 Procedure for manipulating IMES.

upper part of the main screen displays the prototype suggestions such as remove, act, call police, and so on.

WINDOWS ENVIRONMENT

Graphics User Interface

The graphics user interface available in Windows software provides an enhanced environment for conveying messages and displaying formatted text. Graphics makes the interaction between a computer and a user closer by manipulating all the objects on the screen. Windows attracts users to its graphical display and improves usability by providing a user-friendly interface. Each window contains a title bar describing the window; a control menu box consisting of a list of commands such as Resize, Move, Maximize, Minimize, and Close; a Maximize button and Minimize button to alter the size of the window; a menu bar listing the menus available; a vertical and a horizontal scrolling bar to move documents; and the window itself. Windows employs a mouse that allows the user to point to any portion of the screen directly. Each Windows application presents the same user interface described above, which enables users to learn other Windows applications easily. The learning time and cost for Windows are much less than they are for many other software applications.

Windows also provides multitasking. When applications are running, users can invoke other applications. For example, users can invoke an editor to edit a document, a graphics tool to draw charts, and IMES to manage incidents, all at the same time. Multitasking allows real-time monitoring, in which users can monitor several applications at a time. When the user needs to switch from one application to another, there is no need to quit the application being worked on. When it is clicked on, the intended application becomes active. IMES is built into the Windows environment and benefits from Windows' advantages. Multitasking in Windows allows multi-inquiry. Users can invoke IMES more than once, each time as a independent task. According to the inputs for various incidents, each IMES task generates different suggestions. Users can compare and analyze the differences among the suggestions based on the inputs for incidents.

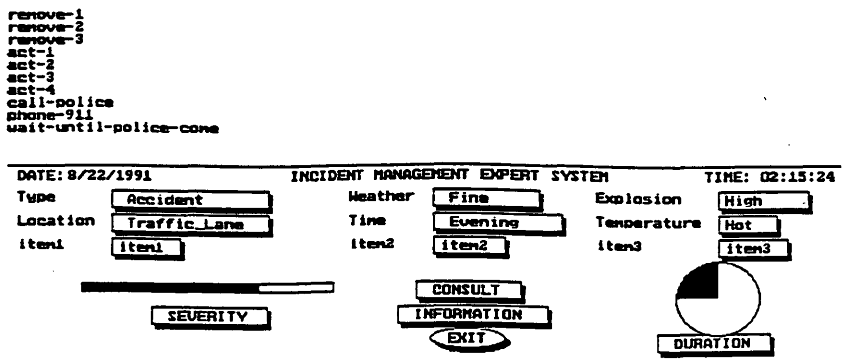
Visual Programming

The programming population is growing rapidly, whereas the structure of programming languages remains, by and large, textual. Computer engineers are striving for solutions to make programming more accessible to this large population. Since the cost of graphics-related hardware and software is decreasing, graphics is becoming more popular. In addition, graphics is considered more powerful than text in many ways (*9*):

1. Graphics is more powerful than text as a medium of communication,
2. Graphics has no language barrier, and
3. Graphics assists understanding.

Visual programming takes advantages of sophisticated graphics and becomes a solution for making programming more accessible.

Visual programming applies meaningful graphics displays to aid users in understanding, creating, and maintaining software (*10*). Visual programming has been widely applied to



```
remove-1
remove-2
remove-3
act-1
act-2
act-3
act-4
call-police
phone-911
wait-until-police-come
```

DATE: 8/22/1991 INCIDENT MANAGEMENT EXPERT SYSTEM TIME: 02:15:24

Type	Accident	Weather	Fine	Explosion	High
Location	Traffic_Lane	Time	Evening	Temperature	Hot
item1	item1	item2	item2	item3	item3

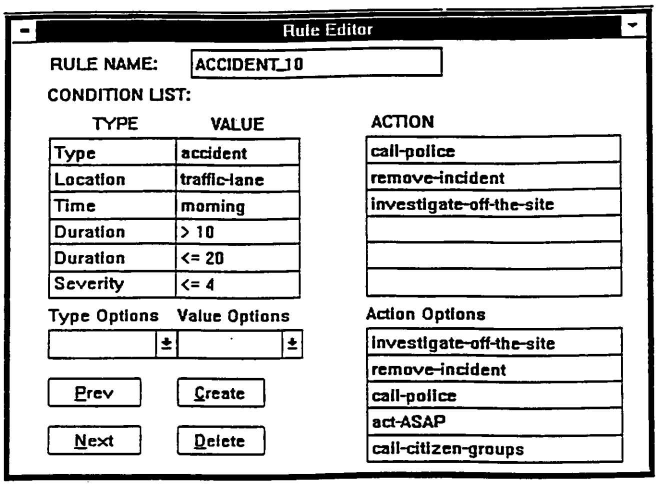
FIGURE 5 Example of expert system response.

several areas. Visual user interfaces, such as Smalltalk and Cedar, use graphical displays to assist the interaction between user and application (*11,12*). Languages for visual interactions, such as ICDL and HI-VISUAL, support various utilities for graphical displays (*13,14*). Visualization of software design, such as Program Visualization and PegaSys, supports all kinds of graphical tools to assist software development throughout the software life cycle (*15,16*). Algorithm animation, such as Balsa, assists the visualization of algorithms (*10,17*). Visual editing, such as Cornell and Garden, provides syntax-directed editors to assist programmers in preparing and maintaining programs (*18–20*).

Following the philosophy of visual programming to provide a more user-friendly software support environment, IMES provides a graphical rule editor (Figure 6) to assist users in maintaining a rule base (*21*). A graphical rule editor provides both conditions and actions. A condition contains menu selections, which are type and value options to construct the condition of a rule. An action contains the responses for constructing the action of a rule. The corresponding rule generated by the graphical rule editor is shown in Figure 6. When the graphical rule editor is invoked, it reads COND.TXT as the possible conditions and RESPONSE.DAT as the actions.

Operating Procedure

To create a new rule, users select conditions from the condition menu list and actions from the action menu list. According to the conditions and actions selected, the rule editor can automatically generate a corresponding rule. To delete



RULE NAME: ACCIDENT_10

CONDITION LIST:

TYPE	VALUE
Type	accident
Location	traffic-lane
Time	morning
Duration	> 10
Duration	<= 20
Severity	<= 4

ACTION
call-police
remove-incident
investigate-off-the-site

Type Options | Value Options

Action Options
investigate-off-the-site
remove-incident
call-police
act-ASAP
call-citizen-groups

```
(defrule ACCIDENT_10 "accident"
        (Type accident)
        (Location traffic-lane)
        (Time morning)
        (Duration ?x)
        (test (and (> ?x 10)
                   (<= ?x 20)
              )
        )
        (Severity ?y)
        (test (<= ?y 4))
=>
        (assert (Macro call-police))
        (assert (Macro remove-incident))
        (assert (Macro investigate-off-the-site))
)
```

FIGURE 6 Rule editor and rule generated.

an existing rule, the user selects a rule and then chooses Delete to remove the rule from the rule base. To modify an existing rule, the user can first delete the rule and create a new rule following the procedure above.

Benefits

The benefits of using the graphical rule editor include the following:

1. Rather than memorize the condition and action items of the rules, users can easily select items from the menu lists. The use of the rule editor reduces the complexity of maintaining IMES. In other words, user productivity can be improved.
2. Users can select items rather than typing them, thus avoiding clerical errors.
3. It is unnecessary that the user be a CLIPS expert to maintain the rule base. After the user selects the items, the system automatically generates the corresponding rules. The generated rules are syntactically correct.
4. The production rule editor is easy to use and learn; the usability of IMES is thus increased.

Future Expansion

Windows provides Dynamic Data Exchange (DDE) to share data among Windows applications. When an alarm occurs, the message can be sent to the computer in the control center. Through DDE, IMES can be automatically invoked and become active. IMES is then ready to take the inputs and generate suggestions. IMES can be expanded with DDE to assist real-time incident management. The user can also use a cut-and-paste function with a communications software to transmit computer suggestions to remote computers.

Windows provides functions to process multimedia data elements, such as voice, sound, and animation elements. IMES can be expanded to accommodate multimedia features to assist in incident management activities, such as voice suggestions. In addition, the pen-computing extension of Windows can help users deal with incidents on the scene without using a keyboard.

USER-DEFINABLE OPERATIONS

IMES can be used intuitively via the graphics-based user interface. The design rationales are intended to allow a user to enhance the expert system, which represents the necessary operational considerations. The system has been designed with three unique user-definable program features:

1. The decision-making production rules are defined in an external text file so the user can easily make modifications through the rule editor.
2. The control responses are also specified in an external text file so that the user can provide specific responses.
3. The user can also create additional study variables for the site-specific requirements. All of these study variables are

stored in an external text file. By defining all operating conditions and allowing the revision of response messages external to the program, the user can reflect the operational requirements without modifying internal program codes.

User-Defined Conditions

IMES allows the user to define new conditions. The user is able to categorize conditions into three groups each with five conditions, for a total of 15 additional conditions available in addition to those built into IMES. To specify new conditions, the user need only change the text file COND.TXT using any word processor that can edit text files.

In user-defined conditions, there are labels and conditions. The label is displayed as the title of the conditions, and the conditions are displayed as the contents of pop-up menus. When a user modifies the labels and the conditions, IMES reads COND.TXT as an input file. Since the modifications do not affect the internal contents of IMES, no recompilation of IMES is needed. When a user runs IMES after modification, the new display will reflect the modifications made.

User-Defined Responses

IMES is a generalized expert system for assisting incident management. Since there are no specific response plans available, generic responses are provided. A user can design his own response messages as needed. IMES will display the messages according to the user-defined responses. IMES provides seven types of response messages: call-point-authority, call-police, act-ASAP, remove-incident, investigate-off-the-site, call-citizen-groups, and call- Texas-SDHPT. Each type of response message is a title for the same type of response messages. Users are free to define response messages by type. For example, users can define call-police, phone-911, and wait-until-police-come under the type of response message (call-police). In RESPONSE.DAT, each type of response message is preceded by a macro followed by user-defined response messages. A macro is added at the end of RESPONSE.DAT.

CONCLUSIONS AND RECOMMENDATIONS

Incidents may cause unexpected congestion on freeways, even when surveillance, communication, and control (SC&C) systems are in operation. Any accident, truck spill, or stalled vehicle on or near mainlanes can significantly affect system performance and create hazardous situations for involved motorists, approaching commuters, and passing traffic. Freeway control and operating strategies are essential for successful system operations. Being an integral component of the freeway control system, incident management is especially important while freeways are operating near, at, or beyond their physical capacities. Engineers must make decisions concerning operational effectiveness and trade-offs, and control decisions may be bound by physical constraints, traffic characteristics, or traffic control practices. Off-line computer software has been developed to assist traffic control operators in identifying unique traffic operating conditions and suitable control strategies necessary for determining when and how computerized traffic control systems should respond.

In this paper a microcomputer-based expert system is described developed in the Windows environment with the implementation of a user-friendly interface, a rule editor, decision-making rules, and a knowledge inference mechanism to automate freeway incident management applications. IMES has been developed as a decision-making assistant for potential users in determining the different actions needed to handle specific freeway incident management problems. The benefits of applying IMES in freeway incident management can be summarized. The rule editor provides visual programming to facilitate the maintenance of the rule base. IMES allows normal users to adjust and customize conditions and responses. Since the conditions and responses are separated from the executable program, any further modification to the conditions and responses requires no recompilation of the executable program. The Windows environment provides a user-friendly environment to enhance usability. The rule base provides quick suggestions to assist incident management. As a result, IMES can facilitate freeway incident management.

The IMES system is presently designed for off-line control strategy evaluation. However, because the system has been designed with external dynamic data linkage, the IMES system can be implemented along with on-line urban highway traffic control systems to automatically identify proper control strategies as soon as nonrecurring arterial and freeway conditions have been identified. With further system validation and verification with realistic incidents, the system can be expanded further to include new generations of the arterial street network and freeway corridor system control concepts to automate on-line, real-time traffic responses and management strategies.

ACKNOWLEDGMENTS

The authors would like to express their appreciation for the research support, personnel cooperation, and coordination assistance during the course of this study as provided in part by the Texas Department of Transportation. Staff support from the Texas Transportation Institute, Texas A&M University System, is also appreciated.

REFERENCES

1. *Incident Management.* Final Report. Trucking Research Institute, ATA Foundation, Inc.; Cambridge Systems, Inc., Oct. 1990.
2. E. C. Chang. *Corridor Analysis for Reconstruction Activities, Traffic Control Strategies, and Incident Management Techniques: Task B. Development of Expert Systems for Freeway Incident Management—Literature Review.* TTI Research Report 1188-2. Texas Transportation Institute, College Station, Feb. 1990.
3. E. C. Chang. Expert Systems Applications on Freeway Incident Management. Presented at International Conference on the Applications of Advanced Technologies in Transportation Engineering, American Society of Civil Engineers, San Diego, Calif., Feb. 6–8, 1989.
4. S. G. Ritchie and N. A. Prosser. A Real-Time Expert System Approach To Freeway Incident Management. In *Transportation*

Research Record 1320, TRB, National Research Council, Washington, D.C., 1991.
5. *CLIPS Reference Manual*, Version 5.1, Vol. 1. Software Technology Branch, Lyndon B. Johnson Space Center, Sept. 1991.
6. *CLIPS User's Guide*, Version 5.1, Vol. 2. Software Technology Branch, Lyndon B. Johnson Space Center, Sept. 1991.
7. P. Coad and E. Yourdon. *Object-Oriented Analysis*, 2nd ed. Prentice Hall, Englewood Cliffs, N.J., 1991.
8. *Dictionary of Computing*. Oxford University Press, 1986.
9. N. C. Shu. *Visual Programming*. Van Nostrand Reinhold Company, Inc., 1988.
10. J. T. Stasko. Tango: A Framework and System for Algorithm Animation. *IEEE Computer*, Sept. 1990.
11. A. Goldberg. *Smalltalk-80: The Interactive Programming Environment*. Addison-Wesley, Reading, Pa., 1984.
12. D. C. Swinehart et al. A Structured View of the Cedar Programming Environment. *ACM Transactions on Programming Languages and Systems*, Vol. 8, No. 4, 1986.
13. C. F. Herot. Spatial Management of Data. *ACM Transactions on Database Systems*, Vol. 5, No. 4, Dec. 1980.
14. M. Hirakawa, N. Monden, I. Yoshimoto, M. Tanaka, and T. Ichikawa. HI-VISUAL: A Language Supporting Visual Interaction in Programming. In *Visual Programming* (S. K. Chang et al., eds.), Plenum Press, 1986.
15. G. P. Brown, R. T. Carling, C. F. Herot, D. A. Kramlich, and P. Souza. Programming Visualization: Graphical Support for Software Development. *IEEE Computer,* Vol. 18, No. 8, Aug. 1985.
16. M. Moriconi and D. F. Hare. Visualizing Program Designs Through PegaSys. *IEEE Computer*, Vol. 18, No. 8, Aug. 1985.
17. M. H. Brown and R. Sedgewick. A System for Algorithm Animation. *Computer Graphics*, Vol. 18, No. 3, 1985.
18. T. Teitelbaum and T. Reps. The Cornell Program Synthesizer: A Syntax-Directed Programming Environment. *Communications of the ACM*, Vol. 24, No. 9, 1981.
19. S. P. Reiss. Garden Tools: Support for Graphical Programming. In *Advanced Programming Environments: Lecture Notes in Computer Science #244* (R. Conradi et al., eds.), Springer-Verlag, 1986.
20. F. Arefi, C. E. Hughes, and D. A. Workman. Automatically Generating Visual Syntax-Directed Editors. *Communications of the ACM*, Vol. 33, No. 3, March 1990.
21. A. L. Ambler and M. M. Burnett. Influence of Visual Technology on the Evolution of Language Environments. *Computer*, Oct. 1989.