IMPACT: Development and Deployment Experience of Network Event Correlation Applications

Gabriel Jakobson, Mark Weissman, Shri Goyal
GTE Laboratories Incorporated
40 Sylvan Road
Waltham, MA 02254, USA
gj00@gte.com

Abstract

The development and deployment experience of IMPACT, an expert system shell dedicated to the tasks of real-time network alarm correlation, is discussed. IMPACT was developed at GTE Laboratories and since 1991 has been used at GTE to build and field various telecommunications network (both wireline and cellular) event correlation applications.

IMPACT incorporates an advanced real-time event correlation model which is based on the principles of model-based reasoning. The IMPACT application development environment contains multiple structural and graphical editors to describe network element classes, network configuration models, correlations, and correlation rules. IMPACT is a heavily graphics-oriented system with components to visualize network connectivity, present network events, and display geographic maps. IMPACT runs on UNIX workstations, including IBM RS6000, Sun Microsystems SPARC, and SGI Indy. On SGI Indy R4400SC, the performance of IMPACT reaches 15–30 events parsed and correlated per second. IMPACT has been integrated with two GTE network alarm monitoring systems, SmartAlert and ISM2000. IMPACT has been developed using C, X Windows, CLIPS, and Tcl/Tk.

This paper summarizes the challenges of alarm correlation application development in the environment of a large telecommunications company, and states the benefits of using dedicated expert system shells.

Introduction

GTE is a major telecommunications company which manages a large number of different voice and data networks, both wireline and wireless. Real-time surveillance and fault management of these networks is becoming increasingly difficult because of the high volume of network alarm, status, and other event messages presented on the network operation terminals. For example, during major trunk failures, the number of generated alarms may reach tens of alarms per second. Inadequate handling of these alarms leads to incorrect interpretation of network events and faults, errors in applying corrective actions, and delays in restoration of network services.

One of the most efficient solutions for resolving the problem of processing large volumes of network alarms is event correlation. First of all, event correlation reduces the network operator information load and increases the semantic content of information. Most importantly, event correlation is a basis for network fault analysis and generation of corrective actions. This paper discusses IMPACT (Intelligent Management Platform for Alarm Correlation Tasks) [JAK93, JAK94], a telecommunications network event correlation expert system developed at GTE Laboratories and fielded at different GTE network management locations. The objective of building IMPACT was two-fold: first, to develop a real-time event correlation model which allows us to capture complex network fault diagnostics and control situations; and second, to implement an efficient event correlation system which could be used to build different network management applications.

Over the last three years, IMPACT has been used in the development and fielding of several GTE network surveillance and fault diagnostics applications, including AMES (Alarm Management Expert System), for correlating alarms in a large voice/data telecommunication network, and CORAL (an alarm correlation application for GTE PCS cellular networks). IMPACT has also been used for development of an alarm correlation capability for NETOP, a system for surveillance of software application alarms such as Follow-Me-Roaming, and for Abuse Monitor, an application for monitoring calling card usage data records to discover fraudulent calls.

Event Correlation Domain

Problem

One of the most difficult problems of network management is real-time alarm surveillance and fault analysis. The original sources for most network alarms are physical faults occurring in the managed network elements (NEs). These faults can be causally related or not, i.e., they can be independent. Causal relations between faults can be represented by a fault propagation graph as shown in Figure 1. Faults which are not directly exhibited by alarms could be recognized by correlating multiple alarms. For example, correlation of alarms a2 and a3 (correlation c2) allows one to make a diagnostic decision, that the root cause for these alarms is fault f4. However, the existence of both correlations c1 and c2 leads to the recognition of fault f2 as the root cause of the faults. One fault could be caused by many faults, where the presence of only one fault is required (OR-node of the fault propagation graph) or by all faults (AND-node of the graph). The model of fault propagation and the use of alarm correlation for fault detection is described in more detail in [JAK95].

Another practical problem of network management arises when multiple subnetworks are hierarchically organized into larger networks. Examples of those networks are networks which have local, regional, area, and national levels of management. Very often those networks have their own management systems at each level (see Figure 2). One of the important information processing tasks here is generalization of information which is passed from lower levels of management to higher levels, and specialization of information is an opposite direction. Both of these information processing tasks could be solved using

event correlation. For example, during generalization, either a pattern of events could be replaced by a correlation or an event could be replaced by its superclass.

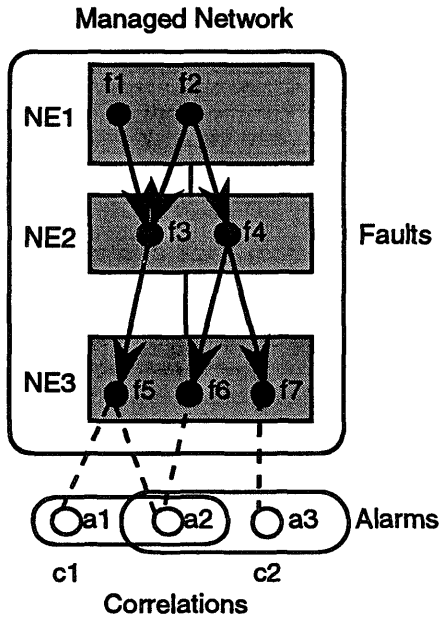

Figure 1. Fault propagation graph.

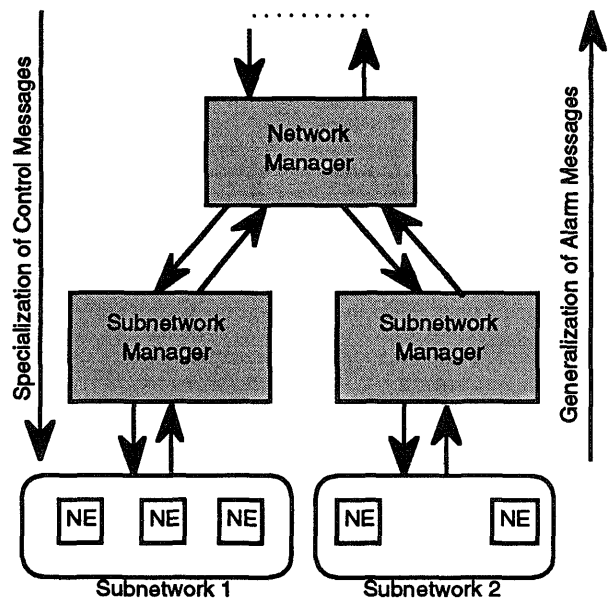

Figure 2. Hierarchical network management.

In general, event correlation supports the following network management procedures:

- Reduction of the information load presented to the network operations staff by alarm filtering, context-sensitive alarm suppression, and alarm clustering.
- Increasing the semantic content of information presented to the network operations staff by aggregation and generalization of events. Alarm aggregation and generalization are important procedures which could be used in multilevel hierarchical networks, where network information is passed from lower level network management systems to higher level network management systems.
- Real-time network fault isolation, causal fault diagnosis, and suggestion of corrective actions. During this task, a procedure may be called to access a database, run a diagnostic test procedure, generate a trouble ticket, or perform any other executable external procedure.
- Analysis of the ramification of events, prediction of network behavior, long-term correlation of historic event, and network behavior trend analysis.

Model

Our approach to event correlation uses the principles on model-based reasoning originally proposed in [DAV82] for troubleshooting electronic circuit boards. The idea of the model-based approach is to reason about the system from representation of its structure and functional behavior. We will extend this model into real-time event correlation. The structural component of the event correlation model describes the network elements and the topology (connectivity and containment relations) of the network. The behavioral representation describes the dynamic processes of event propagation and correlation. These processes are implemented using correlation rules and correlations. Each rule activates a new event (correlation), which in its turn may be used in the firing condition of the next correlation rule.

The correlated events could be alarm, status, and clear messages from network elements, messages from subnetwork management systems; operator commands; and complex events (correlations) already created during previous correlation processes. Applying event correlation rules may yield several results, including suppressing an alarm depending on the operational context, sending an informative or diagnostic message to the operator, and generating a procedure call to access a database, run a diagnostic test, or prepare a trouble ticket.

Each event correlation process has an assigned correlation time window, a maximum time interval during which the component events should happen. Arrival of any component event instantiates a new correlation time window for the correlation. This means that the correlation time window slides in time to capture new options to instantiate the correlation.

The System

Architecture

The IMPACT system architecture is determined by three modes of IMPACT operation: event correlation, correlation analysis, and application development modes. Event correlation is the IMPACT operational mode in which real-time event messages are processed and correlations are formed. Event analysis is a background process invoked at user request to answer questions, such as what is this correlation, where did it happen, what are the conditions

that caused the correlation, and what are the potential ramifications of this correlation. The event analysis process relies heavily on cross-references between different entities in the application knowledge base and in the dynamic correlation memory. Figure 3 displays sample windows of the cellular network event correlation application. The two upper windows display a portion of a cellular network with a correlation icon next to the cell UNIVERSITY, and an IMPACT Event List showing a short alarm correlation message that commercial power was lost at cell site UNIVERSITY. Two lower windows were opened at the request of an operator who wanted to learn more about the correlation. The first window displays a component structure of the correlation COMMERCIAL-POWER-LOSS, while the second window gives more detailed information about the correlation. The application development process concerns building the application knowledge base, network configuration model, geographic data files, and network visualization data. Developing an application with IMPACT essentially means building these entities. Application development is mainly an off-real-time process, however modifications to the knowledge base could be done dynamically during event correlation.

The IMPACT system architecture could be described by a data flow diagram, shown in Figure 4. When IMPACT is in event correlation mode, alarm, status, and other messages collected by the Network Data Collector are sent to the Message Processing Engine. The Message Processing Engine parses the message text and generates a corresponding message object.

The core part of the system is the Correlation Engine, an inferencing program guided by correlation rules. Successful matches of the correlation pattern in the left side of the rule may cause the evaluation of the expression in the right side, which causes either instantiation of a new correlation, clearing existing events (correlations), or, generally, performing any executable procedure. In addition to the correlation classes, correlation rules, and message objects, the Correlation Engine uses the Network Configuration Model to test network element connectivity and containment relations.

The Map Visualization System allows generation of geographic area maps with a variety of geographic objects, such as highways, roads, streets, bodies of water, geographic/administrative areas, etc. The system contains many operations over the maps, including zooming, fading, panning, scrolling, highlighting pointed objects, and distance calculation. The Network Visualization System overlays the schematic/icon representation of the network on the map.

The application development environment of IMPACT contains a variety of high-level tools, including structural knowledge-base (KB) and network configuration editors, a graphical KB editor for visualization and modification of the class hierarchies of the knowledge base, and a KB debugger, a tool for debugging and tracing rules and classes.

IMPACT has been integrated with two other systems developed at GTE, ISM 2000 and SmartAlert,. These systems perform network data collection and preliminary alarm message processing functions, such as providing the network element location code and issuing date/time stamps for messages. ISM 2000 is used for wireline networks, while SmartAlert is used for cellular networks.

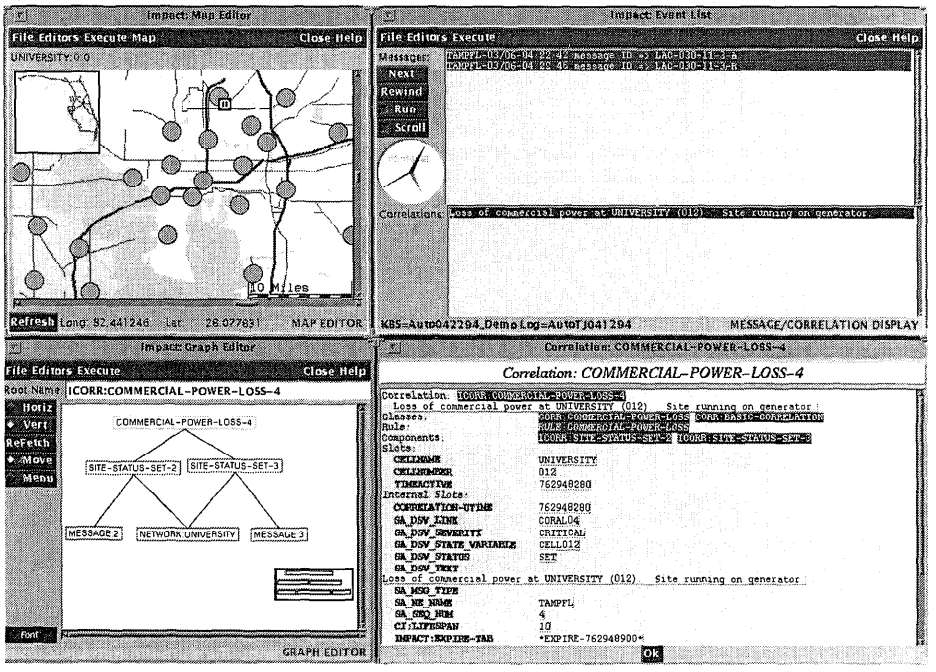

Figure 3. Cellular network alarm monitoring application.

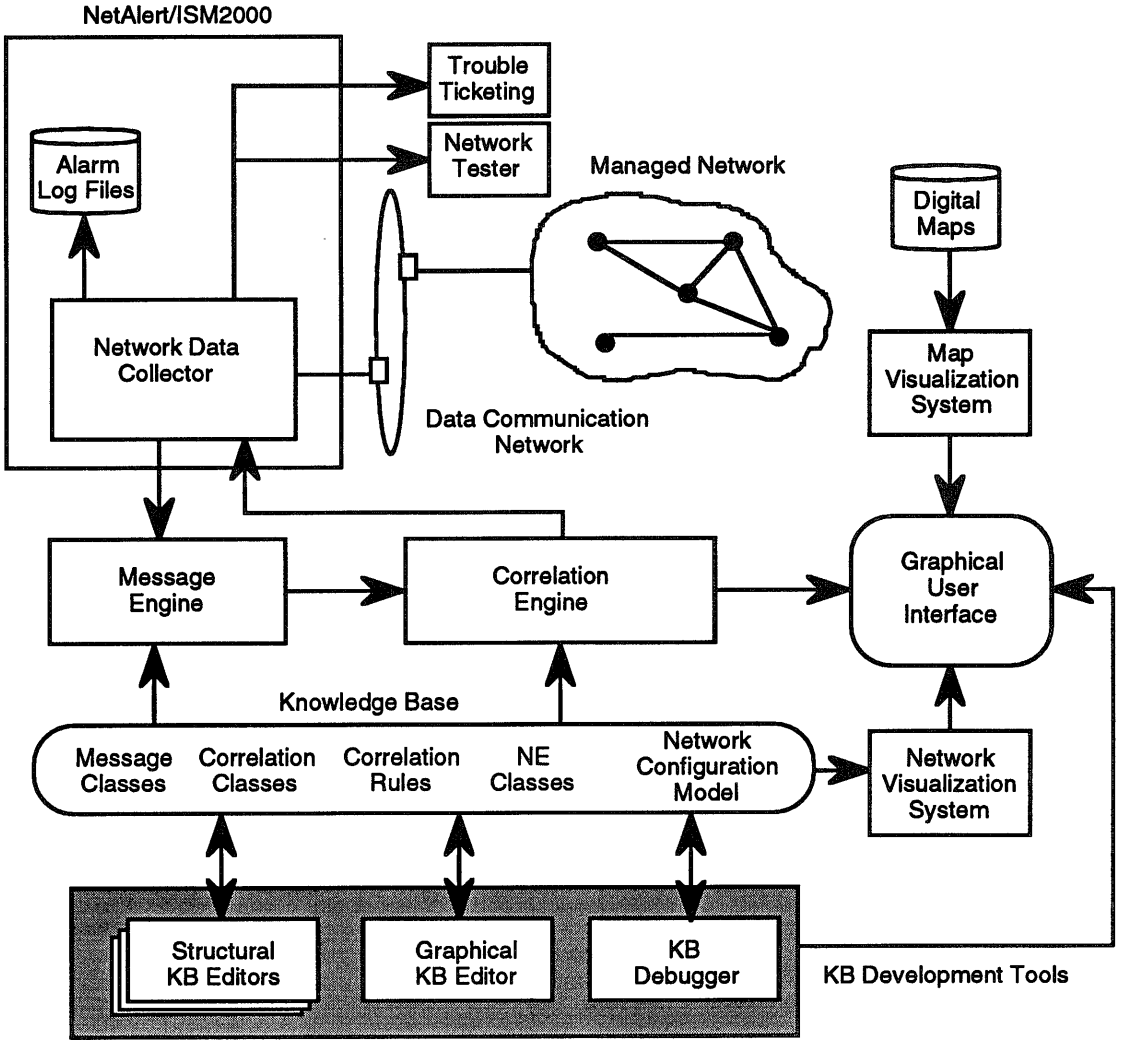

Figure 4. IMPACT architecture.

Application Knowledge Base

An event correlation application is defined by its Knowledge Base. There are five major components of the Knowledge Base: NE Classes, Message Classes, Correlation Classes, Correlation Rules, and the Network Configuration Model. Classes are organized into acyclic class hierarchies with multiple inheritance. Slots, slot values, and constraints defined in a parent node of a hierarchy are inherited by the children. The root nodes of the hierarchies are defined by IMPACT, while the rest of the classes are specific to the applications and are developed by application experts. On an implementation level, each class in the hierarchies is an object-oriented data structure.

The nodes (classes) in the NE Class Hierarchy, except the terminal nodes of the hierarchy, describe mathematical abstractions of existing physical network components, while the terminal nodes describe the network element types produced by manufacturers. For example, the root node of the NE class hierarchy may have several subclasses SWITCH, DIGITAL-CROSS-CONNECT, TRUNK, and others. The class SWITCH may have a subclass CELLULAR-SWITCH, which, in turn, may be a parent for a terminal node which corresponds to a particular cellular switch.

Message classes play a role in message processing similar to the role rules play in inferencing. Developing a message class hierarchy for messages generated by a specific network element means creating a parse tree for the Message Processing Engine of IMPACT to parse these messages. Nodes (classes) in the Message Class Hierarchy represent a declarative encapsulation of a message parsing procedure. This approach of message processing is described in [JAK93].

A correlation class is a generalized description of the state of the network based on the interpretation of network events. The conditions under which correlations are asserted are described in the correlation rules. Each assertion creates an instance of a correlation class. A correlation class contains components, a message template, and parameters (slots). The components may be NEs, alarm messages, or other correlations. Correlation components are

used to pass information from a correlation rule to the asserted correlation. Parameters provide information about a correlation to higher level correlations, of which it may be a component.

An example of a correlation and a correlation rule is given in Figure 5. Correlation rule BAD-CARD-CORRELATION-RULE-1 states: if physical ports *?near-port* and *?far-port* belong to two digital cross-connect systems, respectively, *?near-DEXCS* and *?far-DEXCS*, and these ports are connected by a T1 trunk, and Yellow Carrier Group Alarm *?yellow-msg* is reported from *?far-port*, and Red Carrier Group Alarm *?red-msg* is reported from *?near-port*, then assert BAD-CARD-CORRELATION. After matching the rule conditions, *?near-DEXCS* and *?far-DEXCS* are bound to particular NEs. These NEs are provided as components to BAD-CARD-CORRELATION. Correlation BAD-CARD-CORRELATION contains two components, a digital cross-connect system DEXCS-CLASS, and a physical port PHYSICAL-PORT-CLASS. During assertion, a correlation rule assigns values to the slots CLLI (a universal code which identifies the location of the equipment) and PORT-NUMBER. These values are used by the message template and asserted into the slots DEXCS-ID and PORT-NUMBER. Variable names are identified by a leading question mark.

Network Configuration Model

The Network Configuration Model is a description of an actual physical or logical network. The connectivity model is built by instantiating terminal NE classes from the NE class hierarchy and connecting them according to the network configuration. During the network element class instantiation process, IMPACT enforces the constraints defined in the class specification. For example, the user cannot make connections which violate the physical behavior of the connected elements, or leave the values for required parameters of the elements unspecified.

Implementation

The IMPACT software architecture is shown in Figure 6. One of the chief requirements of IMPACT implementation was high real-time message processing and event correlation performance, 0.2–0.5 messages/second average and 15–30 messages/second burst parsed and correlated. This performance requirement was reached on the SGI Indigo R4400SC. The performance requirement also affected the IMPACT architecture: the Event Correlation Engine was built using CLIPS, which employs the RETE algorithm for incremental rule pattern matching. The initial version of IMPACT was implemented in ART-

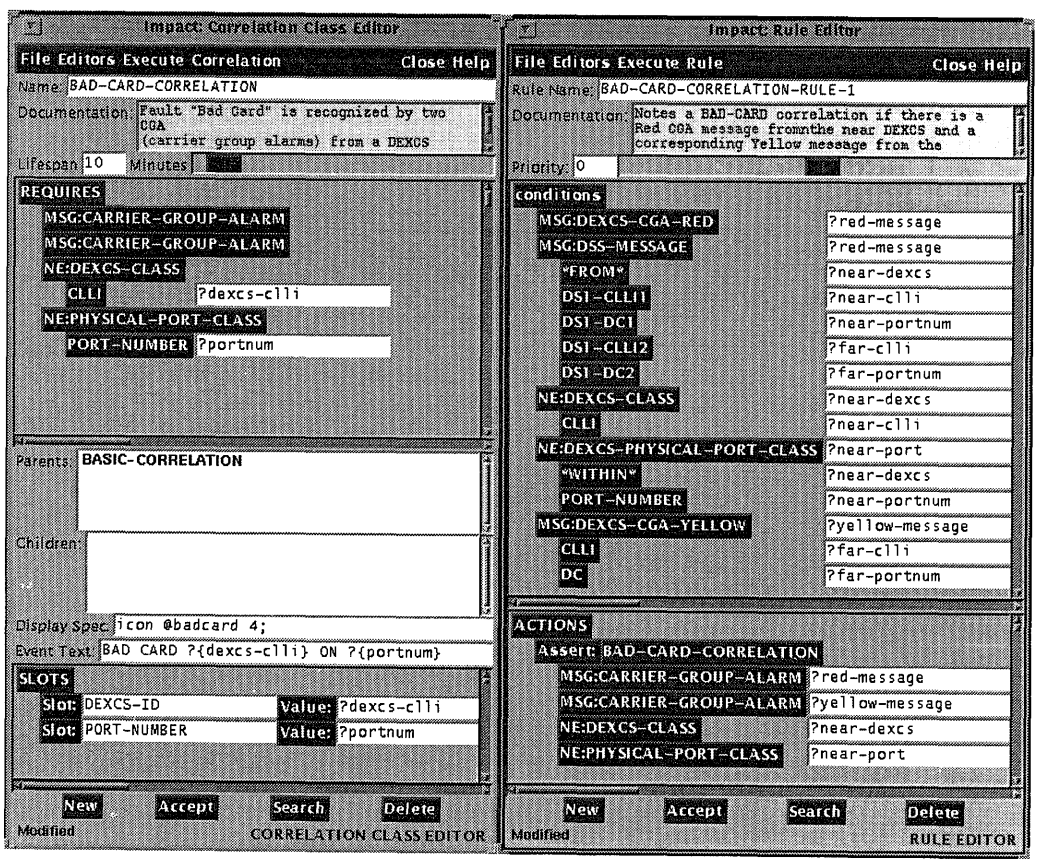

Figure 5. Correlation BAD-CARD-CORRELATION.

IM, which has a rule language and RETE network similar to CLIPS, however an additional requirement to keep IMPACT's cost low led to the reimplementation of the Event Correlation Engine. We didn't find COOL, the object system of CLIPS, flexible enough, and implemented our own object system based on hash-tables.

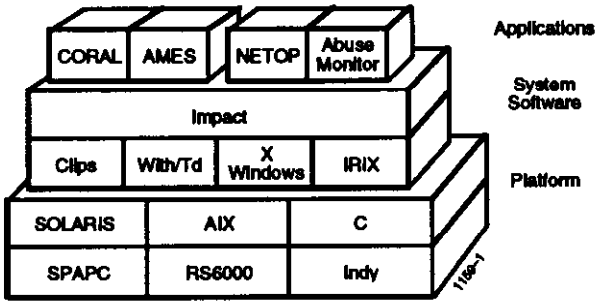

Figure 6. IMPACT software architecture.

The rest of the IMPACT code was implemented in C and Tcl/Tk.. The time-critical portions of the Message Processing Engine as well of the drawing routines of the visualization systems were programmed in C and X Windows. Most of the graphical user interface, knowledge base editors, and data visualization programs were developed in Tcl/Tk. Building IMPACT's GUI in Tcl/Tk and not in Motif was a controversial decision, driven by a need to get initial implementation fielded very quickly.

The overall IMPACT implementation effort is about 20K lines of C, 30K lines of Tcl/Tk, and 25K lines of CLIPS. It took 2.5 man-years to develop the IMPACT shell (not counting development of IMPACT-based application systems).

Application Development and Deployment

Deployment History

IMPACT, start of the project	December 1990
IMPACT, first release	January 1992
AMES, first deployment	February 1992
CORAL, first deployment	December 1993
IMPACT, second release	February 1994
NETOP deployment	April 1994
AMES, second deployment	May 1994
IMX2500 alarm monitoring	November 1994
Abuse Monitoring deployment	November 1994

Experiences and Results of Application Development

We will describe briefly application development technology using the CORAL system as an example. CORAL, GTE's cellular network alarm correlation/fault management system, was developed and deployed in 8 months. The knowledge acquisition sessions with the network operations staff took place during 4 months. Altogether, 6 knowledge acquisition sessions were conducted with 3 teams of experts (2–3 experts in a team) from different cellular network management regions. The development of the correlations, correlation rules, network element classes, network configuration model, and message classes was conducted by GTE TSI, a network management software development arm of GTE, with a medium level of assistance from GTE Labs. After receiving two days of training, the network management personnel were able to modify the network configuration model, correlations, and correlation rules.

Maintenance and Support

Maintenance and support of IMPACT within and outside GTE is provided by GTE TSI. IMPACT has been included in the GTE TSI product line.

Benefits

To explore the benefits provided by IMPACT, the following should be considered:

1. IMPACT introduces a standard uniform approach to the development of multiple different event correlation applications, which share common knowledge representation formalism, common GUI and knowledge engineering tools, and common geographical information processing and network visualization procedures.
2. The application development and correlation analysis environments provide a way for network management experts and operators to think about the problem.
3. Reduce the field service and network maintenance cost. Placing sophisticated network surveillance and fault diagnostic procedures in the hands of network operations personnel at GTE network management centers allows timely location of network faults and required corrective actions.
4. Reduce training requirements. IMPACT has allowed new employees to perform network management operations with higher confidence due to direct participation of the network operations personnel in knowledge base development, modification, and maintenance.

Conclusions and Lessons Learned

Development of IMPACT itself and event correlations applications using IMPACT has strengthened our belief that an important evolution path of expert systems shells is in the direction of dedicated expert system shells, which are used to solve specific engineering, analysis, monitoring, diagnostics, and other problems. Such shells possess general domain models (domain constraints), which could be used during the knowledge acquisition process to ensure correctness and consistency of entered knowledge.

We learned that visual information presentation tools play an important role in creating network management "information space," which significantly enhances the way network management personnel understand network events and undertake required actions. These tools include geographic information systems, network and network event visualization procedures, as well as graphical tools of presenting and editing hierarchical knowledge base entities.

Very high real-time performance requirements combined with sophisticated event correlation algorithms may pose conflicting situations not solvable with current

network management platforms, i.e., platforms in the price range of $30–$40K. One solution is to use hierarchically organized event correlation applications corresponding to the geographical or functional distribution of the applications.

Despite our initial hesitation to directly include network operators in the knowledge acquisition process, we learned quickly that such a move increased the operators' confidence level in the system. On one occasion though, a domain expert practically refused to cooperate in the knowledge acquisition session, fearing that the system deployment will be a threat to job security.

Acknowledgments

The success of IMPACT is the result of the contributions of many organizations and individuals. The authors thank Alan Lemmon and Robert Weihmayer from GTE Laboratories who took part in several stages of IMPACT's development. Special thanks go to our development partners Shrinivas Kumar, Joe Speed, Mark Hewitt, and Thomas Hall from GTE TSI, and Don Reeves, Shawn Waters, and Steve Owens from GTE NMO. Finally, we thank the management and a large group of network domain experts and operators from GTE Mobilnet and Contel Cellular.

References

[DAV82] Davis, R., Shrobe, H., and Hamscher, W. Diagnosis based on description of structure and function. Proceedings of the 1982 National Conference on Artificial Intelligence, Pittsburgh, PA, pp. 137–142.

[GIA93] Giarratano, J. CLIPS user's guide. NASA LBJ Space Center, Software Technology Branch, 1993.

[JAK93] Jakobson, G., and Weissman, M. Alarm correlation. *IEEE Network*, 7 (6), pp. 52–59, 1993.

[JAK94] Jakobson, G., Weihmayer, R., and Weissman, M. A domain-oriented expert system shell for telecommunication network alarm correlation. In *Network Management and Control*, Volume II, (editor M. Malek), Plenum Press, New York, NY, 1994

[JAK95] Jakobson, G., and Weissman, M. Real-Time Telecommunication Network Management: Extending Event Correlation with Temporal Constraints, Proceedings of the Fourth IFIP/IEEE International Symposium on Integrated Network Management, May, 1995, Santa Barbara, CA.

[OUS94] Ousterhout, J., Tcl and the Tk Toolkit, Addison - Wesley Publishing Company, 1994.