Vittorio Gelmi

Preventive Maintenance: Using Data Mining Systems

The collection of alarms from a network is beneficial when a failure in the system is permanent and it is therefore possible to identify the failure itself. However, often temporary failures are reported by the alarms, but the network soon comes back to working conditions. These chronic failures can be understood only by analysing possible correlation with other events, on the basis of synchronicity, logical sequence, etc. It is impossible to define a priori how to search properly for this correlation, but an artificial intelligence system can be used methodically to help the operator. This paper describes how an intelligent data mining system can be designed and its main features.

As many systems control different elements of the network and different databases can hold part of the overall information needed to identify the correlation, the intelligent data mining system must be equipped to interact with these external systems.To optimise the process (permitting parallel activities on different platforms) this interaction is carried out by mobile agents that are sent to the different systems, locally collect and elaborate the necessary information and come back to the intelligent data mining system bringing the required results.

Introduction

Fault analysis in large telecommunications networks usually involves handling a very large amount of alarms data from different subsystems. When a fault becomes stable, usually a correlation among detected alarms can be detected and repair of the fault is feasible. If a fault appears just for a moment, disappears later and (often after hours or days) comes back again, the fault analysis subsystem often cannot isolate the proper cause of this *chronic* fault[3]. A large number of alarm records crowds the database which are not usable. The operator tries unsuccesfully to resolve the problem, analysing alarms in many ways: istograms, time correlation etc. However, because of the large number and variety of subsystems providing data, a systematic analysis is quite difficult.

Looking at telecommunications applications in network management and fault monitoring, some common requirements can be identified.

Vittorio Gelmi:
Sirti S.p.A. Systems Division Cassina De Pecchi (MI 20060 ITALY).
E-mail: V.Gelmi@sirti.it

Generally applications related to fault analysis must:

- interface heterogeneous sources of data (databases for network configuration, alarms, SNMP agent, custom applications etc.),
- collect data from remote hosts with different procedures,
- drive the user (not just 'help') to perform a complex task, and
- coordinate many synchronous actions/tasks on databases, to obtain reports and information.

To support efficiently the localisation of equipment subject to chronic failure and to isolate area(s) of degradation (to enable preventive maintenance to be carried out), we tested a prototype system which provides:

- intelligent assistance to the operator in searching for the cause of problems, and
- a datamining facility to dig into alarm subsystems to correlate and isolate cause(s) of faults.

Elements of the Problem

The analysis of chronic faults is based on some elements of classification of fault type as described in the following sections.

Fault classification

Starting from the basics, a fault in a telecommunications network causes alarms to rise. Faults can be classified in three types:

- 'Steady'**—**fault is persistent, and its effects do not change with time. A relatively small number of alarm records is collected (indeed only one 'start time' and 'end time' record). For example, a break in a (not backed-up) power-supply subsystem of a transmission system causes (usually) a steady fault. A steady fault is often related to a hardware fault (and generally can be automatically isolated).
- 'Transient'— fault produces symptoms which change with time. A number of causes can produce transient alarms in telecommunications networks; for example, thunderbolts (if not actually causing a fault to equipment) can temporarly blind some links or increase trasmission line errors dramatically. Usually (but not necessarily) a steady fault is announced by a transient phase during which a large number of alarm records are collected (sometime expressively defined *ripples*). After this phase, all is silent and steady state is reached. Hardware faults are (often) sources of this kind of transient faults.
- 'Chronic'—fault initially appears to have symptoms similar to a transient fault; after some time symptoms disappear too. Later (and usually without visible correlation with the previous event) alarms appear again and continue in an impredictable (usually not periodic) sequence. For example, if a ground connection of a transmission equipment's shield has high resistance, the equipment may be

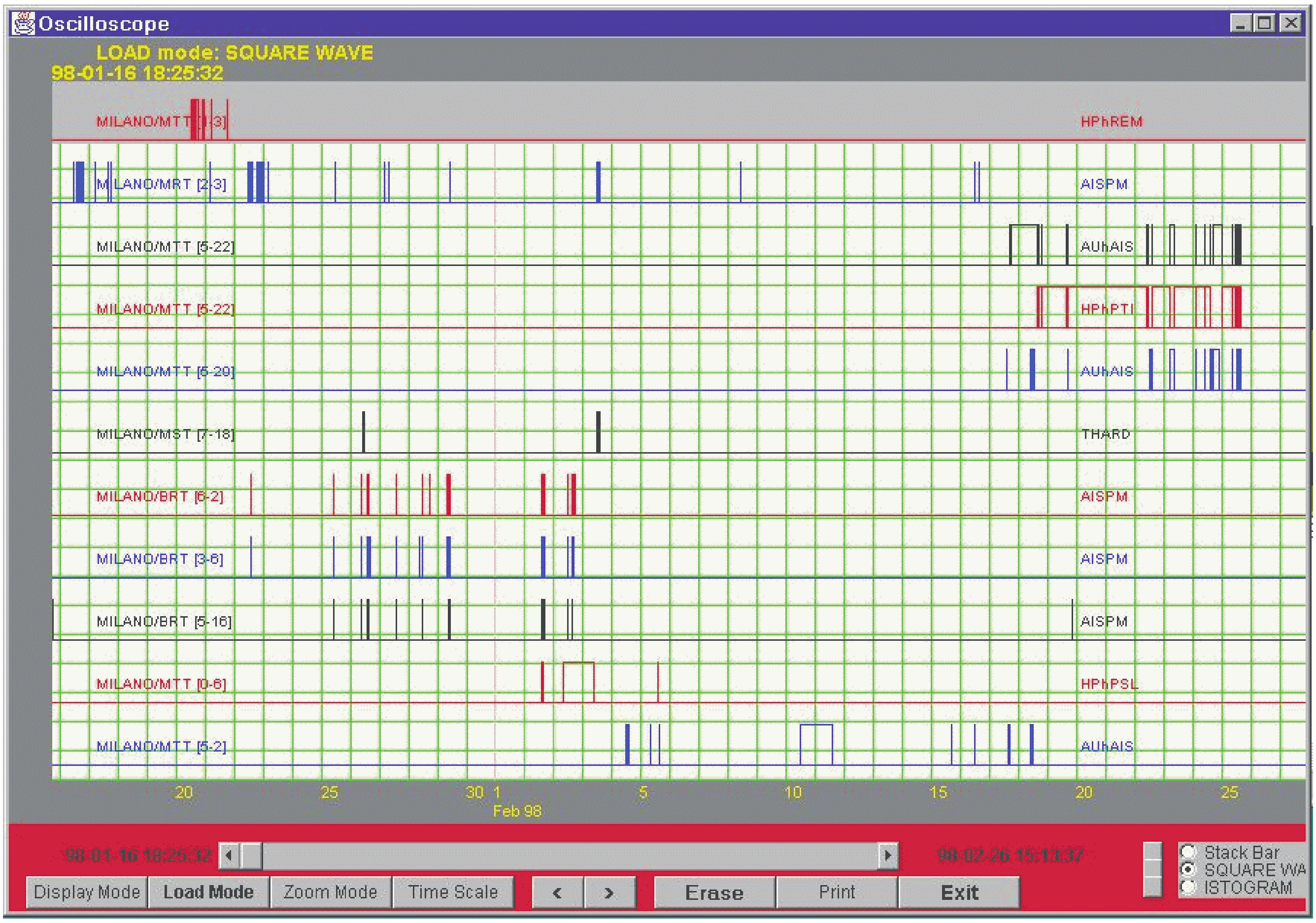


Figure 1—Isochronous alarm patterns in PDH and SDH links. Some isochronism is justified by interconnections

exposed to random electromagnetic influences from other equipments close to it. In this situation alarms are created as the electromagnetic influence exceeds (randomly) some level. Another example is if some hardware is (improperly) working at its thermal limit, as the external temperature grows some reversible malfunctions can appear/disappear creating alarms in the transmission stream.

Alarms classifications

Alarms too can be divided into two categories:

- *source* or *primary* originated by the faulty equipment; for example, a power supply fault creates primary alarms; and
- *derived* or *secondary* risen as a consequence of electrical connections among the elements of a telecommunications system; for example, a transmitter fault in a transmission-chain propagates derived signal alarms along the entire chain.

Maintenance issues

Chronic faults generally are not properly isolated and removed. This may cause progressive degradation of some areas of the telecommunications network thus increasing unavailability time.

The target should be to isolate chronic faults, properly investigate their cause, locate equipment and repair it before faults can reappear. Removing causes is more than repairing a faulty device. Often causes are not only damaged network objects (NOs) (in the sense that they are in network databases or management information bases (MIBs)). Causes of chronic alarms may not be associated with the telecommunications equipment†; for example, poor installations, etc. Experience has demonstrated that the idea of finding in databases or MIBs one or more device(s) responsible for the problem is erroneous. It is not adequate to limit the search only to alarms databases, layout databases and in general all that describes the network: often the search will stop without a clear identification of causes, or a (possibly small) set of NOs is indicated. The human operator (or a set of heuristics about possible causes or classes of causes derived from experience) will complete the search including proper 'external' causes or relations.

Chronic Faults Search Methodology

A prototype has been built to investigate plesiochronous digital hierarchy (PDH) and synchronous digital hierarchy (SDH) circuits and links. For these NOs a way to investigate chronic faults is to analyse signal alarms and some quality indicators*. (See Figure 1.) Signal-alarm records are explored to identify equipment with sufficient grade of isochronicity in alarm duration and start time. Usually alarms patterns do not exactly match: an algorithm used to compare each pattern with others needs, to work well, to be improved with some heuristics.

Derived alarms and (unknown) primary alarms are expected to exhibit very similar patterns. During analysis, network objects which report isochronous alarms are grouped in disjointed *isochronous sets*.

† For example, in a real case signals from a radar installation (not telecommunications related) some miles distant from the telecommunications office has been found to be the cause of chronic faults.

* For example: unavailable seconds (UAS), severely errored seconds (SES) etc. counted hourly.

An NO_A can generally propagate a derived alarm to NO_B due to reciprocal interactions: we collect this information about interactions as relationships like $Pr(A,NO_A,NO_B)$ where A is the alarm type. Elements which support Pr() are grouped into *connected* sets. Pr() relationships can be discovered from the layout of circuits, interconnections, databases and rules specific for each NO. Note that for this analysis we really do not need to know which is the primary alarmed NO in the isochronous set.

Each isochronous set is then decomposed into a union of connected sets. Intersections among connected sets in an isochronous set are NO investigated for chronicity[3]. Indeed field experiments demonstrate that often the network operator can easily locate the faulty device at this point. Some aspects (not described here) make the problem more complex; for example, some external events like a substitution of a piece of equipment, the removal of the cause of a chronic alarm internally to the interval of analysis (not always recognisable from telecommunications databases) need to be considered.

Analysing link and circuit quality parameters

Some quality measures on links and circuits have been proved to work well in locating chronic faults. Data collected on an hourly base in SDH (155 Mbit/s) links or PDH (140 Mbit/s) links or circuits about SES, ES etc. can be treated almost in the same way as alarms. Alarms are merely on-off events (that is, square waves), but SES, ES measures appear as istograms placed along the time axis. Some differences occur in recognising isochronous sets but basically the idea is the same. Despite the poor temporal resolution (data are collected usually on an hourly basis) analysis appears to be much more sensitive in warning about degraded network areas and NOs subject to chronic fault (Figure 2).

Assistance to the Operator

The process for discovering chronic fault in long-distance circuits is a sequence of analysis mainly guided by heuristics and operator's knowledge of the network. The operator needs to be helped but, generally, is annoyed if systems proceed autonomously impeding any deviation from his/her reasoning scheme. Sometimes an event (alarm, layout, relation etc.) captures the attention of the operator, at this point she/he needs to interrupt the system to follow its reasoning sequence.

The operator is very annoyed too when requested to inform the system about (partial) conclusions of his/her analysis. Partial conclusions have been reached just interacting (by user interface with istograms, selections using mouse etc.) with the system itself: the system should 'remember' all that it supplied to the operator.

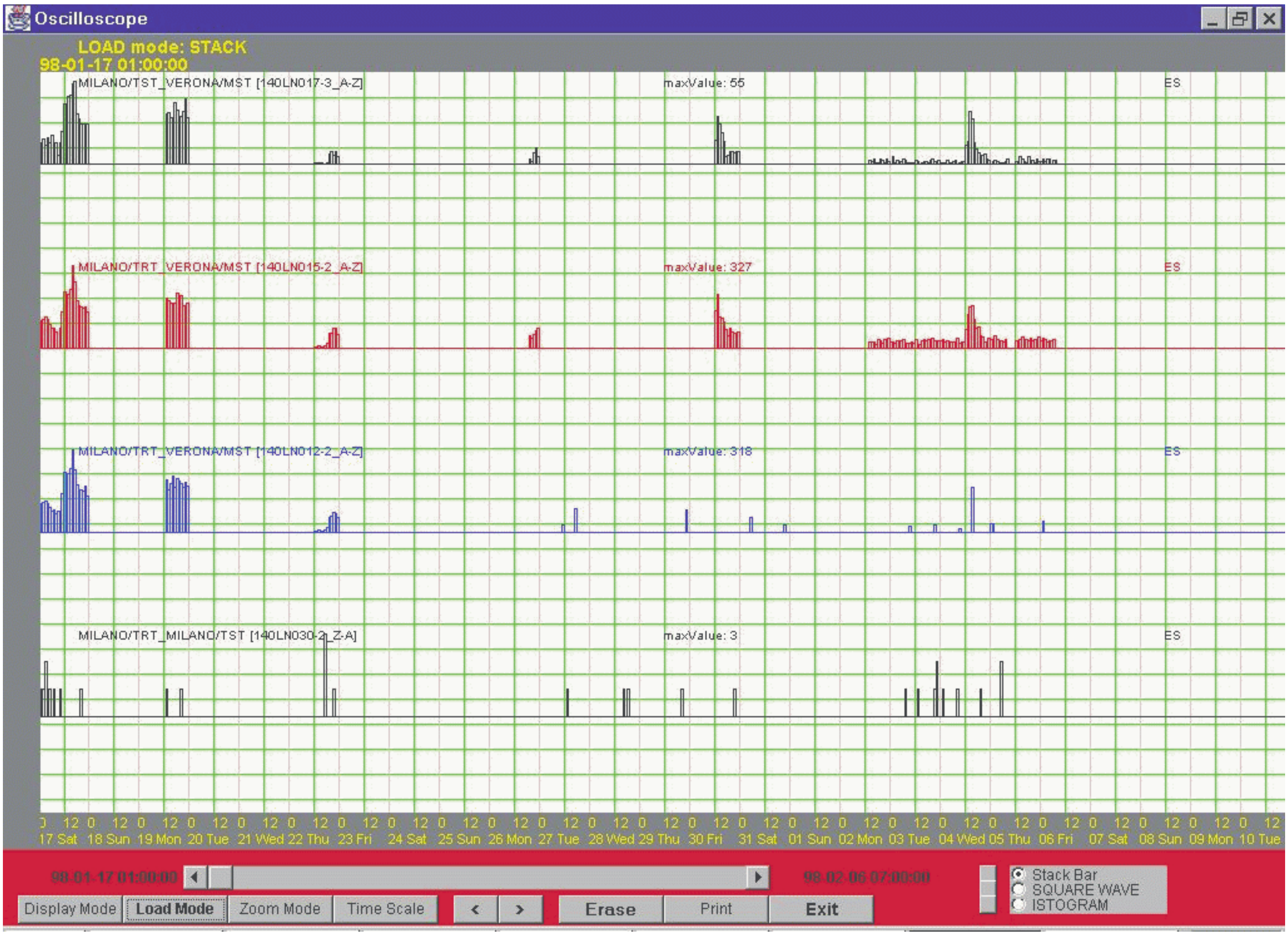
To satisfy these requirements, a three-part architecture has been used including:

- user interface (the client),
- application server, and
- an assistant.

Mining System Structure

Due to the great diversity of telecommunications subsystems from which data are to be collected and analysed, the overall architecture of the application is structured in a collection of modular plug-in

Figure 2—ES measures reveal isochronous patterns. Disturbed links are all carried by a 565 Mbit/s mux



LOAD mode: STACK
98-01-17 01:00:00

Oscilloscope
MILANO/TST_VERONA/MST [140LN017-3_A-Z]
maxValue: 55
ES
MILANO/TRT_VERONA/MST [140LN015-2_A-Z]
maxValue: 327
ES
MILANO/TRT_VERONA/MST [140LN012-2_A-Z]
maxValue: 318
ES
MILANO/TRT_MILANO/TST [140LN030-2_Z-A]
maxValue: 3
ES
0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12 0 12
17 Sat 18 Sun 19 Mon 20 Tue 21 Wed 22 Thu 23 Fri 24 Sat 25 Sun 26 Mon 27 Tue 28 Wed 29 Thu 30 Fri 31 Sat 01 Sun 02 Mon 03 Tue 04 Wed 05 Thu 06 Fri 07 Sat 08 Sun 09 Mon 10 Tue
98-01-17 01:00:00
98-02-06 07:00:00
Display Mode
Load Mode
Zoom Mode
Time Scale
<
>
Erase
Print
Exit
● Stack Bar
○ SQUARE WAVE
○ ISTOGRAM

components over a minimal background structure.

To facilitate interaction with remote applications, a mobile agent structure has been tested[5, 6, 7].

The application has some core components:

- *Mobile code server(s) (MCS)*—one or more on each host that supports applications for alarm collection, circuits or equipment interconnection, etc. This is a Java module responsible for receiving the mobile agent code and running it.
- *Mobile agents*—a module that interacts with remote systems and extracts relations, data etc. as requested.
- *Assistant*—a module including a knowledge base devoted to helping the operator or acting autonomously in a data-mining sequence.
- *User interface.*
- *Portable instruments*—modules with some defined interface usable by mobile agents or a static application to perform a specialised task. Portable instruments could be provided by MCS. Each agent has various portable instruments to easily interface local agents (that is, SNMP agents or custom agents) or the data source and to perform its task. Each portable instrument supports a common interface and can be added to the agent just at the start time and destroyed as used†. Portable instruments can be loaded at the arrival time of the incoming agent as convenient. Portable instruments can also be sent after the arrival of the agent in-loco as a consequence of a request message sent to the control centre.

Functionality

Operations are various and generally not in a fixed sequence. Generally an operator starts by investigating some suspect events. Either at the operator's request or the assistant's, agents are launched to the appropriate host to perform the requested job. As the agent returns, usually it has the answers. A search can be driven by the operator, automatically or jointly.

Mobile Agents

The system should be capable of collecting data from different sources, preprocessing different formats and processing data to support user's (or the assistant's) requirements. To perform this task we used a framework to build mobile agents.

A mobile agent is a software code that can be moved among different hosts (Figure 3). The agent's binary code is platform independent. The same code moves from one host to the other and there performs its task.

Instead of creating a monolithic structure to obtain data from remote databases, process them and extract knowledge, autonomous mobile agents are sent by the application server (or the assistant) to the remote host with a workplan to be performed in-loco. Mobile agents in our prototype are built in Java.

Some general benefits in using mobile agents

- *Efficiency savings* CPU run time is not of the application, but of the agent's host.
- *Space savings* Each agent resides on one node at a time. Mobile agents carry the functionality with them which gives a high degree of customisation.
- *Reduction of network bandwidth / network traffic requirements* The transfer of the mobile agents along the network creates less traffic than transferring the data. The agent is not connected to the application and works standalone.
- *Heterogeneous environment and applications glue* Every system can be targeted by the agent at the price of a small interface and no change to agent structure and code. This results in fast and cheap customising of new applications.
- *Easy software upgrades* The mobile agent can be replaced without change of remaining application's code.

Agents communication

Each agent in the network performs a message-passing facility to exchange data and requests. This structure is used to develop a protocol for interaction from the user interface to the server(s) and assistant.

Database access facility

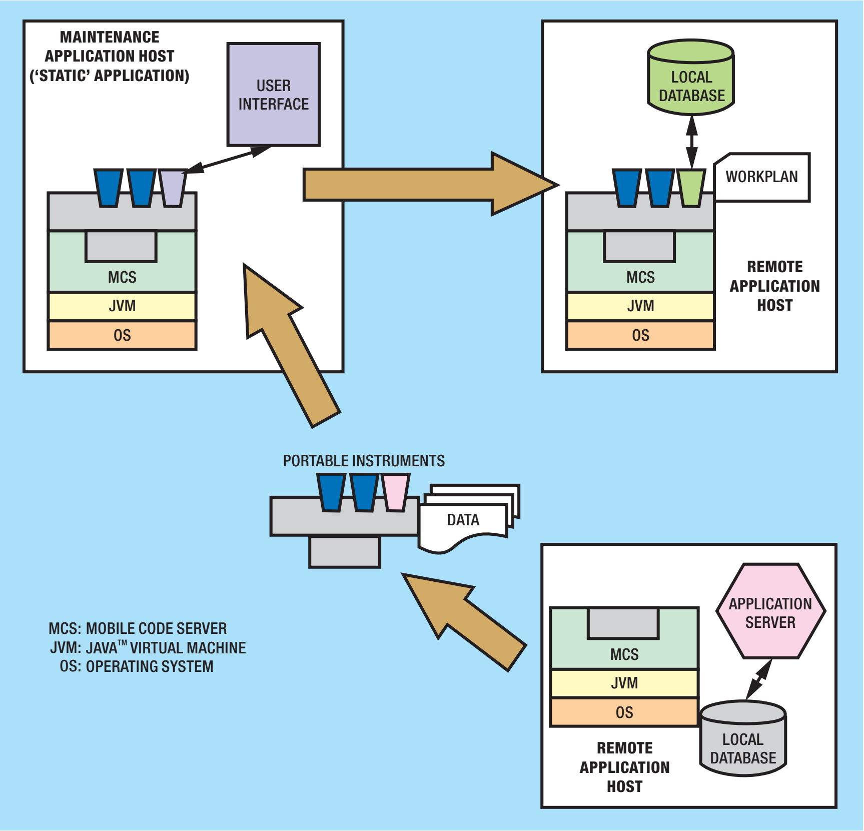
Due to the Java structure of the code of agents, accessing local databases is performed via JDBC and ODBC classes added as portable instruments.

The SNMPv1/2 stack for MIB access is added as a portable instrument too. CORBA (IIOP) is planned.

Assistant Integration

The user interface for the operator is connected to the application server

Figure 3—Mobile agents perform the task of discovering relations and data in remote applications



† Thus reducing network bandwidth requested in migration of the agent.

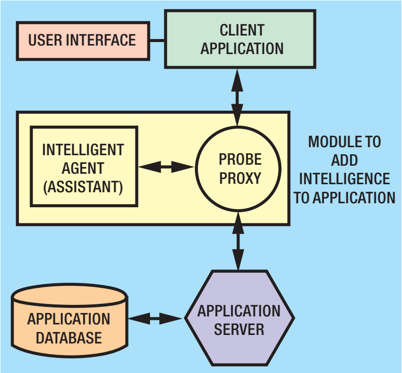


Figure 4—Architecture for assistant insertions

by means of a *probe proxy* node supporting a special facility: each message to/from the server is copied to a third party called an *assistant* (Figure 4). The assistant can send and receive messages to the client or server (without any duplication). In reality, the user interface and application servers are mobile agents 'fixed' in an MCS located in their host.

Usually the assistant does not disturb the interaction between the operator and the application server: it just looks at requests and answers and silently—using its knowledge base—obtains some conclusions. In effect, the operator and the assistant expert systems have the same goal(s) (mainly to identify chronic faults in the network). Each action made by the operator (query, selection of an NO etc.) using the user interface is encoded as a message for the application server and is a significant message too for the knowledge base of the assistant. The operator interacts with some tools to analyse the system being investigated; for example, to ask for:

- alarms density istogram, and
- most alarmed devices in a time interval etc.

As the operator needs help, he/she can perform many different requests to the assistant like:

- obtain a synthesis about facts/ conclusions collected,
- request the assistant to continue autonomously the search operation (and eventually interrupt it as necessary), and
- request the assistant to suggest what should be the best action to continue investigation.

If enabled, the assistant can also spontaneously send messages to the operator.

Obtaining Knowledge about the Network

Mobile agents using specific portable instruments are used to extract from the network layout relations Pr(A,x,y) and some configuration information from resident SNMP agents. Interfaces to custom data sources are also built as portable instruments. As the assistant attempts to reach a sub-goal in its reasoning chain, either it fires locally executed rules or activates a specific mobile agent giving it a workplan for a job to perform on the remote host. A set of portable instruments is supplied to the agent to facilitate its job. The workplan thus is a piece of knowledge code supplied by the assistant. Portable instruments interface with the local application and carry out the work. The result of the action of portable instruments is understood at a level of the knowledge base as a sub-goal realised.

Experienced Result and Future Effort

Currently some tests have been perfomed using a prototype to collect signal alarms and quality measures in PDH and SDH long-distance networks used as a test-bed. Isochronism analysis has been demonstrated to be a valid method, but discovering Pr() relations is quite application specific. User–assistant interactions are well focused but the assistant's knowledge-base tends to be applications specific too. The autonomous action of the assistant unfortunately does not always reach appreciable results in locating chronic faults. The operator often drives the assistant in the first phases of the investigation, gains vantage from the assistant's synthesis until a small set of suspicious devices is located; the final step is often done by the operator.

Actions on local databases are application specific but relations obtained (the knowledge extracted) tend to be quite uniform over many different network structures. Future effort will be identifying some ontologies (see Reference 4 as an example but not in the telecommunications field) to define uniformly actions and objects on which operator and assistant can work and accordingly mobile agents can obtain knowledge.

References

1 Sasisekharan, Raguram; Kao Hsu, Yung; and Simen, David. SCOUT: An Approach to Automating Diagnoses of Faults in Large Scale Networks. Proc. IEEE Globecom 93, IEEE Com. Soc., Pisctaway, N. J., 1993, pp. 212–216.

2 Corn, P. *et al.* An Autonomous Distributed Expert System for Switched Network Maintenance. Proc. IEEE Globecom 88, IEEE Comm. Soc. Pisctaway N. J., 1988, pp. 1530–1537.

3 Sasisekharan, Raguram; Seshadri, V.; and Weiss, Sholom M. Data Mining and Forecasting in Large-Scale Telecommunication Networks. *IEEE Expert Intelligent Systems and their Applications*, Feb. 1996, pp. 37–43.

4 Fernando Lopez, Marian; Gomez-Perez, Asuncion, *et al.* Building a Chemical Onthology Using Methontology and the Ontology Design Environment. *IEEE Expert Intelligent Systems and their Applications*, Jan./Feb. 1999, pp. 37–46.

5 Susilo, Gatot; Bieszczad, Andrzej; and Pagurek, Bernard. Infrastructure for Advanced Network Management based on Mobile Code. Systems and Computer Engineering, Carleton University, 1125 Colonel by Drive, Ottawa, Ontario, Canada K1S 5B6.

6 Bieszczad, Andrzej. Advanced Network Management in the Network Management Perpetuum Mobile Procura Project. SCE Technical Report SCE-97-07. http://www.sce.carleton.ca/netmanage/perpetuum.shtml#Publications

7 Mobile Code Toolkit. Carleton University Ottawa. http://www.sce.carleton.ca/netmanage/mctoolkit

Biography

Vittorio Gelmi
Sirti S.p.A.

Vittorio Gelmi is currently at Sirti S.p.A. He has been working on appling techniques from artificial intelligence to problems in network planning. His interest include approaches in automatic network design, and automatic fault diagnosis.