School of Computer Science and Telecommunications

Department of Signal Theory, Telematics and Communications

University of Granada

Ph.D. Thesis Dissertation:

Business Driven Alerts Correlation in Network Management Environments

Written by:

Saeed A. M. Salah

Supervised by:

Dr. Gabriel Maciá Fernández

Dr. Jesús E. Díaz Verdejo

Granada, 2015

Editorial: Universidad de Granada. Tesis Doctorales
Autor: Saeed A. M. Salah
ISBN: 978-84-9125-125-5
URI: http://hdl.handle.net/10481/40240

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Departamento de Teoría de la Señal, Telemática y Comunicaciones

Universidad de Granada

TESIS DOCTORAL

Correlación de alertas con visión de negocio en entornos de gestión de red

Realizada por:

Saeed A. M. Salah

Dirigida por:

Dr. Gabriel Maciá Fernández

Dr. Jesús E. Díaz Verdejo

Granada, 2015

El doctorando D. Saeed A. M. Salah y los directores de la tesis Dr. D. Gabriel Maciá Fernández y Dr. D. Jesús E. Díaz Verdejo, Profesores Titular y Catedrático de Universidad, respectivamente, del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada

GARANTIZAMOS AL FIRMAR ESTA TESIS DOCTORAL

que el trabajo ha sido realizado por el doctorando bajo nuestra dirección y, hasta donde nuestro conocimiento alcanza, en la realización del trabajo se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Granada, a 19 de Marzo de 2015

Directores de la Tesis

Dr. D. Gabriel Maciá Fernández

Dr. D. Jesús E. Díaz Verdejo

Doctorando

D. Saeed A. M. Salah

Dedications

First and foremost. I dedicate this work to the soul of my father. I will never forget his words of inspiration and encouragement in pursuit of excellence that made me face any problem. I ask Allah (God) to forgive him and grant him his highest paradise.

I am forever indebted to the greatest woman in my life, my beloved mother. Her deep faith, her prayers and supreme trust are always the most efficient motivations to accomplish my ultimate goal. No word can describe what you have done for me. Thank you for your selfless and endless love.

To my beloved, my wife, Yasmeen ♡, her prayers, patience, hard work, energy, dedication, enthusiasm, passion, support, and, most of all, her love and affection; to these I owe where I stand today.

Finally, my lovely kinds, Masa and Mohammad, whom have decorated my life and made it full of happiness and joy.

Acknowledgements

Foremost, my deepest thanks and gratitude go to Allah (God) for providing me with the wisdom, health, means, strength and perseverance to undertake this research task and complete this journey. Without His will and generosity, none of this or any other accomplishment would have been possible.

I would like to express my deepest appreciation and first debt of gratitude to my supervisors Professors Gabriel and Jesús. Their patience, encouragement, frequent insights, and immense knowledge were key motivations throughout my PhD. I have enjoyed the opportunity to watch and learn from their knowledge and experience. I am very grateful to them for putting at my disposal every facility that they had which I needed during the course of study. I have truly learned a lot from them during my doctorate. I am very proud of what we have achieved together. A well-deserved huge "thank you" to you all.

I would like to acknowledge the most important persons in my life, my wife Yasmeen, and my children Masa and Mohammad, whose love and encouragement have been a constant source of strength and inspiration and allowed me to finish this journey.

I owe my sincerest thanks to my family and relatives for their prayers, assistance and encouragement throughout my study. I think words can never express enough how grateful I am to them. I can only say a warmest thanks to my mother for her prayers, patience and untiring support in every way during my long absence from the family. My gratitude is also extended to my brothers and sisters and their families for their motivation and confidence in me.

I also want to thank all my friends for their moral support and motivation who put up with me through the PhD process and helped me with personal challenges, which drives me to give my best. Thanks to all.

Lastly, I would like to thank Al-Quds University for everything they supported me, particularly in providing me the necessary financial support and releasing me on a study leave to undertake my PhD.

Preface

This dissertation entitled "Business Driven Alerts Correlation in Network Management Environments" is submitted for obtaining the PhD degree from the University of Granada. The research described herein was conducted under the supervision of Gabriel Maciá Fernández and Jesús E. Díaz Verdejo, Professors at the Department of Signal Theory, Telematics and Communications, between October 2011 and March 2015.

Along this dissertation, we studied the alerts correlation problem from the point of view of business management in network environments, and described how to use business tickets information to carry out a joint correlation with alerts in order to enhance the correlation process in terms of alerts reduction and semantic information enrichment.

To the best of our knowledge, this work is original. Neither this, nor any substantially similar dissertation has been submitted for any other degree at any other university.

Saeed A. M. Salah

Granada, March 19th, 2015

Abstract

As telecommunication networks evolve rapidly in terms of scalability, complexity and heterogeneity, the efficiency of incident management procedures and the accuracy in the detection of anomalous behaviors are becoming important factors that largely influence on the decision making process in large Information Technology (IT) service companies. For this reason, these companies are doing big efforts investing in new technologies and projects aimed at finding efficient management solutions. One of the challenging issues for network and system management teams is that of dealing with the huge amount of alerts generated by the managed systems and networks. Currently, alerts correlation is the primary technique used to handle this issue. Despite the big amount of research efforts that have been carried out in the alerts correlation field, this is still an active research area in network management. This is mainly due to the fact that the efficiency and robustness of the models used and the algorithms proposed vary from system to system, but none of them have already succeeded to provide an optimal solution to this problem in terms of reducing and aggregating the number of alerts to a single alert per incident.

On the other hand, *Incident Ticketing Systems* (ITSs) play an important role in maintaining modern telecommunication networks and have been mainly introduced to assist in speeding up the incident recovery process and adding more advanced functions to incident solving. As the bulk of the tickets are normally created manually, they constitute ideal candidates to be used into the alerts correlation procedure to add more semantic information and human knowledge, coming from the management staff and also from the end users of the services (through *Service Desks*). Although tickets reflect the business point of view of the incidents, to the best of our knowledge, and despite its potential usefulness, few efforts have been devoted to the incorporation of the information from an incident ticketing system into the alerts correlation procedure itself.

In this work, we propose a generic tickets-alerts correlation architecture composed of three main parts: alerts correlation, incident tickets correlation and tickets-alerts correlation. In the alerts correlation part, a survey of the state-of-the-art in alerts correlation techniques is first presented. Unlike other authors, we consider here that the correlation process is a common problem for different fields in the industry, and not only for network or security management. Thus, we focus on showing the broad influence of this problem. Additionally, we suggest an alerts correlation model capable of modeling current and prospective proposals. Finally, we also review some of the most important commercial products currently available. In the incident tickets correlation part, we first check that, in many cases, the handling of tickets by a management team is not completely systematic and may be incoherent and inefficient. This way, irrelevant or redundant tickets for a same incident are likely issued, thus creating a redundancy in the system that leads to in-

efficiencies. To handle this issue, we suggest a model aimed to correlate redundant tickets in order to ideally reduce the information to a single ticket per incident. Using this model as a basis, we also develop and evaluate a methodology that assesses the efficiency of a management team during the process of tickets creation and management. In the last part, we propose and test a model for the joint correlation of tickets and alerts. Finally, we validate the proposed correlation models by evaluating them with two datasets taken from a real incident ticketing system of an IT service company, in order to analyze their applicability and usefulness by targeting them at three main applications: how can the models be used to evaluate the tickets creation process, how can the models be used to improve the alerts correlation process, and finally, how to use them in evaluating the management team in terms of their speed, accuracy and the influence of each management group in the whole incident resolving process. These analyses can be leveraged for improving both the management groups functioning and the policies for tickets creation and incident management.

The results of this work show that incorporating the ticketing information in the alerts correlation process will permit obtaining better correlation rates, *i.e.*, a bigger and more reliable reduction in the number of alerts. By using these models, decision makers would get more accurate information about the real incidents happening in the network and their descriptions, so that decisions and prioritization procedures would be more precise. At the same time, the proposed methods are based on simple elements and reasonings, making their applications in a real network management system almost straightforward.

Resumen

Al mismo tiempo que las redes de telecomunicaciones evolucionan rápidamente en términos de escalabilidad, complejidad y heterogeneidad, la eficiencia de los procedimientos de gestión de incidentes y la precisión en la detección de comportamientos anómalos se están convirtiendo en factores importantes que influyen en gran medida en el proceso de toma de decisiones en las grandes empresas de servicios TIC. Por esta razón, estas empresas están haciendo grandes esfuerzos de inversión en nuevas tecnologías y proyectos encaminados a encontrar soluciones de gestión eficientes. Una de las cuestiones más complejas de resolver y que va ganando relevancia en este contexto es la de hacer frente a la enorme cantidad de alertas generadas por los sistemas y redes gestionadas. Actualmente, la correlación de alertas es la principal técnica utilizada para resolver este problema. Sin embargo, a pesar de la gran cantidad de trabajos de investigación que se han llevado a cabo en el campo de la correlación de alertas, esta sigue siendo un área de investigación activa en la gestión de redes. Esto se debe principalmente al hecho de que la eficiencia y la robustez de los modelos utilizados y los algoritmos propuestos varían de un sistema a otro, constatándose que aún no se ha alcanzado una solución óptima a este problema en términos de reducción del número de alertas a una sola por incidente.

Por otro lado, los sistemas de gestión de incidentes mediante tiques (ITS, del inglés *Incident Ticketing System*) juegan un papel importante en la gestión y mantenimiento de las redes modernas de telecomunicaciones. Estos se han introducido principalmente para ayudar a acelerar el proceso de recuperación y para la incorporación de funciones avanzadas en la resolución de incidentes. Como la mayor parte de los tiques se crean normalmente de forma manual, resultan candidatos ideales para ser utilizados en el procedimiento de correlación de alertas, de modo que se pueda añadir mayor información semántica proveniente del conocimiento experto aportado por el personal de gestión, asi como también de los usuarios de los servicios (a través de los centros de atención al cliente). Sin embargo, a pesar del aparente potencial que podría suponer utilizar estos sistemas de tiques en combinación con los sistemas de correlación de alertas, podemos destacar el reducido número de intentos, a nivel de investigación, para combinar ambas fuentes de información en un sistema que mejore los actuales.

En este trabajo proponemos un sistema genérico de correlación de tiques y alertas compuesto por tres partes principales: correlación de alertas, correlación de tiques y correlación combinada de tiques y alertas. En relation a la correlación de alertas, se presenta en primer lugar una revisión del estado del arte. A diferencia de otros autores, consideramos aquí que el proceso de correlación es un problema común para diferentes campos de la industria, y no sólo para la gestión de redes o la seguridad. Además, se sugiere un modelo genérico de correlación de alertas capaz de incorporar las propuestas actuales y que es también suficientemente flexible

para considerar propuestas futuras en este campo. Por último, también revisamos algunos de los productos comerciales más importantes disponibles en la actualidad. En relación a la correlación de tiques, comprobamos en primer lugar de forma empírica que, en muchos casos, la gestión de los tiques por parte de un equipo de gestión no es completamente sistemática y puede ser incoherente e ineficiente. De esta manera, se pueden constatar ciertas problemáticas, como la aparición de tiques irrelevantes o múltiples para un mismo incidente, creando así una redundancia en el sistema que conduce a ineficiencias. Para hacer frente a este problema, en esta tesis se sugiere un modelo destinado a realizar la correlación de tiques redundantes con el fin de reducir, idealmente, el número de tiques a uno solo por incidente. Usando este modelo como base, también se desarrolla una metodología que evalúa la eficacia de un equipo de gestión durante el proceso de creación y gestión de tiques. En la última parte, se propone y se evalúa un modelo para la correlación conjunta de tiques y alertas. Finalmente, se validan los modelos de correlación propuestos usando dos conjuntos de datos tomados de un sistema real de gestión de incidentes con tiques de una empresa de servicios TIC, con el fin de analizar su utilidad en dos aplicaciones principales: el uso de los modelos para mejorar el proceso de correlación de alertas y su utilización para la evaluación del equipo de gestión de red en términos de su velocidad, su precisión y su influencia en el proceso de gestión de los incidentes. Estos análisis se pueden aprovechar para mejorar tanto el funcionamiento de los equipos de gestión de red como las políticas para la creación y gestión de tiques.

Los resultados obtenidos a lo largo de este trabajo han reflejado que, incorporando la información de los tiques en el proceso de correlación de alertas, se obtienen mejores tasas de correlación, esto es, una reducción mayor y más fiable en el número de alertas. Por otra parte, los responsables de la gestión de red en las compañías pueden obtener información más precisa sobre los incidentes reales que suceden en la red y sus descripciones, por lo que las decisiones y procedimientos de priorización serían más precisos. Al mismo tiempo, las metodologías se basan en elementos y razonamientos simples, por lo que su aplicación en un sistema de gestión de red real es prácticamente directa.

Acronyms

BML	*Business Management Layer*
CMIP	*Common Management Interface Protocol*
CVE	*Common Vulnerabilities and Exposures*
DoS	*Denial of Service*
DTD	*Document Type Definition*
EML	*Element Management Layer*
FN	*False Negative*
FP	*False Positive*
GPL	*General Public License*
HMM	*Hidden Markov Model*
IDMEF	*Intrusion Detection Message Exchange Format*
IDS	*Intrusion Detection System*
IETF	*Internet Engineering Task Force*
IM	*Incident Management*
IP	*Internet Protocol*
ISO	*International Organization for Standardization*
IT	*Information Technology*
ITIL	*Information Technology Infrastructure Library*
ITS	*Incident Ticketing System*
ITSM	*Information Technology Service Management*
ITU-T	*International Telecommunication Union-Telecommunication*
LAN	*Local Area Network*
MIB	*Management Information Base*
MS	*Management Staff*

NML	*Network Management Layer*
NMS	*Network Management System*
NOC	*Network Operations Center*
NTP	*Network Time Protocol*
OSI	*Open Systems Interconnection*
OSS	*Operations Support Systems*
P2P	*Peer-to-Peer*
QoE	*Quality of Experience*
QoS	*Quality of Service*
RFC	*Request For Comment*
RMON	*Remote Network MONitoring*
SCADA	*Supervisory Control and Data Acquisition*
SD	*Service Desk*
SLA	*Service Level Agreement*
SML	*Service Management Layer*
SNMP	*Simple Network Management Protocol*
TMN	*Telecommunications Management Network*
TN	*True Negative*
TP	*True Positive*
TTS	*Trouble Ticketing System*
UDP	*User Datagram Protocol*
XML	*Extensible Markup Language*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The process of alerts correlation applied to the network management field constitutes the main focus of this work. Essentially, alerts correlation is a complex process aimed at establishing relationships between the alerts in order to summarize or aggregate them. Knowing the basics of this context is a primary task in order to build proposals and adequately situate them. For this reason, in this introduction chapter, we first present the problem of alerts correlation, explaining the basic concepts related to network management and its relevance in incidents' solving. The goal of this chapter is to clearly state the main hypothesis to be validated with this work. After this, the motivations, additional hypotheses and the objectives are described. This is followed by the research methodology and the main contributions. Finally, an outline of the structure of this document is provided.

1.1 Network Management Concepts

Nowadays, communication networks are evolving rapidly in terms of scalability, complexity, and heterogeneity. For example, an infrastructure for a current corporate network may span over a large geographical area; encompass many technologies, multiple vendors' equipments, and different types of subnetworks; run a wide variety of applications and protocols; and operate under strict reliability constraints. In the functioning of this type of networks, it is essential to provide management mechanisms. Therefore, *Network Management System*s, (NMSs), are nowadays among the most important elements for the success in the functioning of *Information Technology* (IT) service providers or IT departments in enterprises. The maintenance and configuration of network devices, servers, and services, as well as the continuous monitoring of the operation of all the devices within the network are the key elements of a NMS.

In the literature, several definitions of network management exist [5–7]. Most of these definitions are provided by standardization organizations that use specific terminology and aim their definitions at specific fields of application. However, the most general and comprehensive definition of network management is found in [8], being defined as:

"Network management refers to the activities, methods, procedures, and tools that pertain to the operation, administration, maintenance, and provisioning of corporate networks. Operation deals with keeping the network (and the services that the network provides) up and running smoothly. It includes monitoring the network to spot problems as soon as possible, ideally before users are affected. Administration deals with keeping track of resources in the network and how they are assigned. It includes all the "housekeeping" that is necessary to keep the network under control. Maintenance is concerned with performing repairs and upgrades –for example, when a line card must be replaced, when a router needs a new operating system image with a patch, when a new switch is added to the network. Maintenance also involves corrective and preventive proactive measures such as adjusting device parameters as needed and generally intervening as needed to make the managed network run "better". Provisioning is concerned with configuring resources in the network to support a given service. For example, this might include setting up the network so that a new customer can receive voice service."

Network management has been attracting the attention of a growing number of researchers over the past years. Researchers in both academic institutions and industrial sectors have invested big efforts in new technologies and projects aimed at finding efficient management solutions in order to cope with the technological revolution in the communication industry. Two reasons may be cited for this interest. The first is the relevance of network management in economic terms; for many companies today, the reliability of the IT services they use has become a critical factor for reaching success in the business market. Due to the industry trend to focus on the core business, IT services are in many cases outsourced to external companies. To ensure that these IT services are provided in a reliable manner, a service contract or *Service Level Agreement* (SLA) [9] is laid down between the customer and the IT service provider. This contract specifies *Quality of Service* (QoS) parameters that describe the performance of the service in question [10], which implies the use of a NMS by the IT service provider. If the QoS parameters are not met, the agreement usually considers certain penalties to cover the resulting consequences for the customer. Anyway, whether IT services are outsourced or not, the accomplishment of the required QoS becomes relevant in economic terms for the business. Thus, effective management procedures to provide this QoS may imply a significant increase in productivity, through a better utilization of the available resources.

The second reason for the interest of the scientific community in network management is the amount of different problems and challenges involved. In order to provide a satisfactory solution for the network management, a lot of research might be required in several areas of computer science such as computer networks, software engineering, database systems, human machine interface and artificial intelligence, among others.

Deploying efficient network management mechanisms, which is an inherently difficult task and by no means an easy option in current corporate networks, is expected to have the following benefits for IT service providers or IT departments in enterprises [8, 11, 12].

- *Reducing cost*: Having a NMS increases profitability, as a management staff can easily monitor and manage network outages, if any, and resolve them almost immediately. Furthermore, instead of several groups managing each subnetwork at every location, an appropriate NMS could permit the implementation of a system with a management staff at a single location to control and manage the entire network.

- *Saving time*: Network management helps a management staff to understand the problems faced by customers quickly. Constant monitoring of networks informs of any network errors that could imply fatal consequences for the company. Besides, being well informed in advance allows to rectify and resolve these errors before they damage the reputation of the company as well.

- *Increasing productivity*: With NMS, all aspects of a corporate network including hardware, software, and peripherals can be managed. All of these components need to be able to communicate with each other. Should one go down, the whole system could be impacted. NMS detects it so that there is no data loss or productivity slowdown. So that, using NMSs, decision makers in IT service providers or IT departments in enterprises can increase their productivity and, ultimately, their profits.

- *Increasing revenues*: Increasing customer acquisition and retention, commonly referred to as the *Quality of Experience* (QoE), becomes very important as well as a significant challenge to the IT service providers with a goal to minimize the customer churn yet maintaining their competitive edge. Thus, having an efficient NMS will increase QoE. This will impact on the reputation of IT service providers, and consequently, it will have a significant increase in their revenues.

Despite the above benefits of network management, there are several issues that present new challenges to current NMSs and must be dealt with in order to get the maximum benefits [11, 13–15]. First of all, as the size and complexity of communication networks continue to grow rapidly, this leads to an increase in the number of elements that make up these networks. This will require better scalability enhancement mechanisms on the designs of network management platforms. In this regard, the used NMS must take care of how to make the configurations of these elements more realizable. Second, as network infrastructures from various sectors converge, heterogeneous network technologies must coexist and inter-work. NMSs must provide efficient mechanisms to implement such seamless integration, and hide the underlying technological heterogeneity from end users. Third, the competitive nature of IT service providers demands economical operation of networks. Consequently, network management solutions must be more self-regulating and self-governing, in order to be economically beneficial. In addition, they also must be kept simple and elegant because, as the development of the Internet has evinced, only simple and elegant solutions would dominate in large-scale heterogeneous networks. Last but not least, as managed devices become more and more powerful, there is an increasing pressure to exploit their processing capabilities in an efficient way. This also leads to an increasing need for distributed or hierarchical NMSs.

1.2 Network Management Standards

In this section, we briefly bring together the basics of three network management frameworks. They are FCAPS, *Telecommunications Management Network* (TMN), and *Information Technology Infrastructure Library* (ITIL).

1.2.1 FCAPS Model

The FCAPS model is a major network management framework [8, 16]. It has been created by the *International Organization for Standardization* (ISO) with the aim of providing focus and consistency in the area of network management. FCAPS categorizes the working objectives that conform the backbone of network management into five key functional areas: *fault management*, *configuration management*, *accounting management*, *performance management* and *security management*. The FCAPS term is an acronym formed by the five initial letters of these functional areas that are described below:

- **Fault management**: The goals and objectives of fault management include early fault recognition, generating notifications when that recognition occurs, isolation of negative effects, fault correction and logging of the corrections to assist in improvement, among others.

- **Configuration management**: It is the process of organizing and maintaining a consistent, repeatable, and audit-able information about all the hardware and software elements that make up the network. Thus, it includes all of the configuration aspects of network elements such as resource initialization, network provisioning, backup and restore, remote configuration, automated software distribution, configuration file management, inventory management and software management, among others.

- **Accounting management**: It provides the necessary mechanisms to make the measurements of the use of network services and the determination of the associated costs to the IT service provider, and the charges to the customer for such use. This process involves tracking network utilization information and informing relevant users, departments, and authorities about the usage of resources, and the associated costs. This makes the most effective use of the systems available, and minimizes operational costs. Furthermore, accounting management area is also responsible for ensuring that users are billed appropriately.

- **Performance management**: It is related with managing the overall performance of the network, *i.e.*, throughput monitoring, network bottlenecks detection, and potential problems identification. A major part of the effort is to identify what improvements will yield the greatest overall performance enhancement. Several statistics are gathered and analyzed to monitor a system and network throughput. Examples of the statistics gathered include system errors, utilization and response times, which are used to identify system trends and plan for future use.

- **Security management**: This area covers the configuration and monitoring for avoiding and minimizing security threats and violations. Examples of the tasks that are managed in this area include securing the network, securing management interfaces and creating audit trails, inspecting traffic payloads to guard against viruses and trojan horses, enforcing policies that limit increases in the amount of traffic to a particular destination or from a particular source to guard against *Denial of Service* (DoS) attacks, "blacklisting" ports and network addresses where suspicious traffic patterns are observed, and providing the access to network devices and corporate resources to authorized individuals.

For a corporate network to be effectively utilized, it is indispensable that all of these areas' features are adequately managed and that there exists an integration among the five management functional areas described above, *i.e.*, the information generated in one area may be useful in other areas and therefore should be available for all.

1.2.2 TMN Model

TMN [7, 17] is another well-known network management standard that is defined by the *International Telecommunication Union-Telecommunication* (ITU-T) (formerly CCITT). Unlike FCAPS, which divides the network management process into functional areas, network management is mainly introduced here as a reference model for a hierarchical network management approach. The main argument behind the hierarchical approach is that it deals better with the complexity of management. Therefore, the management functionality with its associated information, as described by TMN, can be decomposed into four logical layers: *the Element Management Layer (EML)*, *the Network Management Layer (NML)*, *the Service Management Layer (SML)*, and *the Business Management Layer (BML)*. The TMN model segregates the management responsibilities based on these layers. This makes it possible to distribute these functions or applications over the multiple disciplines of IT service providers or IT departments in enterprises, and use different operating systems, different databases and different programming languages. A brief discussion of each of these layers is given below.

- **Element Management Layer (EML)**: This layer mainly deals with vendor specific management functions and hides these functions from the network management layer. Examples of functions performed at this layer include detection of equipment errors, measuring power consumption, measuring the temperature of equipment, collecting statistical data for accounting purposes, measuring the resources that are being used, like CPU-time, buffer space, queue length, etc., logging event notifications and performance statistics and defining interfaces for other network elements, among others.

- **Network Management Layer (NML)**: The network management layer manages relationships and dependencies between network elements, generally required to maintain end-to-end connectivity in the network. It mainly concerns with keeping the network running as a whole, offering a holistic view of the network among the multiple pieces of equipment and independently of device types and vendors. Examples of functions performed at this layer include creation of the complete network view, creation of dedicated paths through the network to support the QoS demands of end users, modification of routing tables, monitoring of link utilization, optimizing network performance and detection of root causes behind faults, among others.

- **Service Management Layer (SML)**: It is concerned with the contractual aspects of services that are being provided to customers. Examples of the main functions of this layer include service creation, service monitoring, service implementation, order handling, QoS management (delay, loss, etc.), accounting, addition and removal of users, maintenance of group addresses and invoicing, among others.

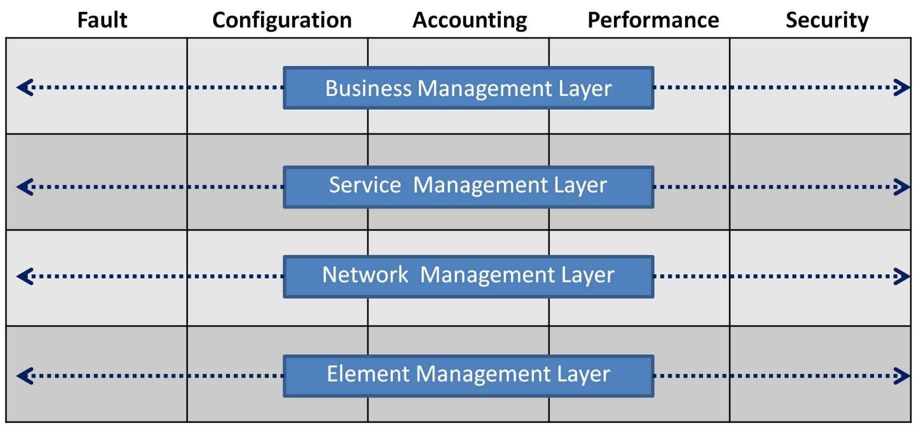


Figure 1.1: TMN logical layer architecture and its relationship with FCAPS model (taken from [1]).

- **Business Management Layer (BML)**: The business management layer can be considered a goal-setting approach: "What are the objectives, and how can the network (and network management specifically) help achieve them?". This includes high level planning, budgeting, goal setting, executive decisions and business level agreements, among others.

In this context, the FCAPS model is considered an extension to the TMN model. Each management layer in the TMN model is responsible for providing the appropriate FCAPS functionality according to the layer definition, and communicates with the layers above and below it. Figure 1.1 shows the relationship between the different layers of the TMN model as well as the relationship with the FCAPS model.

1.2.3 ITIL

Currently, ITIL is the most widely adopted public framework to *Information Technology Service Management* (ITSM) in the world [18, 19]. ITIL is a set of best practices standards that focuses on aligning IT services to meet business needs and goals. It was originally developed by the United Kingdoms's Central Computer Telecommunications Agency and currently it is maintained by the Office of Government Commerce. By the mid 1990's, ITIL became the world-wide de facto standard in service management, which is currently considered as one of the fastest growing business optimization initiatives distributed over the world. ITIL processes are being adapted by organizations both big and small due to its ability to improve business processes. Due to the fact that ITIL focuses on best practices, it can be adapted and adopted in different ways according to specific individual organizations needs. One of the primary factors leading to such rapid growth and adoption of ITIL are the benefits being reported by customers and users. ITIL is organized into sets of texts that are defined by related functions: service support, service delivery, managerial, software support, computer operations, security management, and environmental. Besides the services and products, ITIL also includes materials for several purposes such as training, qualifications, software tools, and user groups such as the IT Service Management Forum .

The service management section of ITIL is made up of eleven different disciplines, split into two sections, *Service Delivery* and *Service Support*. ITIL service delivery includes capacity management, SLA, continuity management, availability management, and IT financial management. Whereas ITIL service support includes configuration management, incident management, problem management, change management, service desk and release management, among others.

In sum, a simple review of the three standards shows that FCAPS, TMN, and ITIL all overlap in terms of the concepts that they address. It is true that they share the same concepts. Yet, they do so at different levels of abstraction. The FCAPS model primarily focuses on the concept of technology management. The TMN model focuses on service management, and presents technology at a level that the business can understand (not just the technical staff). The ITIL framework, on the other hand, is all about how to run an efficient IT organization. Thus, ITIL focuses on processes and workflows. Based on this, we can say that FCAPS started with a technology centric view, TMN layers on top a service and business oriented view, and ITIL added the process optimization and efficiency to the equation.

In the following sections, we look inside one of the main functional areas that the above three management standards describe, *i.e.*, fault management. We do focus on this topic because the presented work in this thesis is centered around it.

1.3 Network Management Architecture

A thorough review in the literature reveals that researchers have described three management architectures that might be deployed in corporate networks, depending on the size and internal policies of a company. These management architectures are *centralized*, *distributed* and *hierarchical* [11, 20]. In a centralized management architecture, there is only one manager responsible of collecting information from all agents and controlling the entire network. This basic model is known in the literature as simply a “manager-agents”. Distributed management encompass multiple managers; each one controls a portion of the network and may communicate directly with other managers. Finally, the hierarchical architecture is considered as a hybrid approach between both centralized and distributed managements. Here, each manager locally manages a subset of network agents and it is assigned with a given degree of responsibility.

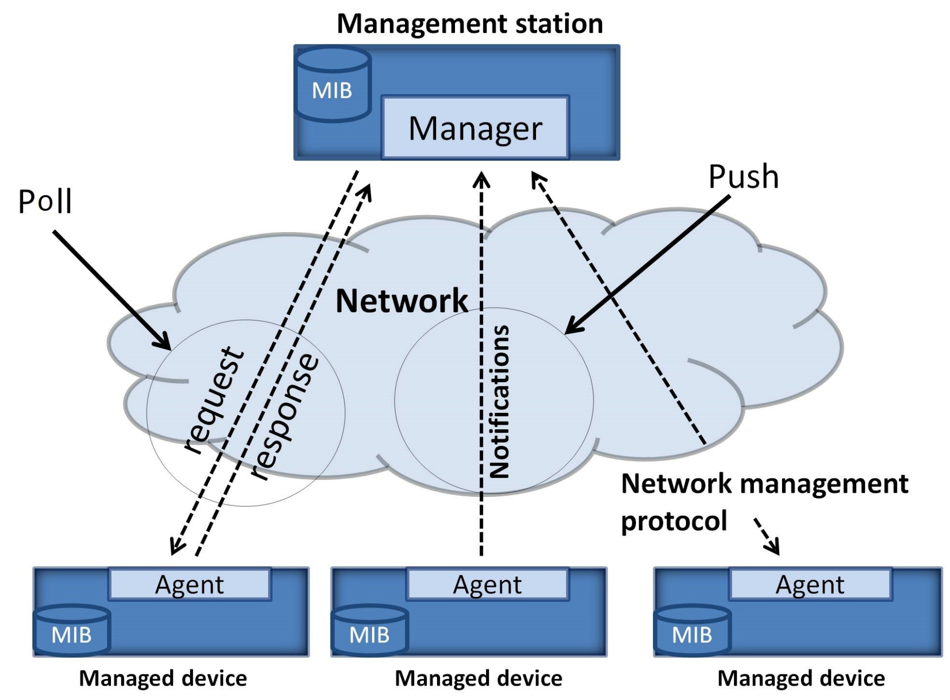


Figure 1.2: Manager/agents model of a network management architecture.

Although the three management architectures have different characteristics, they all share the same basic structure and components. As illustrated in Figure 1.2, there are five main components in a network management architecture: *managers*, *agents*, *network management protocols*, *Management Information Base* (MIB) and *a communication model* [11, 21]. In what follows we give a brief overview about each component.

- **Managers:** The main concept of the network management platform is called a manager or management entity. It is an application, typically with a user interface, normally running in a management station located in a management center of an IT company. This center focuses on the network management operations and is called *Network Operations Center* (NOC). The main key functions of the manager are: it collects network management data from the agents and presents them for analysis by the management staff; it queries agents and gets responses from them; it is able to configure equipments by setting variables in agents; and it is able to acknowledge asynchronous events from agents, among others.

  It is worth to mention here that the work presented in this thesis mainly introduces some contributions to help a management staff enhancing their operations in the manager side, due to the fact that all the fault management tasks are mainly handled by a management staff located at the manager side.

Furthermore, the databases of alerts and tickets investigated in this work are collected and maintained by the management staff of an IT service provider or an IT department of an enterprise. So, the contributions of this thesis can be leveraged by the management staff to efficiently handle the huge amount of alerts that they receive every day. On the other hand, this thesis work does not provide any contributions to enhance functioning and operation of any of the other management components that are listed below.

- **Agents:** An agent is a program that is normally packaged within a managed device or a network element, and that is in charge of monitoring that managed device and communicates with the manager. In this context, a managed device might be a workstation, router, switch or server, among others, that requires some form of monitoring and management. Enabling the agent on the managed device allows it to collect the management information from the device and makes it available to the manager, both asynchronously or when it is queried.

- **MIBs:** Every agent maintains its management information locally in a database that describes the managed device parameters. This database, shared between the agents and the manager, is commonly called MIB. Typically, a MIB contains a standard set of statistical and control values defined for the managed device. The manager uses this database to request the agent for specific information and further translates the information as needed for the NMS.

- **Network management protocol:** The main purpose of the network management protocol is to enable the exchange of information and commands of network devices management and monitoring through a common language across all devices. For example, the communication between the manager and the agents is normally carried out by sending and receiving messages that are defined by a management protocol. The following network management protocols are the most widely adopted, all of them being defined by various *Internet Engineering Task Force* (IETF) standards.

  - *Simple Network Management Protocol* (SNMP) [22]: It is the most widely deployed protocol. It is defined in the *Request For Comment* (RFC) 1157, with the purpose of managing Internet and IP-based internetworks. Most of the current professional-grade network devices come with a bundled SNMP agent. These agents have to be enabled and configured to communicate with the SNMP manager. There are three versions of this protocol currently available: SNMPv1, SNMPv2, and SNMPv3. They all share the same basic structure and components, and they follow the same architecture. SNMPv1 and v2 are the most used versions of SNMP. SNMPv3 has recently started catching up as it is more secure when compared to its older versions, but it has not still reached considerable market share.

  - *Remote Network MONitoring* (RMON) [23]: RMON is really an extension to SNMP that was mainly designed for the management of network traffic. It enables various network monitors and console systems to exchange network-monitoring data. The RMON specification defines a set of statistics and functions that can be exchanged between RMON-compliant console managers and network probes. To do that, nine types of information are collected, including packets and bytes sent, packets dropped, statistics by host, statistics by conversations between two sets of addresses, and certain kinds of events that have occurred. It allows to find out how much bandwidth or traffic each user is imposing on the network, or what web sites are being accessed. Besides, notifications can be sent to the manager in order to be aware of impending problems. There are two versions currently available: RMONv1 and RMONv2 (sometimes referred to as "RMON2").

  - *Common Management Interface Protocol* (CMIP) [24]: It is a network management protocol defined by the ISO/IEC 9595 and 9596 standards. It is built on the *Open Systems Interconnection* (OSI) communication model that defines how to create a common network management infrastructure. While both CMIP and the SNMP define network management standards, CMIP is far more complex. In contrast to the SNMP, which is specifically designed for TCP/IP networks commonly used on corporate networks, CMIP is really only used by some IT service providers for network management. It was mainly proposed as a replacement for SNMP, but has not been adopted by the networking community for widespread implementation because of its complexity.

- **Communication model:** There are two well-known models for exchanging data between the manager and the agent in a NMS: *poll* and *push*. The poll model is based on the request/response paradigm (called data polling, or simply polling, in the SNMP management framework); the manager uses a *request* command to poll the agent in requesting information, and the polled agent collects the requested information, and sends it back to the manager, in a message that is called a *response*. In this approach, the data transfer is always initiated by the manager, and the polling itself can be automatic or user-initiated. In the push model, conversely, an agent first advertises what MIB it supports, and what notifications it can generate; the management staff then subscribe the manager to the data they are interested in, specify how often the manager should receive this data, and disconnect. Later on, each agent individually takes the initiative to push data to the manager, either on a regular basis via a scheduler (*e.g.*, for network monitoring) or asynchronously (*e.g.*, to send notifications).

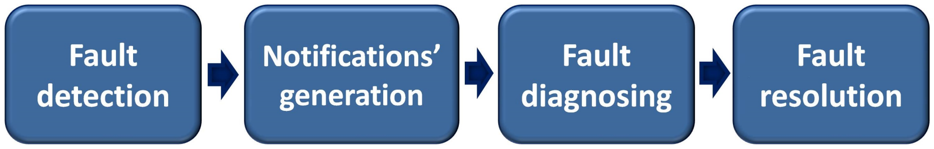


Figure 1.3: The four steps of the fault management process.

The notifications are called traps in SNMP terminology. Traps are simply defined as messages sent to the manager by an agent when something needs to be reported. Basically, there are seven generic types of traps: a *ColdStart* trap means that the agent has detected some changes in the configuration, for example, if a server has a value in its BIOS changed before booting the operating system; a *WarmStart* trap signals that the system has been cut/restored; *LinkDown/NodeDown* and *LinkUp/NodeUp* traps signify that communications through a Node/Link are out of service or that have been restored; an *AuthenticationFailure* trap signals the manager that a request has been received from an unauthorized source; a *NeighborLoss* trap notifies that a relationship between two nodes has been broken; and the last one is *enterpriseSpecific* that informs the manager of an event that occurred in a specific piece of hardware/software defined by the vendor's software. In this context, the specific-type field will contain the MIB that is used to define the exact type of event that has occurred.

It is worth mentioning here that not all notifications are treated equally during the fault management process. They are classified by the management staff based on internal policies, which are mainly based on their effects on the stability of the network. Consequently, the work carried out in this thesis is targeted at specific types of notifications, *i.e.*, those having critical effects on the stability and functioning of the network. In the following chapters, several mechanisms are used to distinguish between these notifications. The aim is to choose relevant notifications and filter out the remaining, *i.e.*, the irrelevant ones.

1.4 The Fault Management Process

As illustrated in Figure 1.3, typical fault management systems use four steps to break down the complicated task of identifying and resolving the faults [16]. They are: *fault detection*, *notifications' generation*, *fault diagnosing*, and *fault resolution*.

- **Fault detection:** It provides the capability to recognize faults and send notifications whenever needed. For this purpose, it uses several mechanisms to monitor devices and report activities occurred in the network. These activities must be accurate, relevant and complete. In this context, two fault detection modes can be configured within the NMS for collecting these activities, *passive* and *active* [16]. In the passive mode, the agents notify the manager when a pre-configured condition has occurred. Passive fault management can track situations and devices' conditions from nominal values, being the responsibility of the management staff to pre-configure the agents with the business' definition of nominal activity. After all, if a device is non-functional, the agent will not raise any notification to the manager. In active fault management, the manager uses several probing mechanisms to check its connectivity with agents. For example, the manager may send a network `PING` command to each agent on a regular basis and listen for the reply. If the agent does not reply after a pre-configured interval, the active monitoring will notify the management staff of a managed device outage. For this reason, active fault management is often also referred to as "up/down monitoring". In all the cases, these data gathering and error forwarding actions are components of the agent that is installed on a managed device. The agent will forward the error, usually through trap or `Syslog`, to the manager for processing.

Small business networks with a small number of managed devices are not likely to have an NMS in place to notify when a fault occurs. In these cases, the basic mechanism for fault detection is not automated, and a fault is considered to appear when customers complain due to a change in the state of their connections or services. Thus, they contact the appropriate personnel in the IT management company or department –often through a service desk– and notify them that they have noticed an anomaly on the network.

- **Notifications' generation:** When a fault occurs and the fault information has been forwarded to the manager, the manager has to accomplish some actions. First, the error must be parsed and identified. Usually it is compared with the pre-generated set of logical rules. When matches occur, the manager will trigger a response such as a notification to the management staff using mechanisms such as: sending notifications to the console, sending an email message, sending an SMS message to a cell phone or pager or executing a script, among others.

  In this context, notifications are also referred to in the literature as *alarms* or *alerts* [25, 26]. They are short messages with a specific textual format defined by vendors of network equipments, and generated as an external manifestation of a potential failure or a disorder occurred in a managed device or service of a corporate network or system. Typically, such alerts contain information regarding the device or service that issues them and the event itself, *i.e.*, the creation and reception time, a description of the fault, the severity of the alert, etc. Besides, alerts may provide information with different levels of detail: specific data regarding the status of the devices or services and their configurations, or higher level details, with aggregated information gathered from several alerts. The management staff can monitor alerts in real-time, while active and historical alerts are usually stored in a relational database.

- **Fault diagnosing:** It is the troubleshooting approach followed to track down the problem, and to find the root cause behind the fault. For complicated problems, fault isolation can involve substantial error analysis and deep diagnosis of the symptoms. These sorts of deep-dive problems can occur in large networks with multiple management teams that manage devices within separate but connected domains of management. Thus, the task of fault diagnosing might involve reviewing alerts across multiple devices and across multiple management domains.

- **Fault resolution:** Fault resolution is the last step in the process of fault management. Once the root cause of the problem is identified and detected, the next step is to commit the corrective actions. Examples of these actions include changes in the configuration of managed devices, automatic correction of potential problem-cause conditions, automatic resolution of actual malfunctions and detailed logging of system status, among others. In addition, some approval and personnel notifications are needed in order to disseminate theses changes through some form of change management. Finally, tools and technology could exist to ensure that the changes are logged into a configuration management database correctly and completely.

1.4.1 Main Roles in Fault Management

During the discussed fault management process, several actors, roles and functions are involved:

- **Service Desk (SD):** It acts as a point of contact for phone calls and emails from customers regarding IT issues and queries. The main responsibilities of the *Service Desk* (SD) staff include receiving, logging and managing calls from customers or employees via telephone or email, logging the fault in the call log, performing the initial fault diagnostics, requesting technical support when required, and updating records (call log, fault sheet) with the resolution, among others.

- **Management Staff (MS):** It is the main technical staff that is responsible for network management in general and, in particular, for solving network faults. Regarding this last duty, its objective is to restore the service as soon as possible. Its responsibilities include analyzing alerts to identify faults, assisting with classification and prioritization of faults, and taking fault resolution actions to restore service to customers, among others. The management staff may encompass three types of actors involved in the fault management process. They are: *fault reporters*, staff members responsible for discovering and informing about faults; *fault resolvers*, staff members responsible for carrying out the resolution tasks; and *solution validators*, staff members responsible for testing whether the applied solution is satisfactory or not.

- **Additional roles:** Additional first line support groups, such as configuration management or change management specialists might be considered. Second and third line support groups, including specialist support groups and external suppliers might be also considered as necessary. Customers also should keep the the service desk informed of any further changes to the state of the affected equipment (sometimes computers start working again when different faults are resolved).

1.4.2 The Role of Alerts Correlation in Fault Diagnosing

Alerts correlation is a widely accepted technology used by a management staff of an IT service provider or an IT department of an enterprise for the purpose of speeding up the fault diagnosing process. It is defined by Jacobson and Weissmann [25] as *"a conceptual interpretation of multiple alarms such that a new meaning is assigned to these alarms. It is a generic process that underlies different network management tasks such as context-dependent alarm filtering, alarm generalization, network fault diagnosing, generation of corrective actions, proactive maintenance, and network behavior trend analysis"*. Gardner and Harle [27] defined it as *"the interpretation of multiple alarms so as to increase the semantic information content associated with a reduced set of messages"*.

This need for alerts correlation comes from several reasons. First, with the growth of the managed networks, it is estimated that in the mid run, the NOC of a medium size network might be receiving hundreds of alert notifications per day, which will render the "manual" processing of all of them practically unfeasible. Also, it is worth to mention here that today's monitoring platforms such as `HP OpenView` [28] trigger a huge amount of so called normal-behavior alerts in response to daily operational tasks that are not really associated to real network faults, *i.e.*, maintenance activities or software updates, among others. Thus, in order to speed up the fault diagnosis process, it is necessary for the management staff to filter out irrelevant alerts, summarize them if possible, and focus on the most crucial ones. Second, alerts that are generated as a response to the detection of malicious activities or faults do not usually include explicit information about their root cause. Third, in collaborative systems, the diversity and heterogeneity of the managed elements poses a significant challenge to the management staff, due to the difficulty of converging alerts coming from multiple data sources in order to develop coherent management strategies. Finally, many of the received alerts do not contain original information of the fault. In fact, the occurrence of a single fault in the supervised network sometimes results in the reception of multiple alerts. Several factors contribute to this situation: an agent may generate several alerts due to a single fault; a fault may be intrinsically intermittent, what implies the sending of an alert at each new occurrence; the fault may result in the sending of an alert each time the service supplied by a managed device is invoked; a single fault may be detected by multiple agents, each one of them emitting an alert; and, finally, the fault of a given device may affect several other devices, causing the fault's propagation.

The Problem of Alerts Correlation

There are some fundamental questions that need answers in order to improve the states of affairs in the alerts correlation process: Which alerts can be filtered out? How can the alerts be grouped and correlated? How can the alerts be prioritized based on their severity? To answer these questions, different alerts correlation techniques, algorithms and models coming from different design approaches have been proposed in both industrial and research communities, each one having its own advantages and disadvantages. All of these correlation techniques try to handle in an efficient way such a huge amount of alerts through a conceptual interpretation of multiple alerts so that a new meaning is assigned to them or some of them are grouped together. A detailed analysis of the alerts correlation process is covered in Chapter 2.

Despite the big amount of research efforts that have been carried out in the alerts correlation field, this is still an active research area in network management. This is mainly due to the fact that the efficiency and robustness of the models used and the algorithms proposed vary from system to system, but none of them have already succeeded to provide an optimal solution to this problem in terms of reducing and aggregating the number of alerts to a single alert per root cause [4, 29].

This thesis is focused on providing a novel approach to the problem of alerts correlation, by means of incorporating a business process vision of the faults, as it will be explained in what follows.

1.5 Incident Management

As previously mentioned, the main objective of this work is to incorporate business and service information in the alerts correlation problem. Therefore, in order to understand this necessity, in this context, the concept of incident is important. In this section we give detailed information about the type of business information found in incidents that will help to achieve the main objective of this thesis.

Citing the ITIL terminology, an incident can be defined as *"an unplanned interruption of an IT service or reduction in the quality of an IT service. Failure of a configu-*

ration item that has not yet impacted service is also an incident" [18, 19]. For example, a user that receives an error message when trying to run an application he/she has never had problems with in the past, or a user that cannot access a new web site would each be examples of individual incidents.

In ITIL, the fault management process is divided into two sub-processes: *incident management* and *problem management*. The objective of the incident management process is to restore normal service operation as quickly as possible, and to minimize the adverse impact on business operations, thus ensuring that the best possible levels of service quality and availability are maintained. This process is primarily aimed at the end user level. On the other hand, problem management deals with resolving the underlying cause of one or more incidents. The focus of problem management is to resolve the root cause of errors and to find permanent solutions. Although every effort will be made to resolve the problem as quickly as possible, this process is focused on the resolution of the problem rather than the speed of the resolution. This process is executed at the enterprise level.

Incident Ticketing Systems

*Incident Ticketing System*s (ITSs) referred to also as *Trouble Ticketing System*s (TTSs) in transitional FCAPS and TMN models, or SDs in ITIL (see Section 1.2), have been introduced as a main tool to assist in the incident management process. They are databases storing reports in a specific form-based structure, and tools to interact with them in order to handle the information and process flow to create, update, and resolve any network incident reported by customers, organization employees or monitoring systems. They might also contain administrative information about customers, workarounds to be applied for common incidents, and other similar data.

When using an ITS, an incident resolution begins with the documentation of an incident within a ticket. The incident ticket may pass through several hands and undergo various degrees of escalation with respect to fault severity or customer priority. Operations that can be performed on a collection of tickets include linking them together if they refer to the same problem and performing statistical analysis to report histories of device behavior and repairs. Basic functions of ITSs include coordination of maintenance, repairing and testing activities, escalation of network problems, operations to perform trend analysis, and the means to design specialized reports on network maintenance and behavior. These reports provide valuable input for studying and evaluating SLAs with customers.

In this thesis, we are interested in exploring the potentials of the relationship between ITSs and alerts correlation systems. Some researchers [30, 31] have pointed out the importance of ITSs for incident resolving, claiming that they should be extended with advanced functions to enhance the incident resolving process, as the information contained in a ticket might be related to incidents generated by events that have already been identified as network failures. Others [32] mentioned that it would be possible to use network status information from the ITS to filter and aggregate alerts that are displayed on the alerts management console, as incident tickets could potentially contain much further information of that alerts, which can be useful for expert system analysis of current network alerts information.

1.6 Objectives and Hypothesis

The main objective of this work is to explore the potentials of the use of ticketing information when incorporated into the alerts correlation process, and to check if the semantic information contained in an ITS, when incorporated in the alerts correlation process, can help in aggregating a higher percentage of alerts when compared with basic alerts correlation systems that only use alerts as the main source of information.

In this context, introducing the ticketing information in the alerts correlation techniques is similar to introducing business information coming from customers. As a matter of fact, currently, customer satisfaction depends not only on the reactive results from after-incident inquiries, but also on the proactive management of the customer's expectations. Thus, we expect that this type of information can really provide valuable input for maintaining and expanding the provider's business. Furthermore, ticketing information, which reflects the business perspectives of incidents, is also richer from the semantic point of view, as they contain information from users and the management staff. Thus, our intention is to incorporate human knowledge and relevance into the alerts correlation process.

This general objective is divided into four specific objectives:

Obj1. To set up a framework to analyze ticketing information coming from an ITS.

Obj2. To propose a generic model for the correlation of tickets and alerts.

Obj3. To evaluate both the framework and the model to show their impacts and limitations in the alerts correlation process.

Obj4. To use of the proposed method as an assessment tool.

The basic and fundamental hypothesis behind this work is that a single incident generated by an unique cause can generate many alerts and tickets. Another hypothesis backing up this work is that ITS information will potentially help in the process of alerts correlation, acting as information provided by an expert system. As the bulk of the tickets in ITSs are normally created manually, they constitute ideal candidates to be used into the alerts correlation procedure to add more semantic information and human knowledge, coming from the management staff and also from the users of the services (through SD).

The following hypotheses regarding the information provided by tickets in ITS are also assumed:

1. There is a correlation between tickets themselves. It is possible that more than one ticket exist for the same incident.
2. There is a correlation between alerts themselves. Similarly, the same incident can generate many alerts.
3. Tickets recorded in the ITS are generated based on the problems and their symptoms. Therefore, we argue that many records in the ITS database contain information related to the incidents generated by events that have already been identified as network failures observed as alerts. Thus, there exist some correlation between alerts and tickets.

1.7 Methodology

During this work, we have followed the scientific methodology based on building theoretical models and contrasting them with empirical data. These data consist of a database from a real IT management company comprised of tickets and alerts (described in Chapters 3, 5 and 6). We have faced an important problem due to the fact that the database is not supervised, *i.e.*, no information about ground truth is present, so we have had to develop mechanisms to validate our results, as it will be explained in the following chapters.

The work plan followed to accomplish the objectives of the thesis has comprised the execution of several sequential phases along the research period. We briefly describe the different phases and their objectives:

1. **State-of-the-art phase:** A state-of-the-art of both alerts correlation and tickets correlation techniques has been first carried out by collecting information from papers, tutorials and online resources, in order to make a classification of the existing techniques and study their advantages and disadvantages.
2. **Tickets exploration phase:** Here, a preliminary exploration of the data contained in the ITS database has been carried out. For this purpose, a statistical analysis on the data was done in order to extract features such as tickets generation procedures, types of tickets, temporal properties, and types of attributes, among others.
3. **Tickets correlation phase:** A methodology for organizing the information provided by an ITS has been first suggested; this methodology has been defined in the most possible general way. Then, based on it, a model to correlate

redundant tickets in an ITS has been proposed. The aim is to reduce the number of records in the database without affecting the relevant information.

4. **Alerts exploration phase:** Here, a preliminary exploration of the data contained in the alert database has been carried out. For this purpose, a statistical analysis on the data has been done in order to extract features such as alert flows, types of alerts, temporal properties, and types of attributes, among others.

5. **Alerts correlation phase:** In order to enable the comparison between our developments and previous techniques, a reference system for alerts correlation has been built by applying basic correlation techniques to the alert database and then obtaining, processing, and analyzing the results.

6. **Tickets-alerts correlation phase:** In this phase, a generic tickets-alerts correlation architecture has been suggested. It mainly consists of three modules: a module for tickets correlation, other for alerts correlation and the last one for tickets-alerts joint correlation.

7. **Evaluation phase:** Results obtained from the new correlation system have been analyzed and interpreted. Furthermore, due to the lack of ground truth information in the studied database, several validation methods adapted to the problem have been developed. The results have also been compared with those obtained from the basic alerts correlation method built in a previous phase.

8. **Application phase:** In this phase, the applicability of the proposed tickets-alerts correlation system has been analyzed by targeting it to some of the current real management scenarios, mainly in alerts reduction and measuring staff efficiency.

1.8 Main Contributions

The main contributions of this work are listed below:

1. A comprehensive state-of-the-art in alerts correlation techniques has been provided. It is based on a newly proposed taxonomy, and provides a global insight on the different efforts made in this field from different research and industry communities within the last few years.

2. A framework to describe the alerts correlation process has been suggested. Based on it, all the stages, techniques, and methodologies that have been suggested in the state-of-the-art have been also surveyed. In addition, a comprehensive study reviewing the most important existing alerts correlation tools,

open source and commercial, that are currently available has been carried out. Besides, we have analyzed their alignment with both the state-of-the-art and the proposed alerts correlation process.

3. A tickets correlation model has been contributed. It is targeted at improving the information provided by an ITS, aimed at reaching the ideal situation at which just a single ticket per incident exists. Furthermore, some metrics to measure the existing redundancy in an ITS have also been suggested, mainly based on the proposed correlation method.

4. A model for the joint correlation of tickets and alerts has been suggested and evaluated. In addition, the most important challenges that should be faced in this process and the possible solutions have also been discussed and addressed in detail.

5. Three possible application fields of the proposed models have been analyzed, *i.e.*, in the ITS assessment, in measuring staff efficiency during the incident management process, and in reducing the number of resulting alerts in the alerts correlation problem.

Published Works

Part of the work presented along this thesis has been published in international refereed journals and is already available for the research community.

1. S. Salah, G. Maciá Fernández and J. E. Díaz Verdejo. "A Model-based Survey of Alerts Correlation Techniques", Computer Networks (Elsevier), Vol. 57, pp. 2718-2732, 2013.

2. S. Salah, G. Maciá Fernández, J. E. Díaz Verdejo and Leovigildo Sánchez-Casado "A Model for Incident Tickets Correlation in Network Management", Journal of Network and Systems Management (Springer), in Press available online, 2015.

3. S. Salah, G. Maciá Fernández and J. E. Díaz Verdejo "A Model for the Joint Correlation of Tickets and Alerts in Network Management". Submitted to the Journal of Network and Computer Applications (Elsevier), 2015.

1.9 Thesis Document Structure

This thesis document is divided into six chapters besides this introduction. They are listed below followed by a brief summary of their contents.

- **Chapter** 2 gives an overview of the current state-of-the-art in alerts correlation techniques, providing basic concepts, theory, proposed models and frameworks, and sources of information considered by different authors in the alerts correlation process. In addition, it also provides a new taxonomy for alerts correlation techniques that covers proposals coming from different research disciplines. Finally, we analyze existing alerts correlation techniques and make a comparative study among them.

- **Chapter** 3 describes the tickets datasets in more detail, including the extraction of some useful statistics, figures and the different types of records. Then, the same is done for the alerts datasets. Finally, this chapter discusses the preliminary results that have be derived after implementing some experiments that have been carried out for analyzing the process of the joint correlation of tickets and alerts, and it describes the main difficulties that could appear during the process of relating tickets, alerts, or both together.

- **Chapter** 4 proposes a framework to describe the alerts correlation process that is capable of covering all the alerts correlation aspects discussed in Chapter 2. It covers in more detail the different correlation components, their tasks and the interconnection among them. Furthermore, it provides a comprehensive comparison of existing commercial products that implement certain correlation techniques. Three different applications for alerts correlation have been considered here, namely NMSs, network and system security, and SCADA systems, and their alignment with the proposed alerts correlation process model was shown.

- **Chapter** 5 first presents a novel model for incident tickets, that mainly focuses on the generic fields that appear in any ticket. A second model, based on the previous one, is for the incident tickets correlation process. It mainly focuses at merging redundant tickets to achieve the ideal situation at which just a single ticket per incident exists. Finally, in this chapter we can find a description of the case study used to validate the proposal by using some suggested performance metrics.

- **Chapter** 6 describes a model for the joint correlation of tickets and alerts. A detailed discussion of the different issues raised during this process is done, mainly focusing on the database specificities. It also gives details about the evaluation carried out by presenting a case study that consists of a database of tickets and alerts taken from a real IT management company.

- **Chapter** 7 details the analysis of some possible application fields of the proposed models suggested in Chapters 5 and 6, making special focus on alerts reduction and evaluating staff performance in handling incidents.

- Finally, **Chapter 8** draws the general conclusions and summarizes the most relevant parts of the thesis.

Chapter 2

State of the Art in Alerts Correlation Techniques

2.1 Introduction

Alerts correlation is a process used in many fields of applications that analyzes the alerts produced by managed devices of these applications and provides a more succinct and high-level view of them in order to achieve several purposes. A thorough review of the literature over the past few decades reveals that the majority of the research efforts carried out in alerts correlation have mainly been originated from three main applications: NMS, network and system security and SCADA (process control systems). NMS is considered as one of the major applications of alerts correlation, and it is the main focus of this work. It has been extensively used by the research community to group and correlate alerts that have same root causes. Alerts correlation is also vastly applied in the network and system security field, considering here the alerts produced by security devices and applications, *e.g.*, *Intrusion Detection System*s (IDSs) or firewalls logs. Here, the aim is to produce compact attack reports of the security status of a managed network without loosing security relevant information. Finally, as in the NMS, the main purpose of the alerts correlation in SCADA systems is to help in speeding up the fault diagnosing process and discovering disturbances in production lines quickly.

Some of the existing alerts correlation techniques still rely at some point on a manual processing, and depend on the expert knowledge of the members of a management staff. As a huge amount of alerts might be generated (alert floods), it could be very difficult for a management staff to manage them in a short period of time. For this reason, recent and ongoing research efforts are going ahead in this discipline trying to find efficient solutions for the alerts correlation problem in terms of scalability and complexity. In this chapter, we review all this related work and propose a new taxonomy for the alerts correlation techniques. Based on it, we also make a comprehensive literature review providing a global insight of the different efforts implemented in this field during the last few decades. Unlike the majority of existing surveys in this field, which are biased to the network and system security field, in this state-of-the-art, the scope was extended to cover many proposed alerts correlation techniques coming also from both NMSs and SCADA systems.

This chapter is structured as follows: Section 2.2 reviews the proposed surveys, models and frameworks related to the alerts correlation problem. Some concepts related to the alerts correlation process are discussed in Section 2.3. In Section 2.4, a review of the different sources of information considered by different authors in the correlation process is provided. An alerts correlation taxonomy is proposed in Section 2.5. Finally, a comparative study of the existing alerts correlation techniques is presented in Section 2.6.

2.2 Existing Alerts Correlation Surveys

In a thorough review of the literature we have found a considerable number of research efforts trying to study the nature of the alerts correlation problem as a whole. Some researchers [33–35] made a very comprehensive review of the state-of-the-art in alerts correlation techniques and listed some of the existing tools proposed by that time both for IDSs and NMSs. Others presented surveys of alerts correlation techniques [36, 37] based on different proposed frameworks that consist of several components, and covered the most recent techniques by that time. Regretfully, these works are targeted to research proposals done only in the IDS field.

Elshoush *et al.* [38] focused on correlation algorithms proposed in collaborative intelligent IDSs, and showed that current correlation techniques are inefficient. They pointed out the necessity of some artificial intelligence and fuzzy logic techniques to satisfy the growing demand of reliable, intelligent and flexible IDS. The same conclusion was derived by Mirheidari *et al.* [4], that is, there were no optimal solutions to solve the alerts correlation problem completely. They concluded that each category of algorithms has its own advantages and disadvantages, and an ideal correlation framework should leverage the strongest features of each category. Their work was mainly focused on algorithms in correlation engines that can work in practical enterprise networks.

In a more recent work, Hubballi *et al.* [39] made a state-of-the-art of the existing false alarm minimization techniques in signature-based network IDS; and Beng *et* *al.* [40] compared existing alerts correlation models in IDS based on: *(i)* the considered attack scenario (single packet or multi-stage attack), *(ii)* their architecture (either centralized or distributed), *(iii)* their accuracy for alert detection, *(iv)* the load imposed by processing, and *(v)* the data required for testing purposes. They concluded that a more flexible and intelligent IDS is required to complement current IDSs by predicting the incoming alerts at sensor level and real time.

For the time being, it can be concluded that all of these contributions surveying alerts correlation techniques mainly share the following limitations:

- They are biased to a specific application domain, normally management or security, the security field being the dominant one.
- The existing alerts correlation models proposed in these cited works did not take into account all the sources of information used by different authors (as it will be discussed in Section 2.4).

In this chapter, an updated and comprehensive state-of-the-art in alerts correlation techniques was made to cover the above limitations. In addition, a new taxonomy of the existing alerts correlation techniques was contributed, and as previously mentioned, this taxonomy covers proposals coming from the three main different applications: NMSs, network and system security and SCADA systems.

2.3 Concepts

The term "correlation" has been utilized in many diverse applications such as science, engineering, biomedical and business, among others, and it has been assigned to many definitions such as that in the work by Pouget and Dacier [33]: *"an action to carry back relations with each other"*. Others as [41] make a formal definition of it as *"a measure of the relation between two or more variables"*. In general, correlation can be useful for giving a predictive degree of the relationship among features, that can be exploited in practice.

Alerts correlation (also referred to in the literature as alarms or events correlation) is one variant of correlation. It is a widely accepted technology for dealing with events coming from many applications and disciplines, *e.g.*, management information coming from large and complex telecommunication networks. The alerts correlation process has also many definitions, but the most used was given by Jakobson and Weissmann in [25]: *"a conceptual interpretation of multiple alarms such that new meanings are assigned to them"*. Gardner and Harle in [27] defined it as *"the interpretation of multiple alarms so as to increase the semantic information content associated with a reduced set of messages"*. In these contexts, alerts are short messages with a specific textual format defined by vendors, and triggered as external manifestations of a potential failure or a disorder occurring in any network element, service or system of the managed network.

The following example illustrates the benefits of applying alerts correlation in NMS. This example is essentially associated with the fault diagnosing process (Figure 1.3). A failure in a network element[1] usually generates many alerts. Yet, only one of them might be considered the indication of the root cause. This is due to the fact that a failure condition of one network element may render other elements inaccessible. In this situation, the polling agents are unable to access the element that has the failure condition, and this motivates that other elements are also affected by this situation. Alerts are then triggered by all of these affected elements indicating that all of them are inaccessible. Thus, in this scenario, what the management staff needs is a single root cause alert. In summary, the need for alerts correlation in network management comes from different reasons:

- Alerts are triggered as a response to the detection of malicious activities, normal management activities, or real network faults. Therefore, they typically contain descriptive information about the faults and their symptoms. Yet, they do not usually include explicit information about the root causes of a fault.
- Management staff in an IT service provider or an IT department of an enterprise may receive hundreds of alerts per day. Most of them might have been triggered as a response to normal-behavior events, *i.e.*, software update or maintenance activities. Therefore, it is necessary for them to filter out irrelevant alerts and focus only on fault-related ones.
- In distributed and collaborative managed systems, the diversity and heterogeneity of networking elements pose a significant challenge to the management staff, due to the difficulty of converging alerts coming from multiple data sources in order to develop coherent management strategies.

The promising technique that alleviates the above problems and translates alerts into more understandable and thus usable information is alerts correlation.

2.4 Sources of Information

In this section, the data sources that could be used in the alerts correlation process will be discussed. The alerts triggered by monitoring systems are first considered, as they constitute the main source of information and they should obviously be present in any alerts correlation system. Yet, there exists many other sources of information that can greatly contribute to the correlation analysis. In this sense, different authors

[1]The term network element is used to refer to a device or a service.

have proposed the use of a wide variety of data sources for information in order to achieve their goals effectively and accurately. Here, an effort to review the most relevant data sources that have been proposed for the correlation process is made.

2.4.1 Alerts Database

This database contains the alerts triggered by the network elements and detection systems in the monitored environment. Alerts are considered the main source of information for any correlation technique. They can either be triggered by network management agents via management protocols through various mechanisms such as traps (generated by SNMP), event reports (generated by CMIP) [24], or by IDSs such as Snort [42] or GrIDS [43], or even by monitoring tools in SCADA systems.

2.4.2 Topological Information

The second most used sources of information in the alerts correlation process are topology databases. The main purpose of topology information is to provide an accurate representation of the monitored environment as a set of links and elements, where the links are used to represent relationships between the elements. The representation of the location of elements, and the direction and connectivity of links are of particular relevance. The topology information contains extensive details of elements structure such as switches, routers or servers, among others; configuration parameters such as IP addresses and their matching to names, subnets, virtual *Local Area Network*s (LANs); and host information such as operating system type and open services. This information is typically gathered by polling agents and stored in a database. For example, the authors in [44–46] used topological information to correlate alerts in their proposed alerts correlation engines. They deployed agents to collect such dynamic topological information.

In a more recent work, Calyam *et al.* [47] proposed a novel topology-aware scheme that can be integrated into perfSONAR monitoring dashboards [48] for detection and diagnosis of network-wide correlated anomaly events across multiple domains. The proposed scheme has two main parts: an adaptive plateau detector to generate anomaly events with low false alarms rate, and spatial and temporal analyses that are applied and combined with topology information to detect correlated events.

2.4.3 Vulnerabilities Database

This source of information has been mainly considered for its use in IDS. It stores all well-known exploits and system vulnerability information, usually with the corresponding security solutions. Like in the topology database, it is built by collecting the configuration information of the monitored resources, such as operating systems or network application services potentially susceptible to be exploited by attackers. A clear example of using this information to solve the alerts correlation problem is the *Common Vulnerabilities and Exposures* (CVE) project [49], that is a well-known vulnerabilities database, free for public use. CVE is a dictionary of publicly known information about security vulnerabilities and exposures. Gula [50] illustrated how vulnerabilities information can elicit high quality alerts from a huge amount of alerts that are primarily false alerts. Also, besides the topological information, Jinqiao *et al.* [45] used CVE, bugtraq [51] and CERT [52] vulnerability identifications for categorizing and sorting vulnerabilities in their proposed collaborative IDS called TRINETR. For each vulnerability reference, there is an associated rule to specify the corresponding evaluation process or action. Porras *et al.* [53] presented M-Correlator, a comprehensive framework for IDS alerts correlation. The system correlates alerts based on well-known vulnerabilities.

2.4.4 Incident Ticketing System (ITS)

*Incident Ticketing System*s (ITSs) are the main workflow tools extensively used by management staffs to track and report ongoing and resolved faults. In several contexts, they have been introduced to assist in speeding up the fault recovery process and adding more advanced functions to maintain the networks [30, 54]. Here, ITSs store tickets, that are usually generated either automatically by management tools like network management platforms or manually by the management staff that creates tickets as a response to the reception of network alerts, or by the service desk, as a consequence of customer calls.

Many of the records in ITSs contain information related to faults generated by events identified as network failures. This makes it desirable to integrate tickets in the process of alerts correlation, as they could implicitly act as information provided by an human expert. The incorporation of this new information into an alerts correlation system would permit to alleviate management staff from decisions, as well as it would allow to improve, speed up and prioritize the diagnosis of faults. Lewis and Dreo [31] emphasized the importance of ITS, and clearly described these research trends suggesting an extension of the ITS framework to provide advanced functions in faults resolving and alerts correlation. Furthermore, Costa *et al.* [55] used ITS information to get feedback from their proposed alarms correlation architecture, that was an adaptive and self-maintained alarms correlation system.

2.4.5 Ontology Database

Ontologies provide powerful constructs and constitute useful tools to deal with such diverse knowledge as that coming from alerts. They include machine interpretable definitions and formal specification of the concepts and relationships that can exist between entities within a domain [56]. As an example of an application in our field of study, Xiao *et al.* [57] proposed an alerts correlation approach composed of two parts. First, a new alerts ontology was constructed with the ability to share and reuse information better and store data in a more reasonable way. Second, they used attribute-based similarity methods and semantic distance when computing the similarity between alerts. Li and Tian [58] proposed an intrusion alerts correlation system based on an ontology knowledge base, and introduced modules for reducing redundant alerts to attack actions, using *Intrusion Detection Message Exchange Format* (IDMEF) [59] and CVE standards. According to alerts information and attacks knowledge, they used the reasoning power of ontologies to infer the correlation of alerts. The same was also done by Coppolino *et al.* [60] who presented an ontology-based approach for correlating IDS attack symptoms coming from diverse information sources, and collected at different architectural levels.

2.4.6 Cases Database

This source of information is mainly used in the analysis of event and error messages. It stores cases or scenarios, each one representing a complete description of a known fault, described by two different elements: situation and solution. The situation describes the context of the case, and consists of associated alerts; and the solution gives reasons about why the fault has occurred and describes the suggested steps to solve it. Each case consists of a set of patterns that can be matched with a list of alerts. *E.g.*, Long *et al.* [61] proposed a case-based reasoning approach for alerts correlation. In this approach, the authors used a cases database to store all cases that were constructed from training data. Also, Holub *et al.* [62] used a cases database in their proposed run-time correlation engine to analyze log data and to provide a mechanism for matching known faults in large volumes of data.

2.4.7 Knowledge Representation

An additional source of information is related to the use of rules or models that somehow represent the relationships among alerts. These rules or models can be explicitly set by experts or inferred from the analysis of the alerts by means of learning procedures, usually from labeled samples (supervised learning). This way, a knowledge representation allows to incorporate human knowledge in the alerts correlation procedure or somehow mimic this knowledge. One of the most common ways to express this information is through the use of expert rules. Most experts are capable of expressing their knowledge in the form of rules for faults solving tasks. The database includes a set of facts used to match against the IF (condition) parts of rules stored in the knowledge-base. Other methods are related to the pattern learning field, in which the knowledge is usually represented by a model (*e.g.*, neural networks, Markov models, Bayesian networks). As an example of its use in the security field, Kabiri and Ghorbani [63] proposed an intrusion alerts correlation system using a rule-based inference engine to derive the correlation between alerts (using an inference engine and a working memory that constitutes an expert system). The inference engine was implemented using a scenario-based knowledge base, and the extraction of attack scenarios was performed by a security expert, before being stored in a knowledge-base to become operational.

2.5 Taxonomy of Alerts Correlation Techniques

After a thorough review of the literature, several efforts focused on providing taxonomies for alerts correlation techniques have been found. Most of them have adopted a classification criteria based only on the used correlation methods [4, 34–37]. For this reason, we consider that they are narrowed on a small picture with limited scope. Thus, in this section we try to provide a global view of the alerts correlation problem, taking into account additional aspects and not only the correlation methods, *i.e.*, the number of data sources, the application field of the correlation techniques, and the architectural design of the system. The suggested taxonomy is presented in Figure 2.1. In what follows, the scope of these aspects is described below, giving more detailed information.

2.5.1 Number of Data Sources

Alerts correlation techniques can be classified according to the number of used data sources. They can either accept the data from one input, *i.e.*, a single data source, or multiple inputs, *i.e.*, more than one data sources (among those described in Section 2.4).

Single Data Source Systems

Single data source systems are those in which the data comes from a single type of sources of information. Note that this does not mean that the data should arrive from a single network element. For example, an alerts database itself is considered as a single source of information despite alerts might be coming from various network elements with different formats and natures.

Single source correlation techniques are usually built in for specific purposes and applications. Although their main advantage is their simplicity, they do not achieve optimal results from the correlation and they are not the best solution for distributed and collaborative monitoring systems. Most of the existing commercial alerts correlation tools that are listed in the tables in Section 4.3 use alerts databases (Column 4) as the only source of information.

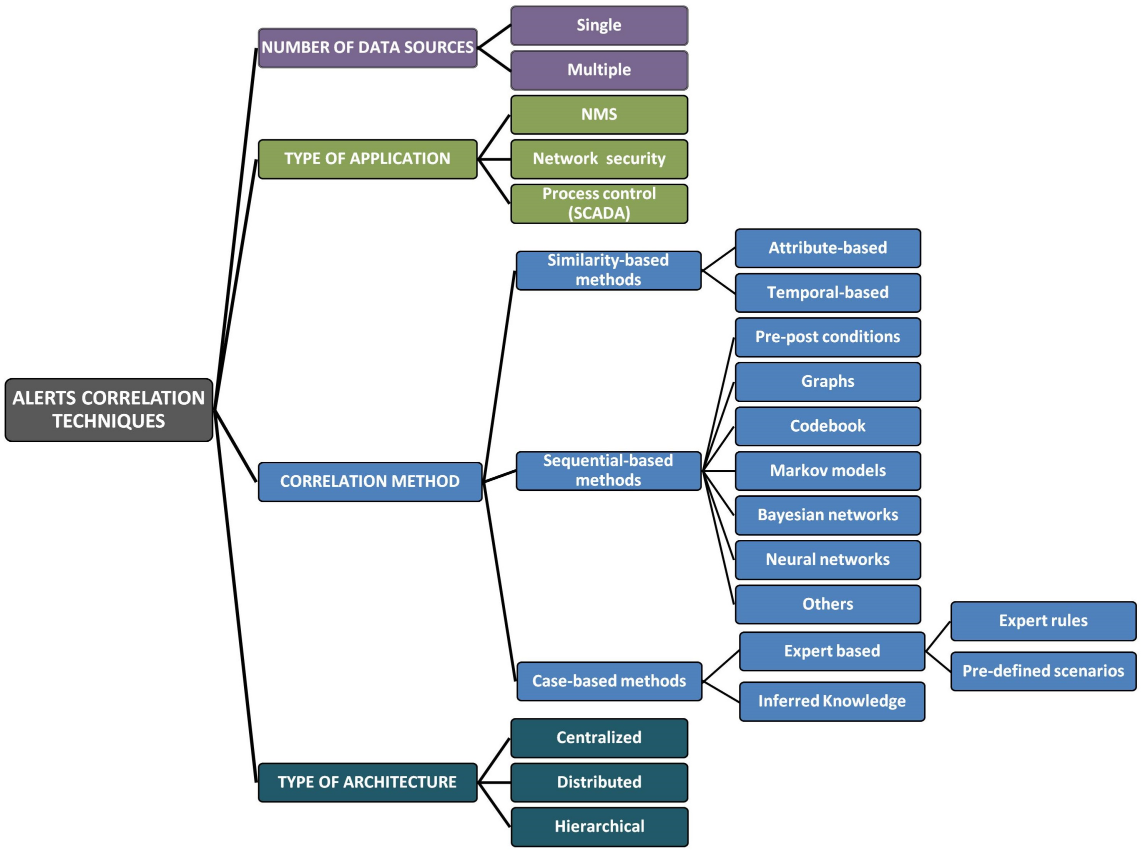


Figure 2.1: A comprehensive taxonomy of alerts correlation techniques.

Multiple Data Sources Systems

In contrast to the case of single source systems, most of the proposed alerts correlation techniques that depend on multiple-sources of information comes from scientific research projects and not from the industry. As current monitoring systems become more and more collaborative, there is a need for more advanced correlation methods with better results in order to cope with this complexity in the managed systems. Obviously, the cost of obtaining better results when multiple data sources are used is a higher complexity in the alerts correlation systems, mainly due to the heterogeneity of the different inputs. Moreover, they need extra amount of resources when compared with single data source techniques.

Many significant examples of these techniques appear in the scientific literature [46, 64–66]. These contributions proposed alerts correlation techniques considering more than one type of data sources, namely alerts database, topology, and vulnerability information.

2.5.2 Type of Application

Typically, and as mentioned above, existing alerts correlation techniques are implemented towards one application. Despite their potential use in many other fields, we have detected three main fields of application where these techniques have been proposed and evaluated: NMSs, network and system security, and SCADA systems.

Network Management Systems

NMSs are used by a management staff to carry out several management tasks such as designing and configuring the network settings and functions; fault management, which deals with faults, their effects and the solutions; performance management, which provides some performance metrics of the network status; and security and accountability management. As explained in Section 1.3, network management protocols are used for the interaction between network elements and the NMS, SNMP [22] being the dominant. Recall that agents are processes running on managed devices that collect information and send it as alerts (traps) to the manager, where further processing is done before showing the information in a console. Alerts correlation is mainly applied here to help the management staff in real-time diagnosis and to speed up the faults diagnosing process. They have been extensively used by the research community to group and correlate alerts that have same root causes [55, 62, 67, 68].

Network and System Security

It is another important application field of alerts correlation. Here, it is used for building a consolidated security picture of the whole monitored system. Alerts are typically triggered by security elements such as network IDSs, host IDSs, firewalls and anti-viruses, among others, as a response to discovering malicious or simply anomalous activities. A high detection sensitivity of these security elements will generate a massive amount of alerts, where some of them could really correspond to normal-behavior events that are mistakenly considered as attacks (*False Positive*s (FPs)). In this regard, alerts correlation helps security experts to verify the validity of those alerts, and to build more succinct and high-level view of occurring or attempted intrusions to detect complex or multi-step attack scenarios.

The main objective of alerts correlation techniques in this field is to produce complete attack reports that coherently capture the set of activities on the monitored system without losing security-relevant information. Thus, [45, 69–71] are examples of alerts correlation techniques proposed in the network and system security field.

SCADA (Process Control Systems)

SCADA systems are also considered as another relevant application field for alerts correlation. In most manufacturing systems, there are a lot of switches, sensors and actuators that could generate a huge amount of alerts in response to process disturbances, attacks or failures. They are mainly used here to monitor and control industrial, infrastructure or facility-based processes. Since manufacturing applications increase in complexity and scale, SCADA systems should incorporate efficient mechanisms to identify the root cause of faults or processes disturbances. This is essential for not delaying the decision making process. Existing alerts management systems are really improved with the help of alerts correlation techniques, and a number of research projects have been carried out to handle this issue. The works in [72–74] are examples of alerts correlation techniques proposed in the field of SCADA systems.

2.5.3 Correlation Method

As previously mentioned, over the past few decades, researchers and vendors, in a joint effort with networking experts, have suggested many proposals coming from different design approaches to solve the alerts correlation problem. Nevertheless, alerts correlation is a complex multi-step transformation process, and the bulk of the existing proposals operate only on partial aspects of the correlation process with different correlation methods such as: alerts filtering, alerts aggregation and alerts correlation. Here, a classification of the different techniques proposed in the field of alerts correlation was made, that is based on the used correlation method. Yet, instead of focusing on the mathematical tools or mechanisms used for the correlation like others, we put our attention on the strategy followed by the different authors to correlate alerts. Thus, three major categories have been identified: similarity-based, sequential-based and case-based methods. For every of these categories, in the following, a more detailed description, along with a survey of relevant proposed research contributions will be provided. Table 2.1 shows a summary of these contributions.

Similarity-based Methods

Similarity-based techniques aim at reducing the total number of alerts by clustering and aggregating them using similarities in their attributes. Each triggered alert has several associated attributes or fields, *e.g.*, in NMS an alert contains these basic attributes: source and destination IP addresses, source and destination port numbers, protocols, alert description and timestamps information, among others. The main assumption behind this method is that similar alerts tend to have the same root causes or similar effects on the monitored system. How to define similarity measures is the key factor and plays a critical role for such kind of techniques. To

Alerts correlation techniques				
Classification				**References for contributions**
Number of data sources	Single			As illustrated in column 4, Table 4.1, Table 4.2, Table 4.3
	Multiple			Zhuang *et al.*[64], Chang *et al.*[65], Hu *et al.*[66], Chyssler *et al.*[46]
Type of application	NMSs			Costa *et al.*[55], Gruschke *et al.*[67], Holub *et al.*[62], Klinger *et al.*[68]
	network and system security			Jinqiao *et al.*[45], Alserhani *et al.*[69], Qin *et al.*[70], Valeur *et al.*[71]
	SCADA (process control systems)			Chen *et al.*[72], Carcano *et al.*[73], Sayegh *et al.*[74]
Correlation method	Similarity-based methods	Attribute		Valdes *et al.*[26], Lee *et al.*[75], Siraj *et al.*[76], Zhuang *et al.*[64], Debar *et al.*[77], Julisch *et al.*[78], Julisch *et al.*[79], Cuppens [80]
		Temporal		Jakobson *et al.*[81], Ma *et al.*[82], Qin *et al.*[83], Morin *et al.*[84], Kelkar *et al.*[85], Zhu *et al.*[86], Ahmadinejad *et al.*[87], Bateni *et al.*[88]
	Sequential-based methods	Pre/Post conditions		Zhaowen *et al.*[89], Ning *et al.*[90], Xiao *et al.*[91], Alserhani *et al.*[69], Alserhani *et al.*[92]
		Graphs		Gruschke *et al.*[67], Roschke *et al.*[93], Wang *et al.*[94], Li *et al.*[95], Jian *et al.* [96]
		Codebook		Yemini *et al.*[97], Klinger *et al.*[68]
		Markov models		Ourston *et al.*[98], Farhadi *et al.*[99], Xin *et al.*[100], Zhicai *et al.*[101]
		Bayesian networks		Steinder *et al.*[102], Qin *et al.*[70], Marchetti *et al.*[103], Harahap *et al.*[104]
		Neural networks		Zhu *et al.*[86], Zhou *et al.*[105]
		Others		Lagzian *et al.*[106], AlMamory *et al.*[107], Alsubhi *et al.*[108]
	Case-based methods	Expert based	Expert rules	Cronk *et al.*[109], Lor [110], Jector *et al.*[111]
			Pre-defined scenarios	Cuppens *et al.*[112], Kemmer *et al.*[113], Eckmann *et al.*[114], Liu *et al.*[115], Cheung *et al.*[116]
		Inferred knowledge		Agrawal *et al.*[117], Lagzian *et al.*[106], Smith *et al.*[118], Katipally *et al.* [119], Sadoddin *et al.*[120], Tongyan *et al.*[121], Jian *et al.*[122], Sizu *et al.*[123]
Type of architecture	Centralized			Jinqiao *et al.*[45]
	Distributed			Mohamed *et al.*[124], Khatoun *et al.*[125]
	Hierarchical			Tian *et al.*[126], Qin *et al.*[127]

Table 2.1: Taxonomy and summary of proposed examples of alerts correlation techniques.

answer this question, several similarity measures have been proposed by many researchers, and some of them are discussed below. The aim of this is to define the suitable similarity function for each attribute, because attributes may have different weights and effects on the correlation process and may need different criteria to calculate these weights. *E.g.*, the similarity between two alerts having the same IP address might be different if they share the same subnetwork address or the same port number.

Techniques that belong to this category exhibit many advantages. First, they are usually implemented with lightweight algorithms with lower complexity than those in other categories, mainly because these algorithms are based on simple logical comparisons. Second, this category has proven its effectiveness in reducing the total number of alerts, which is an essential step in the correlation process, given the usually large number of alerts reported to a management staff. Third, they have high correlation accuracy. Last but not least, they are efficient in the handling of false alerts. However, these techniques also have some weaknesses. The most important one is that they simply work on the attributes level and cannot detect causal relationships between alerts, in order to discover the root causes for the faults.

Similarity-based correlation techniques can be grouped into two categories: those based on attributes similarities and those based on temporal information. Next, the most common proposals followed by different authors in both groups are described.

Attribute-based: Attribute-based similarity correlation techniques correlate alerts by defining similarity measures between some of their attributes or features. Here, several different attributes have been used, like source and destination IPs, timestamps, ports, kind of service or users, among others. A similarity measure is typically calculated by computing certain metrics, such as Euclidean, Mahalanobis, Minkowski or Manhattan distance functions. The resulting scores, when compared with threshold values, determine if these alerts are to be correlated or not. Choosing the suitable distance measure might increase the overall performance of the correlation process, as two alerts might be close or far depending on the considered distance function.

A lot of contributions using these techniques have appeared in the literature, as they are the most widely deployed. Indeed, there exist a lot of variations in the nature of the applied technique, despite the use of a certain similarity metric for the correlation. Some relevant examples of these variations are described next.

Valdes and Skinner [26] presented a probabilistic method to correlate alerts based on a proposed mathematical framework that is able to find the minimum similarity specification to fuse alerts from multiple sensors. The method considered appropriate attributes contained in alerts reports as features for a multivariate matching algorithm. Only those features that overlap are taken into account for the overall similarity calculation. For every matching feature, they defined an appropriate similarity function with range zero (mismatch) to one (perfect match), and the overall similarity is calculated using a predefined equation. Alerts are correlated with a high degree of attribute similarity if there is a match; otherwise, a new thread is generated. Depending on the situation, they incorporated the expectation of match values (which are used to compute a weighted average of the similarity over the overlapping features), as well as a minimum match specification that unconditionally rejects a match if any feature fails to match at the minimum specified value. For each new alert, they computed the similarity for existing meta alerts, and merged the newly created alert with the best matching meta alert, as long as the match passes a threshold value. They realized a reduction of one-half to two-thirds in alert volume in a live environment.

Other approaches like [64, 75–77] adopted different similarity metrics based on the Euclidean distance, pre-assigned different similarity scores for every attribute or other mechanisms such as alphabetical, bit-by-bit, and MAX value comparisons, respectively.

On the other hand, Julisch *et al.* [78, 79] proposed the principle of dissimilarity measure instead of similarity. They defined a dissimilarity function that takes two alerts as input, and returns a numerical value that indicates how adequately these alerts can be modeled by a single generalized alert. Here, dissimilarity is inversely related to similarity, *i.e.*, if the numerical value is very small the two alerts have higher correlation and they can be modeled by a generalized alert. They tried to use a wide variety of attribute types, including numerical, categorical, time and free-text attributes to aggregate alerts.

Instead of using mathematical formulas for calculating the similarity measures, Cuppens [80] defined the similarity relationship by using expert rules applied for four selected attributes: classification, time, source and target. For every attribute they designed a set of expert rules to make the aggregation and correlation.

Temporal-based: Besides using attributes similarity information, temporal-based similarity techniques [81–85] use some form of temporal time constraints to find the relationships between alerts in specific time periods. The idea behind temporal-based similarity alerts correlation techniques is to recognize that alerts caused by the same fault are likely to be observed within a short time after the fault occurrence. The simplest method of temporal correlation relies on time-windows, where only alerts occurring within a time-window are to be correlated. Two alerts are correlated if their temporal similarity is higher than a predefined threshold. Correlated alerts are then signaled as hyper-alerts.

Here, most of the proposed systems uses pairwise alerts correlation in which each new alert is checked with a number of previously received alerts to find possible correlations. To speed up the checking process, some alert selection policies are defined to control the way in which this checking is done. Thus, different window-based selection policies are proposed by different authors such as: select all [86], window-based random selection [87], and random directed selection [88]. Window-based selection policies use a limited time window with a number of sliding time slots, and select alerts from this time window for checking with the current alert.

The main advantage of the temporal alerts correlation is the speed in the correlation process, as it reduces the number of alerts triggered by the management devices. However, the main disadvantage of these approaches is that they are deterministic, what limits their applicability.

Sequential-based Methods

At a first glance, it is observed that the bulk of the work done in this category was restricted to the network and system security field. We consider that the main reason is that these methods are useful to model and analyze complex attack scenarios from the sequence of individual events or steps that are a part of the same attack and they use the causality relationships among alerts. Here, pre-conditions are defined as the necessary requirements that must exist for an attack to be successful, and the consequences of the attack are defined as the effects that appear after a specific attack or a part of it has occurred. This relationship is mainly represented as a logical formula using combinations of predicates of logical operators such as AND/OR connectives.

The main advantages of the techniques belonging to this category are: First, they are scalable in terms of their ability to potentially uncover the causal relationship among alerts, not being restricted to known attack scenarios. Second, the correlation results are easy to understand and directly reflect the possible attack scenarios. On the other hand, their accuracy is low and the correlation results may contain a large number of false correlations, this being for two possible reasons: either the logical predicates are not well configured or the quality of the sensors' alerts is not adequate.

Sequential correlation can be subdivided into several major categories, depending on how they represent the modeled scenarios: pre/post conditions, graphs, codebook, Markov models, Bayesian networks, neural networks, and other techniques. We give a more detailed discussion about each one of them in what follows.

Pre/Post conditions: In this category, the correlation process tries to find causal relationships among alerts through their pre and post conditions. The main assumption here is that older alerts prepare for the later ones. If post conditions of an alert satisfy the pre-conditions of another alert, they are correlated. As an example, suppose an attack against `sadmind`, a remote administration tool. A scanning attack may discover *User Datagram Protocol* (UDP) services vulnerable to certain buffer overflow attacks. Then, the predicate *UDPVulnerableToBOF (VictimIP, VictimPort)* can be used to represent the attacker's discovery (the consequence of the attack), *i.e.*, that the host having the *Internet Protocol* (IP) address *VictimIP* runs a `sadmind` service at UDP port *VictimPort* and that the service is vulnerable to the buffer overflow attack.

Many proposals have been suggested in this category. Zhaowen *et al.* [89] proposed RIAC, a real time alerts correlation system that employs distributed agents to collect alerts information on-line and adopts a pre/post correlation method to analyze and discover attack scenarios and intrusion intents behind alerts. The assumption here states that the components of the attacks are usually not isolated, but related at different stages of the attacks, with the early ones preparing for the later ones. By using logical predicates, they introduced the notion of hyper alerts. Each hyper-alert represents the prerequisite and the consequence of each type of alert, and consists of a tuple (*fact*, *prerequisite*, *consequence*), where *fact* is the set of alerts attribute names, and *prerequisite* and *consequence* are two different sets, each one consisting of a logical combination of predicates expressed as mathematical conditions on variables contained in the set fact.

Ning *et al.* [90] also published a similar work. They presented TIAA, a toolkit for constructing attack scenarios by using predicates as the basic constructs to represent the prerequisites and (possible) consequences of attacks. The main difference between TIAA and JIGSAW, a correlation tool proposed by Templeton and Levittby in [128], is that the former allows partial satisfaction of prerequisites, while the later requires all the capabilities being satisfied to correlate alerts.

Xiao *et al.* [91] proposed an alerts correlation model based on attribute-based similarity and prerequisites and consequences of attacks to fuse alerts that can reduce redundant alerts and recognize complicated attack scenarios. To do that, they firstly made use of a fuzzy clustering algorithm to divide the alerts in approximately equal sets, and then they adopted the method of correlating alerts based on prerequisites and consequences of attacks.

Alserhani *et al.* [69] developed a rule-based correlation language termed MARS, Multi-stage Attack Recognition System, based on the phenomena of cause/effect that is basically used in plan recognition models. Later on, the same author with others proposed in [92] an alerts correlation and aggregation framework based on a requires/provides model. Their main objective in this work was to discover the logical relationships between atomic alerts potentially incorporated in multi-stage attacks.

Graphs: In graph-based alerts correlation techniques, the relationships among alerts can be represented as a directed acyclic graph where the set of nodes represent alerts and the edges connecting those nodes represent the temporal relationship of the connected alerts (nodes) [129]. The purpose of mapping alerts into graphs is to collect the sequential information among them.

This category of techniques has several advantages. First, graphs are relatively easy to generate from whatever management models, especially from object-oriented system models with relations or associations between objects. Second, the operations permitted on graphs can be implemented in a robust manner, *e.g.*, adding or deleting objects and dependencies are easy tasks. Third, graphs are naturally manageable in a distributed manner, as objects and dependencies can be added or deleted by different management staffs independently. However, these techniques require an accurate knowledge of current dependencies among abstract and physical system components. Therefore, their efficiency and accuracy mainly depend on a prior specification of how a failure condition or alert in a given monitored system is related to failure conditions or alerts in other systems.

A clear example of using graphs to solve the alerts correlation problem is illustrated in [67], where the authors proposed an events correlation approach for faults localization in NMS based on dependency graphs. It consists of two components: nodes or objects, which reflect the managed objects in the system, and edges, that collect the functional dependencies between the managed objects. Two objects are correlated if a failure in one of them causes a failure in the other. The proposed algorithm works as follows: First, each received alert is mapped to its corresponding object, which is signaled as faulty in the dependency graph. Next, a breadth-first searching process is started from the initial dependent objects through the whole dependency graph looking for objects from which all (or many) initial objects depend on. These common dependent objects are forwarded as a condensed event. Finally, the algorithm assigns one of two states to the objects of the dependency graph: faulty or correct.

Instead of using a breadth-first search process like in [67], some authors deployed different well-known searching algorithms in order to find the shortest path between two nodes in a graph such as a Floyd-Warshall algorithm, used in [93], a queue qraph, used in [94], and a bipartite graph, used in [95].

In a more recent work, Jian *et al.* [96] proposed a novel measure for assessing the structural correlations to estimate the correlation score of a node in large-scale graph data sets with events. This measure applies hitting time to aggregate the proximity among nodes that have the same event.

Codebook: Codebook techniques encode the relationship between network faults and their symptoms by creating a matrix of problem codes that represent the dependency among observable symptoms and the underlying problems [97]. The basic idea of this category is that problem events are viewed as messages generated by the monitored system and encoded as a set of alarms that they cause. Thus, the problem of correlation is viewed as decoding these alarms to identify the message. The coding technique proceeds in two phases. In the codebook selection phase, an optimal subset of alarms, the codebook, is selected to be monitored. This codebook is selected to optimally pinpoint the problems of interest and ensure a required level of noise insensitivity. The codebook is basically a matrix representation, where events/alerts are represented as rows, and the symptoms of the problems as columns. The matrix contains binary digits (either 0 or 1). The value of 1 at the *ith* position of the matrix generated for problem *j* indicates a cause-effect implication between problem *j* and symptom *i*. In other words, a one in the matrix denotes the appearance of a particular symptom, and a zero denotes that the symptom has not been observed. Distinction among problems is measured by the Hamming distance between their codes. In the decoding phase, observed alarms are analyzed to identify the problems that caused them.

These techniques are efficient in terms of speed and accuracy in detecting network problems, because they are performed only once to detect the root causes. However, they are not suitable for dynamic networks, because any change in the network topology requires regenerating the codebook, which is a time consuming process. Furthermore, they mainly depend on expert knowledge to construct the codebook matrix, which is also time consuming, error prone and tedious.

Klinger *et al.* [68] described a novel approach to event correlation in networks based on these coding techniques. First, by using the Hamming distance they reduced the size of the codebook to contain a smaller set of symptoms capable of accomplishing a desired level of distinction among problems. They defined two metrics to calculate the distance, one is used for deterministic correlation, where the codebook matrix contains either 0 or 1, and the other for probabilistic correlation, where the codebook matrix contains weights in the range [0-1], representing the probability of having a relationship between symptoms and problems. According to their claims, their approach tries to solve some performance issues that existing codebook techniques faced in that time while dealing with high rates of symptom losses and false alarms.

Markov models: A Markov model is a stochastic production model composed of discrete states and a matrix of state transition probabilities associated with each state [130]. In a particular state, an outcome or observation can be generated according to a separate probability distribution associated with the state. In addition, every state has a vector of observable symbol probabilities. Once that a model is defined and their associated probabilities obtained by training the model, a sequence of events can be evaluated, thus obtaining a probability. This probability is formerly compared to a threshold value to decide if a correlation is present or not. *Hidden Markov Model*s (HMMs) [131] are an important variant where only the outcomes and not the internal states are visible to an external observer.

Most of the implemented alerts correlation techniques in this category are focused on the network and system security field and concentrated on HMM. Ourston *et al.* [98] claimed that HMMs are particularly useful and well-suited to address the multi-step attack problems through its prerequisites when there is an order for the actions constituting the attack (that is, for the cases where one action must precede or follow another action in order to be effective).

Markov-based techniques are especially well suited to address problems with a sequential nature. Yet, the main drawback of these models is the amount of data needed for suitably training them and their dependence on tuning parameters, *i.e.*, the detection threshold for deciding whether an alert should be correlated or not.

Examples such as [99–101] proposed alerts correlation systems in IDS using a HMM that mine the stream of alerts and extract the current attack scenario.

Bayesian networks: Bayesian network models, also known as belief networks (or Bayes nets for short), are one of the most powerful probabilistic graphical models [132]. These graphical structures are used to represent knowledge about an uncertain domain. Bayesian networks are mainly specified by two components: *(i)* A graphical component composed of a directed acyclic graph where vertices represent events and edges represent relations between events; and *(ii)* a numerical component consisting of a quantification of different links in the graph by a conditional probability distribution of each node in the context of its parents. In particular, each node in the graph represents a random variable, while the edges between the nodes represent probabilistic dependencies among the corresponding random variables. A Bayesian network consists of several parameters, *i.e.*, prior probability of parent node's states and a set of conditional probability tables associated with child nodes. The main purpose of the conditional probability table is to encode the prior knowledge between a child node and its parent node.

In the alerts correlation problem, the probabilistic relationships among a large number of alerts are represented in order to work out a probabilistic inference from them. Given certain symptoms (received as alerts), a Bayesian network can be used to compute the probability that a specific problem have happened.

Bayesian networks provide several advantages when applied to solve the alerts correlation problem. First, the speed of correlation is high. Second, they can incorporate prior knowledge and expertise by populating the conditional probability tables. Third, they are convenient to introduce partial evidence and find the probability of unobserved variables. Fourth, they are also capable of being adapted to new evidence and knowledge by updates through network propagation. Finally, the correlation output is a probability, rather than a binary result from a logical combination. However, this method needs a large number of training events to obtain the prior probabilities and the correlation relies on experts' knowledge. Furthermore, a probabilistic inference in a Bayesian network is NP-hard, *i.e.*, efficient solutions for large networks are difficult to implement in practice.

The works in [70, 102–104] are examples of correlation techniques based on Bayesian mechanisms to determine the likelihood of any two alerts for being correlated.

Neural networks: Artificial neural networks are generally presented as systems of interconnected processing elements called neurons working jointly to solve specific problems. The neurons are interconnected to each others according to a model inspired by the neural system existing in the human brain. Each neuron is considered as a simple autonomous processing unit, provided with local memory and unidirectional channels for the communication with other neurons. An artificial neural network is typically defined by three types of parameters: *(i)* the interconnection pattern between the different layers of neurons; *(ii)* the learning process for updating the weights of the interconnections; and *(iii)* the activation function that converts a neuron's weighted input to its output activation [133].

The utility of these models relies on the fact that they can be used to infer a function from observations. Thus, they are typically useful in applications where the complexity of the data or task makes the design of such a function by hand impractical, or to find patterns in data where non-linear dependency exists between inputs and outputs.

The most important issue in artificial neural networks is the learning phase that can be accomplished by continuously adjusting the inter-neuron connection strengths (weights) until the overall network yields the desired results for the observations in the training set. The learning process is based on the iteration over the training set until a satisfactory optimum operating point or a predefined threshold is reached.

Mainly, there are three major learning paradigms, each corresponding to a particular abstract learning task. These are supervised learning, unsupervised learning and reinforcement learning. In unsupervised training, hidden neurons find an optimum operating point by themselves, without external influence. Supervised training requires that the network is given sample input and output patterns to learn. Reinforcement learning differs from supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected.

There are several advantages in using an artificial neural network based approach. First, they can be made tolerant against noise in the input. Second, they have better properties than other techniques to generalize the results. This means that a trained network could classify data from the same class as the learning data that it has never seen before. Third, an artificial neural network can acquire knowledge straight from the data without the need for an human expert to build up sets of domain rules and facts. Fourth, once trained, artificial neural networks can be very fast, accurate and have high precision for near real-time applications. Last but not least, they may also use a type of dimensionality reduction in feeding data during the learning phase, that allows to input large amounts of information without efficiency bottlenecks. Yet, they share some weaknesses. The training process to tune its weights may take long sessions. Moreover, there are no particular rules to guide the selection of the number of layers and the number of neurons in each layer; hence, a trial and error process should be performed during the training period until the network finally stabilizes.

Artificial neural networks have been used to solve the alerts correlation problem in several approaches. *E.g.*, in [86, 105] the authors proposed alerts correlation techniques using artificial neural networks. The first one is targeted to discover attack strategies whereas the second is targeted to handle the faults detection and identification method for systems.

Other methods: Besides the major correlation methods cited above, some research efforts suggesting alternative methods were found such as the use of context-free grammars [107], where the authors proposed an alerts post-processing and correlation method for the detection of multi-step intrusions. They called it Alerts Parser. In this method, alerts are treated as tokens, and a modified version of the Left-to-Right parser algorithm is used to generate parsing trees representing the scenario in the alerts. In addition, they used an attribute context-free grammar for representing the multi-step attacks.

Alsubhi *et al.* [108] used fuzzy logic methods to classify alerts. They mainly proposed FuzMet, a novel IDS alerts management system that takes several metrics as an input such as the applicability of the attack, the importance of the victim, the relationship between the alerts under evaluation and previous alerts, and the social activities between the attackers and the victims, and after that, it generates a report scoring alerts based on their severities.

Case-based Methods

Case-based reasoning methods rely on the existence of a knowledge-base system composed by known scenario templates or past cases, that are expressed either by human intervention using expert rules or correlation languages [109, 110, 112, 116] or inferred by using machine learning or data mining techniques [106, 118, 119]. Thus, any case-based reasoning method can be described by a cyclic system that consists of four processes: retrieval, reuse, revise and, finally, retain methods.

The main two questions here are: which are the key attributes of a case? And which attributes should be used to index and access a case? There are two main ways to adapt past cases: reuse the past case solution (transformational reuse) and reuse the past method that constructed the solution (derivational reuse). Several case matching algorithms have been implemented to retrieve the matched case such as nearest neighbor, inductive and knowledge-based indexing.

When a problem is successfully solved, the solution (or its parts) is stored in a knowledge base, called the base case. When a new case is raised, the system searches the cases database for the most similar cases having the same symptoms. When a matching case is found, its associated solution is retrieved and used to suggest solutions to the current problem. If it is successfully solved, this solution or certain parts from it that are likely to be useful in the future are stored. When an attempt to solve a problem fails, then the reason for the failure is identified and “remembered” in order to avoid a recurrence of such a mistake. Therefore, case-based methods use to keep updating the database with the new observed scenarios through some kind of inference mechanism or expert intervention.

Case-based reasoning methods are efficient in solving well-known problems specifying a complete action plan of previously observed scenarios. Therefore, these approaches can help a management staff to discover all the possible scenarios including potential solutions. However, building a database containing a comprehensive set of problems and solutions is not always an easy task. Furthermore, case-based reasoning methods are not able to identify problems not previously observed. In addition, time inefficiency may make them unusable in real-time correlation systems.

The existing solutions belonging to this method can be grouped in two main categories: expert based and inferred knowledge.

Expert based: Expert based techniques build the knowledge database from human intervention. This knowledge is formulated either by using expert rules or predefined scenarios. They tend to imitate the knowledge of a human, that might be either resulting from experience, or from understanding the system behavior from its principles. One of the main difficulties of this approach is the persistency, because updating the knowledge-base according to the evolution of a system is a problem that has to be taken into account when components are subject to frequent changes (topological or functional).

As said, there are two main possibilities to build the database: using expert rules or pre-defined scenarios. In the following we explain in more details and list some contributions for each one of them.

Expert rules: Expert rules or rule-based systems are one of the most dominant methods among all existing alerts correlation techniques. They have been introduced by many researchers and mostly applied in various commercial correlation systems. This approach develops the knowledge as conditional, *if-then* rules. These sets of rules are matched to events when they come in. Each rule consists of two expressions that are well-formed formulas of predicate calculus linked by an implication connective ($\Rightarrow$). The left side of each rule contains a prerequisite that must be satisfied, so that the rule is applicable. The right side describes the action to be executed if the rule is applied. There are two types of rule matching, *i.e.*, exact and partial matching. In exact rule matching, the whole left hand side of the rule must be matched before determining which action should be triggered; while in partial matching, the action is determined if some, but not all, of these conditions are fulfilled.

Rule-based methods are appropriate for systems whose configuration is rarely altered. Moreover, they are simpler, modularized and easy to maintain when applied to small systems. However, they have some weaknesses. First, the high cost of implementation and adaptation to changes make it difficult to apply these strategies to large systems (with a potentially large amount of alerts). Second, they are inefficient in dealing with inaccurate information or unseen problems.

Cronk *et al.* [109] divided a rule-based system into three levels: *(i)* control level, as an inference engine that determines how the rules are applied from the knowledge base to solve a given problem; *(ii)* knowledge level, that is a database repository for all the knowledge about the system in the form of declarative knowledge; and *(iii)* a data level, that is a global database that contains the data about problems being dealt with.

Lor [110] proposed a new method to organize system rules by distinguishing between core and customized knowledge. According to his judge, the customized knowledge allows to accurately isolate a fault from the selected group of system entities. The correlation rules are organized as composite event definitions, as another work suggested in [111]. In this approach, unlike others, the distinction is made between primitive events, *i.e.*, alarms and composite events.

Pre-defined scenarios: Like the rule-based methods, pre-defined scenarios is a methodology that acquires manual knowledge. Here, a specific language is used to implement well-defined scenarios. A huge number of correlation languages have been proposed, especially in the network and system security field, related to the specification of attack scenarios. To build these attack sequences, a straightforward way is to first predefine some attack scenario templates. This approach starts with the hypothesis that alerts belonging to one fault have similar attribute values (*e.g.*, source IP address). If various alerts contribute to the construction of a predefined scenario, they should be correlated.

The advantage of this method is that the correlation result is easy to understand and can help security experts to discover all scenarios variants. However, sometimes it is not easy to exhaustively list all attack sequence templates and consequently they fail to be generic. Another limitation of these methods is that novel attack patterns or obfuscation methods created by attackers may cause that the corresponding attack scenarios are not recognized.

Cuppens and Ortalo [112] presented an attack description language called LAMBDA, used to describe with logical expressions the effects and conditions of an attack starting from the variable state of a victim system. In LAMBDA, an attack is specified using five fields: *(i)* attack pre-condition: a logical condition that specifies the conditions to be fulfilled for the success of the attack; *(ii)* attack post-condition: a logical condition that specifies the effects of the attack when it succeeds; *(iii)* attack scenario: the combination of events that the intruder performs when executing an attack; *(iv)* detection scenario: the combination of events that are necessary to detect an occurrence of the attack; and (v) verification scenario: a combination of events to be launched to check if the attack has succeeded. In LAMBDA, several attack specifications can be merged automatically by comparing pre/post conditions. This enables tracing the progress of an attack to a system. Using an attack history, the system can estimate what actions are to be performed by the attacker.

Unlike in [112], where five fields are used to specify an attack, in the State Transition Analysis Technique (STAT) proposed by Kemmer and Vigna [113], an attack has an initial state and at least one ending state. States are characterized by means of assertions, which are predicates on some system security aspects. Attacks modeled using STAT techniques are represented using the STATL language, which was proposed by the same authors in [114]. STATL is an extensible state/transition based attack description language designed to support intrusion detection scenarios. This language allows describing computer penetrations as sequences of actions that an attacker performs to compromise a computer system. The high-level alert patterns and alerts correlation rules are organized as expert knowledge.

Liu *et al.* [115] proposed an alerts correlation model based on the use of a finite automata for the specification of the scenarios. In this model, they generated three kinds of high-level view of attacks: process-critical scenarios, attacker-critical scenarios and victim-critical scenarios. In process-critical scenarios, a non-deterministic finite automata is used to model the intrusion process that takes place between an attacker and a victim. In attacker-critical scenario, the scenario rebuilds the intrusion process that an attacker implemented towards the whole target network. Finally, victim-critical scenarios rebuild the intrusion actions implemented towards a specific system. According to their judge, the model can generate scenarios that are much more directly-perceived.

Cheung *et al.* [116] proposed a correlated attack modeling language, called CAML. It aims at modeling multistep attack scenarios by representing them as trees. Each scenario is subsequently divided into sub-goals or modules. A module specification consists of three sections, namely, activity, pre-condition, and post-condition. To support event-driven inferences, the activity section is used to specify a list of events needed to trigger the module. After that, it identifies logical steps (sub-goals) in attack scenarios and specifies some relationships among these steps: temporal, attribute values, and prerequisites. Each module is linked to others by using pre/post conditions to recognize attack scenarios.

Inferred knowledge: Expert knowledge-based systems can be built by using inference methods with machine learning algorithms. Here, explicit symbolic classification rules are automatically constructed from some training cases. The classification rule learning task can be defined as follows: Given a set of training examples (alerts or meta alerts for which the classification is known), find a set of classification rules that can be used for prediction or classification of new instances, *i.e.*, new incoming alerts or meta alerts.

The main advantage of these methods is that there are no assumptions about the model that will be used in the correlation process, as it is really learned from training instances. Yet, it is really difficult to find representative datasets for the training. Furthermore, a big issue is to make the models capable of generalizing the results for events not observed in the training dataset. Finally, the main drawback of these methods is the computational load implicit in the process, which usually makes them unsuitable for real time systems.

Some contributions have been done in this line, like the work of Smith *et al.* [118]. In this work, they suggested an alerts correlation system based on unsupervised machine learning algorithms. It is implemented in two stages. First, alerts are grouped together so that each group forms one step of an attack. Second, the groups created at the first stage are combined in such a way that each combination of groups contain alerts for a complete attack process.

Some other researchers have used data mining techniques to automate the process of finding meaningful activities and interesting features from training datasets and build the knowledge base [106, 120–123, 134–136]. Data mining is a set of techniques and tools used for the non-trivial process of extracting and representing implicit knowledge. Specifically, Katipally *et al.* [119] used data mining techniques to generate association rules and build predefined attack scenarios that are used for predicting multistage attacks.

Sadoddin and Ghorabani [120] proposed a framework for real time alerts correlation that incorporates two techniques: one for aggregating alerts into structured patterns, and other for incremental mining of frequent structured patterns. In the proposed framework they used time-sensitive statistical analysis to find the relationships between alerts. These were maintained in an efficient data structure and updated incrementally to reflect the latest trends of patterns.

2.5.4 Type of Architecture

Alerts correlation techniques can also be classified based on the type of architecture they use. Different architectures have been proposed in the literature to enable the effective aggregation and correlation of alerts. We classify these architectures as centralized, distributed and hierarchical.

Centralized

In centralized alerts correlation approaches such as the work done by Jinqiao *et al.* [45], the data collection is done locally by the different network agents and then reported as alerts to a central management server where the correlation analysis is done. Correlation algorithms in this architecture are simpler, easier to implement and can correlate overall alerts quickly. Yet, their scalability is limited and their main drawback is that there exists a single point of failure.

Distributed

During the past few years, many researchers have concluded that the alerts correlation process should be carried out in a distributed fashion. According to their claims, this need for a distributed alerts correlation architecture, or completely distributed architecture as mentioned in the literature, emerges from the fact that current and perspective communication networks are expected to increase their size, complexity, speed and the level of heterogeneity. For this reason, using centralized correlation architectures for processing large volumes of correlation information would be computationally prohibitive and unfeasible. Therefore, distributed alerts correlation techniques that would allow the management of correlation agents to reach the solution collectively are necessary.

In these systems, alerts or high level meta alerts are exchanged, aggregated and correlated in a completely cooperative and distributed fashion. All agents are equally weighted, and there are no hierarchical ranks among them. Besides data collection, a partial correlation is done locally by every agent. All agents keep communicating to each other's using some form of distributed protocols, *e.g.*, *Peer-to-Peer* (P2P) protocols or others. Information from that partial correlation carried out at certain agents could be used by others for optimizing their own correlation. To do that, typically a central correlation unit is randomly selected amongst all agents. Each agent has the chance to be selected as a central unit. When the central unit collapses, another alerts correlation unit can substitute it.

This architecture prominently enhances the scalability and the fault tolerance, when compared with a centralized architecture, since the correlation process is distributed amongst several correlation agents. However, there are some issues that need to be solved. First, it consumes more bandwidth due to the information sharing. Second, there is no coherent and consolidated view of the whole system, because the computations are distributed among several entities. Third, load balancing is considered an important issue in this type of architecture, because some management correlation agents may be overloaded in comparison with others. Finally, the requirements and complexity of hardware and implementation are higher.

Mohamed and Basir [124] proposed a distributed alarms correlation and faults identification approach. They divided the network topology into disjoint management domains, and each management domain is assigned to a dedicated intelligent agent, responsible for monitoring and collecting alarms within its management domain. All agents use majority vote rule to determine the root cause of network malfunctioning. Khatoun *et al.* [125] proposed a decentralized alerts correlation technique to detect distributed DoS attacks. It is based on a P2P architecture that correlates alerts produced by various IDSs and provides a high-level view of attempted intrusions. Each IDS supervises its subnetwork and is also a peer inserted in a P2P system that conforms the global collaborative IDS. A distributed hash table is used to efficiently route resource information about the potential victims, and to share data about possible attacks among peers.

Hierarchical

Hierarchical architectures are also called hierarchical distributed architectures because they embed a form of distributed architecture in its design. Here, the correlation process is performed in a hierarchical and cumulative way [127]. Unlike in a distributed architecture, the management agents in the hierarchical model are distributed across different levels. The management agents are located and partitioned into multiple groups, according to different features such as geography, administrative control, and others. Within each group there is a horizontal communication among peer agents, and vertical communication among agents in different levels. The output of their correlation results are passed upward where higher level dedicated correlation units are found. They correlate alerts from both their own level and their children nodes. Then, the correlated alerts are passed upward to a higher level for sharing and further analysis. This process continues until reaching the root, where a central correlation unit collects all the correlation views that were done in lower levels to build a global correlation picture.

Hierarchical architectures are somehow a form of distributed architectures, so they share the same advantages with the latter, regarding scalability and fault tolerance. Yet, they outperform the distributed architectures in terms of coordination and communication costs, especially when very large systems are being managed. Furthermore, their deployment is simpler. On the other hand, in these architectures, the correlation units of the higher levels in the hierarchy still limit the scalability of the correlation system, and their failure can stop the functioning of their whole sub-tree.

Tian *et al.* [126] proposed a hierarchical alerts correlation algorithm for intrusion alerts. It consists of three stages. First, IDS sensors data are aggregated. Second, some local correlation units correlate alerts and build the local correlation graph. Third, the centralized alerts correlation unit constructs the global correlation graph via the local correlation results.

2.6 Comparative Study of Alerts Correlation Techniques

As shown before, researchers and vendors have proposed and used many different design paradigms for implementing alerts correlation techniques. To the best of our knowledge, there are still no comprehensive comparative studies of alerts correlation techniques except some efforts done by few researchers. However, these works typically only consider one application area, *e.g.*, the security field, and cover only a subset of the literature related to their work. According to our opinion, there are some reasons for that. First, as a consequence of using a wide diversity of methods, a comparative analysis of alerts correlation techniques becomes a difficult, tedious and sometimes an error prone task. This is because these correlation methods have their own philosophy for dealing with the alerts correlation problem. They have their own capabilities, strengths and weaknesses. As a consequence, some correlation techniques perform well in some situations and others outperform in other situations. Second, researchers use different methods for validating their ideas. There is no standard performance strategy or benchmarks for evaluating these techniques. Third, the evaluations are applied on different datasets; mostly built ad-hoc for the considered method. At the time of writing this document, we did not find a publicly available dataset explicitly designed for testing alerts correlation algorithms. Finally, many different authors use the same terminology to refer to different alerts correlation operations. Therefore, some authors talk about alarms correlation when referring to the clustering and fusion process, while others call correlation to the process of creating new scenarios. This issue makes the task of building a comparative study a complex and possibly infeasible task. For this reason, the few comparative study proposals that we have found have either limited the study to a number of papers or are biased toward a specific application. In the following, we describe these efforts.

Yusof *et al.* [2] suggested six capability criteria for evaluating alerts correlation techniques. Their work focused on the security field and, specifically, on IDS. The proposed capabilities are: alerts reduction, alerts clustering, identification of multistep attacks, reduction of false alerts, detection of known attacks and detection of unknown attacks. They first classified alerts correlation techniques into four groups: *(i)* Similarity-based, *(ii)* Pre-defined attack scenarios, *(iii)* Pre-requisites and consequences of individual attacks, and *(iv)* Statistical causality. Then, they made a relationship with the capability measures.

Their conclusions were the following. First, similarity-based alerts correlation techniques have the ability to perform alerts reduction, alerts clustering, and detection of known attacks; whereas they fail to reduce false positives and detect multistep and unknown attacks. Second, pre-defined attack scenarios correlation techniques have the same results as similarity-based techniques. Third, pre-requisites

Technique-name	**Alerts reduction**	**Alerts clustering**	**Multi-step attacks**	**Reduction of false positives**	**Detection of known attacks**	**Detection of unknown attacks**
Similarity-based	YES	YES	NO	NO	YES	NO
Pre-defined attack scenarios	YES	YES	NO	NO	YES	NO
Pre-requiste and consequences of individual attacks	YES	YES	NO	NO	YES	NO
Statistical causuality	YES	YES	NO	NO	YES	YES

Table 2.2: Alerts correlation technique versus proposed capability criteria (capable = YES, incapable = NO), taken from Yusof *et al.* [2].

and consequences of individual attacks correlation techniques have the ability to do alerts reduction, alerts clustering, false positives reduction, and detection of multi-step and known attacks. Finally, statistical causality techniques are the only category that has the ability to detect known and unknown attack scenarios, besides alerts reduction and clustering. However, they fail to reduce false positives and detect multi-step attacks. In summary, the overall conclusion that they extracted from this analysis is that further improvements should be done on the process of detecting known, unknown and multi-step attacks, as these capability criteria shall overcome a large number of false alerts problem. Table 2.2 summarizes their analysis.

Siraj *et al.* [3] suggested a comparative study covering only the most representative work in the alerts correlation area related to the security field. Since the correlation process is a multi-step complex task that consists of many operations, they decided to choose five of them to make the comparison. Their selected operations were: normalization, verification, aggregation, correlation, and attack scenario analysis. Their conclusions were the following: First, statistical and probability based techniques suggested in the discussed papers cover all the operations and, for this reason, they are considered as the top category of alerts correlation techniques. Second, statistical and some of the rule-based techniques cover four operations and are thus considered as the next top most category. Others such as case-based, probabilistic, and some rule based techniques only cover three operations and are then considered as the third top most category. The category that lies at the end of the ranking is the model-based techniques. Table 2.3 summarizes their conclusions in more detail.

As shown in Table 2.4, Mirheidari *et al.* [4] provided a comparison among three main categories of algorithms, similarity-based, knowledge-based and statistical-based. They concluded that each category has its own advantages and disadvantages. Thus, any good solution for the alerts correlation problem should use a hybrid approach gathering more than one category of algorithms.

Sadoddin *et al.* [36] classified the alerts correlation techniques in IDSs into three categories. They were knowledge-base methods (that includes rule-based and scenario-based techniques), statistical-based and temporal-based techniques.

Techniques	**Operations**				
	Normalization	**Verification**	**Aggregation**	**Correlation**	**Attack scenario analysis**
Rule-based (case 1)	YES	NO	YES	YES	NO
Rule-based (case 2)	NO	YES	YES	YES	YES
Rule-based (case 3)	YES	YES	YES	YES	NO
Rule-based (case 4)	NO	YES	YES	YES	NO
Rule-based (case 5)	NO	NO	YES	YES	NO
Model-based	NO	NO	YES	YES	NO
Statistical-based	YES	YES	YES	YES	NO
Probabilistic-based	YES	NO	NO	YES	YES
Probabilistic-based & Case-based	NO	NO	YES	YES	YES
Statistical and Probability-based	YES	YES	YES	YES	YES

Table 2.3: Comparative analysis of existing alerts correlation techniques (capable = YES, incapable = NO), taken from Siraj *et al.* [3].

Charactaristics \Techniques	**Similarity-based**	**Knowledge-based**	**Statistical-based**
Combining alerts from various sensors	YES	YES	NO
Requiring prior knowledge	YES	YES	NO
Detecting false alerts	YES	YES	Guessing
Detecting multi-stage attacks	Hardly	YES	Guessing
Find new attacks	YES	NO	YES
Error rate	Average	Low	High

Table 2.4: Comparison of existing alerts correlation algorithms (capable = YES, incapable = NO), taken from Mirheidari *et al.* [4].

They concluded that the last two approaches are capable of correlating alerts related to unknown attacks, while scenario-based and rule-based correlation methods are capable to detect known attack scenarios, as they are solely dependent on predefined attack scenarios and rules in the knowledge base of the system, respectively. Knowledge-based correlation methods have higher accuracy than statistical and temporal techniques, which are very time consuming. In addition, statistical and temporal correlation methods fail to discover causality relationships between noisy alerts or when there are deliberate delays planned by an attacker among the low-level alerts.

Abouabdalla *et al.* [137] summarized several research efforts that were dealing with the false positives reduction task within the IDS area. They concluded that not all researchers deal with the false positives reduction issue in the same way. Some of them work on the IDS level and try to improve the detection efficiency and, as a consequence, this lead to a reduction of the false alerts and an increment of the detection accuracy at the sensor level. Others work at a higher level than the IDS and study other important data like the actual behavior of network traffic and firewall and router logs. They concluded that, with all the benefits obtained from the proposed methods, there is still not a perfect method and some weak points still remain. For instance, when similarity-based techniques are used to remove false positives, the analyst does not discover the actual reasons of IDSs having triggered these alerts. Therefore, this will be only one step further to reduce the false positives alerts. On the other hand, scenario-based techniques are also inefficient in reducing false positives, mainly because this process is enforced to be done in real time so that the response will be more effective. At last, they observed that different authors use the same false positives reduction terminology to refer to different concepts. Some authors refer to it as data mining and clustering, while others mentioned false positives reduction as the process of correlation. As a consequence, some form of standardization is needed to clarify false positives reduction terminology.

To conclude this section, it is observed that, up to now, the alerts correlation process is still an active area of research in both NMSs and network and system security fields. Despite the big amount of efforts done in this field, there is no clear agreement between researchers and vendors on how to formulate efficient solutions and what the performance criteria that determine their effectiveness are. This is the reason why it is not possible to find any generic architectures and benchmarks for evaluating them. From the above comparative studies, the following conclusions are derived.

1. **Similarity-based techniques:** Most techniques belonging to this category share these characteristics:

    - They are simple, *i.e.*, most of similarity-based algorithms use simple mathematical functions to calculate distances between two alerts based on their features. Therefore, they have higher performance in terms of processing speed, and give results faster than others.
    - They are generic. All of these algorithms can be applied and implemented in a wide variety of tools and scenarios. As it will be described later on in Section 4.3, we have checked the existence of several tools and systems that utilize and apply them in their designs.
    - These techniques are performing well when applied to some correlation operations such as alerts reduction, clustering, aggregation and pattern matching, mainly because they operate on the attribute level and will provide valuable results.
    - They are scalable and can give good results regardless of the size of the dataset, as they do not rely on a prior knowledge.
    - The detection accuracy is low. When applied to the NMSs field for discovering root causes or to the security field to discover attack scenarios, they do not give a high detection rate. As a consequence, many researchers use them as a first step of the correlation process before applying other methods.
    - They can detect very simple known attacks, but usually they fail to detect false positives, multi-step and unknown attack scenarios.

2. **Sequential-based techniques:** Most techniques belonging to this category share these characteristics:

    - The majority of these techniques use complex correlation algorithms to discover root causes or attack scenarios.
    - They are scalable and can operate with unseen problems.
    - The detection accuracy is high; they can detect known problems accurately and precisely.
    - These techniques perform better in network and system security than in NMSs, except codebook based ones, as they are mainly designed to discover root causes in NMSs.
    - Finally, they can detect false positives and unknown attacks.

3. **Case-based techniques:** Most techniques belonging to this category share these characteristics:

    - They perform well in static environments, where the system behaviors and topological information remain unchanged, as they build the correlation based on previous cases and predefined scenarios.
    - They have higher accuracy for detecting well-known problems and have the ability to specify a complete action plan.
    - The scalability is a big issue, because it is inversely proportional to the accuracy. They do not provide answers for new problems. Therefore, they have higher complexity in implementation.
    - They do not perform well when applied to real time systems, since in these types of systems the response should be very fast. These techniques need considerable time to retrieve the most similar case, especially when the database is large.
    - Last but not least, they fail to reduce false positives and detect multi-step and unknown attacks.

Chapter 3

Problem Statement: A Case Study

The main objective of this chapter is to carry out a preliminary exploration of some tickets and alerts datasets in order to provide insights about their structures and specificities such as the types of records they contain, the relationships among tickets, among alerts, and between both tickets and alerts. We are also interested in revealing the potential difficulties that should be circumvented to address the objective of this work. Thus, we start describing the ITS system in more detail, analyzing the structure and the information in tickets and alerts and obtaining some basic statistics for them. Next, an analysis of the potential relationships between tickets, alerts and both together is made. In this analysis, temporary and similarity based links are explored. Finally, we describe the main difficulties that should be faced and solved for finding a successful solution to the correlation of tickets and alerts.

The case study considered here is the network, event data and procedures handled by the IT management company we are collaborating with. It should be pointed out that it is difficult to access to this kind of information, as there are no ITS data from companies publicly available. On the one hand, vendors do not publish these data due to competitive issues. On the other hand, network management teams have also the same lack of motivation for publishing them, because this information is considered confidential and its publications might lead to security issues. We have only found a few research and educational oriented networks that make their ITS data publicly accessible on the Internet [138, 139], although they do not meet all the required properties for the objective of this thesis work.

In our case study, the handling of alerts and tickets is made by two different departments: management staffs, in charge of the effective supervision of the network and, subsequently, the alerts; and service desk staffs that attend the requests and complaints from customers. Both departments have the capability to create tickets, although management staffs do it mainly as a response to alerts generated by some network failure or malfunctioning, while the service desk staffs create tickets from the end user complaints. This is an important fact, from the point of view of the work carried out in this thesis, as it is expected that the tickets created by management staffs contain some information related to the alerts triggering them, while the tickets created by service desk staffs can provide richer information regarding the ongoing incidents, but no information regarding technical details or alerts.

The remainder of this chapter is organized as follows. Section 3.1 provides a detailed description of both the alerts and the tickets datasets considered in this work. Section 3.2 illustrates the difficulties and the challenges that appear during the process of relating tickets, alerts and both of them. Finally, Section 3.3 presents preliminary results of some implemented correlation scenarios.

3.1 Datasets

The experimental data is composed of two different datasets of tickets and alerts, namely *Dataset 1* (DS1) and *Dataset 2* (DS2), that belong to the ITS of a same corporate network, but are generated by two different staff members belonging to two different outsourcing companies. The company that outsources the management of the ITS system is in charge of the management of a regional network formed by a large number of governmental sectors such as academic institutions, health centers and service centers, among others. For privacy reasons, and considering that we are working with sensitive issues regarding management efficiency, we keep some administrative information regarding the management companies hidden, such as the companies names and the regional network in charge of these companies (the hidden information are covered by black boxes in all snapshots that are listed below).

3.1.1 Tickets Datasets Description

DS1 is composed of both tickets and alerts for the whole year of 2011, with a total set of 19,162 raw tickets. It is necessary to mention here that we have obtained DS1 from the first outsourcing company split into twelve batches, which are separated in time, each one for every month. On the other hand, tickets in DS2 are obtained from the second outsourcing company as one large batch file of six months, from October 2013 to the end of March 2014, with a total set of 9,612 raw tickets.

Table 3.1 gives an overview of some useful statistics of tickets found in both datasets. As previously mentioned and emphasized in the table, there are two main

		tickets of DS1	tickets of DS2
Total # of tickets		19,162	9,612
Mean # of tickets per month		1,596.8	1,602
Mean # of affected elements per ticket		1.4	1.42
Mean # of ticket identities per ticket		1.2	1.39
Percentage of created tickets / management group	**MS**	45%	54%
	SD1	44%	39%
	SD2	11%	7%
Mean ticket lifetime [h:m:s]		22:48:02	33:41:37
Minimum ticket lifetime [h:m:s]		00:00:24	00:00:23
Maximum ticket lifetime [h:m:s]		1244:09:42	1397:13:27
Ticket intensity [tickets per hour]		2.22	2.19

Table 3.1: Some statistics of the tickets in both companies' datasets, DS1 and DS2.

management groups that have the responsibility to manually create tickets: MS is the management staff that creates tickets in response to receiving network alerts, while the staff belonging to the SD group creates tickets in response to receiving customer calls. In the case study we are considering, the SD group is further split into two subgroups, SD1 and SD2. SD1 is mainly referred to as call center level 1, whereas SD2 is referred to as call center level 2. Besides the above table, Figure 3.1 shows the number of tickets created in each month for DS1. We observe that both datasets are almost similar. The mean number of tickets per month and the mean number of tickets per hour, *i.e.*, ticket intensity are 1,596.8 and 2.22 for DS1, and 1,602 and 2.19 for DS2, respectively. Furthermore, the mean number of affected elements and ticket identities are also quite similar in both datasets. As will be explained later on in this chapter, an affected element is defined as the number of network systems or devices affected in the problem reported in the ticket, and ticket identity refers to an unique identity of the ticket in the database. Finally, we observe that the number of tickets generated by MS is slightly higher in DS2.

Some snapshots of tickets, as shown in the ITS platform, are illustrated in Figures 3.2 to 3.5, showing the different types of tickets' fields with some values.

The tickets in both datasets share the same structure and format. There are 293 different attributes that describe every ticket with three different data formats: strings, numbers and booleans. We have found that a large number of the attributes are not important for the work carried out in this thesis, or contain empty data, so we have decided to filter them out. In order to do that, with the help of the management staff, we classified these attributes into 5 categories based on the information that they contain. These categories and the most relevant attributes are explained below giving more detailed information.

- **Management groups information:** This category mainly contains information about the management groups that are involved in the ticket management pro-

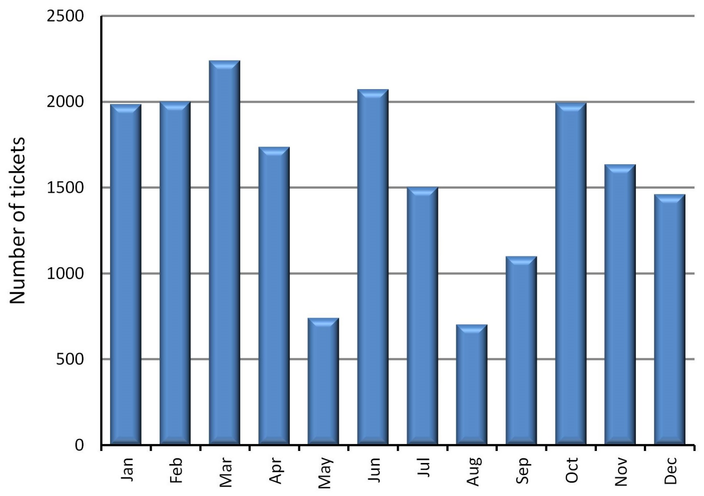
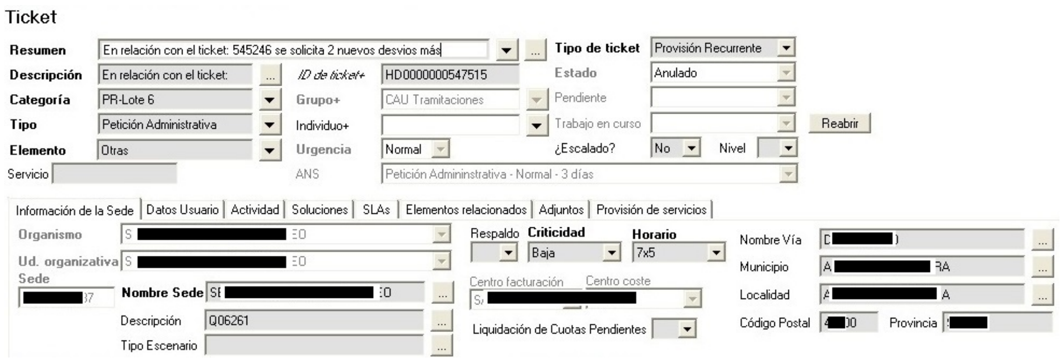


Figure 3.1: Total number of tickets in DS1, for each month.



Ticket

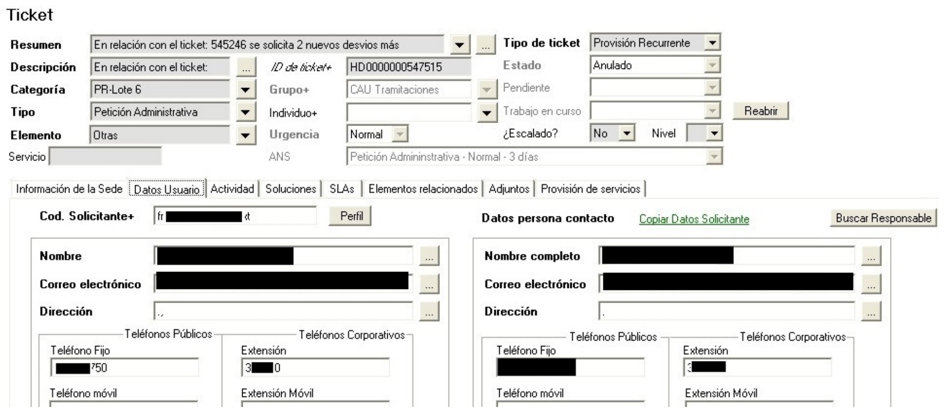
Resumen	En relación con el ticket: 545246 se solicita 2 nuevos desvíos más			Tipo de ticket	Provisión Recurrente
Descripción	En relación con el ticket:	*ID de ticket+*	HD0000000547515	Estado	Anulado
Categoría	PR-Lote 6	Grupo+	CAU Tramitaciones	Pendiente	
Tipo	Petición Administrativa	Individuo+		Trabajo en curso	
Elemento	Otras	Urgencia	Normal	¿Escalado?	No
Servicio		ANS	Petición Admininstrativa - Normal - 3 días	Nivel	

Reabrir

Información de la Sede | Datos Usuario | Actividad | Soluciones | SLAs | Elementos relacionados | Adjuntos | Provisión de servicios

Organismo	S [redacted] EO	Nombre Vía	C [redacted])
Ud. organizativa	S [redacted] EO	Municipio	A [redacted] RA
Sede	[redacted] 37	Localidad	A [redacted] A
Nombre Sede	SE [redacted] EO	Código Postal	4 [redacted] 00
Descripción	Q06261	Provincia	[redacted]
Tipo Escenario			
Respaldo			
Criticidad	Baja		
Horario	7x5		
Centro facturación / Centro coste	S/ [redacted]		
Liquidación de Cuotas Pendientes			

Figure 3.2: A snapshot taken from the ITS system that shows the information of an incident and its affected elements as reported in a ticket.



Ticket

Resumen	En relación con el ticket: 545246 se solicita 2 nuevos desvíos más			Tipo de ticket	Provisión Recurrente
Descripción	En relación con el ticket:	*ID de ticket+*	HD0000000547515	Estado	Anulado
Categoría	PR-Lote 6	Grupo+	CAU Tramitaciones	Pendiente	
Tipo	Petición Administrativa	Individuo+		Trabajo en curso	
Elemento	Otras	Urgencia	Normal	¿Escalado?	No
Servicio		ANS	Petición Admininstrativa - Normal - 3 días	Nivel	

Reabrir

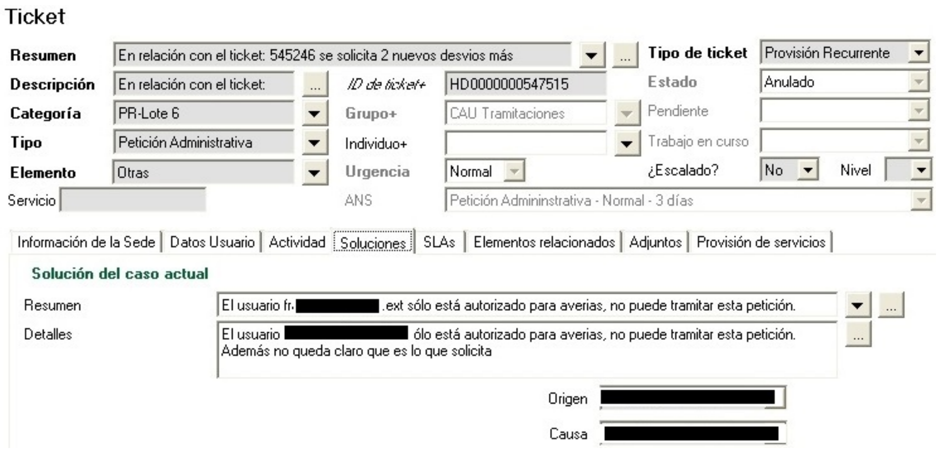
Información de la Sede | Datos Usuario | Actividad | Soluciones | SLAs | Elementos relacionados | Adjuntos | Provisión de servicios

Cod. Solicitante+ fr [redacted] xt   Perfil   Datos persona contacto   Copiar Datos Solicitante   Buscar Responsable

Nombre	[redacted]	Nombre completo	[redacted]
Correo electrónico	[redacted]	Correo electrónico	[redacted]
Dirección	.,	Dirección	.

Teléfonos Públicos	Teléfonos Corporativos	Teléfonos Públicos	Teléfonos Corporativos
Teléfono Fijo	Extensión	Teléfono Fijo	Extensión
[redacted] 750	3 [redacted] 0	[redacted]	3 [redacted]
Teléfono móvil	Extensión Móvil	Teléfono móvil	Extensión Móvil

Figure 3.3: A snapshot taken from the ITS system that shows some administrative information about the affected users as reported in a ticket.



Ticket

Resumen	En relación con el ticket: 545246 se solicita 2 nuevos desvios más			**Tipo de ticket**	Provisión Recurrente
Descripción	En relación con el ticket:	*ID de ticket+*	HD0000000547515	**Estado**	Anulado
Categoría	PR-Lote 6	**Grupo+**	CAU Tramitaciones	Pendiente	
Tipo	Petición Administrativa	Individuo+		Trabajo en curso	
Elemento	Otras	**Urgencia**	Normal	¿Escalado?	No
Servicio		ANS	Petición Admininstrativa - Normal - 3 días	Nivel	

Información de la Sede | Datos Usuario | Actividad | Soluciones | SLAs | Elementos relacionados | Adjuntos | Provisión de servicios

Solución del caso actual

Resumen	El usuario fr[redacted] .ext sólo está autorizado para averias, no puede tramitar esta petición.
Detalles	El usuario [redacted] ólo está autorizado para averias, no puede tramitar esta petición. Además no queda claro que es lo que solicita
Origen	[redacted]
Causa	[redacted]

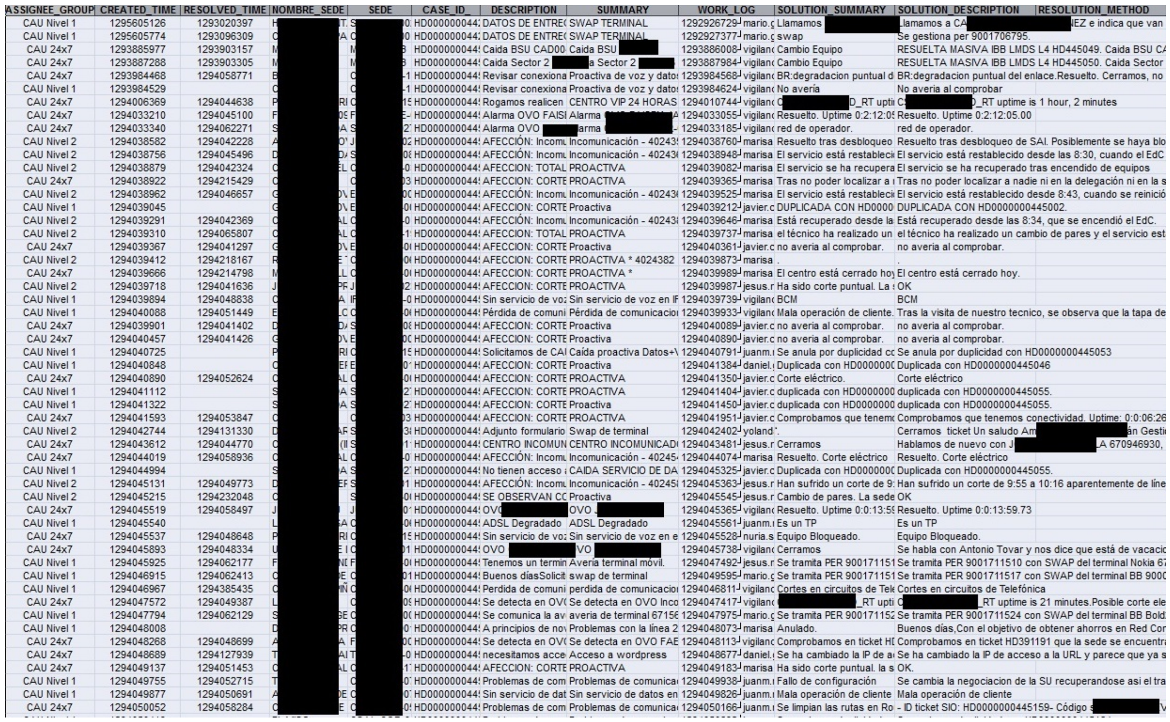
Figure 3.4: A snapshot taken from the ITS system that shows the solution description and the workaround carried out to solve an incident as reported by a ticket.

cess, either ticket creation, ticket resolution or ticket acknowledgment. The most relevant attributes of this category are:

- ASSIGNEE_GROUP: It refers to the identity of the member/group who created the ticket.
- GR_DINAMICO_SEDES: This attribute represents a list of names of staff members who are located in a specific geographical area.
- ASSIGNEED_TO_GROUP: It is a string that represents the name of the management group that is involved in solving an incident reported by a ticket.
- ZASSIGNEDGROUPID: It refers to the group identity of the staff members who are involved in solving an incident reported by a ticket. This attributes takes an integer number.

• **Timestamps:** There exist several attributes related to the lifetime of the ticket (from creation to validation). This category of attributes will help to track the contents of the tickets and also to make some statistical analysis on them. All of the timestamps attributes are in epoch time format, defined as the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time. The following list of attributes are the most important ones.

- TICKET CREATION TIME: It refers to the instant at which the ticket is created by a ticket creator.
- TICEKT RESOLUTION TIME: It is the instant at which the ticket is marked as resolved by a ticket resolver.
- TICKET VALIDATION TIME: This timestamp refers to the instant at which the ticket is marked as validated by a ticket validator.



ASSIGNEE_GROUP	CREATED_TIME	RESOLVED_TIME	NOMBRE_SEDE	SEDE	CASE_ID_	DESCRIPTION	SUMMARY	WORK_LOG	SOLUTION_SUMMARY	SOLUTION_DESCRIPTION	RESOLUTION_METHOD
CAU Nivel 1	1295605126	1293020397	H[redacted]T.	S[redacted]0	HD000000044	DATOS DE ENTRE	SWAP TERMINAL	1292926729┘mario.g	Llamamos [redacted]	Llamamos a CA[redacted]NEZ e indica que van	
CAU Nivel 1	1295605774	1293096309	C[redacted]A	C[redacted]0	HD000000044	DATOS DE ENTRE	SWAP TERMINAL	1292927377┘mario.g	swap	Se gestiona per 9001706795.	
CAU 24x7	1293885977	1293903157	M[redacted]	M[redacted]3	HD000000044	Caida BSU CAD00	Caida BSU [redacted]	1293886008┘vigilan	Cambio Equipo	RESUELTA MASIVA IBB LMDS L4 HD445049. Caida BSU CA	
CAU 24x7	1293887288	1293903305	M[redacted]	M[redacted]3	HD000000044	Caida Sector 2 [redacted]	a Sector 2 [redacted]	1293887984┘vigilan	Cambio Equipo	RESUELTA MASIVA IBB LMDS L4 HD445050. Caida Sector	
CAU 24x7	1293984468	1294058771	B[redacted]	C[redacted]-1	HD000000044	Revisar conexiona	Proactiva de voz y datos	1293984568┘vigilan	BR:degradacion puntual d	BR:degradacion puntual del enlace.Resuelto. Cerramos, no	
CAU Nivel 1	1293984529		C[redacted]	C[redacted]-1	HD000000044	Revisar conexiona	Proactiva de voz y datos	1293984624┘vigilan	No avería	No averia al comprobar	
CAU 24x7	1294006369	1294044638	P[redacted]RI	C[redacted]15	HD000000044	Rogamos realicen	CENTRO VIP 24 HORAS	1294010744┘vigilan	C[redacted]D_RT upti	CS[redacted]D_RT uptime is 1 hour, 2 minutes	
CAU 24x7	1294033210	1294045100	F[redacted]09	F[redacted]E-	HD000000044	Alarma OVO FAISI	Alarma [redacted]A	1294033055┘vigilan	Resuelto. Uptime 0:2:12:05	Resuelto. Uptime 0:2:12:05.00	
CAU 24x7	1294033340	1294062271	S[redacted]A	S[redacted]27	HD000000044	Alarma OVO [redacted]	[redacted]larma [redacted]-	1294033185┘vigilan	red de operador.	red de operador.	
CAU Nivel 2	1294038582	1294042228	A[redacted]O'	J[redacted]02	HD000000044	AFECCIÓN: Incomu	Incomunicación - 40243	1294038760┘marisa	Resuelto tras desbloqueo	Resuelto tras desbloqueo de SAI. Posiblemente se haya blo	
CAU Nivel 2	1294038756	1294045496	D[redacted]D/	S[redacted]08	HD000000044	AFECCIÓN: Incomu	Incomunicación - 40243	1294038948┘marisa	El servicio está restableci	El servicio está restablecido desde las 8:30, cuando el EdC	
CAU Nivel 2	1294038879	1294042324	C[redacted]EL	C[redacted]-0	HD000000044	AFECCION: TOTAL	PROACTIVA	1294039082┘marisa	El servicio se ha recupera	El servicio se ha recuperado tras encendido de equipos	
CAU 24x7	1294038922	1294215429	C[redacted]	C[redacted]03	HD000000044	AFECCION: CORTE	PROACTIVA	1294039365┘marisa	Tras no poder localizar a	Tras no poder localizar a nadie ni en la delegación ni en la s	
CAU Nivel 2	1294038962	1294046657	G[redacted]OV	E[redacted]00	HD000000044	AFECCIÓN: Incomu	Incomunicación - 40243	1294039525┘marisa	El servicio está restableci	El servicio está restablecido desde 8:43, cuando se reinició	
CAU Nivel 1	1294039045		G[redacted]OV	E[redacted]0	HD000000044	AFECCION: CORTE	Proactiva	1294039212┘javier.c	DUPLICADA CON HD0000	DUPLICADA CON HD0000000445002.	
CAU Nivel 2	1294039291	1294042369	C[redacted]AL	C[redacted]-0	HD000000044	AFECCIÓN: Incomu	Incomunicación - 40243	1294039646┘marisa	Está recuperado desde la	Está recuperado desde las 8:34, que se encendió el EdC.	
CAU Nivel 2	1294039310	1294065807	C[redacted]AL	C[redacted]-1	HD000000044	AFECCION: TOTAL	PROACTIVA	1294039737┘marisa	el técnico ha realizado un	el técnico ha realizado un cambio de pares y el servicio est	
CAU 24x7	1294039367	1294041297	G[redacted]OV	E[redacted]0	HD000000044	AFECCION: CORTE	Proactiva	1294040361┘javier.c	no averia al comprobar.	no averia al comprobar.	
CAU Nivel 2	1294039412	1294218167	R[redacted]E	C[redacted]0	HD000000044	AFECCION: CORTE	PROACTIVA * 4024382	1294039873┘marisa	.	.	
CAU 24x7	1294039666	1294214798	M[redacted]LL	C[redacted]0	HD000000044	AFECCION: CORTE	PROACTIVA *	1294039989┘marisa	El centro está cerrado hoy	El centro está cerrado hoy.	
CAU Nivel 2	1294039718	1294041636	J[redacted]PR	J[redacted]2	HD000000044	AFECCION: CORTE	PROACTIVA	1294039987┘jesus.r	Ha sido corte puntual. La	OK	
CAU Nivel 1	1294039894	1294048838	C[redacted]A	IF[redacted]-0	HD000000044	Sin servicio de voz	Sin servicio de voz en IF	1294039739┘vigilan	BCM	BCM	
CAU Nivel 1	1294040088	1294051449	E[redacted]LC	C[redacted]0	HD000000044	Pérdida de comuni	Pérdida de comunicacion	1294039933┘vigilan	Mala operación de cliente.	Tras la visita de nuestro tecnico, se observa que la tapa de	
CAU 24x7	1294039901	1294041402	D[redacted]D/	S[redacted]08	HD000000044	AFECCION: CORTE	Proactiva	1294040089┘javier.c	no averia al comprobar.	no averia al comprobar.	
CAU 24x7	1294040457	1294041426	G[redacted]OV	E[redacted]00	HD000000044	AFECCION: CORTE	Proactiva	1294040890┘javier.c	no averia al comprobar.	no averia al comprobar.	
CAU Nivel 1	1294040725		P[redacted]RI	C[redacted]15	HD000000044	Solicitamos de CA	Caída proactiva Datos+V	1294040791┘juanm.	Se anula por duplicidad co	Se anula por duplicidad con HD0000000445053	
CAU Nivel 1	1294040848		C[redacted]EF	E[redacted]0	HD000000044	AFECCION: CORTE	Proactiva	1294041384┘daniel.	Duplicada con HD0000000	Duplicada con HD0000000445046	
CAU 24x7	1294040890	1294052624	C[redacted]AL	C[redacted]0	HD000000044	AFECCION: CORTE	PROACTIVA	1294041350┘javier.c	Corte eléctrico.	Corte eléctrico	
CAU Nivel 1	1294041112		S[redacted]A	S[redacted]27	HD000000044	AFECCION: CORTE	PROACTIVA	1294041404┘javier.c	duplicada con HD0000000	duplicada con HD0000000445055.	
CAU Nivel 1	1294041322		S[redacted]A	S[redacted]27	HD000000044	AFECCION: CORTE	Proactiva	1294041450┘javier.c	duplicada con HD0000000	duplicada con HD0000000445055.	
CAU 24x7	1294041593	1294053847	C[redacted]	C[redacted]03	HD000000044	AFECCION: CORTE	PROACTIVA	1294041951┘javier.c	Comprobamos que tenem	Comprobamos que tenemos conectividad. Uptime: 0:0:06:26	
CAU Nivel 2	1294042744	1294131330	D[redacted]AR	S[redacted]3	HD000000044	Adjunto formulario	Swap de terminal	1294042402┘yoland	'.	Cerramos ticket Un saludo Am[redacted]án Gesti	
CAU 24x7	1294043612	1294044770	C[redacted](II	S[redacted]1	HD000000044	CENTRO INCOMUN	CENTRO INCOMUNICAD	1294043481┘jesus.r	Cerramos	Hablamos de nuevo con J[redacted]A 670946930,	
CAU 24x7	1294044019	1294058936	C[redacted]AL	C[redacted]-0	HD000000044	AFECCIÓN: Incomu	Incomunicación - 40245	1294044074┘marisa	Resuelto. Corte eléctrico	Resuelto. Corte eléctrico	
CAU Nivel 1	1294044994		S[redacted]A	S[redacted]27	HD000000044	No tienen acceso	CAIDA SERVICIO DE DA	1294045325┘javier.c	Duplicada con HD0000000	Duplicada con HD0000000445055.	
CAU Nivel 2	1294045131	1294049773	D[redacted]EF	S[redacted]1	HD000000044	AFECCIÓN: Incomu	Incomunicación - 40245	1294045363┘jesus.r	Han sufrido un corte de 9:	Han sufrido un corte de 9:55 a 10:16 aparentemente de líne	
CAU Nivel 2	1294045215	1294232048	C[redacted]	S[redacted]0	HD000000044	SE OBSERVAN CC	Proactiva	1294045545┘jesus.r	Cambio de pares. La sede	OK	
CAU 24x7	1294045519	1294058497	J[redacted]	J[redacted]0	HD000000044	OVO[redacted]	OVO J[redacted]	1294045365┘vigilan	Resuelto. Uptime 0:0:13:59	Resuelto. Uptime 0:0:13:59.73	
CAU Nivel 1	1294045540		L[redacted]SA	C[redacted]0	HD000000044	ADSL Degradado	ADSL Degradado	1294045561┘juanm.	Es un TP	Es un TP	
CAU 24x7	1294045537	1294048648	P[redacted]RI	C[redacted]15	HD000000044	Sin servicio de voz	Sin servicio de voz en e	1294045528┘nuria.s	Equipo Bloqueado.	Equipo Bloqueado.	
CAU 24x7	1294045893	1294048334	U[redacted]E I	C[redacted]01	HD000000044	OVO [redacted]	[redacted]VO [redacted]	1294045738┘vigilan	Cerramos	Se habla con Antonio Tovar y nos dice que está de vacacio	
CAU Nivel 1	1294045925	1294062177	F[redacted]NI	F[redacted]0	HD000000044	Tenemos un termin	Avería terminal móvil.	1294047492┘jesus.r	Se tramita PER 900171151	Se tramita PER 9001711510 con SWAP del terminal Nokia 67	
CAU Nivel 1	1294046915	1294062413	C[redacted]DE	C[redacted]01	HD000000044	Buenos díasSolicit	swap de terminal	1294049595┘mario.g	Se tramita PER 900171151	Se tramita PER 9001711517 con SWAP del terminal BB 9000	
CAU Nivel 1	1294046967	1294385435	C[redacted]IÑ	C[redacted]0	HD000000044	Perdida de comuni	perdida de comunicacion	1294046811┘vigilan	Cortes en circuitos de Tele	Cortes en circuitos de Telefónica	
CAU 24x7	1294047572	1294049387	L[redacted]	C[redacted]09	HD000000044	Se detecta en OV	Se detecta en OVO Inco	1294047417┘vigilan	[redacted]_RT upti	C[redacted]_RT uptime is 21 minutes.Posible corte ele	
CAU Nivel 1	1294047794	1294062129	S[redacted]GE	C[redacted]0	HD000000044	Se comunica la av	averia de terminal 67156	1294047975┘mario.g	Se tramita PER 900171152	Se tramita PER 9001711524 con SWAP del terminal BB Bold	
CAU Nivel 1	1294048008		D[redacted]PR	C[redacted]0	HD000000044	A principios de no	Problemas con la línea 2	1294048073┘marisa	Anulado.	Buenos días,Con el objetivo de obtener ahorros en Red Cor	
CAU 24x7	1294048268	1294048699	A[redacted]A	F[redacted]0	HD000000044	Se detecta en OV	Se detecta en OVO FAE	1294048113┘vigilan	Comprobamos en ticket HI	Comprobamos en ticket HD391191 que la sede se encuentra	
CAU 24x7	1294048689	1294127939	T[redacted]AI	T[redacted]-0	HD000000044	necesitamos acce	Acceso a wordpress	1294048677┘daniel.	Se ha cambiado la IP de a	Se ha cambiado la IP de acceso a la URL y parece que ya s	
CAU 24x7	1294049137	1294051453	C[redacted]AL	C[redacted]-1	HD000000044	AFECCION: CORTE	PROACTIVA	1294049183┘marisa	Ha sido corte puntual. la s	OK.	
CAU Nivel 1	1294049755	1294052715	T[redacted]	C[redacted]0	HD000000044	Problemas de com	Problemas de comunica	1294049938┘juanm.	Fallo de configuración	Se cambia la negociacion de la SU recuperandose asi el tra	
CAU Nivel 1	1294049877	1294050691	A[redacted]DE	C[redacted]0	HD000000044	Sin servicio de dat	Sin servicio de datos en	1294049826┘juanm.	Mala operación de cliente	Mala operación de cliente	
CAU 24x7	1294050052	1294058284	C[redacted]E	C[redacted]0	HD000000044	Problemas de com	Problemas de comunica	1294050166┘juanm.	Se limpian las rutas en Ro	- ID ticket SIO: HD0000000445159- Código s[redacted]V	

Figure 3.5: A snapshot of some samples of tickets taken from DS1.

  - TICKET ASSIGNEED TIME: It represents the instant at which the ticket is re-assigned to the appropriate management group/member by either ticket creator or ticket resolver, assuming that there exist multiple management groups or members involved in resolving the same incident.

- **Topological information:** The following list of attributes contains the most relevant topological data that provide information about the origin of the problem within the managed corporate network.

  - ORGANISMO (organization): It contains the name of the main organization in the managed corporate network that is affected by an incident reported by a ticket.
  - NOMBRE_SEDE (headquarter): It is a string that refers to name of the headquarter that corresponds to the main organization. It is worth to mention here that every organization has multiple headquarters that are separated geographically.
  - CODIGO_POSTAL (postal code): It is a string that refers to the area code of the headquarter.
  - SEDE (affected element/service identity): It refers to the affected element or service inside the headquarter. In the managed corporate network the

topology resembles the organizational structure, thus using a hierarchical approach. For example, the format of the SEDE field is anonymously represented as XXX-YYY-ZZZ, where the first three letters, XXX, refer to the first three letters of the organization name, the second three letters, YYY, refer to the first three letters of the headquarter, and finally, the last three refer to the identity of the affected element or service in the headquarter. Consequently, the most important and relevant field that collects all of the topological information about the location of the incident is the SEDE attribute.

- **Problem description:** With the help of this category of attributes, we can understand the problem reported in the ticket. Some important attributes belonging to this category are:
  - CASE_ID (ticket identity): Every ticket in the database has a unique identifier, called CASE_ID. This attribute is a fixed length string that consists of 15 digits; the first two are fixed letters, HD, and the remaining are integers, *e.g.*, HD0000000442623.
  - DESCRIPTION: This attributes contains a human written text of variable length that describes the problem.
  - SUMMARY: This field may take a value from a predefined list of values that exactly describe the type of the problem, *e.g.*, "Sin servicio de datos en el centro" (no data service at the center).
  - ROOT_CAUSE: It is a string that represents the root cause of the problem.
- **Solution:** A large number of attributes that contain useful information about the solution description for a specific problem. Among these relevant attributes we can find:
  - OBSERVACIONES (Observations): It contains text with a human written format that describes all the observations and comments reported by the ticket resolvers.
  - WORK_LOG: It contains the same information as the OBSERVACIONES attribute, but it may also contain all the possible solutions that have been applied by the different tickets resolvers and their results.
  - SOLUTION_DESCRIPTION: It represents a description of the final solution applied to solve the problem.
  - SOLUTION_SUMMARY: It represents a summary of the applied solution. It may also contain a summary of the information described in the SOLUTION_DESCRIPTION field.

- **Ticket severity:** The following attributes are related to the ticket severity, and they give useful information about the ticket in terms of its severity and its relation with the SLA:

  - CRITICIDAD_SERVICIO (service_severity): It is a boolean that represents whether the reported incident is affecting a critical or non critical service/s.
  - PRIORITY: It is an integer that gives the priority level of a ticket. It may take a number from 0 to 5, being 0 the lowest priority and 5 the highest priority.
  - NUM_SERVICIOSAFEC (number of affected services): It is an integer that gives the number of services or elements affected by the reported incident.
  - CRITICIDAD (severity): It gives information about the severity of a ticket. It may take one of four different values: *"baja"* (low), *"alta"* (high), *"media"* (medium) or *"crítica"* (critical).

Coming back to Table 3.1, it is worth to recall here that, besides to the main ticket identifier (CASE_ID field), a ticket may contain identifiers of other tickets that are somehow related to it. For this reason, the mean number of ticket identities is greater than one. The same is also valid for the affected elements, that are referred to also as SEDEs in the list of ticket's attributes that are described above. A ticket may contain, besides the main affected element (SEDE field), other SEDEs that share the same root cause. Thus, the number of SEDEs per ticket may also be greater than one.

3.1.2 Alerts Datasets Description

Like the tickets, DS1 also contains alerts for the whole year of 2011, with a total set of 8,198,052 raw alerts. In addition, alerts are obtained from the first outsourcing company split into twelve batches that are separated in time, each one for every month. Alerts for DS2 are obtained from the second outsourcing company as one large batch file of six months, with a total set of 1,703,662 raw alerts.

Figure 3.6 illustrates a snapshot of a sample of alerts log as captured by the `HP OpenView` platform that is currently deployed in the managed corporate network environment, and the information included in each alert is presented in a fixed structured format as illustrated in Figure 3.7.

The alerts collected by the `HP OpenView` in the IT management company trace has the following characteristics.

1. Alerts are generated by two different procedures: (*i*) polling, in which `HP OpenView` *Operations Support Systems* (OSS) software tries to contact an interface/node and actively collects information or (*ii*) trap, in which the agents generate alerts (SNMP traps) that are collected by the OSS.

2. Alerts corresponding to Cisco/Teldat/Generic agent interface equipments are generated by traps. The rest are generated by polling.

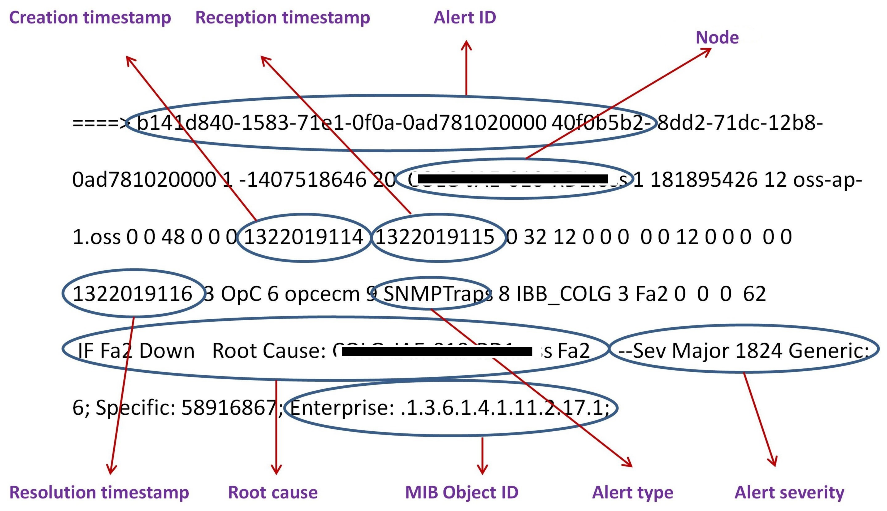
3. The alerts fields are tab separated. The important fields are:

    - Node: This is the affected element (SEDE) that generates the alert.
    - Creation timestamp: It is the instant at which the alert has been generated.
    - Reception timestamp: It is the instant that refers to the reception of the alert at the OSS.
    - Resolution timestamp: It is the instant that refers to the resolution of the alert.
    - Alert description: Here, there are several sub-fields:
        - *Element type:* Type `IF` indicates that an interface has gone to state DOWN or UP. Type `Node` indicates that a node has gone UP or DOWN
        - *Root cause:* Information about the affected element and the interface that have caused the problem.
        - *Sev:* Severity of the alert. Possible values are *major*, *minor*, *critical*, *normal*, *informative*, etc.
        - *Generic:* It represents the generic code of a trap. For all alerts generated by `HP OpenView` (polling) the value is equal to 6.
        - *Specific:* It refers to the specific code of the trap. For all alerts generated by `HP OpenView` (polling) there are two codes: one for `interface up` and other for `interface down`.
        - *Enterprise:* It is the Object ID in the MIB of the affected element in the alert. In the alerts generated by `HP OpenView` it is set to the value .1.3.6.1.4.1.11.2.17.1

In addition, we have obtained the alerts of DS2 from the second outsourcing company after having some preprocessing operations carried out by the management staff. For this reason, we have found that the mean number of alerts per hour, *i.e.*, alert intensity, is 384 for DS2, lower than that for DS1, which is 935.8. Besides, the mean number of affected elements (SEDEs) per alert is very similar in both datasets, 1.2 (DS1) and 1.15 (DS2). Every alert may contain other affected elements besides the main one. This occurs due to the fact that during the management of the corporate network there are intermediate nodes called gateways that collect

```
====> 283634ec-6d4f-71e0-18d5-0ad781020000 97c3c22c-a34d-71dc-066a-0ad781020000
1 -1062675604 20 I[redacted].oss 1 181895426 12 oss-ap-1.oss 0 0 48 0 0 0
1303524754 1303524756 0 32 12 0 0 0  0 0 12 0 0 0  0 0 1303526005 3 OpC 6 opcecm 9
SNMPTraps 10 TESA_DATOS 1 2 0  0  0  70 Teldat Agent Interface Down (linkDown Trap) on
interface 2 --Sev Major 140 Generic: 2; Specific: 0; Enterprise: .1.3.6.1.4.1.2007.1.1.101;
====> 28463f04-6d4f-71e0-18d5-0ad781020000 97c3c22c-a34d-71dc-066a-0ad781020000
1 -1062675604 20 [redacted]1.oss 1 181895426 12 oss-ap-1.oss 0 0 48 0 0 0
1303524754 1303524756 0 32 12 0 0 0  0 0 12 0 0 0  0 0 1303526005 3 OpC 6 opcecm 9
SNMPTraps 10 TESA_DATOS 1 6 0  0  0  70 Teldat Agent Interface Down (linkDown Trap) on
interface 6 --Sev Major 140 Generic: 2; Specific: 0; Enterprise: .1.3.6.1.4.1.2007.1.1.101;
====> 28561f1e-6d4f-71e0-18d5-0ad781020000 97c3c22c-a34d-71dc-066a-0ad781020000
1 -1062675604 20 [redacted]1.oss 1 181895426 12 oss-ap-1.oss 0 0 48 0 0 0
1303524754 1303524756 0 32 12 0 0 0  0 0 12 0 0 0  0 0 1303526005 3 OpC 6 opcecm 9
SNMPTraps 10 TESA_DATOS 1 7 0  0  0  70 Teldat Agent Interface Down (linkDown Trap) on
interface 7 --Sev Major 140 Generic: 2; Specific: 0; Enterprise: .1.3.6.1.4.1.2007.1.1.101;
====>107ed0b8-6d52-71e0-18d5-0ad781020000 7ed568ba-a34d-71dc-066a-
0ad781020000 1 -1062675604 20 I[redacted]l.oss 1 181895426 12 oss-ap-1.oss 0 0
48 0 0 0 1303526004 1303526005 0 2 12 0 0 0  0 0 12 0 0 0  0 0 1303526005 3 OpC 6
opcecm 9 SNMPTraps 10 TESA_DATOS 1 6 0  0  0  67 Teldat Agent Interface Up (linkUp Trap)
on interface 6 --Sev Normal 140 Generic: 3; Specific: 0; Enterprise: .1.3.6.1.4.1.2007.1.1.101;
```

Figure 3.6: A snapshot of some samples of alerts log as captured by `HP OpenView` platform that is currently deployed in the managed corporate network.



```
====> b141d840-1583-71e1-0f0a-0ad781020000 40f0b5b2-8dd2-71dc-12b8-
0ad781020000 1 -1407518646 20 [redacted]s 1 181895426 12 oss-ap-
1.oss 0 0 48 0 0 0 1322019114 1322019115 0 32 12 0 0 0  0 0 12 0 0 0  0 0
1322019116 3 OpC 6 opcecm 9 SNMPTraps 8 IBB_COLG 3 Fa2 0  0  0  62
IF Fa2 Down   Root Cause: [redacted]s Fa2  --Sev Major 1824 Generic:
6; Specific: 58916867; Enterprise: .1.3.6.1.4.1.11.2.17.1;
```

Figure 3.7: Format of an alert taken from the DS1 dataset.

information from low level nodes and aggregate them into one alert. Thus, in this type of alerts, the node attribute represents the identity of the gateway and the root cause field may contain other identities for low level nodes.

Figure 3.8 illustrates the format of the alerts related to DS2 dataset. We observe the level of preprocessing applied here by the management staff to filter all the unnecessary fields.

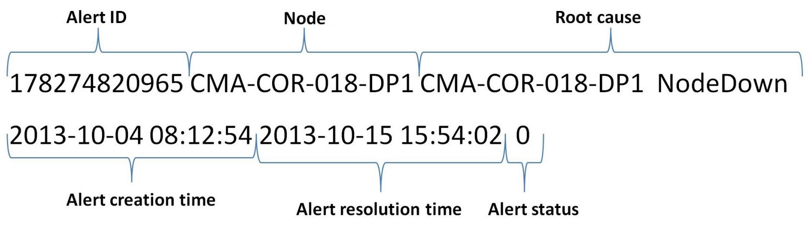


Figure 3.8: Format of an alert taken from the DS2 dataset.

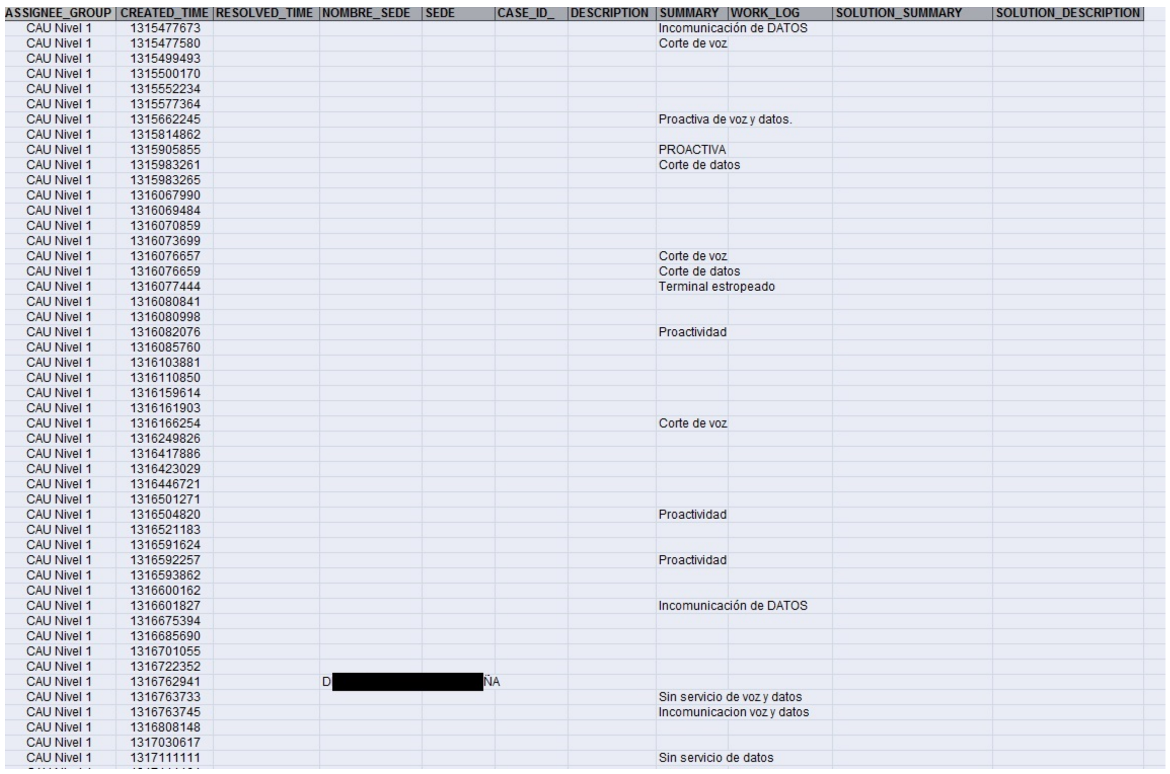
3.2 Insights from Datasets

A major inconvenient to deal with these raw datasets for research purposes is the lack of a "ground truth", *i.e.*, a set of labeled incidents with their corresponding alerts and tickets is not available. Furthermore, and as it will be explained in the following sections, we have found that not all the tickets/alerts are related to real incidents and that some of the tickets/alerts present incoherent, irrelevant or null values. At this point, in order to clarify the nature of the problem, we dig even more into the datasets in order to classify the different types of tickets and alerts, in an attempt to use this information for solving the core problem of this thesis. Next subsections present the information extracted from this study.

3.2.1 Types of Tickets

With a deep manual exploration of the tickets and after many consultations with the management staff, we were able to capture several types of tickets that have different levels of granularity for describing the reported incidents. This has motivated us to study all of these types and try to understand the cause of their existence. We first analyzed the different types of tickets starting from the following hypothesis: in ideal situations, the main target of any ITS system is to register accurate and fully filled informational tickets for describing the incidents as completely as possible. Based on this assumption, in the following, we list all of the types of tickets that we have found, and give some real samples extracted from the case study in order to give details about each one of them.

- *Malformed or void tickets*: During the process of analyzing the attributes of tickets, we have noticed that some mandatory attributes have incomplete or inaccurate information about the described incident. With the help of the management staff, we defined these tickets as *malformed* or *void*, because they are created erroneously either due to a failure in the network management tool, that automatically could generate such kind of tickets, or due to a mis-behavior of any member of the management staff. Figure 3.9 illustrates a snapshot of some samples of malformed tickets extracted from the investi-



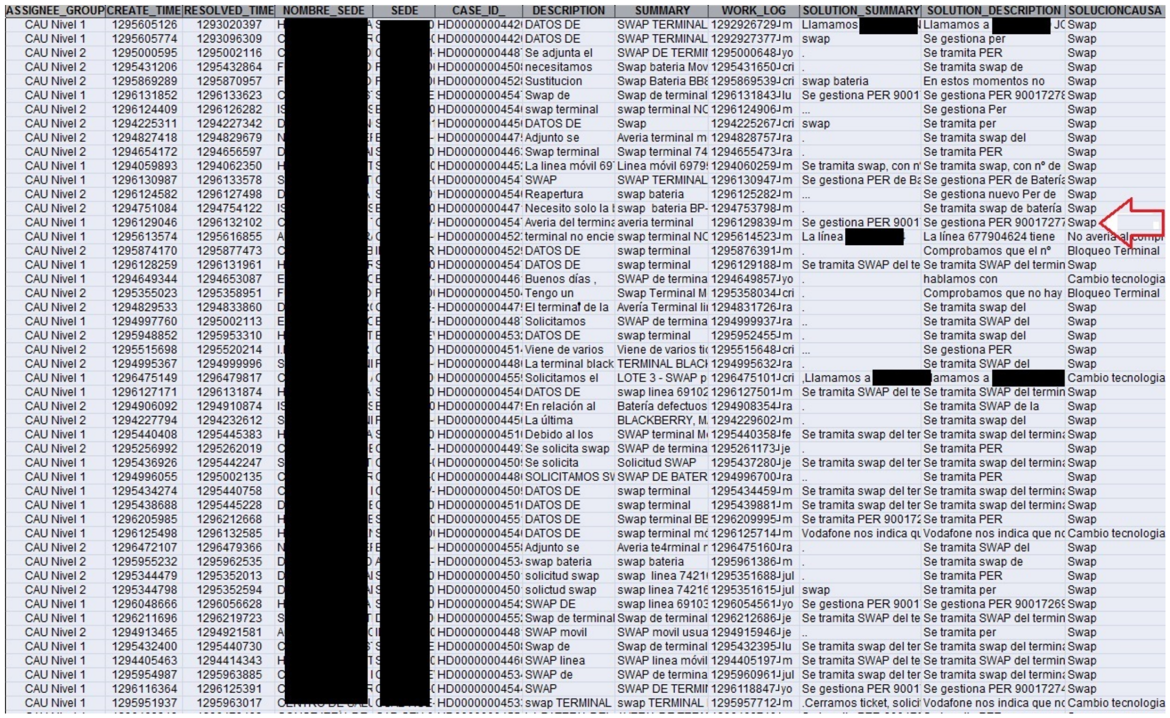
ASSIGNEE_GROUP	CREATED_TIME	RESOLVED_TIME	NOMBRE_SEDE	SEDE	CASE_ID_	DESCRIPTION	SUMMARY	WORK_LOG	SOLUTION_SUMMARY	SOLUTION_DESCRIPTION
CAU Nivel 1	1315477673						Incomunicación de DATOS			
CAU Nivel 1	1315477580						Corte de voz			
CAU Nivel 1	1315499493									
CAU Nivel 1	1315500170									
CAU Nivel 1	1315552234									
CAU Nivel 1	1315577364									
CAU Nivel 1	1315662245						Proactiva de voz y datos.			
CAU Nivel 1	1315814862									
CAU Nivel 1	1315905855						PROACTIVA			
CAU Nivel 1	1315983261						Corte de datos			
CAU Nivel 1	1315983265									
CAU Nivel 1	1316067990									
CAU Nivel 1	1316069484									
CAU Nivel 1	1316070859									
CAU Nivel 1	1316073699									
CAU Nivel 1	1316076657						Corte de voz			
CAU Nivel 1	1316076659						Corte de datos			
CAU Nivel 1	1316077444						Terminal estropeado			
CAU Nivel 1	1316080841									
CAU Nivel 1	1316080998									
CAU Nivel 1	1316082076						Proactividad			
CAU Nivel 1	1316085760									
CAU Nivel 1	1316103881									
CAU Nivel 1	1316110850									
CAU Nivel 1	1316159614									
CAU Nivel 1	1316161903									
CAU Nivel 1	1316166254						Corte de voz			
CAU Nivel 1	1316249826									
CAU Nivel 1	1316417886									
CAU Nivel 1	1316423029									
CAU Nivel 1	1316446721									
CAU Nivel 1	1316501271									
CAU Nivel 1	1316504820						Proactividad			
CAU Nivel 1	1316521183									
CAU Nivel 1	1316591624									
CAU Nivel 1	1316592257						Proactividad			
CAU Nivel 1	1316593862									
CAU Nivel 1	1316600162									
CAU Nivel 1	1316601827						Incomunicación de DATOS			
CAU Nivel 1	1316675394									
CAU Nivel 1	1316685690									
CAU Nivel 1	1316701055									
CAU Nivel 1	1316722352									
CAU Nivel 1	1316762941		D[redacted]ÑA							
CAU Nivel 1	1316763733						Sin servicio de voz y datos			
CAU Nivel 1	1316763745						Incomunicacion voz y datos			
CAU Nivel 1	1316808148									
CAU Nivel 1	1317030617									
CAU Nivel 1	1317111111						Sin servicio de datos			

Figure 3.9: A snapshot of some samples of malformed tickets extracted from the DS1 tickets dataset.

gated case study. It is obvious from these samples that most of mandatory ticket attributes are empty such as DESCRIPTION, TICKET RESOLUTION TIME, SEDE and WORK_LOG, among others. According to the management staff, these attributes are necessary and should contain accurate and full information about the ongoing incidents in order to be valid. In our case, it is hard to get useful information from these tickets regarding the incident and its resolution.

- *Irrelevant tickets:* Besides the malformed tickets, and after manually analyzing other subset of tickets, we have found that despite that they contain full information in all relevant fields, these tickets contain information not related to real network incidents. Consequently, and after validating several samples with the management staff, we called these tickets as *irrelevant* from the point view of real network incidents.

  We have found that, in the studied tickets, or even generically in any ITS system, it is usual that some tickets are created to record issues not directly related to network incidents but to contain information about some other information as, for example, the availability of a new software release in certain network nodes or scheduled maintenance activities. For example, in the case

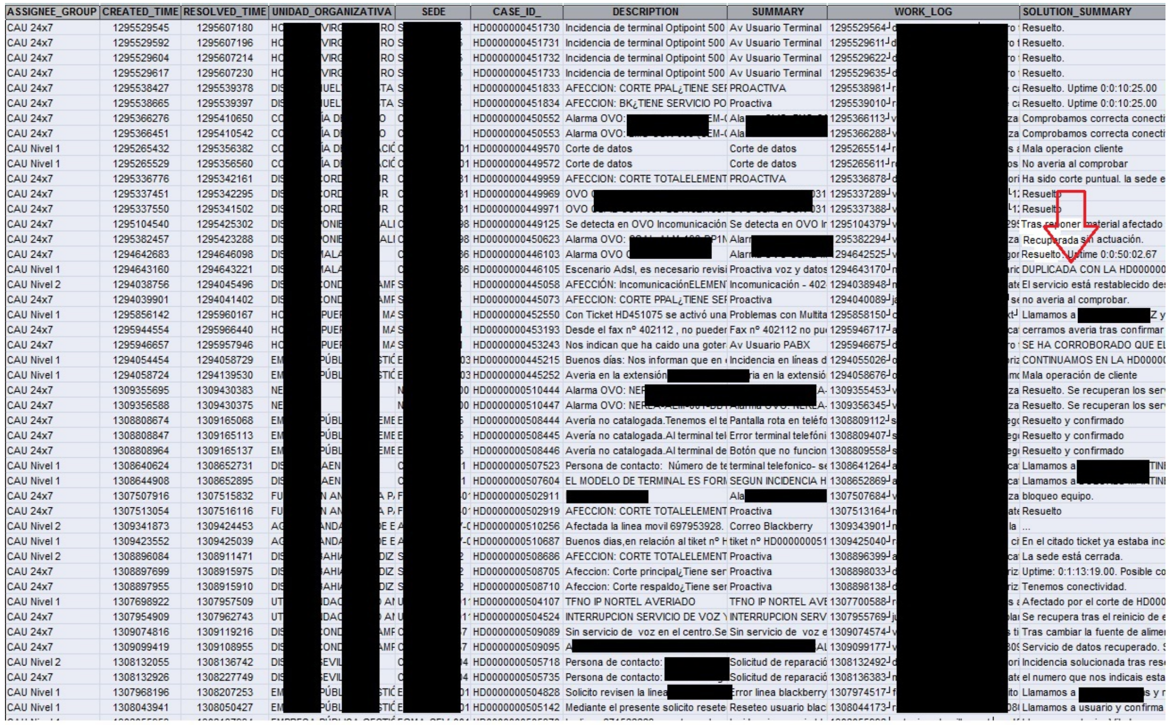


ASSIGNEE_GROUP	CREATE_TIME	RESOLVED_TIME	NOMBRE_SEDE	SEDE	CASE_ID_	DESCRIPTION	SUMMARY	WORK_LOG	SOLUTION_SUMMARY	SOLUTION_DESCRIPTION	SOLUCIONCAUSA
CAU Nivel 1	1295605126	1293020397	[redacted]	[redacted]	HD0000000442	DATOS DE	SWAP TERMINAL	1292926729 m	Llamamos [redacted]	Llamamos a [redacted] JC	Swap
CAU Nivel 1	1295605774	1293096309	[redacted]	[redacted]	HD0000000442	DATOS DE	SWAP TERMINAL	1292927377 m	swap	Se gestiona per	Swap
CAU Nivel 2	1295000595	1295002116	[redacted]	[redacted]	HD0000000448	Se adjunta el	SWAP DE TERMI	1295000648 yo	.	Se tramita PER	Swap
CAU Nivel 2	1295431206	1295432864	[redacted]	[redacted]	HD0000000450	necesitamos	Swap bateria Mov	1295431650 cri	.	Se tramita swap de	Swap
CAU Nivel 2	1295869289	1295870957	[redacted]	[redacted]	HD0000000452	Sustitucion	Swap Bateria BB8	1295869539 cri	swap bateria	En estos momentos no	Swap
CAU Nivel 1	1296131852	1296133623	[redacted]	[redacted]	HD0000000454	Swap de	Swap de terminal	1296131843 lu	Se gestiona PER 9001	Se gestiona PER 90017278	Swap
CAU Nivel 2	1296124409	1296126282	[redacted]	[redacted]	HD0000000454	swap terminal	swap terminal NC	1296124906 m	...	Se gestiona Per	Swap
CAU Nivel 2	1294225311	1294227342	[redacted]	[redacted]	HD0000000445	DATOS DE	Swap	1294225267 cri	swap	Se tramita per	Swap
CAU Nivel 2	1294827418	1294829679	[redacted]	[redacted]	HD0000000447	Adjunto se	Averia terminal m	1294828757 ra	.	Se tramita swap del	Swap
CAU Nivel 2	1294654172	1294656597	[redacted]	[redacted]	HD0000000446	Swap terminal	Swap terminal 74	1294655473 ra	.	Se tramita PER	Swap
CAU Nivel 1	1294059893	1294062350	[redacted]	[redacted]	HD0000000445	La linea móvil 69	Linea móvil 6979	1294060259 m	Se tramita swap, con n	Se tramita swap, con nº de	Swap
CAU Nivel 1	1296130987	1296133578	[redacted]	[redacted]	HD0000000454	SWAP	SWAP TERMINAL	1296130947 m	Se gestiona PER de Ba	Se gestiona PER de Batería	Swap
CAU Nivel 2	1296124582	1296127498	[redacted]	[redacted]	HD0000000454	Reapertura	swap bateria	1296125282 m	...	Se gestiona nuevo Per de	Swap
CAU Nivel 2	1294751084	1294754122	[redacted]	[redacted]	HD0000000447	Necesito solo la	swap bateria BP-	1294753798 m	.	Se tramita swap de batería	Swap
CAU Nivel 1	1296129046	1296132102	[redacted]	[redacted]	HD0000000454	Averia del termina	averia terminal	1296129839 m	Se gestiona PER 9001	Se gestiona PER 9001727	Swap
CAU Nivel 1	1295613574	1295616855	[redacted]	[redacted]	HD0000000452	terminal no encie	swap terminal NC	1295614523 m	La línea [redacted]	La línea 677904624 tiene	No averia al compr
CAU Nivel 2	1295874170	1295877473	[redacted]	[redacted]	HD0000000452	DATOS DE	swap terminal	1295876391 m	.	Comprobamos que el nº	Bloqueo Terminal
CAU Nivel 1	1296128259	1296131961	[redacted]	[redacted]	HD0000000454	DATOS DE	swap terminal	1296129188 m	Se tramita SWAP del te	Se tramita SWAP del termin	Swap
CAU Nivel 1	1294649344	1294653087	[redacted]	[redacted]	HD0000000446	Buenos días ,	SWAP de termina	1294649857 yo	.	hablamos con	Cambio tecnologia
CAU Nivel 2	1295355023	1295358951	[redacted]	[redacted]	HD0000000450	Tengo un	Swap Terminal M	1295358034 cri	.	Comprobamos que no hay	Bloqueo Terminal
CAU Nivel 2	1294829533	1294833860	[redacted]	[redacted]	HD0000000447	El terminal de la	Avería Terminal li	1294831726 ra	.	Se tramita swap del	Swap
CAU Nivel 1	1294997760	1295002113	[redacted]	[redacted]	HD0000000448	Solicitamos	SWAP de termina	1294999937 ra	..	Se tramita SWAP del	Swap
CAU Nivel 2	1295948852	1295953310	[redacted]	[redacted]	HD0000000453	DATOS DE	swap terminal	1295952455 m	.	Se tramita swap del	Swap
CAU Nivel 2	1295515698	1295520214	[redacted]	[redacted]	HD0000000451	Viene de varios	Viene de varios ti	1295515648 cri	...	Se gestiona PER	Swap
CAU Nivel 2	1294995367	1294999996	[redacted]	[redacted]	HD0000000448	La terminal black	TERMINAL BLACK	1294995632 ra	.	Se tramita SWAP del	Swap
CAU Nivel 1	1296475149	1296479817	[redacted]	[redacted]	HD0000000455	Solicitamos el	LOTE 3 - SWAP p	1296475101 cri	,Llamamos a [redacted]	[redacted]amamos a [redacted]	Cambio tecnologia
CAU Nivel 1	1296127171	1296131874	[redacted]	[redacted]	HD0000000454	DATOS DE	swap linea 69102	1296127501 m	Se tramita SWAP del te	Se tramita SWAP del termin	Swap
CAU Nivel 2	1294906092	1294910874	[redacted]	[redacted]	HD0000000447	En relación al	Batería defectuos	1294908354 ra	.	Se tramita SWAP de la	Swap
CAU Nivel 2	1294227794	1294232612	[redacted]	[redacted]	HD0000000445	La última	BLACKBERRY, M.	1294229602 m	.	Se tramita swap del	Swap
CAU Nivel 1	1295440408	1295445383	[redacted]	[redacted]	HD0000000451	Debido al los	SWAP terminal M	1295440358 fe	Se tramita swap del ter	Se tramita swap del termina	Swap
CAU Nivel 2	1295256992	1295262019	[redacted]	[redacted]	HD0000000449	Se solicita swap	SWAP de termina	1295261173 je	.	Se tramita PER	Swap
CAU Nivel 1	1295436926	1295442247	[redacted]	[redacted]	HD0000000450	Se solicita	Solicitud SWAP	1295437280 je	Se tramita swap del ter	Se tramita swap del termina	Swap
CAU Nivel 1	1294996055	1295002135	[redacted]	[redacted]	HD0000000448	SOLICITAMOS SV	SWAP DE BATER	1294996700 ra	..	Se tramita PER	Swap
CAU Nivel 1	1295434274	1295440758	[redacted]	[redacted]	HD0000000450	DATOS DE	swap terminal	1295434459 m	Se tramita swap del ter	Se tramita swap del termina	Swap
CAU Nivel 1	1295438688	1295445228	[redacted]	[redacted]	HD0000000451	DATOS DE	swap terminal	1295439881 m	Se tramita swap del ter	Se tramita swap del termina	Swap
CAU Nivel 1	1296205985	1296212668	[redacted]	[redacted]	HD0000000455	DATOS DE	Swap terminal BE	1296209995 m	Se tramita PER 900172	Se tramita PER	Swap
CAU Nivel 1	1296125498	1296132585	[redacted]	[redacted]	HD0000000454	DATOS DE	swap terminal mó	1296125714 m	Vodafone nos indica qu	Vodafone nos indica que nc	Cambio tecnologia
CAU Nivel 2	1296472107	1296479366	[redacted]	[redacted]	HD0000000455	Adjunto se	Averia te4rminal r	1296475160 ra	.	Se tramita SWAP del	Swap
CAU Nivel 2	1295955232	1295962535	[redacted]	[redacted]	HD0000000453	swap bateria	swap bateria	1295961386 m	.	Se tramita swap de	Swap
CAU Nivel 2	1295344479	1295352013	[redacted]	[redacted]	HD0000000450	solicitud swap	swap linea 7421	1295351688 jul	.	Se tramita PER	Swap
CAU Nivel 2	1295344798	1295352594	[redacted]	[redacted]	HD0000000450	solictud swap	swap linea 74216	1295351615 jul	swap	Se tramita per	Swap
CAU Nivel 1	1296048666	1296056628	[redacted]	[redacted]	HD0000000454	SWAP DE	swap linea 69103	1296054561 yo	Se gestiona PER 9001	Se gestiona PER 90017269	Swap
CAU Nivel 1	1296211696	1296219723	[redacted]	[redacted]	HD0000000455	Swap de terminal	Swap de terminal	1296212686 je	Se tramita SWAP del te	Se tramita SWAP del termin	Swap
CAU Nivel 2	1294913465	1294921581	[redacted]	[redacted]	HD0000000448	SWAP movil	SWAP movil usua	1294915946 je	..	Se tramita per	Swap
CAU Nivel 1	1295432400	1295440730	[redacted]	[redacted]	HD0000000450	Swap de	Swap de terminal	1295432395 lu	Se tramita swap del ter	Se tramita swap del termina	Swap
CAU Nivel 1	1294405463	1294414343	[redacted]	[redacted]	HD0000000446	SWAP linea	SWAP linea móvil	1294405197 m	Se tramita SWAP del te	Se tramita SWAP del termin	Swap
CAU Nivel 1	1295954987	1295963885	[redacted]	[redacted]	HD0000000453	SWAP de	SWAP de termina	1295960961 jul	Se tramita swap del ter	Se tramita swap del termina	Swap
CAU Nivel 1	1296116364	1296125391	[redacted]	[redacted]	HD0000000454	SWAP	SWAP DE TERMI	1296118847 yo	Se gestiona PER 9001	Se gestiona PER 90017274	Swap
CAU Nivel 1	1295951937	1295963017	CENTRO DE SALU	[illegible]	HD0000000453	swap TERMINAL	swap TERMINAL (	1295957712 m	.Cerramos ticket, solici	Vodafone nos indica que nc	Cambio tecnologia

Figure 3.10: A snapshot of some samples of irrelevant tickets extracted from the DS1 tickets dataset.

study considered here, and after discussing this issue with the management staff, they have informed us that they usually use several keywords and terms to distinguish irrelevant tickets such as *"swap terminal"* or *"change technology"*, among others. Figure 3.10 illustrates a snapshot of some samples of irrelevant tickets that are extracted from the case study. For more insights, the ticket marked with the arrow has a description "avería de terminal" (terminal down), and the solution summary is "swap terminal" without any further information. Thus, we claim that this type of tickets are created for solving non network incident-related problems. Furthermore, we observed that a high percentage of these tickets are mainly created by the service desk staff, and not by the management one. This also emphasizes that they are not created as a response to network incidents or to network alerts neither.

- *Duplicated tickets:* Due to the fact that tickets in the ITS might be created and managed by different management groups or even by different members within the same group, we have detected that ITS could contain overlapped or duplicated tickets created for the same incident. This issue might happen due to the format of the *Node name* field in the alerts datasets and the SEDE field in the tickets datasets. We have found that for several alerts that share the same SEDE prefix and have differences in the last number, management staff could create duplicated tickets, each one for every alert type. For example, Node

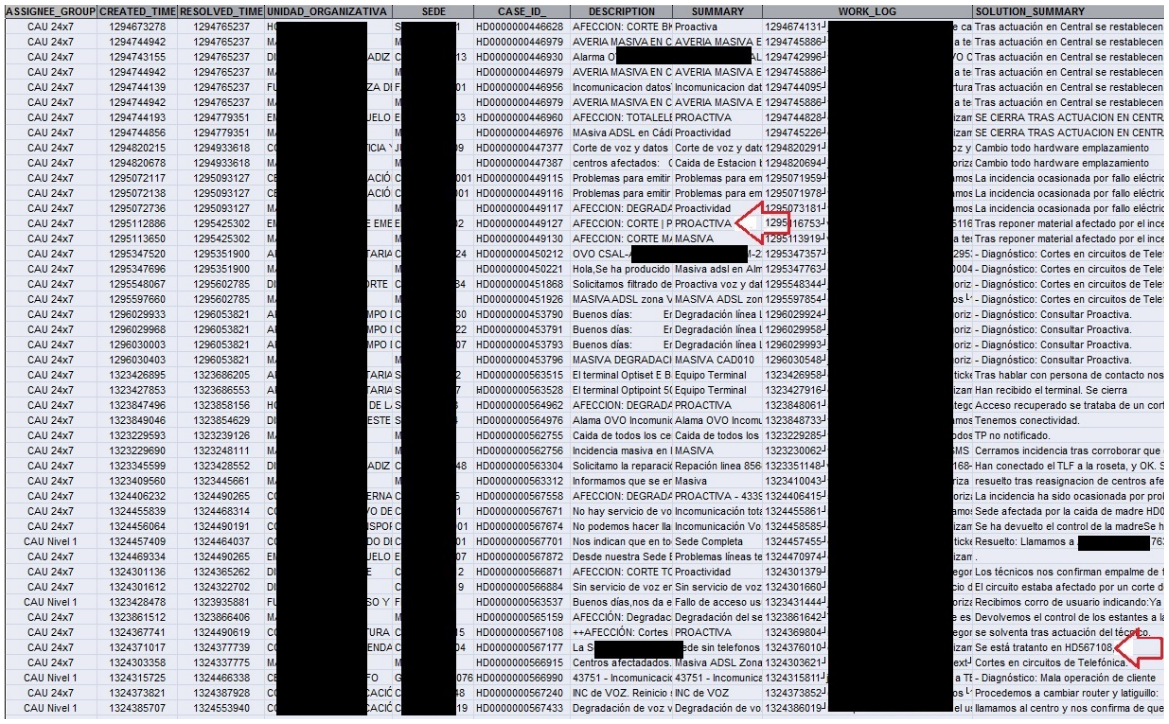


ASSIGNEE_GROUP	CREATED_TIME	RESOLVED_TIME	UNIDAD_ORGANIZATIVA	SEDE	CASE_ID_	DESCRIPTION	SUMMARY	WORK_LOG	SOLUTION_SUMMARY
CAU 24x7	1295529545	1295607180	HO [redacted] VIRG [redacted] RO	S [redacted]	HD0000000451730	Incidencia de terminal Optipoint 500	Av Usuario Terminal	1295529564┘d [redacted] ro	Resuelto.
CAU 24x7	1295529592	1295607196	HO [redacted] VIRG [redacted] RO	S [redacted]	HD0000000451731	Incidencia de terminal Optipoint 500	Av Usuario Terminal	1295529611┘d [redacted] ro	Resuelto.
CAU 24x7	1295529604	1295607214	HO [redacted] VIRG [redacted] RO	S [redacted]	HD0000000451732	Incidencia de terminal Optipoint 500	Av Usuario Terminal	1295529622┘d [redacted] ro	Resuelto.
CAU 24x7	1295529617	1295607230	HO [redacted] VIRG [redacted] RO	S [redacted]	HD0000000451733	Incidencia de terminal Optipoint 500	Av Usuario Terminal	1295529635┘d [redacted] ro	Resuelto.
CAU 24x7	1295538427	1295539378	DIS [redacted] UEL [redacted] TA	S [redacted]	HD0000000451833	AFECCION: CORTE PPAL¿TIENE SEI	PROACTIVA	1295538981┘r [redacted] ca	Resuelto. Uptime 0:0:10:25.00
CAU 24x7	1295538665	1295539397	DIS [redacted] UEL [redacted] TA	S [redacted]	HD0000000451834	AFECCION: BK¿TIENE SERVICIO PO	Proactiva	1295539010┘r [redacted] ca	Resuelto. Uptime 0:0:10:25.00
CAU 24x7	1295366276	1295410650	CO [redacted] ÍA DE [redacted] O	C [redacted]	HD0000000450552	Alarma OVO: [redacted] EM-(	Ala [redacted]	1295366113┘v [redacted] za	Comprobamos correcta conecti
CAU 24x7	1295366451	1295410542	CO [redacted] ÍA DE [redacted] O	C [redacted]	HD0000000450553	Alarma OVO: [redacted] EM-(	Ala [redacted]	1295366288┘v [redacted] za	Comprobamos correcta conecti
CAU Nivel 1	1295265432	1295356382	CO [redacted] ÍA DE [redacted] ACIÓ	C [redacted] 01	HD0000000449570	Corte de datos	Corte de datos	1295265514┘r [redacted] s a	Mala operacion cliente
CAU Nivel 1	1295265529	1295356560	CO [redacted] ÍA DE [redacted] ACIÓ	C [redacted] 01	HD0000000449572	Corte de datos	Corte de datos	1295265611┘r [redacted] os	No averia al comprobar
CAU 24x7	1295336776	1295342161	DIS [redacted] ORD [redacted] R	C [redacted] 31	HD0000000449959	AFECCION: CORTE TOTALELEMENT	PROACTIVA	1295336878┘d [redacted] ori	Ha sido corte puntual. la sede e
CAU 24x7	1295337451	1295342295	DIS [redacted] ORD [redacted] R	C [redacted] 31	HD0000000449969	OVO C [redacted]	[redacted] 031	1295337289┘v [redacted] └1:	Resuelto
CAU 24x7	1295337550	1295341502	DIS [redacted] ORD [redacted] R	C [redacted] 31	HD0000000449971	OVO C [redacted]	[redacted] 031	1295337388┘v [redacted] └1:	Resuelto
CAU 24x7	1295104540	1295425302	DIS [redacted] ONIE [redacted] ALI	C [redacted] 98	HD0000000449125	Se detecta en OVO Incomunicación	Se detecta en OVO Ir	1295104379┘v [redacted] 295	Tras reponer material afectado
CAU 24x7	1295382457	1295423288	DIS [redacted] ONIE [redacted] ALI	C [redacted] 98	HD0000000450623	Alarma OVO: [redacted] P1N	Alarr [redacted]	1295382294┘v [redacted] za	Recuperada sin actuación.
CAU 24x7	1294642683	1294646098	DIS [redacted] MALA [redacted]	C [redacted] 86	HD0000000446103	Alarma OVO C [redacted]	Alarr [redacted]	1294642525┘v [redacted] gor	Resuelto. Uptime 0:0:50:02.67
CAU Nivel 1	1294643160	1294643221	DIS [redacted] MALA [redacted]	C [redacted] 86	HD0000000446105	Escenario Adsl, es necesario revisi	Proactiva voz y datos	1294643170┘n [redacted] aric	DUPLICADA CON LA HD000000
CAU Nivel 2	1294038756	1294045496	DIS [redacted] OND [redacted] AMF	S [redacted]	HD0000000445058	AFECCIÓN: IncomunicaciónELEMEN	Incomunicación - 402	1294038948┘n [redacted] ate	El servicio está restablecido des
CAU 24x7	1294039901	1294041402	DIS [redacted] OND [redacted] AMF	S [redacted]	HD0000000445073	AFECCION: CORTE PPAL¿TIENE SEI	Proactiva	1294040089┘ja [redacted] se	no averia al comprobar.
CAU Nivel 1	1295856142	1295960167	HO [redacted] PUEF [redacted] MA	S [redacted]	HD0000000452550	Con Ticket HD451075 se activó una	Problemas con Multita	1295858150┘c [redacted] xt┘	Llamamos a [redacted] Z y
CAU 24x7	1295944554	1295966440	HO [redacted] PUEF [redacted] MA	S [redacted]	HD0000000453193	Desde el fax nº 402112 , no pueder	Fax nº 402112 no pu	1295946717┘a [redacted] cat	cerramos averia tras confirmar
CAU 24x7	1295946657	1295957946	HO [redacted] PUEF [redacted] MA	S [redacted]	HD0000000453243	Nos indican que ha caido una goter	Av Usuario PABX	1295946675┘d [redacted] ro	SE HA CORROBORADO QUE EL
CAU Nivel 1	1294054454	1294058729	EM [redacted] PÚBL [redacted] STIĆ	E [redacted] 03	HD0000000445215	Buenos días: Nos informan que en	Incidencia en líneas d	1294055026┘c [redacted] riz	CONTINUAMOS EN LA HD00000
CAU Nivel 1	1294058724	1294139530	EM [redacted] PÚBL [redacted] STIĆ	E [redacted] 03	HD0000000445252	Averia en la extensión [redacted]	[redacted] ria en la extensió	1294058676┘c [redacted] amc	Mala operación de cliente
CAU 24x7	1309355695	1309430383	NE [redacted] [redacted]	N [redacted] 00	HD0000000510444	Alarma OVO: NEF [redacted]	[redacted] A-	1309355453┘v [redacted] za	Resuelto. Se recuperan los serv
CAU 24x7	1309356588	1309430375	NE [redacted] [redacted]	N [redacted] 00	HD0000000510447	Alarma OVO: NEF [redacted]	[redacted] A-	1309356345┘v [redacted] za	Resuelto. Se recuperan los serv
CAU 24x7	1308808674	1309165068	EM [redacted] PÚBL [redacted] EME	E [redacted]	HD0000000508444	Avería no catalogada.Tenemos el te	Pantalla rota en teléfo	1308809112┘s [redacted] ego	Resuelto y confirmado
CAU 24x7	1308808847	1309165113	EM [redacted] PÚBL [redacted] EME	E [redacted]	HD0000000508445	Avería no catalogada.Al terminal tel	Error terminal telefóni	1308809407┘s [redacted] ego	Resuelto y confirmado
CAU 24x7	1308808964	1309165137	EM [redacted] PÚBL [redacted] EME	E [redacted]	HD0000000508446	Avería no catalogada.Al terminal de	Botón que no funcion	1308809558┘s [redacted] ego	Resuelto y confirmado
CAU Nivel 1	1308640624	1308652731	DIS [redacted] AEN [redacted]	C [redacted] 1	HD0000000507523	Persona de contacto: Número de te	terminal telefonico- se	1308641264┘a [redacted] cat	Llamamos a [redacted] TINE
CAU Nivel 1	1308644908	1308652895	DIS [redacted] AEN [redacted]	C [redacted] 1	HD0000000507604	EL MODELO DE TERMINAL ES FORI	SEGUN INCIDENCIA H	1308652869┘a [redacted] cat	Llamamos a [redacted] TINE
CAU 24x7	1307507916	1307515832	FU [redacted] N AN [redacted] A P	F [redacted] 01	HD0000000502911	[redacted]	Ala [redacted]	1307507684┘v [redacted] za	bloqueo equipo.
CAU 24x7	1307513054	1307516116	FU [redacted] N AN [redacted] A P	F [redacted] 01	HD0000000502919	AFECCION: CORTE TOTALELEMENT	Proactiva	1307513164┘n [redacted] ate	Resuelto
CAU Nivel 2	1309341873	1309424453	AG [redacted] ANDA [redacted] DE E	A [redacted] Y-(	HD0000000510256	Afectada la linea movil 697953928.	Correo Blackberry	1309343901┘n [redacted] la	...
CAU Nivel 1	1309423552	1309425039	AG [redacted] ANDA [redacted] DE E	A [redacted] Y-(	HD0000000510687	Buenos dias,en relación al tiket nº H	tiket nº HD000000051	1309425040┘r [redacted] ci	En el citado ticket ya estaba inc
CAU Nivel 2	1308896084	1308911471	DIS [redacted] BAHI [redacted] DIZ	S [redacted] 2	HD0000000508686	AFECCION: CORTE TOTALELEMENT	Proactiva	1308896399┘a [redacted] cat	La sede está cerrada.
CAU 24x7	1308897699	1308915975	DIS [redacted] BAHI [redacted] DIZ	S [redacted] 2	HD0000000508705	Afeccion: Corte principal¿Tiene ser	Proactiva	1308898033┘d [redacted] riz	Uptime: 0:1:13:19.00. Posible co
CAU 24x7	1308897955	1308915910	DIS [redacted] BAHI [redacted] DIZ	S [redacted] 2	HD0000000508710	Afeccion: Corte respaldo¿Tiene ser	Proactiva	1308898138┘d [redacted] riz	Tenemos conectividad.
CAU Nivel 1	1307698922	1307957509	UT [redacted] IDAC [redacted] AN	U [redacted] 11	HD0000000504107	TFNO IP NORTEL AVERIADO	TFNO IP NORTEL AVE	1307700588┘r [redacted] s a	Afectado por el corte de HD000
CAU 24x7	1307954909	1307962743	UT [redacted] IDAC [redacted] AN	U [redacted] 11	HD0000000504524	INTERRUPCION SERVICIO DE VOZ Y	INTERRUPCION SERV	1307955769┘ju [redacted] plai	Se recupera tras el reinicio de e
CAU 24x7	1309074816	1309119216	DIS [redacted] OND [redacted] AMF	C [redacted] 7	HD0000000509089	Sin servicio de voz en el centro.Se	Sin servicio de voz e	1309074574┘v [redacted] s ti	Tras cambiar la fuente de alimer
CAU 24x7	1309099419	1309108955	DIS [redacted] OND [redacted] AMF	C [redacted] 7	HD0000000509095	A [redacted]	[redacted] AL	1309099177┘v [redacted] 309	Servicio de datos recuperado. S
CAU Nivel 2	1308132055	1308136742	DIS [redacted] SEVIL [redacted]	C [redacted] 4	HD0000000505718	Persona de contacto: [redacted]	Solicitud de reparació	1308132492┘d [redacted] ori	Incidencia solucionada tras rese
CAU 24x7	1308132926	1308227749	DIS [redacted] SEVIL [redacted]	C [redacted] 4	HD0000000505735	Persona de contacto: [redacted]	Solicitud de reparació	1308136383┘n [redacted] ate	el numero que nos indicais esta
CAU Nivel 1	1307968196	1308207253	EM [redacted] PÚBL [redacted] STIĆ	E [redacted] 01	HD0000000504828	Solicito revisen la linea [redacted]	Error linea blackberry	1307974517┘f [redacted] ito	Llamamos a [redacted] s y r
CAU Nivel 1	1308043941	1308050427	EM [redacted] PÚBL [redacted] STIĆ	E [redacted] 01	HD0000000505142	Mediante el presente solicito resete	Reseteo usuario blac	1308044173┘r [redacted] 08	Llamamos a usuario y confirma

Figure 3.11: A snapshot of some samples of duplicated tickets extracted from the DS1 tickets dataset.

name of alert 1 is QQQ-MMM-WWW-01 and Node name of alert 2 is QQQ-MMM-WWW-02. Thus, in the ticket dataset we may have two tickets, one for every alert that share the same SEDE, QQQ-MMM-WWW. Furthermore, as illustrated in Figure 3.11 that gives some samples of duplicated tickets extracted from the ITS dataset and created by two different management groups, the ticket marked with the arrow has written in its SOLUTION_SUMMARY attribute the following text "DUPLICADA CON LA HD0000000446103". It is explicitly mentioned here that this ticket is a duplicated one. This is also a clear indicator that there exist duplicated tickets.

Furthermore, and after discussing this issue with the management staff, we have concluded that the creation of duplicated tickets might happen due to the lack of coordination or centralization in the tickets creation process. For example, if an element of the network is down, *i.e.*, a router, the staff at SD1 may receive calls from end users complaining about some problems in accessing services or applications affected by this element. Consequently, a ticket containing some preliminary information about the ongoing incident is created by staff members belonging to the SD1 group. At the same time, the management staff may receive alerts triggered by the same element announcing the existence of the same incident. Consequently, the staff creates another



ASSIGNEE_GROUP	CREATED_TIME	RESOLVED_TIME	UNIDAD_ORGANIZATIVA	SEDE	CASE_ID_	DESCRIPTION	SUMMARY	WORK_LOG	SOLUTION_SUMMARY
CAU 24x7	1294673278	1294765237	HC [redacted]	S [redacted] 1	HD0000000446628	AFECCION: CORTE BK	Proactiva	1294674131┘ [redacted] e ca	Tras actuación en Central se restablecen
CAU 24x7	1294744942	1294765237	MA [redacted]	M [redacted]	HD0000000446979	AVERIA MASIVA EN C	AVERIA MASIVA E	1294745886┘ [redacted] a te	Tras actuación en Central se restablecen
CAU 24x7	1294743155	1294765237	DI [redacted] ADIZ	C [redacted] 13	HD0000000446930	Alarma O [redacted]	[redacted] AL	1294742996┘ [redacted] VO C	Tras actuación en Central se restablecen
CAU 24x7	1294744942	1294765237	MA [redacted]	M [redacted]	HD0000000446979	AVERIA MASIVA EN C	AVERIA MASIVA E	1294745886┘ [redacted] a te	Tras actuación en Central se restablecen
CAU 24x7	1294744139	1294765237	FU [redacted] ZA DI	F [redacted] 01	HD0000000446956	Incomunicacion datos	Incomunicacion dat	1294744095┘ [redacted] tura	Tras actuación en Central se restablecen
CAU 24x7	1294744942	1294765237	MA [redacted]	M [redacted]	HD0000000446979	AVERIA MASIVA EN C	AVERIA MASIVA E	1294745886┘ [redacted] a te	Tras actuación en Central se restablecen
CAU 24x7	1294744193	1294779351	EM [redacted] UELO	E [redacted] 03	HD0000000446960	AFECCION: TOTALELE	PROACTIVA	1294744828┘ [redacted] izan	SE CIERRA TRAS ACTUACION EN CENTR
CAU 24x7	1294744856	1294779351	MA [redacted]	M [redacted]	HD0000000446976	MAsiva ADSL en Cádi	Proactividad	1294745226┘ [redacted] izan	SE CIERRA TRAS ACTUACION EN CENTR
CAU 24x7	1294820215	1294933618	CO [redacted] TICIA Y	JI [redacted] 9	HD0000000447377	Corte de voz y datos	Corte de voz y dato	1294820291┘ [redacted] oz y	Cambio todo hardware emplazamiento
CAU 24x7	1294820678	1294933618	MA [redacted]	M [redacted]	HD0000000447387	centros afectados: (	Caida de Estacion b	1294820694┘ [redacted] oriza	Cambio todo hardware emplazamiento
CAU 24x7	1295072117	1295093127	CE [redacted] ACIÓ	C [redacted] 001	HD0000000449115	Problemas para emitir	Problemas para em	1295071959┘ [redacted] amos	La incidencia ocasionada por fallo eléctric
CAU 24x7	1295072138	1295093127	CE [redacted] ACIÓ	C [redacted] 001	HD0000000449116	Problemas para emitir	Problemas para em	1295071978┘ [redacted] amos	La incidencia ocasionada por fallo eléctric
CAU 24x7	1295072736	1295093127	MA [redacted]	M [redacted]	HD0000000449117	AFECCION: DEGRADA	Proactividad	1295073181┘ [redacted] amos	La incidencia ocasionada por fallo eléctric
CAU 24x7	1295112886	1295425302	EM [redacted] E EME	E [redacted] 2	HD0000000449127	AFECCION: CORTE | P	PROACTIVA	1295116753┘ [redacted] 5116	Tras reponer material afectado por el ince
CAU 24x7	1295113650	1295425302	MA [redacted]	M [redacted]	HD0000000449130	AFECCION: CORTE MA	MASIVA	1295113919┘ [redacted] a te	Tras reponer material afectado por el ince
CAU 24x7	1295347520	1295351900	AL [redacted] TARIA	C [redacted] 24	HD0000000450212	OVO CSAL-A [redacted]	[redacted] M-2	1295347357┘ [redacted] 295:	- Diagnóstico: Cortes en circuitos de Tele
CAU 24x7	1295347696	1295351900	MA [redacted]	M [redacted]	HD0000000450221	Hola,Se ha producido	Masiva adsl en Alm	1295347763┘ [redacted] 0004	- Diagnóstico: Cortes en circuitos de Tele
CAU 24x7	1295548067	1295602785	DI [redacted] ORTE	C [redacted] 34	HD0000000451868	Solicitamos filtrado de	Proactiva voz y dat	1295548344┘ [redacted] oriz	- Diagnóstico: Cortes en circuitos de Tele
CAU 24x7	1295597660	1295602785	MA [redacted]	M [redacted]	HD0000000451926	MASIVA ADSL zona V	MASIVA ADSL zon	1295597854┘ [redacted] os └	- Diagnóstico: Cortes en circuitos de Tele
CAU 24x7	1296029933	1296053821	AL [redacted] MPO I	C [redacted] 30	HD0000000453790	Buenos días: Er	Degradación línea L	1296029924┘ [redacted] oriz	- Diagnóstico: Consultar Proactiva.
CAU 24x7	1296029968	1296053821	AL [redacted] MPO I	C [redacted] 22	HD0000000453791	Buenos días: Er	Degradación línea L	1296029958┘ [redacted] oriz	- Diagnóstico: Consultar Proactiva.
CAU 24x7	1296030003	1296053821	AL [redacted] MPO I	C [redacted] 07	HD0000000453793	Buenos días: Er	Degradación línea L	1296029993┘ [redacted] oriz	- Diagnóstico: Consultar Proactiva.
CAU 24x7	1296030403	1296053821	MA [redacted]	M [redacted]	HD0000000453796	MASIVA DEGRADACI	MASIVA CAD010	1296030548┘ [redacted] oriz	- Diagnóstico: Consultar Proactiva.
CAU 24x7	1323426895	1323686205	AL [redacted] TARIA	S [redacted] 2	HD0000000563515	El terminal Optiset E B	Equipo Terminal	1323426958┘ [redacted] ticke	Tras hablar con persona de contacto nos
CAU 24x7	1323427853	1323686553	AL [redacted] TARIA	S [redacted] 7	HD0000000563528	El terminal Optipoint 5(	Equipo Terminal	1323427916┘ [redacted] izan	Han recibido el terminal. Se cierra
CAU 24x7	1323847496	1323858156	HC [redacted] DE L	S [redacted] 3	HD0000000564962	AFECCION: DEGRADA	PROACTIVA	1323848061┘ [redacted] tegc	Acceso recuperado se trataba de un cort
CAU 24x7	1323849046	1323854629	DI [redacted] ESTE	S [redacted] 4	HD0000000564976	Alama OVO Incomunic	Alama OVO Incomu	1323848733┘ [redacted] amos	Tenemos conectividad.
CAU 24x7	1323229593	1323239126	MA [redacted]	M [redacted]	HD0000000562755	Caida de todos los ce	Caida de todos los	1323229285┘ [redacted] odos	TP no notificado.
CAU 24x7	1323229690	1323248111	MA [redacted]	M [redacted]	HD0000000562756	Incidencia masiva en l	MASIVA	1323230062┘ [redacted] SMS	Cerramos incidencia tras corroborar que
CAU 24x7	1323345599	1323428552	DI [redacted] ADIZ	C [redacted] 48	HD0000000563304	Solicitamo la reparació	Repación linea 856	1323351148┘ [redacted] 168-	Han conectado el TLF a la roseta, y OK. S
CAU 24x7	1323409560	1323445661	MA [redacted]	M [redacted]	HD0000000563312	Informamos que se er	Masiva	1323410043┘ [redacted] riza	resuelto tras reasignacion de centros afe
CAU 24x7	1324406232	1324490265	CO [redacted] ERNA	C [redacted] 5	HD0000000567558	AFECCION: DEGRADA	PROACTIVA - 4339	1324406415┘ [redacted] oriz	La incidencia ha sido ocasionada por prol
CAU 24x7	1324455839	1324468314	CO [redacted] VO DE	C [redacted] 1	HD0000000567671	No hay servicio de vo	Incomunicación tota	1324455861┘ [redacted] amos	Sede afectada por la caida de madre HD0
CAU 24x7	1324456064	1324490191	CO [redacted] ISPOI	C [redacted] 01	HD0000000567674	No podemos hacer lla	Incomunicación Vo	1324458585┘ [redacted] izan	Se ha devuelto el control de la madreSe h
CAU Nivel 1	1324457409	1324464037	CO [redacted] DO DI	C [redacted] 01	HD0000000567701	Nos indican que en to	Sede Completa	1324457455┘ [redacted] ticke	Resuelto: Llamamos a [redacted] 76
CAU 24x7	1324469334	1324490265	EM [redacted] UELO	E [redacted] 07	HD0000000567872	Desde nuestra Sede E	Problemas líneas te	1324470974┘ [redacted] izan	.
CAU 24x7	1324301136	1324365262	DI [redacted] E	C [redacted] 2	HD0000000566871	AFECCION: CORTE TC	Proactividad	1324301379┘ [redacted] egor	Los técnicos nos confirman empalme de t
CAU 24x7	1324301612	1324322702	DI [redacted]	C [redacted] 9	HD0000000566884	Sin servicio de voz er	Sin servicio de voz	1324301660┘ [redacted] cio d	El circuito estaba afectado por un corte d
CAU Nivel 1	1323428478	1323935881	FU [redacted] SO Y	F [redacted]	HD0000000563537	Buenos días,nos da e	Fallo de acceso us	1323431444┘ [redacted] oriza	Recibimos corro de usuario indicando:Ya
CAU 24x7	1323861512	1323866406	MA [redacted]	M [redacted]	HD0000000565159	AFECCIÓN: Degradac	Degradación del se	1323861642┘ [redacted] e es	Devolvemos el control de los estantes a la
CAU 24x7	1324367741	1324490619	CO [redacted] TURA	C [redacted] 15	HD0000000567108	++AFECCIÓN: Cortes	PROACTIVA	1324369804┘ [redacted] egor	se solventa tras actuación del técnico.
CAU 24x7	1324371017	1324377739	CO [redacted] ENDA	C [redacted] 04	HD0000000567177	La S [redacted]	[redacted] ede sin telefonos	1324376010┘ [redacted] izan	Se está tratando en HD567108.
CAU 24x7	1324303358	1324337775	MA [redacted]	M [redacted]	HD0000000566915	Centros afectadados.	Masiva ADSL Zona	1324303621┘ [redacted] ext┘	Cortes en circuitos de Telefónica.
CAU Nivel 1	1324315725	1324466338	CE [redacted] FO	G [redacted] 076	HD0000000566990	43751 - Incomunicaci	43751 - Incomunica	1324315811┘ [redacted] a TE	- Diagnóstico: Mala operación de cliente
CAU 24x7	1324373821	1324387928	CO [redacted] CACIÓ	C [redacted] 48	HD0000000567240	INC de VOZ. Reinicio :	INC de VOZ	1324373852┘ [redacted] os └	Procedemos a cambiar router y latiguillo:
CAU Nivel 1	1324385707	1324553940	CO [redacted] CACIÓ	C [redacted] 19	HD0000000567433	Degradación de voz v	Degradación de vo	1324386019┘ [redacted] el us	llamamos al centro y nos confirma de que

Figure 3.12: A snapshot of some samples of related tickets extracted from the DS1 tickets dataset.

ticket related to the same incident. Therefore, this lack of coordination between different groups can lead to the creation of many duplicated tickets.

- *Related tickets:* For more complicated scenarios, we have found that the management staff may create related tickets in which several tickets might be linked to each other if they are related to the same root cause. We have found that this process normally happens when the management staff decides that an ongoing incident is related to another one that has an active ticket. Therefore, they create a new ticket and make a reference to the other one. Furthermore, we have found that there are different methods that could be used by management staff to relate tickets. For example, Figure 3.12 shows a snapshot of some samples of related tickets. In these samples, the management staff refers a ticket to another one just by writing some administrative information such as affected elements, affected services or ticket identities among others, or by using some keywords like "MASIVA" (MASSIVE) and "PROACTIVA" (PROACTIVE) (see the tickets marked with arrows for the representation of both cases).

Type of alert	Classification	Type of alert	Classification	Type of alert	Classification
NodeDown	Yes	CardRemoved	No	FlappingInterface	Informative
HSRPStateChange	Yes	tCyberTechTrap	No	CiscoColdStart	No
Reiterado	Yes	ConnectionDown	No	alarma2	No
ManagementAddressICMPResponseTimeHigh	Yes	linkStatusTrap	No	CarrierTraficoOK_V3	No
CarrierTraficoError V3	Yes	alarma53	No	CustomPollMajor	No
BGPEstablished	Maybe	SNMPWarmStart	No	IslandGroupDown	No
BGPBackwardTransition	Yes	dataPortStatusTrap	No	RateCorrelation	No
IetfVrrpStateChange	Yes	ascEvoStatusNotif_V2	No	alarma3	No
jnxPowerSupplyFailure	Yes	jnxFruPowerOn	No	CardInserted	No
PowerSupplyOutOfRangeOrMalfunctioning	Yes	DuplicateCorrelation	No	alarma5	No
SNMPLinkUp	No	NnmHealthOverallStatus	No	CarrierTraficoSaturado_V3	No
SNMPLinkDown	Maybe	radarDetectTrap	No	CiscoFRURemoved	No
AddressNotResponding	Maybe	jnxPowerSupplyOK	No	CustomPollCritical	No
SNMPColdStart	No	alarma47	No	CustomPollCritical	No
CiscoLinkDown	No	alarma48	No	CustomPolledInstanceOutOfRange	No
CiscoLinkUp	No	alarma49	No	CustomPolledInstanceOutOfRange	No
InterfaceDown	Maybe	alarma50	No	CiscoFRUInserted	No
jnxFruOnline	No	TrapStorm	No	alarma4	No
jnxFruInsertion	No	DDOS_Conex_seg_esp_OK	No		

Table 3.2: Types of alerts and their classifications (critical (Yes)/ non critical (No)) carried out with the help of the management staff.

3.2.2 Types of Alerts

As in the case of ITS, after carefully exploring the alerts datasets, we were able to identify up to 56 different types of alerts. Due to the fact that current monitoring systems are very sensitive, a huge amount of alerts are generated as a result of critical or even non critical issues. In fact, alerts in NMS are usually classified according to their severity and/or criticality. Thus, not all the alerts equally affect the stability of the managed system and, thus, the management staff may be prone to ignore or postpone the creation of a ticket for non critical alerts, especially if they are busy trying to solve an incident with higher priority.

As explained in Chapter 2, a large volume of the alerts in the alerts dataset are classified as normal, duplicated or related to the same root cause. Table 3.2 shows a list of types of alerts in the studied alerts datasets, classified according to their relevance as labeled by the management staff. A large part of the generated alerts are classified as normal alerts and the management staff does not open a ticket for their occurrence.

During the classification process, we have observed that, even though the staff members have a deep knowledge of the managed corporate network, they do not have a clear classification of alerts. Some alerts are clearly classified as critical, others as non critical and the remaining mainly depend on several internal policies and issues. Therefore, the alert classification made by the management staff will help to focus on important alerts only with some degree of confidence.

3.3 A Preliminary Tickets-Alerts Correlation

In this section, and after exploring both alerts and tickets datasets, we discuss in more detail all the preliminary analysis that have been carried out for the purpose of relating tickets with alerts, and the main difficulties that we have faced during this process. Our purpose was to identify the main challenges that should have to be solved during this thesis development.

We have started with a simple correlation scenario that uses a temporal and attribute based similarity method to correlate tickets with alerts. This scenario takes the creation time of a ticket as a reference point and searches backward for correlated alerts that have some attributes in common, mainly the same affected element, *i.e.*, *Node name* field of an alert equals to the SEDE field of a ticket. The main reasoning behind this analysis is that alerts are firstly generated by managed devices and after that management staff decides to open a ticket or not. Thus, the normal behavior is that in which alerts appear earlier than tickets at the management console. In this experiment, we have selected a sample of one week of tickets, taken randomly from DS1, namely in March. The experiment was repeated many times, using a variable time window by changing the beginning and the end of the time window for alerts. For the beginning of the research, the time window is extended with an added number of days. The same also applied for the end of the window, but here we considered three scenarios depending on the different timestamps found in tickets: TICKET CREATION TIME (t^{CT}), TICKET RESOLUTION TIME (t^{RT}), and TICKET VALIDATION TIME (t^{VT}). The experiments were repeated for each of these timestamps and the results were compared. The correlation results are shown in Figures 3.13 to 3.15. These figures represent the mean number of correlated alerts per ticket, percentage of tickets correlated with alerts and percentage of alerts correlated with tickets, respectively. The first measure is calculated as the total number of correlated alerts divided by the total number of correlated tickets. The second one is calculated as the number of correlated tickets divided by the number of raw tickets in the study period, *i.e.*, one week. Finally, the last measure was calculated as the number of correlated alerts divided by the number of alerts in the considered time window.

From these results, we observe that the process of the joint correlation of alerts and tickets is not straightforward. As we have seen, even though the time window is extended to approximately one month, there exist a large portion of tickets and alerts that are not correlated, *i.e.*, 85% for the alerts dataset. As a second interesting finding from these results, note that the mean number of correlated alerts per ticket increases as we forward from ticket creation time towards resolution or validation times. This means that part of the correlated alerts appeared after the creation of the corresponding ticket, *i.e.*, during the ticket lifetime, from ticket's creation to validation.

We also performed a second preliminary experiment to check whether there are duplicated or related tickets in the correlated subset. The results, that are shown in Figure 3.16, represent the amount of alerts correlated with at least two different tickets considering only a sample of raw tickets of one day that are also taken from

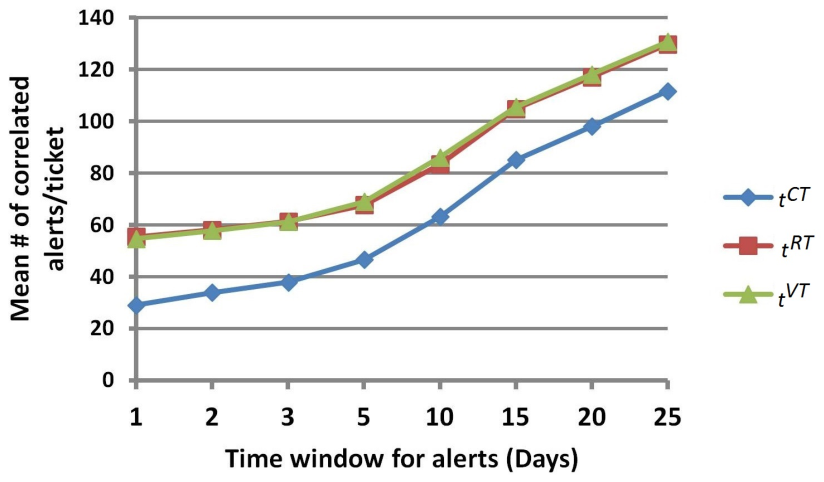


Figure 3.13: Mean number of correlated alerts per ticket in the studied period.

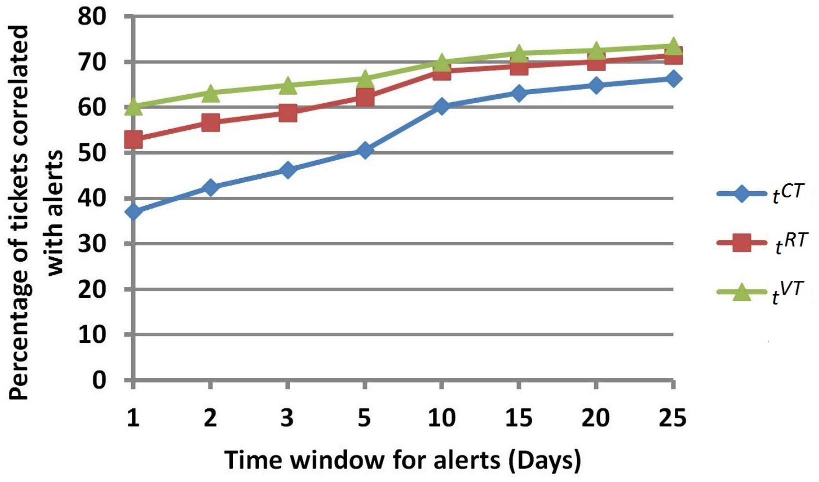


Figure 3.14: Percentage of tickets correlated with alerts in the studied period.

the same month. The results emphasize that there is a level of redundancy in the ITS system, *i.e.*, in some cases there are several tickets created for the same incident. This redundancy has a negative effect on the correlation process, because in this example, alerts are counted many times (one for each correlated ticket).

As the above preliminary results showed that a large portion of tickets and alerts are not related, this point stimulated us to extend the time window to consider the whole alerts dataset. The results, that are shown in Table 3.3, were surprising as there is about 40% of tickets that are not correlated with at least one alert from the amount of alerts of the whole year of DS1 dataset. So, we had some doubts that there is some incoherency in the datasets, or a large number of records in the tickets datasets were not related to real incidents. Furthermore, the mean number of

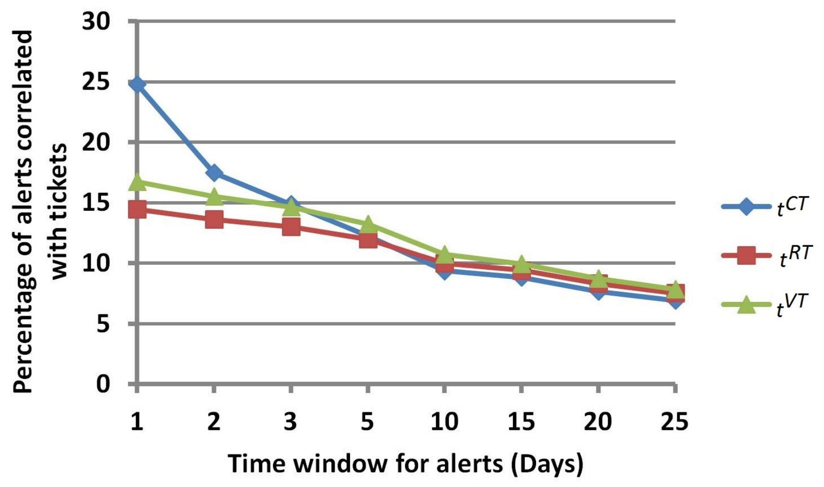


Figure 3.15: Percentage of alerts correlated with tickets in the studied period.

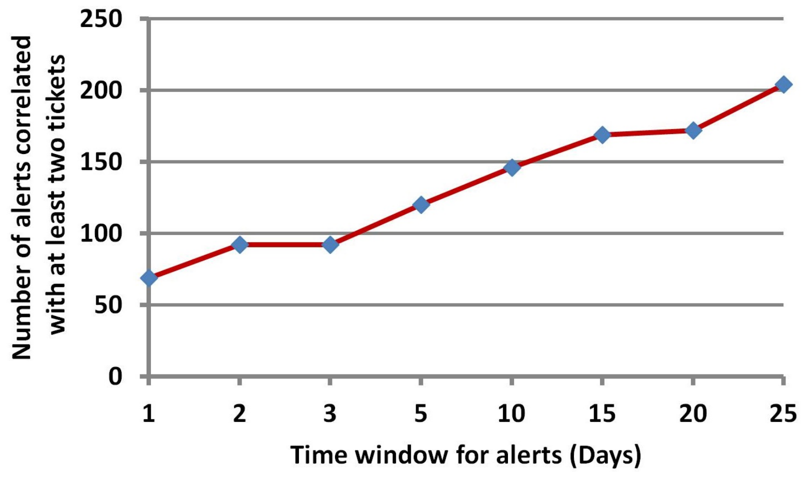


Figure 3.16: Number of alerts correlated with at least two different tickets in the studied period.

Amount of tickets in the study period (one week)	347
Mean amount of alerts in the study period	160,360
Amount of tickets which have at least a correlated alert	229
Total number of correlated alerts in the studied period	20,318
Mean number of correlated alerts per ticket	88.7
Percentage of tickets correlated with alerts	61.06%
Percentage of alerts correlated with tickets	12.67%

Table 3.3: Correlation results for other correlation scenarios.

correlated alerts per ticket is really high, *i.e.*, 88.7, because we have found that many alerts are flapping in the considered time window. Other statistics were obtained in order to find how many affected elements matched in both datasets. Table 3.4 gives

Total number of unique node names in the alerts dataset	8,561
Total number of unique node names in tickets dataset	7,132
Total number of matched node names found in both datasets	5,746
Number of non matched node names found only in alerts dataset	2,815
Number of non matched node names found only in tickets dataset	1,386
Percentage of non matched node names with respect to alerts dataset	32.9%
Percentage of non matched node names with respect to tickets dataset	19.4%

Table 3.4: Some useful analytical results about the matched and non matched affected elements found in both alerts and tickets datasets.

some useful statistics about the matched and non matched list of affected elements extracted from the whole dataset of DS1, tickets and alerts. From these numbers, we observed that there is about a 32.5% of affected elements that belong to the alerts of DS1 dataset and are not matched with any of those extracted from the DS1 tickets dataset. This means that, in the process of correlating alerts with tickets, we expect to have a high percentage of alerts not correlated with tickets. The same is also valid for tickets, in which about 19.4% of the affected elements that belong to the tickets dataset are not matched with any of those extracted from the alerts dataset. Thus, this analysis pointed out to an important issue: there is a significant portion of both datasets that is irrelevant from the point of view of incident solving and, consequently, should be removed from the whole procedure.

In order to study the behavior of the correlated subsets of alerts and tickets, some useful delay values were tested. As illustrated in Figure 3.17, four different delay measures are defined. They are Delay 1, Delay 2, Delay 3 and Delay 4. Delay 1 is calculated as the time difference between the creation time of a ticket and the minimum creation time among correlated alerts. Delay 2 is measured as the delay between the creation time of a ticket and the maximum creation time among correlated alerts. The same is also valid for Delays 3 and 4, but here the values are calculated based on the resolution times of both tickets and alerts. For the same subset of tickets and alerts of the previous experiments, we calculated the mean values and standard deviations of these delays. The results are shown in Table 3.5 for two cases. Case 1 considers all correlated alerts created before the validation of the ticket, while, case 2 also considers the alerts that are resolved after the validation of the ticket. For case 1, in average, Delay 1 is 6.89 h, Delay 2 is -7.65 h, Delay 3 is 23.5 h and finally Delay 4 is 13.7 h. The delay values for case 2 are almost the same as in case 1, but here there are few samples having Delay 4 negative, meaning that some alerts are still active after validating their correlated tickets.

3.3.1 Conclusions from the Preliminary Analysis

These sets of experiments besides others not mentioned here emphasized that the process of the joint correlation of alerts and tickets is not an easy task. Several

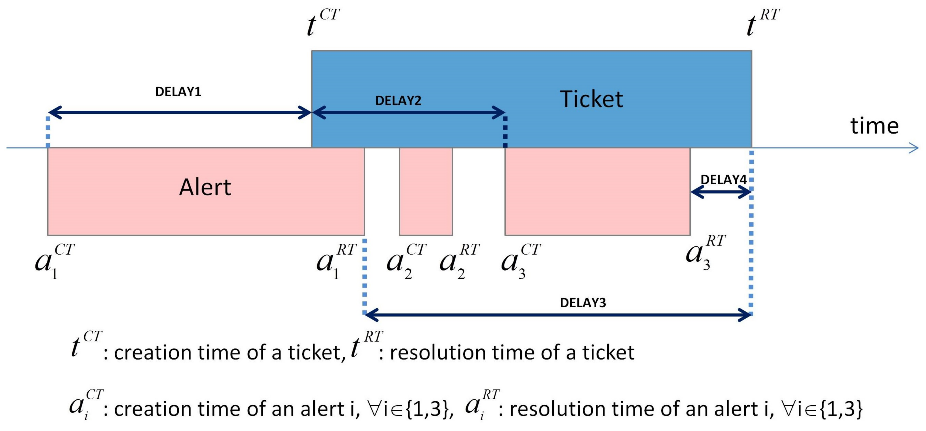

t^{CT}
t^{RT}
Ticket
time
DELAY1
DELAY2
DELAY3
DELAY4
Alert
a_1^{CT}
a_1^{RT}
a_2^{CT}
a_2^{RT}
a_3^{CT}
a_3^{RT}
t^{CT}: creation time of a ticket, t^{RT}: resolution time of a ticket
a_i^{CT}: creation time of an alert i, ∀i∈{1,3}, a_i^{RT}: resolution time of an alert i, ∀i∈{1,3}

Figure 3.17: Implemented scenario for the purpose of analyzing several delay values.

		Delay 1	**Delay 2**	**Delay 3**	**Delay 4**
	Mean	6.89	-7.65	23.6	13.7
case1	**SD**	14.2	28.4	36.2	20.5
	Mean	6.89	-11.9	23.6	8.4
case2	**SD**	14.2	35.2	36.2	23.1

Table 3.5: Values of some delay measures of the joint correlation of alerts and tickets.

issues could appear that may have a negative effect on the correlation results. As a consequence, they must be resolved before applying any joint correlation method.

After exploring the tickets datasets, we observed that there are different types of tickets: malformed, irrelevant, duplicated or related. Malformed and irrelevant tickets are not related to real network incidents and, consequently, they are not related to any alerts from the alerts dataset. Duplicated and related tickets share the same reported incident, so that they are expected to correlate with the same set of alerts, leading to inconsistencies in the results. Furthermore, tickets may encompass a large number of attributes, many of them being not related to incident solving, as they only contain informative data for administrative issues.

Alerts datasets also present similar problems. We observed that a high percentage of alerts are classified by the management staff as normal (non critical), others as critical, and some of them depend on internal policies. We have found that a large part of the alerts are not related to tickets, even those that are classified as critical.

Finally, after applying the variable time window to correlate tickets with alerts, we have found that there are no clear parameters for the window definition when correlating alerts with tickets. Choosing the beginning and the end of the time window will have a great effect on the correlation results and the correlation accuracy as well. For example, using a large time window may lead to many correlated alerts but, at the same time, the number of FPs will be increased as the scenario may consider correlated alerts for other tickets. On the other hand, using small time windows, the percentage of correlated subsets might be very low and the correlation accuracy will be high in terms of *True Positive*s (TPs), but raising a high number of *True Negative*s (TNs) as well.

In the following chapters we tackle all of the above issues by suggesting three different correlation models. In Chapter 4, we propose a generic model for the process of alerts correlation. We consider here all of the stages that are necessary for any alerts correlation method that includes filtering, aggregating, correlation and prioritization of alerts. The same is carried out in Chapter 5, but here the proposed correlation process is targeted at the tickets in the ITS. The aim is to filter malformed and irrelevant tickets, and to correlate overlapped and related tickets that share the same root cause. Finally, in Chapter 6, we propose a model for the joint correlation of alerts and tickets starting from the output of previously proposed models. The aim of this last model is to correlate a high percentage of alerts with tickets keeping the accuracy at higher levels.

Chapter 4

A Generic Model for the Alerts Correlation Process

4.1 Introduction

Many authors have described in their works the process of alerts correlation. While some of them consider the different stages of this process, like the works in [63, 64, 66, 71, 72, 93], they normally focus on proposing and evaluating a multi-component alerts correlation system, tool, method or algorithm. We claim that previous works do not completely aim at providing a comprehensive enough view of the whole process of alerts correlation, mainly because they only present a description of this process as an introduction for describing a specific approach and, in many cases, their point of view is biased towards the techniques that they propose. In this chapter, we suggest an alerts correlation model that is intended to survey all the stages, techniques and methodologies suggested in the state-of-the-art (see Chapter 2), trying to achieve the following two main contributions:

- First, as mentioned in Chapter 2, due to the complexity of the correlation process, most of the proposed models conceptualize it as a set of interrelated processes linked to each other sequentially. Each correlation component performs its tasks locally and sends the results to the next one in a sequential process, without knowing what other components have done. We suggest that this sequentiality does not fit well in the correlation process and, accordingly, in this chapter we give arguments for that and present a new model that considers feedback mechanisms.

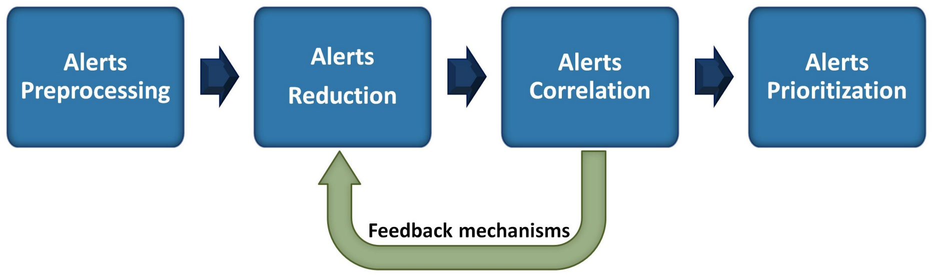


Figure 4.1: Proposed model for the alerts correlation process.

- Second, we make a comprehensive study that reviews the most important existing alerts correlation tools, both open source and commercial, that are currently available, and analyze their alignment with both the state-of-the-art presented in Chapter 2 and the architectural model proposed here.

This chapter is structured as follows. In Section 4.2, we propose the cited generic model for the alerts correlation process. In Section 4.3 we present the comparison of existing commercial products that implement certain correlation techniques.

4.2 A Model for the Alerts Correlation Process

After a thorough analysis of the techniques and models in Chapter 2, we propose a comprehensive model for the process followed to correlate alerts. As shown in Figure 4.1, it is composed of four modules:

- *Alerts preprocessing module:* It accepts raw alerts and converts them into a unified data format understandable by other modules.
- *Alerts reduction module:* This module is intended to filter and validate alerts.
- *Alerts correlation module:* It aggregates similar alerts and converts them into a higher level view.
- *Alerts prioritization module:* It analyzes the severity of alerts on the system and provides a classification methodology.

The correlation model proposed in this chapter has two main features. First, it tries to generalize the usual model by selecting the most important correlation modules generally accepted by the research community in several applications. Note that the scope of the correlation model proposed here is generic, which means that it tries to cover all the aspects of alerts correlation. Nevertheless, it implies that it is not mandatory for any existing implementation to consider all of them. Second, the alerts correlation process is not represented as a sequential process, but it is iterative. Iterations are considered between the alerts reduction and the alerts correlation modules by means of a feedback mechanism between them, aimed at refining their outputs. This feedback mechanism is leveraged to enhance the alerts correlation process in the process of discovering root causes or malicious activities efficiently, due to the fact that, in many real problems, it is quite complicated to identify the causes of a network problem in a single iteration.

In order to assess the iterations in this model, an especial type of alert is defined: *a representative alert*. These are alerts resulting from an iteration in the modules of reduction and correlation. Representative alerts are generated as outputs from the alerts correlation module, and are supposed to be returned and fed back as inputs to the alerts reduction module, where the correlation process repeats itself again. Thus, a representative alert is supposed to represent (substitute) a set of correlated alerts after one of the steps of the overall alerts correlation procedures.

This iterative nature of the model is well-suited for complex or hierarchical scenarios, in which multiple reduction and correlation phases are required. This way, representative alerts can be considered as new alerts that can be filtered and correlated again consequently to produce new representative alerts that can possibly feed the system in new iterations. This mechanism can ease the procedures, simplify the required correlation techniques and, subsequently, improve the results.

The convenience for this iterative process can be clarified by analyzing two examples taken from the network and system security field. As a first case, consider a distributed DoS attack being targeted to a network segment or server. Here, many alerts may be generated at different border routers or links warning about an increase in the traffic. To ease the detection of this attack and to provide some insights on its impact, many steps of reduction and correlation for the alerts can be considered, each one targeted to a different geographic scope of the influence of the attack. For example, a first step would be targeted at generating representative alerts for the routers and links in a subnetwork level. A second iteration would check the scope of the attack at an operator level.

A second case that could illustrate the usefulness of the feedback mechanism is a low-rate scan attack. This kind of attacks are characterized by the scanning of ports at multiple targets at a very low rate, so that detection mechanisms could be bypassed. Yet, while each host is scanned for a reduced number of ports with high scanning period, a visualization of the events in the whole network might reveal a lot of scans in a relatively short period of time, but targeted at many different hosts. In this scenario, one of the detection approaches should go through two different steps of reduction and correlation for the generated alerts. One of the steps should focus on the generation of representative alerts for one of the dimensions of the problem, *i.e.,* correlation of many connection attempts from a single computer or subnet to the different hosts being monitored. After this, representative alerts are generated, and a second iteration should focus on the existence of multiple events of this type in the whole network.

In the next subsections, a detailed description of these modules and their internal phases are provided. Furthermore, as a survey, several significative research efforts related to these phases are also detailed.

4.2.1 Alerts Preprocessing

Currently, in order to provide a better network monitoring and give a global view of intrusion activities, most of the organizations use collaborative and heterogeneous monitoring systems from different vendors. Examples of such monitoring systems are NMS, host IDS, network IDS, firewalls, anti-virus systems, etc. These systems detect abnormality conditions of monitored networks by using different detection methods and, consequently, they generate alerts with different data formats. The alerts preprocessing module implements the necessary processes that are used to clean and transform all raw alerts into an unified and integrated format in order to be understood by other modules.

Davis and Clark [140] presented a review of the state-of-the-art on data preprocessing techniques used by anomaly-based network IDSs. They divided the techniques into different groups according to the network traffic features used for detection, *e.g.*, packet header anomaly detection, protocol anomaly detection, content anomaly detection, etc. They concluded that these techniques are valid for NMSs, but they are insufficient for the security field, since attackers use other methods such as web-based attacks and crafted application data that may need more complicated preprocessing operations.

As shown in Figure 4.2, the preprocessing module can be mainly divided into two phases: *normalization* and *feature construction*, which are discussed below.

Normalization

A normalization phase is first executed to convert heterogeneous alerts from multiple sources into a standard format that is acceptable by other modules. For example, Holub *et al.* [62] used generic log adapters to convert the events from various logs to the common base event format [141] before using them in their proposed correlation engine for system monitoring and testing.

Another relevant example widely used in network IDSs and implemented by the IETF in cooperation with the Intrusion Detection Working Group is IDMEF [59]. It is a kind of reporting language that uses an object-oriented representation to model the alert data generated by IDSs. Each alert is translated into a vector of attributes with the following contents: {*Alert ID, Sensor ID, Detection time, Source IP Address,*

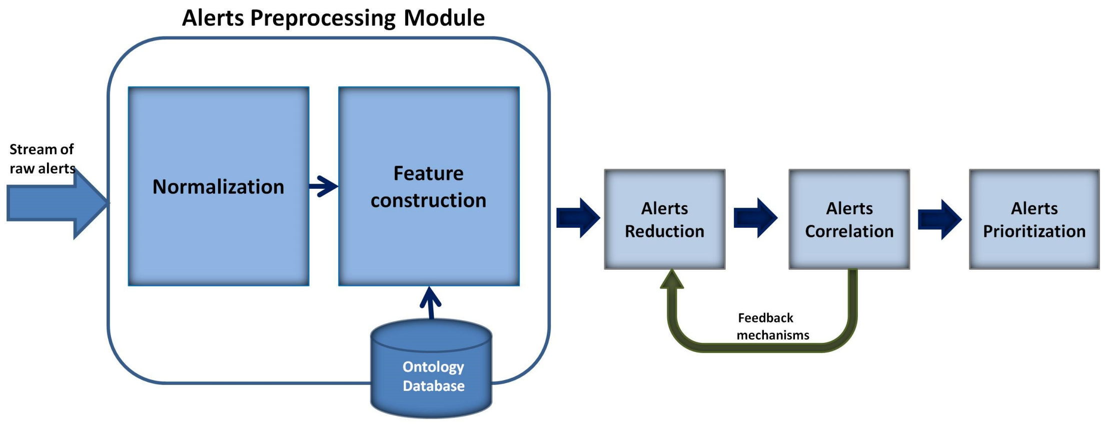


Figure 4.2: Architecture of the alerts preprocessing module.

SourcePort, Destination IP address, Destination port, Service Protocol, Alert Type}. One of the main goals of the IDMEF model is to be able to express relationships between alerts, and exchange procedures for sharing information of interest to intrusion detection and response systems and to the management systems that may need to interact with them. *Document Type Definition* (DTD) has been proposed to describe the IDMEF model data format, with an implementation that uses the *Extensible Markup Language* (XML).

Feature Construction

Since the monitoring systems might omit some of the fields expected in the normalized format of the alerts (*i.e.*, start time, end time), the feature construction phase aims to supply, as accurately as possible, the missing attributes in alerts, and also to create additional features by using different methods, *i.e.*, time-based statistical measures, that would have better discriminative ability than the initial features set.

This issue has been treated in many research contributions. To solve it, some authors have suggested the use of ontology databases consisting of attribute-value pairs for different types of nodes to provide the global information necessary for the feature construction process. For example, IDMEF [59] has strong features that make it able to do this function properly. There are three different time classes defined in the IDMEF standard: CreateTime (the time at which the alert is created), DetectTime (the time at which the events producing an alert are detected), and the AnalyzerTime (the time at which the alert is sent by the IDS to the correlation system). By using different approximation functions, the missing time attributes are substituted, provided that network equipment clocks are synchronized by, for example, using *Network Time Protocol* (NTP) [142].

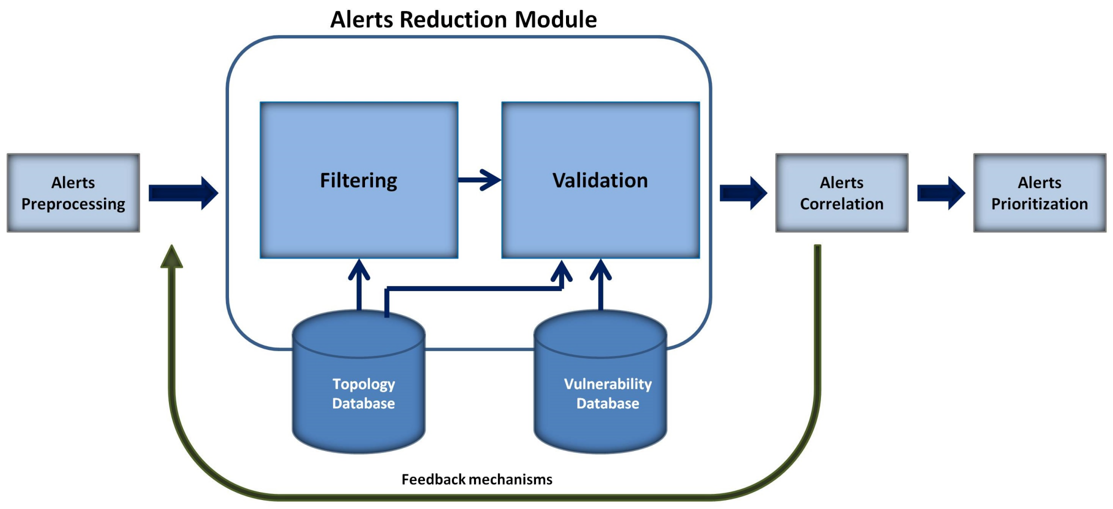


Figure 4.3: Architecture of the alerts reduction module.

4.2.2 Alerts Reduction

As previously discussed, network monitoring and detection systems may overwhelm staff when triggering a huge number of alerts, especially if many of them are redundant or there is a considerable number of false positives (normal events being predicted as abnormal) [143]. Being able to filter out and to validate a high percentage of those uninteresting alerts increases the accuracy of the correlation process and the efficiency as well.

As shown in Figure 4.3, the alerts reduction module is composed of two main phases: *Filtering*, where redundant, duplicated and uninteresting alerts are removed; and *validation*, aimed at making a first assessment of whether an alert is a true or false positive.

Filtering

The preprocessed alerts are fed as inputs to the filtering phase, where uninteresting and redundant alerts are filtered out. In the literature, we can find two types of filters: *pre-defined* and *inferred filters*. *Pre-defined filters* are rules defined by experts and are updated manually. As an example, one of the components of the correlation engine proposed in [84] is a powerful filtering scheme called composite filtering, defined as a set of basic and composite filters organized in a logical tree. Filters are applied in depth-first search order where a descendant filter is considered to be a refinement of the ascendant filter.

On the other hand, *inferred filters* are learned by some kind of inference method. They adapt the rules periodically or on demand by using filtering algorithms that have the ability to use some topological network information that is mainly stored in a topology database. Lin *et al.* [144] proposed an inferred alarm filtering system capable of classifying alarms with high confidence. Chyssler *et al.* [46] proposed an automatic filter detector based on Naive Bayesian learning that is designed to filter syslog records by using a trained classifier that looks at the words contained in the alerts.

Validation

Alerts validation is one of the most relevant and sensitive tasks in the correlation process. The main function of the validation phase is to make a first assessment of the validity of each alert according to its effect on the overall monitored system. In order to actively and accurately distinguish true alerts from FPs, the validation process uses several sources of information and tries to find the logical connections among them. To do that, it extensively makes a deep comparison among all the available sources of information, and then evaluates the coherency between alerts and the monitored system. Here, the main sources of information used by authors are the alerts themselves, which contain useful information about the operating systems, network services and others, and a vulnerabilities database, which stores known exploits and system vulnerabilities information, together with the corresponding security solutions. The validation process is performed by using *passive* or *active* techniques.

Passive techniques carry out the checking of the validity of an alert by using prior information stored in vulnerabilities or topology databases. For example, for remote attacks, passive methods might be used to check whether malicious packets can possibly reach the target, given the network topology and the firewall configuration rules. The main advantage of these techniques is that it is not necessary to perform additional network data collection, thus not interfering with the normal operation of the network. In addition, it is not necessary to perform additional tests that delay the notification to the management staff or the start of active countermeasures. Their main disadvantage is that it is not easy to promptly update the databases; there is a potential difference between the state stored in them and the actual status of the network. Furthermore, the type of information that can be gathered in advance is limited. For example, in the network and system security field, when the signature of an attack is matched against a packet sent to a vulnerable target, the attack could fail for a large number of other reasons (*e.g.*, an incorrect offset for a buffer overflow exploit).

Active techniques update the vulnerabilities database automatically by using several real time scanning tools that actively monitor the whole network and update the system state information that gives a correct view about the current status of the network, and then they update the vulnerability database whenever necessary. For example, one possible approach to carry out active validation in IDSs is to leverage vulnerability scanners [71]. When an attack has been detected, a scanner can be used to check for the vulnerability that this attack attempts to exploit. These techniques have still some drawbacks when compared to passive techniques: they may generate some extra alerts, and they also consume more bandwidth and network resources. Furthermore, the scanning process could make some services crash.

Some authors like Xiao *et al.* [91] adopted a compromise solution that collects together the advantages of both passive and active techniques. Firstly, they collected the configuration information of resources in the protected network to form the vulnerability base of resources, and then they periodically scanned the whole protected network.

Note that this phase normally involves extensive analysis and processing, implying high load procedures. Thus, for performance issues, the validation phase is located after the filtering phase.

4.2.3 Alerts Correlation

The alerts correlation module is at the heart of the whole correlation process. It receives alerts from the alerts reduction module and tries to find out the logical relationships among them, in order to give a higher level and coherent view of the status of the monitored network and discover the root causes. Figure 4.4 shows the structure of the alerts correlation module. Here, the process is divided into two phases, *the aggregation phase*, in which alerts that are expected to share the same root cause are merged, and *the cause identification phase*, in which a higher level processing is handled to generate representative alerts, *i.e.*, alerts that have higher level and richer information about the root causes of a problem.

Aggregation

As said, the target of the alerts aggregation phase is to merge multiple alerts that share the same root cause. Thus, this phase receives alerts from the alerts reduction module and tries to compare them with previously existent or aggregated alerts. The meaning of the new alerts is an induced generalization of those used for the aggregation. Alerts in a given group are supposed to be similar to each other and dissimilar to those in other groups. For example, the works presented in [26, 64, 75–77, 87, 88] used attribute and temporal-based similarity to find the relationships among alerts. Here, certain metrics are first used to compute the similarity among alerts, and then the resulting scores, when compared with some threshold values, determine if these alerts are to be aggregated or not. Sometimes, it is necessary for

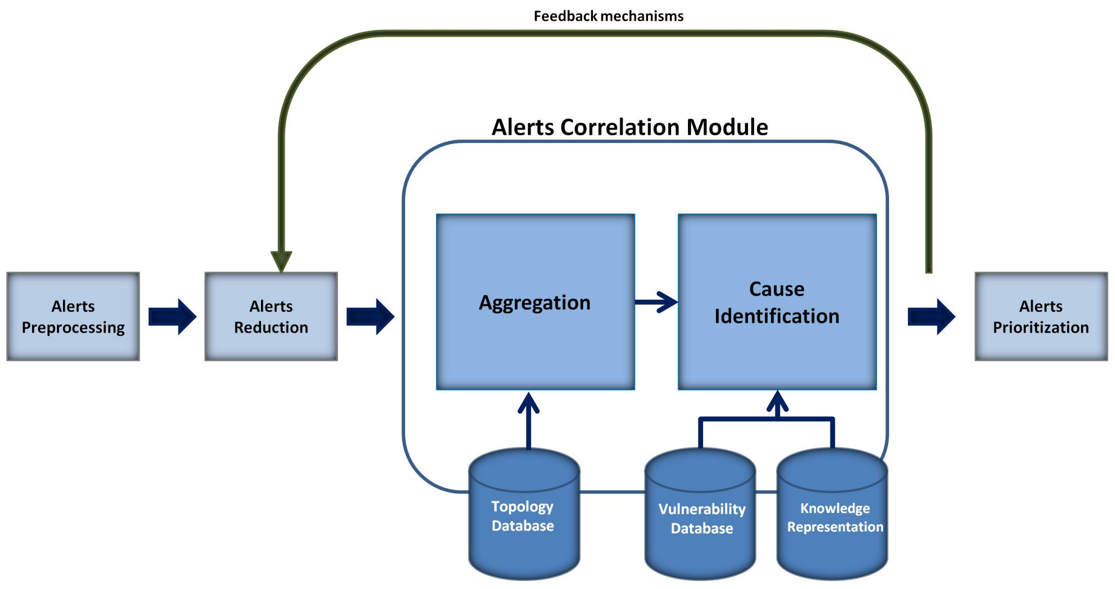


Figure 4.4: Architecture of the alerts correlation module.

the aggregation process to obtain topological information, which is mainly stored, as mentioned in Chapter 2, in a topology database.

Cause Identification

The cause identification phase receives the aggregated alerts generated by the aggregation process and tries to discover and recognize the logical relationships among them. Although this phase may not have a significant effect in reducing the number of alerts, its main purpose is to convert these aggregated alerts into representative alerts, that have more meaningful and semantic contents when compared with the previous ones. In order to correlate aggregated alerts and produce representative alerts, the cause identification phase may use several sources of information like a vulnerability database or knowledge representation (see Section 2.4). For example, the works in [109, 110, 112, 116] propose correlation systems that used knowledge representation methods to correlate alerts, while the works in [45, 53, 71] are some examples of correlation techniques that used the vulnerabilities databases to correlate alerts. In general, this module might implement any combination of the techniques detailed in Chapter 2.

The output of the alerts correlation module has two main destinations: one is the input to the next module, *i.e.*, alerts prioritization for further processing; the other is entering in a cyclic feedback process and being introduced again in the alerts reduction module for filtering and validation. As previously justified, this cyclic process enriches the correlation process with extra information.

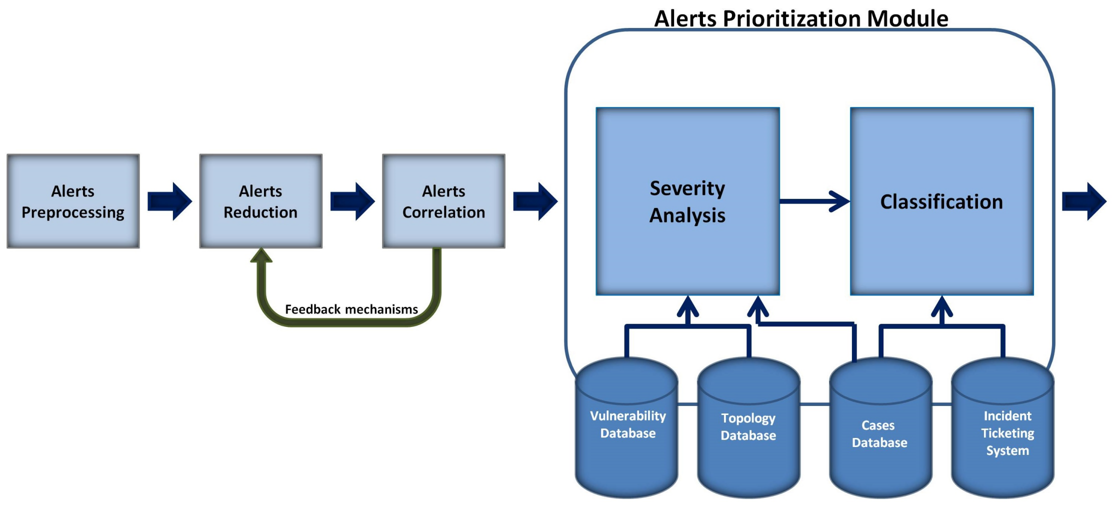


Figure 4.5: Architecture of the alerts prioritization module.

4.2.4 Alerts Prioritization

The last module of the proposed correlation model is called alerts prioritization. Its purpose is to analyze representative alerts received from the alerts correlation module and to classify them based on their severity. As shown in Figure 4.5, the alerts prioritization module can be divided into two main phases: *severity analysis* and *classification*.

Severity Analysis

The purpose of the severity analysis phase is to determine the importance of the network alerts or representative alerts. The severity analysis is a complicated task that needs several sources of information to determine the criticality of a particular alert in the overall system. Certain sources of information, like cases, vulnerability and network topology databases might be useful to provide information about the causes, their descriptions, their dependencies on the operating systems and hardware and software platforms, among others. The severity analysis results depend on the nature of the services and the network being monitored. As an example, a mission impact intrusion report correlation system, M-Correlator, implemented by Porras *et al.* [53], focused on intrusion attacks. Here, the system is used to classify alerts based on severity and take appropriate actions to deal with each one of the alert classes.

Classification

In this final phase, representative alerts are classified based on their severity measurements. The output of this phase is usually sent to a network expert as a report containing alerts and their relevance. In order to efficiently classify alerts, the classification phase may use several sources of information like the cases database or the ITS (see Section 2.4). As these databases contain historical information of previously resolved problems, they might be considered here to help in the alert classification process.

In [53], the classification is done by assigning a relevance score for each alert, that is produced through a comparison of the alert target's known topology against the vulnerability requirements of the incident type. Next, a priority calculation is performed per alert to indicate *(i)* the degree to which the alert is targeted at critical assets; *(ii)* the amount of interest the user has registered for this alert type, and *(iii)* an overall incident rank is assigned to each alert, that brings together the priority of the alert with the likelihood of success.

Zomlot *et al.* [145] presented a method to classify intrusion alerts using an extended Dempster-Shafer theory [146]. The proposed method first calculates confidence scores for hypotheses on an alerts correlation graph, and then it prioritizes the results based on these scores. According to their judge, the computed scores derived from the proposed method provide an effective ranking for the correlated alerts based on the correlations' trustworthiness.

Alsubhi and others [147] proposed two techniques, one for alert rescoring and prioritization based on a fuzzy logic inference mechanism, and another to show the early steps of the attackers. Wallin *et al.* [148] proposed a neural network-based approach for alarm filtering and prioritization by using the knowledge gained from alarm flow properties and trouble ticketing information. Jiang *et al.* [149] proposed a novel peer review mechanism based on rule based systems to rank the importance of alerts resulting in the top ranked alerts being more likely to be true positives. After comparing a metric value against a threshold to generate alerts, the algorithm compares the value with the equivalent thresholds from many other rules to determine the importance of alerts. The output of the classification phase is sent to the network operator as a report that usually contains the causes ordered by severities.

4.3 Existing Alerts Correlation Tools

In this section, some of the alerts correlation systems currently in use are described. The main intention here is not to provide a complete list of the existing tools, but to make a comparison that allows to check which of the previously described techniques are used in deployed alerts correlation systems and which ones are not.

We first classify the correlation tools based on their application field, as discussed in Chapter 2. Some network management tools are listed in Table 4.1. Some of the most important security tools are listed in Table 4.2, and Table 4.3 collects some of the SCADA management systems. In these tables, the “Sources of information (alignment with Section 2.4)” indicates the type of the input data used in the evaluated tool. The “License” column indicates if the tool is commercial,“Comm.”, or open source with *General Public License* (GPL). The “Version” column provides the tool’s version evaluated here. The specific correlation operations and techniques of each tool are listed in the column “Correlation techniques (alignment with Section 2.5.3)”. The alerts correlation modules used in these tools are listed in the column “Correlation process model (alignment with Section 4.2)”. Finally, the webpage for the tool is listed in the column “Homepage”.

4.3.1 Network Management Tools

Table 4.1 summarizes the relevant information for some of the most widely deployed network management tools. In particular, the main focus is on the types of inputs and the correlation techniques used.

Some remarks related to Table 4.1 worth mentioning. First, the majority of these tools are implemented by important private companies like HP, IBM and others. Therefore, most of them are commercially distributed. Second, most of them were implemented to deal with a single type of information sources (Section 2.4), which are alerts with different formats. Third, a large number of these tools implement an expert rules correlation engine. Besides, they use very simple versions of correlation operations such as alerts reduction, clustering, aggregation, etc. We think that the main reason for that is because these tools were not initially designed to deal with alerts correlation specifically, that is, they were implemented for system monitoring and diagnostics purposes. Some of them have implemented other correlation engines, but they have proprietary methods that prevent us from clearly understand which kind of correlation approaches were implemented. But, generally speaking, and according to the data sheets, it seems that they mostly use rule-based correlation approaches. Fourth, we notice that there is a big gap between the sophisticated correlation techniques that are presented in the research, and revised in Section 2.5, and those implemented in these tools. In our opinion, this big gap results from the fact that the industrial and research communities have different algorithms design goals. The research community usually propose algorithms to prove some research ideas, so the algorithms may be very complex in terms of processing time and resource consumption, whereas private companies goal is to build scalable and efficient tools with lightweight algorithms. Furthermore, it seems that they implemented correlation techniques that will save implementation time and cost, in order to enter the market quickly. Thus, most of the commercial tools simply handle only one type of information sources, as illustrated in Table 4.1 –Column 4–, and this can be related to the lack of complex correlation techniques.

4.3.2 Network and System Security Tools

Similar observations result from Table 4.2, that lists a number of security tools. First, some of these tools are open source like Snort and Bro; others are commercial like Bitacora and Net Forensics Console. Second, the majority of these tools were designed to accept Snort alerts and Syslog messages as input formats. Therefore, they were implemented for specific purposes. Third, like in the case of NMS tools, most of the listed security tools have implemented simple correlation operations like filtering, pattern matching, validation and others. Some of them, like Snort and OSSIM, have implemented more complex mechanisms, as rule based correlation techniques. Finally, the security tools present the same issue as the NMS tools regarding the design gap between the complexities of the correlation techniques that were suggested by the research community compared to what is implemented in these tools.

4.3.3 SCADA Tools

With regard to Table 4.3, that contains some of the current SCADA management systems, some observations were also derived. First, unlike the NMS and network and system security tools, we observe that alerts correlation in SCADA systems still needs much work to be done in both industrial and research sectors. Second, the majority of these tools are commercial with free-trial versions. Third, they were implemented for various industrial applications that accept various alarms formats. Last but not least, after a thorough review of the data sheets of these products it has been found that most of them mainly use attribute based similarity for doing several operations like filtering, aggregation and prioritization to discover root causes.

Table 4.1: Survey table of NMS tools.

Product name	Manufacturer	Input information	Sources of information (alignment with Section 2.4)	License	Version	Correlation techniques (alignment with Section 2.5.3)	Correlation process model (alignment with Section 4.2)	Homepage
Pandora FMS	Artica Soluciones Tecnologicas Ltd.	SNMP traps, Syslog	2.4.1. Alerts database	Comm.	4.0.1	Filtering, Aggregation, Validation, temporal-based similarity	Filtering, Validation, Aggregation	http://pandorafms.org/
Osmius	Peopleware SL	SNMP traps, Syslog	2.4.1. Alerts database	Comm.	11.01.0	Filtering, Data mining, Aggregation, Cause identification, attribute-based similarity	Filtering, Aggregation, Cause identification	http://www.osmius.com/
HP OpenView	HP	SNMP traps, Syslog	2.4.1. Alerts database 2.4.2. Topology information	Comm.	2014	Filtering, Verification, Aggregation, expert rules	Filtering, Validation, Aggregation	http://www.hp.com/
Avaya VPFM	Avaya Inc.	SNMP traps, Syslog	2.4.1. Alerts database 2.4.2.Topology information	Comm.	7.5	Filtering, Aggregation, Cause identification	Filtering, Aggregation, Cause identification	http://www.avaya.com/
NagiosXI	worldwide Nagios community	SNMP traps, Syslog	2.4.1. Alerts database	Comm.	2011R1.9	Filtering, Data mining, Pattern matching, Aggregation	Filtering, Aggregation	http://www.nagios.org/
IBM Tivoli Enterpise Consol (TEC)	IBM	SNMP traps, Syslog	2.4.1. Alerts database 2.4.2. Topology information	Comm	39	Normalization, Filtering, Aggregation, Cause identification, Prioritization, expert rules	Filtering, Validation, Aggregation, Cause identification, Classification	http://www.ibm.com/
OpenNMS	The Open NMS Group	SNMP traps, Syslog	2.4.1. Alerts database	Open source GPLv2	1.8.16	Feature construction, Filtering, Validation, expert rules	Feature construction, Filtering, Validation	http://opennms.org/
AggreGate Network Manager	Tibbo Technology Inc.	SNMP traps, Syslog	2.4.1. Alerts database 2.4.2. Topology information	Comm.	4.50.01	Normalization, Filtering, Clustering, Aggregation, expert rules	Feature construction, Filtering, Aggregation	http://aggregate.tibbo.com/
AccelOps	AccelOps, Inc.	SNMP traps, Syslog	2.4.1. Alerts database	Comm.	2010	Normalization, Filtering, Clustering, Validation, Aggregation, Prioritization, expert rules	Normalization, Filtering, Validation, Aggregation, Classification	http://www.accelops.com/
SolarWinds Orion Application Performance Monitor	SolarWinds	SNMP traps, Syslog	2.4.1. Alerts database 2.4.2. Topology information	Comm.	2011	Filtering, Verification, Aggregation	Filtering, Validation, Aggregation	http://www.solarwinds.com/
up.time	Uptime Software	SNMP traps, Syslog	2.4.1. Alerts database	Comm.	7	Filtering, Validation	Filtering, Aggregation	http://www.uptimesoftware.com/
NetCrunch	AdRem Software, Inc.	SNMP traps, Syslog	2.4.1. Alerts database 2.4.2. Topology information	Comm.	6	Normalization, Event suppression, Aggregation, expert rules	Normalization, Filtering, Aggregation	http://www.adremsoft.com/
VMware vCenter Operation Management Suite	Vmware, Inc.	SNMP traps, Syslog	2.4.1. Alerts database	Comm.	5.4.1	Filtering, Validation, Cause identification	Filtering, Validation, Cause identification	http://www.vmware.com/
Verax NMS	Verax Systems Corp.	SNMP traps, Syslog	2.4.1. Alerts database	Comm.	1.9.0	Filtering, Aggregation, expert rules	Filtering, Aggregation	http://www.veraxsystems.com/

Table 4.2: Survey table of network and system security tools.

Product name	Manufacturer	Input information	Sources of information (alignment with Section 2.4)	License	Version	Correlation techniques (alignment with Section 2.5.3)	Correlation process model (alignment with Section 4.2)	Homepage
Snort	Sourcefire, Inc.	Snort alerts	2.4.1. Alerts database	Open source GPL	2.9.2.2	Filtering, Aggregation, Prioritization, expert rules	Filtering, Aggregation, Classification	http://www.snort.org/
Bro	Bro team	Snort alerts, Syslogs	2.4.1. Alerts database	Open source BSD	2.0Beta	Filtering, Pattern matching, temporal-based similarity, dependency graphs	Filtering	http://www.bro-ids.org/
Bitacora	S21sec	Snort alerts, Syslogs	2.4.1. Alerts database 2.4.3. Vulnerabilities database	Comm.	5	Normalization, Filtering, Validation, Real-time correlation	Normalization, Filtering, Validation	http://www.s21sec.com/
OSSEC HIDS	Trend Micro	SMTP traps, IDS alerts, Syslogs	2.4.1. Alerts database	Open source GPLv3	2.6	Filtering, Verification, Severity analysis, Cause identification, expert rules	Filtering, Validation, Cause identification, Severity analysis	http://www.ossec.net/
OSSIM	AlienVault	Snort alerts	2.4.1. Alerts database	Open source GPL	3.1	Normalization, Filtering, expert rules	Normalization, Filtering	http://www.ossim.net/
Prelude SIEM	PreludeIDS Technologies	SMTP traps, Syslogs	2.4.1. Alerts database	Open source GPL	1	Normalization, Filtering, Aggregation, expert rules	Normalization, Filtering, Aggregation	http://www.prelude-ids.com/
ACID	CERT	SMTP traps, Syslogs	2.4.1. Alerts database	Open source GPL	0.0.9.6b23	Filtering, Validation, Clustering, Prioritization, attribute-based similarity	Filtering, Validation, Aggregation, Classification	http://acidlab.sourceforge.net/
neuSECURE	GuardedNet Inc.	SMTP traps, Syslogs, IDS alerts	2.4.1. Alerts database	Comm.	3	Normalization, Validation, Aggregation, Prioritization, Multi-variant correlation	Normalization, Validation, Aggregation, Classification	http://www.guarded.net/
SecurityCenter	Tenable Network security	Syslogs, IDS alerts	2.4.1. Alerts database 2.4.3. Vulnerabilities database 2.4.4. ITS information	Comm.	4.4	Filtering, Validation, Aggregation, Cause identification, Severity analysis, Prioritization	Filtering, Validation, Aggregation, Cause identification, Severity analysis, Classification	http://www.tenable.com/
Net Forensics Console	netForensics	Syslogs, IDS alerts	2.4.1. Alerts database	Comm.	1.2f	Normalization, Filtering, Aggregation, expert rules	Normalization, Filtering, Aggregation	http://www.netforensics.com/
ArcSight ETRM Platform	Arcsight Inc. HP Company	Alerts, Syslogs	2.4.1. Alerts database 2.4.2. Topology information 2.4.3. Vulnerabilities database	Comm.	2011	Aggregation, Prioritization, alerts correlation	Aggregation, Classification	http://www.arcsight.com/
Virtuoso	ConostixInc.S.A.	Syslogs, IDS alerts	2.4.1. Alerts database	Comm.	Hardware LogCollector LC-360	Normalization, Filtering, real-time correlation	Normalization, Filtering	http://www.conostix.com/product-virtuoso.html/

Table 4.3: Survey table of SCADA tools.

Product name	Manufacturer	Input information	Sources of information (alignment with Section 2.4)	License	Version	Correlation techniques (alignment with Section 2.5.3)	Correlation process model (alignment with Section 4.2)	Homepage
SIMATIC WinCC	Siemens	Alerts	2.4.1. Alerts database	Comm.	7	Feature construction, Filtering, Aggregation, attribute-based similarity	Feature construction, Filtering, Aggregation	http://www.siemens.com/
U.C.ME -OPC	Control-See	Alerts	2.4.1. Alerts database	Comm.	2012	Filtering, Validation, Aggregation, Severity analysis, Classification, attribute-based similarity	Filtering, Validation Aggregation, Severity analysis, Classification	http://www.controlsee.com/
Shopfloor-Online	Lighthouse Systems	Alerts	2.4.1. Alerts database	Comm.	2009	Aggregation, Classification, attribute-based similarity	Aggregation, Classification	http://www.lighthousesystems.com
PlantTriage	ExperTune	Alerts	2.4.1. Alerts database	Comm.	9	Filtering, Clustering, Prioritization, Cause identification, sequential-based methods/Graphs	Filtering, Aggregation, Classification, Case identification	http://www.expertune.com
PlantTriage	ExperTune	Alerts	2.4.1. Alerts database	Comm.	9	Filtering, Clustering, Prioritization, Cause identification, sequential-based methods/Graphs	Filtering, Aggregation, Classification, Case identification	http://www.expertune.com
WINLOG pro	SIELCO SISTEMI	Alerts	2.4.1. Alerts database	Comm	2.07.11	Normalization, Prioritization	Filtering, Classification	http://www.sielcosistemi.com/
AggreGate SCADA	Tibbo Technology Inc.	Alarms	2.4.1. Alerts database	Comm.	40.50.01	Normalization, Filtering, Clustering, attribute-based similarity	Normalization, Filtering, Aggregation	http://www.aggregate.tibbo.com/
IGSS	7-Technologies A/S	Alerts	2.4.1. Alerts database	Comm.	9	Feature construction, Filtering, expert rules	Feature construction, Filtering	http://www.igss.com/
Argos	Centro de Innovacion Tecn. del Aluminio	Alerts	2.4.1. Alerts database	Open source GPLv3	0.6.670	Normalization, Aggregation	Normalization, Aggregation	http://www.cintal.com.ve/
Mango M2M2	Serotonin Software Tech.	Alerts	2.4.1. Alerts database	Comm	1.13.0	Filtering, Prioritization	Filtering, Classification	http://www.mango. serotoninsoftware.com/
IntegraXor	EcavaSdn. Bhd	Alerts	2.4.1. Alerts database	Comm	3.7.1	Filtering, Grouping, Cause identification	Filtering, Aggregation, Cause identification	http://www.integraxor.com/
MESbox SCADA	OridinalSotware Inc.	Alerts	2.4.1. Alerts database	Comm	2012	Filtering, Aggregation, Severity analysis	Filtering, Aggregation, Severity analysis	http://www.ordinal.fr/

Chapter 5

A Model for Incident Tickets Correlation

5.1 Introduction

As explained in Section 1.5, ITSs have been introduced as a main tool to assist in the incident management process [30–32]. They are databases that store reports in a specific form-based structure, and a set of tools enabling actions to create, update and resolve any network incident reported by customers, organization employees, or monitoring systems. They might also contain administrative information about customers, workarounds to be applied for common incidents, and other similar data.

Normally, there are two main different procedures for creating incident tickets. On the one hand, they can be created by the management staff in response to a network failure discovered by management software, *e.g.*, alerts from OpenView [28] or other network management platforms. On the other hand, they can also be created by the SD staff in response to a call from an end user facing problems in accessing some services or applications (for more information see Section 1.4.1).

An incident resolution begins with the creation of a ticket that contains the information describing the ongoing incident. Next, the ticket may be sequentially reassigned to several technical groups involved in the incident solving procedure. Finally, it is validated when the incident is completely solved and, optionally, a crosscheck of the solution is made. Therefore, the ticket may pass through several hands and undergo various degrees of escalation with respect to incident severity or customer priority before being completely acknowledged.

Although ITSs play an important role in the incident management, the process of creating the tickets is not completely systematic and may be incoherent and inefficient. For example, the management team may create so-called *irrelevant tickets*, *i.e.*, tickets that do not reflect the existence of an actual network problem; or it may create multiple tickets that are related to the same incident. It is also possible that different staff groups or departments in a company create different tickets for a single incident, as pointed out in Section 3.2.1.

In this chapter, we propose a model for incident tickets correlation. The main objective of the correlation of tickets is to ideally obtain a single ticket per incident. This single ticket should merge all the information contained in an ITS regarding the incident. The remainder of this chapter is organized as follows. Section 5.2 summarizes the research efforts done in the tickets management and correlation field. Section 5.3 presents an incident ticket lifecycle, while a model for incident tickets is proposed and discussed in Section 5.4. In Section 5.5 we explore how to enhance the information obtained from an ITS by means of developing a correlation method specifically designed for tickets. Finally, the proposed correlation method is experimentally evaluated in Section 5.6.

5.2 Related Work

A thorough review of the literature on tickets management and correlation shows several research tendencies, as emphasized by Lewis and Dreo [31]. In this survey, the authors analyzed the challenges and research trends for extending ITSs for the automatic generation of tickets, the diagnosis of faults and the correlation of multiple views of network incidents and behavior. For this, Lewis and Dreo suggested various techniques, such as the use of filtering and grouping of tickets with respect to language, function, time, and topology; the use of rule-based reasoning or case-based reasoning for acquisition and representation of fault diagnostics knowledge; or the use of fuzzy logic for correlating multiple views of network incidents.

A similar work was done by Johnson, who proposed the RFC 1297 [32]. This report mainly discussed the most important extensions that can improve the network operations efficiency, and emphasized the importance of ITSs for network management teams. Nevertheless, it did not provide a methodology to analyze the contents of a ticket itself.

Dreo [150] went forward and proposed the use of tickets correlation for the discovery of problems and the access to problem-solving expertise. One of the main conclusions of Dreo's work is that a high-quality tickets correlation needs to use good models for the functional and topological aspects of any network service.

Miao *et al.* [151, 152] focused on enhancing the ticket management lifecycle by proposing a unified ticket generation model that characterizes the lifecycle of a ticket using both the content and the routing sequence of the ticket, and a Markov model-based approach to predict the resolver of the ticket based on the expert group that previously processed it. The aim was to enhance the routing and minimize the number of transfer steps before it reaches a resolver.

Tang *et al.* [153] suggested an automatic approach to discover the false negatives from those incident tickets that are created by humans in order to improve the configurations of the monitoring system. They applied a text classification model for analyzing the descriptions of tickets and identifying the corresponding system issues.

The work done by Li *et al.* [154] was intended to give an estimate of the mean effort for an incident ticket, by means of a two-stages approach. First, a meta-model was proposed, and some handling priority rules were used to compute what the authors call "attention duration". Then, a maximum likelihood approach was used to estimate the mean effort for a ticket class by using such attention information.

However, in spite of the above works about tickets management, there is not much work focused on tickets correlation in the literature, except for some preliminary works that were done in order to achieve different specific purposes. Next, a summary of these efforts is presented.

Some authors tried to correlate the tickets in an ITS with another sources of information, as in [155–158]. In these papers, the authors proposed models to correlate two types of tickets, the resource tickets, that are created by the monitoring system, and the service tickets, that are generated by the management teams. Their main goal was trying to improve the accuracy and effectiveness of the management process in real time. The authors concluded that tickets overlapping is one of the main challenges for their models, but they did not provide any solutions to handle it.

A similar work was also proposed in [159–162], in which the authors used simple tickets preprocessing operations to reduce the total number of tickets before correlating them. Nevertheless, the focus of these papers was to study and characterize the nature and causes of routing changes and the observed instability. Therefore, they did not deeply analyze ITSs, and the correlation of the tickets was not targeted at reaching one ticket per incident.

Other approaches used different techniques to process the tickets and extract the description of the incidents, like the one of Potharaju *et al.* [54]. In this work, the authors proposed NetSieve, a system that analyzes natural language text in tickets to infer the problem symptoms, troubleshooting activities and resolution actions. They used statistical natural language processing, knowledge representation and ontology modeling to achieve these goals.

Medem *et al.* [163] used machine learning techniques to process the tickets and extract a concise description of the incidents. To do that, they modeled each ticket as a vector of keywords whose frequencies are used as weights to create a hierarchical clustering. The resulting clusters are used to correlate the tickets and extract the incident. However, despite the usefulness of clustering tickets that share the same keywords, it was limited to just some free-text fields and ignored some important features such as timestamps and topological information.

In addition to the papers described above, some researchers used ITSs for other purposes. For example, Tanaka [164] and Pándi [165] made some statistical measurements and characterized tickets from several networks taken as case studies. From these analyses, the authors proposed some recommendations to enhance the performance of ITSs and their use in the context of SLA. They also presented a comprehensive and detailed taxonomy for the categorization of equipments, services, and users affected by events. However, in their work they did not focus on preprocessing operations or the correlation of tickets from the point of view of reducing the number of tickets per incident.

Other authors used ITSs as an assessment tool to validate their research ideas. This is the case of the work done by Huang *et al.* [166]. They proposed to use network-wide analysis of routing information to diagnose network disruptions. In a similar work, Labovitz *et al.* [167] made an experimental study of Internet stability and the origins of failure in Internet backbones utilizing the information provided by an ITS.

Finally, despite the availability of many commercial service desk products, such as HP Service Center [28], ServiceNow [168], BMC Remedy [169], or Tivoli SCCD [170], few efforts have been devoted to enhance the tickets creation process itself. Most of these tools use their own procedures for resolving network incidents and enhancing the whole system to comply with SLAs. For the time being, and to the best of our knowledge, the procedures used for incident tickets correlation in these tools are not publically available, because vendors are not interested in publishing them and, therefore, they keep them hidden from the public due to competitive issues. So, the inherited complexity of these tools makes it difficult to inspect which type of correlation operation they use, if any.

In summary, as previously mentioned, some authors pointed out an important problem during the process of handling the tickets in ITSs, that is, the redundancy in creating tickets; all tickets that have the same root cause should be correlated. Furthermore, other authors argued that good models for tickets' correlation need to use the functional and topological aspects of any network service or element. Consequently, the main objective of this chapter is to propose a model to correlate incident tickets that targets at improving the ITSs to achieve the ideal situation at which just a single ticket per incident exists. Furthermore, the topological and temporal information are used as the main dimensions to achieve this purpose.

To the best of our knowledge, there are no systematic methods currently available for improving the efficiency of the tickets creation process. Previous work on ITSs has focused on improving the whole system to achieve the corresponding SLAs. Notwithstanding with this, few efforts have been devoted to optimize the efficiency of the tickets creation process itself that, at the same time, would improve the overall task of solving the incidents. In our opinion, this is partly due to the fact that the ticketing information is confidential, and this fact hides both the real tickets and the used procedures from the research community. Furthermore, as previously mentioned, vendors are not interested in publishing neither the data nor the procedures due to competitive issues, while management teams either do not keep track of this data or treat it as confidential information due to similar motives. Therefore, in this context, we present three main contributions in this chapter.

- First, a novel model for an incident ticket is proposed. Unlike others, we try to make the model as general as possible by focusing only on the generic fields that appear in tickets such as staff information, temporal and topological information, etc.

- Second, we propose an incident tickets correlation model targeted at improving the ITS to achieve the ideal situation at which just a single ticket per incident exists. The topological and temporal information are used as the main dimensions for the proposed tickets correlation process.

- Third, we propose some metrics used to measure the redundancy in the ITS, according to the proposed correlation method. These metrics are then used to evaluate the procedures for creating tickets.

5.3 The Incident Ticket Lifecycle

As previously discussed, the principal motivation of this work is to find mechanisms for building a simple and manageable system to deal with incidents in a network. ITSs are a main tool used to record and report incidents within an operational system. Therefore, they are considered as a main tool for tracking resolution activities associated with incidents. For this reason, the work presented here will rely on the management of tickets, under the hypothesis that they are the best tool for representing an incident lifecycle. Nevertheless, it is worth to mention here that, although there is a direct relationship between tickets and incident lifecycles, there could exist significant differences between them that will be discussed in what follows.

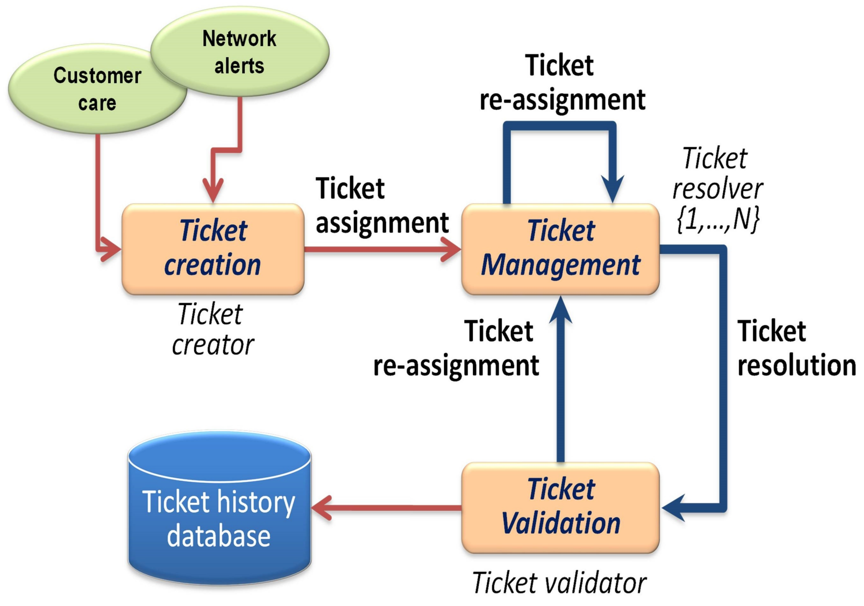


Figure 5.1: Generic incident ticket lifecycle.

For these reasons, the starting point is the understanding of such a ticket lifecycle. Thus, in Figure 5.1 a new generic incident ticket lifecycle is proposed. It consists of the different stages and actions involved in handling with the ticket. They are: *ticket creation*, *ticket assignment*, *ticket management*, *ticket reassignment*, *ticket resolution* and *ticket validation*. In what follows, the different stages and their functions are described in more detail.

- **Ticket creation:** An incident ticket is created by a management staff or a SD staff, here denoted as ticket creator, either: *(i)* in response to the reception of network alerts triggered by proactive network monitoring systems due to service disruptions or failures of network elements; or *(ii)* in response to calls or notifications, typically coming from customer care units or directly from end users who face technical problems in the access to services or applications.

  In a normal situation, the time elapsed between the origin of an incident and its corresponding ticket creation time should not be long, although in many situations it can be in the range of minutes or hours, especially when tickets are created manually and not by an automated system. In these situations, the ticket creator may register the timestamp corresponding to the incident as reported by the monitoring systems, or the time when a customer reported a failure or a management team member detected it.

- **Ticket assignment:** After being created, a ticket is then handed off to an appropriate technical person or group in the company, denoted as ticket resolver, who is expected to resolve the incident.

- **Ticket management:** This is the main stage in the incident ticket lifecycle. During it, the *ticket resolver* analyzes the incident reported in the ticket, finds potential solutions, tests them and reports the results.

- **Ticket reassignment:** It could happen that the ticket resolver, after having investigated the incident –and possibly having also applied several solutions– detects that the incident solution is out of the scope of his/her responsibilities. In this situation, the ticket resolver will again hand off the ticket to another group for its management. This process is the ticket reassignment. In general, when this happens, a ticket can be managed by N different ticket resolvers, and, therefore, it can go through the ticket management stage several times.

  Normally, before doing a reassignment, the ticket resolver registers in a work-log (on the ticket itself) the different activities carried out to solve the incident and his/her results.

- **Ticket resolution:** Once the incident reported on the ticket is solved or a work-around is discovered, the ticket resolver registers the final solution and the corresponding tests and results that are applied to the ticket. Finally, he/she notifies that the incident is resolved.

- **Ticket validation:** After the ticket resolution, the ticket may arrive to another member or group, denoted as *ticket validator*. It is in charge of verifying whether the solution is satisfactory or not. If it is validated, the ticket is usually saved in a ticket history database that is commonly used to build knowledge for future incidents. On the contrary, if the ticket solution is not validated, the ticket is reassigned again to a ticket resolver. Therefore, the ticket lifetime can be defined as the time elapsed between the creation of a ticket and its resolution or validation in some cases.

5.4 A Model for Incident Tickets

ITSs produce tickets in different and heterogeneous formats, containing a wide variety of fields with several possible values for them. Some of these fields are inherent to the operation of the ITS, and therefore, they could be considered as generic fields. Others are specific, *i.e.*, they collect specific requirements established by the different management teams.

As mentioned in the related work section, it has been detected that there are no successful efforts in the research literature to develop a model for incidents. Here, a generic model for an incident ticket is defined. It mainly focuses on the generic fields that appear in all the tickets, assuming that every one will contain these common fields. The proposed model is based on two main categories of common fields. In what follows, a more detailed explanation about these categories is presented:

- **Timestamps fields:** This category contains the most important time indexes that are mainly related to the lifecycle of the tickets.

  - *Ticket creation time* for ticket i (t_i^{CT}): It is the instant at which the ticket creator generates the ticket in the ITS.

  - *Ticket resolution time* for ticket i (t_i^{RT}): This time is recorded on the ticket when the solution for the ticket is notified by the ticket resolver to the ticket validator.

  - *Ticket validation time* for ticket i (t_i^{VT}): It represents the instant at which the ticket validator acknowledges the solution given to the incident, and optionally, he/she sends the ticket to the ticket history database for future use.

  For every ticket i, we define a list of timestamps T_i^{TS} as:

$$T_i^{TS} = \{t_i^{CT},\ t_i^{RT},\ t_i^{VT}\}$$

- **Identifiers' fields:** This category of fields in a ticket is used to univocally identify the ticket in the database, associate it to the different ticket resolvers involved in the ticket management stages, and also relate it with the network elements affected by the incident reported in the ticket.

  For each ticket i, we identify three types of field lists:

  - *Ticket IDs* (TIDs): Each ticket in the database has a unique identifier that is typically an alphanumeric value representing the unique reference of the ticket itself. We will refer to this identifier as the *main ticket ID*.

    Although there is a unique *main ticket ID* field in a ticket, during the management stage, ticket creators/resolvers normally may directly or indirectly mention the *main ticket IDs* of other tickets. This reference is usually included in other fields, especially those with free text format, such as incident description, worklog, or incident resolution fields. For this reason, we also consider a list of ticket IDs extracted from these fields and containing all these related identifiers. For every ticket i, we have the following list:

$$T_i^{TID} = \{TID_i^1, TID_i^2, ..., TID_i^n\}$$

where TID_i^1 is the *main ticket ID* for ticket i.

- *Group IDs* (GIDs): Due to the fact that every ticket can be managed by different ticket resolvers during its lifecycle, the tickets contain information about them.

  Here, a list of ticket resolvers (TR) involved in the management of a ticket is defined. Every ticket resolver j that is involved in the management of ticket i is identified by TR_i^j , resulting in the list:

  $$T_i^{TR} = \{TR_i^1, TR_i^2, ..., TR_i^m\}$$

  where TR_i^1 is the identifier for the first ticket resolver of the ticket, that is, the ticket resolver chosen by the ticket creator in the assignment action for the ticket i.

- *Object IDs* (OIDs): Each node, service, component or element in a network is usually referred to with an unique identifier, usually a string, called *the object ID*. When an incident is detected in a network, the corresponding ticket is associated with the identifier of an object of the network that is called *the main object ID*. In addition, in the ticket creation process or during the ticket management stage, the ticket creators/resolvers usually include other object IDs, mainly in the incident description, worklog, or the incident resolution fields, that are related to the incident described by the ticket. Thus, a list of object IDs for every ticket i that contains all the previously described objects' identifiers is defined as:

  $$T_i^{OID} = \{OID_i^1, OID_i^2, ..., OID_i^l\}$$

  where OID_i^1 is the *main object ID* for ticket i.

In summary, the proposed model represents a ticket by a tuple containing the following elements (note that every element is really a list of values):

$$T_i = \{T_i^{TS}, T_i^{TID}, T_i^{TR}, T_i^{OID}\} \tag{5.1}$$

It should be pointed out that tickets normally include many other fields not considered in the suggested model, *e.g.*, the priority of the ticket. Just to simplify the proposed model, we have only included those fields that will be used in advance for the correlation process algorithms proposed here. Yet, an extension of the model to include additional information is straightforward.

5.5 The Incident Tickets Correlation Process

In this section, a tickets correlation process is proposed. It is aimed to enhance the information provided by the ITS, trying to rebuild the ITS database in order to obtain, for every incident in the network, only one ticket containing accurate and aggregated information about every incident.

The proposed global procedure can be decomposed in three main phases: *tickets preprocessing*, *tickets reduction* and *tickets correlation* (Figure 5.2). Thus, starting from a set θ of O original tickets stored in the ITS, a first preprocessing step is required in order to normalize the tickets and extract the relevant features, as it will be explained next. Here, the preprocessing phase is also divided into two subphases: *normalization* and *parameterization*. After that, the processed tickets are entered into the tickets reduction phase to filter irrelevant tickets that are not related to real network incidents. The resulting set, ρ, composed of P tickets, is then passed through the tickets correlation procedures to produce a new set, δ, composed of G supposedly de-correlated tickets (that is, the final set of tickets). The set δ is expected to contain a single ticket per incident. As shown in Figure 5.2, the proposed correlation procedure can be further split into two subphases: *local correlation* and *global correlation*, each of them targeted at removing the redundant tickets according to different properties. In the following subsections, a more detailed discussion about each step is provided.

5.5.1 Tickets Preprocessing

The first step for dealing with the original ITS database, θ, is to apply a preprocessing mechanism aimed to obtain, in a proper format, the tickets that are relevant from the point of view of incident solving. The model that was previously proposed and defined in Section 5.4 –Equation (5.1)– is initially extracted for every ticket. To do this, the preprocessing phase is divided into two subphases:

Normalization

Here, tickets are first converted into a common format that is acceptable by the other phases. Tickets that do not pass the normalization step are filtered out. Therefore, in this step *malformed* and *void* tickets are filtered. Recall from Chapter 3 that malformed tickets are those erroneously created either due to a failure in the network management tool, that automatically could generate such kind of tickets, or due to

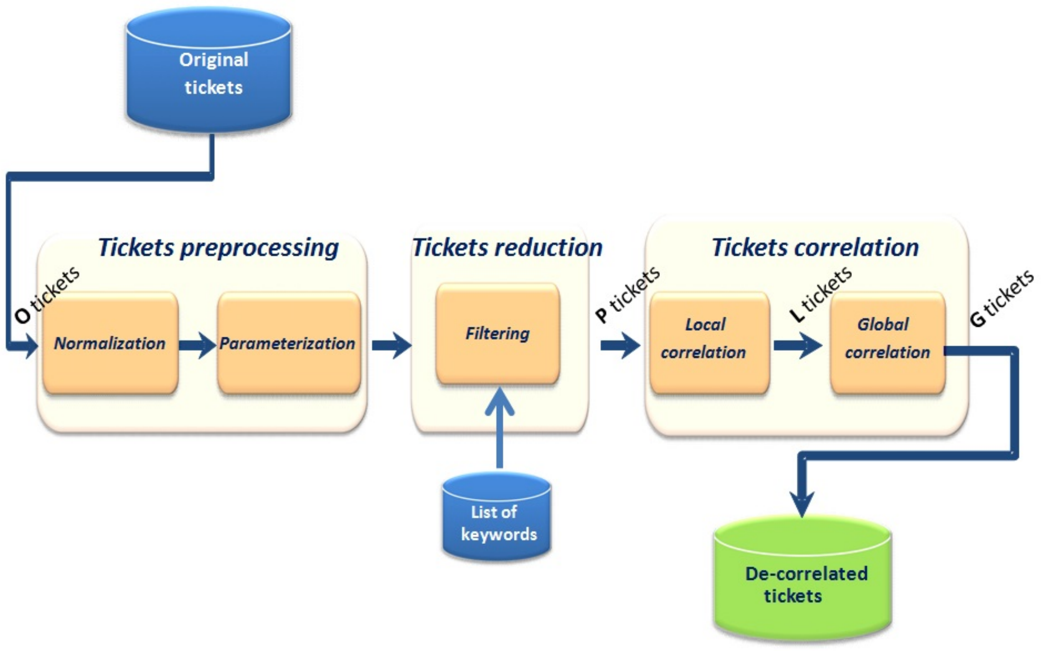


Figure 5.2: The proposed tickets correlation process.

a misbehavior of any member of the management staff, so that they could create tickets without complete or accurate information. Void tickets are those tickets that have a number of required fields filled with empty data, mainly those describing the affected nodes, services, or the resolution time. Malformed and void tickets are erroneously created and they should be removed from the whole procedure.

Parameterization

In this step, the list of parameters for the proposed ticket model is extracted, as illustrated in Equation (5.1). To do this, for every ticket we extract these relevant fields: *main object ID* (OID_i^1), *main ticket ID* (TID_i^1), *first ticket resolver ID* (TR_i^1), and the list of timestamps (T_i^{TS}). These are normally mapped in a ticket field, so they are directly extracted from their corresponding fields in the tickets. The other identifiers in the model, *e.g.*, TID_i^2, OID_i^2, or TR_i^2, are extracted by using pattern matching methods to search for the occurrence of keywords and text conformation to given formats in other ticket fields; normally in incident description, worklogs, and solution fields.

5.5.2 Tickets Reduction

In this phase, we select only network incident-related tickets. Note that there are many tickets in the ITS that have not really been created because of an incident occurrence. As explained in Chapter 3, there exist *irrelevant* tickets, *i.e.*, those that are not related to real network incidents. In many ITSs, it is usual that some tickets are created to record issues not directly related to network incidents but to contain information about some other issues as, for example, the availability of a new software release in certain network nodes or maintenance activities. These irrelevant tickets are not directly related to network incidents and, therefore, should be also removed to carry out the correlation. To identify these tickets, a list of keywords provided by the management staff is used and any matching procedure could be followed for this purpose.

After applying the above three steps, the output of this phase is a set ρ of P tickets, $\{T_i\}$, in the standard form given in Equation (5.1).

$$\rho = \{T_1, T_2, ..., T_P\}$$

5.5.3 Tickets Correlation

The tickets' correlation phase has two main goals. First, it is used to reduce the total number of tickets by substituting every subset of tickets that share some common properties into a single one that is termed the *representative ticket* for that subset. Second, the correlation process is expected to increase the semantic information of the data on the tickets. This means that a representative ticket will contain more accurate information about the real incident than the bunch of tickets taken as input to this phase. This is mainly because the information provided by a representative ticket summarizes all the information in the tickets related to the same incident.

The tickets correlation phase is divided into two subphases: *local* and *global* correlations. In the following a more detailed description of each one of them is presented.

Local Correlation

Local correlation is used to correlate tickets that collect information about different problems reported for the same network node or service that really correspond to the same incident. The main hypothesis here is that if several tickets have the same *main object ID* and they overlap in time, those tickets are related to the same incident, and thus, should be correlated. We call this process a local correlation process because it works on a per-node level and ignores any tickets potentially related to the same incident but reported for different *main object IDs*. These representative tickets will be treated as the input to the next subphase, global correlation.

For the local correlation process, we propose an algorithm that takes several overlapped tickets having the same *main object ID* and reduce them to a single ticket, T'_R, termed *local representative ticket*. The algorithm is explained in what follows.

Suppose that, from the complete tickets database, we identify the subsets $S_k \subset \rho$, each composed of a number N_k of incident tickets $S_k = \{T_1, T_2, ..., T_{N_k}\}$, so that $\cup_{\forall k} S_k = \rho$ and $S_i \bigcap S_j = \emptyset, \forall i \neq j$. Each subset S_k is composed of several tickets in ρ that share the following properties:

- They have the same *main object ID, i.e.,*

$$OID_i^1 = OID_j^1, \forall i, j \in [1, N_k]$$

- They overlap in time; that is, for every ticket $T_i \in S_k$, there is at least another ticket $T_j \in S_k$ so that their intervals $[t_i^{CT}, t_i^{RT}]$ and $[t_j^{CT}, t_j^{RT}]$ overlap, *i.e.,*

$$\left\{\{t_i^{CT} \leq t_j^{CT} \leq t_i^{RT}\} \vee \{t_j^{CT} \leq t_i^{CT} \leq t_j^{RT}\}\right\} = true$$

- There are no time gaps between them, *i.e.,*

$$\forall t \in \left\{min_{i \in [1,N_k]}\{t_i^{CT}\}, max_{i \in [1,N_k]}\{t_i^{RT}\}\right\}, \text{there exists at least one active ticket.}$$

  where an active ticket at time t is defined as a ticket that has a creation time previous than t, and a resolution time after than t.

For every subset S_k, the correlation algorithm creates a *local representative ticket*, T_k', that will replace all the tickets in S_k, following these rules:

$$t_k'^{CT} = min_{i \in [1,N_k]}\{t_i^{CT}\} \tag{5.2}$$

$$t_k'^{RT} = max_{i \in [1,N_k]}\{t_i^{RT}\} \tag{5.3}$$

$$t_k'^{VT} = max_{i \in [1,N_k]}\{t_i^{VT}\} \tag{5.4}$$

$$T_k'^{x} = \bigcup_{i=1}^{N_k} T_i^x, where\ x = \{TID, TR, OID\} \tag{5.5}$$

$$OID_k'^1 = OID_i^1, \forall i \tag{5.6}$$

Note that the value for the fields $T_k'^{TID}$, $T_k'^{TR}$, and $T_k'^{OID}$ is extracted as the union (concatenation without repetition) of the corresponding lists in the tickets from the subset S_k. This means that, in the *local representative ticket*, the meaning of $TID_i'^1$ and $TR_i'^1$ as the *main ticket and ticket resolver IDs* is lost. However, the value $OID_i'^1$ still represents the *main object ID*, as this value is the same for every ticket in S_k.

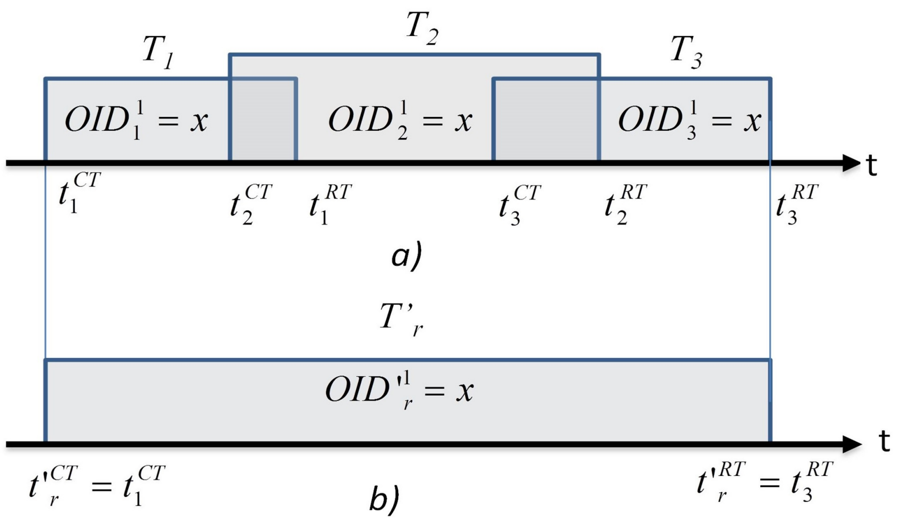


Figure 5.3: Local correlation example: a) original tickets, b) local representative ticket.

It must also be noted that it could happen that a subset S_k is composed by a single ticket, in case that this ticket is not overlapped with any other in ρ for the same *main object ID*. In such a case, this ticket is itself the representative ticket for that subset.

The output of this local correlation process is a set ξ of L local representative tickets, each one representing a different subset $S_k \subset \rho$.

$$\xi = \{T'_1, T'_2, ..., T'_L\}$$

The operation of the algorithm can be explained through a simplified example as shown in Figure 5.3. In this example we consider one subset, S_k composed of three tickets ($N_k = 3$). As depicted in Figure 5.3a), the creation time of the ticket T_2 (t_2^{CT}) occurs during the interval $[t_1^{CT}, t_1^{RT}]$; besides, T_3 is created (t_3^{CT}) within the interval $[t_2^{CT}, t_2^{RT}]$; finally, these three tickets have the same *main object ID* (node x), *i.e.*, $OID_1^1 = OID_2^1 = OID_3^1 = x$. Thus, T_1, T_2, and T_3 , according to the proposed method, should be locally correlated into the representative ticket T'_r, as shown in Figure 5.3b, where $t'^{CT}_r = min_{i\in[1,3]}\{t_i^{CT}\} = t_1^{CT}$ and $t'^{RT}_r = max_{i\in[1,3]}\{t_i^{RT}\} = t_3^{RT}$. Besides, T'^{TID}_r, T'^{TR}_r, and T'^{OID}_r would be extracted as the union of the corresponding lists in T_1, T_2, and T_3.

Global Correlation

After having correlated the overlapping tickets related to the same *main object ID* (local correlation), here the proposed correlation process is extended to consider other tickets with different *main object IDs*, leading to a global correlation. The aim now is to identify tickets reporting problems in different nodes or services but corresponding to the same incident.

In the ticket creation process or along the resolution of the incident, ticket creators/resolvers usually include information in a ticket about other tickets (TIDs) or other objects in the network (OIDs) that are related to the incident described by the ticket. This information appears, for a ticket T_i, in the lists T_i^{TID} and T_i^{OID}.

In a first approach, we could say that two tickets T_i and T_j are related to the same incident if they refer to a common OID or TID in any of the considered lists, that is, if at least one of the following conditions is fulfilled

$$Condition1 : \exists TID \in \{T_i^{TID} \bigcap T_j^{TID}\}$$
$$Condition2 : \exists OID \in \{T_i^{OID} \bigcap T_j^{OID}\}$$

Note that although we impose this restriction, the simple existence of a common TID/OID value in two tickets does not necessarily imply a relationship between them. Indeed, it is common that a ticket includes a reference to other TID/OID where a workaround for solving the incident can be found. For this reason, an additional constraint for relating two tickets to the same incident is imposed here, that is temporal overlapping.

Finally, we make a final assumption. We assume that the transitive property is valid in the relationships among tickets, *i.e.*, if tickets T_i and T_j are related, and tickets T_j and T_k are also related, then tickets T_i, T_j, and T_k are all related.

The global correlation has the same target as the local correlation, that is, to obtain only one ticket per incident; however, as a difference in this case, it will be referred to different main object IDs. Therefore, it is considered as an almost identical process as the local correlation, in which every subset of related tickets in ξ is substituted by a single *global representative ticket*, denoted as $\overline{T}$.The complete algorithm is described in what follows.

Suppose that, from the set ξ, we identify different subsets $C_k \subset \xi$, such that $\bigcup_{\forall k} C_k = \xi$ and $C_i \bigcap C_j = \emptyset, \forall i \neq j$. Each subset C_k is composed by a number N_k' of local representative tickets $C_k = \{T_1', T_2', ..., T_{N_k}'\}$, namely all the tickets contained in C_k share the following properties:

- Every local representative ticket $T_i' \subset C_k$ is related to at least one local representative ticket $T_j' \subset C_k$, *i.e.*,

$$\left\{\{T_i'^{TID} \bigcap T_j'^{TID}\} \bigcup \{T_i'^{OID} \bigcap T_j'^{OID}\}\right\} \neq \emptyset, \forall i, j \in [1, N_k]$$

- They are overlapped in time, that is, for every ticket $T'_i \in C_k$, there is at least another ticket $T'_j \in C_k$ so that their intervals $[t'^{CT}_i, t'^{RT}_i]$ and $[t'^{CT}_j, t'^{RT}_j]$ are overlapped, *i.e.*,

$$\left\{\{t'^{CT}_i \leq t'^{CT}_j \leq t'^{RT}_i\} \vee \{t'^{CT}_j \leq t'^{CT}_i \leq t'^{RT}_j\}\right\} = true$$

- There are no time gaps between them, *i.e.*,

$\forall t \in \left\{min_{i\in[1,N_k]}\{t'^{CT}_i\}, max_{i\in[1,N_k]}\{t'^{RT}_i\}\right\}$, there exists at least one active ticket.

As in the local correlation process, the global correlation process creates a *global representative ticket*, $\overline{T}_k$, for the subset of tickets C_k, which will replace all the tickets in C_k, according to the following rules:

$$\bar{t}^{CT}_k = min_{i\in[1,N_k]}\{t'^{CT}_i\} \tag{5.7}$$

$$\bar{t}^{RT}_k = max_{i\in[1,N_k]}\{t'^{RT}_i\} \tag{5.8}$$

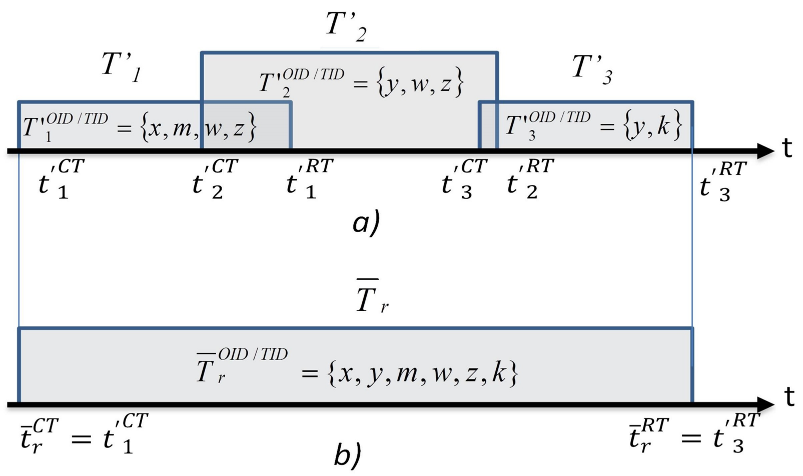
$$\bar{t}^{VT}_k = max_{i\in[1,N_k]}\{t'^{VT}_i\} \tag{5.9}$$

$$\overline{T}^x_k = \bigcup_{i=1}^{N_k} T'^x_i, where\ x = \{TID, TR, OID\} \tag{5.10}$$

It is important to mention here that the values for the fields $\overline{T}^{TID}_k$, $\overline{T}^{TR}_k$, and $\overline{T}^{OID}_k$ are extracted as the union of the corresponding fields in the tickets from the subset C_k. After this process, the meaning of OID'^1_i, TID'^1_i, and TR'^1_i as the main object, ticket and ticket resolver IDs are lost.

As in the case of local correlation, Figure 5.4 shows an example of the application of the algorithm using three tickets, $N'_k = 3$. As depicted in Figure 5.4a), $T'^{OID}_1 = \{x, m, w, z\}$, $T'^{OID}_2 = \{y, w, z\}$, and finally $T'^{OID}_3 = \{y, k\}$. T'_1 and T'_2 are related since they have two OIDs in common, namely w and z, and they temporally overlap. Furthermore, T_2 and T_3 are also related since they have one OID in common, namely y, and they are temporally overlapped. Therefore, the three *local representative tickets* are correlated into a *global representative ticket* $\overline{T}_r$ as illustrated in Figure 5.4b) having $\bar{t}^{CT}_r = min_{i\in[1,3]}\{t'^C_i T\} = t'^{CT}_1$, $\bar{t}^{RT}_r = max_{i\in[1,3]}\{t'^R_i T\} = t'^{RT}_3$ and finally $\overline{T}_r^{(OID/TID)} = \{x, y, m, w, z, k\}$, where $\overline{T}_r^{(OID/TID)}$ is the list of OIDs and TIDs extracted as the union (concatenation without repetition) of the corresponding lists in the correlated tickets.

The output of this global correlation is a set δ of G tickets, that is composed by the list of representative tickets, each one for every different subset $C_k \subset \xi$.



T'_2
T'_1
T'_3
$T'^{OID/TID}_2 = \{y, w, z\}$
$T'^{OID/TID}_1 = \{x, m, w, z\}$
$T'^{OID/TID}_3 = \{y, k\}$
t
t'^{CT}_1
t'^{CT}_2
t'^{RT}_1
t'^{CT}_3
t'^{RT}_2
t'^{RT}_3
a)
$\overline{T}_r$
$\overline{T}^{OID/TID}_r = \{x, y, m, w, z, k\}$
t
$\bar{t}^{CT}_r = t'^{CT}_1$
b)
$\bar{t}^{RT}_r = t'^{RT}_3$

Figure 5.4: Global correlation example: a) original tickets, b) global representative ticket.

$$\delta = \{\overline{T}_1, \overline{T}_2, ..., \overline{T}_G\}$$

Ideally, if all the incidents have been reported in the ITS, every one of these representative tickets is expected to represent a single incident in the network, *i.e.*, $G = I$, where I is the total number of incidents.

5.6 Experimental Results

For the experimental results to validate the model proposed here, we have used the datasets described in Chapter 3, DS1 and DS2. As a first step in the experimental evaluation, an alignment of the fields of tickets in both datasets is done with the proposed ticket model fields. This correspondence is presented in Table 5.1. In these datasets, the alignment has been done easily since each selected field clearly matches with a field in the proposed ticket model. This fact emphasizes the generality and simplicity of the proposed model that consists of the most common fields that are found in any ITS.

5.6.1 Performance Indicators

Before describing the experimental evaluation, we also suggest some performance metrics that help to evaluate the efficiency of the proposed correlation algorithms.

The process of normalizing the metrics is not straightforward, as there are two different types of tickets defined here, *i.e.*, those extracted from the original database, and the representative tickets that are produced by the correlation process using the proposed correlation algorithms. For this reason, the estimation of the metrics will depend on each phase, where there are different input entries, *i.e.*, malformed, void

Company ITS field	**Incident ticket model field**
TICKET CREATION TIME	t_i^{CT}
TICKET RESOLUTION TIME	t_i^{RT}
TICKET VALIDATION TIME	t_i^{VT}
SEDE	OID_i^1
CASE_ID	TID_i^1
ASSIGNED_TO_GROUP	TR_i^1

Table 5.1: Alignment of the considered ITS fields with the proposed incident ticket model fields.

and irrelevant tickets should be extracted from the original database during the preprocessing and reduction phases, locally correlated tickets should be extracted from the filtered database and replaced by their representatives; and finally, globally correlated tickets should also be extracted and replaced from the locally correlated database.

The global process is depicted in Figure 5.5. Recall from our previous definition that O is the number of records in the original database θ; P is the number of records in the filtered database, ρ, *i.e.*, after having removed malformed, void and irrelevant tickets; L is the number of records in the locally correlated database, ξ, *i.e.*, after having substituted each subset of locally correlated tickets by a local representative ticket; and finally, G is the number of records in the globally correlated database, δ, *i.e.*, after having substituted each subset of globally correlated tickets for a global representative ticket. In every step, part of the tickets is removed from the input, as they are supposed to be useless for incident solving (preprocessing and reduction)[1] or redundant (local and global correlation). In the latter case, although the original tickets are removed, a new representative ticket summarizing each subset of correlated tickets is introduced. Therefore, in order to assess each of the involved steps, a measure concerning the percentage of tickets that are removed in each phase has been defined.

- The tickets reduction percentage for the preprocessing and reduction phases, φ_P, is defined as:

$$\varphi_P = (\frac{O-P}{O}) \cdot 100 \tag{5.11}$$

[1]For simplicity, from now on, the word preprocessing will be used to refer to both phases, preprocessing and reduction.

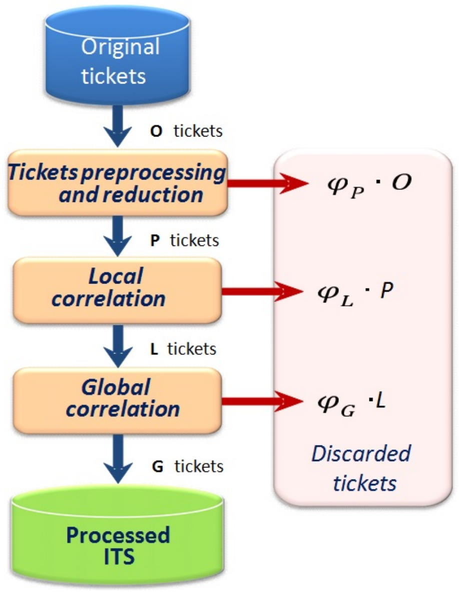


Figure 5.5: Evolution of the number of tickets along the proposed system.

- Similarly, the tickets reduction percentage for the local correlation phase, φ_L, is defined as:

$$\varphi_L = (\frac{P-L}{P}) \cdot 100 \tag{5.12}$$

- Finally, the tickets reduction percentage for the global correlation phase, φ_G, is defined as:

$$\varphi_G = (\frac{L-G}{L}) \cdot 100 \tag{5.13}$$

After doing all of the above steps, the main hypothesis is that the remaining number of tickets in the processed database, G, approximately equals the number of incidents, I. Thus, $I \cong G$. Therefore, the overall tickets' reduction percentage, $\varphi_{overall}$, is defined as:

$$\varphi_{overall} = (\frac{O-G}{O}) \cdot 100 \tag{5.14}$$

It should be noted that:

$$1 - \varphi_{overall} = (1-\varphi_P)(1-\varphi_L)(1-\varphi_G) \tag{5.15}$$

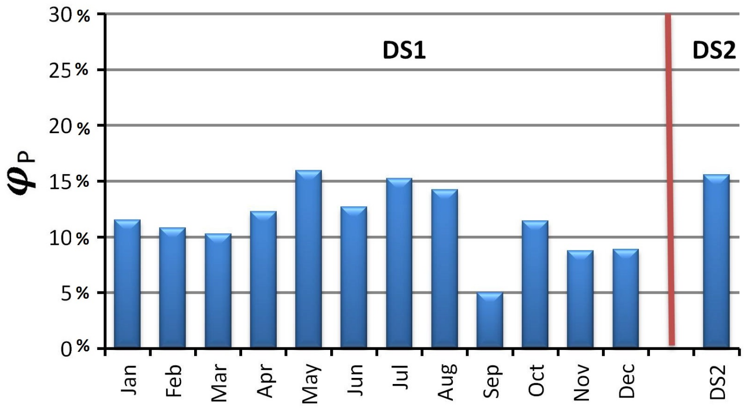


Figure 5.6: Reduction percentage of irrelevant tickets extracted from both datasets.

5.6.2 Evaluating the Tickets Preprocessing Phase

The above parameters are used as performance indicators to measure the reduction percentages and the overall efficiency of the tickets' creation process in both datasets. As indicated above, DS1 is kept split into months and the results for each one of them are shown separately, besides the results obtained from DS2.

In our case, after many consultations with the management staff responsible of the considered ITS datasets, the criterion to label a ticket as malformed or void is to identify those which have one or more of these fields with an empty value: *main Object ID*, *resolution time* or *incident description*. Furthermore, we identified the tickets that contain any word from a specific list of keywords built with the help of the management staff as irrelevant. The list of keywords used is {*swap/change terminal*, *change battery*, *update software*, *scheduled maintenance*, *change technology*}.

After applying the preprocessing methodology presented in Subsection 5.5.1 to extract *incident-related tickets*, Figure 5.6 illustrates the reduction percentage of irrelevant tickets (φ_P) for each month of DS1 and DS2. On average, 11.4% (DS1) and 15.6% (DS2), respectively, of the original tickets are considered irrelevant from the point of view of incident solving. The final number of *incident-related tickets* after completing the preprocessing phase is P=17,008 (DS1) and 8,105 (DS2), respectively. Although many of the irrelevant tickets are really created by the management team, either by management staff or SD, it is clear from the DS1 results that the analyzed ITS had several malfunctionings, especially during May, July and August, due to the high rate of such kind of tickets. So, we conclude that the behavior of the management team regarding the creation of irrelevant tickets is not stable during the whole year, being specially affected during holiday periods. This methodology would point to a revision of the configuration of the platform in order to improve its performance.

Also, we conclude that, in the case study, both the ITS and the management team always generate a large number of tickets that are not related to real network incidents. These types of tickets may have negative effects on the performance of the ITS as a whole, because creating such kind of tickets consume time and labor work, which are two important factors that affect the revenue and QoS of the IT department of an enterprise or an IT service provider.

5.6.3 Evaluating the Tickets Correlation Phase

Here, the local and global correlation processes are evaluated separately in order to analyze the effect of each of them on the whole system.

A. Evaluating the Local Correlation Algorithm

In order to extract the locally correlated tickets, the correlation algorithm that was discussed in Subsection 5.5.3 is applied to the filtered databases. Recall that, the main assumption here is that two tickets are locally correlated if they share the same *main object ID* and they temporally overlap. The evaluation phase is divided in two subsections. First, we analyze the datasets and extract the results. Second, we present two methods to validate the correlation results.

A.1. Results for the Local Correlation

The reduction percentage of locally correlated tickets (φ_L), extracted for each month in DS1 and for the whole period in DS2, is shown in Figure 5.7. Furthermore, Table 5.2 gives an overview of some useful statistics for each month of DS1 and also for DS2: the number of processed tickets, P; the number of obtained local representative tickets, L; the reduction percentage, φ_L; and some information related to the locally correlated subsets, *i.e.*, the number of tickets that have been correlated, the number of subsets of correlated tickets (representative subsets) and the size of the subsets (mean and standard deviation, SD). From these results we found that the proposed local correlation algorithm is able to discover, on average, more than 14.4% (DS1) and 10.7% (DS2) of the created tickets as redundant tickets. Furthermore, we observed from the DS1 results that the behavior of the management team regarding locally correlated tickets during the whole year is somehow stable with few variations in the second half of the year, where about 21% of the tickets are considered redundant.

A.2. Validation Process

As the investigated databases are unsupervised, *i.e.*, they do not have tags indicating the incident associated to every ticket, there is an inherent difficulty in the validation process itself. Notwithstanding with this, here we try to validate the usefulness

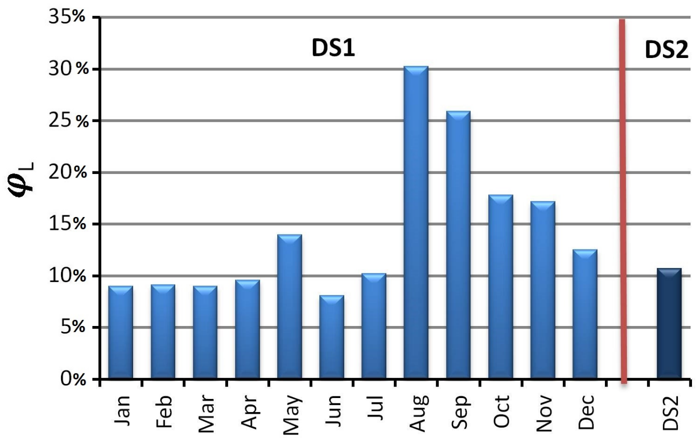


Figure 5.7: Reduction percentage of locally correlated tickets extracted from both datasets.

dataset	period	# input tickets (P)	# output tickets (L)	φ_L	locally correlated subsets			
					# correlated tickets	# representative subsets	# tickets/subset	
							Mean	SD
DS1	Jan	1,756	1,598	8.99%	270	112	2.41	0.83
	Feb	1,784	1,621	9.14%	275	112	2.46	1.30
	Mar	2,008	1,827	9.01%	308	127	2.43	1.50
	Apr	1,524	1,378	9.58%	241	95	2.54	1.76
	May	622	535	13.98%	136	49	2.78	1.95
	Jun	1,808	1,662	8.08%	248	102	2.43	0.64
	Jul	1,273	1,143	10.2%	223	93	2.39	1.07
	Aug	602	420	30.2%	240	58	4.14	6.19
	Sep	1,045	774	25.9%	392	121	3.24	3.38
	Oct	1,764	1,450	17.8%	545	231	2.36	1.31
	Nov	1,491	1,235	17.2%	443	187	2.37	2.08
	Dec	1,331	1,164	12.5%	308	141	2.18	0.82
DS2	6M	8,105	7,240	10.7%	1,569	704	2.23	0.64

Table 5.2: Local correlation results for both datasets.

of the proposed local correlation algorithm using two different validation methods. First, the results are reviewed under the hypothesis that, in any ITS, if several tickets have the same *main object ID* (node, service, location, etc.), they overlap in time, and they are resolved at almost the same time, we can assume that they really belong to the same incident. Hence, a study of the distribution of the delay for resolution times for each subset of locally correlated tickets could help to validate the proposed correlation process.

This analysis is applied to the two datasets; for three randomly selected months of DS1 (Jan., June, Dec.) and for DS2. Figure 5.8 illustrates the distribution of the mean delay between the local representative ticket resolution time and the mean value of the resolution times for all the tickets in the subset, that is:

$$t_k'^{RT} - \frac{1}{N_k}\sum_{i=1}^{N_k} t_i^{RT}, \quad \forall T_i \subset S_k$$

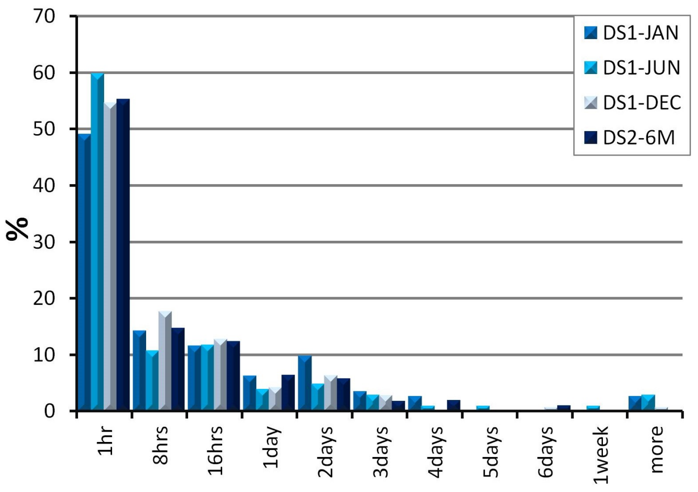


Figure 5.8: Distribution of the mean delay among resolution times of tickets belonging to locally correlated groups, for subsets from both datasets.

This is an indicator of the dispersion of the resolution times for the correlated tickets. From this figure, we observe that the delay distribution of both datasets, the three selected months in DS1 and the whole period in DS2, is very similar. The graph is decreasing nearly exponentially with an average of more than 54.5% (DS1) of the subsets having all the related tickets resolved within one hour between them; more than 68.8% are resolved within the first 8 hours and so on. For DS2, 58% of the subsets having all the related tickets resolved within one hour between them, and about 72% are resolved within the first eight hours. We found that more than 80% of the subsets of correlated tickets (81.3%, 86.3%, and 89.4%, respectively for DS1 and 88.9% for DS2) contain tickets with differences in the resolution times lower than one day. This analysis applied on two different datasets (generated by staff following different management procedures) is a good indicator of the correct behavior of the proposed correlation algorithm.

In an additional validation, besides the delay analysis described above, we randomly selected 100 samples from the locally correlated subsets; 20 from each of the three selected months of DS1, and 40 from DS2. Here we divided the analysis into two groups, those subsets with a delay lower than or equal to one day and those subsets with a delay greater than one day. Half of the samples are taken from the first group, while the other half of the samples are taken from the second one. Figure 5.9 represents a histogram of the number of correlated tickets per subset for the selected samples. For each sample subset, we manually inspected all the tickets and checked whether they can be considered as a TP or FP. In order to do that, we looked at the fields in the tickets that contain text-free information, especially those that characterize any incident such as incident description, worklog history, and solution description, and checked whether they share the same incident symptoms or

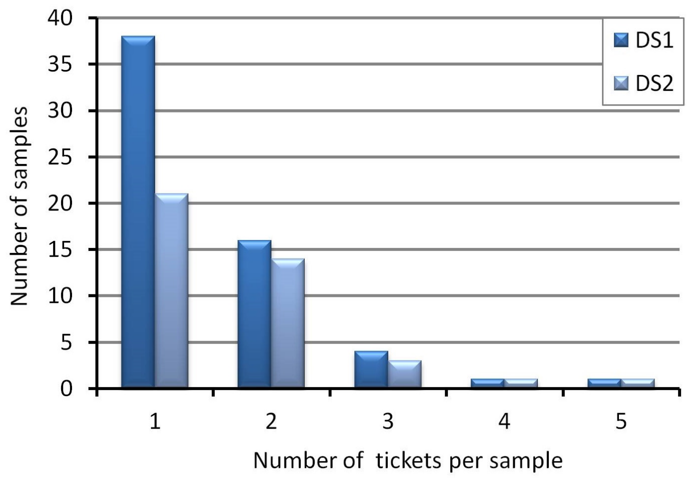


Figure 5.9: Histogram of the number of tickets per subset for the selected samples from both datasets.

not. The correlation results showed that all of the samples taken from both datasets and belonging to both groups (delay less than or equal to one day and delay greater than one day) are validated as TP. Therefore, we argue that the accuracy of the local correlation algorithm is high, not finding FPs in the sampling process considering samples from both groups. Thus, we conclude that the proposed local correlation algorithm can correlate any subset of locally correlated tickets into a local representative one that correctly describes the incident with a high confidence (efficiency is 100% for the sampled subset). This finding emphasizes our claim that in any ITS, if several tickets have the same *main Object ID* and they temporally overlap, there is a high probability that they really belong to the same incident and, therefore, should be correlated.

B. Evaluating the Global Correlation Algorithm

The global correlation algorithm discussed in Section 5.5.3 is applied to the locally correlated datasets, ξ, *i.e.*, after every locally correlated subset is replaced by a local representative ticket. As previously explained, DS1 is kept split into months and any border issues that could happen between two consecutive months are ignored. As in the local correlation phase, the evaluation process is divided into two subsections. First, we provide general results and extract several findings. Second, the same methods as in the previous subsection are used to validate the results.

B.1. Results for the Global Correlation

Figure 5.10 shows the reduction percentage of globally correlated tickets (φ_G) that are extracted from the locally correlated dataset ξ. Furthermore, Table 5.3 gives

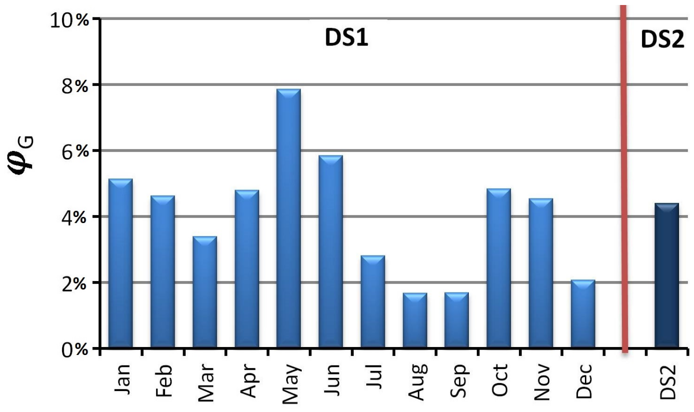


Figure 5.10: Reduction percentage of globally correlated tickets extracted from both datasets.

dataset	period	# input local repr. tickets (L)	# output tickets (G)	φ_G	globally correlated subsets			
					# correlated tickets	# representative subsets	# tickets/subset	
							Mean	SD
DS1	Jan	1,598	1,516	5.13%	128	46	2.78	1.62
	Feb	1,621	1,546	4.63%	118	43	2.74	1.28
	Mar	1,827	1,765	3.39%	99	37	2.68	1.65
	Apr	1,378	1,312	4.79%	98	32	3.06	3.31
	May	535	493	7.85%	46	4	11.5	19.0
	Jun	1,662	1,565	5.84%	157	60	2.62	1.82
	Jul	1,143	1,111	2.79%	59	27	2.19	0.42
	Aug	420	413	1.67%	14	7	2.0	0.49
	Sep	774	761	1.67%	23	10	2.30	0.67
	Oct	1,450	1,380	4.83%	107	37	2.89	2.14
	Nov	1,235	1,179	4.53%	82	26	3.15	4.75
	Dec	1,164	1,140	2.06%	39	15	2.60	1.12
DS2	6M	7,240	6,922	4.39%	516	198	2.6	2.42

Table 5.3: Global correlation results for both datasets.

an overview of the same statistics that are discussed in Table 5.2, for each month of DS1 and also for DS2. But, here, the input is the number of local representative tickets, L; the output is the number of global representative tickets, G; and the reduction percentage is, φ_G. From these results we can extract some conclusions. First, as a general finding, we observe that there is another level of redundancy in which the proposed global correlation algorithm is able to discover, on average, about 4.1% (DS1) and 4.4% (DS2) of the local representative tickets as redundant tickets. Second, through a manual inspection of some samples of globally correlated tickets for both datasets, we observe that the management staff usually uses either TID or OID fields to relate tickets to each other and sometimes they use both in the same ticket.

The final number of tickets in the globally correlated database, G, is 14,181 (DS1) and 6,922 (DS2), which is also supposed to be the number of incidents, I.

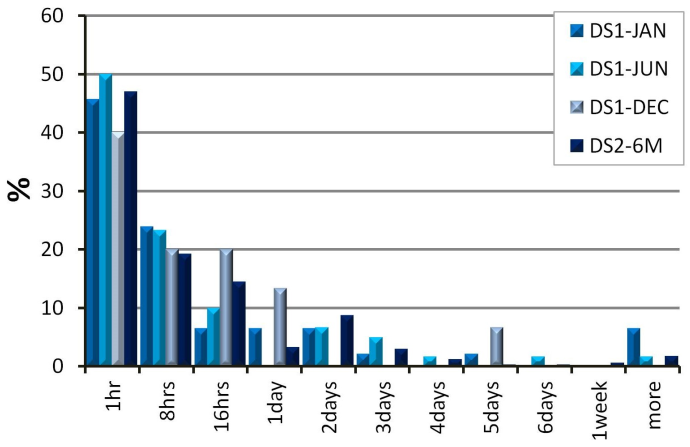


Figure 5.11: Distribution of the mean delay among resolution times of representative tickets belonging to globally correlated groups, for subsets from both datasets.

B.2. Validation Process

Here, the same validation methods as in the local correlation are used to validate the results from the global correlation algorithm. Figure 5.11 illustrates the delay results. From this figure we observe that for DS1, on average, more than 86.5% of the subsets have tickets with differences in resolution times lower than one day and for DS2, more than 88.1%. Again, this finding is useful, since we have two datasets with different properties and that have coherent results of the global correlation. This points to the validity of the proposed global correlation algorithm.

Second, for the sampling validation process, as in the local correlation process, the analysis was divided into two groups, those correlated subsets with delay lower than or equal to one day and those with delay greater than one day. Again, half of the samples are taken from the first group, while the other half of the samples are taken from the second one. We show the validation results for 55 samples from DS1, which are also randomly selected except for December, since we have only 15 subsets in this month (we selected all of them), and 40 samples are taken from DS2 with a total of 95 samples. The histogram in Figure 5.12 shows, for both datasets, the number of samples vs. the number of local representative tickets that they contain. The correlation results show that all of the samples taken from both datasets and belonging to the first group are validated as TP, whereas five samples from the second one –three from DS1 and two from DS2– are validated as FPs.

Regarding the five samples validated as FPs, we found out that the management staff sometimes refers to previous solved tickets if the ongoing incident has mainly the same preliminary symptoms. Additionally, sometimes ticket creators relate a ticket with others just by writing TID or OID in some fields and do not describe the

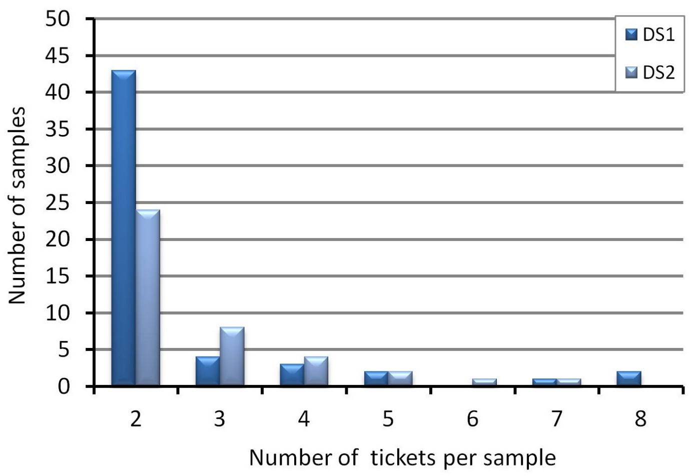


Figure 5.12: Histogram of the number of locally representative tickets per subset for the selected samples from both datasets.

incident very well which makes it difficult to decide whether they are really related or not. Thus, these not clearly correlated tickets are counted as FPs.

As a conclusion, the global correlation algorithm efficiency is really high (100% of the sampled subset) in the first group and 90% (DS1) / 95% (DS2) in the second one. On average, the global correlation algorithm efficiency for both datasets is around 95%.

5.6.4 Evaluating the Convergence of the Algorithms

We are also interested in evaluating the convergence of the algorithms when the provided datasets are split and the process is applied separately on the different subsets. This analysis is relevant, since the correlation process could be parallelized. Figure 5.13 illustrates the strategy that has been followed to evaluate the stability. Here, DS2 was split into two parts: those tickets created by the management staff and those created by the SDs; SD1 and SD2. The local correlation algorithm is applied on each part separately (left part in Figure 5.13), and the results are obtained and analyzed. Next, the local correlation algorithm is again applied to the already correlated tickets of both parts and compared the results with those extracted from the whole dataset (right part in Figure 5.13). The same was done for the global correlation algorithm as well. Table 5.4 shows the local/global correlation results of each part of the experiment. The number of local representative tickets of both datasets (L1+L2) is 7,398, while the number of representative tickets for the output of the local correlation algorithm applied on L1+L2 is 7,240, which is exactly the same result obtained in the previous experiments in Section 5.6.3 for the whole dataset, DS2. Regarding the global correlation, the number of input tickets of both

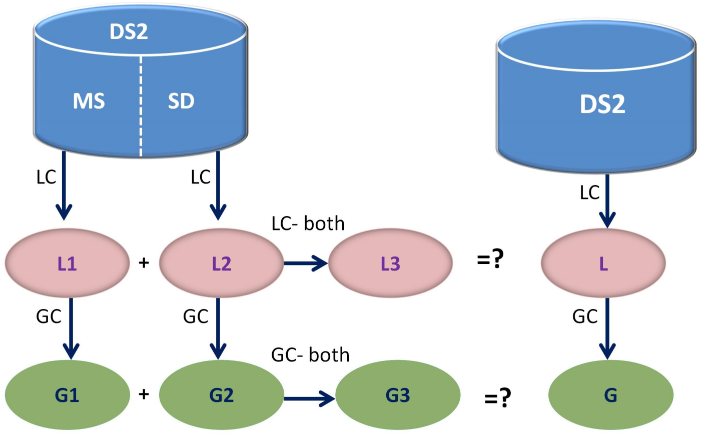


Figure 5.13: Algorithms stability evaluation chart.

correlation process	dataset	# of input tickets	# of output tickets	reduction percentage φ	correlated subsets			
					# correlated tickets	# representative subsets	# tickets/subset	
							Mean	SD
LC	DS2-MS	4,929	4,456	$\varphi_L = 9.6\%$	907	434	2.1	0.37
	DS2-SD	3,176	2,942	$\varphi_L = 7.36\%$	402	168	2.4	0.95
LC-both	L1+L2	7,398	7,240	$\varphi_L = 2.1\%$				
GC	DS2-MS	4,456	4,310	$\varphi_G = 3.1\%$	259	113	2.29	1.07
	DS2-SD	2,942	2,825	$\varphi_G = 3.87\%$	174	57	3.05	3.8
GC-both	G1+G2	7,135	6,922	$\varphi_G = 2.9\%$				

Table 5.4: Correlation results for local and global correlation algorithms for the stability analysis.

datasets (G1+G2) is 7,135, and the output of the global correlation algorithm applied on this dataset (G3) is 6,922 global representative tickets. We can check that this is also the same result presented in Section 5.6.3. Therefore, we conclude that, as expected, the results from local and global correlation algorithms are stable and independent on whether it is applied over a partitioned dataset.

5.7 Chapter's Conclusions

Along this chapter, we proposed a novel, simple, and effective approach to correlate and merge incident tickets in an ITS. The approach is based on a generic model for the tickets that preserves and categorizes the relevant information and enables the comparison of their properties with relatively simple functions. No additional sources of information, apart from the tickets themselves and the information they contain, is required during the merging procedure. Despite its simplicity, the model has revealed to be useful to reduce the number of tickets generated and handled by ITS users, which is a desirable target in order to improve the workflow in a management company.

The experiments on two different datasets from a real company have shown that contrary to what is expected, there is a significant amount of redundant tickets being generated by the different actors in the ITS. The proposed model and method can be easily incorporated in the in-production systems, not allowing the generation of additional tickets when a related one is active, or automatically removing those detected as redundant. The first is quite straightforward for the local level of the algorithm, which can be applied in real-time at the time of the creation of the ticket. The global phase uses information that could not be available at the time of the creation of the ticket but added later and, therefore, would imply an a posteriori filtering of the tickets.

The proposed algorithms are iterative in the sense that they keep merging tickets as far as there exist at least two overlapping tickets that should be merged according to the selected criterions. This does not represent any problem either from the convergence or stability points of view, as a single final solution exists and it is always reached. This is due to the transitive nature of the merging procedure.

Chapter 6

A Model for the Joint Correlation of Tickets and Alerts

As discussed in previous chapters, it is notable that, despite the fact that many of the records in ITSs contain rich semantic information related to incidents, to the best of our knowledge, only reduced efforts have been devoted to the incorporation of this information in the alerts correlation procedure. Some research efforts such as those in [25, 30, 31, 55] have pointed to the importance of ticket information for incident resolving, claiming that ITSs can be extended with advanced functions to enhance the incident resolving process. Their main argument is that the information in tickets are related to incidents generated by events that have already been identified as network failures. Other efforts such as those in [54, 153, 159–162, 171] tried to correlate alerts with tickets to achieve specific purposes, such as studying and characterizing the nature and causes of routing changes, and the observed network instability. These efforts, despite their potential usefulness in obtaining statistical measures to study the nature of the incidents and their effects on the network stability, have not been devoted to the joint correlation of ticketing information and alerts, with the goal of reducing the number of redundant alerts and the significance of the resulting events by using the tickets relevance feedback.

In this chapter, a generic correlation model aimed to establish a relationship between alerts and tickets is first presented. This model works as a basic building block for some parts of the whole correlation system. Then, based on this model, a lightweight algorithm for the joint correlation of alerts and tickets is also proposed and implemented. Then, a comparative study between the results we obtain and those obtained by a applying basic correlation techniques that only use alert databases is carried out.

The structure of the chapter is as follows. A basic model for event correlation is presented in Section 6.1. Based on this model, a complete system for tickets-alerts correlation is proposed and discussed in Section 6.2. A thorough explanation of the proposed tickets-alerts correlation model, challenges and suggested solutions are explained in Section 6.3. The correlation model is experimentally tested and evaluated through the database of tickets and alerts from an IT management company in Section 6.4.

6.1 A Basic Model for Events Correlation

Before discussing the tickets-alerts correlation process in more detail, in this section we give a brief overview about a basic event correlation model that will be used as a building block for the correlation process. It is worth to mention that the basic correlation model proposed here is derived from the one suggested in Chapter 5 to correlate tickets. Yet, we now refer to the correlated entities generically as *events* instead of tickets, as we use this term to refer to both alerts and tickets interchangeably. Therefore, instead of proposing a new terminology to describe the correlation model and its events, we somehow adapt the same terminology proposed in that chapter.

In this basic correlation model, as previously stated, we first consider both the appearance of alerts in the NMS and the generation of tickets in the ITS system as generic events. This way, whenever an incident takes place in a monitored network, a set of different events related to that incident appear. Let us denote as I the set of m different events that appear as a consequence of an incident occurring in a network, $I = \{e_1, e_1, \cdots, e_m\}$. Every event e_i has a different duration, that spans from the instant of its creation or appearance, that we will call event creation time, $t_{e_i}^{CT}$, to the instant at which this event disappears or is resolved, which we call event resolution time, $t_{e_i}^{RT}$.

In addition, every event e_i is associated to a list of one or more affected elements of the network, referred to as the *object IDs* of that event. For example, in a "node down" alert, the identity of the node of the network that has gone down is the affected object ID of that event. An event could have several affected object IDs. For example, if a ticket is created due to the failure of several nodes in a network, all of them are really the affected object IDs for that event. In general, we will say that every event e_i will have a set of affected object IDs, E_i, that is a list of the different *identifiers* of the network object IDs, applications, services, *etc.*, affected by the incident described in that event. An identifier here could be an IP address, a node name, an application name, etc.

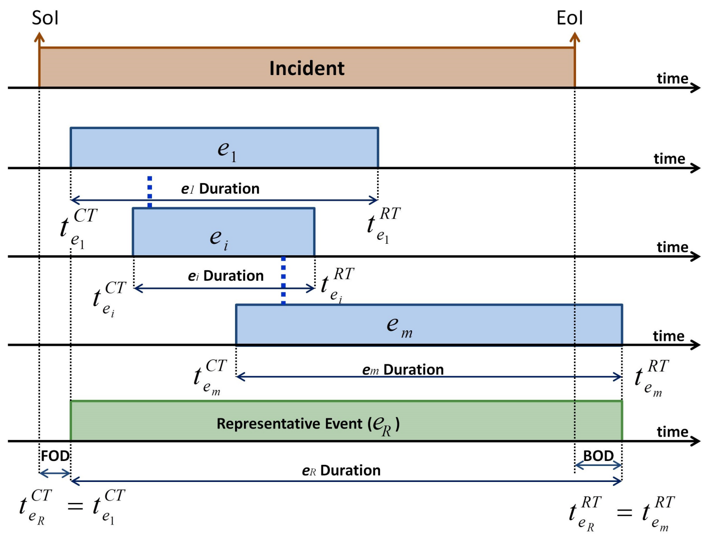


Figure 6.1: Correlation of m events belonging to the same incident into a representative event.

Furthermore, every event e_i will be also specified by an event description, D_i, that is usually a free text field describing the own event, its effects on the network, and/or the root cause for its appearance.

In this scenario, we are interested in the correlation of all the events that belong to an incident I, so that a single event could represent the whole incident. We refer to this single event as the *representative event* for incident I, e_R. To be coherent with the description of the set I, the duration of the event e_R should span from the event creation time of the first event of the incident to the last event resolution time observed in the set of events for that incident. Figure 6.1 shows an example of this definition, where a set of m events belonging to the incident I are represented by a single representative event e_R. It is worth to mention here that, in realistic scenarios, the duration of the representative event could not correspond with the incident duration, mainly due to the fact that when the incident starts, a delay could occur until the first event appear, and the same could happen when the incident finishes, that is, it might be usual to have a delay between the end of the incident and the closing (resolution) of a ticket. Thus, we define a *Forward Offset Delay*, FOD (Figure 6.1), as the delay between the start of incident (SoI) and the time at which the first event appears. In addition, we define a *Backward Offset Delay*, BOD, *i.e.*, the delay between the end of the incident (EoI) and the end of all the related events. Note that BOD could take a negative value, in case that the last event ends before the own incident. For example, if we consider that the events are tickets created and closed manually by staff members, it could happen that they believe that an incident has finished while it is still active. In this case, they would close the ticket (the end of the event), thus making BOD to take a negative value. We will consider these two delays in the tickets-alerts correlation model suggested in the next section.

6.1.1 Basic Correlation Method

Suppose now that we have a set of m events and we do not have any information about the incidents they are related to. We are interested in obtaining the same number of representative events as the incidents that originated those events. In order to do this, in this basic model for event correlation, we assume this hypothesis:

Two events that (i) *have a similar description or have an affected object ID in common,* i.e.*, are related to at least a common network node or service, and* (ii) *happen simultaneously in time, will likely belong to the same incident.*

Mathematically, the first condition, *i.e.*, the similarity in the description or the affected object IDs of two events e_i and e_j, can be described by the following expression:

$$\{E_i \cap E_j\} \bigcup \{D_i \cap D_j\} \neq \emptyset \tag{6.1}$$

while the second condition, *i.e.*, the simultaneous occurrence of two events e_i and e_j, is given by:

$$\Big\{\{t_{e_i}^{CT} \leq t_{e_j}^{CT} \leq t_{e_i}^{RT}\} \vee \{t_{e_j}^{CT} \leq t_{e_i}^{CT} \leq t_{e_j}^{RT}\}\Big\} = true \tag{6.2}$$

In our basic correlation algorithm, if the above two rules are fulfilled by any group of l events, we derive that all of them are related to the same incident, and we simply aggregate them into one representative event, e_R, having the following properties (see Figure 6.1):

$$t_{e_R}^{CT} = min_{i\in[1,l]}\{t_{e_i}^{CT}\} \tag{6.3}$$

$$t_{e_R}^{RT} = max_{i\in[1,l]}\{t_{e_i}^{RT}\} \tag{6.4}$$

$$E_{e_R} = \bigcup_{i=1}^{l} E_i \tag{6.5}$$

$$D_{e_R} = \bigcup_{i=1}^{l} D_i \tag{6.6}$$

We join all the descriptions D_i of the different events and the set of event affected object IDs, E_i, as we consider that any information in one of the events of an incident will complement the information provided in other events of the same incident.

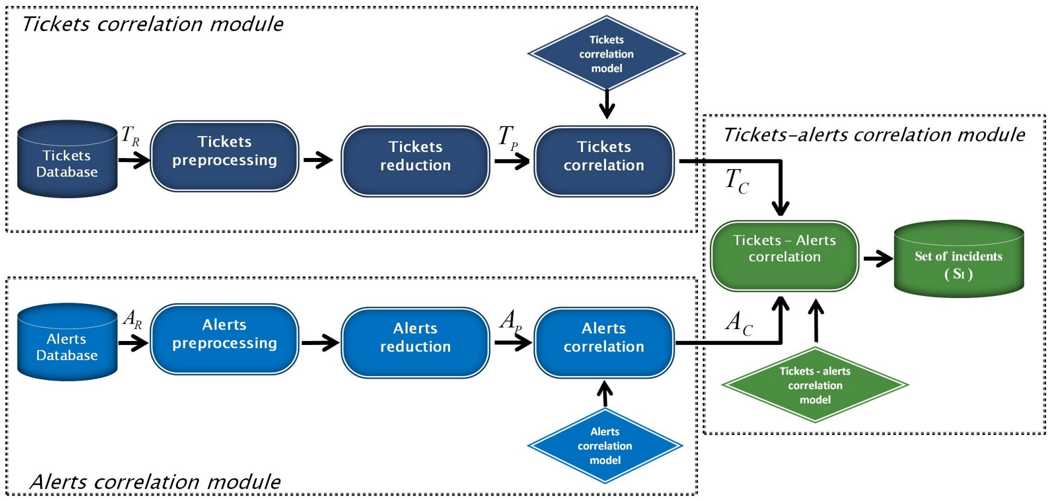

Tickets correlation module
Tickets correlation model
Tickets Database
T_R
Tickets preprocessing
Tickets reduction
T_P
Tickets correlation
Tickets–alerts correlation module
T_C
Tickets – Alerts correlation
Set of incidents (S_I)
Alerts Database
A_R
Alerts preprocessing
Alerts reduction
A_P
Alerts correlation
A_C
Tickets - alerts correlation model
Alerts correlation model
Alerts correlation module

Figure 6.2: Proposed architecture for the tickets-alerts correlation system.

6.2 Tickets-Alerts Correlation System

Figure 6.2 shows the architecture of the proposed tickets-alerts correlation system. It mainly consists of three modules: a module for tickets correlation, other for alerts correlation and the last one for tickets-alerts joint correlation. Later on, we will justify why we split the processing in these three modules. The tickets correlation module is represented in the upper part of the figure. Here, we follow the same methodology proposed in Chapter 5 to correlate tickets. A set of raw tickets, T_R, obtained from the ITS, is entered as input to the tickets preprocessing and reduction phases to extract only incident-related tickets. The resulting processed set, T_P, is then passed through a tickets correlation phase that produces a new set of global representative tickets, T_C. Every global representative ticket, which is ideally expected to represent a single incident, contains the summary of a group of correlated tickets.

The lower part in Figure 6.2 represents the alerts correlation module. Here, a set of raw alerts, A_R, is entered as input to the alerts preprocessing and reduction phases to extract only incident-related alerts, as will be explained next. The resulting processed set, A_P, is then passed through an alerts correlation phase based on the basic model for events correlation (Section 6.1), that produces a new set A_C, that is, the final set of global representative alerts. As in the tickets correlation module, every global representative alert is expected to ideally represent a single incident, and contains a summary of the information provided by a group of correlated alerts.

Finally, the right part in Figure 6.2 represents the tickets-alerts correlation module. Here, the outputs of the alerts and tickets correlation modules, A_C and T_C, are entered as inputs, and the processing is done according to the correlation model that is presented in Section 6.3. The aim is to produce a final set of incidents, S_I, that will more accurately represent the real incidents in the network when compared with methods that only take into account alerts correlation. As in Chapter 5, where the details of the tickets correlation model were explained, in the following subsections we provide a more detailed discussion about the alerts and tickets-alerts correlation modules.

6.2.1 Alerts Correlation Module

As previously explained (see Chapter 3), alerts are usually generated by network elements and obtained by management platforms, *e.g.*, `HP OpenView`, using management protocols such as SNMP (Chapter 2). Each alert is a short message with a specific textual format defined by equipment vendors, and generated as an external manifestation of a potential failure, or a disorder occurring in an equipment of the managed network or system (Figures 3.7 and 3.8). Typically, alerts contain the same relevant information as that described in our basic model presented in Section 6.1, such as: *(i)* affected object ID identifier, *e.g.*, node name and interface name, *(ii)* the timing information of the alert, *i.e.*, the creation and resolution times, *iii)* a description of the fault, *i.e.*, the root cause and the severity of the alert, among others. Besides, alerts may provide information with different levels of details, such as specific data regarding the status of the devices and their configurations, or higher level details, with aggregated information gathered from several alerts.

Alerts are first passed to both preprocessing and reduction modules, with the aim of normalizing the alerts, selecting only incident-related and filtering normal-behavior ones that are generated in response to daily operational tasks that are not really associated to real network incidents, *i.e.*, maintenance activities or software updates, among others.

The output of the alerts preprocessing and reduction phases is fed into the alerts correlation module. Here, we use the basic events correlation model (Section 6.1) in two steps. First, we consider only alerts that are related to a single affected object ID and, in a second step, we incorporate those alerts that are related to a list of multiple affected object IDs. These last alerts are normally generated by intermediate network object IDs that are really doing a correlation of other alerts and generating a new one with the summarized information.

It is remarkable to say that, traditionally, this is the only module that has been implemented in network alerts correlation systems, and a lot of research efforts have been devoted to study it (Chapter 2). In our case, we are not as interested in refining this module as in evaluating if the incorporation of tickets information would improve the alerts correlation process. For this reason, and for the sake of easiness, we have opted for this implementation.

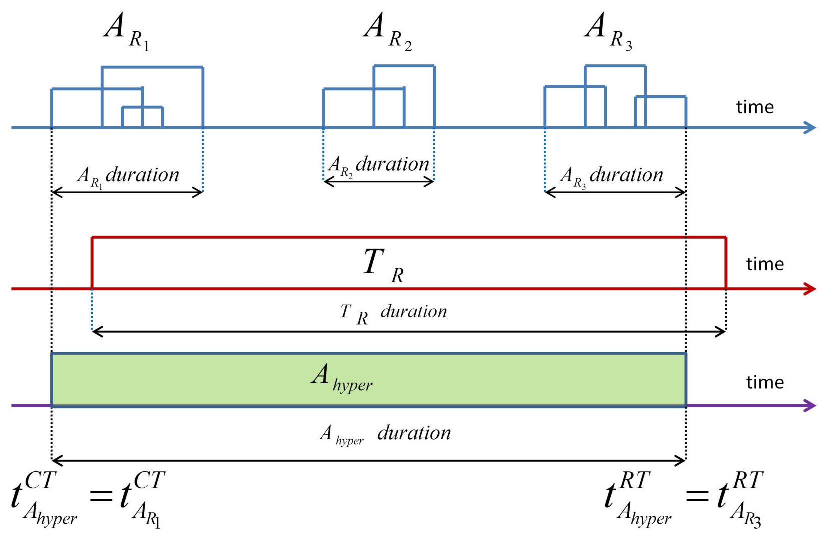


Figure 6.3: Example of alerts reduction using the proposed model.

6.2.2 Tickets-Alerts Correlation Module

This module works with the information provided by both the alerts and tickets correlation modules. As previously stated, we are interested in evaluating whether introducing this module would result in a benefit in the correlation process.

Our intuition is that the tickets can introduce relevant information in the procedure, incorporating human knowledge and significance to the events. An example of a scenario revealing this is depicted in Figure 6.3. We observe in the first timeline the result from an alerts correlation process performed with our basic correlation algorithm, where three groups of alerts are summarized in three global representative alerts: A_{R1}, A_{R2} and A_{R3}. The second timeline represents the output from the tickets correlation process, where a single global representative ticket T_R has been obtained. The third timeline represents the time duration of the incident that generated the different events (alerts and tickets). Note that the alerts in this incident appear intermittently in time, and that this makes the correlation process to consider that they are not overlapped in time and, thus, they are not likely to belong to the same incident. However, the existence of ticket makes it possible to observe the concurrence in time between the three groups of alerts and the ticket, thus allowing the correlation of all of them to represent a single incident.

In what follows, we show that applying the basic event correlation model for both tickets and alerts is not straightforward, and describe the problems that appear and our proposals to tackle them.

6.3 Tickets-Alerts Correlation Model

In order to apply the basic event correlation model suggested in Section 6.1 to correlate both tickets and alerts, it is important first to understand the specificities of both tickets and alerts, and then properly adapt the correlation algorithm. In the following, we first discuss the specific issues to be taken into account, and then present our proposal for the correlation algorithm.

- *Tickets provide better semantic information than alerts:* As shown in Chapter 5, although tickets might be automatically generated by the NMS (automatic tickets), they are usually generated manually by the members of the staff either as a response to alerts or from customers' complaints. Every ticket represents a complete record of an incident to be used by the management staff during the incident management lifecycle.

  Normally, tickets contain more semantic information about the incidents than alerts. First, every ticket contains many free text fields that are used by ticket creators and resolvers to describe the incident, its possible causes and the solutions applied to solve it. In alerts, these fields are normally automatically generated by network facilities and thus, the semantic information is very restricted to a list of possible values. Second, tickets are generated by humans when alert events are considered so important that a record is needed for an incident, or when an user is somehow affected in its normal usage of the network or services. For example, the appearance of alerts regarding non production services, or generated by low-priority nodes in a network, or warning alerts of low-priority should not cause the creation of tickets, as these events should not be considered as incidents.

  Thus, tickets are expected to provide better information than alerts for identifying the actual number of incidents that occur in a network. If we assume that the number of incidents derived from a tickets correlation process is I_T, the number of incidents pointed out by an alerts correlation process is I_A, and the number of real incidents is I_{actual}, we expect to have the following relation:

$$I_{actual} < I_T \ll I_A$$

  For this reason, we will show in our correlation algorithm that we pay more attention to tickets when deciding the number of incidents.

- *Alerts provide better temporal information than tickets:* In contrast with our higher confidence in the semantic information contained in tickets, we claim that the temporal information found in them is less trustable than that provided by alerts. This is due to the fact that, in the ITS system, a large number of the tickets are created or closed manually by the management staff and, thus, their responsiveness is not as fast as in the alerts management system, where alerts are generated automatically in few milliseconds when an event is perceived. So, for determining the beginning and the end instants of an incident, we consider that the timestamps provided by the alerts are the best approximation.

Going back to Figure 6.2 in which we show the complete system, note that instead of considering both tickets and alerts as general events and apply our basic event correlation algorithm to the complete set, we separate both into two processes. This will allow us to determine and identify the number of different incidents (and their semantic information) from the tickets correlation process, and adjust their temporal information from the feedback provided by the alerts correlation process. In summary, the output of tickets correlation is refined with the alerts correlation output, in order to adjust the time information of the incidents pointed out by tickets.

- *Dealing with border effects and the existence of consecutive incidents:* As previously mentioned, it is expected to have some temporary misalignment between the start and the end of an incident, and the creation and resolution times of the associated ticket(s) due to, presumably, human response times. This effect has been included in the basic correlation model through the parameters FOD and BOD. At first sight (Figure 6.4), the problem with these two delays is that it is possible to exclude or include events related or not really related, respectively, to the ongoing incident in the representative event and, therefore, in the incident as perceived after the correlation process. As depicted in the example in Figure 6.4, depending on whether the first event in the events line is an alert or a ticket, it is even possible the appearance of two different incidents at the beginning, while two different incidents can be merged at the end. On the other hand, it is also possible for the management staff to prematurely close a ticket if they have the perception that the problem is solved, thus resulting in a negative value for BOD. In this case, it is highly probable that another ticket really related to the same incident appears after some delay. In fact, this is exactly the former situation in case the first event in the events line is a ticket. To handle this situation, our main argument is that there is a high probability that potentially correlated events that occur in the proximity of other representative events really belong to the same incident. This way, the simultaneity condition –Equation (6.2)– used to merge events is relaxed by using FOD and BOD as thresholds to cover alerts and tickets in the proximity as included in the same incident.

From the point of view of the correlation method, the major impact is expected to arise from the "orphan" alerts, that is, from that alerts at the beginning or the end of an incident that are not assigned to it due to the border effects.

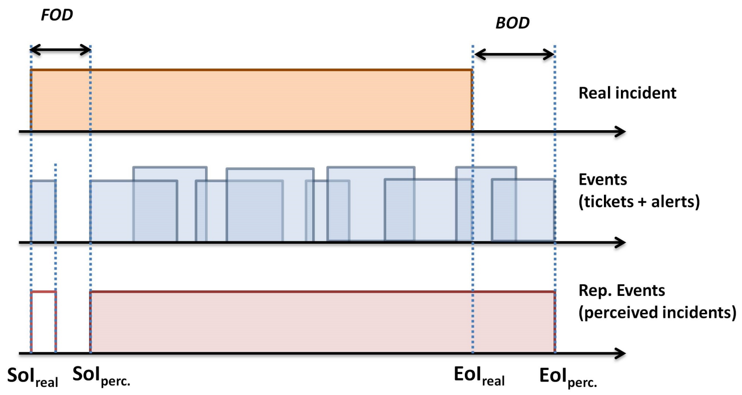


Figure 6.4: Potential effects of FOD and BOD on the correlation results.

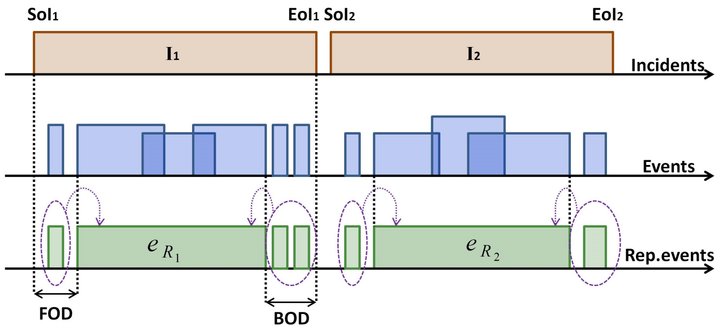


Figure 6.5: Example showing the problem of directly applying the basic event correlation model on two consecutive incidents.

Therefore, some experimental tuning is needed to estimate the values for both FOD and BOD. Obviously, this will be addressed in the experimental setup.

Nevertheless, the scenario can become a bit more complex when consecutive incidents appear. The problem is how to discriminate between any two consecutive incidents having some properties in common, *i.e.*, how to correctly separate between events that could correspond to both incidents or even deciding that both incidents are the same and should be merged. In order to clarify this point, we show an example in Figure 6.5. Here, we assume that we have two really consecutive incidents, I_1, I_2, each one starting and ending at the instants shown in the incidents timeline. We also have a sequence of events, each one starting and ending as shown in the events timeline. Furthermore, we assume that each of these events is related either to I_1 or I_2. If we apply the basic event correlation model suggested in Section 6.1, we will get two global representative events, e_{R1} and e_{R2}, respectively, as shown in the third timeline (*Rep. events*) that, at the same time, will be considered as the incidents from our point of view. If we look carefully at this example, we observe that some events are discarded from the correlation process and they are not correlated, simply because they are not overlapped with any other event. We can assume that the non overlapped events belong to other different incidents, in which case we would have ended up with seven different incidents, far more than the actual two incidents. Thus, directly applying the basic event correlation model in this example would lead to inaccuracies, especially when alert events are considered, because, as mentioned above, alerts may appear earlier than tickets and might not be overlapped with them, not being considered in the correlation process.

In addition, note that there is another problem when consecutive incidents are considered as in our example. We must decide the specific incident, if any, to which the events in between both belong to. In our example, there are three events between e_{R1} and e_{R1}. The choice of the assignment between event-incident modify the temporal duration of both incidents, thus affecting the accuracy of the system.

6.3.1 Tickets-Alerts Correlation Algorithm

To handle all of the above issues, the basic correlation model is modified to consider non overlapped subsets of tickets and alerts as explained before. As shown in Figure 6.1, FOD and BOD will be used as extra time delays thresholds, so that an event is correlated to a representative event, e_R, that is active in the time interval $[t_{CT}, t_{RT}]$ if that event accomplishes Equation (6.1), that is, it satisfies the similarity criteria, and is active in the interval

$$[t_{CT} - FOD, t_{RT} + BOD]$$

Note that, with the expansion of the intervals with FOD and BOD, it could happen that the extended intervals of two consecutive incidents sharing some affected object IDs might overlap. In this case, two operations are considered: *(i)* any potentially related event falling in these intervals will be assigned to the representative event with lower time distance, and *(ii)* the incidents will be merged only if, after adding the in-between events, they are overlapped. Thus, extended intervals are not considered valid for merging incidents simply based on the new limits.

It is obvious that the selection of the values FOD and BOD directly affects to the performance of the correlation algorithm. In Section 6.4 we will show how to experimentally determine optimal values for these parameters and how they affect the overall results.

In summary, we propose an algorithm (Algorithm 1) that starts from an empty list of incidents and consists of two iterations. First, it takes every representative ticket from the correlated ITS database. It is worth to mention that, as the result of the tickets correlation, neither of those tickets are overlapped in time and having at least one object ID or ticket ID in common. For each incident (or global representative ticket), the different model parameters are extracted (creation and resolution times, affected object IDs and descriptions). After that, the algorithm searches for correlated global representative alerts. Every correlated ticket has a list of correlated alerts assigned to it and the temporary limits (SoI and EoI) of the incidents are revisited according to the new information. Second, after extracting a tuple of correlated tickets and alerts, in a second iteration, the algorithm takes every correlated ticket and its associated list of correlated alerts and searches for other tickets having at least one alert in common. The target of this second step is to join the global representative tickets that resulted overlapped and sharing one or more object IDs after adding the alerts in the first step. All matched tickets are aggregated into one having all the information included in the whole group. Finally, SoI and EoI are determined by, respectively, taking the first of the creation times of any of the alerts in the correlated set or the ticket creation time (line 36 in Algorithm 1), and the last of the resolution times of any of the alerts or the ticket resolution time (line 37 in Algorithm 1). The final output is a set of k incidents S_I, such that $S_I = \{I_1, I_2, ..., I_k\}$, being a single incident: $I = \{T_C, A_C, T_E, T_D\}$, where T_C and A_C are the subsets of correlated tickets and alerts for this incident, respectively, T_E is the list of affected object IDs, and T_D is the description of the incident. This set S_I is the estimation of the actual incidents represented by all the events (tickets and alerts).

6.4 Experimental Results

Once the basic model and the methods to handle the challenges in tickets-alerts correlation are described, next we present the experimental assessment of the proposal applied to the set of alerts and tickets presented as dataset DS2 (see Table 3.1). Then, we analyze the results and give several findings.

6.4.1 Real Scenario: Dataset and Preprocessing

As explained in Chapter 3, the lack of labeled data can introduce some confusion in the interpretation of the results, as not all the real alerts are to generate tickets, due to the fact that the management staff can consider them irrelevant at a given time. In fact, alerts in NMS are usually classified according to their severity and/or criticality (Table 3.2). Thus, not all the alerts present the same effects on the stability of the managed system and the management staff is prone to ignore or postpone the creation of a ticket for non critical alerts, especially if they are busy trying to solve an incident with higher priority. As a consequence, even if the number of tickets were accurate, not all the alerts would be correlated to a ticket, *i.e.*, to an incident, which could be interpreted as a failure in the proposed method.

Algorithm 1 Tickets-alerts joint correlation

```
procedure TicketsAlertsCorrelation
    Initialize: incidentList()=null, FOD, BOD
    loop                                        ▷ First step: for each representative ticket
        $ticket$ = getNewRepresentativeTicket( )
        $L$ = listOfAlerts(ticket)
        $T_{CT}$ = getTicketCreationTime(ticket)
        $T_{RT}$ = getTicketResolutionTime(ticket)
        $T_E$ = getListAffectedobject IDs(ticket)
        $T_D$ = getListDescriptions(ticket)
        loop
            $alert$ = getNewIncidentAlert( )
            $A_{CT}$ = getAlertCreationTime (alert)
            $A_{RT}$ = getAlertResolutionTime(alert)
            $A_E$ = getAlertAffectedElement(alert)
            $A_D$ = getAlertDescription(alert)
            if $(A_{CT} \geq T_{CT} - FOD \ and \ A_{RT} \leq T_{RT} + BOD) and \ (A_E \in T_E \ or A_D \in T_D)$
                L.add (alert)
            end if
        SoI = min(getFirstCreationTimeAlerts(L),$T_{CT}$)
        EoI = max(getLastResolutionTimeAlerts(L),$T_{RT}$)
    for $i = 1$ to $m$                          ▷ Second step: merge newly related tickets
        $T_i$=getCorrelatedTicket()
        $L_i$=getlistOfCorrelatedAlerts($T_i$)
        $T_{CTi}$=getTicketCreationTime($T_i$)
        $T_{RTi}$=getTicketResolutionTime($T_i$)
        for $j = i + 1$ to $m$
            $T_j$= getOtherCorrelatedTicket( )
            $L_j$= getlistOfCorrelatedAlerts($T_j$)
            if $L_j \cap L_i \neq \emptyset$ then
                $L_i$.add($L_j$)
                $T_{Ei}$.add($T_{Ej}$)
                $T_{Di}$.add($T_{Dj}$)
                del $T_j, L_j$
                $CT$ = getTicketCreationTime(T)
                $RT$ = getTicketResolutionTime(T)
                SoI = min(getFirstCreationTimeAlerts($L_i$), $T_{CT}$, $CT$)
                EoI = max(getLastResolutionTimeAlerts($L_i$),$T_{RT}$, $RT$)
            end if
        end for
    end for
    incidentList.add (SoI, EoI, $T_{Ei}$, $T_{Di}$)
```

DS2	**Alerts**	**Tickets**
Total number of records	1,703,662	9,162
Mean number of affected object IDs/record	1.15	1.42
Number of records after preprocessing and reduction	913,042	8,105
Number of relevant records	7,436	520 (348 MS/172 SD)
Number of representatives (before joint correlation)	1,539	432 (286 MS/146 SD)
Mean number of records/repr. set	4.8	1.2

Table 6.1: Some useful statistics of both the alerts and tickets taken from DS2.

To tackle with this situation, we have checked the performance of the correlation method presented here with an especial subset of events, that is, we only consider relevant incidents. By relevant incidents we refer to those affecting the operation of the network in a critical way and that, consequently, must present associated tickets. According to the technical procedures of the company, two are the situations in which an alert should mandatorily trigger a ticket from the management staff: critical alerts, which are those really affecting critical object IDs; and massive alerts, which are alerts automatically generated by the NMS as a response for a big number of alerts from topologically related object IDs in the network. The first situation is identified by using a list of network nodes (the critical ones) and the type of critical alerts (*i.e.* NodeDown, InterfaceDown) so that a critical alert in a critical node should generate a ticket by the management staff. Therefore, the main criterion for measuring the performance of the proposed correlation procedure can be stated as

All the relevant alerts should be assigned to tickets.

Some relevant figures for DS2 –both the tickets and the alerts– handled during the phases previous to the tickets-alerts correlation are provided in Table 6.1. It is remarkable that we distinguish between tickets generated by the *Management Staff*, (MS), and *Service Desk*, (SD), to ease the assessment of the results.

6.4.2 Experimental Tuning and Validation

As explained in Section 6.3, it is necessary to consider some "border effects" in the correlation procedure due to potential misalignments between the real timing of an incident and its manifestation in tickets and alerts, probably due to the humans involved not being reactive enough. To handle this, it is necessary to obtain an estimate for FOD, that is, the maximum accepted time from the appearance of the first alert of an incident and its corresponding ticket, and for BOD, that is, the maximum accepted time from the end of the incident and the closing of the ticket. For this, a set of experiments was carried out by varying the value for FOD and BOD. The percentage of alerts correlated to tickets as a function of the value for FOD and BOD is presented in Figure 6.6. As shown, as FOD increases also the percentage

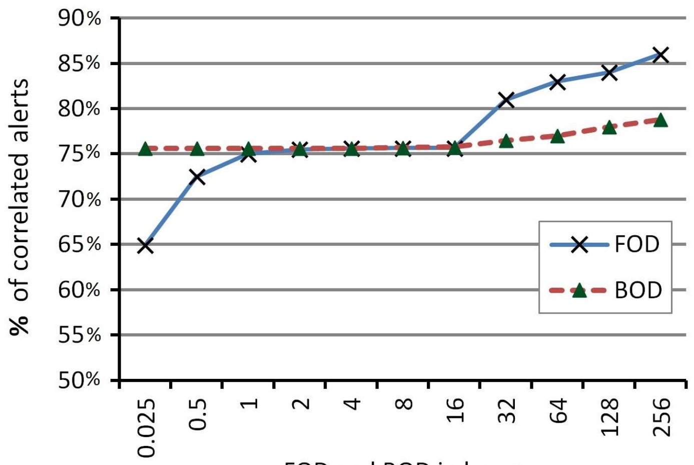


Figure 6.6: Percentage of correlated alerts at different values of FOD and BOD.

of correlated alerts does, which would imply that, the greater FOD is, the best the correlation. But this might be an erroneous conclusion, as big values for FOD would merge together independent incidents involving some common affected object ID. On the other hand, having big delays is not reasonable in ticket creation for critical incidents. Therefore, a careful analysis of the results taking into account the tickets/alerts that resulted inappropriately merged (correlation errors) is required. Anyway, taking into account the shape of the graph in Figure 6.6, we selected FOD = 4 h as the initial candidate to start with testing. Therefore, we try to assess the validity of the results regarding the correlated object IDs by inspecting some samples next.

Regarding BOD, as shown in Figure 6.6, there is no relevant influence of this value in the results. This was somehow expected, as the delay in the validation of the last ticket should be mainly associated to a lack of related alerts, and not to their existence.

Validation of the Correlated Sets

As previously stated, the dataset lacks of a ground truth to relate the existing incidents with their corresponding alerts and tickets. Furthermore, manually labeling the dataset is unaffordable. Thus, validating the results is not straightforward. To overcome this issue, we validate our correlation results by manually inspecting many samples and applying the knowledge and rules of thumb provided by the company management staff, which helped us during this procedure. Thus, for the cases that were not clear enough for us, we got additional feedback from the company.
First, a set of 100 randomly chosen incident samples of correlated tickets and alerts

Ticket lifetime	58.8 h
Incident lifetime	81.4 h
Number of repr. alerts/ sample	5.5
Number of alerts/ repr. alerts	4.8
D_{PR}	33.3 h

Table 6.2: Mean values for some variables for the correlated subset.

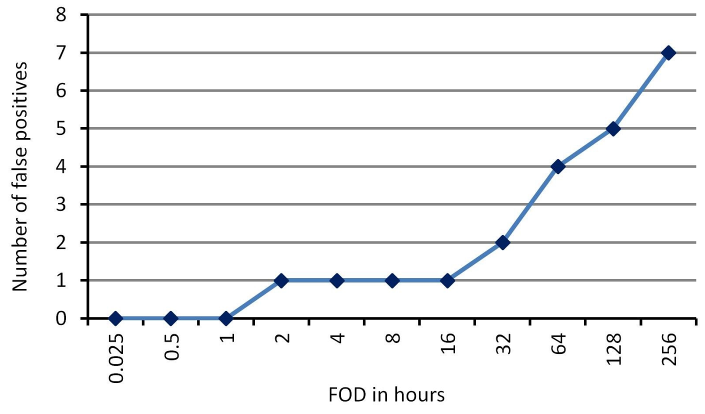


Figure 6.7: Number of false positives at different values of FOD.

were considered and studied manually, for FOD = 4 h and BOD = 0 h. Previously, some statistics (Table 6.2) were collected from the correlated subset; we define some of them in the next sections. We found that 99 of them were undoubtedly classified as correctly correlated. The remaining sample is not a clear case, as there is not enough information in the ticket and the alerts as to decide whether they are related or not. Thus, assuming the worst case, there is a single error in 100 samples, providing an estimated value of 99% of a posteriori accuracy, that is, 99% of the found correlations are correct at this operation point.

For the 100 chosen samples, we have studied them by varying the value of FOD, starting at FOD = 0.025 h, that leads to removing/adding some events in each incident, providing the values in Figure 6.7 for the number of FPs for which we found inappropriately assigned events for some samples. Consequently, after combining the information from Figures 6.6 and 6.7 it is evident that a value for FOD = 1 h represents the best compromise in the case study we are considering here. In this case, choosing FOD = 1 h will obtain high correlation percentage with zero FPs.

As this sampled validation is not enough, we went deeper and followed another strategy. As explained in Section 6.3, one of the most conflictive cases for the correlation was related to consecutive incidents. To be more confident about the results and have an insight on this particular case, we can consider the delay between the maximum resolution time among all the previous tickets (T_{Pi}) having the same af-

DS2	**Alerts**	**Tickets**
Number of initial alerts/tickets	7,436	348
Number of representatives alerts/tickets	1,539	286
Number of initial alerts/tickets correlated	5,686	302
Number of representatives alerts/tickets correlated	1,178	215
Percentage of representatives alerts/tickets correlated	76.5%	75.5%
Number of correlated incidents	215	

Table 6.3: Correlation results considering only tickets created by the management staff (MS group).

fected object ID as the current one, and the first appeared alert of the current ticket. This delay value, labeled D_{PR} in Table 6.2, was calculated as:

$$D_{PR} = t^{CT}_{A_c} - max_{i \in [1,m]} \{ t^{RT}_{T_{Pi}} \}$$

where the first term represents the creation time of the first alert related to the current ticket, and the second term represents the maximum resolution time among all the previous tickets having the same affected object ID.

In our case, D_{PR} has a mean value of 33.3 h, which is significantly greater than the selected value for FOD. This can be interpreted as a clear indication that the first appeared alert is related to the current incident and not to a previous one for the selected value FOD = 1 h.

After choosing the value for FOD, the correlation results taking into account only the tickets created by management staff are presented in Table 6.3. As shown, around 75% of the tickets and alerts potentially related to relevant incidents are correlated. For the tickets, this means that 75% of them are really related to relevant incidents, which provides no further information on the quality of the correlation itself, as the remaining 25% could be related to the critical nodes, but not to a critical episode. On the other hand, having only 76.5% of relevant alerts correlated to a ticket is, at first sight, not a very good result, although it represents and advance as no other similar system has been described. Anyway, this result needs a deeper analysis in order to find the potential causes for such a figure, what is addressed next.

Analysis of the Non Correlated Alerts

Although the effectiveness of the proposed technique in terms of improperly correlated events has shown to be high, around 1/4 of the relevant alerts are not assigned tickets, what requires further analysis. According to the protocols in use at the company, all of the considered alerts should have generated tickets from the management staff. This incoherency can be initially attributed to the fact that the

Working shift	% of relevant alerts	% of non correlated alerts
RS	50.6%	53%
AS	16.9%	28%
NS	32.5%	19%

Table 6.4: Distribution of the number of alerts over working shifts.

proposed method is not accurate enough. But an improper application of that policy or some problems with the staff could also explain it. Therefore, we analyzed those non correlated alerts taking into account three potentially influential factors: *(i)* the working shift at the time of their creation, *(ii)* alert durations and *(iii)* the interarrival delay of alerts having the same main object ID.

The working shifts can reveal relevant information, as the fact that the number of persons in charge of the management is not the same for all the shifts and also the workload is different. After some consultation to the company, we found that there are three working shifts in a day: the morning shift (RS) from 7:00 AM to 15:00 PM; the afternoon shift (AS), from 15:00 PM to 23:00 PM; and the night shift (NS) from 23:00 PM to 7:00 AM. Furthermore, the characteristics of the working shifts change during weekends or holidays.

The analysis of non correlated relevant alerts as a function of the working shift is summarized in Table 6.4. As shown, there exist differences in the working shift regarding the distributions of incidents (alerts) and the percentage of non correlated alerts. Thus, while almost half of the alerts appear during the busiest working shift, that is, RS, the same percentage of non correlated alerts appear. Nevertheless, during AS, the percentage of non correlated alerts is not in consonance with the percentage of existing alerts, which can imply a shortage in personnel for AS or an improper behavior during it. The opposite occurs during NS. Anyway, the differences are not significant enough as to explain the appearance of non correlated alerts.

An additional analysis of the duration of the non correlated alerts was made, showing the results provided in Figure 6.8. As a first conclusion, we observed that about 70% of them present a duration lower than 10 m. This means that the alert is shown as active only for at most 10 m in the management staff console. With the help of the management staff, we reached to the conclusion that this is an acceptable threshold value to distinguish between normal-behavior and relevant alerts before triggering the ticket creation, and also that the dedication of the staff to other tasks can hide these kind of short alerts.

Finally, for alerts having a duration greater than 10 m, we analyzed the interarrival delay between consecutive alerts having the same main object ID in order to see if they are created close in time or there is a time gap between them. Fig-

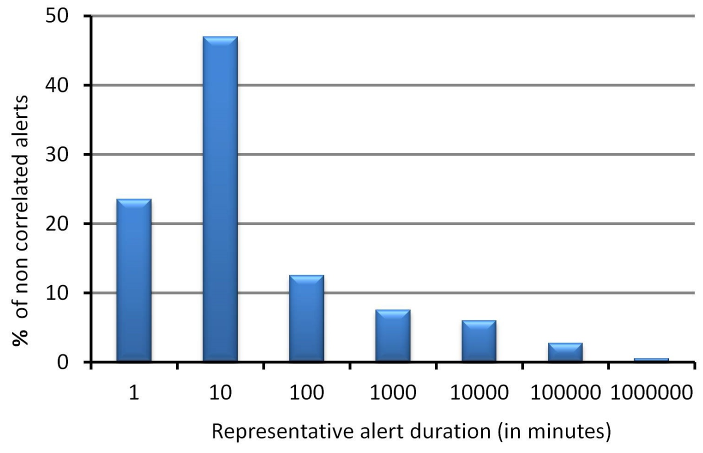


Figure 6.8: Histogram of non correlated representative alerts vs. alert duration.

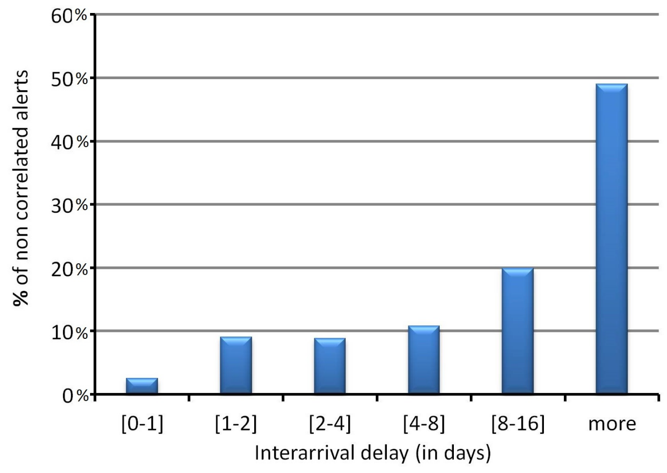


Figure 6.9: Interarrival delay between consecutive non correlated alerts having the same main object ID.

ure 6.9 shows the histogram of the interarrival delays. We found that less than 10% of them were repeated within 2 days, that is, most of them appear and, despite its long or short duration, no additional alerts related to the same affected main object ID appear in at least two days. This means that the alert is scaling down in the list of active alerts on the management staff console. Therefore, we conclude that the lack of an associated ticket can be possibly attributed to the existence of a “window of opportunity” for the creation of the tickets. Thus, if an alert does not trigger a ticket within a given period and it is not repeated, it is likely that it will not trigger a ticket at all. After discussing this question with the company’s management staff, they partially agreed with this observation.

Additionally, besides the above conclusion for the last subset, we found that about 38% of the non correlated alerts included names for main object IDs not con-

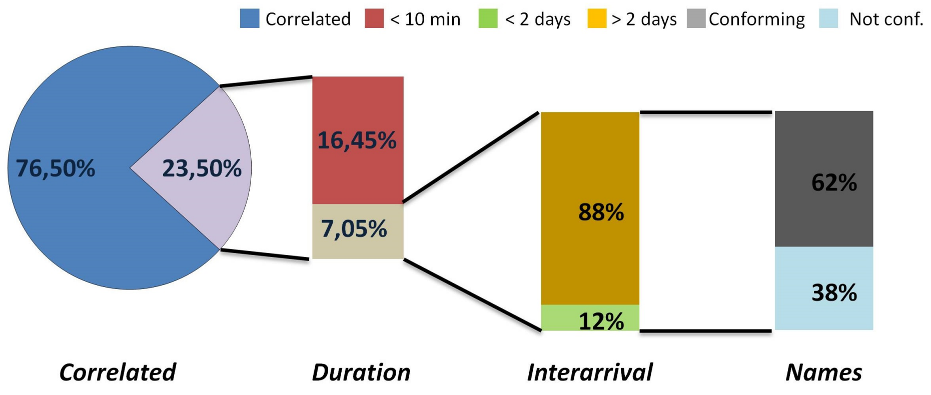


Figure 6.10: Distribution of alerts.

forming to the naming convention in use. After consulting with the management staff we were informed that these types of IDs, despite classified as critical ones, have special functions not being used by other object IDs. So, the management staff does not usually open tickets for those types of affected object IDs. That is, we were initially provided with an inaccurate list of critical nodes.

As a resume, Figure 6.10 shows the results from the assessment of both correlated and non correlated alerts. It is worth to note that, if we accept that alerts lasting less than 10 minutes are prone to be ignored, only 7.05% of the initial representative alerts remain non correlated without an explanation. Considering those with names conforming the critical nodes list, the percentage of non correlated alerts is considerably reduced to a 4.37%.

6.5 Chapter's Conclusions

In this chapter, we proposed a model for the joint correlation of tickets and alerts in network management. One of the main drives for the proposed method is to enhance usual alerts correlation methods under the assumption that the tickets provide additional relevant information about incidents. This information is also richer from the semantic point of view, as they contain information from users and network administrator, thus, incorporating human knowledge and relevance into the process.

The algorithm was implemented and validated through a series of experiments that showed that the accuracy of the algorithm mainly depends on the value of FOD, which is the overriding factor in determining the number of *False Positive*s, (FPs). In our case, we got the best correlation results for the tested dataset with FOD = 1 h. Consequently, we conclude that, choosing the right value of FOD is an important task and may affect on both the accuracy and the performance of the system. At the same time, the proposed method is based on simple object IDs and reasoning, making its application in a real NMS almost straightforwardly.

Due to the fact that the dataset tested here is not supervised, *i.e.*, no information about ground truth is present, we developed our own mechanisms to validate the results that are mainly based on inspecting many samples manually. In this regard, the validation method applied here considered a subset of alerts and tickets for which we were confident about their correlations.

Chapter 7

Applications of the System

As revealed in the different results and conclusions drawn from Chapters 3 to 6, the process of handling the tickets by a management team is not completely systematic and may be incoherent. While there exists a large number of commercial tools for ITS focused on improving the whole system to meet SLAs, regretfully, only few efforts have been devoted to improve the efficiency of the tickets generation process itself. This, at the same time, would improve the overall task of solving the incidents, having a direct effect on the stability of the managed network as well.

In this chapter, three possible applications of the tickets and alerts correlation are presented. First, using the proposed models as an ITS assessment tool, analysts can obtain some insights about the efficiency of the management teams and the processes they use, in particular for the first stage of the ticket management lifecycle, *i.e.*, the ticket creation. Second, the suggested models can also be used to measure the staff efficiency in the incident management process, in which they can help analysts to answer several questions: Do SD systems help in the tickets-alerts correlation process? Do the different working shifts behave the same? How fast/accurate is the staff? among others. The results can be useful from the point of view of enhancing both the management teams and the policies for tickets creation.

The third candidate application is targeted at the alerts correlation problem. The proposed tickets-alerts correlation model can help the management staff to aggregate a higher percentage of alerts and to speed up the incident resolution process.

In order to check the usability of these proposals, the proposed models have been applied to a case study composed of alerts and tickets taken from the database DS1, already explained in Chapter 3.

The three following sections will describe in detail each of these applications.

7.1 ITS Processes Assessment

As a first application, the proposed tickets correlation method can be used as an assessment tool for measuring the quality of the tickets creation process. According to the proposed method, the efficiency was measured by obtaining the three levels of tickets reduction, *e.g.* preprocessing, local and global (Section 5.5). For each level, the reduction rate was calculated separately, since there are different types of entries in each one.

As shown in Figure 3.1, the whole process consists of three main steps. First, malformed and irrelevant tickets are filtered. Second, the number of tickets is reduced by substituting each subset of locally correlated tickets into a local representative ticket, using for that purpose the local correlation algorithm discussed in Section 5.5.3. Third, the total amount of tickets is decreased by using the global correlation algorithm proposed in Section 5.5.3 to obtain a global representative ticket for each subset of globally correlated tickets. The procedure takes the original ticket database (θ) as an input and provides the processed database (δ) as an output, presenting the different performance indicators for each step.

The efficiency of the process of creation of tickets (E) for every step x (preprocessing and reduction, local, global, overall process) can be defined as:

$$E_x = 100 - \varphi_x, \ where \ x = P, L, G, overall \tag{7.1}$$

where φ_x is calculated from Equations (5.11) to (5.14).

The ideal ITS would be that in which the number of tickets generated equals the number of incidents, *i.e.*, $I \cong O$. In this case, according to Equation (5.14), $\varphi_{overall} = 0\%$, that is, there would be no malformed, irrelevant and redundant tickets and, therefore, $E_{overall} = 100\%$.

The efficiency results obtained when the algorithms are applied to the DS1 dataset are shown in Figure 7.1, where it is noticeable the high number of irrelevant tickets (up to 16% in May). From the point of view of the ITS, irrelevant tickets should have not been created, as they are not related to incidents and/or do not contain the minimum required information to be useful for incident solving. An inspection of those irrelevant tickets shows that many of them are not related to incident solving but to other issues, *i.e.*, administrative ones. This could be considered a misuse of the ITS depending on the active policies.

Also in Figure 7.1, a major redundancy in tickets can be found at the local level, that is not reasonable from the management's point of view as the related tickets are created overlapped in time for the same affected object ID. If the results are ana-

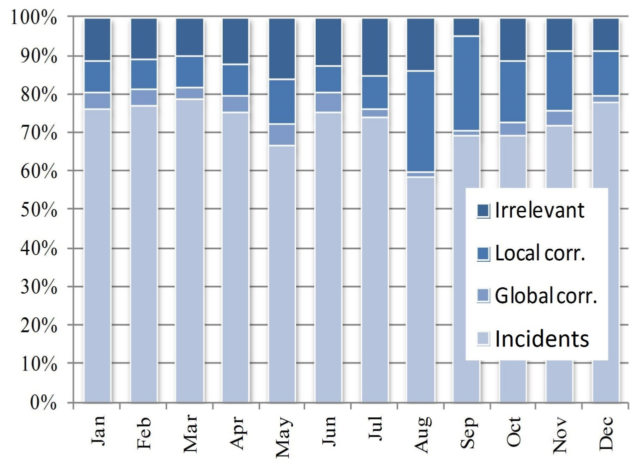


Figure 7.1: Percentage of tickets removed in each step for DS1: irrelevant tickets, local correlation reduction, global correlation reduction, final number of global representative tickets (incidents).

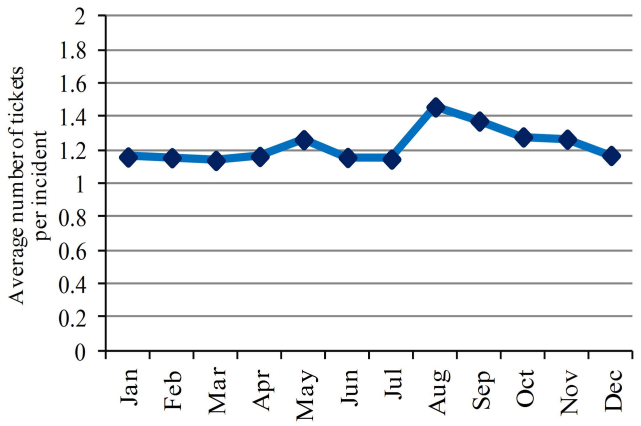


Figure 7.2: Average number of tickets created per incident in DS1.

lyzed from a different point of view, a significant indicator would be the relationship between the number of incident-related tickets (P) and the number of incidents (I). Assuming that G is approximately equal to I, Figure 7.2 shows the mean number of tickets created for every incident. On average, there are 1.23 tickets per incident after filtering malformed and irrelevant tickets. This means that there is room for improvement in the creation and handling of the tickets.

In real situations, the three levels of redundancy could have different effects on the overall efficiency of the IT company, *e.g.*, the cost of creating irrelevant tickets may not have the same weight as local or global redundant tickets. Therefore, instead of considering the overall efficiency –Equation (7.1)–, the different partial efficiency measures can be used in a weighted form to evaluate the impact of each level of redundancy. This way, it would be possible to account for several reasons that play important roles, such as the degree of coordination between management groups, especially in a multi-team or multi-shift environment; updates on the calls and incidents coming on no matter who worked last on this particular incident; incidents extended over shifts; incidents that might be addressed by several different teams in the same shift; the mean time between correlated tickets that might be used as a good indicator of the behavior of the management team for creating such kinds of redundant tickets and others like incident severity and SLA. Next, we highlight how to obtain some of these metrics.

7.1.1 Insights from Tickets Management Groups

As a complement to the assessment of the overall performance, it would be interesting to evaluate the amount of redundancy in the tickets creation process contributed by each of the management groups. Doing so might considerably help in identifying procedural problems and failures in coordination. Thus, in the following subsections, the behavior of each of the management groups was studied separately, and analyzed based on its relation to the overall performance according to the proposed correlation model. As noted earlier, there are three groups that have the right to manually create tickets; they are MS, SD1, and SD2. The first one is the management staff that creates tickets according to network alerts, while the last two are the service desks, that create tickets as a response to received customer calls.

Insights from Preprocessing Analysis

Figure 7.3 shows the distribution of irrelevant tickets generated by the three management groups. From the figure, only 1% of irrelevant tickets are created by *Management Staff* (MS), while up to 73% by SD1 and finally 26% by SD2. Thus, we can conclude that SD1 presents very low efficiency in the tickets generation process due to the creation of a high amount of unnecessary tickets: more than 3/4 of irrelevant tickets was created by staff members belonging to this group. These figures point to a potential fault in the procedures used by them. Nevertheless, the nature of SD1 can partially explain this extremely high value compared to the others. As noted previously, SD1 is classified by the company as a call center level 1; that is, it is the first department that receives customers' calls. Consequently, the staff may create a large number of irrelevant tickets because they may receive many calls made by customers complaining about non-existent or non-networking related problems and/or providing insufficient information to properly identify the fault as a result of the customers not having enough knowledge about the normal operation of the system they are working with. Anyway, a review of the procedures used by this group is advisable. On the opposite side, the MS group created few irrelevant tickets, what is coherent with the procedures, as the staff belonging to this manage-

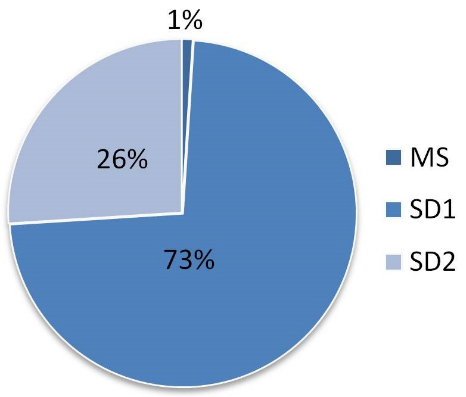


Figure 7.3: Distribution of irrelevant tickets created by each management group.

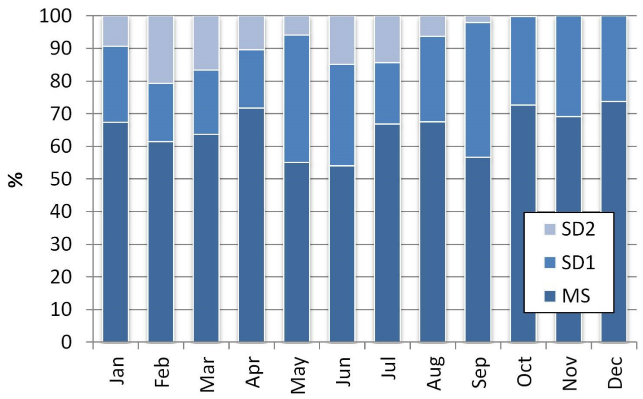


Figure 7.4: Distribution of locally correlated tickets as generated by the three management groups.

ment group creates tickets based only on receiving alerts and, thus, the chances of creating irrelevant tickets are lower.

Insights from the Local Correlation Analysis

Once those malformed and irrelevant tickets are filtered, the local correlation is used to identify the tickets that can be merged at this stage. Figure 7.4 illustrates the distribution of these tickets according to the creator group. We observe that, on average, more than 65% of the locally redundant tickets are created by MS, 26.6% by SD1, and finally 8.4% by SD2. Unlike in the case of unnecessary tickets, a high percentage of these redundant tickets were created by the staff members belonging to MS, *i.e.*, most of the redundancy is somehow related to MS. In order to check whether the problem arises from tickets being created by more than one group or by duplicated tickets from the same group, an additional analysis was made. We take each group of locally correlated tickets, that is, those tickets that will be merged

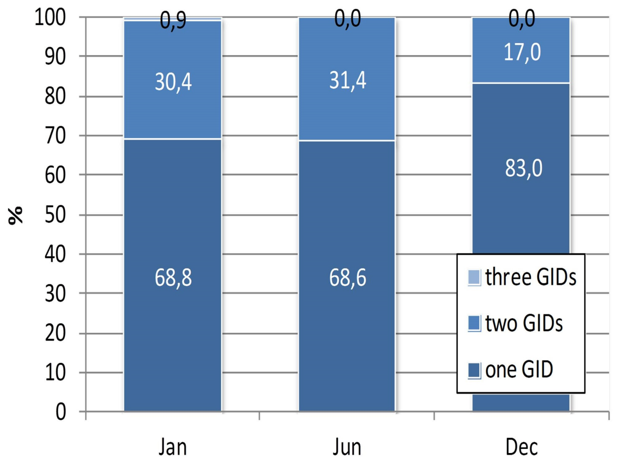


Figure 7.5: Distribution of the number of locally correlated subsets as generated by management groups.

together by the local correlation algorithm, and examine whether they were created by the same or by different management groups.

The results for the three randomly chosen months, shown in Figure 7.5, reveal that, on average, more than 73.5% of the locally correlated subsets included tickets generated by a single group. This is a surprising result, as redundancy was expected to be mainly generated due to the existence of different management groups, each of them generating tickets for the same incident. As shown in Figure 7.6, a deeper insight into these results reveals that, from these subsets, 72.8% (mean value) of them are due to MS, 22.1% to SD1, and 5.1% to SD2. Obviously, the procedures used for tickets creation by MS should be revised.

Keeping on with the original analysis, 26.2% of the locally correlated subsets include tickets created from two groups, while only 0.3% of them were generated from the three groups. A manual inspection of many samples revealed that, as expected, the bulk of tickets coming from two different groups involved MS and SD1. Obviously, this is due to a lack of coordination between management groups (inter-management). For example, if a given element of the network is down, *i.e.*, a router, the staff at SD1 may receive calls from end users complaining about some problems in accessing services or applications affected by this element. Consequently, a ticket containing some preliminary information about the ongoing incident is created. At the same time, the staff at MS may receive alerts triggered by the same element announcing the existence of the same incident. Consequently, the staff creates another ticket related to the same incident. Therefore, this lack of coordination between different groups can lead to the creation of many redundant tickets.

In the above mentioned scenarios, the local correlation algorithm could be easily

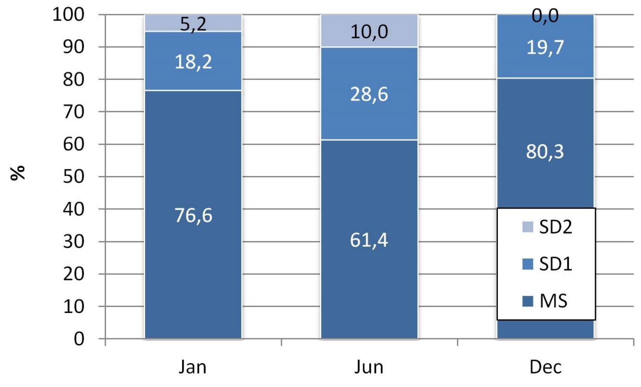


Figure 7.6: Distribution of the number of locally correlated subsets having only one group ID as creator of the tickets.

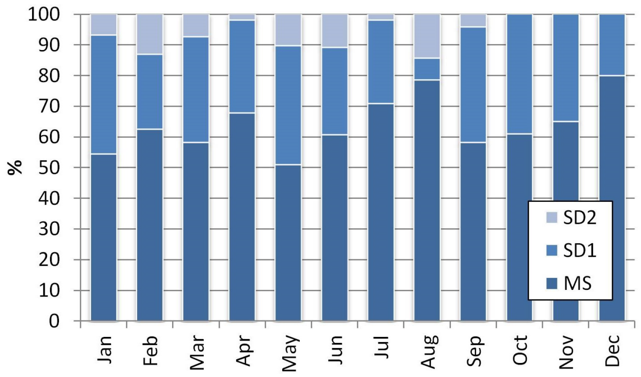


Figure 7.7: Distribution of globally correlated tickets as generated by the three management groups.

implemented in real time as a filter to detect these situations and avoid creating additional tickets that are a duplicate of an active one.

Insights from the Global Correlation Analysis

Similarly to the local analysis, Figure 7.7 illustrates the distribution of the globally correlated tickets for the three management groups. As in the local case, it is MS who is responsible for the majority of the redundant tickets (64%), followed by SD1 (30%) and SD2 (6%). The analysis of the sources for groups of related tickets is summarized in Figure 7.8, with similar behavior to the case of local correlation. Thus, more than 70% of the groups of related tickets contain tickets that were created by the same management group, again with MS being the dominant one (up to 69.3% of them). A manual inspection of many samples of globally correlated subsets reveals that staff at MS normally relate several tickets to each other if they are located nearly in the same network region. For example, the MS may receive many

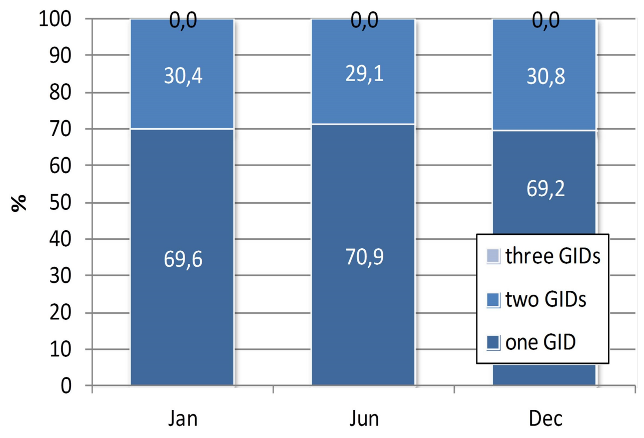


Figure 7.8: Distribution of the number of globally correlated subsets as generated by management groups.

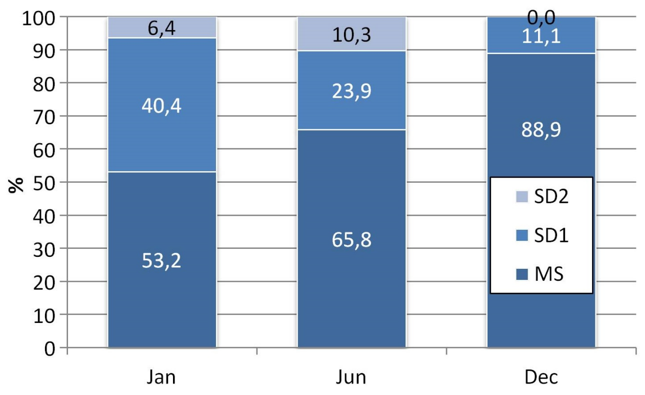


Figure 7.9: Distribution of the number of globally correlated subsets having only one group ID as creator of the tickets.

alerts triggered from two different network elements located nearly in the same geographical region. Consequently, they create two different tickets related to the same incident. The remaining subsets contain tickets created by two different group IDs, namely MS and SD1. As before, this can be due to the lack of coordination between the management groups involved in incident resolving tasks.

7.2 Measuring Staff Efficiency

A second good candidate application for the proposed tickets-alerts correlation model is providing some insights on how to evaluate the efficiency of a management staff. The proposed models can help analysts in answering several questions such as: Do Service Desks help in the correlation process? How fast/accurate is the staff?

Database	Alerts	Tickets
Number of initial alerts/tickets	7436	520
Number of representatives alerts/tickets	1539	432
Number of initial alerts/tickets correlated	6104	346
Number of representatives alerts/tickets correlated	1261	259
Percentage of representatives alerts/tickets correlated	82%	60%
Number of correlated incidents	259	

Table 7.1: Correlation results for relevant tickets created by both groups: the management staff (MS) and the service desk staff (SD).

among others. In the following subsections we study in more detail all of these questions.

7.2.1 Service Desk Systems

If the tickets that are related to both MS and SD are considered, it might be interesting to check whether SD systems play an important role in the incident solving process by applying the correlation algorithm to the tickets created by both groups. The results, shown in Table 7.1, when compared with those in Table 6.3, that contains correlation results for only tickets created by the MS group, evidence that the proposed tickets-alerts correlation model is capable of correlating 25.6% of SD tickets, and increasing the number of correlated representatives alerts in an additional 5.5% (82%-76.5%).

Therefore, the main conclusions derived from this analysis are: *(i)* SD systems are meaningful to assist in the incident solving problem and are not just a call center for handling customer calls and complaints; and *(ii)* the proposed method is able to incorporate relevant information, that is not available from any other source, into the correlation process, thus improving the quality of the results and reducing the number of elements at the output.

7.2.2 Staff Accuracy and Speed

Another possible application for the proposed system is to provide some insights on how to evaluate the efficiency of the management staff during the incident resolving process. The proposed system might help analysts in answering several questions related to the quality of the management, as: Do all the working shifts and management groups behave in the same way? How fast/accurate is the staff? An example regarding the first question has already been considered in Section 6.4.2, revealing a lower performance than expected for AS working shift.

As an additional example, consider the case in which the analysts are interested in measuring the reaction time of the management staffs, a group of persons or even an individual member of the staff, in dealing with incidents. For this, some measures could be useful such as how much time the management staff needs to open a ticket for an ongoing incident, *i.e.*, the delay between the first appeared alert and the first ticket creation time of an incident. In this case, the average value obtained from the considered case study is 3.2 hours. But, if we need to measure how much time the management staff needs to close a ticket for an already resolved incident, the delay between the first resolved alert and the last resolved ticket related to an incident can be measured. In the considered case study, the mean value for this magnitude is 112.4 hours, that is certainly a big value. Similarly, other measures from the model can be used and interpreted.

7.3 Alerts Reduction

As concluded in Chapter 2, alerts correlation is still an active research area in both NMS and network and system security. The proposed tickets-alerts correlation model is a strong candidate to be used in this field, to increase the reduction percentage of alerts that overwhelm the management staff, and also to enrich the semantic information from both technical staff and end users in the alerts correlation process.

The results regarding the capabilities of the tickets-alerts correlation provided in Section 6.4.2 are a clear indicator of the potentialities of the proposed method. As shown in Table 6.3, there is a big reduction in the number of representative alerts, that is, in the number of different alerts after alert clustering when including the information from the tickets. In fact, from the initial 1539 alerts to consider, 1178 of them are clustered in 215 incidents, with an average of 5.47 representative alerts per incident. Therefore, the number of final alert sets to consider is of only 576, that is, a third part of the original correlated set and 1/12 of the original number of alerts. Furthermore, the analysis of the alerts that could not be correlated evidences a low confidence on its relevance.

These results confirm our intuition regarding related alerts not overlapped in time (Figure 6.3) and the inclusion of additional relationships created by tickets. Therefore, we conclude from this interesting finding that incorporating tickets information in the alerts correlation process will definitely help in reducing a higher percentage of related alerts.

Nevertheless, it is important to mention here that not all of the potentialities of the system have been used, as the tickets from SD have not been considered during the assessment of the method. As a matter of fact, the information in these tickets can be far more significant than that in the MS tickets, as they incorporate the end users perception of the incident.

7.4 Chapter’s Conclusions

Along this chapter, we have described through a case study that the proposed correlation models can be used as an assessment tool to provide some insights about the management staff’s efficiency at the tickets creation and incidents handling processes. This way, it is possible to identify some deficiencies in the procedures, policies or behaviors of the different actors involved in the ITS management process, enabling corrective actions to be taken and evaluated. Second, the models might also be used to reduce the number of alerts in the alerts correlation problem. Incorporating ticketing information in the alerts correlation process will definitely help in reducing a higher percentage of related alerts as compared with generic alerts correlation techniques that only use the alert database as the main source of information.

Finally, the main argument of this work is that by simplifying the analysis of incident tickets and incorporating their information in the alerts correlation process, analysts can assess the procedures in order to increase enterprise’s profits and management staffs can organize their works and speed up the incident resolving process.

Chapter 8

Conclusions

Over the past few years, telecommunication networks have evolved rapidly in terms of scalability and complexity, leading to a significant increase in the number of network elements. This has motivated that management teams have had to deal with massive amounts of monitoring data. This problem has itself stimulated them to think about methods and techniques that make them able to manage, monitor and follow up all these data rapidly and accurately. Consequently, the alerts correlation concept has emerged and it is considered as one of the promising methods that have been proposed to handle the huge amount of alerts generated by monitoring systems.

In this thesis, first, we have reviewed the research efforts carried out in the alerts correlation field, and provided a classification for them based on the type of application, the number of data sources, the correlation method used and the type of architecture. We have shown that it is possible to classify the alerts correlation techniques as belonging to different applications: NMS, network and system security and SCADA systems.

Afterwards, we have carefully studied the design architecture behind the existing alerts correlation techniques and found that most of them formulated the correlation process as a sequential process. We claim that this sequentiality does not describe the alerts correlation process well enough. Therefore, we have proposed a comprehensive alerts correlation model with feedback mechanisms that also considers all the sources of information that could be used in the alerts correlation process.

In addition, we have made a review of some available correlation tools, intended for three different applications: NMS, network and system security and SCADA systems. We have noticed that there is a big gap between the complexities behind the alerts correlation techniques suggested by the research community and those that are really implemented in the commercial or freely available tools.

Also, we have found that, despite the big amount of efforts having been done in the alerts correlation field, it is still an active area of research in both NMS and network and system security applications. Furthermore, we have noticed that there is not a clear agreement among researchers and vendors on how to formulate efficient solutions. Therefore, most of the existing solutions have their own limitations, such as the lack of standard benchmarks for evaluating and comparing them; architecture and scalability concerns, in which a large number of solutions are based on centralized architectures that have a limited scalability; and finally detection accuracy, in which the existing alerts correlation techniques are to be improved in this area.

Regarding ITSs, we have first reviewed the research efforts carried out in the tickets correlation field, and found out that there is a lack of works focusing in the tickets correlation process or in the process of the joint correlation of tickets and alerts.

Second, we have proposed a novel, simple, and effective approach to correlate and merge incident tickets. The approach is based on a generic model for the tickets that preserves and categorizes the relevant information and enables the comparison of their properties with relatively simple functions. No additional sources of information, apart from the tickets themselves and the information they contain, is required during the merging procedure. Despite its simplicity, the model has revealed to be useful for reducing the number of tickets generated and handled by ITS users, what is a desirable target in order to improve the workflow in an IT department of an enterprise or an IT service provider.

Third, the experiments carried out on two different datasets from a real IT management company have shown that, contrary to what we expected, there exists a significant amount of malformed, irrelevant and redundant tickets being generated by the different actors in the ITS. On the one hand, the results of the tickets preprocessing and reduction phases have shown that part of the original tickets is considered irrelevant from the point of view of incident solving. On the other hand, the proposed tickets correlation model –Chapter 5– can discover two levels of redundancy in creating tickets: local and global. For the local level, the proposed local correlation algorithm was able to discover a significant number of tickets as redundant tickets. The same is also valid for the global correlation level, in which the output of the global correlation algorithm has shown that there is another higher level of redundancy, *i.e.*, part of the local representative tickets is considered as redundant.

Fourth, due to the fact that the datasets tested here are not supervised, *i.e.*, no information about ground truth is present, we have developed our own mechanisms to validate the results that are mainly based on inspecting many correlation samples manually and proposing several temporal analysis measures.

Fifth, after applying the proposed validation methods, the results have shown that, first, the local correlation algorithm is able to correlate any subset of locally correlated tickets into a local representative one that correctly describes the incident with very high accuracy. The same is also valid for the global correlation algorithm efficiency, as it can correlate any subset of local representative tickets into a global representative one that correctly describes the incident with high accuracy. Regarding the samples validated as FPs, we have noticed that the management staff sometimes refers to previous solved tickets if the ongoing incident has mainly the same preliminary symptoms, despite the related tickets being really associated to the same incident or not.

Sixth, the proposed incident tickets correlation model can be easily incorporated in production systems, not allowing the generation of additional tickets when a related one is active, or automatically removing those detected as redundant. The first task is quite straightforward for the local correlation algorithm, that can be applied in real time at creation of the ticket. The global level uses information that could not be available at the time of the creation of the ticket but added later and, therefore, would imply a posterior filtering of the tickets.

Finally, as an additional use of the model and the proposed procedure, we have also described through a case study how it can be used as an assessment tool to provide some measures about the management staff's efficiency at the tickets creation process. This way, it would be possible to identify some deficiencies in the procedures, policies, or behaviors of the different actors involved in the ITS management process, enabling corrective actions to be taken and evaluated.

Regarding the joint correlation of tickets and alerts, we have checked that part of the potentialities for improving current ITSs can be attributed to the different nature of the information provided by some of the actors, *i.e.*, information gathered from automatically generated alerts vs. information from customers, that are complementary. We have demonstrated that the mixture of these sources of information produces a better alerts reduction rate and adds more semantic information in the process of correlating tickets with alerts as well. As an extension of this idea, we have proposed a model for the joint correlation of tickets and alerts, and shown that the correlation process in this type of systems is not straightforward and many challenges might be appeared that could lower the accuracy of the correlation results.

The most important challenge that we have faced during this process is that we are dealing with two databases, alerts and tickets, having different characteristics. On the one hand, tickets contain more semantic information about incidents than alerts, but, on the other hand, the temporal information they contain is less trustable than that provided by alerts. We have solved this issue by refining the output of the tickets correlation with the alerts correlation output, in order to adjust the timing information of the incidents pointed out by the tickets.

Another challenge is how to deal with border effects and the existence of consecutive incidents, in which alerts at the beginning or the end of an incident that are not assigned to it might be related. To solve this issue, a series of experiments were carried out to tune the two thresholds' values, FOD and BOD, that are suggested for this purpose. The results have shown that the accuracy of the algorithm mainly depends on FOD and changing the value of BOD does not have any effect on the correlation results. Thus, the value for FOD is the overriding factor that determines the accuracy of the proposed algorithm.

Finally, the experimental results have shown that the number of tickets and alerts correctly correlated by the algorithm increases when tickets from the SD group are added to those from the MS one. We have concluded from this interesting finding that incorporating ticketing information in the alerts correlation process will definitely help in reducing a higher percentage of related alerts, *i.e.*, having a significant increase in the alerts correlation rate.

Bibliography

[1] Benoit Claise and Ralf Wolter. Network Management: Accounting and Performance Strategies. Cisco Press, 2007. ISBN: 1587051982.

[2] Robiah Yusof, Siti Rahayu Selamat, and Shahrin Sahib. Intrusion alert correlation technique analysis for heterogeneous log. *International Journal of Computer Science and Network Security*, 8(9):132, 2008.

[3] Mohd Hashim and Siti Zaiton. Network intrusion alert correlation challenges and techniques. *Journal Teknologi Maklumat*, 20(2):12–36, 2008.

[4] SeyedAli Mirheidari, Sajjad Arshad, and Rasool Jalili. Alert correlation algorithms: A survey and taxonomy. *Lecture Notes in Computer Science*, volume 8300, pages 183-197, 2013.

[5] J. Graham and S.J. Lowe. The Penguin Dictionary of Telecommunications. Penguin books. Penguin Books, 1991. ISBN: 9780140512373.

[6] ISO/IEC Information Technology Task Force (ITTF). ISO/IEC 7498-4: Information Processing Systems - Open Systems Interconnection- Basic Reference Model -Part 4: Management Framework. 1989. Available at `https://www.iso.org/obp/ui/#iso:std:iso-iec:7498:-4:ed-1:v1:en`, [Online; Last accessed: 2015-02-17].

[7] Abeck S. Hegering, H.G. and B. Neumair. Integrated Management of Networked Systems: Concepts, Architectures and their Operational Application. The Morgan Kaufmann Series in Networking. Penguin Books, 1999. ISBN: 9781558605718.

[8] Alexander Clemm. Network Management Fundamentals. Cisco Press, 2006. ISBN: 1587201372.

[9] E. Marilly, O. Martinot, H. Papini, and D. Goderis. Service level agreements: a main challenge for next generation networks. In *Proc. of the 2nd European Conference on Universal Multiservice Networks (ECUMN)*, pages 297–304, 2002.

[10] Jens Schmitt and Lars Wolf. Quality of Service - An Overview. Technical Report TR-KOM-1997-01, Darmstadt University of Technology, 1997.

[11] Dinesh Chandra Verma. Principles of Computer Systems and Network Management. Springer Publishing Company, 1st edition, 2009. ISBN: 9780387890081.

[12] Lynda M. Applegate, Robert D. Austin, and Warren F. Mcfarlan. Corporate Information Strategy and Management: Text and Cases. McGraw-Hill/Irwin, 2005. ISBN: 0072947756.

[13] Mo Li and K. Sandrasegaran. Network management challenges for next generation networks. In *Proc. of the 30th Anniversary IEEE Conference on Local Computer Networks*, pages 593–598, 2005.

[14] Raouf Boutaba and Jin Xiao. Network management: State of the art. In *Proc. of the 17th IFIP World Computer Congress*, pages 127–146, 2002.

[15] A. Pras, J. Schoenwaelder, M. Burgess, O. Festor, G. Martinez Perez, R. Stadler, and B. Stiller. Key research challenges in network management. *IEEE communications magazine*, 45(10):104–110, 2007.

[16] G. Shields and Realtimepublishers.com. The Shortcut Guide to Network Management for the Mid-Market. Realtimepublishers.com, 2007. ISBN: 9781931491723.

[17] Divakara K. Udupa. TMN: Telecommunications Management Network. McGraw-Hill, Inc., 1st edition, 1999. ISBN: 0070658153.

[18] The Office of Government Commerce (OGC). IT Infrastructure Library (ITIL). Available at `http://www.itil-officialsite.com/`. [Online; Last accessed: 2015-02-17].

[19] The Office of Government Commerce (OGC). Service Operation, IT Infrastructure Library version 3 (ITIL v3). Technical report, The Stationary Office, 2007.

[20] Said El brak, Mohammed Bouhorma, and Anouar A. Boudhir. Network management architecture approaches designed for mobile ad hoc networks. *International Journal of Computer Applications*, 15(6):14–18, 2011.

[21] Aiko Pras. Network management architectures. 1995. PhD Thesis, University of Twente. ISBN: 1381-3617.

[22] J.D. Case, K. McCloghrie, M.T. Rose, and S. Waldbusser. Protocol operations for version 2 of the Simple Network Management Protocol (SNMPv2). RFC 1905, 1996. Available at `https://tools.ietf.org/html/rfc1905`, [Online; Last accessed: 2015-02-17].

[23] S. Waldbusser. Remote Network Monitoring Management Information Base (RMON). RFC 2819, 1995. Available at `https://tools.ietf.org/html/rfc2819`, [Online; Last accessed: 2015-02-17].

[24] ISO/IEC Information Technology Task Force (ITTF). Information Processing Systems- OSI, ISO standard 9596-1: Common Management Information Protocol, part 1: Specification. 1998. Available at `https://www.iso.org/obp/ui/#iso:std:iso-iec:9596:-1:ed-3:v1:en`, [Online; Last accessed: 2015-02-17].

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

[26] Alfonso Valdes and Keith Skinner. Probabilistic alert correlation. In *Proc. of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID'01)*, pages 54–68, 2001.

[27] R.D. Gardner and D.A. Harle. Methods and systems for alarm correlation. *In Proc. of the Global Telecommunications Conference (GLOBECOM '96)*, volume1, pages 136–140, 1996.

[28] HP Openview. Available at `http://www8.hp.com/us/en/software/enterprise-software.html`. [Online; Last accessed: 2015-02-17].

[29] Jacques-H. Bellec and M-Tahar Kechadi. Towards a formal model for the network alarm correlation problem. In *Proc. of the 6th International Conference on Simulation, Modelling and Optimization (SMO'06)*, pages 458–463, 2006.

[30] Gabi Dreo and Robert Volta. Using master tickets as a storage for problem-solving expertise. In *Proc. of the 4th IFIP/IEEE International Symposium on Integrated Network Management IV*, pages 328–340, 1995.

[31] L. Lewis and G. Dreo. Extending trouble ticket systems to fault diagnostics. *IEEE Network*, 7(6):44–51, 1993.

[32] D. Johnson. NOC internal integrated trouble ticket system functional specification wishlist. RFC 1297, 1992. Available at `http://www.ietf.org/rfc/rfc1297.txt`. [Online; Last accessed: 2015-02-17].

[33] Fabien Pouget and Marc Dacier. Alert correlation: Review of the state of the art. Technical Report EURECOM+1271, 2003. Available at `http://www.eurecom.fr/publication/1271`. [Online; Last accessed: 2015-02-17].

[34] Ma Steinder and Adarshpal S. Sethi. A survey of fault localization techniques in computer networks. *Science of Computer Programming*, 53(2):165–194, 2004.

[35] Urko Zurutuza and Roberto Uribeetxeberria. Intrusion detection alarm correlation: A survey. In *Proc. of the IADAT International Conference on Telecommunications and Computer Networks*, pages 1–3, 2004.

[36] Reza Sadoddin and Ali Ghorbani. Alert correlation survey: framework and techniques. In *Proc. of the International Conference on Privacy, Security and Trust (PST '06)*, pages 37:1–37:10, 2006.

[37] Tianning Zang, Xiaochun Yun, and Yongzheng Zhang. A survey of alert fusion techniques for security incident. In *Proc. of the 9th International Conference on Web-Age Information Management (WAIM '08)*, pages 475–481, 2008.

[38] H.T. Elshoush and I.M. Osman. Reducing false positives through fuzzy alert correlation in collaborative intelligent intrusion detection systems- a review. In *Proc. of the IEEE International Conference on Fuzzy Systems*, pages 1–8, 2010.

[39] Neminath Hubballi and Vinoth Suryanarayanan. False alarm minimization techniques in signature-based intrusion detection systems: A survey. *Computer Communications*, 49(0):1 – 17, 2014.

[40] Leau Yu Beng, Sureswaran Ramadass, Selvakumar Manickam, and Tan Soo Fun. A survey of intrusion alert correlation and its design considerations. *IETE Technical Review*, 31(3):233–240, 2014.

[41] StatSoft, Inc. electronic statistics textbook: Nonparametric statistics. 2013. Available at `http://www.statsoft.com/textbook/nonparametric-statistics/`. [Online; Last accessed: 2015-02-17].

[42] Martin Roesch. SNORT - lightweight intrusion detection for networks. In *Proc. of the 13th USENIX Conference on System Administration (LISA '99)*, pages 229–238, 1999.

[43] S. Staniford-chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagl, K. Levitt, C. Wee, R. Yip, and D. Zerkle. GrIDS - a Graph based Intrusion Detection System for large networks. In *Proc. of the 19th National Conference on Information Systems Security*, 1996.

[44] He Yan, Lee Breslau, Zihui Ge, Daniel Massey, Dan Pei, and Jennifer Yates. G-RCA: a generic root cause analysis platform for service quality management in large IP networks. *IEEE/ACM Transactions on Networking*, 20(6):1734–1747, 2012.

[45] Jinqiao Yu, Y. V. Ramana Reddy, Sentil Selliah, Srinivas Kankanahalli, Sumitra Reddy, and Vijayanand Bharadwaj. TRINETR: An intrusion detection alert management system. In *Proc. of the 21st IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'04)*, pages 235–240, 2004.

[46] Tobias Chyssler, Simin Nadjm-Tehrani, Stefan Burschka, and Kalle Burbeck. Alarm reduction and correlation in defence of IP networks. In *Proc. of the 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE '04)*, pages 229–234, 2004.

[47] Prasad Calyam, Manojprasadh Dhanapalan, Mukundan Sridharan, Ashok Krishnamurthy, and Rajiv Ramnath. Topology-aware correlated network anomaly event detection and diagnosis. *Journal of Network and Systems Management*, 22(2):208–234, 2014.

[48] perfSONAR. Available at `http://www.perfsonar.net/`. [Online; Last accessed: 2015-02-17].

[49] The MITRE corporation, common vulnerabilities and exposures. Available at `http://cve.mitre.org/`. [Online; Last accessed: 2015-02-17].

[50] Ron Gula. Correlating IDS alerts with vulnerability information. Technical Report 4, Tenable Network Security, 2011. Available at `http://www.tenable.com/whitepapers/correlating-ids-alerts-with-vulnerability-information/`. [Online; Last accessed: 2015-02-17].

[51] BUGTRAQ, security focus online. Available at `http://www.securityfocus.com/`. [Online; Last accessed: 2015-02-17].

[52] CERT Division. Available at `http://www.cert.org/`. [Online; Last accessed: 2015-02-17].

[53] Phillip A. Porras, Martin W. Fong, and Alfonso Valdes. A mission-impact-based approach to INFOSEC alarm correlation. In *Proc. of the 5th International Conference on Recent Advances in Intrusion Detection (RAID'02)*, pages 95–114, 2002.

[54] Rahul Potharaju, Navendu Jain, and Cristina Nita-Rotaru. Juggling the Jigsaw: Towards automated problem inference from network trouble tickets. In *Proc. of the 10th USENIX Conference on Networked Systems Design and Implementation (NSDI'13)*, pages 127–142, 2013.

[55] Raúl Costa, Nuno Cachulo, and Paulo Cortez. An intelligent alarm management system for large-scale telecommunication companies. In *Proc. of the 14th*

Portuguese Conference on Artificial Intelligence: Progress in Artificial Intelligence (EPIA '09), pages 386–399, 2009.

[56] Stanford University Protégé teaching website. Available at `http://protege.stanford.edu/`. [Online; Last accessed: 2015-02-17].

[57] Yongjian Yang Min Xiao and Zhanwei Du. Ontology based alarm correlation technology in TD-SCDMA network. *Journal of Computational Information Systems*, 9(3):933–940, 2013.

[58] Wan Li and ShengFeng Tian. Preprocessor of intrusion alerts correlation based on ontology. In *Proc. of the International Conference on Communications and Mobile Computing (CMC '09)*, volume 3, pages 460–464, 2009.

[59] The Intrusion Detection Message Exchange Format (IDMEF). RFC 7465. Available at `http://www.ietf.org/rfc/rfc4765.txt`. [Online; Last accessed: 2015-02-17].

[60] L. Coppolino, S. D'Antonio, M. Esposito, and L. Romano. Exploiting diversity and correlation to improve the performance of intrusion detection systems. In *Proc. of the International Conference on Network and Service Security (N2S'09)*, pages 1–5, 2009.

[61] Jidong Long and Daniel G. Schwartz. Case-oriented alert correlation. *WSEAS Transactions on Computers*, 7(3):98–112, 2008.

[62] Viliam Holub, Trevor Parsons, Patrick O'Sullivan, and John Murphy. Run-time correlation engine for system monitoring and testing. In *Proc. of the 6th International Conference on Autonomic Computing (ICAC '09)*, pages 43–44, 2009.

[63] Peyman Kabiri and Ali A. Ghorbani. A rule-based temporal alert correlation system. *International Journal of Network Security*, 5(1):66–72, 2007.

[64] Xin Zhuang, Debao Xiao, Xuejiao Liu, and Yugang Zhang. Applying data fusion in collaborative alerts correlation. In *Proc. of the International Symposium on Computer Science and Computational Technology (ISCSCT '08)*, pages 124–127, 2008.

[65] Jun Chang, Jiang Yu, and Yijian Pei. MS2IFS: A multiple source-based security information fusion system. In *Proc. of the International Conference on Communications and Intelligence Information Security (ICCIIS '10)*, pages 215–219, 2010.

[66] Jen-Wei Hu, Hui-Min Chen, Te-Lung Liu, Hui-Min Tseng, Daniel Lin, Chu-Sing Yang, and C. Eugene Yeh. Implementation of alarm correlation system

for hybrid networks based upon the perfSONAR framework. In *Proc. of the 24th IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA '10)*, pages 893–898, 2010.

[67] Boris Gruschke. Integrated event management: Event correlation using dependency graphs. In *Proc. of the 9th International Workshop on Distributed Systems Operation Management (DSOM'98)*, pages 130–141, 1998.

[68] Shmuel Kliger, Shaula Yemini, Yechiam Yemini, David Ohsie, and Salvatore Stolfo. A coding approach to event correlation. In *Proc. of the 4th IFIP/IEEE International Symposium on Integrated Network Management IV*, pages 266–277, 1995.

[69] Faeiz Alserhani, Monis Akhlaq, Irfan U. Awan, Andrea J. Cullen, and Pravin Mirchandani. MARS: Multi-stage attack recognition system. In *Proc. of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA '10)*, pages 753–759, 2010.

[70] Xinzhou Qin. A probabilistic-based framework for infosec alert correlation. 2005. PhD Thesis, Georgia Institute of Technology. ISBN: 0-542-24360-1.

[71] Fredrik Valeur, Giovanni Vigna, Christopher Kruegel, and Richard A. Kemmerer. A comprehensive approach to intrusion detection alert correlation. *IEEE Transactions on Dependable and Secure Computing*, 1(3):146–169, 2004.

[72] Yan Chen and J. Lee. Autonomous mining for alarm correlation patterns based on time-shift similarity clustering in manufacturing system. In *Proc. of the IEEE Conference on Prognostics and Health Management (PHM)*, pages 1–8, 2011.

[73] A. Carcano, A. Coletta, M. Guglielmi, M. Masera, I.N. Fovino, and A. Trombetta. A multidimensional critical state analysis for detecting intrusions in SCADA systems. *IEEE Transactions on Industrial Informatics*, 7(2):179–186, 2011.

[74] Naoum Sayegh, Imad H. Elhajj, Ayman Kayssi, and Ali Chehab. SCADA intrusion detection system based on temporal behavior of frequent patterns. In *Proc. of the 17th IEEE Mediterranean Electrotechnical Conference (MELECON)*, pages 432–438, 2014.

[75] Keunsoo Lee, Juhyun Kim, Ki Hoon Kwon, Younggoo Han, and Sehun Kim. DDoS attack detection method using cluster analysis. *Expert Systems and Applications*, 34(3):1659–1665, 2008.

[76] A. Siraj and R.B. Vaughn. Multi-level alert clustering for intrusion detection sensor data. In *Proc. of the Annual Meeting of the North American on Fuzzy Information Processing Society (NAFIPS)*, pages 748–753, 2005.

[77] Hervé Debar and Andreas Wespi. Aggregation and correlation of intrusion-detection alerts. In *Proc. of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID'01)*, pages 85–103, 2001.

[78] Klaus Julisch. Clustering intrusion detection alarms to support root cause analysis. *ACM Transactions on Information and System Security*, 6(4):443–471, 2003.

[79] Klaus Julisch and Marc Dacier. Mining intrusion detection alarms for actionable knowledge. In *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*, pages 366–375, 2002.

[80] F. Cuppens. Managing alerts in a multi-intrusion detection environment. In *Proc. of the 17th Annual Conference on Computer Security Applications (ACSAC '01)*, pages 22–31, 2001.

[81] G. Jakobson and M. Weissman. Real-time telecommunication network management: extending event correlation with temporal constraints. In *Proc. of the 4th International Symposium on Integrated Network Management*, pages 290–301, 1995.

[82] Jie Ma, Zhi-tang Li, and Wei-ming Li. Real-time alert stream clustering and correlation for discovering attack strategies. In *Proc. of the 5th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD '08)*, pages 379–384, 2008.

[83] Xinzhou Qin and Wenke Lee. Statistical causality analysis of INFOSEC alert data. *Lecture Notes in Computer Science*, volume 2820, pages 73-69, 2003.

[84] Benjamin Morin and Heré Debar. Correlation of intrusion symptoms: An application of chronicles. *Lecture Notes in Computer Science*, volume 2820, pages 94-112, 2003.

[85] A. Kelkar, U. Naiknaware, S. Sukhlecha, A. Sanadhya, M. Natu, and V. Sadaphal. Analytics-based solutions for improving alert management service for enterprise systems. In *Proc. of the IEEE 13th International Conference on Data Mining Workshops (ICDMW)*, pages 219–227, 2013.

[86] Bin Zhu and Ali A. Ghorbani. Alert correlation for extracting attack strategies. *International Journal of Network Security*, 3(3):244–258, 2006.

[87] Seyed Hossein Ahmadinejad and Saeed Jalili. Alert correlation using correlation probability estimation and time windows. In *Proc. of the International Conference on Computer Technology and Development (ICCTD '09)*, pages 170–175, 2009.

[88] M. Bateni and A. Baraani. Time window management for alert correlation using context information and classification. *International Journal of Computer Network Information Security*, 5(11):6–19, 2013.

[89] Lin Zhaowen, Li Shan, and Ma Yan. Real-time intrusion alert correlation system based on prerequisites and consequence. In *Proc. of the 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*, pages 1–5, 2010.

[90] Peng Ning, Yun Cui, Douglas S. Reeves, and Dingbang Xu. Techniques and tools for analyzing intrusion alerts. *ACM Transactions on Information and System Security*, 7(2):274–318, 2004.

[91] Shisong Xiao, Yugang Zhang, Xuejiao Liu, and Jingju Gao. Alert fusion based on cluster and correlation analysis. In *Proc. of the International Conference on Convergence and Hybrid Information Technology (ICHIT '08)*, pages 163–168, 2008.

[92] F. Alserhani. A framework for multi-stage attack detection. In *Proc. of the Saudi International Conference on Electronics, Communications and Photonics (SIECPC)*, pages 1–6, 2013.

[93] Sebastian Roschke, Feng Cheng, and Christoph Meinel. A new alert correlation algorithm based on attack graph. In *Proc. of the 4th International Conference on Computational Intelligence in Security for Information Systems (CISIS'11)*, pages 58–67, 2011.

[94] Lingyu Wang, Anyi Liu, and Sushil Jajodia. Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts. *Computer Communications*, 29(15):2917–2933, 2006.

[95] L. Li and F. Lifang. Fast fault localization for Internet services based on bipartite graph. *International Journal of Digital Content Technology and its Applications*, 5(7):8–16, 2011.

[96] Jian Wu, Ziyu Guan, Qing Zhang, Ambuj K. Singh, and Xifeng Yan. Static and dynamic structural correlations in graphs. *IEEE Transactions on Knowledge and Data Engineering*, 25(9):2147–2160, 2013.

[97] S.A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie. High speed and robust event correlation. *Communications Magazine, IEEE*, 34(5):82–90, 1996.

[98] D. Ourston, S. Matzner, W. Stump, and B. Hopkins. Applications of hidden Markov models to detecting multi-stage network attacks. In *Proc. of the 36th Annual Hawaii International Conference on System Sciences*, volume 9, pages 334–344, 2003.

[99] M. Farhadi, H. AmirHaeri and M. Khansari. Alert correlation and prediction using data mining and HMM. *ISeCure, The ISC International Journal of Information Security*, 3(2):77–102, 2011.

[100] Xin Zan, Feng Gao, Jiuqiang Han, and Yu Sun. A hidden Markov model based framework for tracking and predicting of attack intention. In *Proc. of the International Conference on Multimedia Information Networking and Security (MINES '09)*, volume 2, pages 498–501, 2009.

[101] Shi Zhicai and Xia Yongxiang. A novel hidden Markov model for detecting complicate network attacks. In *Proc. of the IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, pages 312–315, 2010.

[102] M. Steinder and A.S. Sethi. Probabilistic fault localization in communication systems using belief networks. *IEEE/ACM Transactions on Networking*, 12(5):809–822, 2004.

[103] M. Marchetti, M. Colajanni, and F. Manganiello. Identification of correlated network intrusion alerts. In *Proc. of the 3rd International Workshop on Cyberspace Safety and Security (CSS)*, pages 15–20, 2011.

[104] E. Harahap, W. Sakamoto, and H. Nishi. Failure prediction method for network management system by using Bayesian network and shared database. In *Proc. of the 8th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT)*, pages 1–6, 2010.

[105] Jing Zhou, Aihuang Guo, Branko Celler, and Steven Su. Fault detection and identification spanning multiple processes by integrating PCA with neural network. *Applied Soft Computing*, 14(0):4 – 11, 2014.

[106] S. Lagzian, F. Amiri, A. Enayati, and H. Gharaee. Frequent item set mining-based alert correlation for extracting multi-stage attack scenarios. In *Proc. of the 6th International Symposium on Telecommunications (IST)*, pages 1010–1014, 2012.

[107] Safaa O. Al-Mamory and Hongli Zhang. IDS alerts correlation using grammar-based approach. *Journal in Computer Virology*, 5(4):271–282, 2009.

[108] Khalid Alsubhi, Issam Aib, and Raouf Boutaba. FuzMet: a fuzzy-logic based alert prioritization engine for intrusion detection systems. *International Journal of Network Management*, 22(4):263–284, 2012.

[109] R.N. Cronk, P.H. Callahan, and L. Bernstein. Rule-based expert systems for network management and operations: an introduction. *Network, IEEE*, 2(5):7–21, 1988.

[110] Kar-Wing Edward Lor. A network diagnostic expert system for acculink multiplexers based on a general network diagnostic scheme. In *Proc. of the 3rd International Symposium on Integrated Network Management*, pages 659–669, 1993.

[111] G. Liu, A.K. Mok, and E. J. Yang. Composite events for network event correlation. In *Proc. of the 6th IFIP/IEEE International Symposium on Integrated Network Management, Distributed Management for the Networked Millennium*, pages 247–260, 1999.

[112] Frédéric Cuppens and Rodolphe Ortalo. LAMBDA: A language to model a database for detection of attacks. In *Proc. of the 3rd International Workshop on Recent Advances in Intrusion Detection (RAID '00)*, pages 197–216, 2000.

[113] Giovanni Vigna, S.T. Eckmann, and R.A. Kemmerer. The STAT tool suite. In *Proc. of the DARPA Conference on Information Survivability and Exposition (DISCEX '00)*, volume 2, pages 46–55, 2000.

[114] Steven T. Eckmann, Giovanni Vigna, and Richard A. Kemmerer. STATL: an attack language for state-based intrusion detection. *Journal of Computer Security*, 10(1-2):71–103, 2002.

[115] Lei Liu, Kangfeng Zheng, and Yixian Yang. An intrusion alert correlation approach based on finite automata. In *Proc. of the International Conference on Communications and Intelligence Information Security (ICCIIS)*, pages 80–83, 2010.

[116] S. Cheung, U. Lindqvist, and M.W. Fong. Modeling multistep cyber attacks for scenario recognition. In *Proc. of the DARPA Conference on Information Survivability and Exposition*, volume 1, pages 284–292, 2003.

[117] Rakesh Agrawal and Ramakrishnan Srikant. Mining sequential patterns. In *Proc. of the 11th International Conference on Data Engineering (ICDE '95)*, pages 3–14, 1995.

[118] Reuben Smith, Nathalie Japkowicz, Maxwell Dondo, and Peter Mason. Using unsupervised learning for network alert correlation. In *Proc. of the 21st Canadian Society Conference on Advances in Artificial Intelligence (AI'08)*, pages 308–319, 2008.

[119] Rajeshwar Katipally, Wade Gasior, Xiaohui Cui, and Li Yang. Multistage attack detection system for network administrators using data mining. In *Proc. of the 6th Annual Workshop on Cyber Security and Information Intelligence Research (CSIIRW '10)*, pages 51:1–51:4, 2010.

[120] Reza Sadoddin and Ali A. Ghorbani. Real-time alert correlation using stream data mining techniques. In *Proc. of the 20th National Conference on Innovative Applications of Artificial Intelligence (IAAI'08)*, pages 1731–1737, 2008.

[121] Tongyan Li and Xingming Li. Novel alarm correlation analysis system based on association rules mining in telecommunication networks. *Information Sciences*, 180(16):2960–2978, 2010.

[122] Jian Wu and Xingming Li. Communication network alarm correlation based on multi-dimensional fuzzy association rules mining. In *Proc. of the 2nd International Conference on Electric Information and Control Engineering (ICEICE '12)*, pages 439–443, 2012.

[123] Hou Sizu and Zhang Xianfei. Alarms association rules based on sequential pattern mining algorithm. In *Proc. of the 5th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD '08)*, volume 2, pages 556–560, 2008.

[124] A.A. Mohamed and O. Basir. Fusion based approach for distributed alarm correlation in computer networks. In *Proc. of the 2nd International Conference on Communication Software and Networks (ICCSN '10)*, pages 318–324, 2010.

[125] R. Khatoun, G. Doyen, D. Gaiti, R. Saad, and A. Serhrouchni. Decentralized alerts correlation approach for DDoS intrusion detection. In *Proc. of the International Conference on New Technologies, Mobility and Security (NTMS '08)*, pages 1–5, 2008.

[126] Donghai Tian, Hu Changzhen, Yang Qi, and Wang Jianqiao. Hierarchical distributed alert correlation model. In *Proc. of the 5th International Conference on Information Assurance and Security (IAS '09)*, volume 2, pages 765–768, 2009.

[127] Xinzhou Qin and Wenke Lee. Attack plan recognition and prediction using causal networks. In *Proc. of the 20th Annual Conference on Computer Security Applications*, pages 370–379, 2004.

[128] Steven J. Templeton and Karl Levitt. A requires/provides model for computer attacks. In *Proc. of the International workshop on New security paradigms (NSPW '00)*, pages 31–38, 2000.

[129] M. Hasan, B. Sugla, and Ramesh Viswanathan. A conceptual framework for network management event correlation and filtering systems. In *Proc. of the 6th IFIP/IEEE International Symposium on Integrated Network Management*, pages 233–246, 1999.

[130] Sean Meyn and Richard L. Tweedie. Markov Chains and Stochastic Stability. Cambridge University Press, 2nd edition, 2009. ISBN: 0521731828.

[131] L. Rabiner and B. Juang. An introduction to hidden Markov models. *IEEE Signal Processing Magazine*, 3(1):4–16, 2003.

[132] Richard E. Neapolitan. Learning Bayesian Networks. Prentice-Hall, Inc., 2003. ISBN: 0130125342.

[133] Simon Haykin. Neural Networks: A Comprehensive Foundation. Prentice Hall PTR, 2nd edition, 1998. ISBN: 0132733501.

[134] Wu Jian and Li Xing ming. A dynamic mining algorithm of association rules for alarm correlation in communication networks. In *Proc. of the 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE)*, pages 799–802, 2008.

[135] Kenji Yamanishi and Yuko Maruyama. Dynamic syslog mining for network failure monitoring. In *Proc. of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD '05)*, pages 499–508, 2005.

[136] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proc. of the 20th International Conference on Very Large Data Bases (VLDB '94)*, pages 487–499, 1994.

[137] O. Abouabdalla, H. El-Taj, A. Manasrah, and S. Ramadass. False positive reduction in intrusion detection system: A survey. In *Proc. of the 2nd IEEE International Conference on Broadband Network Multimedia Technology (IC-BNMT '09)*, pages 463–466, 2009.

[138] APAN Tokyo XP network operations center. Available at `http://www.jp.apan.net/NOC/`. [Online; Last accessed: 2015-02-17].

[139] SWITCH, Swiss Networking Academy. Available at `http://www.switch.ch/network/operations/trouble-tickets/`. [Online; Last accessed: 2015-02-17].

[140] Jonathan J. Davis and Andrew J. Clark. Data preprocessing for anomaly based network intrusion detection: A review. *Computers Security*, 30(6-7):353–375, 2011.

[141] International Business Machines Corporation. Understanding common base events specification v1.0.1. Available at `http://www.ibm.com/developerworks/autonomic/books/fpy0mst.htm#HDRAPPA`. [Online; Last accessed: 2015-02-17].

[142] David Mills, Jim Martin, Jack Burbank, and William Kasch. Network Time Protocol Version 4: Protocol and Algorithms Specification. RFC 5905, 2010. Available at `http://www.rfc-editor.org/rfc/rfc5905.txt`. [Online; Last accessed: 2015-02-17].

[143] Roberto Perdisci, Giorgio Giacinto, and Fabio Roli. Alarm clustering for intrusion detection systems in computer networks. *Journal of Engineering Applications of Artificial Intelligence*, 19(4):429–438, 2006.

[144] Heng-Sheng Lin, Hsing-Kuo Pao, Ching-Hao Mao, Hahn-Ming Lee, Tsuhan Chen, and Yuh-Jye Lee. Adaptive alarm filtering by causal correlation consideration in intrusion detection. In *Proc. of the International Symposium on Intelligent Decision Technologies*, pages 437–447, 2009.

[145] Loai Zomlot, Sathya Chandran Sundaramurthy, Kui Luo, Xinming Ou, and S. Raj Rajagopalan. Prioritizing intrusion analysis using Dempster-Shafer theory. In *Proc. of the 4th ACM Workshop on Security and Artificial Intelligence (AISec '11)*, pages 59–70, 2011.

[146] G. Shafer. A Mathematical Theory of Evidence. Limited paperback editions. Princeton University Press, 1976. ISBN: 9780691100425.

[147] K. Alsubhi, E. Al-Shaer, and R. Boutaba. Alert prioritization in intrusion detection systems. In *Proc. of the IEEE Conference on Network Operations and Management Symposium (NOMS '08)*, pages 33–40, 2008.

[148] S. Wallin, V. Leijon, and L. Landén. Statistical analysis and prioritisation of alarms in mobile networks. *International Journal of Business Intelligence and Data Mining*, 4(1):4–21, 2009.

[149] Guofei Jiang, Haifeng Chen, Kenji Yoshihira, and Akhilesh Saxena. Ranking the importance of alerts for problem determination in large computer systems (ICAC '09). In *Proc. of the 6th International Conference on Autonomic computing*, pages 3–12, 2009.

[150] Gabrijela Dreo. A framework for supporting fault diagnosis in integrated network and systems management: Methodologies for the correlation of trouble tickets and access to problem-solving expertise. PhD Thesis, University of Munich, 1995.

[151] Qihong Shao, Yi Chen, Shu Tao, Xifeng Yan, and Nikos Anerousis. Efficient ticket routing by resolution sequence mining. In *Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*, pages 605–613, 2008.

[152] Gengxin Miao, Louise E. Moser, Xifeng Yan, Shu Tao, Yi Chen, and Nikos Anerousis. Generative models for ticket resolution in expert networks. In *Proc. of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*, pages 733–742, 2010.

[153] Liang Tang, Tao Li, L. Shwartz, and G.Ya. Grabarnik. Identifying missed monitoring alerts based on unstructured incident tickets. In *Proc. of the 9th International Conference on Network and Service Management (CNSM)*, pages 143–146, 2013.

[154] Ying Li and Ta-Hsin Li. A method of effort estimation for incident tickets in IT services. In *Proc. of the IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, pages 311–316, 2013.

[155] P. Marcu, G. Grabarnik, L. Luan, D. Rosu, L. Shwartz, and C. Ward. Towards an optimized model of incident ticket correlation. In *Proc. of the IFIP/IEEE International Symposium on Integrated Network Management (IM '09)*, pages 569–576, 2009.

[156] Andreas Hanemann. Automated IT Service Fault Diagnosis Based on Event Correlation Techniques. PhD Thesis, University of Munich, 2007.

[157] A. Hanemann and P. Marcu. Algorithm design and application of service-oriented event correlation. In *Proc. of the 3rd IEEE/IFIP International Workshop on Business-driven IT Management (BDIM)*, pages 61–70, 2008.

[158] Liang Tang, Tao Li, Larisa Shwartz, Florian Pinel, and Genady Ya Grabarnik. An integrated framework for optimizing automatic monitoring systems in large IT infrastructures. In *Proc. of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '13)*, pages 1249–1257, 2013.

[159] A. Medem, R. Teixeira, N. Feamster, and M. Meulle. Joint analysis of network incidents and intradomain routing changes. In *Proc. of the International Conference on Network and Service Management (CNSM)*, pages 198–205, 2010.

[160] A. Medem, R. Teixeira, and N. Usunier. Predicting critical intradomain routing events. In *Proc. of the IEEE Global Telecommunications Conference (GLOBECOM '10)*, pages 1–5, 2010.

[161] Nick Feamster and Hari Balakrishnan. Detecting BGP configuration faults with static analysis. In *Proc. of the 2nd Conference on Symposium on Networked Systems Design and Implementation (NSDI'05)*, pages 43–56, 2005.

[162] Jeffrey C. Mogul Stefan Savage Daniel Turner, Kirill Levchenko and Alex C. Snoereni. On failure in managed enterprise networks. Technical report, HP Laboratories , HPL-2012-10, 2012. Available at `http://www.hpl.hp.com/techreports/2012/HPL-2012-101.pdf`, [Online; Last accessed: 2015-02-17].

[163] A. Medem, M.-I. Akodjenou, and R. Teixeira. Troubleminer: Mining network trouble tickets. In *Proc. of the IFIP/IEEE International Symposium on Integrated Network Management-Workshops (IM '09)*, pages 113–119, 2009.

[164] J. Tanaka. Analysis of trouble tickets issued by APAN JP NOC. 2004. Available at `http://www.jp.apan.net/meetings/busan03/noc/`. [Online; Last accessed: 2015-02-17].

[165] Zsolt Pándi. Analysis of public trouble ticket data. Technical report, Budapest University of Technology and Economics, Department of Telecommunications, 2005.

[166] Yiyi Huang, Nick Feamster, Anukool Lakhina, and Jim (Jun) Xu. Diagnosing network disruptions with network-wide analysis. In *Proc. of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '07)*, pages 61–72, 2007.

[167] C. Labovitz, A. Ahuja, and F. Jahanian. Experimental study of Internet stability and backbone failures. In *Proc. of the 29th International Symposium on Fault-Tolerant Computing*, pages 278–285, 1999.

[168] ServiceNow, Inc. Available at `http://www.servicenow.com/products/it-service-automation-applications/incident-management.html`. [Online; Last accessed: 2015-02-17].

[169] Remedy, BMC. Available at `http://www.bmc.com/it-solutions/remedy-itsm.html/`. [Online; Last accessed: 2015-02-17].

[170] Tivoli, IBM. Available at `http://www-01.ibm.com/software/tivoli/`. [Online; Last accessed: 2015-02-17].

[171] Daniel Turner, Kirill Levchenko, Alex C. Snoeren, and Stefan Savage. California fault lines: Understanding the causes and impact of network failures. *SIGCOMM Computer Communication Review*, 40(4):315–326, 2010.