



Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Dirección General de Estudios de Posgrado
Facultad de Ingeniería Electrónica y Eléctrica
Unidad de Posgrado

**Desarrollo de un gestor de notificaciones multiagente
para la reducción de los tiempos de diagnóstico y
mitigación de fallos en redes IP**

TESIS

Para optar el Grado Académico de Magíster en Ingeniería de
Tecnologías de la Información y Telemática

AUTOR

Juan Carlos ZEGARRA VÁSQUEZ

ASESOR

Néstor Adolfo MAMANI MACEDO

Lima, Perú

2020



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

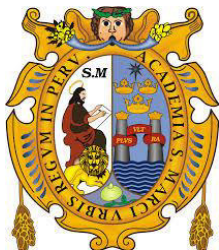
Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

Referencia bibliográfica

Zegarra, J. (2020). Desarrollo de un gestor de notificaciones multiagente para la reducción de los tiempos de diagnóstico y mitigación de fallos en redes IP. [Tesis de maestría, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería Electrónica y Eléctrica, Unidad de Posgrado]. Repositorio institucional Cybertesis UNMSM.

HOJA DE METADATOS COMPLEMENTARIOS

Código ORCID del autor	https://orcid.org/0000-0002-4176-9356
DNI o pasaporte del autor	09822790
Código ORCID del asesor	https://orcid.org/0000-0001-8406-0235
DNI o pasaporte del asesor	25632333
Grupo de investigación	_____
Agencia financiadora	_____
Ubicación geográfica donde se desarrolló la investigación	Lugar: Chorrillos - Lima -12.19816 -77.00079
Disciplinas OCDE	Ingeniería eléctrica, Ingeniería electrónica https://purl.org/pe-repo/ocde/ford#2.02.01



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
(Universidad del Perú, DECANA DE AMÉRICA)
FACULTAD DE INGENIERÍA ELECTRÓNICA Y ELÉCTRICA

UNIDAD DE POSGRADO

Calle Germán Amezaga N.º 375 Lima (Perú)
Teléfono (51 – 1) 6197000 Anexo 4204
Correo: postfie@unmsm.edu.pe



«AÑO DE LA UNIVERSALIZACIÓN DE LA SALUD»

**ACTA DE SUSTENTACIÓN DE TESIS PARA OPTAR EL GRADO ACADÉMICO
DE MAGÍSTER EN INGENIERÍA DE TECNOLOGÍAS DE LA INFORMACIÓN Y
TELEMÁTICA**

Siendo las 15:30 horas. del 21 de noviembre de 2020, los suscritos miembros del jurado reunidos en el salón de Grados de la Facultad de Ingeniería Electrónica y Eléctrica, el Jurado Examinador presidido por el Dr. Hugo Avila Vargas, Mg. Jesús Otto Villanueva Napurí, Mg. Cayo Víctor León Fernández, Mg. Juan Gamarra Moreno y el Mg. Néstor Adolfo Mamani Macedo.

*Se reunió para la sustentación oral y pública de la Tesis para optar el Grado Académico de Magíster en Ingeniería de Tecnologías de la Información y Telemática, que solicitó el alumno **Juan Carlos Zegarra Vásquez** con código N° 10197020, el cual procedió hacer la exposición oral y pública de su Tesis Titulada "**DESARROLLO DE UN GESTOR DE NOTIFICACIONES MULTIAGENTE PARA LA REDUCCIÓN DE LOS TIEMPOS DE DIAGNÓSTICO Y MITIGACIÓN DE FALLOS EN REDES IP**"*

Concluida la exposición, el Jurado Examinador procedió a formular las preguntas reglamentarias y, luego de una deliberación en privado, decidió otorgarle la siguiente calificación:

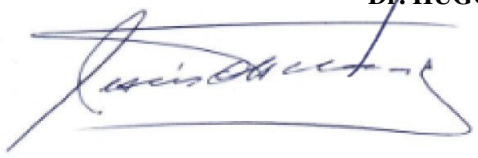
EXCELENTE	19	DIECINUEVE
NÚMERO		LETRAS

*A continuación, el Presidente Jurado recomienda que la Unidad de Posgrado proceda con el trámite correspondiente para que se otorgue el Grado Académico de Magíster en Ingeniería de Tecnologías de la Información y Telemática al alumno **Juan Carlos Zegarra Vásquez**.*

Siendo las 17:20 horas. se levantó la Sesión, recibiendo el graduado las felicitaciones de los señores miembros del Jurado y público asistente.



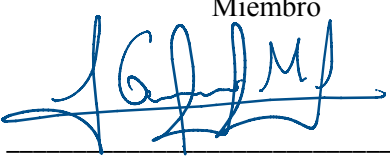
Dr. HUGO AVILA VARGAS
Presidente



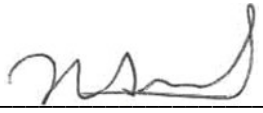
Mg. JESÚS OTTO VILLANUEVA NAPURÍ
Miembro



Mg. CAYO VÍCTOR LEÓN FERNÁNDEZ
Miembro



Mg. JUAN GAMARRA MORENO
Miembro



Mg. NÉSTOR ADOLFO MAMANI MACEDO
Asesor

INDICE GENERAL

LISTA DE FIGURAS	VI
LISTA DE TABLAS	XII
RESUMEN	XIII
ABSTRACT	XIV
CAPÍTULO I	1
1. INTRODUCCIÓN	1
1.1. Situación problemática	1
1.2. Formulación del problema	3
1.3. Alcance del trabajo de investigación a desarrollar	3
1.4. Justificación teórica	4
1.5. Justificación práctica	5
1.6. Objetivos de la investigación	5
1.6.1. Objetivos generales	5
1.6.2. Objetivos específicos	5
1.7. Hipótesis	6
1.7.1. Hipótesis general	6
1.7.2. Hipótesis específicas	6
1.8. Aspectos metodológicos	7
1.8.1. Metodología a emplear	7
1.8.2. Ambiente de estudio	7
1.8.3. Análisis e interpretación de la información	8
CAPÍTULO II	9
2. MARCO TEÓRICO	9
2.1. Marco filosófico o epistemológico de la investigación	9
2.1.1. El nacimiento de la Inteligencia Artificial	9
2.1.2. La Inteligencia artificial distribuida	10
2.1.3. Problemas de la inteligencia artificial distribuida	11
2.2. Antecedentes de investigación	11
2.2.1. Un sistema de monitoreo de red basada en múltiples agentes	12
2.2.2. Gestión de redes basada en arquitectura multiagente	16
2.2.3. Gestión de red usando sistemas multiagente	17
2.2.4. Conclusiones de los antecedentes	20
2.3. Bases teóricas	21
2.3.1. Aspectos del modelo TCP/IP	21
2.3.2. Aspectos del protocolo SYSLOG	25
2.3.3. Aspectos de gestión de red	28
2.3.4. Noción de agentes	31
2.3.5. Metodología Ingenias para el desarrollo de software orientado a agentes	44

2.3.6.	El Ciclo de Vida	56
2.3.7.	Ingenias Development Kit	58
2.3.8.	Framework para sistemas multiagente (JADE)	59
2.3.9.	ESPER	68
2.3.10.	EXPECT	76
2.3.11.	GNS3	79

CAPÍTULO III 81

3.	METODOLOGÍA	81
3.1.	Introducción	81
3.2.	Análisis y diseño del sistema	82
3.2.1.	Descripción general del sistema	82
3.2.2.	Análisis del softbot gestor de notificaciones	89
3.2.3.	Etapas de diseño del SMA	105
3.3.	Ontología	119

CAPÍTULO IV 121

4.	RESULTADOS Y DISCUSIÓN	121
4.1.	Metodología de pruebas del producto a desarrollar	121
4.1.1.	Unidades de prueba	121
4.1.2.	Población para pruebas	121
4.1.3.	Tamaño de la muestra para pruebas finales	121
4.1.4.	Variables	122
4.1.5.	Técnica de recolección de datos para pruebas	123
4.1.6.	Análisis e interpretación de la información resultante de las pruebas	123
4.2.	Etapas previas antes de la ejecución de las pruebas	124
4.2.1.	Entorno de pruebas	124
4.2.2.	Verificación de la plataforma de desarrollo	127
4.3.	Prueba de los agentes desarrollados en el softbot gestor de notificaciones	130
4.3.1.	El “Agente Recolector”	130
4.3.2.	El “Agente Experto”	133
4.3.3.	El “Agente Gráfico”	138
4.4.	Pruebas de hipótesis y presentación de los resultados	139
4.4.1.	Detección y procesamiento de eventos simples a través de “listeners”	139
4.4.2.	Detección y procesamiento de eventos correlacionados a través de “suscribers”	161
4.5.	Demostración de la hipótesis	166
4.5.1.	Demostración 1.	166
4.5.2.	Demostración 2	168
4.6.	Resumen y discusión final	170

CONCLUSIONES 173

RECOMENDACIONES 175

REFERENCIAS BIBLIOGRÁFICAS 176

ANEXO A	182
ANEXO B	184
ANEXO C	185

LISTA DE FIGURAS

- Fig. 2.1 Arquitectura propuesta del sistema basada en múltiples agentes
- Fig. 2.2 Arquitectura multiagente
- Fig. 2.3 Diagrama de colaboración de los agentes
- Fig. 2.4 Las cuatro capas de la suite de protocolos TCP/IP
- Fig. 2.5 Interacción entre el “originator” y el “collector”
- Fig. 2.6 Ejemplo de mensaje SYSLOG
- Fig. 2.7 Flujo de información de gestión
- Fig. 2.8 Bucle de información de gestión
- Fig. 2.9 Arquitectura abstracta FIPA
- Fig. 2.10 Ejemplo de entrada de directorio
- Fig. 2.11 Plataforma de Agentes FIPA
- Fig. 2.12 Relaciones entre los meta modelos
- Fig. 2.13 Tres vistas de una organización
- Fig. 2.14 Descripción estructural
- Fig. 2.15 Descripción social
- Fig. 2.16 Descripción funcional
- Fig. 2.17 Comportamiento del agente
- Fig. 2.18 Elementos del meta modelo de agente
- Fig. 2.19 Elementos en la definición de tareas
- Fig. 2.20 Relaciones entre tareas y objetivos
- Fig. 2.21 Relación entre los diferentes aspectos de la interacción

Fig. 2.22 Ejemplo de meta-modelado realizado con IDK

Fig. 2.23 Plataforma JADE y Agentes

Fig. 2.24 Ejemplo del front end del framework JADE

Fig. 2.25 Esquema de distribución de containers y plataformas

Fig. 2.26 Eventos de tipo W (1 a 6) que llegan en el tiempo

Fig. 2.27 Eventos de tipo W que arriban en una longitud de ventana 5

Fig. 2.28 Eventos de tipo W que arriban en una longitud de ventana 5, donde son truncados los eventos con temperatura bajo cero

Fig. 2.29 Eventos que no activan “listeners”

Fig. 2.30 Eventos de temperatura recogidos en los últimos cuatro segundos en la ventana de datos

Fig. 2.31 Eventos de temperatura recogidos y mantenidos en los últimos cuatro segundos en la ventana de datos

Fig. 2.32 Un ejemplo del uso de Expect

Fig. 3.1 Visión a alto nivel del trabajo experimental propuesto

Fig. 3.2 Arquitectura a alto nivel del softbot gestor de notificaciones

Fig. 3.3 Entorno de emulación de la red WAN

Fig. 3.4 Casos de uso del “softbot gestor de notificaciones”

Fig. 3.5 Meta-modelo de organización del “softbot gestor de notificaciones”

Fig. 3.6 Metamodelo del Agente: “Recolector de Eventos”

Fig. 3.7 Metamodelo de Objetivos y Tareas: TLeerSyslog

Fig. 3.8 Metamodelo de Objetivos y Tareas: TClasificarYFiltrar

Fig. 3.9 Metamodelo de Objetivos y Tareas: TTransformar

- Fig. 3.10 Metamodelo de Objetivos y Tareas: TAlmacenarSyslog
- Fig. 3.11 Interacción (I). Envío de eventos filtrados del agente recolector al agente clasificador
- Fig. 3.12 Interacción (II). Envío de eventos filtrados del agente recolector al agente clasificador
- Fig. 3.13 Metamodelo del Agente: Experto
- Fig. 3.14 Metamodelo de Objetivos y Tareas: TRazonarCEP
- Fig. 3.15 Metamodelo de Objetivos y Tareas: TDetectarFlapping
- Fig. 3.16 Metamodelo del Agente: Gráfico
- Fig. 3.17 Metamodelo de Objetivos y Tareas: TMuestraSyslog
- Fig. 3.18 Metamodelo de Objetivos y Tareas: TMuestraRecomendaciones
- Fig. 3.19 Metamodelo de Objetivos y Tareas: TMuestraAcciones
- Fig. 4.1 Imagen de la red y el entorno de pruebas
- Fig. 4.2 Configuración OSPF y SYSLOG del enrutador “HUANCAYO”
- Fig. 4.3 Redes que conoce el enrutador “HUANCAYO”
- Fig. 4.4 Configuración “bridge” entre red emulada y red física
- Fig. 4.5 Tabla de enrutamiento del equipo de gestión
- Fig. 4.6 Librerías externas empleadas por el IDE de desarrollo del softbot multiagente
- Fig. 4.7 Registro del “Agente_Recolector” en el framework Jade
- Fig. 4.8 Eventos registrados por el “Agente_Recolector”
- Fig. 4.9 Bloque de código que realiza filtrado de eventos a través de expresiones regulares
- Fig. 4.10 Bloque de código que muestra la comunicación que realiza el

“Agente_Recolector” con el “Agente_Experto”

Fig. 4.11 Registro del “Agente_Experto” en el framework Jade

Fig. 4.12 Lista parcial de las clases “listeners” y código de arranque del motor de razonamiento

Fig. 4.13 Bloque de código que apaga la interface en flapping

Fig. 4.14 Registro del “Agente_Grafico” en el framework Jade

Fig. 4.15 Eventos de prueba enviados al softbot gestor de notificaciones

Fig. 4.16 La notificación “LINK-3-UPDOWN”

Fig. 4.17 Diagnóstico y recomendación al detectar la notificación “LINK-3-UPDOWN”

Fig. 4.18 La notificación “OSPF-4-ERRRCV”

Fig. 4.19 Diagnóstico y recomendación al detectar la notificación “OSPF-4-ERRRCV”

Fig. 4.20 La notificación “CDP-4-NATIVE_VLAN_MISMATCH”

Fig. 4.21 Diagnóstico y recomendación al detectar la notificación “CDP-4-NATIVE_VLAN_MISMATCH”

Fig. 4.22 La notificación “SFF8472-5-THRESHOLD_VIOLATION”

Fig. 4.23 Diagnóstico y recomendación al detectar la notificación “SFF8472-5-THRESHOLD_VIOLATION”

Fig. 4.24 La notificación “OSPF-4-CONFLICTING_LSAID”

Fig. 4.25 Diagnóstico y recomendación al detectar la notificación “OSPF-4-CONFLICTING_LSAID”

Fig. 4.26 La notificación “PLATFORMASM-0-FAULT RECOVERY_ERR”

Fig. 4.27 Diagnóstico y recomendación al detectar la notificación “OSPF-4-CONFLICTING_LSAID”

- Fig. 4.28 La notificación "PLATFORM-ASM-3-MALLOC_FAILED"
- Fig. 4.29 Diagnóstico y recomendación al detectar la notificación "PLATFORM-ASM-3-MALLOC_FAILED"
- Fig. 4.30 La notificación "PLATFORMENVMON-0-CARD OVERTEMP SHUTDOWN"
- Fig. 4.31 Diagnóstico y recomendación al detectar la notificación "PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN"
- Fig. 4.32 La notificación "PLATFORMENVMON-2-CHASSIS OVERTEMP"
- Fig. 4.33 Diagnóstico y recomendación al detectar la notificación "PLATFORMENVMON-2-CHASSIS_OVERTEMP"
- Fig. 4.34 La notificación "PLATFORM-ENVMON-2-FAN_FAIL"
- Fig. 4.35 Diagnóstico y recomendación al detectar la notificación "PLATFORM-ENVMON-2-FAN_FAIL"
- Fig. 4.36 La notificación "PLATFORM-ENVMON-2-FAN"
- Fig. 4.37 Diagnóstico y recomendación al detectar la notificación "PLATFORM-ENVMON-2-FAN"
- Fig. 4.38 La notificación "OSPF-5-ADJCHG"
- Fig. 4.39 Diagnóstico y recomendación al detectar la notificación "OSPF-5-ADJCHG"
- Fig. 4.40 La notificación "LINK-3-UPDOWN"
- Fig. 4.41 Diagnóstico y recomendación al detectar la notificación "LINK-3-UPDOWN"
- Fig. 4.42 La notificación "LINEPROTO-5-UPDOWN"
- Fig. 4.43 Diagnóstico y recomendación al detectar la notificación "LINEPROTO-5-UPDOWN"

Fig. 4.44 La notificación “LINK-3-LINK_FAULT”

Fig. 4.45 Diagnóstico y recomendación al detectar la notificación “LINK-3-LINK_FAULT”

Fig. 4.46 La notificación “LINK-3-FCS_ERROR”

Fig. 4.47 Diagnóstico y recomendación al detectar la notificación “LINK-3-FCS_ERROR”

Fig. 4.48 La notificación “LINK-3-TOOSMALL”

Fig. 4.49 Diagnóstico y recomendación al detectar la notificación “LINK-3-TOOSMALL”

Fig. 4.50 Acceso a enrutador “Arequipa”

Fig. 4.51 Apagado de interfaz F0/1 del enrutador “Arequipa”

Fig. 4.52 Registro de las notificaciones recibidas por el softbot

Fig. 4.53 Diagnóstico y recomendación ante apagado de interfaz de red de enrutador “Arequipa” a nivel de capa 1

Fig. 4.54 Diagnóstico y recomendación ante apagado de interfaz de red de enrutador “Arequipa” a nivel de capa 2

Fig. 4.55 Diagnóstico y recomendación ante apagado de interfaz de red de enrutador “Arequipa” a nivel de capa 3

Fig. 4.56 Flapping en la interfaz “FastEthernet 1/0” del enrutador “Arequipa”

Fig. 4.57 Alerta generada por el softbot cuando correlaciona “flapping”

Fig. 4.58 Notificación de la acción “apagado de interfaz” generada por el softbot cuando correlaciona “flapping”

Fig. 4.59 Generación de conocimiento a partir de datos

Fig. 4.60 Disminución el tiempo empleado en el proceso de mitigación de fallos de inestabilidad en la red (flapping OSPF)

LISTA DE TABLAS

Tabla 2.1	Elementos del mensaje SYSLOG
Tabla 2.2	Palabras clave de los niveles de gravedad
Tabla 2.3	Asociaciones entre los elementos RUP y entidades MAS GRASIA
Tabla 2.4	Adecuación de las etapas de RUP a los meta-modelos
Tabla 3.1	Características del computador para el desarrollo del softbot
Tabla 3.2	Características del sistema operativo para desarrollo del softbot
Tabla 3.3	Datos de las interfaces de gestión de los enrutadores y redes
Tabla 3.4	Información genérica de los enrutadores de la red gestionada
Tabla 3.5	Procesos de gestión y actividades del softbot
Tabla 3.6	Requisitos de la gestión de recolección de eventos
Tabla 3.7	Requisitos de la gestión de experto o CEP
Tabla 3.8	Especificación detallada del razonamiento automático en base a los eventos previamente clasificados
Tabla 3.9	Requisitos de la gestión de ejecución de solución
Tabla 3.10	Requisitos de la gestión de ejecución de solución
Tabla 3.11	Definición cualitativa del “Agente Recolector”
Tabla 3.12	Definición cualitativa del “Agente Experto”
Tabla 3.13	Definición cualitativa del “Agente Gráfico”
Tabla 3.14	Atributos del concepto LOG_FILTRADO
Tabla 3.15	Diálogo entre el agente recolector y el agente experto
Tabla 4.1	“Listeners” definidos mediante el lenguaje “ESPER”
Tabla 4.2	Definición de los “subscribers” mediante el lenguaje “ESPER”

RESUMEN

En la gestión de redes, la recolección y clasificación de eventos son tareas críticas, porque permiten conocer en todo momento lo que está sucediendo en la red, además ante eventos inesperados, como fallas de hardware y software, las notificaciones recolectadas suelen ser de vital importancia para realizar el diagnóstico adecuado de la falla ocurrida, y proponer y aplicar soluciones eficientes a los problemas presentados. En las redes, el tiempo que tarda un operador en diagnosticar y mitigar un problema en base a la cantidad de notificaciones recolectadas es muy grande, lo que ocasiona que, ante un fallo importante, el tiempo de disponibilidad de la red sea menor, perjudicando el servicio hacia los clientes. El problema de la mejora del tiempo de diagnóstico y mitigación de problemas de la red es tratado desde la perspectiva de agentes de software a través del trabajo denominado “DESARROLLO DE UN GESTOR DE NOTIFICACIONES MULTIAGENTE PARA LA REDUCCIÓN DE LOS TIEMPOS DE DIAGNÓSTICO Y MITIGACIÓN DE FALLOS EN REDES IP”, el cual, genera conocimiento útil y acción de mitigación ante un fallo la red. Este sistema multiagente fue desarrollado bajo el abordaje de agentes de software, utilizándose para ello la metodología INGENIAS para el análisis y diseño del prototipo, como herramienta de modelado el INGENIAS Development Kit, éste último es un entorno de desarrollo integrado para el análisis, diseño e implementación de sistemas multiagente y como motor de razonamiento a la biblioteca ESPER, el cual, realiza el procesamiento complejo de eventos, encontrando e identificando patrones de fallos comunes en la red. El sistema multiagente desarrollado tiene como característica el ser proactivo, para lo cual, está permanentemente “escuchando” las notificaciones provenientes de una red de enrutadores IP e interactuando con ella para disminuir el tiempo de diagnóstico ante fallos en la red, mitigar problemas inestabilidad en la red y proponer soluciones ante fallos en la red.

Palabras clave: Sistemas multiagente, inteligencia artificial, gestión de redes, procesamiento complejo de eventos, modelado, enrutador IP.

ABSTRACT

In network management, the collection and classification of events is a critical task, they are critical tasks, because they allow knowing at all times what is happening in the network, in addition to unexpected events, such as hardware and software failures, the notifications collected are usually be of vital importance to make the proper diagnosis of the failure that occurred, and thus, propose and apply efficient solutions to the problems presented. In networks, the time it takes for an operator to diagnose and mitigate a problem based on the number of notifications collected is very large, which means that, in the event of a major failure, the network uptime is shorter, damaging the customer service. The problem of improving the time of diagnosis and mitigation of network problems is dealt with from the perspective of software agents through the work called " "DEVELOPMENT OF A MULTIAGENT NOTIFICATION MANAGER TO REDUCE DIAGNOSTIC TIMES AND FAILURE MITIGATION IN IP NETWORKS", which generates useful knowledge and mitigation action in the event of a network failure. This multi-agent system was developed under the approach of software agents, using the INGENIAS methodology for the analysis and design of the prototype, as a modeling tool the INGENIAS Development Kit, the latter is an integrated development environment IDE for the analysis, design and implementation of multi-agent systems and as a reasoning engine to the ESPER library, which performs complex event processing, finding and identifying patterns of common failures in the network. The multi-agent system developed has the characteristic of being proactive, for which, it is permanently "listening" to the notifications coming from a network of IP routers and interacting with it to reduce the diagnosis time in the event of network failures, mitigate problems instability in the network and propose solutions to failures in the network.

Keywords: Multi-agent systems, artificial intelligence, network management, complex event processing, modeling, IP routers.

CAPÍTULO I

1. INTRODUCCIÓN

1.1. Situación problemática

La gestión de red abarca la planificación, organización, supervisión y control de componentes de telecomunicaciones para garantizar un adecuado nivel de servicio de acuerdo con un determinado coste (Barba Martí, 1999). La liberalización de los monopolios de telecomunicaciones a finales de la década de los 80 trajo consigo importantes cambios en la gestión de red, entre las que destacan, la introducción de nuevos servicios a los usuarios, el control de un número creciente de equipos de red y la integración de distintas tecnologías adquiridas a diferentes proveedores (Millán, 2004). La eficiencia de los administradores de red para gestionar redes disminuye a medida que estas se vuelven más complejas y heterogéneas. Administrar grandes redes creó una crisis para muchas organizaciones (Pavón et al, 2003). Es por ello que, no es suficiente conocer que fallo puede estar produciéndose en la red en un momento dado, sino que, resulta indispensable anticiparse y conocer que acciones tomar. Un problema comúnmente observado en los Centros de Gestión de Redes (Network Operation Center – NOC) es el gran volumen de notificaciones procedentes de las redes de telecomunicaciones.

Por otro lado, (PSC, 2020) dice: “Todos los días, los servidores y otros dispositivos de red producen decenas de miles de notificaciones. De hecho, la empresa promedio acumulará hasta 4 GB de archivos de notificaciones al día. Más del 95% de las notificaciones dentro de los archivos de registro consisten en entradas detalladas que registran cada evento o transacción exitosa que tiene lugar en el sistema”. El autor afirma que, en proceso de revisión manual de notificaciones, la posibilidad de conocer y localizar un problema relacionado con la seguridad o el desempeño

de la red disminuye exponencialmente mientras más equipos existan produciendo datos de registros.

(Grimes, 2018) expone que recopilar demasiadas notificaciones de la red abrume a los sistemas y al personal. El autor menciona que:

- Muchas empresas están recopilando demasiadas notificaciones, de a menudo miles de millones de eventos.
- Una montaña de datos no les proporciona a las empresas tanta información útil como les gustaría.
- Si se recibe tantas alertas que no pueden responderlas adecuadamente, entonces algo debe cambiar.

En general, en los sistemas y dispositivos se tiene muchos ficheros de logs con muchos eventos, lo cual lleva a un trabajo laborioso para su revisión manual por parte del operador o administrador de la red. “La búsqueda de problemas puede ser una tarea tediosa y que desperdicia tiempo” (Sematext, 2019). Las notificaciones recibidas de la red a menudo son analizadas por el gestor de red, lo cual, hace que el trabajo de identificación del origen del fallo sea muchas veces tedioso (conversión de datos en información), lento, no confiable e inexacto debido a que no pocas veces el personal que realiza estas labores no es experto.

(Kent & Souppaya, 2003) explican que, en la mayoría de las organizaciones, los administradores de redes y sistemas han sido tradicionalmente responsables de realizar análisis de las notificaciones recibidas desde la red para identificar y priorizar eventos de interés. Los autores afirman que los administradores responsables de realizar el análisis de registros a menudo no reciben capacitación para hacer este trabajo de manera eficiente y efectiva.

Lo anteriormente explicado da como resultado la obtención de información a destiempo para la toma de decisiones en la solución de incidentes dentro de la red de telecomunicaciones, es decir, los tiempos de identificación de los problemas de red son muchas

veces mayores, incrementando la indisponibilidad de los servicios de red, que se traduce en perjuicios a los usuarios finales en el uso de los mismos. Además, debido a la pandemia existente, muchas empresas empiezan a utilizar el teletrabajo en forma masiva, lo que requiere que las redes tengan un mayor grado de disponibilidad, lo que se logra con un diagnóstico y solución de fallos de red más rápido que los métodos manuales tradicionales.

1.2 Formulación del problema

En los Centros de Operaciones de Redes (Networking Operations Center – NOC), la gran cantidad de notificaciones recibidas hacen que los tiempos de diagnóstico y mitigación manual de fallos sean altos. Se presentan dos problemas:

- a) ¿Cómo reducir el tiempo empleado en el proceso de diagnóstico manual de fallos en redes IP?
- b) ¿Cómo reducir el tiempo empleado en el proceso de mitigación manual de problemas de inestabilidad en redes IP?

1.3. Alcance del trabajo de investigación a desarrollar

El presente trabajo de tesis desarrolla un prototipo funcional de un gestor de notificaciones para redes IP basado en tecnologías multiagente (inteligencia artificial), con características de autogestión, desatención, escalabilidad y pro actividad que impacte en la mejora de los tiempos de diagnóstico y mitigación fallos en la red y, consecuentemente, mejorar el rendimiento del mismo en favor de los usuarios que exigen una mayor calidad de servicio. En ese sentido, un sistema de gestión de eventos bien diseñado permite que el personal en el NOC sea proactivo y efectivo para detectar y diagnosticar problemas de red (Cisco, 2019). El prototipo gestor de notificaciones, recolecta notificaciones provenientes de una red equipos enrutadores y realiza lo siguiente:

- a) Filtra en tiempo real los eventos e identifica veinte (20) notificaciones importantes o incidentes. En función a su base de

conocimientos notifica al operador de red que acciones debe tomar para solucionar el incidente detectado. El gestor de notificaciones debe ser extensible para que en un futuro incluir una cantidad mayor de notificaciones (sin limitaciones) previamente seleccionadas por importancia.

- b) Correlaciona aquella secuencia de eventos que permite determinar la condición de fallo perjudicial para la red llamada “flapping del protocolo OSPF” que ocasiona que el rendimiento de la red disminuya notablemente debido a intermitencias en el estado de las interfaces físicas de red. En función a la información almacenada en su base de conocimientos y a las notificaciones provenientes de la red, el gestor de notificaciones mitiga el “flapping” apagando la interfaz de red afectada y notifica al operador de red la acción realizada en forma desatendida. El gestor de notificaciones debe ser extensible para incluir una cantidad mayor de correlaciones y otros protocolos previamente seleccionadas por importancia

1.4. Justificación teórica

La ingeniería de software se basa en un conjunto de principios fundamentales de la ingeniería que tiene como objetivo desarrollar de manera económica software fiable y que funcione con eficiencia en computadores reales (Pressman, 1998). El desarrollo de software es un conjunto de actividades creativas y la ingeniería del software intenta de organizar estas actividades con el fin de minimizar el riesgo de fracaso en la obtención de los objetivos deseados por medio de técnicas que han demostrado que funcionan en base a experiencias previas.

En este sentido, el aporte se justifica en el uso nuevas metodologías como INGENIAS, para el análisis y diseño de sistemas de software orientados a agentes aplicado al desarrollo de sistemas de gestión de redes.

1.5. Justificación práctica

Se hace uso de la programación orientada a agentes sobre la clásica programación orientada a objetos debido a que aporta mejoras significativas en cuanto a autonomía, habilidad social, reactividad y proactividad en los sistemas de procesamiento distribuido y sistemas expertos (Wooldridge, 2002).

Se justifica en forma práctica con el desarrollo de un prototipo experimental de un gestor de notificaciones para redes con las siguientes características:

- Basado en tecnologías multiagente (inteligencia artificial)
- Autogestión, desatención, escalabilidad y proactividad

Además, la solución será la base para el desarrollo de sistemas de gestión de redes más complejos que aprovechan las ventajas de la ingeniería de agentes inteligentes.

1.6. Objetivos de la investigación

1.6.1. Objetivos generales

Desarrollar un sistema multiagente gestor de notificaciones recibidas de una red, que impacte en forma positiva en la mejora de los tiempos empleados en:

- El proceso de diagnóstico de las fallas en redes IP.
- El proceso de mitigación de problemas de inestabilidad en redes IP.

1.6.2. Objetivos específicos

- a) Desarrollar un agente clasificador que periódicamente reciba, clasifique y procese mensajes de eventos provenientes de los equipos de telecomunicaciones, acerca de alarmas o fallos en dichos equipos.

- b) Desarrollar un agente experto que efectúa razonamiento complejo de eventos basado en la información pre procesada y obtenida del agente clasificador. La agente diagnóstica y obtiene conclusiones y hace recomendaciones que posteriormente serán mostradas al usuario humano.
- c) Desarrollar un módulo de software proactivo-solucionador que reciba órdenes del agente experto y que actuará en forma desatendida sobre el equipo de red tratando de mitigar un problema presentado, si es que la solución del problema está al alcance del agente.
- d) Desarrollar un agente interfaz, que muestre los eventos recibidos, las recomendaciones y/o acciones realizadas por el sistema al usuario humano.

1.7. Hipótesis

1.7.1. Hipótesis general

Los sistemas multiagente mejoran el tiempo empleado en los procesos de diagnóstico y mitigación de fallos en redes IP.

1.7.2. Hipótesis específicas

En base a las notificaciones recibidas de la red IP:

- Los sistemas multiagente, haciendo uso de la inteligencia artificial, mejoran el tiempo empleado en el proceso de diagnóstico de fallos.
- Los sistemas multiagente, haciendo uso de la inteligencia artificial, mejoran el tiempo empleado en el proceso de mitigación de fallos de inestabilidad en la red (flapping OSPF).

1.8. Aspectos metodológicos

1.8.1. Metodología a emplear

El tipo de investigación a realizar es un proyecto experimental a través de un prototipo funcional que será la una implementación simplificada, una arquitectura multiagente. El presente trabajo de investigación hace uso extensivo de la metodología Ingenias (Pavon et al., 2008), la cual, se emplea para el desarrollo de sistemas multiagente (SMA), donde se define un conjunto de meta-modelos que describen el sistema desde diferentes enfoques: agentes en solitario, entorno de los agentes, organizaciones de agentes, interacciones entre agentes, roles de los agentes, tareas y objetivos de los agentes. La secuencia para desarrollar el sistema es la siguiente:

- **Análisis del sistema:** Su objetivo es realizar el análisis del sistema a través de la descripción del sistema a alto nivel, análisis de casos de uso, análisis de requisitos y elaboración del meta-modelo de organización empleando la metodología Ingenias.
- **Diseño del sistema:** Posterior al análisis del sistema. Su objetivo el modelar el sistema implementando los meta-modelos de agente, objetivos y tareas e interacción entre los agentes a través de la metodología Ingenias.
- **Pruebas y recopilación de información:** Realización de pruebas para demostrar la hipótesis. Pruebas de recepción, clasificación, razonamiento complejo de eventos y notificación en base a la información pre procesada.

1.8.2. Ambiente de estudio

- Una red virtualizada basada en la tecnología IP y compuesta por enrutadores o nodos que emulan una red WAN (Lima, Trujillo, Huancayo y Arequipa).

- Un framework para el desarrollo de sistemas multiagente en el lenguaje JAVA.
- Una herramienta para el diseño y modelamiento de sistemas multiagente.
- Una librería, en JAVA, para el desarrollo de aplicaciones para el procesamiento complejo de eventos.
- Un entorno de desarrollo flexible para la programación del prototipo.

1.8.3. Análisis e interpretación de la información

Los datos que recibe el gestor de notificaciones multiagente para redes IP, son hechos que actúan sobre una base de datos de conocimientos y un motor de inferencia, el cual razona y obtiene conclusiones sobre alguna condición de error en el equipo de comunicaciones. El gestor de notificaciones multiagente para redes IP propone al operador de red la solución un conjunto de fallos y actúa sobre la red, mitigando intermitencias ocasionadas por interfaces de equipos de comunicaciones. Las evidencias de los fallos, notificaciones, recomendaciones y solución son mostradas al operador de red mediante ventanas visuales. En ese sentido se mejorarán las siguientes variables:

- Tiempo promedio de identificación y planteamiento de solución manual de una notificación importante entre miles de notificaciones recibidas para un administrador poco experto: TPM
- Tiempo promedio de identificación y de solución manual de conjunto de notificaciones de intermitencia importantes entre miles notificaciones recibidas para un administrador poco experto: TPSM

CAPÍTULO II

2. MARCO TEÓRICO

2.1. Marco filosófico o epistemológico de la investigación

Desde tiempos inmemoriales siempre ha existido el interés del ser humano en comprender su propia inteligencia y poder replicarla en otros organismos o máquinas. En este punto, veremos algunos fundamentos de la Inteligencia Artificial (IA) y la Inteligencia Artificial Distribuida (IAD)

La IA entra en contacto con la filosofía en diversos puntos, pero muy particularmente en las áreas de la lógica y la epistemología. Los cuestionamientos de la epistemología tradicional comienzan a extrapolarse hacia los procesos considerados como inteligentes en las máquinas (Rendueles, et al., 2007). En este sentido, los estudios de la inteligencia artificial en las máquinas se han dividido en dos grandes corrientes: las que estudian la forma de actuar de los humanos y las que toman como referencia un concepto ideal de inteligencia, que llamaremos racionalidad. Un sistema es racional hace “lo correcto”, en función de su conocimiento (Russel & Norvig, 2008).

2.1.1. El nacimiento de la Inteligencia Artificial

Entre los intelectuales que más han aportado al inicio de la IA destaca Alan Turing que ideó una prueba, llamada “Test de Turing”, para determinar si una computadora podía pensar (González, 2011).

El test consiste en que se evaluará conversaciones en lenguaje natural entre un humano y una máquina diseñada para generar respuestas similares a las de un humano. La máquina pasará el test, si el ser humano no puede determinar si la máquina es un ser humano o no. (Turing, 1951) propone que “si alguna máquina particular puede ser descrita como un cerebro, solo debemos programar la máquina digital para imitar a esta y también será un

cerebro”. Según Turing, la inteligencia es una función computable de una máquina, donde no interesa el material con que esta hecha, y que crear inteligencia artificial significa diseñar máquinas que imitan lo que hace el cerebro humano.

2.1.2. La Inteligencia artificial distribuida

La IAD surge como una rama de la IA que tiene como objetivo estudiar sistemas inteligentes conformados por un grupo de agentes (Arias et al, 2011), con el objetivo de resolver aquellos problemas donde la conducta colectiva es más eficiente que la conducta individual, en contraste con el estudio la IA que hace el análisis de un único componente que se ubica en un entorno que no cambia y que trata resolver todo el problema por sí solo.

En lugar de tener un gran sistema centralizado que recolecta la información completa del sistema, se tiene un número pequeño de subsistemas o agentes, que están enfocados en el esfuerzo de cooperar para resolver un problema determinado. En realidad, esto no significa que los sistemas tengan que ser divididos en pequeños subsistemas, más bien implica tener varias aplicaciones separadas, cada una con la capacidad de hacer frente a un determinado aspecto de un problema. Todas estas aplicaciones pueden comunicarse entre ellas, intercambiar puntos de vista y encontrar diversas estrategias para obtener como resultado una solución determinada.

Lo explicado anteriormente se denomina “Distributed Problems Solutions” (DPS) (Durfee et al., 1994), donde cada uno de los subsistemas que cooperan en su conjunto puede ser considerado un agente. Desde el punto de vista humano, un agente se puede considerar en términos de sus creencias, deseos e intenciones (BDI). La implementación de un sistema con más de un agente se denomina sistema multiagente (SMA), en el cual, todos los agentes participan en la solución del problema de acuerdo a la función

asignada por el diseñador del sistema o asignada por un ser humano.

2.1.3. Problemas de la inteligencia artificial distribuida

(Bond, 1988) identifica una serie de problemas comunes que se presentan en la IAD, los cuales son:

- Cómo enunciar, describir, dividir y asignar problemas.
- Cómo condensar los resultados generados por un grupo de agentes.
- Cómo entrenar a los agentes para que se interactúen y se comuniquen.
- Cómo asegurar que los agentes actúen coherentemente en el momento de realizar acciones o tomar decisiones.
- Cómo desarrollar agentes que puedan representar y raciocinar acerca de planes, acciones y el conocimiento de otros agentes.
- Cómo identificar y conciliar otros puntos de vista e intenciones conflictivas.
- Cómo emplear técnicas de ingeniería de software y desarrollar software con IAD.

2.2. Antecedentes de investigación

En esta última década, varios investigadores han propuesto trabajos que integran la gestión de redes con ingeniería de agentes inteligentes. Cada trabajo aborda un problema en particular de una parte de la realidad de la gestión, por lo que es importante tener en cuenta los avances investigativos en este campo. En este sentido exponemos los siguientes sistemas:

- Un sistema de monitoreo de red mejorado que usa tecnología basada en múltiples agentes.

- Gestión de redes basada en arquitectura multiagente.
- Gestión de red usando sistemas multiagente.

2.2.1. Un sistema de monitoreo de red basada en múltiples agentes

Los autores (Akinyokun, et. al., 2017) presentan un sistema basado en múltiples agentes para monitorear nodos en un entorno de red. Los modelos desarrollados para el sistema propuesto definen ciertos agentes deliberativos que interactúan entre sí para lograr los objetivos y requisitos de la organización de múltiples agentes, y la integración de servicios de fallas, servicios de seguridad y servicios de configuración en una única plataforma. El modelo desarrollado tiende a resolver ciertos desafíos como interfaces en conflicto, espacio de almacenamiento en el disco duro, consumo de memoria y, sobre todo, el problema de interoperabilidad que enfrentan los administradores con respecto a la instalación de más de una aplicación de monitoreo de red para monitorear y administrar un entorno de red. El monitor de red basado en múltiples agentes propuesto se desarrolló utilizando Core Java, HTML, preprocesador de hipertexto (PHP), JavaScript, JQuery y MySQL.

El sistema de monitoreo de red automatizado integra servicios de configuración, servicios de administración de fallas y servicios de seguridad en una única plataforma de software para monitorear los nodos en un entorno de red de manera efectiva utilizando un enfoque basado en múltiples agentes según lo recomendado por (Akinyokun, et al., 2017).

La figura 2.1 muestra nuestra la arquitectura del sistema basado en ocho agentes que cooperan en el mismo dominio para realizar servicios de monitoreo de red para cumplir con el objetivo principal. Las funciones de los agentes son:

- **Agente de servidor:** Realiza los procesos de configuración inicial, pruebas de red, recopilación de datos, análisis y funciones de minería programados por parte del administrador de la red. Puede aceptar y enviar solicitudes hacia y desde otros

agentes. Envía mensajes al gestor de informes para su posterior transmisión al sistema de notificación de pantalla del administrador de la red y también se integra con los servicios web para alertas de SMS a través del teléfono móvil. El facilitador de directorio controla y coordina todas las operaciones y procesos de los agentes, mientras que el subsistema de mensajería ayuda en las comunicaciones de los agentes.

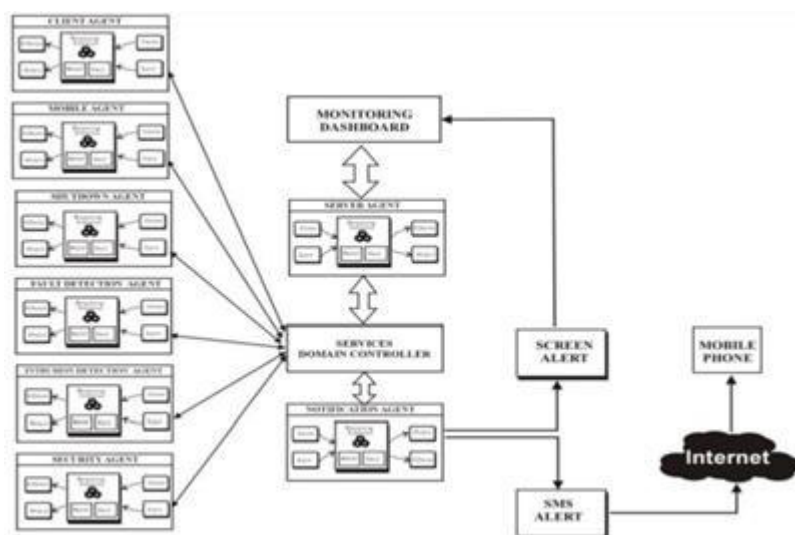


Figura 2.1.- Arquitectura propuesta del sistema basada en múltiples agentes.

Fuente (Akinyokun, et al., 2017)

- **Agente cliente:** Ayuda a buscar información sobre cada computadora en la que reside y también pasa instrucciones al sistema operativo del cliente para ejecutar tarea específica como la operación de apagado. También ayuda a recopilar información actualizada sobre cada computadora o nodo de cliente para la transmisión posterior al agente servidor utilizando el agente móvil.
- **Agente de detección de fallas:** El agente de detección de fallas está diseñado como un agente reactivo que monitorea todos los nodos en la red de área local para detectar nodos en estado activo o inactivo, utilizando para ello un sistema de sondeo activo y una técnica basada en reglas para clasificación. El

mecanismo de detección de fallas genera un mensaje de alerta que indica la falla detectada a través de un agente de notificación (alerta de pantalla/sms) tan pronto como cualquier estado del sistema cambie de estado de activo a inactivo. Este cambio de estado puede ser el resultado del apagado del sistema o un problema del cable de red.

- **Agente de detección de intrusiones:** El agente de detección de intrusos usa un mecanismo de detección anomalías en los nodos del entorno de red. El agente está entrenado para la tarea de detección de intrusos utilizando las técnicas Fuzzy Art. El sistema constantemente supervisa el tráfico de la red, recopila y analiza patrones de datos en tiempo real de todos los dispositivos en la red. Toda la información de intrusión se registra y los mensajes de alerta de pantalla/sms también se envían al administrador de la red.
- **Agente de notificación:** Realiza la función de enviar un mensaje de alerta casi en tiempo real de todos los eventos de red como mensajes de alerta en la pantalla del monitor de red y servicio de mensajes cortos (sms) al teléfono móvil del administrador de la red.
- **Agente de apagado:** Es un agente autónomo y reactivo diseñado para apagar todos los nodos perfilados con estado activo en la red, utilizando un tiempo preprogramado. Les da a los usuarios de la computadora un tiempo de gracia de 1 minuto para guardar sus trabajos antes de ejecutar el comando de apagado.
- **Agente de seguridad:** Crea una comunicación segura de extremo a extremo entre el agente del servidor y el agente del cliente residente en cada nodo de la red, encriptando y desencriptando mensajes (nombre del nodo, dirección IP,

dirección MAC y nombre del sistema) antes y después transmisión para evitar que agentes malintencionados se apoderen de información confidencial utilizando un algoritmo de doble DES.

- **Controlador de dominio de servicio:** Se encarga directamente de supervisar y controlar todos los servicios de los agentes en el dominio. Primero debe iniciarse antes de activar cualquier otro agente.
- **Agente móvil:** Es el agente de mensajería que se puede enviar desde el controlador de dominio de servicio a otros nodos en el entorno de red. Ayuda a monitorear cada nodo en la red, entregar la operación de comando de apagado a cada agente del cliente y actualizar los registros entre el cliente y los agentes del servidor.

Conclusión: Según los autores, las pruebas de la aplicación de monitoreo de red propuesta utilizando el modelo de múltiples agentes resultó altamente eficiente. La integración de los servicios de ontología de fallas, detección de intrusos y configuración en una sola plataforma que usa múltiples agentes para el entorno de red es un avance importante para resolver los problemas del administrador de la red. El sistema desarrollado propuesto puede compararse relativamente con algunas soluciones de software estándar en términos de tiempo de respuesta y velocidad. Además, se indica que, en el futuro, las soluciones de monitoreo/administración de red deben ser muy inteligentes y flexibles para reducir la carga sobre los administradores y recursos de la red. Más servicios de red requieren una integración perfecta en una plataforma de solución única que utiliza arquitectura de tecnología de múltiples agentes.

2.2.2. Gestión de redes basada en arquitectura multiagente

Los autores del documento (Raya-Díaz, et al., 2017) proponen una arquitectura de gestión de red basada en agentes inteligentes. El enfoque de esta arquitectura es la gestión autónoma de redes mediante el diseño de un método colaboración basado en el dominio orientado a tareas. Este método implementa un mecanismo de negociación en los agentes para mejorar el desempeño de la gestión de red a través de una arquitectura jerárquica multiagente que se presenta en la figura 2.2.

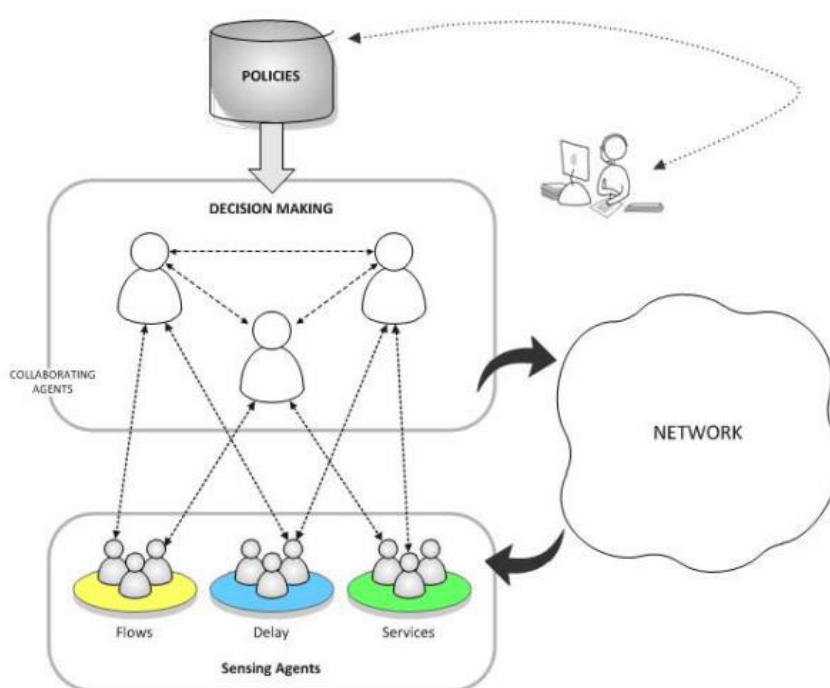


Figura 2.2.- Arquitectura multiagente. Fuente (Raya-Díaz, et al., 2017)

El administrador de la red especifica y mantiene las políticas de alto nivel. En el nivel inferior, hay un conjunto de agentes que recopilan datos relacionados con los parámetros de QoS, que, al detectar el estado actual de la red, estos agentes transfieren los datos al nivel que contiene el módulo de toma de decisiones; a través de un mecanismo de negociación entre un conjunto de agentes en colaboración que se rigen por las políticas de red. Como resultado de la negociación, las acciones a realizar por los agentes se determinan por la ejecución de actuadores de bajo nivel, dando

como resultado la optimización de red. La arquitectura multiagente propuesta especifica dos tipos de agentes:

- Agentes sensores (As): son agentes reflejos basados en modelos que almacenan información sobre el estado actual del mundo utilizando un modelo interno; la acción realizada por un agente se basa en su percepción y reglas. Un agente sensor es definido por una tupla $\langle S, R, T \rangle$ donde S es el estado real del parámetro que está monitoreando, R es un conjunto de reglas y T es un conjunto de las posibles tareas a ejecutar.
- Agentes Colaboradores (Ac): son agentes basados en la utilidad que utilizan un modelo del mundo y una función de utilidad que permitirá la toma racional de decisiones realizada por cada agente. Un agente colaborador Ac se define por una tupla $\langle S, T, P, C, U \rangle$, donde S es el estado del mundo, T es el conjunto de posibles tareas, P son las políticas o reglas, C es el costo del acuerdo δ , finalmente hay una utilidad U para el acuerdo δ , calculado por el costo de ejecutar T_i menos el costo de aceptar el acuerdo δ .

Conclusión: De esta manera, los autores emplean el dominio orientado a tareas como una forma de utilizar métodos de inteligencia artificial para implementar un protocolo de negociación porque los agentes tienen una función que califica la utilidad de cada acuerdo, para que puedan tomar una decisión y optimizar su actual utilidad.

2.2.3. Gestión de red usando sistemas multiagente

Los autores (Isaza, et al., 2017) presentan un sistema multiagente para la gestión de redes que definen un conjunto de agentes inteligentes que interactúan para lograr los objetivos y requisitos de

la organización multiagente. Estos agentes tienen la propiedad de ser adaptativos, adquirir conocimientos y habilidades para tomar decisiones de acuerdo con el estado real de la red que está representada en la base de información (MIB) de dispositivos que usan el protocolo SNMP. El estado ideal de la política de red lo define el usuario final, el cual configura los valores que deben tener las variables de rendimiento y otros parámetros, como la frecuencia con la que estas variables deben ser monitoreadas. Los autores indican que una arquitectura basada en agentes aumenta la integración, adaptabilidad, cooperación, autonomía y la operación eficiente en un entorno heterogéneo en la supervisión de la red. El sistema multiagente está compuesto por el “Agente Interfaz”, “Agente Gestor”, “Agente de Información” y “Agente de Traps”. La figura 2.3 muestra el diagrama de colaboración del sistema donde los agentes y los mensajes son intercambiados con sus propias performativas y se representa la acción implementada en la ontología del dominio particular.

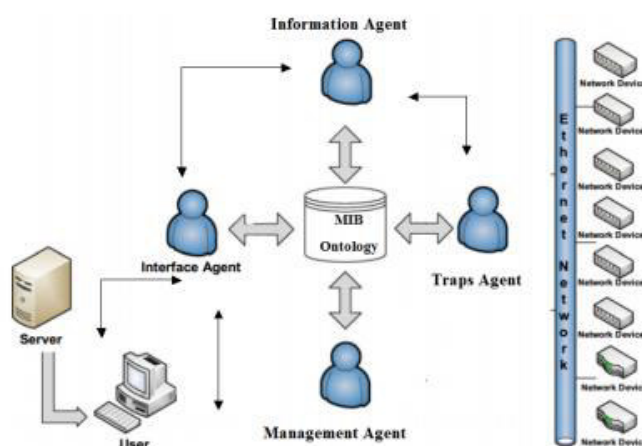


Figura 2.3. Diagrama de colaboración de los agentes. Fuente:(Isaza, et al., 2017)

A continuación, se describen los agentes del sistema multiagente desarrollado:

- Agente de Interfaz: Este agente está dedicado a recibir las direcciones IP y OID especificados por el usuario, esta información es necesaria para conocer el dispositivo, además,

este agente presentará la información relacionada con acciones anormales durante el funcionamiento de la red, esta información se muestra como notificaciones y alarmas.

- Agente de información: Este agente se dedica a interactuar con el “Agente de traps”, con el “Agente gestor” y con la API SNMP utilizada para la comunicación con los dispositivos de red. Este agente actúa como intermediario entre el sistema multiagente y la red monitoreada.
- Agente Traps: Este agente se dedica a recibir los traps SNMP que se envían desde el “Agente de información” y almacena los datos (fecha, dirección IP de origen y OID-SNMP) en una base de datos. Notifica al “Agente Gestor” sobre la notificación recibida.
- Agente de gestión: Es el intermediario entre el “Agente de Interfaz” y el “Agente de información”. Recibe la solicitud de OID y la envía al “Agente de Información”; además, recibe los resultados y responde al agente de origen (“Agente interfaz”). Este agente rastrea los OID en algunos dispositivos para conocer el comportamiento de la red para generar las alarmas apropiadas de acuerdo con el umbral definido por el usuario.
- Ontología: La ontología desarrollada consta de los conceptos como dirección IP, OID, Valor, Descripción y de las acciones Buscar, Actualizar y Alarmar.

Conclusión: Los autores indican que la ejecución del sistema en un entorno controlado dio los resultados esperados de acuerdo con las limitaciones de las políticas establecidas y los umbrales definidos. El efecto de la ejecución del sistema multiagente es monitorear la red y poder detectar si se está violando la política asociada a la

ubicación del dispositivo. El comportamiento de los agentes y la detección de eventos entre ellos a través del intercambio de mensajes permiten escanear la red; luego, el “Agente de administración” genera una alarma que se almacena en la base de datos y se envía al “Agente de interfaz” que se encarga de mostrar la información a los usuarios.

2.2.4. Conclusiones de los antecedentes

La investigación referente a gestión de redes y SMA actualmente se encuentra en etapa de desarrollo incipiente debido a que en internet es difícil encontrar trabajos publicados al respecto, así como herramientas, metodologías y especificaciones técnicas. Se espera que la investigación en este ámbito se desarrolle hasta tener la suficiente madurez para que la industria comercialice productos basados en el paradigma orientado a agentes, que, en teoría, supone importantes ventajas con respecto a los modelos de gestión de redes tradicionales.

Si bien los prototipos académicos mostrados en el anterior ítem, demuestran que la implementación de sistemas de gestión de red empleando SMA es factible, se espera que un futuro surja nuevas propuestas innovadoras al respecto, que aporten nuevas características y funcionalidades para la solución de problemas en la gestión de redes que están presentado los sistemas actuales debido a la complejidad de las redes y sus componentes gestionados.

Los estudios académicos que relacionan la gestión de redes y los sistemas multiagente abarcan sólo algunos aspectos puntuales de la gestión en redes de área local pequeñas, quedando mucho campo por explorar para otros tipos de redes como redes metropolitanas o redes de área amplia.

En cuanto a las metodologías empleadas, no existe consenso, en cuanto al uso de las metodologías de desarrollo de software orientado a agentes. El desarrollo de sistemas orientados a agentes

no se ha visto desarrollado e impulsado masivamente en la creación de soluciones comerciales, como sí sucede en el desarrollo de software orientado a objetos.

2.3. Bases teóricas

A continuación, se exponen el grupo central de conceptos que se utilizan para formular y desarrollar el presente trabajo. Esto se refiere a las ideas básicas que forman la base para los argumentos que son necesarios para desarrollar un trabajo cohesivo y convincente en base a la metodología orientada a agentes llamada INGENIAS.

El objetivo de la presente exposición es especificar las variables de estudio en forma concreta, de tal forma que, el desarrollo posterior del trabajo esté obedezca a los lineamientos del marco teórico desarrollado, así como, seleccionar las definiciones más importantes y representativas de los tópicos que sustentan el planteamiento del problema específico a enfrentar en la solución propuesta, en otras palabras, realizaremos las definiciones en los campos teórico, tecnológico y metodológico que nos llevan a la operacionalización de las variables que posteriormente serán influidas en forma positiva para solución del problema planteado.

2.3.1. Aspectos del modelo TCP/IP

El Protocolo de Internet (IP) se ha convertido en el protocolo universal de la comunicación global porque hace posible que cualquier dispositivo o equipo de cualquier tecnología pueda consumir recursos compartidos por otros equipos conectados a la red. Sobre el protocolo IP se implementa el Protocolo de Datagrama de Usuario (UDP), que es un protocolo a nivel de transporte que se emplea en el caso que los protocolos orientados a conexión no se adecuen a ciertas aplicaciones, así como, para la transmisión de información en tiempo real como voz o videoconferencia, donde no es posible realizar retransmisiones por

los requisitos de retardo que exigen estas aplicaciones. Existe una gran variedad de protocolos de servicio que se implementan sobre el protocolo IP y UDP, pero entre ellos, destaca el protocolo para el envío de mensajes relacionados con eventos que suceden en equipos de red o SYSLOG.

2.3.1.1. Que es TCP/IP

TCP/IP es un modelo que describe un conjunto de protocolos de red desarrollados en la década de los 70 por Vinton Cerf y Robert E. Kahn (UPM, 2019). Fue implementado por primera vez en la red ARPANET (Advanced Research Projects Agency Network), la primera red de área amplia norteamericana desarrollada por encargo de DARPA (Defense Advanced Research Projects Agency), la agencia del Departamento de Defensa de los Estados Unidos, la cual posteriormente se convierte en Internet. TCP/IP se llama a veces como Modelo de Internet o Modelo DARPA

La suite de protocolos TCP/IP permite que computadoras, teléfonos inteligentes y dispositivos integrados de todos los tamaños, suministrados por muchos vendedores de computadoras diferentes y funcionando con diferente software, puedan comunicarse entre sí (Fall et al., 2012).

2.3.1.2. Capas del Modelo TCP/IP

Los protocolos del modelo TCP/IP se desarrollan en capas, donde cada capa es responsable de solucionar un problema diferente de la comunicación (Stevens, 1994). La familia de protocolos TCP/IP, es la agrupación de diferentes protocolos en capas superpuestas. TCP/IP se implementa como un modelo de cuatro capas, tal como se muestra en la Figura 2.4.

Aplicación	Syslog, HTTP, Telnet, FTP, e-mail, etc.
Transporte	TCP, UDP
Red	IP, ICMP, IGMP
Enlace	Controlador de tarjeta de interfaz

Figura 2.4. Las cuatro capas del modelo de protocolos TCP/IP

Cada capa tiene una responsabilidad diferente:

- a) La capa de enlace determina los mecanismos de acceso al medio físico. Físicamente consiste en una interfaz de red o tarjeta de red que emplea un software controlador para la propia tarjeta proporcionado por el sistema operativo. La tarjeta de red y el software asociado gestiona la forma en que los bits son enviados y recibidos desde y hacia el medio físico.
- b) La capa de red gestiona el encaminamiento de paquetes a través de la red. El protocolo principal de la capa de red se denomina “Internet Protocol” (IP). IP es independiente del medio físico debido a que fragmenta los paquetes si estos pasan de un medio físico con un “Unidad Máxima de Transferencia” (MTU) mayor a otro con MTU menor. La decisión de encaminamiento generalmente se toma en base al uso de protocolos en enrutamiento dinámico como RIP, OSPF u otros.
- c) La capa de transporte proporciona flujos de paquetes entre dos equipos extremos en una red. Esta capa consta de dos protocolos que se pueden usar dependiendo de la aplicación que se quiera transportar:
 - TCP (Protocolo de control de transmisión) y UDP (Protocolo de datagramas de usuario). TCP (IETF 793, 1981) proporciona un flujo de datos confiable entre dos

equipos de la red. TCP segmenta los datos que se transmiten desde la capa de aplicación en segmentos del tamaño adecuado para que la capa de red los pueda procesar, identifica los paquetes recibidos, establecer tiempos de espera y usa acuses de recibo para confirmar que la transmisión sea confiable. TCP proporciona un flujo confiable de datos, de tal forma que la capa de aplicación ignora lo que sucede en esta capa.

- UDP (Postel, 1980), es un protocolo que se encarga del envío de paquetes de datos llamados datagramas dos equipos. Este protocolo no garantiza que los datagramas lleguen al otro equipo. Si se requiere confiabilidad, este debe ser proporcionado por la capa de aplicación.
- d) La capa de aplicación maneja los servicios proporcionados a los usuarios. Existen miles de aplicaciones TCP/IP que son de uso estandarizado o privado. Entre algunos ejemplos de estas aplicaciones o servicios tenemos:
- Telnet para inicio de sesión remoto (Postel & Reynolds, 1983),
 - FTP, el protocolo de transferencia de archivos (Postel & Reynolds, 1985),
 - SMTP, el protocolo para correo electrónico,
 - SMTP (Klensin, 2008) y muchos otros más.

2.3.1.3. Protocolo de Internet o IP

El Protocolo Internet definido oficialmente el (IETF 791, 1981) proporciona las funciones necesarias para enviar paquetes desde un equipo origen hacia un equipo destino a través de un sistema de redes interconectadas. IP no dispone de

mecanismos para mantener la fiabilidad de datos entre los equipos extremos, por lo que, normalmente se apoya para esta función en el protocolo TCP.

2.3.2. Aspectos del protocolo SYSLOG

SYSLOG es un protocolo estándar a nivel de aplicación encapsulado sobre UDP que implementan los equipos de red y que permite enviar mensajes de eventos que suceden en la red, hacia equipos que almacenan dichos eventos. Los componentes mínimos que actúan en el protocolo SYSLOG son los “originators” y los “collectors”, y la especificación completa del protocolo lo define el Internet Engineering Task Force (Gerhards, 2009). Los componentes fundamentales del protocolo SYSLOG son los siguientes:

- a) **“originators”**: El envío de mensajes se realiza, porque cada equipo de la red implementa el protocolo SYSLOG. Los mensajes son enviados en forma básica, tal como el equipo los genera, es decir, el equipo hace un procesamiento muy elemental de la información generada con poca interpretación acerca del evento generado. Básicamente, el “originator” genera el evento dentro del equipo de la red IP con el contenido a ser transportado en el mensaje (Gerhards, 2009).
- b) **“collectors”**: Los “collectors” recogen el contenido del mensaje SYSLOG generados por los “originator”s, para su posterior almacenamiento. Los mensajes que se transportan entre “originators” y “collectors”, son mensajes UDP. El “originator” envía en forma de texto plano el mensaje y el “collector”, tiene a la escucha el puerto UDP 514, por donde recibirá el mensaje en texto plano. El “collector” almacenará la información en un archivo de texto, quedando a disposición del usuario

humano la interpretación del evento y de algún otro software externo que realice el tratamiento de la información almacenada para mostrarla al operador de red en una forma más adecuada. La figura 2.5 muestra la interacción entre un “originator” y un “collector”.

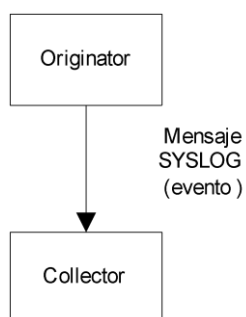


Figura 2.5.- Interacción entre el “originator” y el “collector”

Los mensajes SYSLOG del sistema pueden contener hasta 80 caracteres y un signo de porcentaje (%), que sigue el número de secuencia opcional o la información de la marca de tiempo, si está configurado (Cisco, 2008). Los mensajes se muestran en este formato:

seq no:timestamp: %facility-severity-MNEMONIC:description

La tabla N° 2.1 describe los elementos de los mensajes de SYSLOG:

Tabla N° 2.1.- Elementos del mensaje SYSLOG

Elemento	Descripción
seq no	Marca que registra los mensajes con un número de secuencia.
timestamp	Fecha y hora del mensaje o evento.
facility	La facilidad a la que se refiere el mensaje (por ejemplo,

	SNMP, SYS, etc.)
severity	Código de un solo dígito de 0 a 7 que es la gravedad del mensaje.
mnemonic	Cadena de texto que describe de forma única el mensaje.
description	Cadena de texto que contiene información detallada sobre el evento que se informa.

La tabla N° 2.2 describe las palabras clave de los niveles de gravedad. También enumera las definiciones de SYSLOG UNIX correspondientes desde el nivel más grave hasta el nivel menos grave.

Tabla 2.2- Palabras clave de los niveles de gravedad (severity)

Palabra clave	Nivel	Descripción
emergencia	0	Sistema inestable
alerta	1	Acción inmediata necesaria
critico	2	Condiciones críticas
error	3	Condiciones de error
alerta	4	Condiciones de advertencia
notificación	5	Condición normal pero significativa
información	6	Mensajes informativos
depuración	7	Mensajes de depuración

La figura 2.6 muestra un mensaje parcial de equipo access point:

```

00:00:46: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet0/1, changed state to up
00:00:47: %LINK-3-UPDOWN: Interface GigabitEthernet0/2, changed state to up
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to down
00:00:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed
state to down 2
*Mar  1 18:46:11: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
18:47:02: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Mar  1 18:48:50.483 UTC: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)

```

Figura 2.6.- Ejemplo de mensaje SYSLOG. Fuente: (Cisco, 2008-II)

Para la versión 15 del sistema operativo de los enrutadores Cisco, el campo facility se emplea para identificar un tipo de fuente que generó el evento. (Cisco, 2008) ha publicado la lista de códigos de facility para poder identificar las fuentes que generan los eventos.

Los códigos facility más comunes empleados en la identificación de problemas en las redes son: LINK, LINEPROTO, OSPF, SONET, SENSOR. El grado de relevancia se identifica mediante el “severity”. La descripción completa de todos los mensajes SYSLOG de Cisco se encuentra publicado en el site de Cisco (Cisco, 2012).

2.3.3. Aspectos de gestión de red

Se puede decir que la gestión de red es el conjunto de actividades realizadas para que la red brinde un adecuado nivel de servicio a los usuarios. Los objetivos principales de la gestión de red consisten en mejorar la disponibilidad y el rendimiento de los elementos del sistema, así como incrementar su efectividad.

Las recomendaciones de la OSI han definido las siguientes áreas funcionales para la gestión de redes (Barba Martí, 1999, p. 89):

- Supervisión y fallos: son aquellas actividades que permiten la identificación y solución de fallos a nivel de red.

- Configuración: consiste en configurar los equipos de manera óptima, así como, identificar y obtener información de los equipos para el apoyo a la operación del servicio.
- Contabilidad: permite obtener información del uso de los recursos de la red para calcular el costo que se cobra a los abonados o usuarios.
- Prestaciones: permite evaluar la performance de los equipos de la red para poder optimizar sus prestaciones.
- Seguridad: permite asegurar la confidencialidad, integridad y disponibilidad del servicio de red a los usuarios y abonados de la misma.

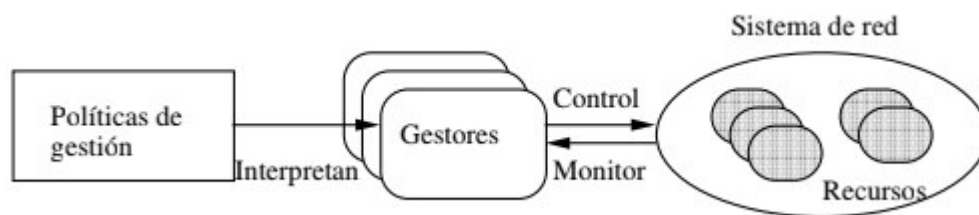


Figura 2.7.- Flujo de información de gestión. Fuente (Barba Martí, 1999, p. 89)

Los nodos de la red (recursos) tienen implementados agentes de software que recolectan datos de gestión. El software gestor en forma permanente monitoriza a los agentes recibiendo datos de sucesos o mediciones. El gestor interpreta las los datos y toma acción de acuerdo a políticas pre establecidas modificando las condiciones de la red para lograr un objetivo óptimo (Figuras. 2.7 y 2.8).

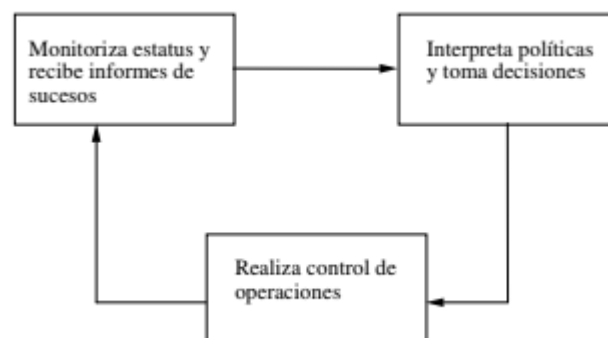


Figura 2.8.- Bucle de información de gestión. Fuente (Barba Martí, 1999, p. 15, 90)

2.3.3.1. Gestión de fallos

Un aspecto de vital importancia para la gestión de red es identificar y solucionar los problemas que se presenten, en este sentido, la gestión de fallos implica (Millán, 1999):

- Determinar el efecto de la falla
- Aislar la falla
- Solucionar la falla
- Verificar si la solución a la falla es correcta
- Almacenar la información de la falla y su solución como lecciones aprendidas.

2.3.3.2. Gestión de configuración

La gestión de configuración de red es el proceso de organizar y mantener información sobre todos los componentes de una red informática (Rouse, 2019). La gestión de configuración es un grupo de actividades realizadas que posibilitan identificar, obtener, proveer datos a objetos gestionados y controlar con el propósito de apoyar a la operación de los servicios de interconexión.

2.3.3.3. Gestión de prestaciones

Según (Acosta, 2016), gestionar prestaciones proporciona un conjunto de funciones destinadas a evaluar e informar acerca del comportamiento de los equipos de la red y por ende, la eficiencia de la misma.

2.3.3.4. Gestión de tarificación

La gestión de tarificación es un conjunto de facilidades que permiten establecer costos por el uso de determinados servicios de red (Barba, 1999).

2.3.3.5. Gestión de seguridad

La seguridad de las comunicaciones tiene que ver con la autenticidad, los controles de acceso, evitar que la comunicación sea vista por quien no debe y la integridad de los datos que se transmiten en cualquier comunicación entre clientes y sistemas (Acosta, 2016). En resumen, proporciona mecanismos de seguridad para mantener la confidencialidad, disponibilidad e integridad de la información.

2.3.4. Noción de agentes

La IA es una rama de las ciencias de la computación que más desarrollo ha tenido en las últimas décadas, dada la gran variedad de beneficios que ofrece. El campo de la ciencia e ingeniería tiene una amplia variedad de aplicaciones tales como sistemas de reconocimiento de voz, texto y formas faciales. La IA tiene múltiples ramas. Una rama de la IA se encarga de la investigación de las diferentes formas en que una entidad autónoma de software o componente de software autónomo pueda procesar, aprender y manejar información, y tomar decisiones lógicas que lleven al objetivo establecido. Esta entidad autónoma es llamada agente. Cuando un agente cuenta con estas características, se dice que es un agente inteligente o racional (Wooldridge y Jennings, 1995). Un agente es cualquier ente con la capacidad de percibir el ambiente en el que se encuentra, y que tiene la capacidad de actuar en el mismo (Russell y Norvig, 2004). Según este concepto, un agente racional puede ser una persona, una máquina, un software o cualquier objeto que tiene sus propias maneras de percibir y actuar sobre su entorno.

(Mallcu, 2008) explica que los agentes son racionales porque realizan la acción correcta, acción que los lleva a tener éxito en la realización de sus tareas, en ese sentido los agentes racionales son autónomos, porque son capaces de tomar decisiones de manera propia sin necesidad de tomar en cuenta a otros agentes para poder lograr los objetivos que persiguen. Una de las ramas de la IA es conocida como IAD (Inteligencia Artificial Distribuida), que tiene como meta la búsqueda de maneras en las que los agentes pueden cumplir sus objetivos en base a la información proporcionada por otros agentes. En base a los conceptos de IA e IAD expuestos anteriormente, (Mallcu, 2008) indica que la IAD estudia e intenta construir un conjunto de entidades autónomas e inteligentes que cooperan para realizar un trabajo y se comunican por medio de mecanismos basados en el envío y recepción de mensajes.

Los sistemas multiagente son una de las áreas de mayor dinamismo y desarrollo en el ámbito de la IA aplicada a las tecnologías de la información y comunicaciones. Un sistema multiagente está compuesto de un conjunto de componentes hardware y software autónomos que interactúan para resolver problemas que están fuera del alcance de cada componente individual (Burguillo, 2009).

Los SMA pueden ser utilizados para resolver problemas que no son fáciles o muy difíciles de resolver para un agente individual o un sistema tradicional (Torres et al., 2011). Esto implica que los SMA pueden resolver con mayor eficiencia situaciones donde los paradigmas tradicionales no suelen ser robustos y flexibles. (Sycara, 1998) ha identificado algunas ventajas que proporciona el empleo de SMA:

- Los agentes al ejecutarse en paralelo tienen mayor velocidad y eficiencia.
- El sistema sigue funcionando y rendimiento disminuye poco ante la falla de un agente.

- En sencillo seguir agregando nuevos agentes al sistema.
- Es más económico crear e implementar pequeños agentes que desarrollar un sistema completo.
- Al ser modular, es fácil desarrollar y mantener los agentes.

2.3.4.1. Concepto de Agente

En la literatura se distinguen dos usos generales del concepto de agente. El primero es una noción débil y poco controversial; el segundo es fuerte y potencialmente más controversial:

2.3.4.2. Noción débil de un Agente

Quizás la forma más general en la cual el término agente es usada, es para denotar un sistema de hardware o un sistema de cómputo basado en software que contiene las siguientes propiedades (Wooldridge y Jennings, 1995):

- Autonomía: Los agentes funcionan en forma independiente y sin la intervención directa de seres humanos, y tienen algún tipo de control sobre sus acciones y estado interno.
- Habilidad social: Los agentes interactúan entre si vía alguna forma de lenguaje de comunicación de agentes.
- Reactividad: Los agentes observan su ambiente y responden en una forma oportuna a los cambios que ocurren en su entorno.
- Proactividad: Los agentes no actúan simplemente en respuesta a su ambiente, ellos exhiben un comportamiento dirigido a metas tomando la iniciativa.

Los agentes se comunican con sus semejantes intercambiando mensajes en un lenguaje de comunicación de agentes

expresivo. Mientras los agentes pueden ser tan simples como subrutinas, típicamente ellos son entidades más grandes con alguna clase de control persistente (Villarrubia, 2000).

2.3.4.3. Noción fuerte de un Agente

Para algunos investigadores (aquellos que particularmente trabajan en IA) más interesados en adoptar el comportamiento humano en la noción de agente, el término agente tiene un significado fuerte y más específico que la noción débil de agente. Estos investigadores proponen que un agente es un sistema computacional que tiene las propiedades de agente débil, pero, además, es conceptualizado e implementado usando conceptos que son más usualmente aplicados a humanos. Por ejemplo, es muy común en IA caracterizar un agente usando nociones mentales, tales como conocimientos, deseos, IA han ido más allá, y consideran agentes emocionales. La base de la noción fuerte de agencia es la intencionalidad (Wooldridge y Jennings, 1995).

2.3.4.4. Clasificación de los Agentes

La primera clasificación de agentes, en base a su función, fue elaborada por Hyacinth Nwana, según esta taxonomía, los agentes se clasifican en (Nwana, 1996):

- a) Agentes Colaborativos:** Se caracterizan por tener autonomía, a la vez que cooperan con otros agentes para alcanzar los objetivos de sus dueños. Pueden llegar a aprender, aunque esta no es su principal habilidad. Para obtener coordinación entre los agentes colaborativos, es posible llegar a tener que negociar para alcanzar acuerdos mutuamente aceptables entre los agentes negociantes.

- b) Agentes de Interfaz:** Se caracterizan por tener autonomía y usar técnicas de aprendizaje automático para ayudar a los seres humanos. Proporcionan asistencia para que el usuario aprenda a utilizar una aplicación particular. Observa y supervisa las actividades realizadas por el usuario en la interfaz, aprendiendo nuevas opciones y sugiriendo mejores alternativas para efectuar las tareas encomendadas.
- c) Agentes Móviles:** Tienen la capacidad de trasladarse a través de la red para interactuar con otros agentes y sistemas, con el objetivo de obtener información para el usuario y regresando a su origen cuando su función se haya completado. De esta forma los agentes se pueden trasladar por la red interactuando en cada lugar donde estén con otros agentes en representación del usuario con el fin de satisfacer sus objetivos, para después regresar y comunicar sus resultados al usuario. Estos deberes pueden ir desde una reserva de vuelo hasta la gestión de una red de telecomunicaciones. Sin embargo, la movilidad no es ni una condición necesaria ni suficiente para el agente. Los agentes móviles son agentes, porque son autónomos y cooperan, aunque de manera diferente a los agentes de colaboración. Por ejemplo, pueden cooperar o comunicarse con un agente que realiza la ubicación de algunos de sus objetos internos y métodos conocidos por otros agentes. Al hacer esto, un agente intercambia datos o información con otros agentes sin dar necesariamente toda su información.
- d) Agentes de Información:** Realizan la búsqueda de información sobre grandes repositorios o bases de datos de datos. La labor que realizan los agentes de información

es filtrar datos provenientes de múltiples fuentes de manera autónoma y desatendida. Este tipo de agentes pueden ser estáticos o móviles, no sociales o sociales y pueden aprender o no aprender, es por ello que no hay un modelo único que defina su forma de operación.

- e) **Agentes Reactivos:** Actúan en forma reactiva ante una notificación o estímulo externo producido en el entorno del sistema multiagente. Al no disponer de una representación del entorno que los rodea, solo reaccionan en base a la lógica de estímulo-respuesta. Estos agentes son limitados en autonomía, por lo que, normalmente forman grupos que realizan labores particulares y autónomas.
- f) **Agentes Híbridos:** Son agentes que poseen dos o más características de los agentes anteriormente indicados. Combinan las ventajas de poder reaccionar ante eventos percibidos desde su entorno exterior y hacen uso de características deliberativas para planificar sus actividades.
- g) **Sistemas de Agentes Heterogéneos:** A diferencia de los sistemas híbridos descritos en la sección anterior, se refieren a una configuración integrada de al menos dos o más agentes que pertenecen a dos o más clases de agentes diferentes, donde cada agente puede pertenecer a alguna de las clases de mencionadas anteriormente. Los agentes heterogéneos también pueden contener agentes híbridos.

2.3.4.5. Arquitecturas para construir agentes

Las arquitecturas especifican cómo se descomponen los agentes en un conjunto de módulos que interactúan entre sí para lograr la funcionalidad requerida.

Básicamente existen tres clases de arquitecturas (Mas et al, 2005, pp. 1-40):

- Arquitecturas deliberativas
- Arquitecturas reactivas.
- Arquitecturas híbridas o basadas en capas

a) Arquitecturas deliberativas

Emplean modelos de representación simbólica del conocimiento, como por ejemplo la teoría de la planificación. Emplean un modelo simbólico del entorno donde se encuentran, en el cual, toman decisiones para lograr sus objetivos a través de mecanismos de razonamiento lógico y manipulación simbólica.

Los agentes de las arquitecturas deliberativas parten de un estado inicial y tienen la habilidad de elaborar planes para lograr sus objetivos establecidos (Maes, 1989).

b) Arquitecturas reactivas

(Herrero, 2015) destaca la arquitectura de subsunción o inclusión de Rodney Brooks (Brooks, 1986). Según este autor, Brooks define el comportamiento de la arquitectura a través de una red de módulos de “sensado” y activación motora orientados a la realización de una tarea específica. Estos módulos son implementados en base a máquinas de estados finitos aumentadas (Augmented Finite State Machine - AFSM), que pueden interactuar con otros módulos.

c) Arquitecturas Híbridas

(Palomares, 2010) indica que, para tratar de solucionar las debilidades de las arquitecturas deliberativas y reactivas, se ha propuesto una arquitectura que combina las características de ambos modelos. Los SMA implementados en arquitecturas híbridas implementan dos subsistemas; el deliberativo, que emplea un modelo simbólico del entorno para planificar y el reactivo, orientado en reaccionar ante información proveniente del entorno. Según (Ferguson, 1991), estas arquitecturas son adecuadas para ser estructuradas mediante capas:

- Vertical: una sola capa tiene acceso a los sensores y actuadores.
- Horizontal: todas las capas tienen acceso a los sensores y actuadores.

2.3.4.6. Organización social de los agentes

(Mallcu, 2008) explica la organización social de los agentes como la manera como estos están constituidos en un instante determinado. La organización social es estática o dinámica y está relacionada con la arquitectura de sus componentes, características, roles, responsabilidades, necesidades y la forma como estos componentes se comunican.

Los agentes necesitan de habilidades sociales y comportamiento interactivo para comunicarse, cooperar, coordinar y negociar entre ellos. Una sociedad de agentes está constituida por tres elementos:

- Un grupo de agentes
- Un conjunto de tareas a realizar
- Un conjunto de recursos

- a) **Coordinación:** La coordinación en el ámbito de los SMA, se refiere a la distribución de tareas fin de alcanzar objetivos comunes (Dignum et al, 2005). Tal distribución se consigue cuando los agentes pueden seleccionar las acciones apropiadas, las veces adecuadas y en la secuencia correcta (Chen et al., 2004).

- b) **Cooperación:** Los sistemas cooperativos multiagente son aquellos en los que varios agentes intentan, a través de su interacción, resolver de manera conjunta tareas o maximizar su utilidad. Debido a las interacciones entre los agentes, la complejidad de un SMA puede aumentar rápidamente con el número de agentes o su sofisticación de comportamiento. El desafío que esto representa en la labor de programación de soluciones a problemas de SMA ha dado lugar a un creciente interés en las técnicas de aprendizaje automático, para automatizar el proceso de búsqueda y optimización. (Liviu et al., 2005).

- c) **Comunicación:** La comunicación entre los agentes implica cooperación, coordinación y negociación que se dan entre los agentes (ACA, 2012). Permite a los agentes coordinar acciones, transmitir y recibir información, así como, resolver conflictos en la solución de un problema, etc.

2.3.4.7. Foundation for Intelligent Physical Agents

(FIPA, 2018) es un comité de la IEEE que promueve tecnologías basadas en agentes a través de normas y propuestas de estándares. FIPA, como organización para la estandarización de agentes y SMA fue formalmente oficializado por el IEEE en el 11 comité de normas, en el año 2005. FIPA se constituyó originalmente como organización para producir estándares de software, especificaciones para los agentes que interactúan entre sí y SMA.

FIPA ha desempeñado un rol importante en el desarrollo de estándares para agentes, y ha fomentado iniciativas y foros que han contribuido al desarrollo y la aceptación de la tecnología de agentes. Además, algunas de las ideas que nacieron y que desarrolló FIPA han llegando ahora al primer plano en las nuevas implementaciones de la tecnología de web y especificaciones relacionadas (FIPA, 2012).

2.3.4.8. Arquitectura abstracta de FIPA

El objetivo principal de la “Arquitectura Abstracta” de FIPA es crear significado semántico a los mensajes intercambiados entre los agentes que puedan estar utilizando diferentes medios de transporte de mensajería, diferentes lenguajes de comunicación de agentes, o diferentes lenguajes de contenido (FIPA, 2002). El ámbito de aplicación de la arquitectura abstracta FIPA incluye un modelo de servicios y mecanismos para descubrir servicios disponibles para agentes y otros servicios, protocolos para el transporte de mensajes, soporte a varios formatos de representación de lenguajes de comunicación de agentes, soporte a varios formatos de lenguaje de contenido y, soporte para la implementación de servicios de directorio.

En la “Arquitectura Abstracta” se definen un conjunto de elementos comunes a los sistemas subyacentes y los reúne en una especificación común. Los desarrolladores en base a cualquier lenguaje de programación pueden implementar o realizar una implementación concreta, que al cumplir con “Arquitectura Abstracta” puede ser interoperable con otros sistemas. Ver figura 2.9.

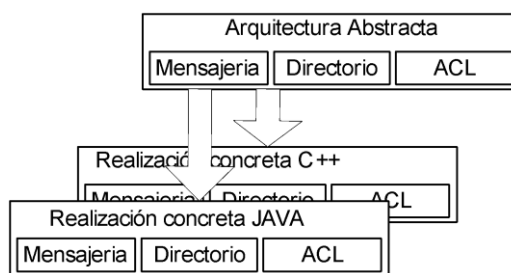


Figura 2.9.- Arquitectura abstracta FIPA

Los agentes se comunican mediante el intercambio de mensajes en un lenguaje estándar de comunicación de agentes llamado ACL. Para ello, los agentes hacen uso de servicios de soporte: el servicio de directorio y el servicio de transporte de mensajes. Estos servicios se podrán implementar como agentes o como software para ser invocado (C++, Java, etc.):

a) Servicio de Directorio: La arquitectura abstracta de FIPA existe para proporcionar un cambio común e inmutable punto de referencia para implementaciones compatibles con FIPA (Bellifemine, 2007). La función básica del servicio de directorio es proporcionar una forma consistente por el cual los agentes y los servicios pueden descubrir servicios. Operativamente, el directorio de servicios proporciona una ubicación donde se pueden registrar descripciones de los servicios como entradas. Además, los agentes y los servicios pueden buscar en el directorio de servicio los servicios adecuados a sus necesidades. Las entradas en un directorio de servicio son descripciones de los servicios que consisten en una tupla que contiene un nombre de servicio, tipo de servicio, un servicio de localizador para comunicarse con otro agente y un conjunto de servicios-atributos o propiedades asociadas a un agente que son opcionales.

b) Mensajería: Definidos en el lenguaje de comunicación de agentes (FIPA-ACL), los mensajes están formados por pares

clave-valor (Ver figura 2.10). El contenido del mensaje viene expresado por un lenguaje de contenido (típicamente SL). Opcionalmente se puede hacer referencia a una ontología que definirá los términos que pueden aparecer. Estos mensajes se encapsulan dentro de los mensajes de transporte que añadirán una cabecera con atributos de seguridad, datos de emisor y receptor, etc.

Nombre del agente :	AgenteTest
Localizador :	tipo de transporte dirección de transporte
	HTTP http://www.misagentes.com/AgenteTest
	SMTP AgenteTest @misagentes.com
Atributos de Agente :	Idiomas: Español, English
	Negociación preferida : Contract-net

Figura 2.10.- Ejemplo de entrada de directorio

2.3.4.9. Plataforma de agentes FIPA

La especificación de los agentes y su gestión proporcionan un entorno de trabajo donde los agentes existan y operan entre si a través de un modelo de referencia para la inicialización, creación, registro, ubicación, comunicación, movilidad y eliminación de agentes. La “Arquitectura Abstracta” define la plataforma de agentes que está compuesta por un grupo de entidades que ofrecen servicios (Amor, 2003). Ver figura 2.11.

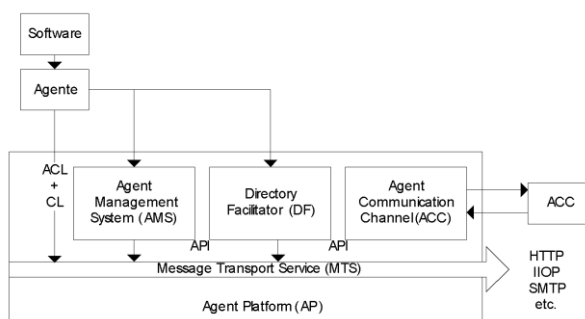


Figura 2.11.- Plataforma de Agentes FIPA

A continuación, se explican los principales componentes de la plataforma FIPA:

- a) **Agent Management System (AMS):** El AMS (FIPA, 2002) es el componente que conoce el estado de la plataforma y sus agentes. El AMS controla el ciclo de vida de los agentes en sus estados y transiciones de estados; iniciado, activo, suspendido y esperando.
- b) **Directory Facilitator (DF).** JADE implementa un agente de Facilitador de directorio (DF) según lo especificado por FIPA. El DF a menudo se compara con la guía telefónica "Páginas amarillas". Los agentes que deseen anunciar sus servicios se registran en el DF. Los agentes visitantes pueden entonces preguntar (buscar) al DF buscando agentes que brinden los servicios que desean (Vaucher & Ncho 2003). Los agentes usan el DF para registrar los servicios que ofrecen. Si un agente busca un servicio, este buscará en el DF aquel agente que pueda satisfacerlo.
- c) **Agent Communication Channel (ACC).** (Fipa, 2002) El servicio de transporte de mensajes (MTS) proporciona un mecanismo para la transferencia de mensajes ACL entre agentes. Los agentes involucrados pueden ser locales para una única plataforma o en diferentes plataformas. Cada agente tiene acceso a un ACC. Si un agente tiene la intención de enviar un mensaje a otro agente en la plataforma donde el agente se encuentra u otra plataforma remota, envía el mensaje al ACC donde el agente esta, y este a su vez se envía el mensaje al ACC de la plataforma donde se encuentra el otro agente.
- d) **Platform Message Transport (IPMT):** The internal Platform Message (IPMT) provee un servicio de enrutamiento de mensajes para los agentes en una plataforma particular (Sugumaran, 2010). IPMT implementa los mecanismos

necesarios para facilitar que dos agentes dentro de la misma plataforma puedan comunicarse entre sí.

- e) **El Agent Platform Message Transport (APMT)** cumple la misma función que el IPMT, pero entre plataformas (ACC).

2.3.4.10. Servicio de ontología

Toda la plataforma FIPA se basa en la comunicación entre agentes mediante ACL. Un mensaje ACL consiste básicamente en una expresión de un determinado lenguaje y con un conjunto de términos específicos de la ontología usada. Esto asegura que estos agentes atribuyen el mismo significado a los símbolos utilizados en los mensajes. Una ontología es un conjunto de palabras asociadas con su significado. Gracias a la ontología, los agentes pueden entenderse entre sí. Cuando un agente se registra en el DF (Directory Facilitator) este hace publica las ontologías que conoce. Cuando otro agente desea comunicarse con este agente, utilizará una de estas ontologías publicadas a fin de poder tener un entendimiento mutuo (Corchado, 2005).

2.3.4.11. Movilidad de agentes en la plataforma

En el 2000 un nuevo set de especificaciones FIPA fue publicado, el cual se incluye el FIPA 00087 para la movilidad de agentes (Braun et al, 2005). La movilidad de agentes especificada por FIPA permite a las implementaciones integrar estas capacidades dentro de la plataforma.

2.3.5. Metodología Ingenierías para el desarrollo de software orientado a agentes

En forma genérica una metodología es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software (Piattini,

1996). Una metodología de software sigue al menos un modelo de ciclo de vida, el cual indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto, pero no cómo hacerlo.

El paradigma de “Agentes Inteligentes” va un paso más allá de donde llegan otros paradigmas de programación de sistemas, en lo que a nivel de abstracción y versatilidad se refiere. Se plantea como una evolución lógica en sucesión con los anteriores paradigmas de construcción de software, traspasando los límites de la orientación a objetos. Todo aumento en el nivel de abstracción implica un incremento del nivel de complejidad de los conceptos que se manejan, puesto que estos conceptos engloban a una mayor cantidad de otros conceptos previamente definidos. Los Agentes Inteligentes, claro está, no son una excepción. Por lo tanto, dado que suponen una mayor complejidad que los paradigmas anteriores, se hace necesaria la creación de nuevas metodologías de diseño y herramientas de trabajo que nos permitan lidiar con la nueva problemática de forma eficaz y eficiente. Así pues, las metodologías tradicionales (Unified Process, CommonKADS, etc.) no están preparadas para el diseño de las arquitecturas orientadas a agentes, como pueden ser la especificación de planificación de tareas, la comunicación mediante lenguajes específicos, la movilidad del código o la motivación de los componentes del sistema (los agentes) (Gómez Sanz, 2003).

Debido a que las metodologías tradicionales (como UML) no están preparadas para el diseño de arquitecturas orientadas a agentes, diversos investigadores de las ciencias de la computación, han propuesto metodologías derivadas de sus propias investigaciones. Existen metodologías que intentan reutilizar los conceptos de programación orientada a objetos hacia los agentes, otras intentan diseñar sistemas multiagente en base a estructuras organizacionales, etc. Actualmente no existe un consenso o

metodología estándar para el análisis y diseño de sistemas multiagente, por lo que, los desarrolladores deben escoger aquella metodología que se adapte mejor a sus necesidades, incluso pueden emplearse más de una metodología en cada etapa de análisis, diseño e implementación de sistemas orientados a agentes.

El método de desarrollo de SMA especificado en la metodología INGENIAS (Gómez Sanz, 2002) entiende el SMA como la representación informática de un grupo de modelos. Cada uno de los modelos tiene una visión parcial del SMA: Los agentes propiamente dichos, las interacciones entre los agentes, cómo se organizan los agentes para proporcionar la funcionalidad del sistema, qué información es importante en el dominio y como es el entorno en que se ubicará el SMA a desarrollar. Para facilitar la representación informática de los modelos asumimos que la implementación se realiza parametrizando un armazón de software. Para especificar cómo deben de ser estos modelos, se definen los meta-modelos. Se define un meta modelo como una representación de los tipos de entidades que pueden existir en un modelo, sus relaciones y restricciones de aplicación (Gómez Sanz, 2002, p. 29).

Ingenias define cinco meta-modelados que giran alrededor de dos identidades: Organización y Agente. La figura 2.12 muestra las relaciones entre los meta modelos ingenias.

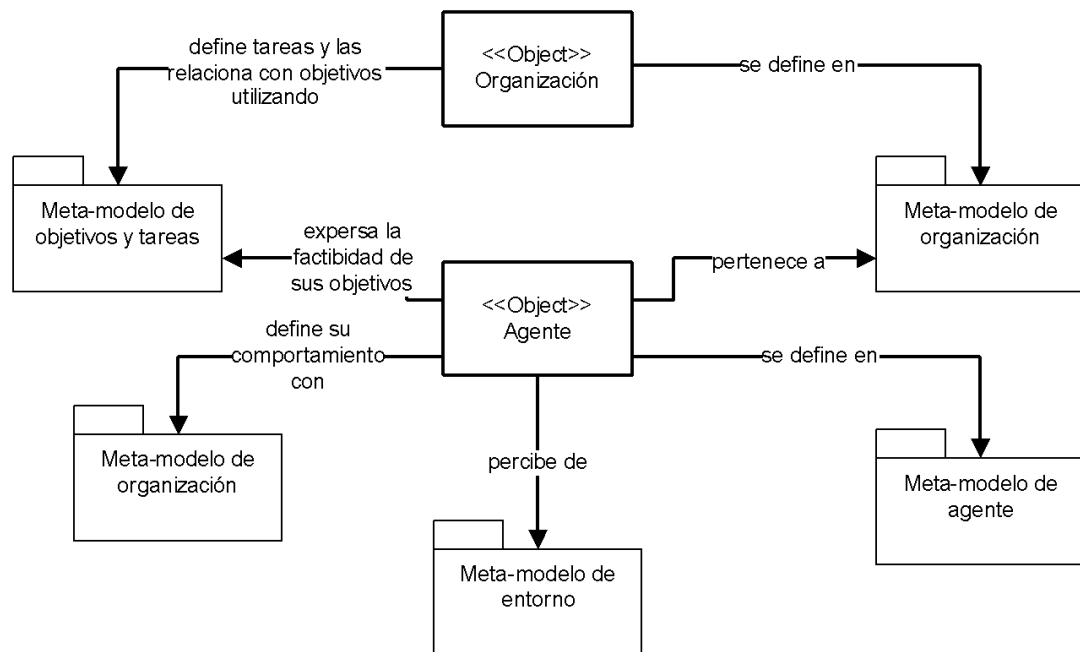


Figura. - 2.12. Relaciones entre los meta modelos. Fuente (Gómez Sanz, 2002, p. 30)

A continuación, se resume la funcionalidad de cada uno de los modelos de la estrategia metodológica a seguir:

2.3.5.1. Meta-modelo de Organización

El modelo de organización es la representación de la arquitectura del SMA. El modelo de organización se asemeja a las organizaciones humanas (ver figura 2.13); sus flujos e interacciones. Este meta-modelo refleja con detalle cómo coordinan los participantes, dando un punto de vista amplio de los objetivos que se persiguen globalmente (Gómez Sanz, 2002, p. 95). La importancia de este modelo radica en que define restricciones en el comportamiento de los agentes mediante relaciones como la de subordinación permitiendo que el diseñador asegure que unos agentes harán lo que dicen otros agentes o que se comprometerán a ejecutar ciertas tareas bajo demanda de tareas respetando sus prioridades.

La siguiente figura muestra el meta-modelo de organización, donde se muestran tres aspectos interrelacionados del mismo (Gómez Sanz, 2002, p. 96).

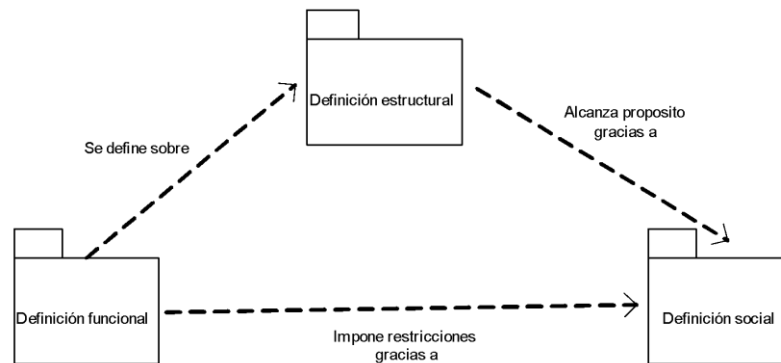


Figura 2.13.- Tres vistas de una organización. Fuente (Gómez Sanz, 2002, p. 97)

La estructura de la organización define los componentes que constituyen las relaciones sociales y la organización a alto nivel que se establecen entre los componentes para poner limitaciones al comportamiento de la organización. En forma más detallada tenemos (Gómez Sanz, 2002, p. 97-102):

- **Descripción Estructural:** La descripción estructural es el armazón donde están los agentes, recursos y propósitos (Ver figura 2.14). Las entidades están asociadas mediante relaciones de agregación y herencia. También se definen relaciones que inducen la formación de flujos de trabajo y restricciones sociales.

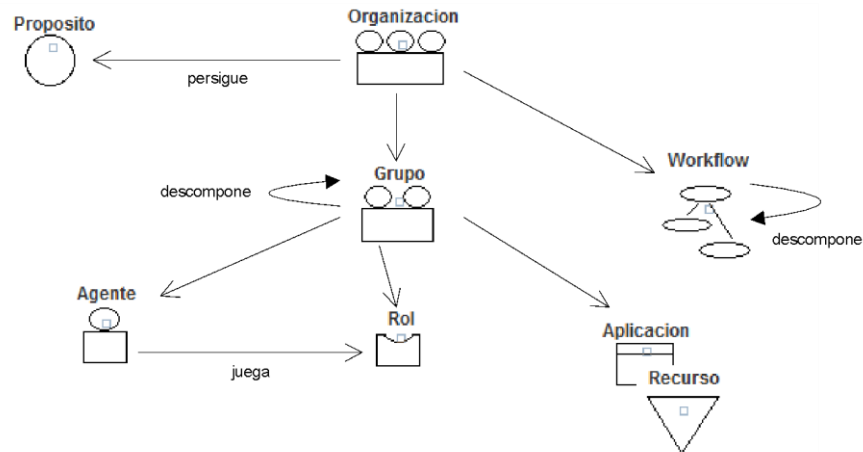


Figura 2.14.- Descripción estructural. Fuente (Gómez Sanz, 2002)

- Descripción Social:** A través de restricciones, se establece las limitaciones entre la interacción de organizaciones, grupos y agentes (Ver figura 2.15). La descripción social establece las relaciones de:
 - Subordinación: El subordinado obedece lo que se le ordena. La obediencia incondicional acepta cualquier tipo de orden y la condicional acepta las ordenes en forma similar un contrato, donde si este se incumple el contrato deja de tener validez.
 - Prestación de un servicio: Referencia un servicio ofrecido.
 - Cliente de un servicio: Cliente que busca un servicio determinado.

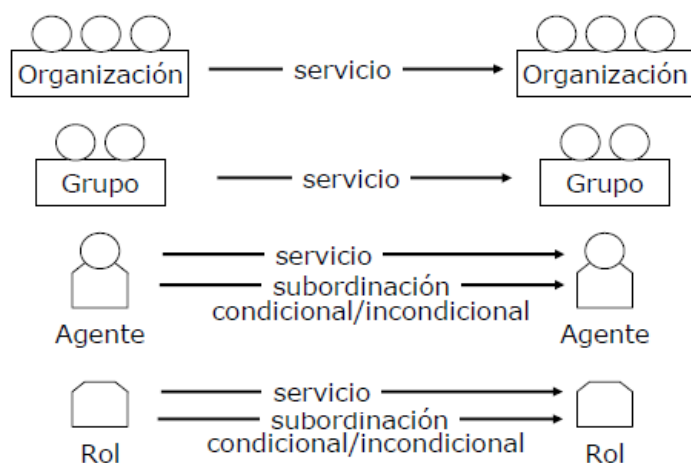


Figura 2.15.- Descripción social. Fuente (Gómez Sanz, 2002)

• Descripción Funcional

La descripción funcional dispone que es lo que ofrece la organización, cómo se asignan los recursos, qué tareas se requieren para lograr los objetivos y quienes son los responsables de ejecutarlas. El flujo del trabajo presenta cómo se relacionan unas tareas con otras y cómo estas se ejecutan. La figura 2.16 muestra las relaciones funcionales de la organización.

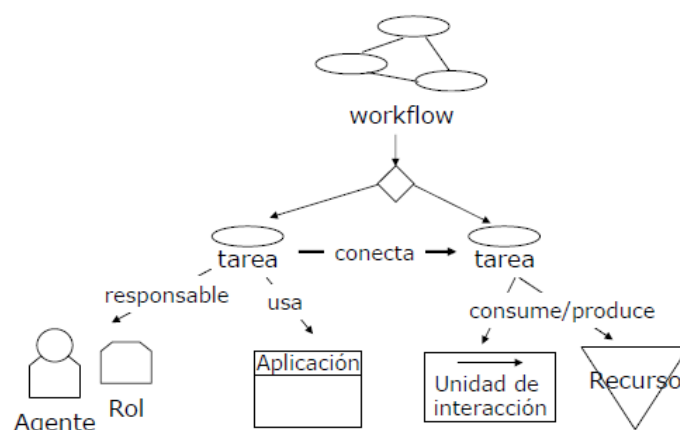


Figura 2.16.- Descripción funcional. Fuente (Gómez Sanz, 2002)

2.3.5.2. Meta-modelo de Agente

El meta modelo de agente define agentes individuales sin tomar en cuenta las interacciones con otros agentes, en este sentido,

un agente es una entidad autónoma que tiene una identidad propia y propósitos. Como entidad autónoma, un agente persigue objetivos, alcanzándolos mediante sus roles y ejecutando tareas. Como ente racional, el agente juega un rol y realiza las acciones que conllevan a lograr sus objetivos. Los objetivos se alcanzan cuando el entorno se ha modificado debido a las tareas efectuadas por el agente. La modificación del entorno produce evidencias, que pueden ser nuevos estados mentales del agente o la modificación del entorno donde el agente actúa (Gómez Sanz, 2002, p. 39 & 46). La figura 2.17 muestra las relaciones involucradas en el comportamiento del agente.

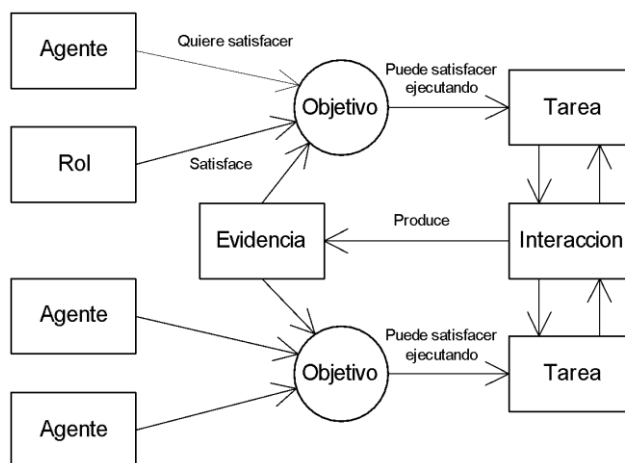


Figura 2.17.- Comportamiento del agente. Fuente (Gómez Sanz, 2002)

El estado mental de un agente puede concebirse como toda aquella información que emplea el agente para la toma de decisiones. El estado mental es dinámico y cambia en el tiempo, por lo tanto, el gestor de estado mental haciendo uso de operaciones de creación, monitoreo, modificación y destrucción de la inteligencia del agente mantiene la coherencia del conocimiento y lo hace evolucionar. El agente toma decisiones a través del Procesador de Estado Mental. Las decisiones pueden tomarse mediante varias técnicas: por algoritmos, en función de tareas y beneficios, etc. La figura 2.18 muestra los elementos del meta-modelo de agente,

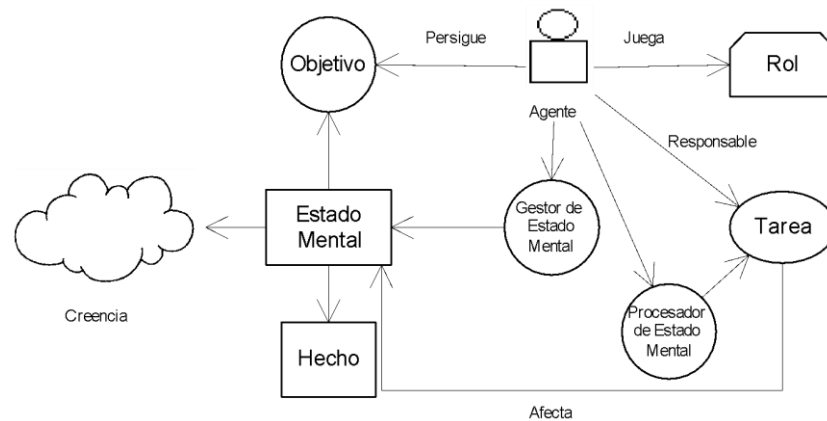


Figura 2.18.- Elementos del meta-modelo de agente. Fuente (Gómez Sanz, 2002)

2.3.5.3. Meta-modelo de Objetivos y Tareas

Este meta-modelo define lo siguiente (Gómez Sanz, 2002, p. 82-83):

- Las acciones definidas en los meta-modelos de organización, agente e interacciones.
- Define las consecuencias de ejecutar las tareas y por qué se deben llegar a ejecutar.

En el meta-modelo se define la relación entre los objetivos, los agentes, los roles y las organizaciones. Una organización es un conjunto de agentes que se comporta como una entidad autónoma que persigue objetivos. Los agentes, por poder aparecer en los flujos de trabajo y ser una entidad autónoma, como la organización, pueden asociarse con cualquiera de los dos.

Una tarea se puede ver como una acción transformadora del estado global e incluir tareas en el meta-modelo cumple a que juegan un rol importante en la evolución del estado mental de los agentes. La figura 2.19 muestra los elementos en la definición de las tareas.

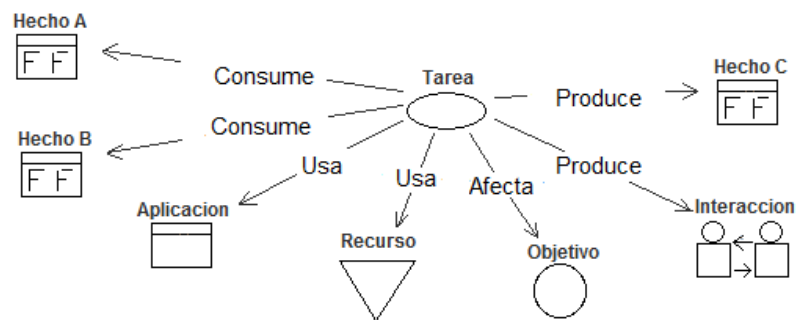


Figura 2.19.- Elementos en la definición de tareas. Fuente (Gómez Sanz, 2002)

Tal como se observa en la figura 2.20, las tareas afectan a los objetivos, satisfaciendo o fallando dichos objetivos, siendo los objetivos entidades de control, porque indican al agente cuál es su motivación y compromiso.

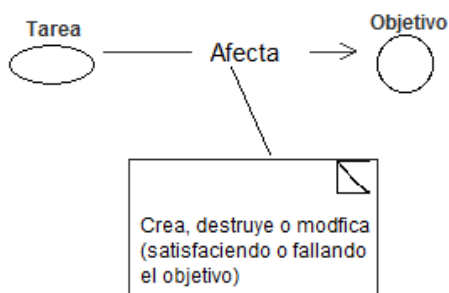


Figura 2.20.- Relaciones entre tareas y objetivos. Fuente (Gómez Sanz, 2002)

2.3.5.4. Meta-modelo de Interacciones

El comportamiento del agente está determinado por el meta-modelo de interacción. Según el autor (Gómez Sanz, 2002, p. 63-69), el comportamiento del agente está en función de sus objetivos y por ende de las tareas a ejecutar. Esto se refleja en la reacción del agente cuando actúan sobre él a través de interacciones. El meta-modelo de interacciones describe cómo coordinan y se comunican los agentes del sistema. Este meta-modelo implica los actores del sistema, las unidades de interacción, los protocolos y cómo estos afectan la el comportamiento entre los agentes. El meta-modelo de

interacción se implementa sobre agentes, interacciones, roles, objetivos y unidades de interacción. Los agentes y roles son los componentes de las interacciones. Durante una interacción se ejecutan unidades de interacción (intercambio de mensajes) en las que existe un iniciador (emisor) y uno o más colaboradores (receptores).

Estos elementos se relacionan a través de un conjunto de asociaciones, las cuales se categorizan en cuatro grupos: naturaleza, contexto, ejecución y notación. (ver Figura 2.21).

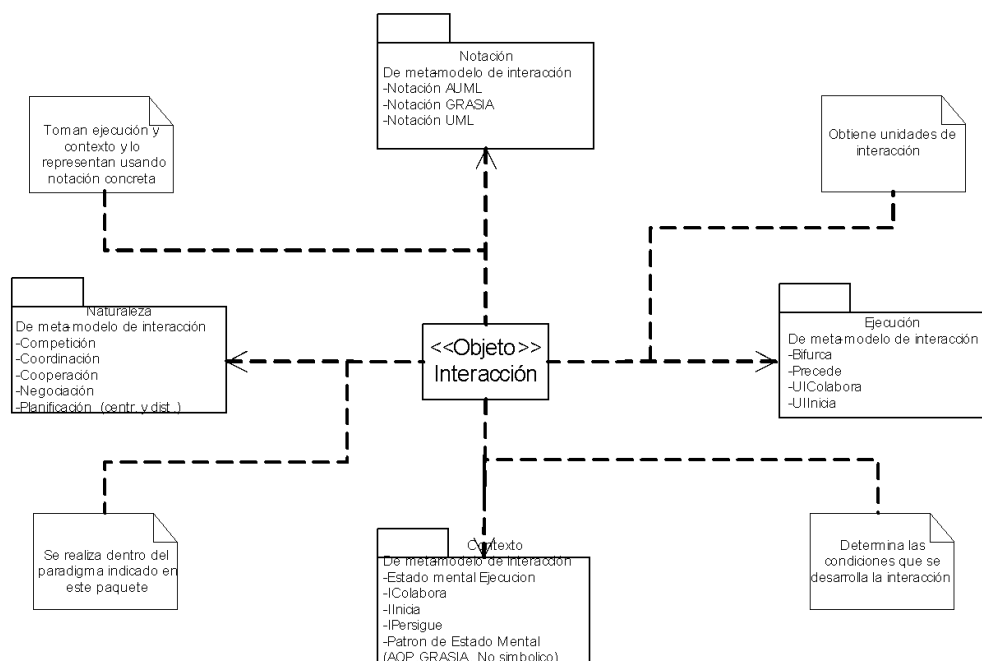


Figura 2.21.- Relación entre los diferentes aspectos de la interacción. Fuente (Gómez Sanz, 2002)

2.3.5.5. Meta-modelo de Entorno

El entorno define las entidades o relaciones que van a interactuar con el SMA. El agente recibe información del entorno donde se encuentra y ejecuta acciones sobre el mundo que le rodea a través de operaciones que son definidas en los propios agentes.

(Gómez Sanz, 2002, p. 112) explica que el propósito del meta modelo de entorno es clasificar el tipo de entidades en el entorno y restringir la interacción con ellas. De esta forma, el entorno contiene sólo recursos, aplicaciones y agentes, y se limita la percepción y actuación de los agentes. Los mecanismos de percepción de los agentes están en función de cómo se describen las entidades del sistema. Este meta modelo define las relaciones de los agentes con entidades y los tipos de mecanismos utilizados para percibir datos del entorno. Las acciones realizadas en el entorno son las operaciones que se definen sobre estas entidades.

Recursos: Los recursos son objetos pueden pertenecer a un agente o a ninguno. Los recursos pueden ser consumibles o no consumibles. Los recursos consumibles son aquellos que se agotan, pero posteriormente pueden ser restablecidos. Los recursos no consumibles se auto restituyen después de ser realizada una acción sobre estos. La capacidad de los recursos se define con tres atributos de tipo real, entero y tipo compuesto. Los atributos son nivel de uso mínimo, estado y nivel de sobrecarga. La utilización de un recurso hace que su estado decrezca hasta el nivel de uso mínimo, después del cual el recurso se inhabilita. Otro ejemplo de inhabilitación es el uso excesivo del recurso, y sucede cuando se incrementa su estado por encima del nivel de sobrecarga. Las tareas se relacionan con los recursos mediante tres relaciones de consumo, producción y limitación. La relación de producción hace que una tarea restablezca un recurso. La de limitación impone pre condiciones para lanzar una tarea en función de la disponibilidad de recursos. Finalmente, el consumo indica utilización de un recurso, lo que conlleva al decremento del estado del recurso. La capacidad de razonamiento de los agentes está limitada por los recursos disponibles. Ejemplos de recursos son: capacidad de procesador, memoria, descriptores de archivos, número de

procesos o hebras requeridos, ancho de banda, número de conexiones necesarias con bases de datos, etc (Gómez Sanz, 2002, p. 112).

2.3.6. El Ciclo de Vida

Desde el punto de vista de ingeniería, los meta modelos sirven de guía, y muestran cómo se puede estructurar el desarrollo de SMA. Para analizar la aplicación de los meta modelos a procesos de desarrollo de software, se han definido los meta-modelos como un lenguaje de especificación para el desarrollo de SMA. INGENIAS integra los meta-modelos en el Rational Unified Process (RUP) en las fases de análisis y diseño, por lo tanto, su ciclo de vida es iterativo e incremental, emplea casos de uso con el fin de determinar la funcionalidad del sistema e interacciones entre agentes para emular diagramas de colaboración y de secuencia. Para facilitar la integración, se plantean una serie de asociaciones entre elementos del RUP y elementos de los meta modelos (Gómez Sanz, 2002, p. 123-126). El cuadro 2.1 muestra la relación entre las entidades RUP y MAS GRASIA.

Tabla 2.3- Asociaciones entre los elementos RUP y entidades MAS GRASIA

Entidad MAS GRASIA	Entidad RUP
Agente	Clase
Organización	Arquitectura
Grupo	Subsistema
Interacción	Escenario
Roles, tareas y flujos de trabajo	Funcionalidad

Fuente: (Gómez Sanz, 2002, p. 126).

El agente tiene como equivalencia a la clase, similar como en la clase, define tipos. En ese sentido, un agente en ejecución equivale un objeto en RUP. La organización es equivalente a la arquitectura

en RUP debido a que abarca toda la estructura. La organización es la funcionalidad del SMA como un todo y permite tener una visión global agrupando agentes, recursos roles, aplicaciones, interacciones. El grupo es un sub conjunto definido de la organización, su equivalente con RUP radica en que se utiliza para organizar elementos en unidades de abstracción mayores. Además, define un conjunto de interfaces para interaccionar los roles.

La interacción es similar a los diagramas de colaboración y secuencia de RUP. Finalmente, los roles, las tareas y flujos de trabajo dan el encapsulamiento de acciones que en el RUP dan interfaces y métodos.

En RUP, el análisis y diseño se encuentra definido en las fases de inicio, elaboración y construcción. Dentro de cada fase se desarrollan un conjunto de iteraciones que son ciclos de desarrollo que incluyen análisis, diseño, implementación y pruebas, que van construyendo gradualmente el sistema. En el cuadro 1 se muestra la adaptación de las fases de RUP a los meta modelos de metodología INGENIAS.

Tabla 2.4.- Adaptación de las fases de RUP a los meta-modelos

		FASES		
FLUJOS DE TRABAJO FUNDAMENTALES		Inicio	Elaboración	Construcción
	Análisis	- o Generar casos de uso e identificar realizaciones de los casos de uso con modelos de interacciones. - o Esbozar la arquitectura con un modelo de organización. - o Generar modelos del entorno para trasladar la captura de requisitos a los modelos	- o Refinar casos de uso. - o Generar modelos de agente para detallar los elementos de la arquitectura. - o Continuar con los modelos de organización identificando flujos de trabajo y tareas. - o Modelos de tareas y objetivos para generar restricciones de control (objetivos principales, descomposición de objetivos) - o Refinar modelo de entorno para incluir nuevos elementos.	o Estudiar resto de casos de uso.
	Diseño	o Generar un prototipo con herramientas de prototipado rápido, como ZEUS o Agent Tool	- o Centrar el modelo de organización en el desarrollo de flujos de trabajo. - o Llevar las restricciones identificadas a modelos de tareas y objetivos para dar detalles acerca de las necesidades y resultados de las tareas y su relación con los objetivos del sistema. - o Expresar la ejecución de tareas dentro de modelos de interacción. - o Generar modelos de agente para detallar <i>patrones de estado mental</i>.	- o Generar nuevos modelos de agente o refinar los existentes. - o Depurar la organización centrando el desarrollo en las relaciones sociales.

Fuente: (Gómez Sanz, 2002, p. 126)

Las etapas de implementación y pruebas no se han incluido. La fase de pruebas es igual a la del software convencional. En cuanto a la implementación, se puede generar código XML en forma automática a partir de la correcta especificación de cada uno de los meta modelos. Con ello se consigue que sea independiente de la plataforma de agentes o del lenguaje de agentes utilizado. También es posible, en esta etapa, que el desarrollador pueda actuar como en desarrollos usuales, diseñando elementos computacionales que realicen lo que se ha especificado.

2.3.7. Ingenias Development Kit

INGENIAS Development Kit (IDK) es un marco para el análisis, diseño e implementación de sistemas multiagente. Se basa en la especificación de los meta modelos de la metodología Ingenias. IDK dispone de un editor donde el diseñador modela en forma gráfica cada uno de los meta modelos de la metodología Ingenias y a partir de ahí, la herramienta realiza en forma automática la generación de código.

IDK es la herramienta de soporte de desarrollo para metodología INGENIAS (figura 2.22). Se distribuye como software GPL y está desarrollado en Java. La distribución se puede descargar de <http://ingenias.sourceforge.net>. Mientras se actualiza, las versiones siempre están disponibles en los repositorios de subversión alojados en el proyecto sourceforge de INGENIAS. Esta herramienta fue creada en el año 2002 como parte de una tesis sobre el modelado de sistemas multiagente. El proyecto es bastante activo y se ha originado alrededor de 5800 descargas desde su creación (Gómez Sanz, et al, 2015).

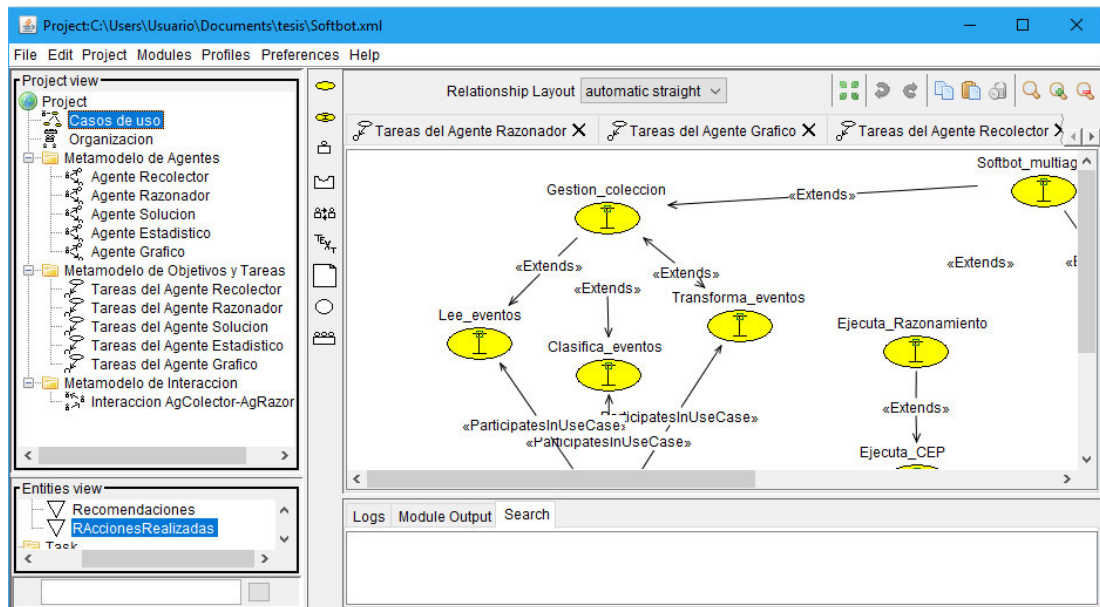


Figura 2.22.- Ejemplo de meta-modelado realizado con IDEK

2.3.8. Framework para sistemas multiagente (JADE)

JADE (Java Agent DEvelopment Framework) es un software marco implementado en el lenguaje de programación JAVA. JADE facilita el desarrollo de sistemas multiagente a través de un middleware que es compatible con las especificaciones de FIPA y, a tiene una interfaz gráfica y un set de herramientas que soportan fases de debugging y deployment. La plataforma de agentes puede ser distribuida a través de máquinas, las cuales no necesariamente pueden tener el mismo sistema operativo. La configuración puede ser administrada a través de una interface remota GUI y esta puede ser cambiada en tiempo real moviendo los agentes de una máquina a otra cuando sea requerido. JADE está completamente implementado en lenguaje JAVA y los requerimientos mínimos del sistema es la versión 1.4 de JAVA (el entorno runtime de JDK) (Telecom Italia Lab, 2019). La figura 2.23 muestra la plataforma JADE y su relación con equipos de red.

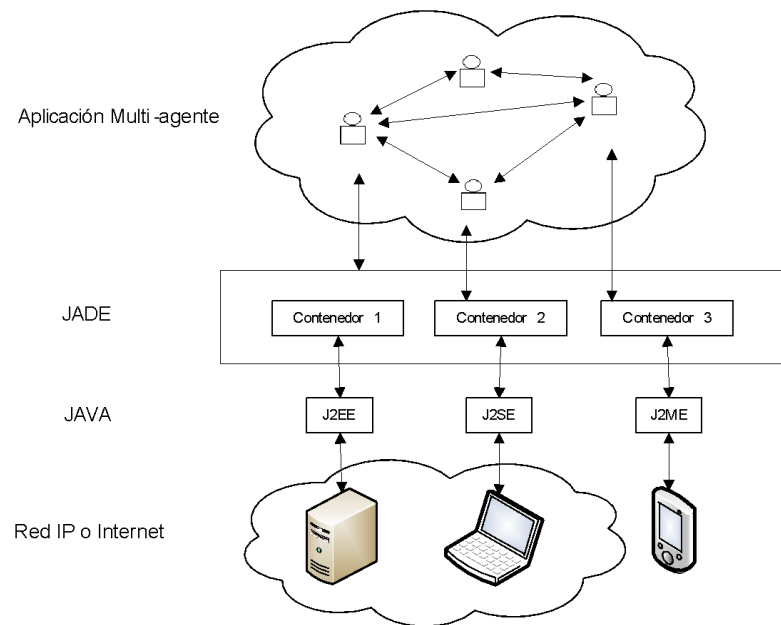


Figura 2.23.- Plataforma JADE y Agentes

JADE es software libre, y es distribuido por Telecom Italia, los derechos de autor del software están en los términos de la licencia LGPL (Lesser General Public License Version 2). Desde mayo de 2003, JADE Board ha sido supervisado por los managers del proyecto JADE. Actualmente en el JADE Board participan 5 miembros: Telecom Italia, Motorola, Whitestein Technologies AG, Profactor GmbH, and France Telecom R&D. La figura 2.24 muestra el front end de la plataforma JADE.

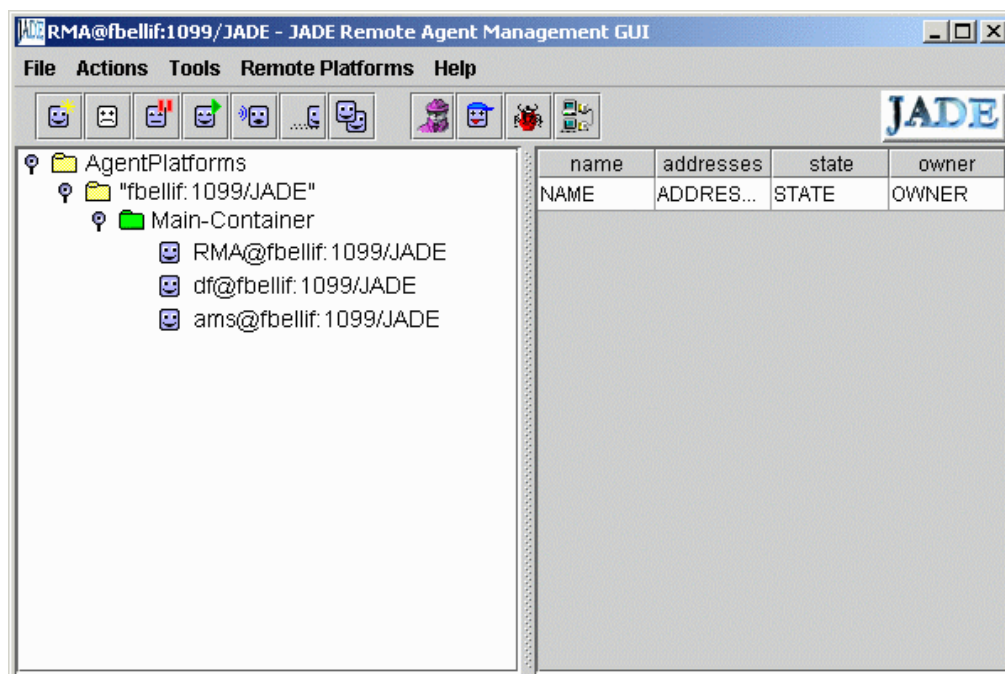


Figura 2.24.- Ejemplo del front end del framework JADE

El framework Jade tiene un conjunto de librerías en JAVA que permiten programar agentes en cualquier sistema operativo que soporte JAVA. El entorno de ejecución permite tener un ambiente donde los agentes vivan y se comuniquen. Está implementado íntegramente en JAVA y proporciona una serie de herramientas que facilitan al desarrollador validar y depurar agentes en tiempo real (Garamendi, 2004).

2.3.8.1. Características de Jade

Este framework posee las siguientes características:

- a) **P2P (Arquitectura peer-to-peer):** Un agente puede iniciarla la comunicación o responder a peticiones que le hagan otros agentes.
- b) **Interoperabilidad:** JADE es compatible con las especificaciones de FIPA, esto quiere decir que los agentes implementados en JADE pueden interactuar con

agentes desarrollados bajo otras plataformas diferentes que JADE que cumplan las especificaciones FIPA.

- c) **Portabilidad:** La API que proporciona JADE no depende de plataforma sobre la que se va a desarrollar, no de la versión de JAVA empleada. Se puede usar bajo las API JEE, JSE y JME.
- d) **Intuitiva:** Las API de JADE ofrecen un uso sencillo, fácil de aprender y fácil de manejar.
- e) **Unicidad de nombres:** Cada agente JADE tiene la habilidad de descubrir y comunicarse con otros agentes a través de nombres únicos.
- f) **Servicios:** Los agentes publican sus servicios para que otros agentes busquen y puedan consumir dichos servicios
- g) **Comunicación:** Los agentes se comunican entre sí a través de mensajes asíncronos. Esto significa que el agente que envía el mensaje y el agente que debe recibir el mensaje no necesariamente puede estar disponible en ese momento, inclusive el agente destinatario no necesariamente puede existir en ese momento. El mensaje se puede enviar a un solo agente en particular o se puede enviar a múltiples agentes que se desconocen, pero se sabe que tienen ciertas funcionalidades.
- h) **Seguridad:** Mediante mecanismos de seguridad que JADE proporciona, se puede implementar agentes que no se les está permitido comunicarse con otros agentes, así la aplicación puede verificar la identidad del agente que

recibe el mensaje, así como, del agente que envía el mensaje, y no permitir intercambio de datos no permitido.

- i) **Cumple con FIPA:** La estructura para el intercambio de mensajes está basada en el lenguaje ACL (Agent Communication Language) que ha sido especificado por FIPA.
- j) **Movilidad:** JADE permite a los agentes moverse de un host a otro. El agente puede interrumpir su ejecución en un momento dado, moverse hacia otro host y continuar la ejecución en la misma parte de tarea que se interrumpió. Esta característica permite realizar balanceo de carga debido a que al agente se puede mover a otros hosts menos cargados. En resumen, JADE proporciona un entorno multiagente el cual se puede ejecutar los agentes y herramientas para monitoreo y debugging.

2.3.8.2. Arquitectura de JADE

Gracias a JADE, los agentes disponen de un entorno donde estos puedan ejecutarse. JADE permite crear instancias llamadas containers donde se ejecutan los agentes.

Un conjunto de los containers se le denomina platform o plataforma. La plataforma proporciona una capa que oculta al desarrollador que libera a este de tener que lidiar con temas de bajo nivel. Cada plataforma debe tener un contenedor especial llamado contenedor principal (main container). La característica principal que diferencia al contenedor principal del resto de contenedores es que este contiene dos agentes especiales:

- a) **AMS (Agent Management System):** Es un agente especial que proporciona un servicio de nombres de tal forma que se

asegure que el agente tenga un nombre único. Este agente permite crear y matar agentes en contenedores remotos.

b) DF (Directory Facilitator): Es un agente especial que proporciona el servicio de Páginas Amarillas. Un agente puede buscar y encontrar otros agentes que proporcionen los servicios necesarios para lograr sus objetivos.

La figura 2.25 muestra la arquitectura de JADE, donde se observa tres plataformas diferentes, con un contenedor principal y dos contenedores remotos.

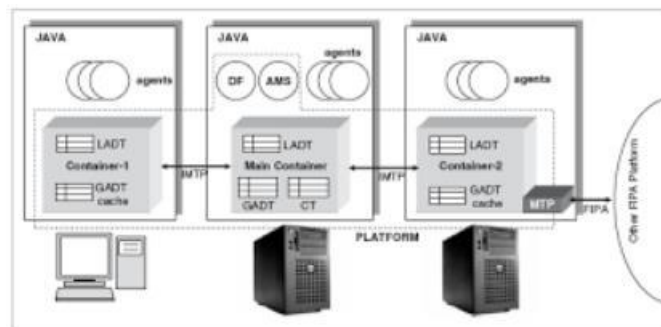


Figura 2.25.- Esquema de distribución de containers y plataformas

2.3.8.3. Programación de Agentes en JADE

Las siguientes funcionalidades son proporcionadas por JADE al desarrollador:

- Los agentes de ejecutan en forma distribuida.
- Posee una interfaz gráfica con herramientas de monitoreo y debugging de agentes, tanto en host local como remotos.
- Permite instanciar agentes móviles.
- Permite la ejecución de tareas en paralelo.
- Cumple con las especificaciones FIPA.
- Permite el intercambio de mensajes ACL entre agentes.

- Permite el registro automático de agentes a través del agente especial AMS.
- Dispone de un servicio de nombres a través del agente DF.

2.3.8.4. Complex Event Processing

El crecimiento explosivo de las redes genera eventos complejos. La complejidad de estos eventos está dada porque estos provienen de distintas fuentes y generalmente tienen formatos distintos que deben ser re formateados antes de poder procesarlos.

Los sistemas de información controlados por eventos exigen un procesamiento sistemático y automático de los eventos. El Complex Event Processing (CEP) abarca métodos, técnicas y herramientas para procesar eventos mientras ocurren, es decir, de manera continua y oportuna. CEP deriva valioso conocimiento de alto nivel de eventos de nivel inferior; este conocimiento toma la forma de los llamados eventos complejos, es decir, situaciones que solo pueden reconocerse como una combinación de varios eventos (Eckert & Bri, 2009).

CEP consiste de un conjunto de técnicas que se aplican para el tratamiento de muchos eventos que se producen a una tasa muy alta y entre los cuales puede existir cierta correlación con los eventos en actuales o con los eventos ocurridos en el pasado. Al procesamiento de eventos tradicional se agrega la capacidad de que éstos pueden provenir de muchas fuente, donde, se pueden definir patrones de comportamiento, otros eventos más complejos o patrones poco evidentes. Para dar un ejemplo concreto, se pueden estar procesando eventos que indican, llamadas a un call center, alarmas en equipos de comunicaciones, variaciones el en precio de acciones, etc. (Gavilán, 2018).

Según (Etzion & Niblett, 2009) CEP implica una cantidad de técnicas, en las que destacan:

- Detección de patrones de eventos
- Abstracción de eventos
- Filtrado de eventos
- Agregación y transformación de eventos.
- Modelado de eventos
- Detectar relaciones (como causalidad, pertenencia o sincronización) entre eventos

a) Detección de patrones de eventos: La correlación de patrones se obtiene a partir de las actividades de detección, clasificación y extracción de características, en el cual, cada objeto es representado por una colección de descriptores. El sistema de reconocimiento debe asignar a cada objeto su categoría conjunto de entidades que comparten alguna característica que la diferencia del resto.

b) Abstracción de eventos: Las operaciones que forman la lógica de procesamiento de eventos se pueden separar de la lógica de la aplicación, lo que permite modificarlas sin tener que cambiar las aplicaciones productoras y consumidoras. Es similar a un sistema experto donde el motor de razonamiento está separado del código de aplicación.

c) Filtrado de eventos: Muchas aplicaciones de procesamiento de eventos realizan algún tipo de filtrado de eventos. Una aplicación podría tener productores de eventos, como sensores o fuentes de datos, que generan una gran cantidad de eventos, no todos estos eventos son relevantes, por lo que necesita filtrar los irrelevantes.

- d) **Agregación y transformación de eventos:** Un agente de procesamiento de eventos agregado es una transformación que toma como entrada una colección de eventos y crea un solo evento derivado mediante la aplicación de una función de agregación sobre los eventos de entrada.
- e) **Modelado de eventos:** Se usa de manera opcional para analizar las relaciones entre diferentes instancias de procesos disparados por eventos con el objeto de tratar de predecir mejoras o estrategias hacia una visión más general u holística.
- f) **Detectar relaciones:** Conexión de eventos con otros eventos donde el primer evento es en parte responsable del segundo, y el segundo evento depende en parte del primero.

2.3.8.5. Event Processing Language

El lenguaje de procesamiento de eventos o EPL (Event Processing Language en inglés) es un lenguaje de consulta para Complex Event Processing (CEP) similar a SQL que contiene las cláusulas: SELECT, FROM, WHERE, GROUP BY, HAVING y ORDER BY.

EPL es un lenguaje similar a SQL. Este lenguaje se integra con la tecnología CEP para el manejo de datos en tiempo real. CEP además de procesar eventos, ofrece la posibilidad de añadir una "dimensión temporal" (Sánchez, 2014, p. 20).

Las sentencias en EPL son consultas continuas tipo SQL que se evalúan constantemente sobre el flujo de eventos que ingresan al sistema. La sintaxis general de EPL es:

```

[annotations]
[expression_declarations]
[context context_name]
[insert into insert_into_def]
select select_list
from stream_def [as name] [, stream_def [as name]] [,...]
[where search_conditions]
[group by grouping_expression_list]
[having grouping_search_conditions]
[output output_specification]
[order by order_by_expression_list]
[limit num_rows]

```

2.3.9. ESPer

Esper es un motor CEP desarrollado en lenguaje Java que permite crear aplicaciones que procesen eventos o mensajes en tiempo real. Esper permite procesar y analizar eventos complejos en base a flujos de estos. Las consultas a los flujos proporcionan un medio para implementar eventos encadenados como ventanas, uniones, agregaciones y análisis de funciones. Las consultas usan la sintaxis de EPL para implementar aplicaciones en tiempo real capaces de escuchar y enviar a la vez continuos flujos de eventos. Debido a que los flujos de eventos están disponibles en la aplicación, el desarrollador tiene la opción de crear filtros para escuchar eventos específicos de su interés o relacionados con eventos generados por un sistema específico.

Esper es un motor de código abierto en lenguajes Java y .NET bajo licencia GPL, desarrollado por la compañía EsperTech Inc. Esper se emplea en Event-Driven Architecture (EDA) que requiere procesar información en tiempo real para lanzar acciones personalizadas que se ejecutan cuando se cumplan

ciertas condiciones sobre los flujos de eventos. Esper tiene la habilidad de correlacionar miles de notificaciones por segundo.

2.3.9.1. ESPER y Java VM

Esper es un lenguaje, un compilador y un entorno de ejecución. El lenguaje Esper es el lenguaje de procesamiento de eventos (EPL). Es un lenguaje declarativo orientado a datos para tratar con eventos de alta frecuencia basados en el tiempo. EPL es compatible con el estándar SQL-92 y extendido para analizar series de eventos y de tiempo. El compilador de Esper compila el código fuente de EPL en el código de bytes de la Máquina Virtual Java (JVM) para que el código ejecutable resultante se ejecute en una JVM dentro del entorno de tiempo de ejecución de Esper. El tiempo de ejecución de Esper se ejecuta en la parte superior de una JVM. Puede ejecutar el código byte producido por el compilador de Esper usando el runtime de Esper. La arquitectura Esper es similar a la de otros lenguajes de programación que se compilan en el código de bytes JVM, como Scala, Clojure y Kotlin, por ejemplo. Sin embargo, Esper EPL no es un lenguaje de programación imperativo (procesal) (Espertech, 2018, p. 1-2).

2.3.9.2. Representación de eventos

Hay varias opciones para representar eventos en Esper, desde un POJO, a una matriz de objetos (Object []) java.util.Map. Un simple POJO con beans como getters y setters es generalmente una buena opción (Espertech, 2018, p. 37).

2.3.9.3. Declaraciones y oyentes

Esper proporciona un motor donde se registran consultas continuas a los flujos de datos. Estas consultas se denominan "statements", y la idea básica es que los "listeners" se pueden

adjuntar a las “statements”, lo que significa que se puede desencadenar una acción personalizada cada vez que se cumple un evento que coincide con las condiciones de la consulta.

2.3.9.4. Modelo de procesamiento de flujo

El modelo de procesamiento de EPL es continuo: los “listeners” reciben datos actualizados tan pronto como el motor procesa eventos para esa declaración, de acuerdo con la elección de las secuencias de eventos. Los “listeners” se implementan como clases en JAVA, por cada secuencia de eventos pre definidos, se debe implementar un “listener”.

a) Insertar un flujo de eventos: La secuencia de inserción denota los nuevos eventos que llegan a una ventana de datos o agregación. La figura 2.26 muestra una serie de eventos de tipo W (1 a 6) que llegan en el tiempo:

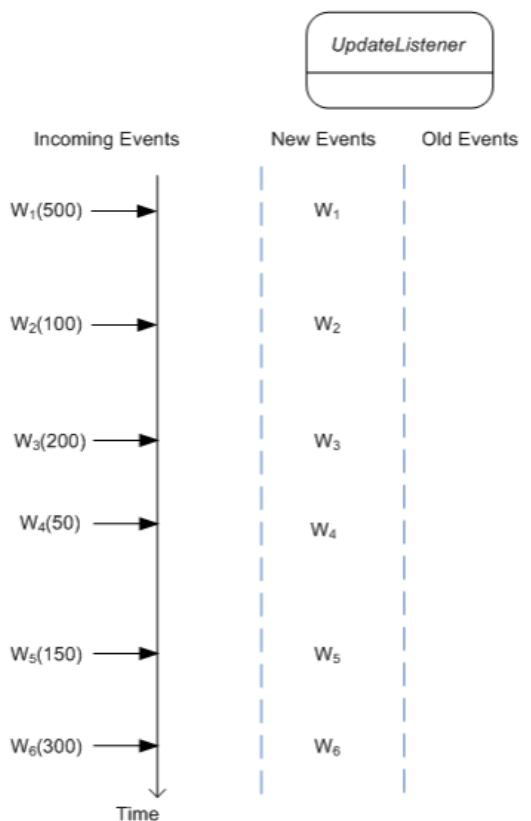


Figura 2.26.- Eventos de tipo W (1 a 6) que llegan en el tiempo. Fuente: (Espertech, 2018)

La consulta:

```
select tempInCelsius from com.app.sensors.TemperatureEvent
```

Dará como resultado un total de 6 eventos nuevos para el “listener”

b) Definiendo un número fijo de eventos para monitorear:

Una ventana de longitud es el camino a seguir cuando queremos decirle al motor que guarde los últimos N eventos para un flujo de eventos, según se muestra en la figura 2.27.

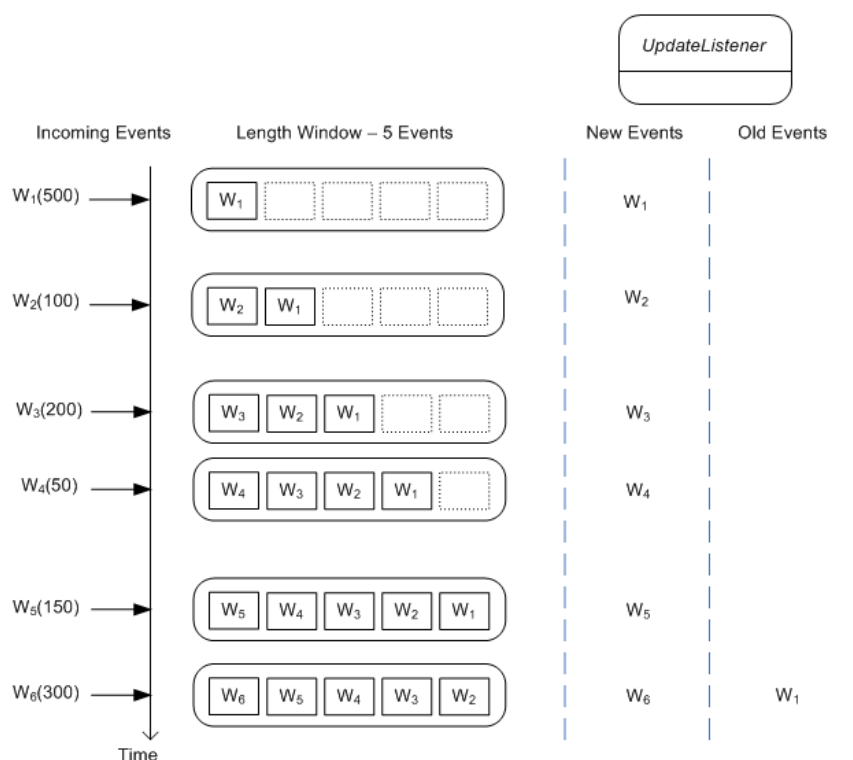


Figura 2.27.- Eventos de tipo W que arriban en una longitud de ventana 5. Fuente: (Espertech, 2018)

La consulta:

```
select tempInCelsius from com.app.sensors.TemperatureEvent#length(5)
```

Mantendrá como máximo cinco eventos (ordenado por FIFO), y esos se pasarán al “listener”.

c) Eliminar un flujo de eventos: Mientras que el término insertar flujo de eventos denota nuevos eventos que llegan, el término eliminar flujo de eventos denota eventos que salen de una ventana de datos o que cambian los valores de agregación. En este ejemplo, la secuencia de eliminación es la secuencia de eventos que dejan la ventana, y dichos eventos se publican en los “listeners” como eventos antiguos”.

d) Filtrado: Se pueden aplicar criterios a las secuencias de eventos, lo que hace que los eventos se filtren antes de que ingresen a la ventana de datos. Por ejemplo, la consulta:

```
select tempInCelsius from  
com.app.sensors.TemperatureEvent(tempInCelsius>0)#length(5)
```

Truncará todas las temperaturas bajo cero y nunca entrarán en la ventana de longitud 5, según se muestra en la figura 2.28.

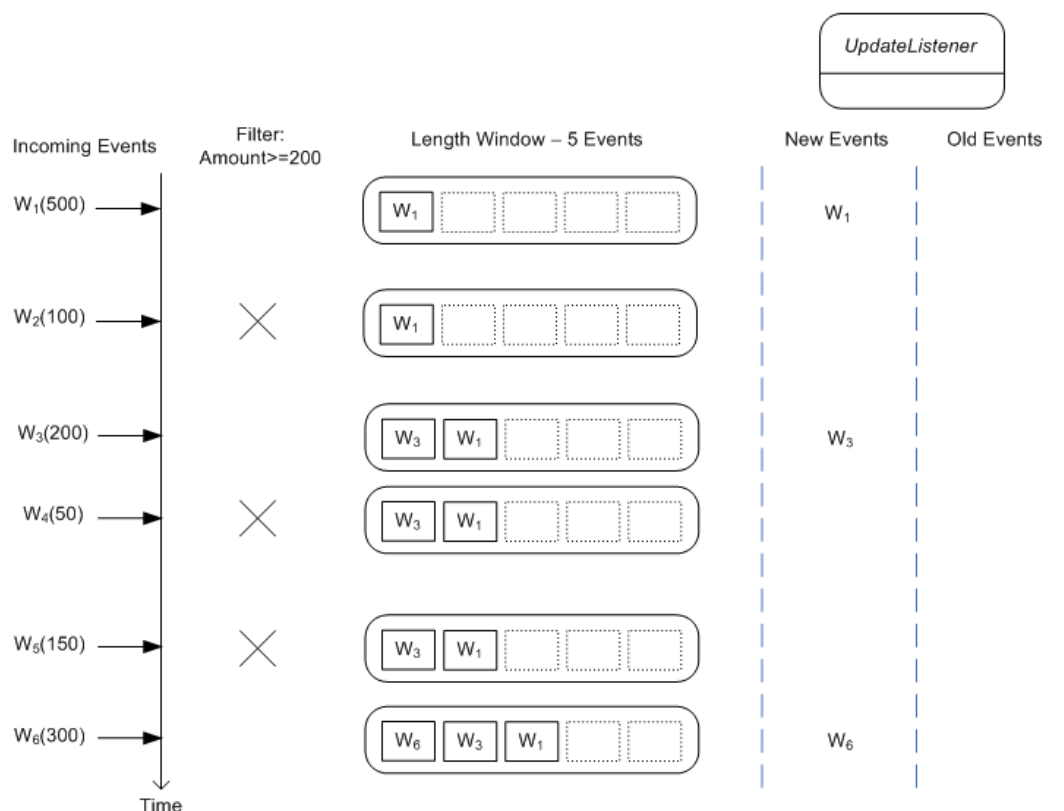


Figura 2.28.- Eventos de tipo W que arriban en una longitud de ventana 5, donde son truncados los eventos con temperatura bajo cero. Fuente: (Espertech, 2018)

- e) **La cláusula Where:** Son criterios que se pueden aplicar a los flujos de eventos, sin embargo, la diferencia con los filtros es que los eventos se mantienen en la ventana de datos, aunque es posible que el evento no se active. Esta consulta:

```
select * com.app.sensors.TemperatureEvent#length(5) where
templnCelsius > 0
```

Significa que los eventos cuya temperatura es inferior a cero no activan “listeners”, sin embargo, los eventos se mantienen en la ventana de datos, según se muestra en la figura 2.29.

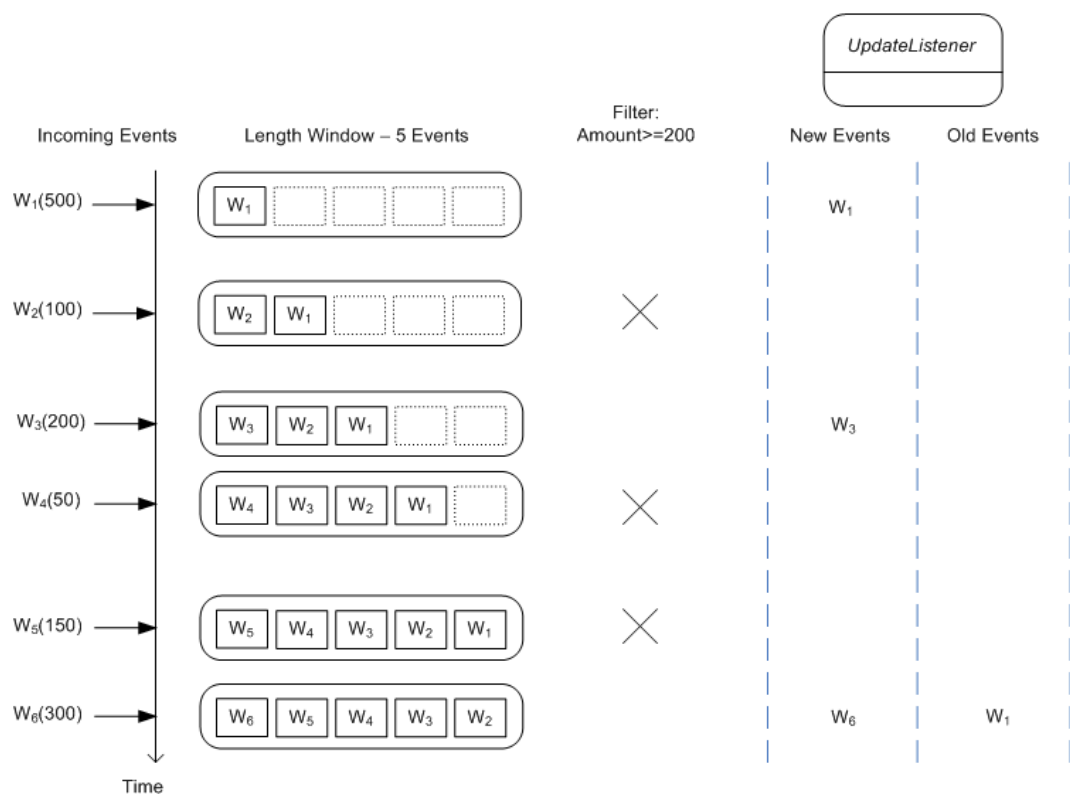


Figura 2.29.- Eventos que no activan “listeners”. Fuente: (Espertech, 2018)

f) Ventanas de tiempo: Una ventana de tiempo es una ventana móvil que se extiende al intervalo de tiempo especificado en el pasado. La consulta:

```
select * com.app.sensors.TemperatureEvent#time(4 sec)
```

Mantendrá los eventos de temperatura recogidos en los últimos cuatro segundos en la ventana de datos, según se muestra en la figura 2.30.

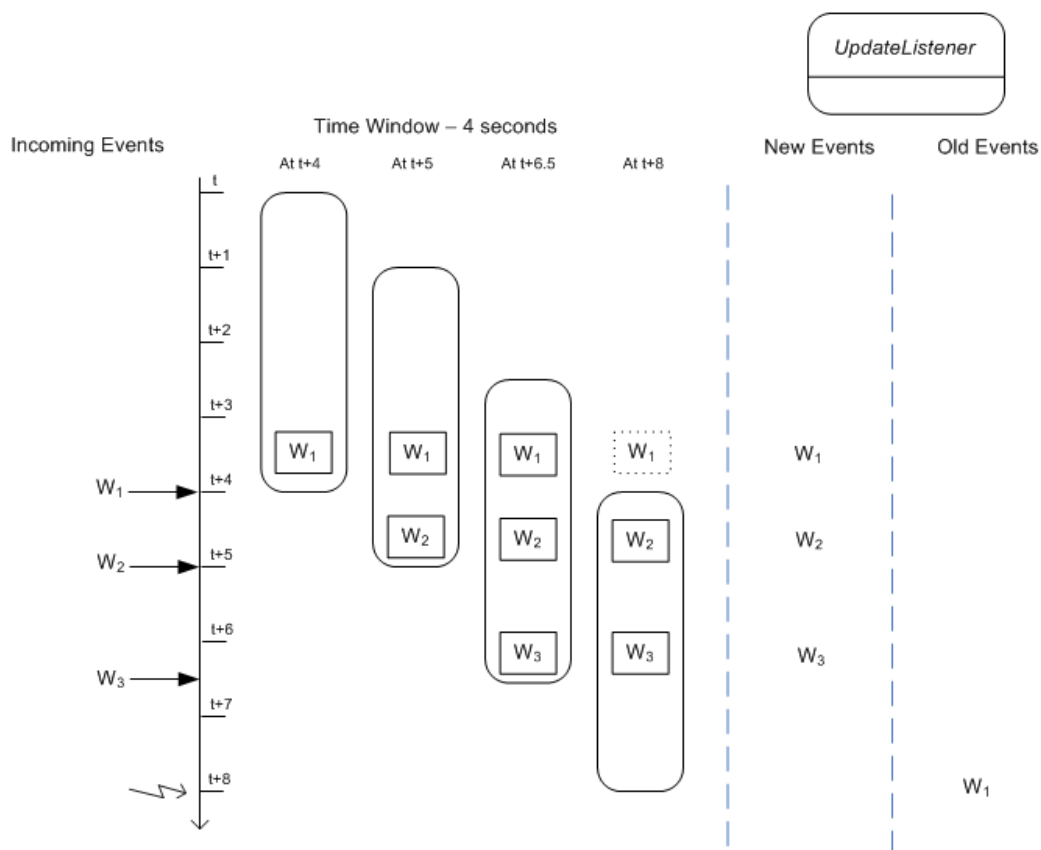


Figura 2.30.- Eventos de temperatura recogidos en los últimos cuatro segundos en la ventana corrediza de datos. Fuente: (Espertech, 2018)

g) Lotes de tiempo: El lote de tiempo almacena eventos y los libera a intervalos cuantificados (por ejemplo, cada cuatro segundos se reinicia el búfer). La consulta:

```
select * from com.app.sensors.TemperatureEvent#time_batch(4 sec)
```

Los eventos se mantendrán en intervalos de cuatro segundos. Cuando llega un evento, su ventana de datos tendrá los eventos que han llegado desde el inicio de ese espacio de cuatro segundos hasta ese momento. Esto se traduce en la figura 2.31:

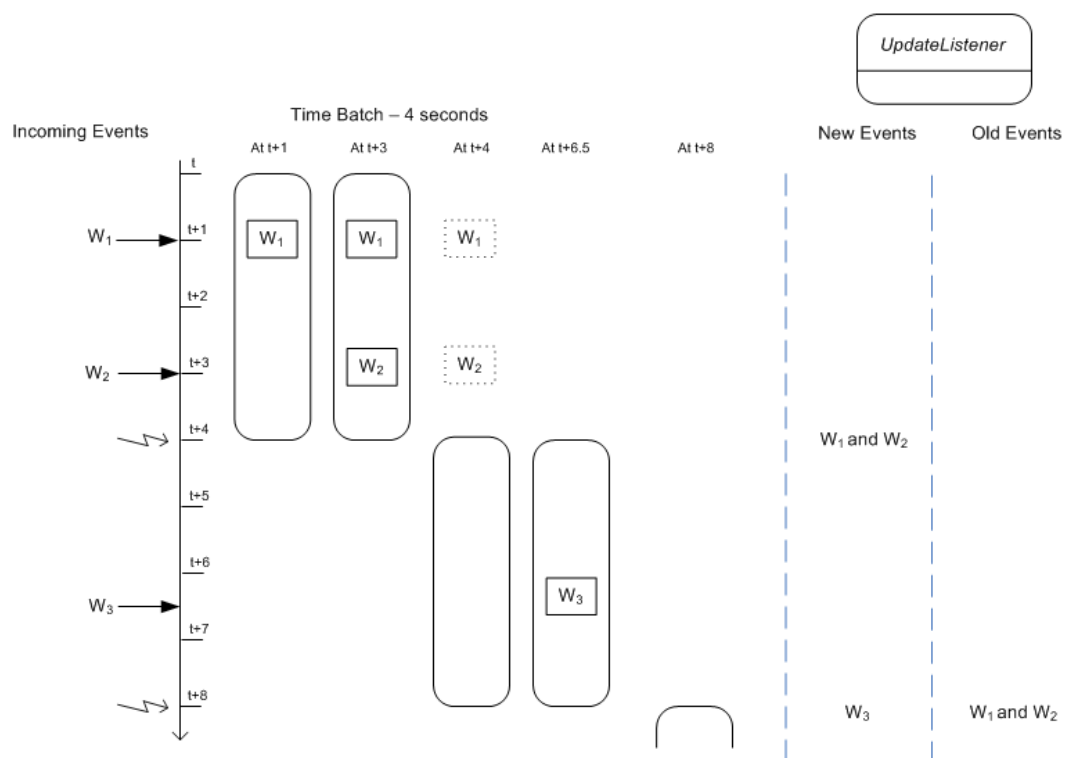


Figura 2.31.- Eventos de temperatura recogidos y mantenidos en los últimos cuatro segundos en la ventana de datos. Fuente: (Espertech, 2018)

2.3.10. EXPECT

En los Network Operation Centers (NOC), cuando sucede un evento importante con un equipo de comunicaciones, el operador accede remotamente al equipo en forma remota mediante alguna herramienta de emulación de terminal remota como telnet o ssh. Durante el proceso de conexión con el equipo, el operador realiza la tarea repetitiva de ejecutar el cliente telnet, digitar la el usuario y contraseña de seguridad de acceso, ejecutar comandos administrativos, etc.

En muchos casos (bajo ciertas condiciones) se detecta un tipo de alarma especial, donde se conoce paso a paso que es lo que se debe hacer. El operador ejecuta siempre la misma secuencia de comandos en el equipo de comunicaciones, los cuales, deben estar especificadas en una tarea que es parte de un proceso mayor. La herramienta Expect permite acceder a los equipos de

comunicaciones en forma remota, ejecutar una serie de acciones previamente determinadas con el objetivo de evitar que el operador ejecute labores repetitivas. De esta forma se facilita el trabajo, al hacer posible la automatización de algunas tareas en forma de scripts interactivos.

Expect, una extensión del lenguaje de script TCL escrito por Don Libes. El programa automatiza las interacciones con programas que exponen una interfaz de terminal de texto. Expect, originalmente escrito en 1990 para la plataforma Unix, desde entonces está disponible para Microsoft Windows y otros sistemas.

(Libes, 1995) Expect es un programa para controlar aplicaciones interactivas. Estas aplicaciones interactivamente preguntan y esperan que un usuario ingrese pulsaciones de teclas en respuesta. Al usar Expect, se puede escribir scripts simples para automatizar estas interacciones (ver figura 2.32). Y usando interactiva automatizada programas, se pueden resolver problemas que nunca antes se hubieran considerado.

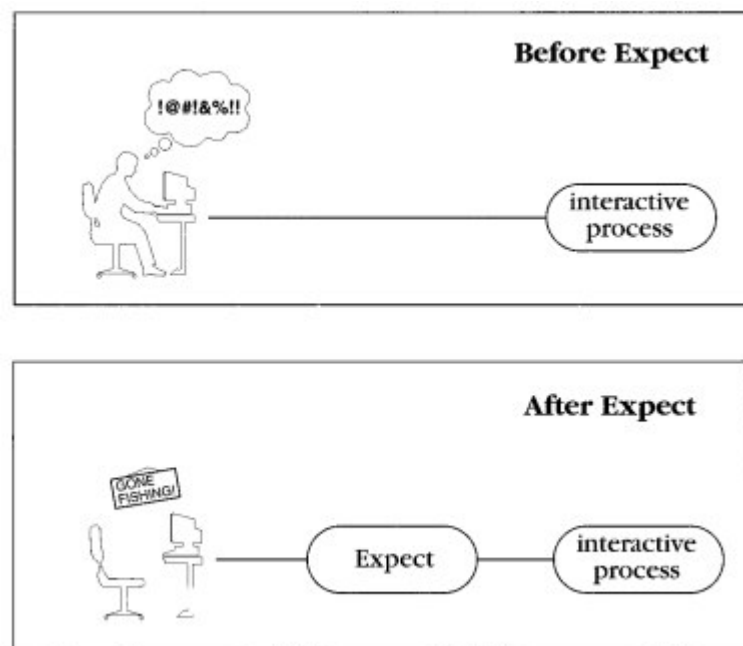


Figura 2.32.- Un ejemplo del uso de Expect. Fuente (Libes, 1995).

2.3.10.1. Instalación de Expect

En sistemas Linux, la instalación de Expect se realiza de la siguiente forma (Howtoinstall, 2019):

```
apt-get install expect
```

2.3.10.2. Creación y ejecución de scripts Expect

Los scripts expect se editan con cualquier editor de textos estándar de Linux como vi o nano. Se suele crear el script con extensión exp u otra extensión que indique que el script está editado para Expect. Dentro del editor de texto primero se escribe una secuencia de comandos que necesitará entradas de los usuarios y luego se escribe la secuencia de comandos utilizando la función de espera para automatizar la tarea. Esta secuencia de comandos interactúa con un programa que se ha ejecutado previamente y está a la espera de recibir órdenes y responder. El siguiente bloque de código muestra un sencillo programa de ejemplo que va a interactuar con un script Expect:

```
#!/bin/bash

echo "Ingrese su nombre"
read $REPLY

echo "Ingrese su edad"
read $REPLY

echo "Ingrese su salario"
read $REPLY
```

A continuación, se muestra el guion para responder preguntas del guion anterior:

```
#!/usr/bin/expect -f

set timeout -1
spawn ./que.sh
expect "Ingrese su nombre\r"
send -- "Yo soy Carlos\r"
expect "Ingrese su edad\r"
send -- "24\r"
expect "Ingrese su salario\r"
send -- "100k\r"
expect eof
```

2.3.10.3. Aplicación de Expect en redes de comunicaciones

Por experiencia, algunas fallas en las redes de comunicaciones son conocidas y estas fallas se pueden mitigar en forma automatizada sin intervención del operador humano. En ese sentido, el uso de “Expect” se puede volver relevante. Creando un script “Expect” adecuado y pasandole a este los parámetros del equipo, se consigue actuar en forma automática sobre dicho equipo para mitigar la falla.

2.3.11. GNS3

GNS3 es un emulador gráfico de red que permite diseñar topologías de red complejas y poner en marcha virtualización sobre dispositivos de red, permitiendo implementar redes que son una combinación de dispositivos reales y virtuales. GNS3 usa una interfaz gráfica de usuario (GUI) fácil de usar que permite a los usuarios configurar un componente de red en una máquina virtual que ejecuta el sistema operativo igual que el componente de red original (Emiliano, 2015). GNS3 permite realizar emulaciones debido a que usa los siguientes componentes:

- Dynamips es un emulador de sistemas operativos de routers Cisco que permite ejecutar imágenes IOS de Cisco Systems.(Welsh, 2013).
- Dynagen es un entorno para crear redes con un lenguaje basado en texto. Es usado por Dynamips.
- VPCS es un emulador de computadoras de escritorio con funciones mínimas de networking
- Qemu y VirtualBox son hipervisores capaces de virtualizar máquinas virtuales como de firewalls o sistemas operativos.

CAPÍTULO III

3. METODOLOGÍA

3.1. Introducción

El presente trabajo de tipo experimental ha sido desarrollado empleando el lenguaje de programación JAVA. El producto es un sistema multiagente, que es la una implementación de la arquitectura propuesta. El trabajo emplea la “Metodología INGENIAS” para el análisis y desarrollo del sistema. El ambiente de pruebas ha sido implementado, creando una red virtual basada en la tecnología IP y compuesta por cinco enrutadores o nodos que emulan una red WAN (nodos Trujillo, Arequipa, Piura, Cuzco y Lima) y un entorno multiagente donde se ejecuta el softbot gestor de notificaciones propuesto. La red virtualizada genera notificaciones en respuesta a fallos u otros eventos que suceden en la red. Las notificaciones son colectadas por el softbot gestor de notificaciones, el cual, selecciona solo un conjunto de notificaciones importantes (se han escogido 20 tipos de notificaciones relevantes entre miles de posibles notificaciones) y el softbot gestor de notificaciones diagnostica, propone al operador la solución del problema encontrado y puede actuar en ciertas circunstancias sobre la red para mitigar un problema encontrado.

Para manejar un gran volumen de notificaciones, se ha desarrollado un aplicativo generador de notificaciones, el cual, genera y envía al softbot gestor de notificaciones muestras con 2000 notificaciones (aproximadamente 196KB de tamaño). Entre las 2000 notificaciones enviadas al softbot gestor de notificaciones, existe solo una notificación importante que el softbot gestor de notificaciones debe saber identificar para que este tome las acciones de diagnóstico o mitigación. La figura 3.1 muestra gráficamente la visión a alto nivel de este trabajo de investigación:

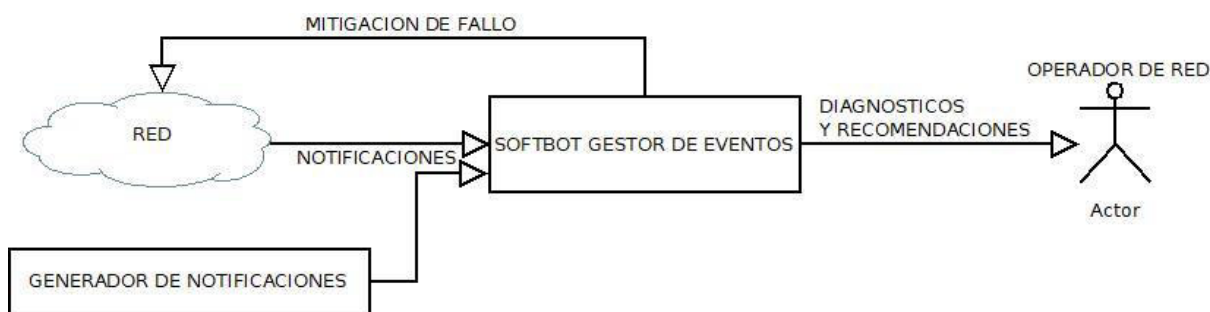


Figura 3.1.- Visión a alto nivel del trabajo experimental propuesto

En el anexo B se incluye el diagrama de procesos de las etapas de análisis y diseño del sistema. Además, al inicio del capítulo IV se define en detalle la metodología de pruebas del sistema a desarrollar.

3.2. Análisis y diseño del sistema

3.2.1. Descripción general del sistema

La descripción general del sistema se divide en dos partes bien definidas:

- Entorno donde el softbot gestor de notificaciones realizará las tareas de gestión
- Plataforma de desarrollo.
- Descripción general del softbot gestor de notificaciones.

3.2.1.1. Entorno general donde el softbot realizará las tareas de gestión

La arquitectura a alto nivel del softbot gestor de notificaciones y el entorno que se rodea se muestra en la figura 3.2:

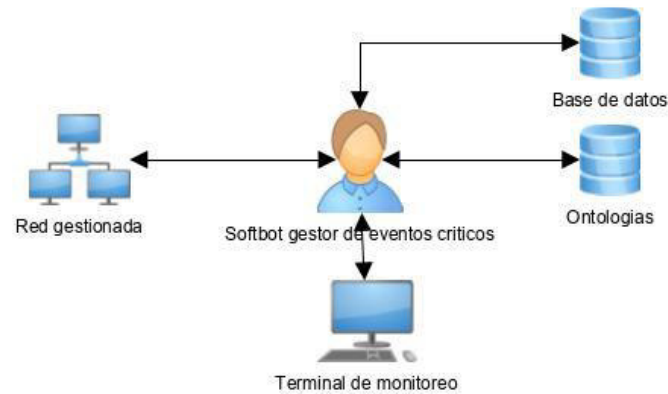


Figura 3.2.- Arquitectura a alto nivel del softbot gestor de notificaciones

Los componentes de la arquitectura de alto nivel son los siguientes:

- **Red gestionada:** Está compuesta por una red de enrutadores y redes de área local interconectadas a los enrutadores mediante enlaces ethernet. Para lograr la interconexión entre las redes, los enrutadores están configurados con el protocolo OSPF. Todos los enrutadores de la red gestionada están configurados para envíen notificaciones al softbot gestor de notificaciones.
- **Base de datos:** Contiene una tabla donde se almacena las recomendaciones que debe hacer el softbot gestor de notificaciones al operador del NOC ante un incidente en la red gestionada. Cuando el softbot requiere dar una respuesta a una incidente consulta a esta tabla antes de proporcionar dicha respuesta.
- **Ontologías:** Almacena la información del lenguaje común que usan los agentes del softbot gestor de notificaciones para que estos puedan dialogar y comprenderse.

- **Terminal de monitoreo:** Muestra por la pantalla del computador la información generada por el softbot gestor de notificaciones, como eventos, recomendaciones y acciones.
- **Softbot gestor de notificaciones:** Recibe notificaciones o hechos provenientes de la red gestionada, tiene una base de datos de conocimientos y un motor de inferencias que le permite razonar ante la llegada de notificaciones y de acuerdo a su lógica, aconseja al operador del NOC que acciones tomar ante un evento importante o toma acción en forma desatendida sobre la red, mitigando alguna falla crítica que él ha detectado.

3.2.1.2. Plataforma de desarrollo

En entorno de desarrollo está implementado bajo un computador AMD Ryzen, en el cual se ha instalado el sistema operativo Linux Ubuntu 19.04 y la plataforma de desarrollo de software Eclipse IDE 4.12.0 para desarrolladores JAVA. La red WAN de prueba ha sido implementada sobre el software de emulación de redes GNS3. Las características y configuraciones de cada uno de los componentes del entorno de desarrollo son los siguientes:

- **Configuración del computador:** La tabla 3.1 muestra las características del computador usado en la etapa de desarrollo del sistema.

Tabla 3.1.- Características del computador para el desarrollo del softbot

Característica	Descripción
Procesador	AMD Ryzen 7 1700. 3Ghz. 8 núcleos
Memoria RAM	16 GB. Frecuencia 2600Mhz.
Disco duro	1 TB SATA. 5400 RPM

Placa madre	Asus Prime A320M-K
Teclado	Micronics en español
Monitor	LG 19"

- **Configuración del sistema operativo:** La tabla 3.2 muestra las características del sistema operativo usado en la etapa de desarrollo del sistema.

Tabla 3.2.- Características del sistema operativo para desarrollo del softbot

Característica	Descripción
Sistema operativo	Linux Ubuntu 19.04
Kernel	5.0.0
Entorno de escritorio	Unity
Dirección IP (softbot gestor de notificaciones)	192.168.1.61

- **Configuración del entorno de emulación GNS3:** La configuración del entorno de emulación está instalada sobre Linux Ubuntu 19.04 y está conformada por un grupo de enrutadores, según se muestra en la figura 3.3.

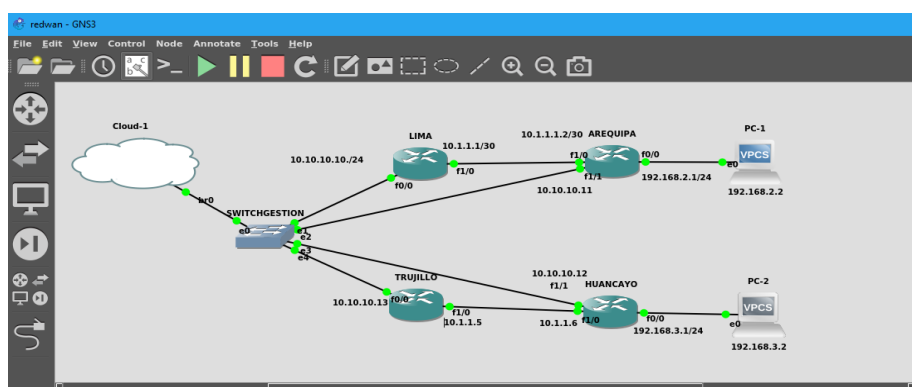


Figura 3.3: Entorno de emulación de la red WAN

Según el gráfico anterior, se dispone de una red de área amplia (WAN) virtualizada con GNS3 conformada por 4 enrutadores de la marca Cisco 7200. Los enrutadores interconectan redes de clientes en las ciudades de Lima, Trujillo, Huancayo y Arequipa. Tenemos que tomar en consideración que debido a que se están empleando sistemas operativos de enrutadores reales, el consumo de memoria de la red emulada con GNS3 es alto. Es por ello que, el entorno de pruebas para la interconexión de la red WAN se ha limitado a cuatro enrutadores.

Los enrutadores se interconectan a través de enlaces de cobre e interfaces “fast ethernet”. Cada enrutador dispone de una interfaz que pertenece a la red de gestión (10.10.10.0/24). Las interfaces de la red de gestión de los enrutadores son las siguientes (ver tabla 3.3):

Tabla 3.3.- Datos de las interfaces de gestión de los enrutadores y redes

Nombre del enrutador o switch	Interfaz de gestión	Dirección IP de la interfaz de gestión	Red de área local conectada
Lima	FasEthernet0/0	10.10.10.10/24	-
Arequipa	FastEthernet1/1	10.10.10.11/24	192.168.2.0/24
Huancayo	FastEthernet1/1	10.10.10.12/24	192.168.3.0/24
Trujillo	FasEthernet0/0	10.10.10.13/24	-
Switch de gestión	Ethernet 0	10.10.10.9/24	10.10.10.0/24

La red puede ser extensible a un tamaño mucho mayor, si se dispone de una gran cantidad de memoria para ello. Los enrutadores empleados en la red WAN implementada son de la

marca Cisco y serie 7200. La tabla 3.4 muestra información genérica de los enrutadores de la red WAN implementada.

Tabla 3.4.- Información genérica de los enrutadores de la red gestionada

Nombre del equipo	Marca del enrutador	Modelo del enrutador	Sistema operativo del enrutador	Dirección IP de la interfaz de gestión del enrutador
LIMA	CISCO	7206VXR	IOS15.2(4)M7	10.10.10.10
AREQUIPA	CISCO	7206VXR	IOS15.2(4)M7	10.10.10.11
HUANCAYO	CISCO	7206VXR	IOS15.2(4)M7	10.10.10.12
TRUJILLO	CISCO	7260VXR	IOS15.2(4)M7	10.10.10.13

A través de su interfaz de gestión cada equipo envía sus logs de eventos generados al switch de gestión (SWITCHGESTION) que a su vez se conecta con el Centro de Gestión de Redes o Network Operation Center (NOC). El en NOC se encuentra instalado del softbot gestor de notificaciones, el cual, recibe todos los eventos generados por los enrutadores a través de enrutamiento dinámico. La red gestionada emplea el protocolo de enrutamiento dinámico OSPF para enrutar los paquetes provenientes de las redes de los clientes y también se usa el mismo protocolo para llevar los mensajes SYSLOG hacia la red de gestión donde está el softbot gestor de notificaciones. Todos los enrutadores tienen activados el protocolo SYSLOG o “logging” para la generación y envío de eventos log hacia el softbot gestor de notificaciones.

Existen fallas de hardware, memoria, temperatura, refrigeración del chasis del equipo, etc, los cuales son difíciles de generar en el entorno GNS3. Para propósitos prácticos y demostrativos algunos de estos eventos se generan mediante un programa de desarrollo propio llamado “generador de logs”, el cual, es capaz de enviar miles de notificaciones entre los cuales se pueden insertar algunos eventos importantes que la lógica del softbot gestor de notificaciones debe identificar.

3.2.1.3. Descripción general del softbot gestor de notificaciones

La ocurrencia de uno o más fallos en protocolos de comunicaciones o enlaces de la red WAN o hardware de los equipos ocasionará que los enrutadores envíen notificaciones al softbot gestor de notificaciones. Dependiendo de la naturaleza del fallo, el softbot gestor de notificaciones tomará un rol pasivo (informar) o activo (mitigar el impacto del fallo en la red). En forma general el softbot gestor de notificaciones se describe a través de las siguientes funciones:

- Recolección de eventos a través de notificaciones provenientes de la red WAN gestionada.
- Filtrado de eventos. Solo se analizarán los eventos relevantes, dejando de lado a aquellos eventos que no son importantes.
- Clasificación de los eventos de acuerdo a su naturaleza. Clasificación en base a fallo por protocolo, fallo de enlace físico, etc.
- Detección de relaciones de causalidad entre eventos (causa y efecto). Por ejemplo, un fallo a nivel de capa física ocasiona un fallo a nivel de capa de red.

- Diagnóstico del problema y recomendaciones de solución o mitigación a través de mecanismos de razonamiento automático.
- Ejecución de acciones sobre la red WAN gestionada, con el fin de mitigar aquel evento que tenga impacto negativo.
- Informar a los operadores del NOC acerca del diagnóstico, la solución y/o las acciones de mitigación ante un fallo que tenga impacto negativo en la red WAN gestionada.

3.2.2. Análisis del softbot gestor de notificaciones

3.2.2.1. Procesos de gestión

Los trabajos que desarrollan los operadores del NOC y el softbot gestor de notificaciones para la gestión de fallos de la red, se pueden organizar mediante procesos y actividades, los mismos que se ven explicados en la tabla 3.5:

Tabla 3.5.- Procesos de gestión y actividades del softbot

Agente	Proceso de gestión	Actividad
Colector	Recolección de eventos de la red	Recolección de eventos a través de notificaciones provenientes de la red WAN gestionada.
Colector	Recolección de eventos de la red	Filtrado de eventos, solo se analizan los eventos relevantes, dejando de lado a aquellos eventos que no son importantes.
Colector	Clasificación de eventos de de red	Clasificación de los eventos de acuerdo a su naturaleza. Clasificación en base a fallo por protocolo, fallo de enlace físico, hardware, etc.
Experto	Motor de inferencia	Aplicación de reglas lógicas a la base

		de conocimientos para deducir nueva información
Experto	Motor de inferencia	Detección de relaciones de causalidad entre eventos (causa y efecto). Por ejemplo, un fallo a nivel de capa física ocasiona un fallo a nivel de capa de enlace que a su vez genera un fallo a nivel de capa de red.
Ejecución	Mitigar eventos de impacto negativo la red	Ejecución de acciones sobre la red WAN gestionada, con el fin de mitigar aquel evento que tenga impacto negativo en la red.
Interface	Mostrar información a los operadores del NOC.	Informar a los operadores del NOC acerca del diagnóstico, la solución y/o las acciones de mitigación ante un fallo que tenga impacto negativo en la red WAN gestionada, así como información estadística de la red.

3.2.2.2. Especificación de requerimientos

Con la información obtenida del ítem 3.2.2.1 y aplicando como herramienta, el análisis en base a experiencia en gestión de red, obtenemos como resultado los requisitos de la estrategia de solución propuesta, es decir con la información obtenida de la tabla 3.5 definimos las siguientes “gestiones” del sistema multiagente:

- a) Gestión de recolección de eventos
- b) Gestión del experto
- c) Gestión de ejecución de solución
- d) Gestión de interface hombre-softbot

A continuación, se describe cada una de las gestiones del SMA:

- a) Gestión de colección de eventos:** Los eventos que ocurren en la red WAN son registrados por los equipos enrutadores (logging) y estos, a la vez son enviados a la gestión de recolección de eventos empleando el protocolo SYSLOG. Una vez leídos los eventos, la gestión de recolección de eventos acepta aquellos eventos que son considerados relevantes. Además, se realiza la clasificación de los eventos de acuerdo a su naturaleza, es decir fallo por protocolo, fallo de enlace físico, hardware, etc, según se muestra en la tabla 3.6:

Tabla 3.6.- Requisitos de la gestión de recolección de eventos

Requisito simplificado	Requisito detallado
Req. 1.- Lee eventos	Req. 1.- Abre un puerto de comunicaciones (SYSLOG “collector”) que escucha los datagramas (SYSLOG “originator”) que llegan de la red.
Req. 2.- Clasifica eventos	Req. 2.- De todos los mensajes SYSLOG recibidos, solo se seleccionan y clasifican los eventos de acuerdo a su importancia, es decir se permiten aquellas notificaciones que tengan como facility a LINK, LINEPROTO, OSPF y con severity o severidad mayor o igual a 3.
Req. 3.- Transformación de eventos	Req. 3.- Se realiza la transformación de los eventos, obteniendo eventos derivados LINK, LINEPROTO y OSPF.

Req. 4.- Almacenado de eventos	Req. 4.- Almacenar los datos de los eventos filtrados en un archivo en disco duro.
---------------------------------------	---

b) Gestión del experto

Los eventos filtrados son enviados a un motor de Complex Event Processing (CEP) que realiza el procesamiento automático de los eventos mientras estos ocurren, encontrándose patrones o correlaciones con otros eventos ocurridos en el pasado según se define en la tabla 3.7.

Tabla 3.7.- Requisitos de la gestión de experto o CEP

Requisito simplificado	Requisito detallado
Req. 5.- Razonamiento automático	Req. 5.- Ejecuta el CEP. El evento LINK, LINEPROTO, OSPF, etc son detectados por el CEP el cual realiza el razonamiento automático según se detalla en la tabla 3.5
Req. 6.- Detección de intermitencias en la red	Req. 6.- El fenómeno intermitencia denominado “flapping” en redes OSPF resulta muy perjudicial para el desempeño de la red. El “flapping” ocurre cuando por problemas físicos, una interfaz de red se desactiva y activa constantemente. Esto ocasiona el recálculo constante de las bases de datos de estado de enlace de los enrutadores y problemas de enrutamiento y performance serios. La ocurrencia de tres (3) eventos seguidos en un minuto de cualquiera del tipo de mensaje: • LINEPROTO-5-UPDOWN

	Ocasionará eventos OSPF-5-ADJCHG. Este fenómeno “fappling” también ocasiona saturación de procesamiento en todos los equipos de red o area OSPF. Para mitigar esta condición, el softbot gestor de notificaciones informa al operador del NOC acerca de este evento.
--	---

La especificación detallada de la forma como debe ser el razonamiento automático se muestra en la tabla 3.8.

Tabla 3.8.- Especificación detallada del razonamiento automático en base a los eventos previamente clasificados

Evento N°	Evento previo	Facility	Severity	Mnemo	Explicación	Recomendación/Solución/Acción
1	NO	LINK	3	LINK_F AULT	El equipo genera una alarma de falla de enlace cuando los problemas con la capa física de un puerto causan una transmisión de datos no confiable. Una condición típica de falla de enlace es la pérdida de señal o reloj. La alarma de falla de enlace se borra automáticamente cuando se borra la condición de falla de enlace.	En general, si las configuraciones de velocidad y dúplex están configuradas en valores fijos y los errores de CRC siguen ocurriendo, es posible que cambie el cable, probar otro puerto en el equipo o cambiar la NIC. Los errores de CRC suelen estar relacionados con el hardware.
2	NO	LINK	3	FCS_E RROR	Este problema se debe a un cable defectuoso, a una tarjeta de interfaz de red (NIC) con una configuración dúplex equivocada.	1. Si es posible, establezca la configuración de velocidad y dúplex tanto en el puerto como en el dispositivo conectado. 2. Verifique el cable con un probador o reemplácelo con un cable en buen estado. 3. Si la conexión es a un panel de conexiones, omita el panel de conexiones y conéctese

						directamente al dispositivo final. 4. Verifique los controladores NIC en el dispositivo final para asegurarse de que admitan la configuración de dúplex y velocidad requerida. 5. Verifique el puerto conectando el dispositivo final a un puerto diferente en el switch. Monitoree los contadores de puertos para ver si se incrementa el recuento de errores de FCS.
3	NO	LINK	3	TOOS MALL	Se detectó un paquete de salida que era más pequeño que el tamaño mínimo de datagrama permitido. Un error en otro controlador de un equipo vecino o un error en el software del sistema probablemente activó este mensaje de error.	Este es solo un mensaje informativo. No se requiere ninguna acción.
4	NO	LINK	5	CHANGED	Error de capa 1. La interfaz está caída por cualquiera de las siguientes causas: 1.Cable no conectado. 2. Cable incorrecto 3.Comando shutdown aplicado. 4.Desajuste de velocidad (Ethernet) 4.El dispositivo en el otro extremo del cable está apagado.	1. Verificar si el cable está desenchufado y enchufarlo de ser necesario. 2. Verificar si el cable es correcto, si el cable es incorrecto, cambiarlo. 3. Aplicar el comando no shutdown. 4. Asegurarse que está bien configurada la velocidad (Ethernet). 5. Asegurarse que el dispositivo en el otro extremo del cable está encendido.
5	NO	LINE PROTO	5	UPDOWN	El software que maneja el enlace afirma que la interfaz no se puede utilizar. El protocolo de línea de nivel de enlace de datos cambió de estado.	1.Aplicar el comando show interfaces. Si aparecen errores de CRC se debe: 2.Verificar y corregir de ser necesario la configuración duplex en los equipos.. 3.Verificar y cambiar el cable UTP de ser necesario..

						4. Si el cable UTP es categoría 5, cambiarlo por un cable de categoría 6. 5. Verificar y asegurarse que la velocidad de los puertos son compatibles.
6	Evento N° 4	LINEP ROTO	5	UPDO WN	La causa es la del evento N° 4	La solución es la del evento N° 4
7	Evento N° 2	LINEP ROTO	5	UPDO WN	La causa es la del evento N° 2	La solución es la del evento N° 2
8	Evento N° 1	LINEP ROTO	5	UPDO WN	La causa es la del evento N° 1	La solución es la del evento N° 1
9	NO	OSPF	5	ADJCH G	La adyacencia no se producirá debido a las siguientes razones. 1. Discrepancia en el área ID. 2. Discrepancia en la configuración OSPF con el comando network. 3. Desajuste en la configuración de la autenticación 4. Desajuste de los intervalos dead.	En la configuración OSPF verificar y corregir de ser necesario: 1. El área ID. 2. Los comandos network. 3. La autenticación (si la hay) 4. El intervalo dead
10	Evento N° 5	OSPF	5	ADJCH G	Fallo en OSPF debido al software que maneja el enlace afirma que la interfaz no se puede utilizar. El protocolo de línea de nivel de enlace de datos cambió de estado.	1. Aplicar el comando show interfaces. Si aparecen errores de CRC se debe: 2. Verificar y corregir de ser necesario la configuración dúplex en los equipos.. 3. Verificar y cambiar el cable UTP de ser necesario.. 4. Si el cable UTP es categoría 5, cambiarlo por un cable de categoría 6. 5. Verificar y asegurarse que la velocidad de los puertos son compatibles.
11	NO	OSPF	4	ERRR CV	El mensaje de error OSPF-4-ERRRCV indica que un enrutador ha recibido un paquete OSPF no válido. Estas son las posibles causas:	1. Si el área ID no coincide. El enrutador que genera este mensaje recibió un paquete OSPF no válido de la interfaz de red vecina. El paquete no es válido porque su ID de área

					1.ID de área no coincidente 2.Mala suma de comprobación 3.OSPF no habilitado en la interfaz receptora 4.OSPF no habilitado en la interfaz receptora debido al error bug CSCdr48014 5.Versión de OSPF incorrecta 6.Tipo inválido de mensaje OSPF 7.Recuento de anuncios de actualización de estado de enlace incorrecto 8.Longitud de actualización de estado de enlace incorrecta	es diferente. Para evitar estos mensajes, asegúrese de que ambas partes tengan el mismo ID de área al verificar la declaración de red en OSPF, en la configuración del enrutador. 2. Mala suma de comprobación. El enrutador que genera este mensaje ha recibido un paquete OSPF no válido del vecino. El paquete no es válido porque la suma de comprobación OSPF es incorrecta. La causa de la mala suma de comprobación es difícil de definir. Algunas causas posibles del problema son: -Un dispositivo entre los vecinos, como un conmutador, está corrompiendo el paquete. -El paquete recibido por el enrutador no es válido. En este caso, la interfaz del enrutador que envió el paquete es incorrecta o el error se debe a un error de software. -El enrutador receptor está calculando la suma de comprobación incorrecta. En este caso, la interfaz del enrutador de recepción es incorrecta o el error se debe a un error de software. Esta es la causa menos probable de este mensaje de error. Solución. Este problema puede ser difícil de solucionar, pero puede comenzar con esta solución. Cisco ha descubierto que esta solución es efectiva en el 90 por ciento de los casos. Es importante completar estos pasos en orden:
--	--	--	--	--	---	---

					-Cambia el cable entre los enrutadores. En el ejemplo anterior, este sería el cable entre el enrutador que envía el paquete incorrecto y el enrutador que alerta de estos paquetes incorrectos. Si el paso anterior no soluciona el problema, use un puerto diferente en el conmutador entre los enrutadores. Si el paso anterior no soluciona el problema, conecte los enrutadores directamente mediante un cable cruzado (siempre que la ubicación física lo permita). Si no recibe más mensajes, lo más probable es que el conmutador corrompa los paquetes. Si nada de lo anterior resuelve el problema, comuníquese con el Soporte Técnico de Cisco y trabaje con un ingeniero para buscar un error en el Software Cisco IOS®, o para una posible Autorización de Devolución de Material (RMA) para el reemplazo parcial o total de las piezas. 3.OSPF no habilitado en la interfaz de recepción. El enrutador que genera este mensaje recibió un paquete en su interfaz, pero OSPF no está habilitado en la interfaz. Este mensaje solo se genera una vez para una interfaz que no sea OSPF. Solución. Este problema rara vez se ve en un enrutador. Para resolver este problema, asegúrese de que OSPF esté habilitado en la interfaz. Intente volver a ingresar la declaración de red en la
--	--	--	--	--	--

						configuración del enrutador. Para verificar si OSPF está habilitado en la interfaz mencionada anteriormente, escriba el siguiente comando: show ip ospf interface Si OSPF no está habilitado, la salida del comando estará vacía o dirá que OSPF no está habilitado en la interfaz. 4.OSPF no está habilitado en la interfaz de recepción debido a un error CSCdr48014. Puede haber una situación en la que las actualizaciones de OSPF se puedan corromper en el Enrutador Cisco 7500 configurado con OSPF, MPLS y CEF. Las rutas IP se eliminan temporalmente de la tabla de enrutamiento IP y puede ocurrir una pérdida de conectividad. Solución. Actualice el Cisco IOS a la última versión de IOS.
12	NO	SFF8472	5	THRESHOLD_VIOLATION	1.El transceiver SFP está transmitiendo fuera de especificaciones. 2. Problemas en la fibra óptica.	1.Reemplazar el transceiver SFP. 2.La fibra no está insertada o correctamente en uno de los bordes del puerto SFP o el borde está un poco sucio, por lo que necesita limpiarlo.
13	NO	OSPF	4	CONFLICTING_LSAID	La creación LSA es prevenida por otro LSA existente con el mismo LSID pero una máscara distinta.	La solución para el este escenario es redistribución cuidadosa de las rutas. Al redistribuir las rutas (/32 específicamente), asegúrese de que no está en conflicto con LSA existente un de los ID.

14	NO	CDP	4	NATIVE_VLAN_MISMATCH	Ambos lados del enlace trunk están utilizando una vlan nativa diferente.	Asegurarse que la VLAN nativa del enlace trunk sea la misma para ambos puertos.
15	NO	PLATFORM-ASM	0	FAULT_RECOVERY_ERR	Cuando se produce un error de hardware y el controlador del dispositivo no puede corregirlo, se requiere que el Platform Manager reinicie los ASIC. Este error indica que Platform Manager no pudo restablecer los ASIC y que el hardware en el PRP puede no estar en un estado operativo. Esta condición afecta a la funcionalidad de todo el enrutador.	Si hay una tarjeta de procesador redundante: 1.Forzar una conmutación por error con el siguiente comando en el modo de administración: 'redundancy switchover' 2.Vuelva a cargar la tarjeta del procesador que falló con el siguiente comando en el administrador modo: 'reload location RP/0/x/CPU0' donde x es la ranura del procesador de ruta que generó el error. Si no tarjeta redundante, deberá reiniciar el enrutador. emitiendo el comando reload.
16	NO	PLATFORM-ASM	3	MALLOC_FAILED	Error de asignación de memoria fatal	Reducir la actividad del sistema para aliviar las demandas de memoria. Si las condiciones lo ameritan, actualice a una configuración de memoria más grande.
17	NO	PLATFORM-ENVMON	0	CARD_OVERTEMP_SHUTDOWN	La temperatura de la tarjeta ha aumentado por encima del umbral de apagado. Inicio de apagado automático de la tarjeta.	Verifique el fallo usando el CLI: (admin) # show env Si el estado del ventilador está en reset use CLI para volver a poner la tarjeta en servicio, o desenchufe y vuelva a enchufar la tarjeta manualmente.
18	NO	PLATFORM-ENVMON	2	CHASSIS_OVERTEMP	La temperatura ambiente del chasis ha aumentado en por encima del umbral de temperatura de apagado. El apagado del chasis comenzando.	Verifique el fallo usando el CLI: (admin) # show env Verifique que las velocidades del ventilador hayan aumentado para ayudar a bajar la temperatura, la

						temperatura ambiente debería disminuir con el aumento de las RPM del ventilador. El chasis puede continuar funcionando en esta condición, sin embargo, las tarjetas se apagarán cuando la temperatura de su punto de acceso alcance el umbral de apagado.
19	NO	PLATF ORM- ENVM ON	2	FAN	El arreglo de ventiladores ha alcanzado un nivel crítico o de advertencia y se acerca a una condición que está fuera del rango aceptable.	Use 'sh env fan' y 'sh env trace'. Verifique la falla y llame a su representante de soporte técnico para su reparación o reemplazo.
20	NO	PLATF ORM- ENVM ON	2	FAN_F AIL	Fallo en arreglo de ventiladores.	Verifique la falla, use el CLI: (admin) # show env Verifique la salida del ventilador Si la falla del ventilador no se soluciona después de 30 minutos,, llame a su representante de soporte técnico para la reparación o reemplazo de la bandeja de ventiladores.
21	Evento N° 4	OSPF	5	ADJCH G	Fallo en OSPF debido a error de capa 1. La interfaz está caída por cualquiera de las siguientes causas: 1.Cable no conectado. 2. Cable incorrecto 3.Comando shutdown aplicado. 4.Desajuste de velocidad (Ethernet) 4.El dispositivo en el otro extremo del cable está apagado.	1. Verificar si el cable está desenchufado y enchufarlo de ser necesario. 2. Verificar si el cable es correcto, si el cable es incorrecto, cambiarlo. 3. Aplicar el comando no shutdown. 4. Asegurarse que está bien configurada la velocidad (Ethernet). 5. Asegurarse que el dispositivo en el otro extremo del cable está encendido.
22	Evento N° 1	OSPF	5	ADJCH G	Fallo en OSPF debido a que el equipo genera una alarma de falla de enlace	En general, si las configuraciones de velocidad y dúplex están configuradas

					cuando los problemas con la capa física de un puerto causan una transmisión de datos no confiable. Una condición típica de falla de enlace es la pérdida de señal o reloj. La alarma de falla de enlace se borra automáticamente cuando se borra la condición de falla de enlace.	en valores fijos y los errores de CRC siguen ocurriendo, es posible que cambie el cable, probar otro puerto en el equipo o cambiar la NIC. Los errores de CRC suelen estar relacionados con el hardware.
23	Evento N° 2	OSPF	5	ADJCH G	Fallo en OSPF debido a un cable defectuoso, a una tarjeta de interfaz de red (NIC) con una configuración dúplex equivocada.	1. Si es posible, establezca la configuración de velocidad y dúplex tanto en el puerto como en el dispositivo conectado. 2. Verifique el cable con un probador o reemplácelo con un cable en buen estado. 3. Si la conexión es a un panel de conexiones, omita el panel de conexiones y conéctese directamente al dispositivo final. 4. Verifique los controladores NIC en el dispositivo final para asegurarse de que admitan la configuración de dúplex y velocidad requerida. 5. Verifique el puerto conectando el dispositivo final a un puerto diferente en el switch. Monitoree los contadores de puertos para ver si se incrementa el recuento de errores de FCS.

c) Gestión de ejecución de solución

Representa la forma como el softbot gestor de notificaciones modifica el entorno con que interactúa. El softbot gestor de notificaciones ejecuta acciones sobre la red WAN gestionada, con el fin de mitigar aquel evento que tenga impacto negativo en la red. La tabla 3.9 muestra los requisitos de la gestión de ejecución de solución:

Tabla 3.9.- Requisitos de la gestión de ejecución de solución

Requisito simplificado	Requisito detallado
Req. 7.- Ejecuta solución del problema	Req. 7.- El fenómeno denominado “flapping” en redes OSPF resulta muy perjudicial para el desempeño de la red. El “flapping” ocurre cuando por problemas físicos, una interfaz de red se desactiva y activa constantemente. Esto ocasiona el recálculo constante de las bases de datos de estado de enlace de los enrutadores y problemas de enrutamiento y performance serios. La ocurrencia de tres (3) eventos seguidos en un minuto de cualquiera del tipo de mensaje: • LINEPROTO-5-UPDOWN Ocasionará eventos OSPF-5-ADJCHG. Este fenómeno “fappling” también ocasiona saturación de procesamiento en todos los equipos de red o área OSPF. Para evitar esta condición, el softbot gestor de notificaciones no sólo avisa al operador del NOC acerca de este evento, sino también tomará la decisión en forma autónoma de apagar la interfaz que está ocasionando el “flapping”. Para apagar la interfaz, el softbot gestor de notificaciones envía información del enrutador e interfaz a un script “expect”, el cual, en forma automatizada ingresa a la interfaz afectada y la apaga en forma administrativa

d) Gestión de interface hombre-softbot

El softbot informa mediante ventanas gráficas y alertas a los operadores del NOC acerca del diagnóstico, la solución y/o las acciones de mitigación ante un fallo que tenga impacto negativo en la red WAN gestionada. La tabla 3.10 muestra los requisitos de la gestión de la interface hombre-softbot.

Tabla 3.10.- Requisitos de la gestión de ejecución de solución

Requisito simplificado	Requisito detallado
Req. 11.- Muestra eventos SYSLOG	Req. 11.- Mediante interfaz gráfica, el softbot gestor de notificaciones muestra los eventos SYSLOG al operador del NOC.
Req. 12.- Muestra recomendaciones	Req. 12.- Mediante interfaz gráfica, el softbot gestor de notificaciones muestra el resultado de las recomendaciones y las acciones a tomar al operador del NOC.
Req. 13.- Muestra la acción y la solución	Req. 13.- Mediante interfaz gráfica, el softbot gestor de notificaciones muestra al operador del NOC las acciones tomadas para mitigar el “flapping” OSPF (apagado de interface en “flapping”)

3.2.2.3. Casos de uso

En esta etapa se analizan los casos de uso que se consideran importantes para ir bosquejando la arquitectura de la red. En esta primera etapa se tiene una idea de los agentes involucrados y las tareas a realizar para alcanzar los objetivos tal como se muestra en la figura 3.4.

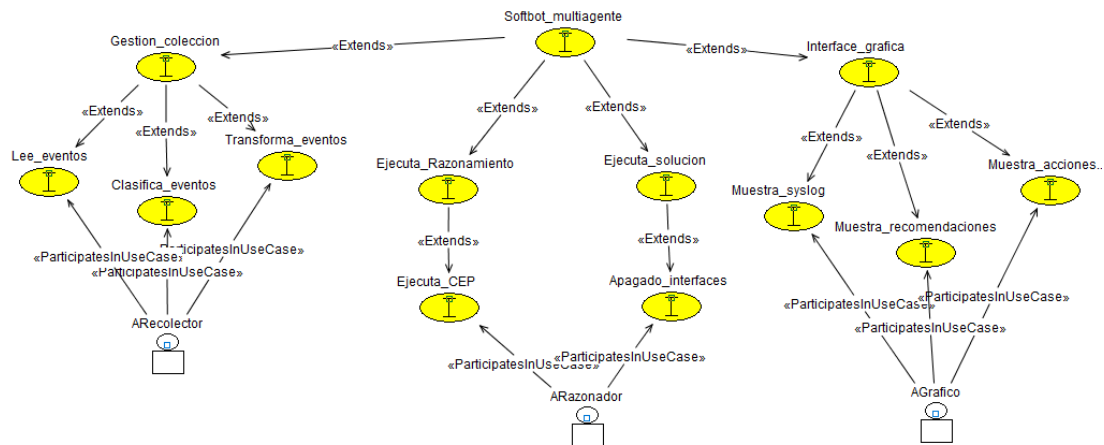


Figura 3.4.- Casos de uso del “softbot gestor de notificaciones”

3.2.2.4. Meta-modelo de Organización

La estructura organizacional del SMA “collector” se divide en tres grupos lógicos de agentes (ver figura 3.4):

- **Colección:** Agrupa lógicamente al agente que recolecta y filtra información de los de eventos SYSLOG. La recolección es automática y se realiza cada vez que se detecta algún evento proveniente de la red.
- **Experto:** Agrupa lógicamente al agente que recibe los eventos previamente filtrados y en base a patrones, el agente realiza razonamiento para obtener conocimiento de lo que está sucediendo en la red. Además, el agente que actúa sobre la red y trata de corregir incidentes “flapping” que impacta en forma negativa en los servicios a los clientes.
- **Gráfico:** Agrupa lógicamente al agente que muestra conocimiento al operador del NOC. La información se muestra en forma gráfica.

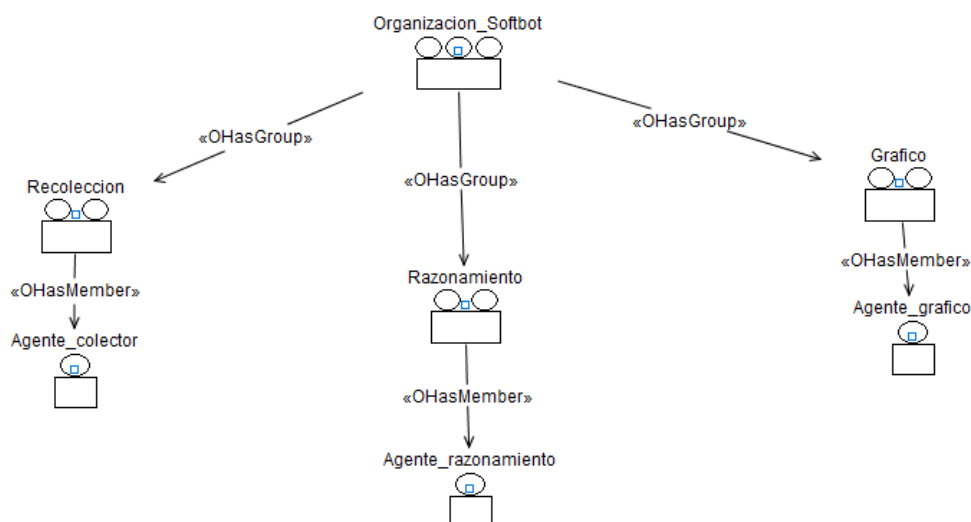


Figura 3.5.- Meta-modelo de organización del “softbot gestor de notificaciones”

3.2.3. Etapa de diseño del SMA

Se describirán los meta-modelos que implementan las características propuestas para el sistema.

3.2.3.1. Meta-modelo del “Agente Recolector”

En la tabla 3.1 muestra la definición cualitativa del “Agente Recolector” y la figura 3.6 muestra el modelo de “Agente Recolector”.

Tabla 3.11.- Definición cualitativa del “Agente Recolector”

Descripción del agente	El “Agente Recolector” gestiona la recolección de eventos obtenidos del servidor SYSLOG o “collector”. El agente verifica el formato de los eventos, los filtra, los transforma y los envía al “Agente Razonamiento”.
Funcionalidad del agente	T.1.- Abre un puerto de comunicaciones (SYSLOG

	“collector”) que escucha los datagramas (SYSLOG “originator”) que llegan de la red. T.2.- De todos los mensajes SYSLOG recibidos, solo se seleccionan y clasifican los eventos de acuerdo a su importancia, es decir se permiten aquellas notificaciones que tengan como facility a LINK, LINEPROTO, OSPF y con severity o severidad mayor o igual a 3, según lo definido en la etapa de especificación de requerimientos. T.3.- Se realiza la transformación del evento, obteniendo eventos derivados LINK, LINEPROTO, OSPF, etc., según lo definido en la etapa de especificación de requerimientos. T.4.- Almacenar los datos de los eventos filtrados en un archivo de texto.
Qué tareas sabe ejecutar el agente	• TLeerSyslog • TClasificarYFiltrar • TTransformar • TAlmacenarSyslog
Qué objetivos se compromete a alcanzar	“Gestión de Recolección”. Obtener eventos de la red WAN y transferirlos en forma adecuada a la “Gestión de Razonamiento”.
Estado mental del agente	• Estado de recolección activo

	• Estado de recolección en espera • Estado de transmisión
--	--

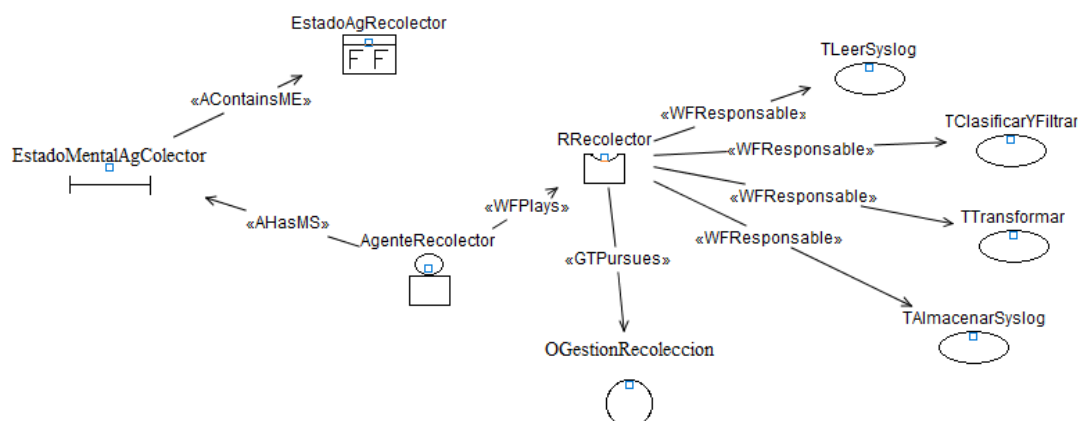


Figura 3.6.- Meta-modelo del Agente: “Recolector de Eventos”

El “Agente Recolector” tienen como objetivo la gestión de colección de eventos o “OGestionRecoleccion”. Para ello juega el rol de recolector o “RColector” y en ese rol ejecuta las tareas de leer, filtrar, clasificar, transformar y almacenar eventos SYSLOG (TLeerSyslog, TClasificarYFiltrar, TTransformar, TAlmacenarSyslog).

3.2.3.2. Meta-modelo Objetivos y Tareas del “Agente Recolector”

La tarea “TLeerSyslog” es una de las tareas que permiten cumplir el objetivo del agente “OGestionColeccion”. “TLeerSyslog” lee un evento o un conjunto de eventos del servidor syslog (AEventLogSlot). Los datos leídos están conformados por registros log. Los datos obtenidos se almacenan temporalmente en el slot “LogRecolectados”. La figura 3.7 muestra el modelo de tarea “TLeerYEnviarSyslog”.

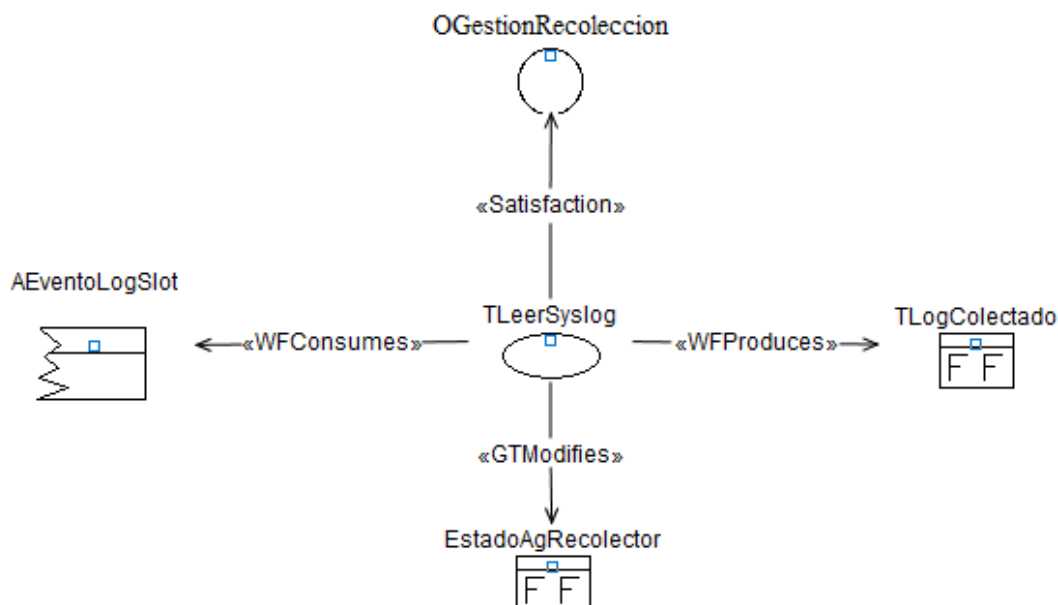


Figura 3.7.- Meta-modelo de Objetivos y Tareas: TLeerSyslog

La tarea “TClasificarYFiltrar” es una de las tareas que permiten cumplir el objetivo del agente “OGestionColeccion”. “TClasificarYFiltrar” lee el evento o “TLogColectado” de la tarea “TLeerSyslog”. Los datos son clasificados y filtrados según lo especificado en los requerimientos detallados, para después almacenarlos temporalmente en el slot “LogClasificadosYFiltrados”. La figura 3.8 muestra el modelo de tarea “TClasificarYFiltrar”.

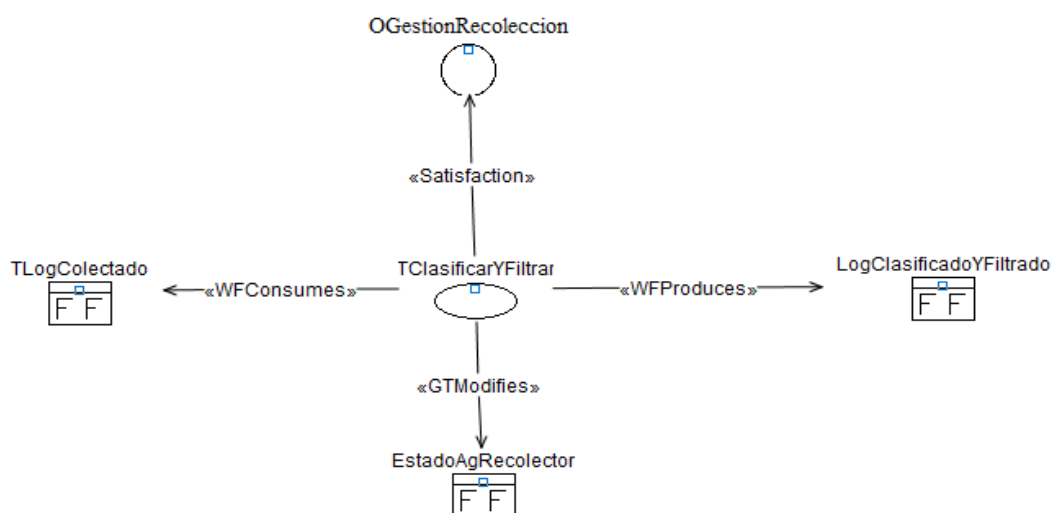


Figura 3.8.- Meta-modelo de Objetivos y Tareas: TClasificarYFiltrar

La tarea “TTransformar” es una de las tareas que permiten cumplir el objetivo del agente “OGestionColeccion”. “TTransformar” lee el evento o hecho “LogClasificadoYFiltrado” de la tarea “TClasificarYFiltrar”. Los datos son formateados y adecuados, para después almacenarlos temporalmente en el slot “LogTransformado”. La figura 3.9 muestra el modelo de tarea “TTransformar”.

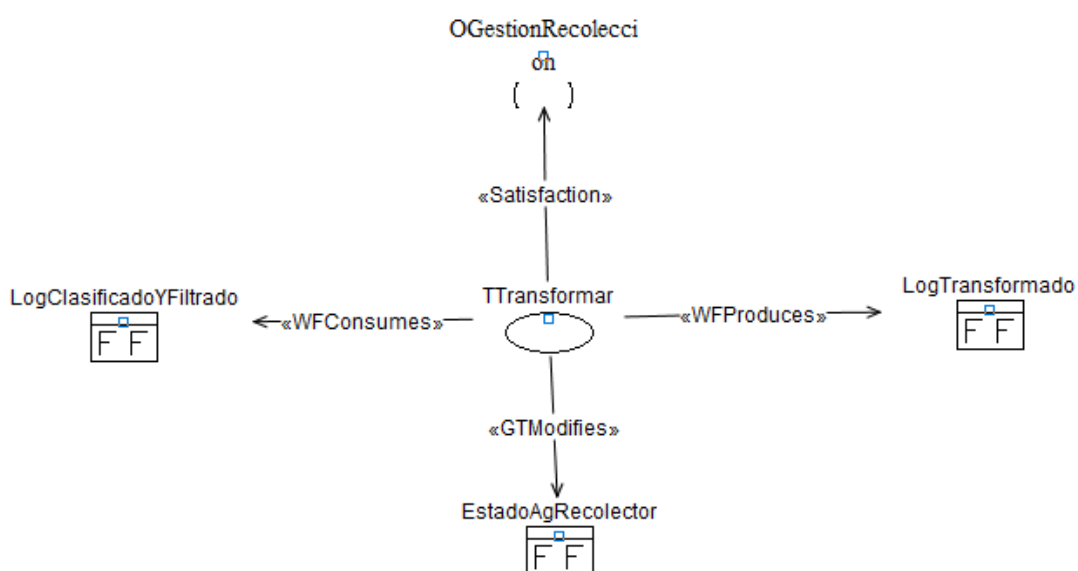


Figura 3.9.- Meta-modelo de Objetivos y Tareas: TTransformar

La tarea “TAlmacenarSyslog” es una de las tareas que permiten cumplir el objetivo del agente “OGestionColeccion”. “TAlmacenarSyslog” lee el evento o hecho “LogTransformado” de la tarea “TTransformar”. Los datos son almacenados en una base de datos. La figura 3.10 muestra el modelo de tarea “TAlmacenarSyslog”.

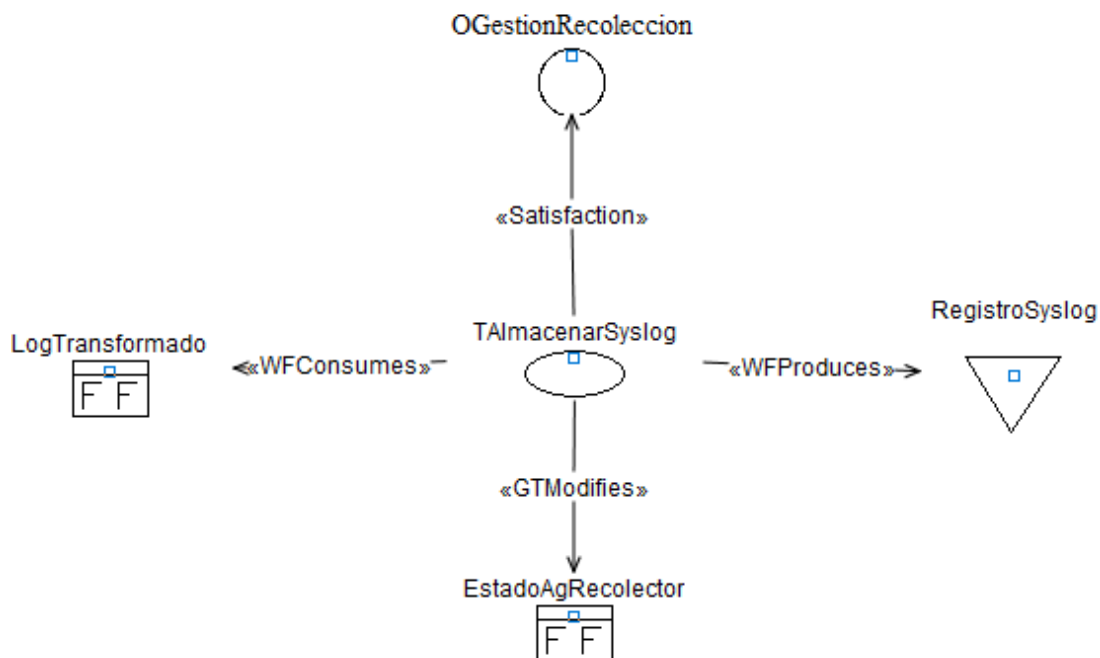


Figura 3.10.- Meta-modelo de Objetivos y Tareas: TAlmacenarSyslog

3.2.3.3. Meta-modelo de Interacción

El rol “RRecolector” del “Agente Recolector” interactúa con el rol “RSupervisor” del “Agente Supervisor”. La interacción entre los agentes Recolector y Supervisor es parte del análisis del sistema y define qué rol de agente inicia la interacción y que rol de agente responde. La figura 3.11 muestra las interacciones entre los agentes Recolector y Supervisor.

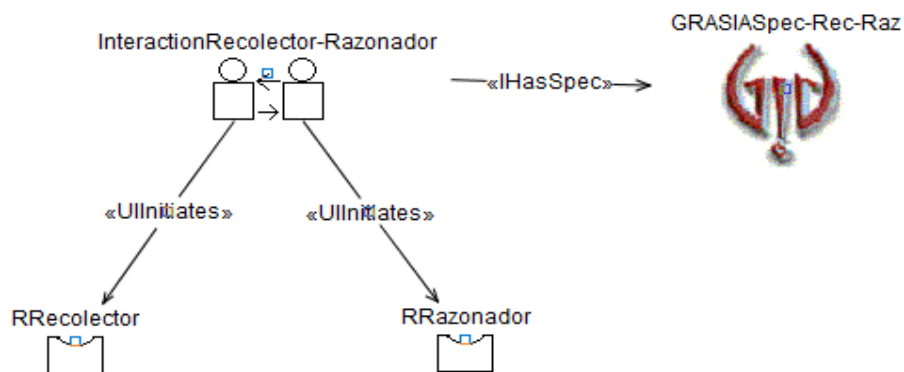


Figura 3.11.- Interacción (I). Envío de eventos filtrados del agente recolector al agente clasificador

El rol “RRecolector” inicia un requerimiento enviando los datos “LogsTransformado”. El “Agente Razonador” retornara a través del rol “RRazonador” un acuse de recibo, que servirá para que el “Agente Colector” cambie de estado mental y sepa que los datos fueron comunicados en forma satisfactoria. La figura 3.12 muestra la interacción (II). Requerimiento de envío de eventos log al “Agente Razonador”.

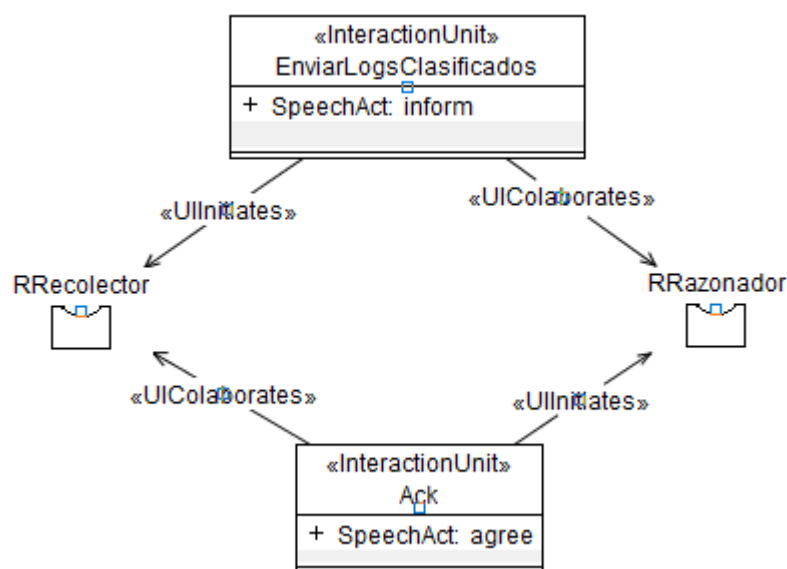


Figura 3.12.- Interacción (II). Envío de eventos filtrados del agente recolector al agente clasificador

3.2.3.4. Meta-modelo de Agente del “Agente Experto”

En la tabla 3.12 muestra la definición cualitativa del agente experto y la figura 3.13 muestra el modelo de agente experto.

Tabla 3.12.- Definición cualitativa del “Agente Experto”

Descripción del agente	El “Agente Experto” gestiona el razonamiento del softbot gestor de notificaciones en base a los eventos filtrados y clasificados, recibidos
------------------------	---

	desde la gestión de colección. En base a patrones previamente establecidos, el agente realiza razonamiento para obtener conocimiento de lo que está sucediendo en la red.
Funcionalidad del agente	T.1.- Ejecuta el CEP. Los eventos LINK, LINEPROTO, OSPF, etc, son detectados por el CEP, el cual, realiza el razonamiento automático especificado en los requerimientos del sistema. Genera la recomendación para la solución del problema y la almacena en un registro. T.2.- La ocurrencia de tres (3) eventos seguidos (“flapping”) en un minuto del tipo de mensajes: LINEPROTO-5-UPDOWN Hará que se detecte esta condición según lo indicado en la etapa de requerimientos del sistema. T.5.- En caso de la ocurrencia de eventos “flapping” (como lo especificado en T5) que generen alto procesamiento en los equipos y bajo rendimiento de la red, es detectado por la “Gestión de Razonamiento”, como medida de protección, el softbot gestor de notificaciones

	ejecuta la acción de apagado de la interface de red que está ocasionando la falla. T.6.- Almacenar los datos de la acción realizada en un registro.
Qué tareas sabe ejecutar el agente	• TRazonarCEP • TDetectarFlapping
Qué objetivos se compromete a alcanzar	El objetivo es la “Gestión de Razonamiento”. Obtener conclusiones y recomendaciones en forma automatizada a través de “Complex Event Processing”. En caso que se requiera, ejecutar acciones sobre la red para minimizar inestabilidad en la red o “flapping” apagando la interfaz de red intermitente. Almacenar en un registro el resultado de las acciones realizadas.
Estado mental del agente	• Estado de razonamiento activo • Estado de razonamiento en espera

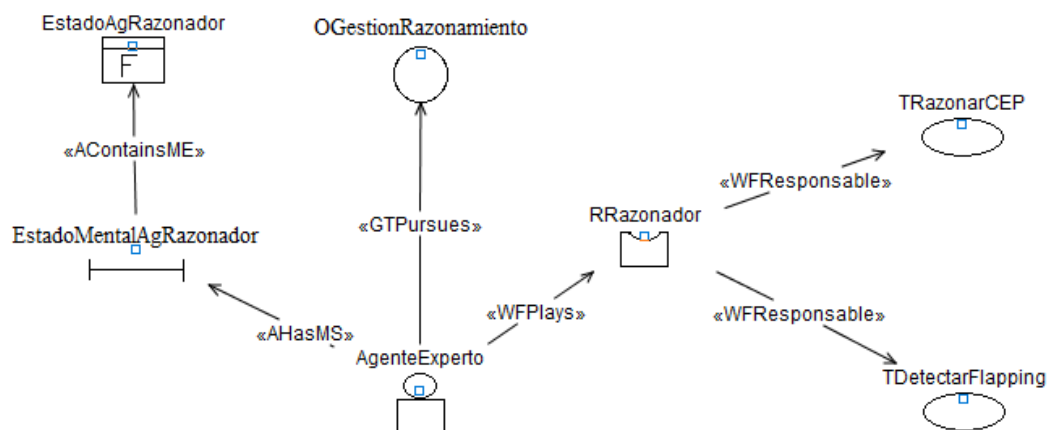


Figura 3.13.- Meta-modelo del Agente: Experto

3.2.3.5. Meta-modelo Objetivos y Tareas del “Agente Experto”

La tarea “TRazonarCEP” especificada en la figura 3.14, es una de las tareas que permiten cumplir el objetivo del agente “OGestionRazonamiento”. “TRazonarCEP” lee un evento filtrado (LogTransformado) del “Agente Colector”. Los datos leídos están conformados por registros de eventos críticos, los cuales son procesados por el motor “Complex Event Processing”. El resultado del razonamiento se almacena en un registro denominado “Recomendaciones”.

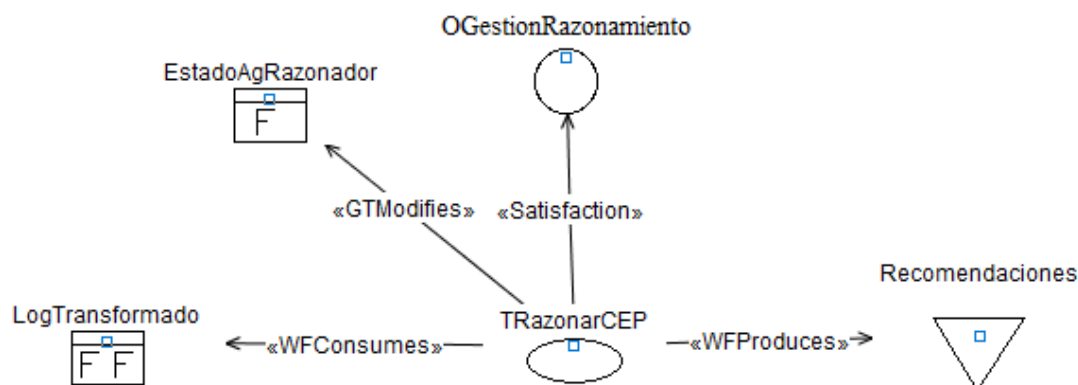


Figura 3.14.- Meta-modelo de Objetivos y Tareas: TRazonarCEP

La tarea “TDetectarFlapping” es una de las tareas que permiten cumplir el objetivo del agente “OGestionRazonamiento”. “TDetectarFlapping” lee el hecho “LogTransformado”

proveniente del del “AgenteRecolector” y como medida de protección, el softbot gestor de notificaciones ejecuta la acción de apagado de la interfaz de red que está ocasionando la falla. La tarea produce una acción sobre una aplicación (“expect”) dándole a ésta los datos para que la propia aplicación apague la interfaz de red intermitente. El resultado del trabajo de correlación de eventos y la acción se almacena en un registro denominado “AccionesRealizadas”. La figura 3.15 muestra el modelo de tarea “TDetectarFlapping”.

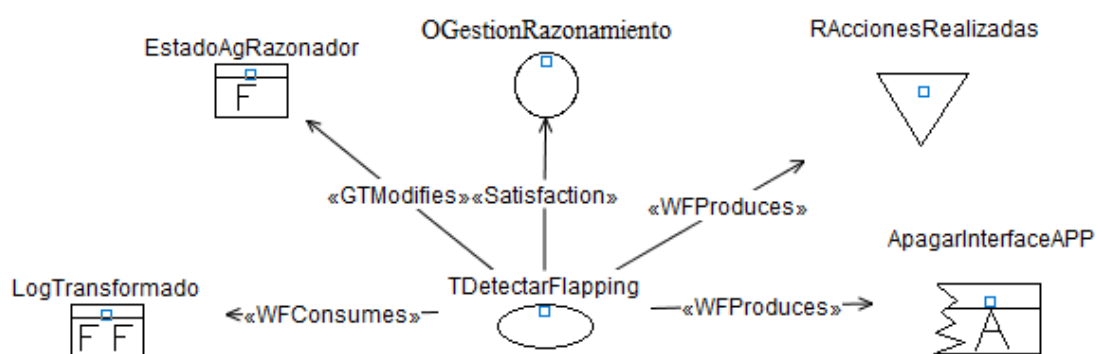


Figura 3.15.- Meta-modelo de Objetivos y Tareas: TDetectarFlapping

3.2.3.6. Meta-modelo de Agente del “Agente Gráfico”

En la tabla 3.13 muestra la definición cualitativa del “Agente Gráfico” y la figura 3.16 muestra el modelo de “Agente Gráfico”.

Tabla 3.13.- Definición cualitativa del “Agente Gráfico”

Descripción del agente	El “Agente Gráfico” gestiona la muestra conocimiento al operador del NOC. La información se muestra en forma gráfica.
Funcionalidad del agente	T.1.- Mediante ventana gráfica, el softbot gestor de notificaciones muestra los eventos SYSLOG al

	operador del NOC. T.2.- Mediante ventana gráfica, el softbot gestor de notificaciones muestra el resultado de las recomendaciones, soluciones y las acciones tomadas operador del NOC. T.3.- Mediante ventana gráfica, el softbot gestor de notificaciones muestra información de las acciones realizadas para mitigar el “flapping” en la red.
Qué tareas sabe ejecutar el agente	• TMuestraSyslog • TMuestraRecomendaciones • TMuestraAcciones
Qué objetivos se compromete a alcanzar	“Gestión de interface hombre-softbot”. Muestra la información en forma de ventanas gráficas a los operadores del NOC.

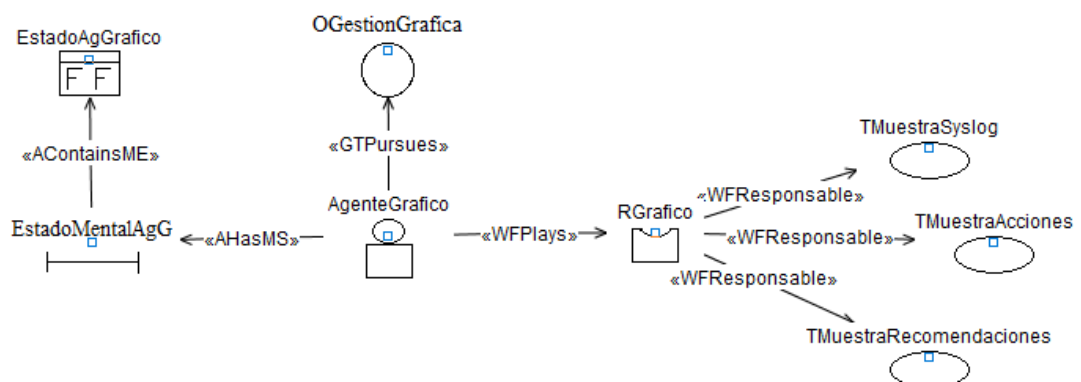


Figura 3.16.- Meta-modelo del Agente: Gráfico

La tarea “TMuestraSyslog” es una de las tareas que permiten cumplir el objetivo del agente “OGestionGrafico”. “TMuestraSyslog” muestra datos de los eventos SYSLOG. La figura 3.17 muestra el modelo de tarea “TMuestraSyslog”.

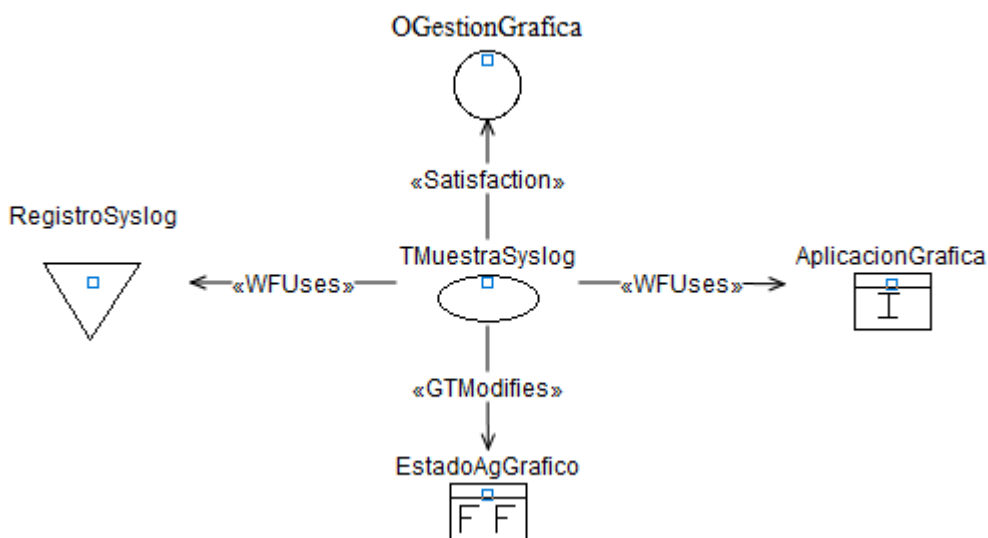


Figura 3.17.- Meta-modelo de Objetivos y Tareas: TMuestraSyslog

La tarea “TMuestraRecomendaciones” es una de las tareas que permiten cumplir el objetivo del agente “OGestionGrafica”. “TMuestraRecomendaciones” muestra al operador del NOC las recomendaciones que realiza el softbot gestor de notificaciones para la solución del problema. La figura 3.18 muestra el modelo de tarea “TMuestraRecomendaciones”.

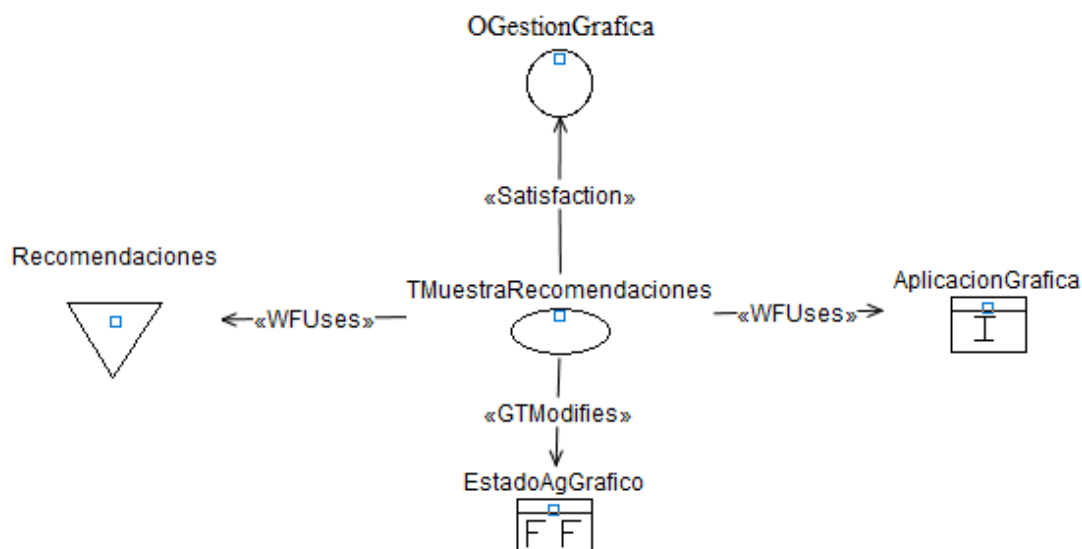


Figura 3.18.- Meta-modelo de Objetivos y Tareas: MuestraRecomendaciones

La tarea “**TMuestraAcciones**” es una de las tareas que permiten cumplir el objetivo del agente “**OGestionGrafica**”. “**TMuestraAcciones**” muestra datos de las acciones realizadas por el softbot gestor de notificaciones sobre la red. La figura 3.19 muestra el modelo de tarea “**TMuestraAcciones**”.

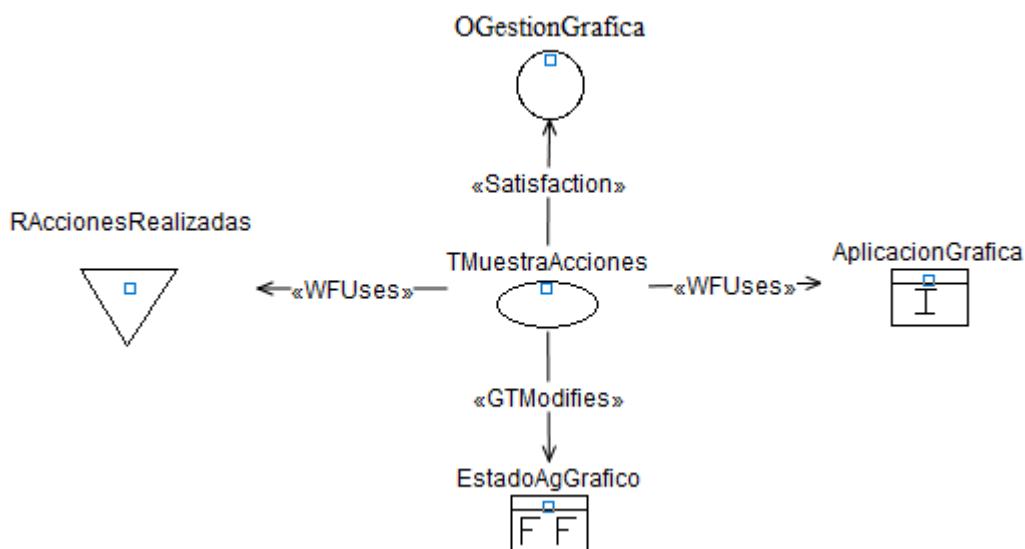


Figura 3.19.- Meta-modelo de Objetivos y Tareas: TMuestraAcciones

3.3. Ontología

Para comunicar los agentes se ha definido una ontología llamada “SoftbotOntology” la cual corresponde a la definición formal de los tipos, propiedades, y relaciones entre los agentes que se van a comunicar para el dominio donde los agentes van a dialogar. Esta ontología es bastante sencilla. La ontología consta del concepto LOG_FILTRADO, el cual tiene un conjunto de atributos que los agentes (Recolector y Experto) que se comunican comprenden. El agente recolector recibe y filtra las notificaciones provenientes de la red, generando un “java bean” llamado LOG_FILTRADO. El agente recolector comunica al agente experto el log filtrado. La tabla 3.15 muestra los atributos del concepto LOG_FILTRADO:

Tabla 3.14.- Atributos del concepto LOG_FILTRADO

Concepto	Atributos del concepto
LOG_FILTRADO	LOG_FILTRADO_DIRECCION_IP
	LOG_FILTRADO_DATE
	LOG_FILTRADO_HOUR
	LOG_FILTRADO_FACILITY
	LOG_FILTRADO_SEVERITY
	LOG_FILTRADO_MNEMONIC
	LOG_FILTRADO_DESCRIPCION

Las notificaciones previamente filtradas por el agente recolector son comunicadas al agente experto mediante una acción denominada ENVIAR_LOG_FILTRADO. Esta acción comunica el concepto LOG_FILTRADO, como se observa en la tabla 3.16.

Tabla 3.15.- Diálogo entre el agente recolector y el agente experto

Agente origen	Agente destino	Concepto	Acción
Recolector	Experto	LOG_FILTRADO	ENVIAR_LOG_FILTRADO

CAPÍTULO IV

4. RESULTADOS Y DISCUSIÓN

4.1. Metodología de pruebas del producto a desarrollar

4.1.1. Unidades de prueba

Las unidades de prueba son notificaciones provenientes de una red IP. Durante el proceso de análisis de notificaciones por parte del operador de red, se define:

- El tiempo que tarda un operador en identificar y determinar un problema de red entre un conjunto notificaciones recibidas desde la red.
- El tiempo que tarda un operador en identificar y mitigar un problema de inestabilidad en la red entre un conjunto notificaciones recibidas desde la red.

4.1.2. Población para pruebas

Una red virtual de laboratorio basada en la tecnología IP, configurada con el protocolo de enrutamiento dinámico OSPF que genera mensajes o notificaciones. La red virtual está compuesta por cinco enrutadores o nodos que emulan una red WAN (Trujillo, Arequipa, Piura, Cuzco y Lima) y un software desarrollado en forma personal que envía notificaciones en forma programada desde uno hasta un millón de notificaciones de prueba.

4.1.3. Tamaño de la muestra para pruebas finales

Por propósitos de prueba, se han escogido en forma controlada dos mil notificaciones poco importantes con una notificación importante. Se ha escogido las dos mil notificaciones para que el operador de red realice las pruebas de análisis en forma manual, esta cantidad de notificaciones se justifica debido a que el tiempo de demora en identificar un fallo entre dos mil notificaciones es aproximadamente de 15 minutos. Si se toma un millón de notificaciones el operador podría demorar hasta 5 días enteros en encontrar el problema, lo

cual no es realista en un entorno real. El tiempo que demora el operador en realizar las pruebas se debe comparar con el tiempo que demora el softbot gestor de notificaciones en realizar las mismas pruebas.

El tamaño de los datos de muestra en 2000 notificaciones (entrada) que tiene un tamaño de 196KB. Las notificaciones son generadas por la red de prueba y el software generador de notificaciones.

Además, la cantidad de notificaciones se reciben en forma continua en el tiempo y tiende a ser infinita, solo como referencia, si se tiene un tamaño de la población de 1'000,000 de eventos (N), con un nivel de confianza del 85% (z, que depende el valor de confianza), y un margen de error (e) del 5%, aplicando la fórmula:

$$\text{Tamaño de la muestra} = \frac{\frac{z^2 \times p(1-p)}{e^2}}{1 + \left(\frac{z^2 \times p(1-p)}{e^2 N} \right)}$$

Se obtiene un tamaño de muestra de 208 notificaciones, lo cual es muy cercano a la muestra de 200 tomada del tiempo promedio que demora un operador en el diagnóstico de una falla de la red. Por lo que, para la simulación se tomarán 200 notificaciones para las pruebas finales del sistema.

4.1.4. Variables

- **Variable independiente:** notificaciones recibidas desde la red (**n**).
- **Variables dependientes:** el tiempo que demora el operador en realizar el diagnóstico de un fallo de red (**Y1**) y el tiempo que demora el operador en mitigar problemas de inestabilidad OSPF en la red (**Y2**).

$$Y1 = f(n)$$

$$Y2 = f(n)$$

4.1.5. Técnica de recolección de datos para pruebas

La técnica empleada para coleccionar los datos es la recolección de notificaciones provenientes de la red y del software generador de notificaciones a través del protocolo SYSLOG.

4.1.6. Análisis e interpretación de la información resultante de las pruebas

Después del análisis, diseño y despliegue del sistema en un entorno controlado, se generan 2000 notificaciones poco importantes y una notificación importante. Entonces:

- a) Se toma el tiempo que un operador poco experto demora en encontrar un fallo en el entorno controlado y se compara con el tiempo que demora el softbot gestor de notificaciones en encontrar el mismo fallo.
- b) Se repite 19 veces más el paso a). para 19 diferentes tipos de fallo.

Para las 20 pruebas realizadas se define:

- El tiempo promedio de identificación y planteamiento de solución manual ante una notificación importante entre 2000 notificaciones recibidas para un operador poco experto: **TPM**.
- El tiempo promedio de identificación y de solución manual ante un conjunto de notificaciones de intermitencia OSPF entre 2000 notificaciones recibidas para un operador poco experto: **TPSM**

El softbot gestor de notificaciones tiene como prueba objetivo realizar el mismo trabajo de los operadores para reducir **TPM** y **TPSM**.

4.2. Etapa previa antes de la ejecución de las pruebas

4.2.1. Entorno de pruebas

Se verifica todos los componentes de la red, tanto física como emulada. Por parte de la red emulada con GNS3, esta se conecta a la red física donde se encuentra el “Softbot Multiagente” a través de un switch de gestión llamado “SWITCHGESTION”. Cada enrutador de la red tiene una interfaz que se interconecta a dicho switch. A través de esta interfaz el enrutador envía las notificaciones que autogenera al “Softbot Multiagente” que se encuentre detrás de un bridge llamado “Cloud1” como se observa en la figura 4.1.

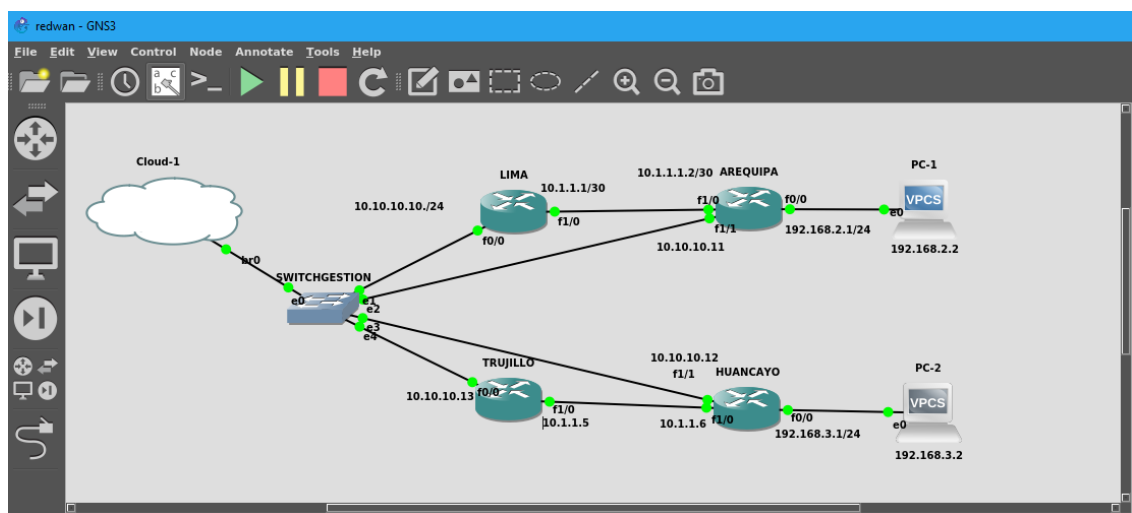
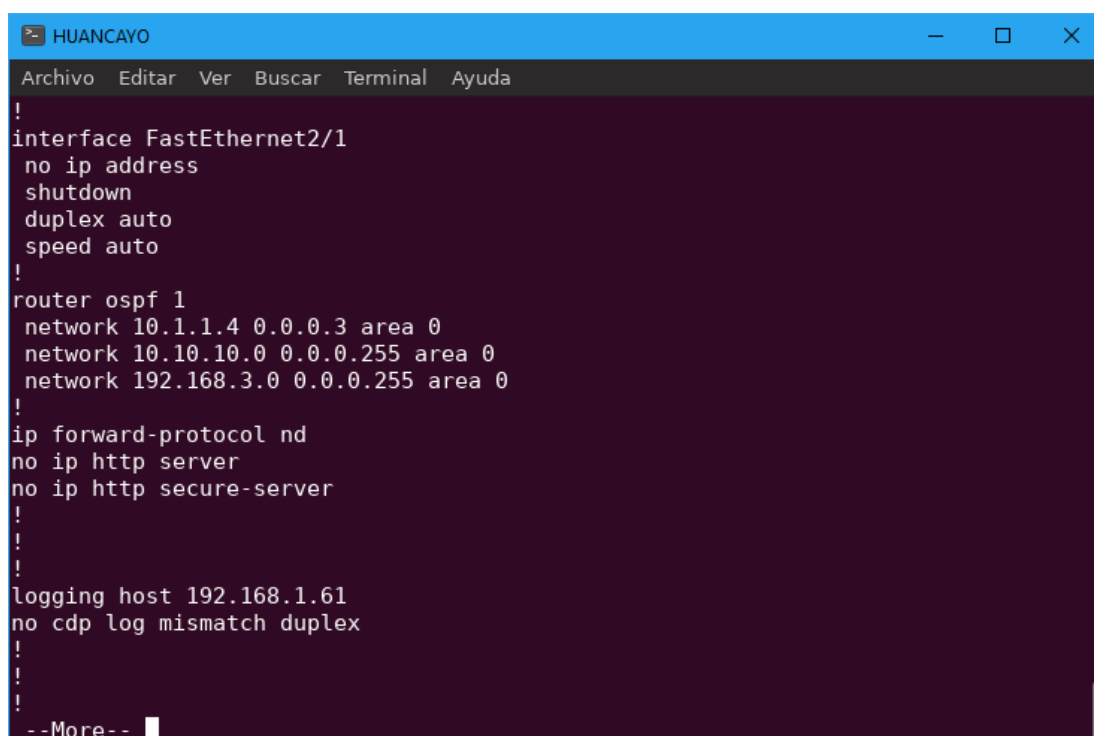


Figura 4.1.- Imagen de la red y el entorno de pruebas

Los enrutadores están configurados con el protocolo OSPF. Por ejemplo, la configuración del enrutador “HUANCAYO” se muestra en la figura 4.2. Además, se observa que el enrutador está configurado para el envío de todos los mensajes SYSLOG auto generados por el propio enrutador al “Softbot Multiagente” (192.168.1.61).



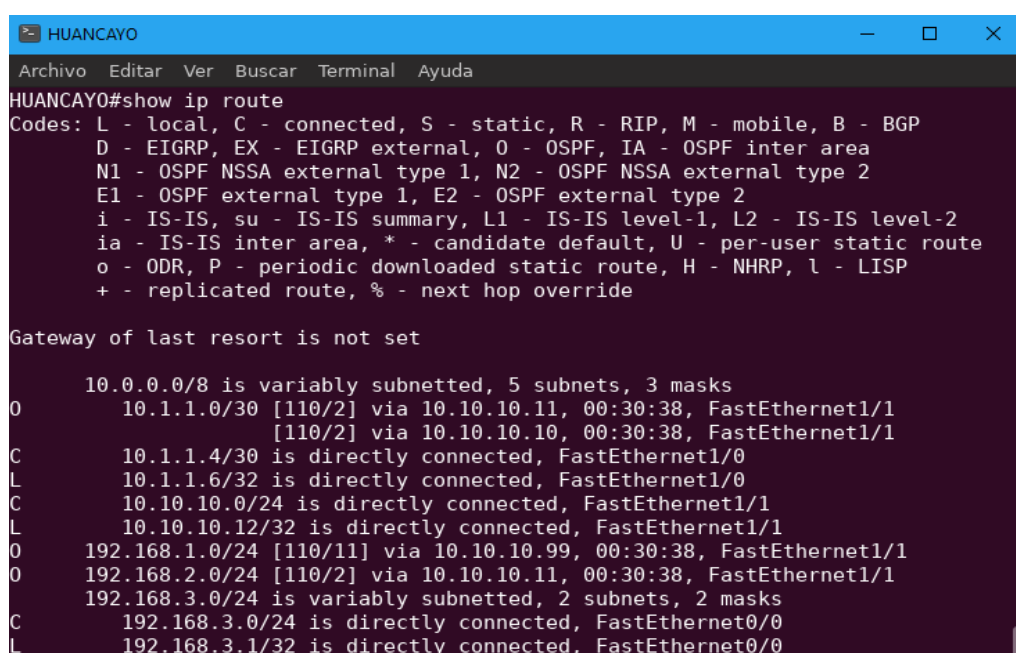
```

HUANCAYO
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
!
interface FastEthernet2/1
  no ip address
  shutdown
  duplex auto
  speed auto
!
router ospf 1
  network 10.1.1.4 0.0.0.3 area 0
  network 10.10.10.0 0.0.0.255 area 0
  network 192.168.3.0 0.0.0.255 area 0
!
ip forward-protocol nd
no ip http server
no ip http secure-server
!
!
!
logging host 192.168.1.61
no cdp log mismatch duplex
!
!
!
--More--

```

Figura 4.2: Configuración OSPF y SYSLOG del enrutador “HUANCAYO”

La figura 4.3 muestra cómo, a través de enrutamiento dinámico con OSPF, la red de “HUANCAYO” “descubre” todas las redes, incluido la red de gestión donde se encuentra el “Softbot Multiagente”.



```

HUANCAYO
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
HUANCAYO#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

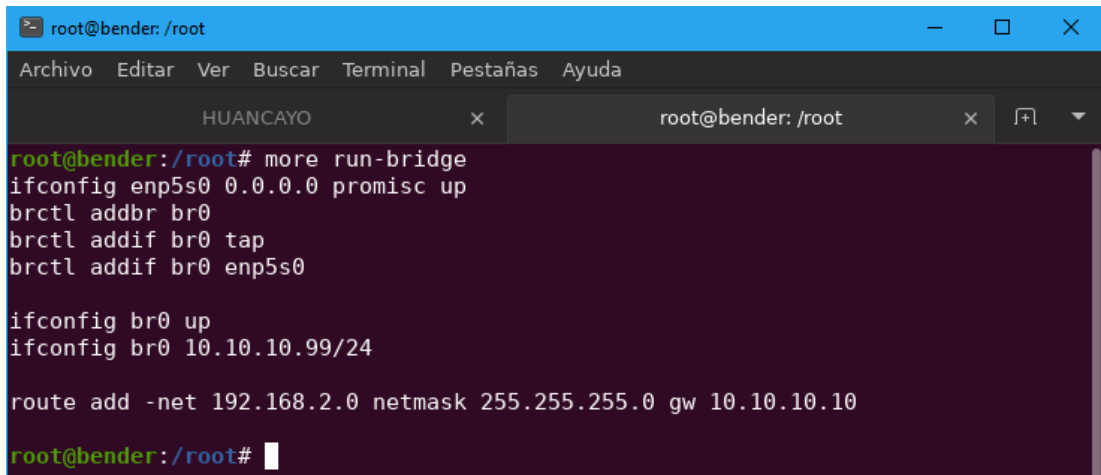
Gateway of last resort is not set

10.0.0.0/8 is variably subnetted, 5 subnets, 3 masks
O   10.1.1.0/30 [110/2] via 10.10.10.11, 00:30:38, FastEthernet1/1
    [110/2] via 10.10.10.10, 00:30:38, FastEthernet1/1
C   10.1.1.4/30 is directly connected, FastEthernet1/0
L   10.1.1.6/32 is directly connected, FastEthernet1/0
C   10.10.10.0/24 is directly connected, FastEthernet1/1
L   10.10.10.12/32 is directly connected, FastEthernet1/1
O   192.168.1.0/24 [110/11] via 10.10.10.99, 00:30:38, FastEthernet1/1
O   192.168.2.0/24 [110/2] via 10.10.10.11, 00:30:38, FastEthernet1/1
    192.168.3.0/24 is variably subnetted, 2 subnets, 2 masks
C   192.168.3.0/24 is directly connected, FastEthernet0/0
L   192.168.3.1/32 is directly connected, FastEthernet0/0

```

Figura 4.3: Redes que conoce el enrutador “HUANCAYO”

La red WAN emulada con GNS3 envía todas las notificaciones autogeneradas al “Softbot Multiagente”. Para ello está establecido un “bridge” o puente ethernet entre la red WAN emulada y la red física (red donde se encuentra el “Softbot Multiagente”). La figura 4.4 muestra la configuración del Linux Ubuntu para que funcione como “bridge” entre ambas redes.



```
root@bender: /root
Archivo  Editar  Ver  Buscar  Terminal  Pestañas  Ayuda
HUANCAYO x root@bender: /root x
root@bender:/root# more run-bridge
ifconfig enp5s0 0.0.0.0 promisc up
brctl addbr br0
brctl addif br0 tap
brctl addif br0 enp5s0

ifconfig br0 up
ifconfig br0 10.10.10.99/24

route add -net 192.168.2.0 netmask 255.255.255.0 gw 10.10.10.10
root@bender:/root#
```

Figura 4.4: Configuración “bridge” entre red emulada y red física

La red de gestión donde se encuentra el “Softbot Multiagente” debe ser conocida por la red WAN que emplea como protocolo de enrutamiento dinámico a “OSPF”. A través del software “QUAGGA” (que implementa OSPF en Linux) instalado en el sistema operativo Linux Ubuntu, se anuncia por el protocolo “OSPF” la red de gestión a la red a la red gestionada para que la red de gestión sea alcanzable por todos los enrutadores de la red gestionada. La figura 4.5 muestra la tabla de enrutamiento conocida por el software “QUAGGA” del equipo donde está el “Softbot Multiagente” donde se aprecia las rutas hacia todas las redes que conforman la red gestionada.

```

root@bender:/root# telnet localhost zebra
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Hello, this is Quagga (version 1.2.4).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

User Access Verification

Password:
Router> ena
Password:
Router# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel, N - NHRP,
       > - selected route, * - FIB route

O>* 10.1.1.0/30 [110/11] via 10.10.10.10, br0, 01:45:16
   *                via 10.10.10.11, br0, 01:45:16
O>* 10.1.1.4/30 [110/11] via 10.10.10.12, br0, 01:24:45
O  10.10.10.0/24 [110/10] is directly connected, br0, 01:49:22
C>* 10.10.10.0/24 is directly connected, br0
C>* 127.0.0.0/8 is directly connected, lo
K * 169.254.0.0/16 is directly connected, virbr0 inactive
O  192.168.1.0/24 [110/10] is directly connected, wlan0, 02:20:30
C>* 192.168.1.0/24 is directly connected, wlan0
O  192.168.2.0/24 [110/11] via 10.10.10.11, br0, 01:45:46
K>* 192.168.2.0/24 via 10.10.10.10, br0
O>* 192.168.3.0/24 [110/11] via 10.10.10.12, br0, 01:19:52

```

Figura 4.5: Tabla de enrutamiento del equipo de gestión

4.2.2. Verificación de la plataforma de desarrollo

Se ha usado el software Eclipse IDE 4.12.0 con las siguientes librerías externas instaladas para el correcto desarrollo del sistema (ver figura 4.6).

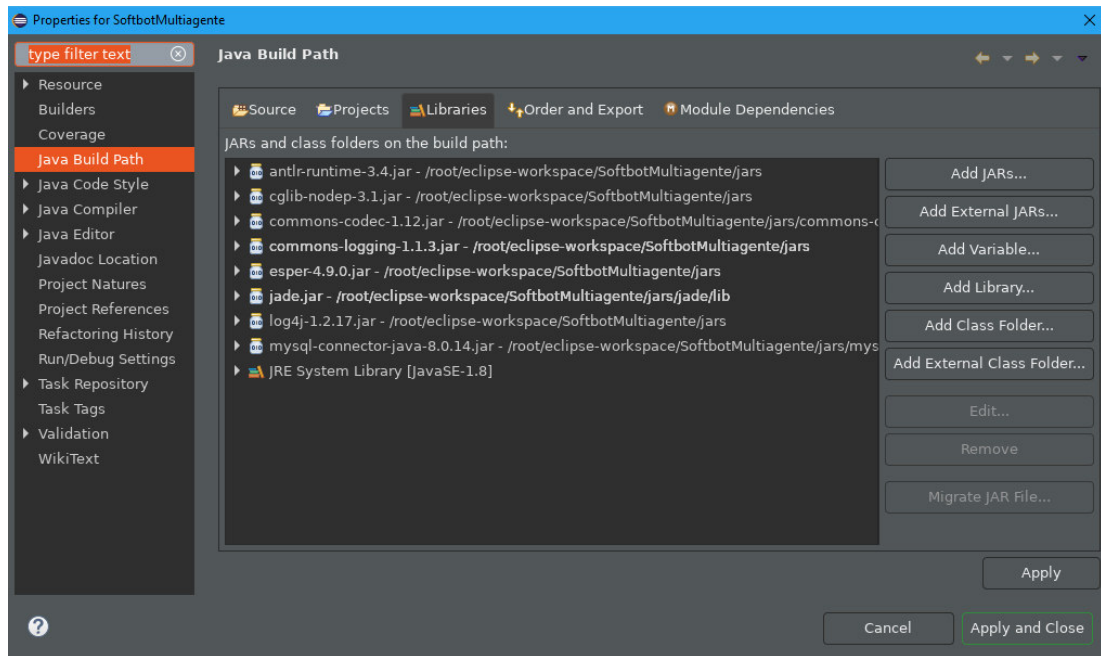


Figura 4.6.- Librerías externas empleadas por el IDE de desarrollo del softbot multiagente

Las librerías externas empleadas para el desarrollo del softbot multiagente fueron las siguientes:

- **antlr-runtime-3.4.jar:** ANTLR (ANother Tool for Language Recognition) es una herramienta que opera sobre lenguajes, proporcionando un marco para construir reconocedores (parsers), intérpretes, compiladores y traductores de lenguajes a partir de las descripciones gramaticales de los mismos (conteniendo acciones semánticas a realizarse en varios lenguajes de programación). Esta librería es utilizada por la librería para “Complex Event Processing”, Esper.
- **cglib-nodep-3.1.jar:** CGLIB (Byte Code Generation Library) es una API de alto nivel para generar y transformar el “bytecode” de Java. Es utilizado en AOP (programación orientada a aspectos), para pruebas y marcos de acceso a datos para generar objetos proxy dinámicos. Esta librería es utilizada por la librería para “Complex Event Processing”, Esper.

- **commons-codec-1.12.jar:** El software Apache Commons Codec proporciona implementaciones de codificadores y decodificadores comunes como Base64, Hex, Phonetics y URL.
- **esper-4.9.0.jar:** Esper es un producto de software de código abierto basado en Java para el “Complex Event Processing” (CEP) y procesamiento de flujo de eventos, que analiza series de eventos para derivar conclusiones de ellos. El softbot emplea esta librería para identificar en tiempo real patrones de eventos generados por la red de enrutadores.
- **jade.jar:** JADE (Java Agent DEvelopment Framework) es un marco de software completamente implementado en el lenguaje Java. Simplifica la implementación de sistemas de múltiples agentes a través de un middleware que cumple con las especificaciones FIPA y a través de un conjunto de herramientas gráficas que soportan las fases de depuración e implementación.
- **mysql-connector-java-8.0.14.jar:** Mysql Connector es un driver creado para Mysql que permite trabajar con Mysql desde programas escritos en Java.
- **log4j-1.2.17.jar:** Log4j es una biblioteca open source desarrollada en Java por la Apache Software Foundation que permite a los desarrolladores de software escribir mensajes de registro, cuyo propósito es dejar constancia de una determinada transacción en tiempo de ejecución.

4.3. Prueba de los agentes desarrollados en el softbot gestor de notificaciones

4.3.1. El “Agente Recolector”

El “Agente Recolector” se registra como servicio en el AMS de la plataforma de agentes JADE. El nombre de registro para este agente es “Agente_Recolector” (ver figura 4.7).

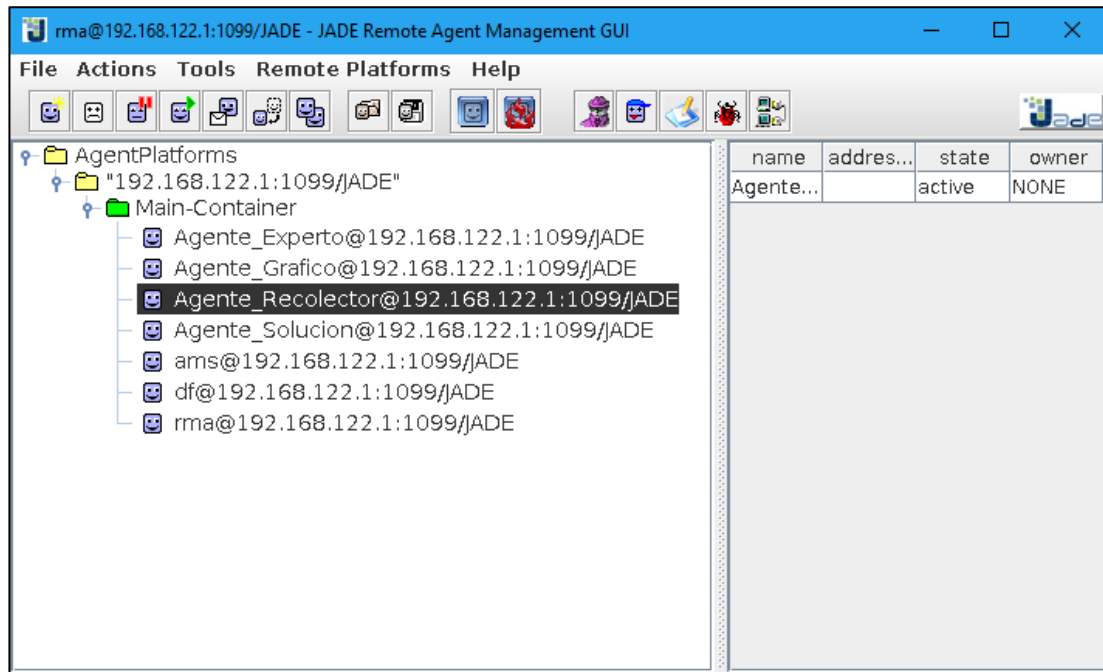
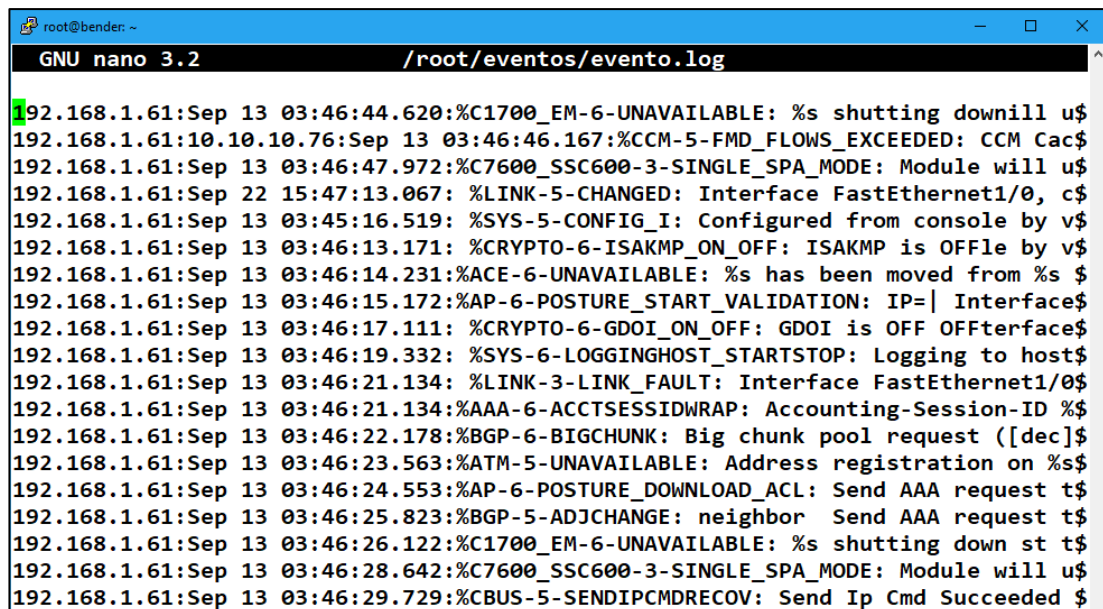


Figura 4.7.- Registro del “Agente_Recolector” en el framework Jade

El “Agente Recolector” hace uso del comportamiento “CyclicBehaviour”, el cual en forma continua ejecuta el comportamiento del agente en forma cíclica. El comportamiento del agente es el siguiente:

- a) Abre el puerto UDP 514 a la espera de mensajes SYSLOG provenientes de la red. Todo mensaje SYSLOG entrante es registrado en el archivo “/root/eventos/eventos.log”, según se observa en 4.8.



```

root@bender: ~
GNU nano 3.2 /root/eventos/evento.log
192.168.1.61:Sep 13 03:46:44.620:%C1700_EM-6-UNAVAILABLE: %s shutting downll u$
192.168.1.61:10.10.10.76:Sep 13 03:46:46.167:%CCM-5-FMD_FLOWS_EXCEEDED: CCM Cac$
192.168.1.61:Sep 13 03:46:47.972:%C7600_SSC600-3-SINGLE_SPA_MODE: Module will u$
192.168.1.61:Sep 22 15:47:13.067: %LINK-5-CHANGED: Interface FastEthernet1/0, c$
192.168.1.61:Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by v$
192.168.1.61:Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFFle by v$
192.168.1.61:Sep 13 03:46:14.231:%ACE-6-UNAVAILABLE: %s has been moved from %s $
192.168.1.61:Sep 13 03:46:15.172:%AP-6-POSTURE_START_VALIDATION: IP=| Interface$
192.168.1.61:Sep 13 03:46:17.111: %CRYPTO-6-GDOI_ON_OFF: GDOI is OFF OFFterface$
192.168.1.61:Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host$
192.168.1.61:Sep 13 03:46:21.134: %LINK-3-LINK_FAULT: Interface FastEthernet1/0$
192.168.1.61:Sep 13 03:46:21.134:%AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %$
192.168.1.61:Sep 13 03:46:22.178:%BGP-6-BIGCHUNK: Big chunk pool request ([dec]$
192.168.1.61:Sep 13 03:46:23.563:%ATM-5-UNAVAILABLE: Address registration on %s$
192.168.1.61:Sep 13 03:46:24.553:%AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request t$
192.168.1.61:Sep 13 03:46:25.823:%BGP-5-ADJCHANGE: neighbor Send AAA request t$
192.168.1.61:Sep 13 03:46:26.122:%C1700_EM-6-UNAVAILABLE: %s shutting down st t$
192.168.1.61:Sep 13 03:46:28.642:%C7600_SSC600-3-SINGLE_SPA_MODE: Module will u$
192.168.1.61:Sep 13 03:46:29.729:%CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded $

```

Figura 4.8.- Eventos registrados por el “Agente_Recolector”

b) Haciendo uso de expresiones regulares (ver figura 4.9), se realiza el filtrado de eventos, donde solo es permitido un conjunto limitado de eventos (previamente identificados en la etapa de análisis del sistema) para procesarlos mediante “Complex Event Processing”. Aquellos que están permitidos para procesar tienen las siguientes palabras clave:

- LINK-3-UPDOWN
- LINEPROTO-5-UPDOWN
- LINK-5-CHANGED
- OSPF-5-ADJCHG
- OSPF-4-CONFLICTING_LSAID
- SFF8472-5-THRESHOLD_VIOLATION
- OSPF-4-ERRRCV
- PLATFORMASM-0-FAULT_RECOVERY_ERR
- PLATFORMASM-3-MALLOC_FAILED
- PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN
- PLATFORMENVMON-2-CHASSIS_OVERTEMP
- PLATFORMENVMON-2-FAN_FAIL
- PLATFORMENVMON-2-FAN

- LINK-3-LINK_FAULT
- LINK-3-FCS_ERROR
- LINK-3-TOOSMALL
- CDP-4-NATIVE_VLAN_MISMATCH
- EQUIPMENT-5-DOWN
- EQUIPMENT-5-UP

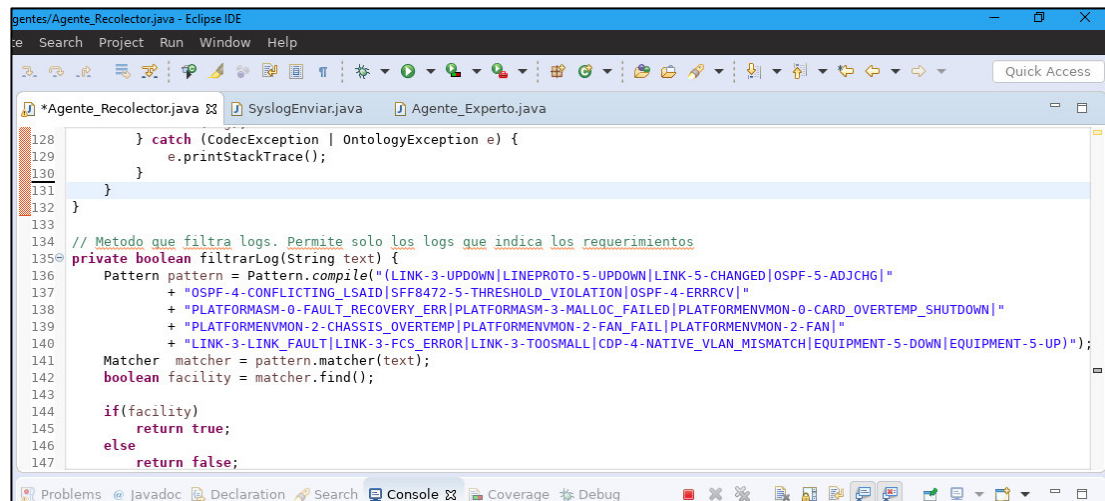


Figura 4.9.- Bloque de código que realiza filtrado de eventos a través de expresiones regulares

- Cada mensaje de evento previamente filtrado es decodificado y cada uno de los componentes de su contenido el almacenado en un objeto “java bean” llamado “LogBean”.
- El “Agente Colector” busca en el AMS de la plataforma JADE al “Agente Razonador”.
- Mediante un acto comunicativo, el “Agente Colector”, comunica y ordena “Agente Razonador” procesar los datos enviados según se observa en la figura 4.10.

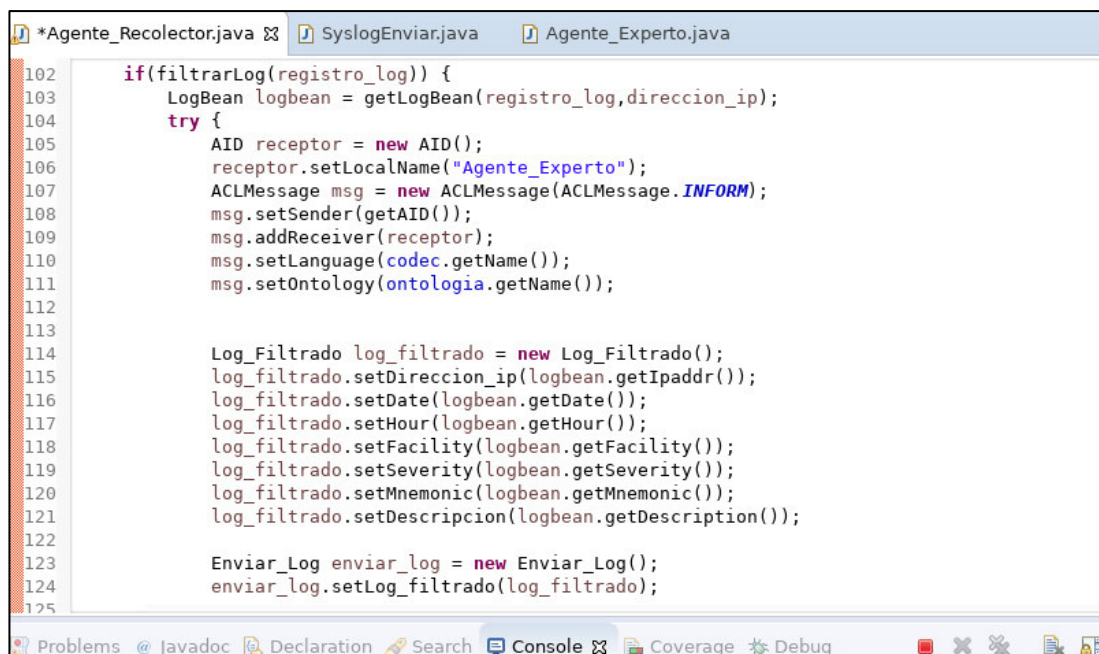


Figura 4.10.- Bloque de código que muestra la comunicación que realiza el “Agente_Recolector” con el “Agente_Experto”

4.3.2. El “Agente Experto”

El “Agente Experto” se registra en el AMS de la plataforma de agentes JADE según se muestra en la figura 4.11. El nombre de registro para este agente es "Agente_Experto".

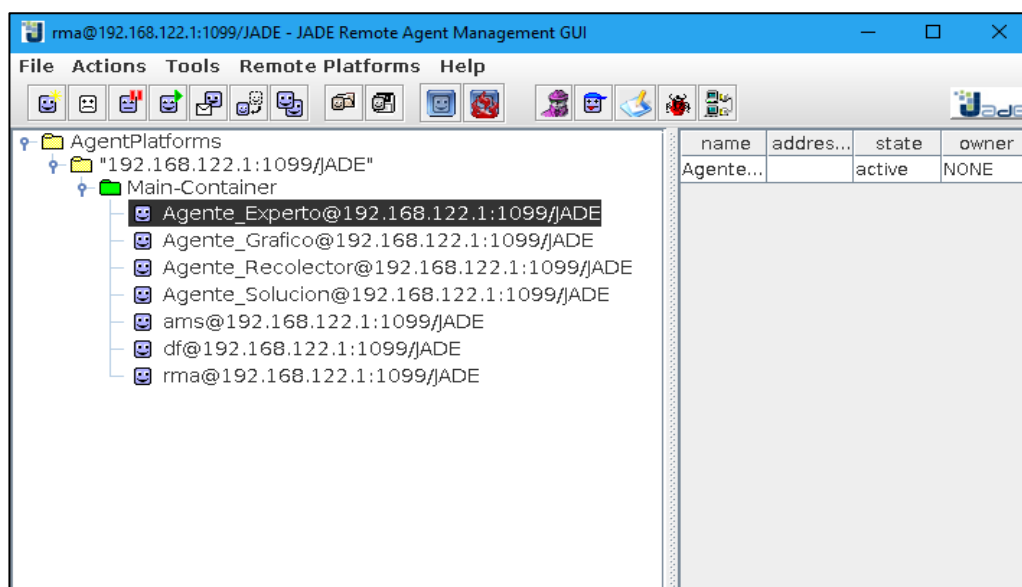


Figura 4.11.- Registro del “Agente_Experto” en el framework Jade

El “Agente Experto” realiza el “Complex Event Processing” en base a los eventos que son comunicados por el “Agente_Recolector”. El “Agente_Experto” hace uso del comportamiento “SimpleBehaviour”, el cual, se ejecuta cada vez que el agente recibe un evento por parte del “Agente Razonador”. El comportamiento del agente es el siguiente:

- a) Configura y arranca el motor de razonamiento
- b) Configura y arranca los “listeners” o “escuchadores” de “Esper”.

Los “listeners” capturan los eventos complejos definidos en el proceso de análisis del sistema. En la figura 4.12 se muestra una lista parcial de las clases “listeners”, así como, el código fuente de arranque del motor de razonamiento “Esper”.

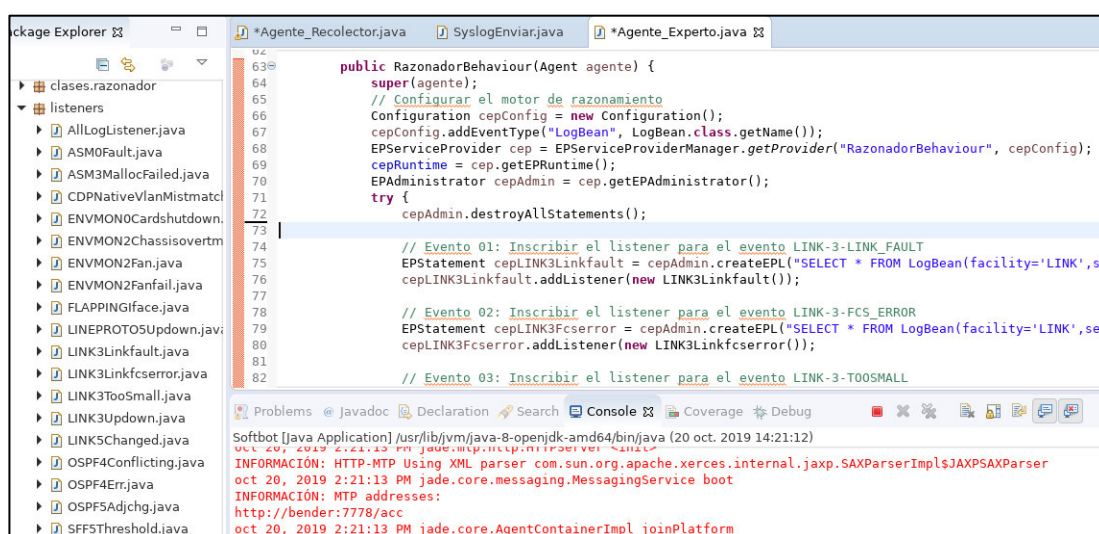


Figura 4.12.- Lista parcial de las clases “listeners” y código de arranque del motor de razonamiento

- c) Mediante la librería “Esper”, se han creado en base a un lenguaje similar al “SQL” un conjunto de “listeners” para eventos que cumplen las siguientes condiciones mostradas en la tabla 4.1:

Tabla 4.1.- “listeners” definidos mediante el lenguaje “ESPER”

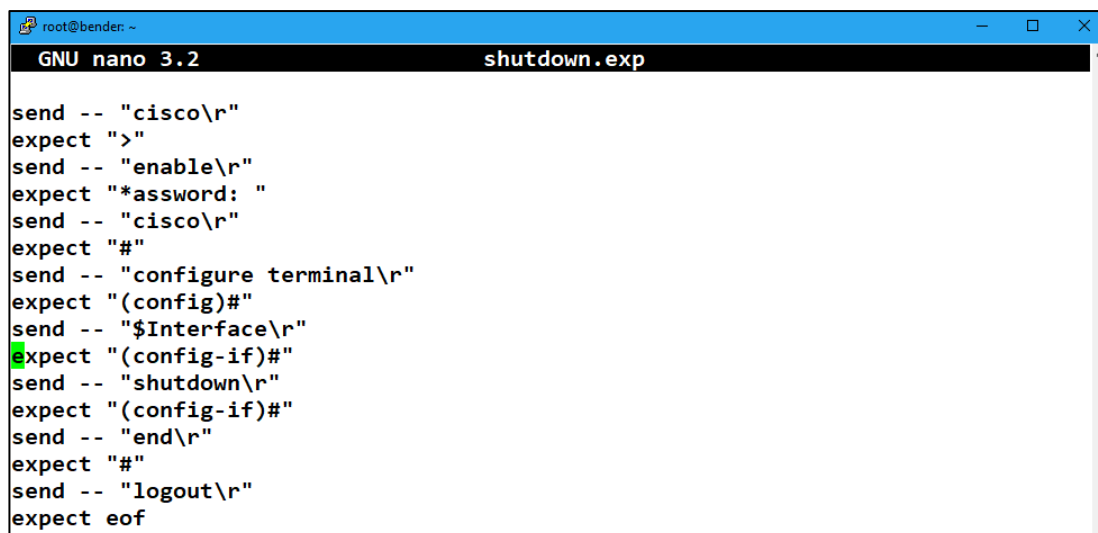
Descripción del “listener”	Definición del “listener”
----------------------------	---------------------------

Listener para el evento LINK-3-LINK_FAULT	SELECT * FROM LogBean(facility='LINK',severity='3',mnemonic='LINK_FAULT')
Listener para el evento LINK-3-FCS_ERROR	SELECT * FROM LogBean(facility='LINK',severity='3',mnemonic='FCS_ERROR')
Listener para el evento LINK-3-TOOSMALL	SELECT * FROM LogBean(facility='LINK',severity='3',mnemonic='TOOSMALL')
Listener para el evento LINK-3-UPDOWN	SELECT * FROM LogBean(facility='LINK',severity='3',mnemonic='UPDOWN')
Listener para el evento LINK-5-CHANGED	SELECT * FROM LogBean(facility='LINK',severity='5',mnemonic='CHANGED')
listener para el evento LINEPROTO-5-UPDOWN	SELECT * FROM LogBean(facility='LINEPROTO',severity='5',mnemonic='UPDOWN')
Listener para el evento OSPF-5-ADJCHG	SELECT * FROM LogBean(facility='OSPF',severity='5',mnemonic='ADJCHG')
Listener para el evento OSPF-4-ERRRCV	SELECT * FROM LogBean(facility='OSPF',severity='4',mnemonic='ERRRCV')
Listener para el evento CDP-4-NATIVE_VLAN_MISMATCH	SELECT * FROM LogBean(facility='CDP',severity='4',mnemonic='NATIVE_VLAN_MISMATCH')
Listener para el evento SFF8472-5-THRESHOLD_VIOLATION: RX	SELECT * FROM LogBean(facility='SFF8472',severity='5',mnemonic='THRESHOLD_VIOLATION')
Listener para el evento OSPF-4-CONFLICTING_LSAID	SELECT * FROM LogBean(facility='OSPF',severity='4',mnemonic='CONFLICTING_LSAID')
Listener para el evento PLATFORM-ASM-0-FAULT_RECOVERY_ERR	SELECT * FROM LogBean(facility='PLATFORMASM',severity='0',mnemonic='FAULT_RECOVERY_ERR')
listener para el evento PLATFORM-ASM-3-MALLOC_FAILED	SELECT * FROM LogBean(facility='PLATFORMASM',severity='3',mnemonic='MALLOC_FAILED')
Listener para el evento PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN	SELECT * FROM LogBean(facility='PLATFORMENVMON',severity='0',mnemonic='CARD_OVERTEMP_SHUTDOWN')
Listener para el evento PLATFORM-ENVMON-2-CHASSIS_OVERTEMP	SELECT * FROM LogBean(facility='PLATFORMENVMON',severity='2',mnemonic='CHASSIS_OVERTEMP')

Listener para el evento PLATFORM-ENVMON-2-FAN	SELECT * FROM LogBean(facility='PLATFORMENVMON',severity='2',mnemonic='FAN')
Listener para el evento PLATFORM-ENVMON-2-FAN_FAIL	SELECT * FROM LogBean(facility='PLATFORMENVMON',severity='2',mnemonic='FAN_FAIL')
Listener para el flapping de interface	SELECT * FROM LogBean.win:time(120) where facility='LINEPROTO' and severity='5' and mnemonic='UPDOWN' GROUP BY ipaddr HAVING count(*) > 4

a) El agente detecta el “flapping” de una interfaz de red, obtiene la dirección IP del equipo y el nombre de la interfaz de red que está teniendo “flapping”. Si la cantidad de veces que una interface “cae” y “levanta” es más de cuatro veces en 120 segundos, el softbot gestor de notificaciones apaga la interface afectada. La clase que maneja este comportamiento se denomina L2OSPF5AdjchgSubscriber. Esta clase realiza las siguientes actividades:

- Extrae del objeto “eventMap” (proporcionado por la librería CEP Espert) la dirección IP y la información de la interfaz de red que está en estado flapping.
- Utiliza expresiones regulares Java para extraer el nombre de la interface de red que está en estado “flapping”.
- Ejecuta el script “expect” llamado “/root/expect/shutdown.exp”, el cual se conecta en forma automática al enrutador y deshabilita administrativamente la interface en estado “flapping”. La figura 4.13 muestra parte del script “Expect” que apaga la interface en estado “flapping”.



```

root@bender: ~
GNU nano 3.2      shutdown.exp

send -- "cisco\r"
expect ">"
send -- "enable\r"
expect "*assword: "
send -- "cisco\r"
expect "#"
send -- "configure terminal\r"
expect "(config)#"
send -- "$Interface\r"
expect "(config-if)#"
send -- "shutdown\r"
expect "(config-if)#"
send -- "end\r"
expect "#"
send -- "logout\r"
expect eof

```

Figura 4.13.- Bloque de código que apaga la interface en flapping

- b) Configura y arranca los “susbcribers” o “suscriptores”. Los “susbcribers” capturan secuencias de eventos complejos definidos en el proceso de análisis del sistema. Mediante la librería “Esper”, se han creado en base a un lenguaje similar al “SQL” un conjunto de “susbcribers” para eventos secuenciales en el tiempo que cumplen las siguientes condiciones:

Tabla 4.2- Definición de los “subscribers”

Descripción del “susbcribers”	Definición del “susbcribers”
Activación de OSPF debido a cambio de protocolo capa 2 de interfaz de red	String expresion = "select * from LogBean " + "match_recognize (" + "partition by ipaddr " + " measures A as event1, B as event2 " + " pattern (A B) " + " define " + " A as A.facility = 'LINEPROTO' and A.severity = '5', " + " B as B.facility = 'OSPF' and B.severity = '5') ";
Activación de OSPF debido a cambio de enlace capa 1 de interface de red	String expresion = "select * from LogBean " + "match_recognize (" + " partition by ipaddr " + " measures A as event1, B as event2, C as event3 " + " pattern (A B C) "

	<pre> + " define " + " A as A.facility = 'LINK' and A.severity = '3'," + " B as B.facility = 'LINEPROTO' and B.severity = '5', " + " C as C.facility = 'OSPF' and C.severity = '5') "; </pre>
Fallo en OSPF debido a cambio de protocolo capa 2 de interfaz de red	<pre> String expresion = "select * from LogBean " + "match_recognize (" + " partition by ipaddr " + " measures A as event1, B as event2, C as event3 " + " pattern (A B C) " + " define " + " A as A.facility = 'LINK' and A.severity = '3', " + " B as B.facility = 'LINEPROTO' and B.severity = '5') "; </pre>

4.3.3. El “Agente Gráfico”

El “Agente Gráfico” se registra en el AMS de la plataforma de agentes JADE según se muestra en la figura 4.14. El nombre de registro para este agente es "Agente_Grafico". Este agente muestra las ventanas gráficas donde se muestra la información.

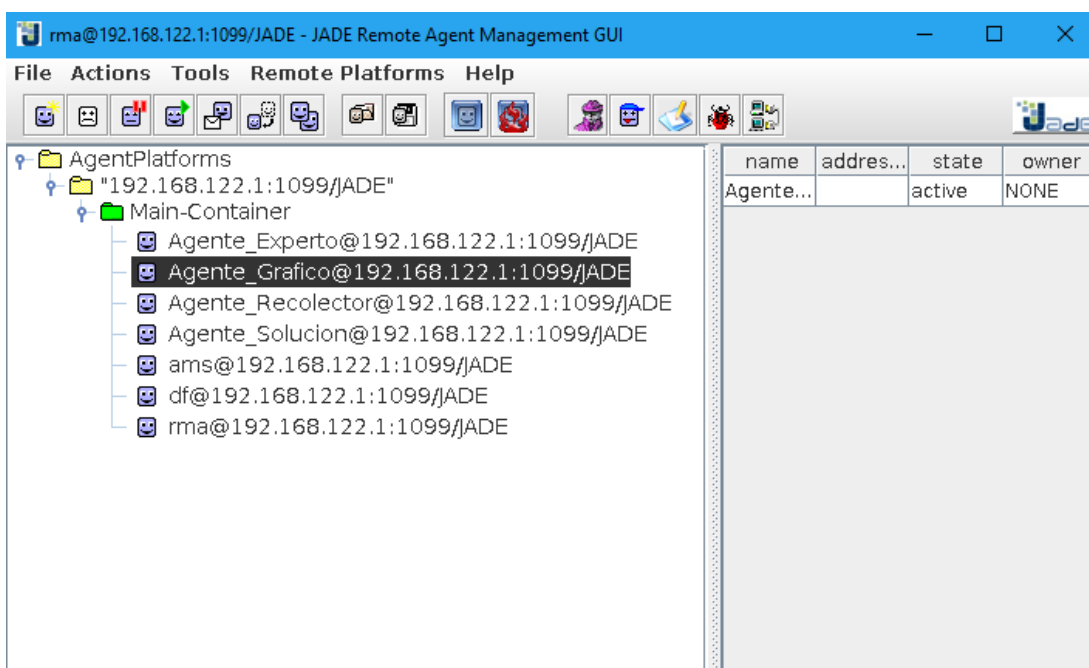


Figura 4.14.- Registro del “Agente_Grafico” en el framework Jade

4.4. Pruebas de hipótesis y presentación de los resultados

4.4.1. Detección y procesamiento de eventos simples a través de “listeners”

El softbot gestor de notificaciones dispone de un conjunto de “listeners” que están “escuchando” notificaciones de eventos provenientes de la red. Si alguno de los eventos recibidos cumple con los patrones establecidos en la base de datos de hechos, el softbot gestor de notificaciones alerta al operador del NOC a través de una ventana gráfica de eventos. Las pruebas de los “listeners” consisten en el envío de dos mil eventos (ruido) y un evento importante. El softbot gestor de notificaciones es capaz de identificar y extraer el evento importante del “ruido” y alertar al personal operador del NOC del evento sucedido y las acciones a realizar para solucionar el problema. Los eventos importantes se muestran en una ventana gráfica llamada “recomendaciones”. La figura 4.15 muestra una parte de los dos mil eventos de prueba enviados:

```
10.10.10.11:Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99) from EXSTART to DOWN, Neighbor Down: Inter-Interface down or detached
10.10.10.18:Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
10.10.10.21:Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
10.10.10.43:Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSION=%s
10.10.10.98:Sep 13 03:46:17.111: %CRYPTO-6-GDOI_ON_OFF: GDOI is OFF OFF
10.10.10.21:Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated10.10.17:Sep 13 03:46:14.335:
10.10.10.19:Sep 13 03:46:21.134: %AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.75:Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
10.10.10.34:Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
10.10.10.67:Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
10.10.10.91:Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
10.10.10.23:Sep 13 03:46:26.122: %C1700-EM-6-UNAVAILABLE: %s shutting down
10.10.10.11:Sep 22 15:46:26.067: %LINK-3-LINK_FAULT: Interface FastEthernet1/0
10.10.10.53:Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the limit for the platform.
10.10.10.64:Sep 13 03:46:28.642: %C7600-SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
10.10.10.90:Sep 13 03:46:29.729: %BUS-5-SENDIPCMRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code %d
10.10.10.78:Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s: All %sadjacencies cleared
10.10.10.17:Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
10.10.10.88:Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
10.10.10.45:Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
10.10.10.18:Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configuration to standby, so standby will be r
10.10.10.76:Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated10.10.17:Sep 13 03:46:14.335:
10.10.10.45:Sep 13 03:46:38.671: %AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.81:Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
10.10.10.11:Sep 13 03:46:41.903: %C1700-EM-6-UNAVAILABLE: %s shutting down
10.10.10.58:Sep 13 03:46:42.112: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the limit for the platform.
10.10.10.37:Sep 13 03:46:43.179: %C7600-SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
10.10.10.29:Sep 13 03:46:44.620: %C1700-EM-6-UNAVAILABLE: %s shutting down
10.10.10.76:Sep 13 03:46:46.167: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the limit for the platform.
10.10.10.71:Sep 13 03:46:47.972: %C7600-SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
```

Figura 4.15.- Eventos de prueba enviados al softbot gestor de notificaciones

4.4.1.1. Prueba de detección de eventos del tipo “LINK-3-UPDOWN”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una

notificación importante llamado “LINK-3-UPDOWN”, según se muestra en la figura 4.16.

```

10.10.10.11:Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTAR
10.10.10.18:Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
10.10.10.21:Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
10.10.10.43:Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s
10.10.10.98:Sep 13 03:46:17.111: %CRYPTO-6-GDOI_ON_OFF: GDOI is OFF OFF
10.10.10.21:Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 star
10.10.10.19:Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.75:Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing w
10.10.10.34:Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
10.10.10.67:Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access
10.10.10.91:Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
10.10.10.23:Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.43:Sep 13 03:46:26.124: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
10.10.10.53:Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata
10.10.10.64:Sep 13 03:46:28.642: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec]
10.10.10.90:Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x,
10.10.10.78:Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s: All %sadjacencies cleared
10.10.10.17:Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
10.10.10.88:Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
10.10.10.45:Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
10.10.10.18:Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the ru
10.10.10.76:Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 star
10.10.10.45:Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.81:Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing w
10.10.10.11:Sep 13 03:46:41.903: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.58:Sep 13 03:46:42.112: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata
10.10.10.37:Sep 13 03:46:43.179: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec]
10.10.10.29:Sep 13 03:46:44.620: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.76:Sep 13 03:46:46.167: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata
10.10.10.71:Sep 13 03:46:47.972: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec]

```

Figura 4.16.- La notificación “LINK-3-UPDOWN”

Tal como muestra la figura 4.17, el softbot gestor de notificaciones extrae la notificación “LINK-3-UPDOWN”, busca en su base de datos la recomendación correspondiente al evento “LINK-3-UPDOWN” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

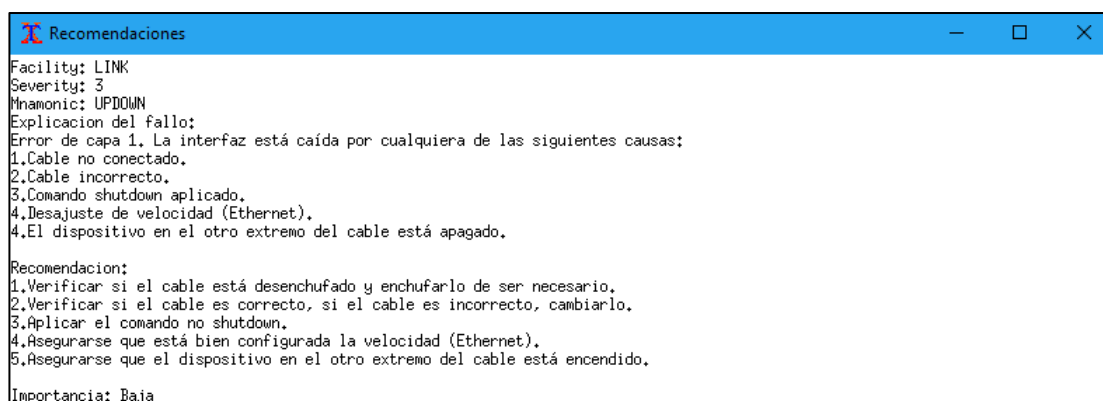


Figura 4.17.- Diagnóstico y recomendación al detectar la notificación “LINK-3-UPDOWN”

4.4.1.2. Prueba de detección de eventos del tipo “OSPF-4-ERRRCV”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “OSPF-4-ERRRCV”, según se muestra en la figura 4.18.

```

10.10.10.11:Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN
10.10.10.18:Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
10.10.10.21:Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
10.10.10.43:Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s
10.10.10.98:Sep 13 03:46:17.111: %CRYPTO-6-GDOI_ON_OFF: GDOI is OFF OFF
10.10.10.21:Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI
10.10.10.19:Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.11:Sep 13 03:46:21.137: %LINK-3-TOOSMALL: Interface POS3/0, Output runt packet of 0 bytes
10.10.10.75:Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
10.10.10.34:Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
10.10.10.67:Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control
10.10.10.91:Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
10.10.10.23:Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.43:Sep 13 03:46:26.124: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
10.10.10.53:Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows ex
10.10.10.64:Sep 13 03:46:28.642: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
10.10.10.90:Sep 13 03:46:28.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must
10.10.10.90:Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d,
10.10.10.78:Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s: All %sadjacencies cleared
10.10.10.17:Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
10.10.10.88:Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
10.10.10.45:Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
10.10.10.18:Sep 13 03:46:26.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running cor
10.10.10.76:Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI
10.10.10.45:Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.81:Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
10.10.10.11:Sep 13 03:46:41.903: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.58:Sep 13 03:46:42.112: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows ex
10.10.10.37:Sep 13 03:46:43.179: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
10.10.10.29:Sep 13 03:46:44.620: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.76:Sep 13 03:46:46.167: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows ex
10.10.10.71:Sep 13 03:46:47.972: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.

```

Figura 4.18.- La notificación “OSPF-4-ERRRCV”

Tal como muestra la figura 4.19, el softbot gestor de notificaciones extrae la notificación “OSPF-4-ERRRCV”, busca en su base de datos la recomendación correspondiente al evento “OSPF-4-ERRRCV” y muestra a través de una ventana gráfica, La información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

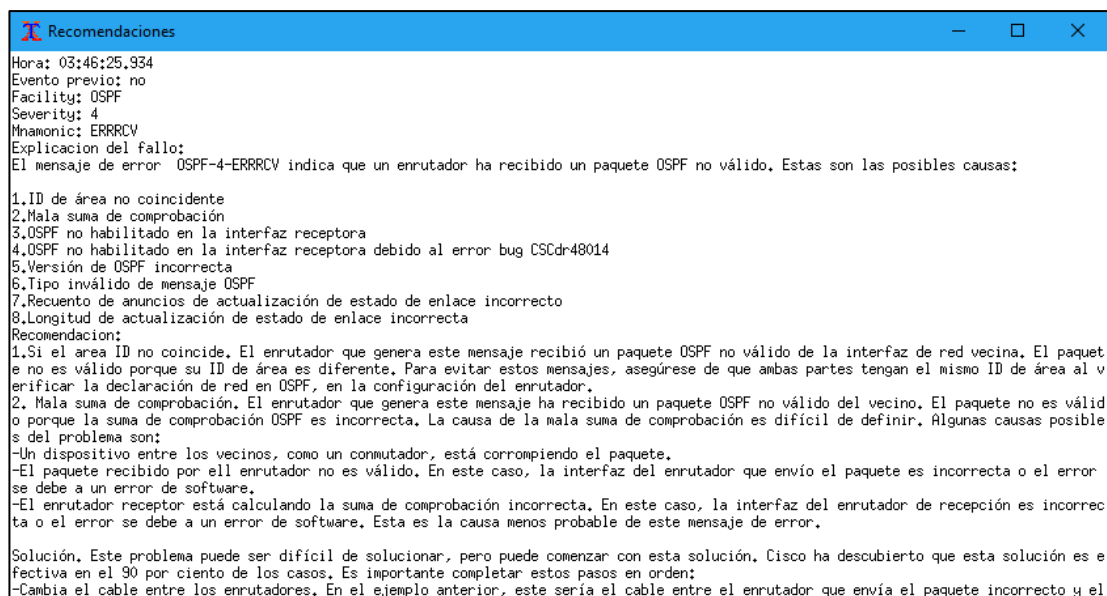


Figura 4.19.- Diagnóstico y recomendación al detectar la notificación “OSPF-4-ERRRCV”

4.4.1.3. Prueba de detección de eventos del tipo “CDP-4-NATIVE VLAN MISMATCH”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “CDP-4-NATIVE_VLAN_MISMATCH”, según se muestra en la figura 4.20.

```

10.10.10.11:Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN,
10.10.10.18:Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
10.10.10.21:Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
10.10.10.43:Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s
10.10.10.98:Sep 13 03:46:17.111: %CRYPTO-6-GDOI_ON_OFF: GDOI is OFF OFF
10.10.10.21:Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI in
10.10.10.19:Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.11:Sep 13 03:46:21.137: %LINK-3-TOOSMALL: Interface POS3/0, Output runt packet of 0 bytes
10.10.10.75:Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
10.10.10.34:Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
10.10.10.67:Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control lis
10.10.10.91:Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
10.10.10.23:Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.43:Sep 13 03:46:26.124: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
10.10.10.53:Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exce
10.10.10.64:Sep 13 03:46:28.642: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
10.10.10.90:Sep 13 03:46:28.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be
10.10.10.90:Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cm
10.10.10.78:Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
10.10.10.17:Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
10.10.10.88:Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
10.10.10.45:Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
10.10.10.45:Sep 13 03:46:35.657: %CDP-4-NATIVE_VLAN_MISMATCH: Native VLAN mismatch discovered on FastEthernet0/11 (30)
10.10.10.18:Sep 13 03:46:26.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running confi
10.10.10.76:Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI in
10.10.10.45:Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.81:Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
10.10.10.11:Sep 13 03:46:41.903: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.58:Sep 13 03:46:42.112: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exce
10.10.10.37:Sep 13 03:46:43.179: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
10.10.10.29:Sep 13 03:46:44.620: %C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.76:Sep 13 03:46:46.167: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exce

```

Figura 4.20.- La notificación “CDP-4-NATIVE_VLAN_MISMATCH”

Tal como muestra la figura 4.21, el softbot gestor de notificaciones extrae la notificación “CDP-4-NATIVE_VLAN_MISMATCH”, busca en su base de datos la recomendación correspondiente al evento “CDP-4-NATIVE_VLAN_MISMATCH” y muestra a través de una ventana gráfica, La información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

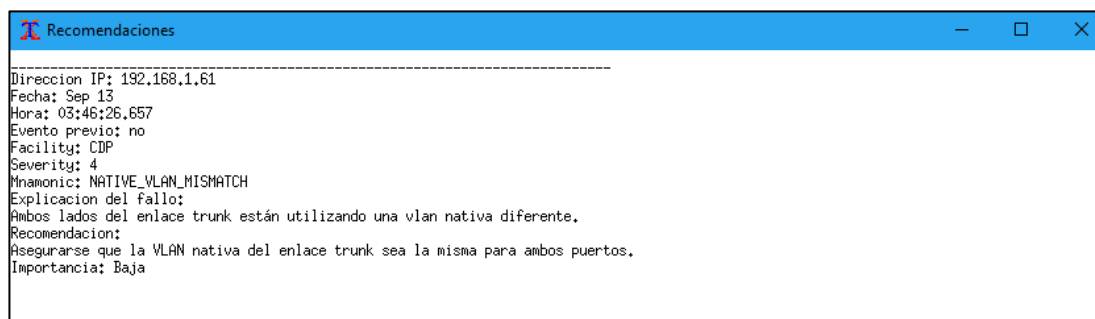


Figura 4.21.- Diagnóstico y recomendación al detectar la notificación “CDP-4-NATIVE_VLAN_MISMATCH”

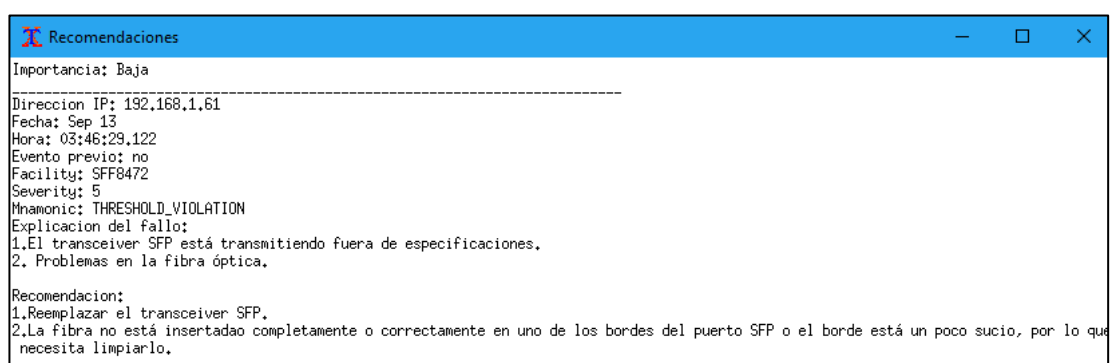
4.4.1.4. Prueba de detección de eventos del tipo “SFF8472-5-THRESHOLD VIOLATION”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “SFF8472-5-THRESHOLD_VIOLATION”, según se muestra en la figura 4.22.

```
*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Neighbor
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111: %CR
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be virtua
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.122: %SFF8472-5-THRESHOLD_VIOLATION: Tel1/1: Rx power low warning; Operating value: -13.8 dBm, Thr
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded th
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configurati
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
```

Figura 4.22.- La notificación “SFF8472-5-THRESHOLD_VIOLATION”

Tal como muestra la figura 4.23, el softbot gestor de notificaciones extrae la notificación “SFF8472-5-THRESHOLD_VIOLATION”, busca en su base de datos la recomendación correspondiente al evento “SFF8472-5-THRESHOLD_VIOLATION” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.



Recomendaciones

Importancia: Baja

Direccion IP: 192.168.1.61
 Fecha: Sep 13
 Hora: 03:46:29.122
 Evento previo: no
 Facility: SFF8472
 Severity: 5
 Mnemonic: THRESHOLD_VIOLATION
 Explicacion del fallo:
 1.El transceiver SFP está transmitiendo fuera de especificaciones.
 2. Problemas en la fibra óptica.

Recomendacion:
 1.Reemplazar el transceiver SFP.
 2.La fibra no está insertadao completamente o correctamente en uno de los bordes del puerto SFP o el borde está un poco sucio, por lo que necesita limpiarlo.

Figura 4.23.- Diagnóstico y recomendación al detectar la notificación
“SFF8472-5-THRESHOLD_VIOLATION”

4.4.1.5. Prueba de detección de eventos del tipo “OSPF-4-CONFLICTING_LSAID”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “OSPF-4-CONFLICTING_LSAID”, según se muestra en la figura 4.24.

```
*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Neighbor Down: Inter
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111: %CRYPTO-6-GDOI_
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be virtual-link but n
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %OSPF-4-CONFLICTING_LSAID LSA origination prevented by existing LSA with same LSID but a different mask
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code %d
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the limit for
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configuration to standby
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
```

Figura 4.24.- La notificación “OSPF-4-CONFLICTING_LSAID”

Tal como muestra la figura 4.25, el softbot gestor de notificaciones extrae la notificación “OSPF-4-CONFLICTING_LSAID”, busca en su base de datos la recomendación correspondiente al evento “OSPF-4-CONFLICTING_LSAID” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

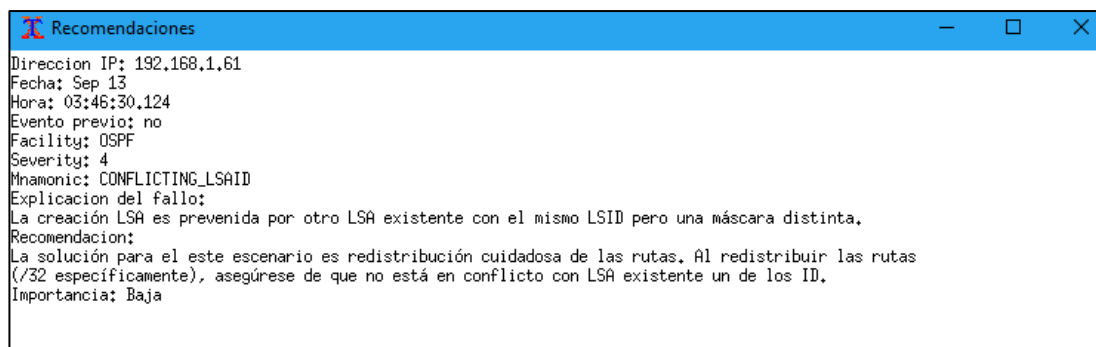


Figura 4.25.- Diagnóstico y recomendación al detectar la notificación “OSPF-4-CONFLICTING_LSAID”

4.4.1.6. Prueba de detección de eventos del tipo “PLATFORMASM-0-FAULT_RECOVERY_ERR”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “PLATFORMASM-0-FAULT_RECOVERY_ERR”, según se muestra en la figura 4.26.

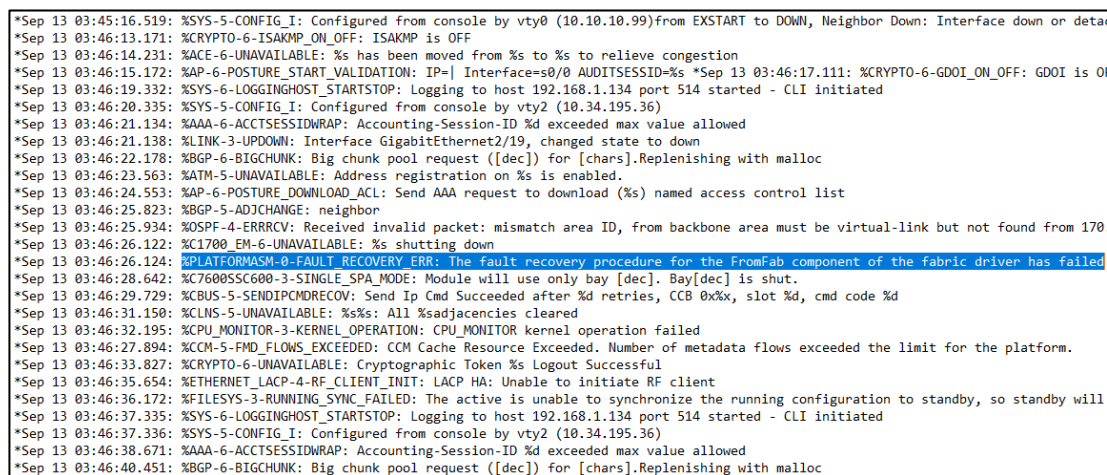


Figura 4.26.- La notificación “PLATFORMASM-0-FAULT_RECOVERY_ERR”

Tal como muestra la figura 4.27, el softbot gestor de notificaciones extrae la notificación “PLATFORMASM-0-FAULT_RECOVERY_ERR”, busca en su base de datos la recomendación correspondiente al evento “PLATFORMASM-0-FAULT_RECOVERY_ERR” y muestra a través de una ventana

gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

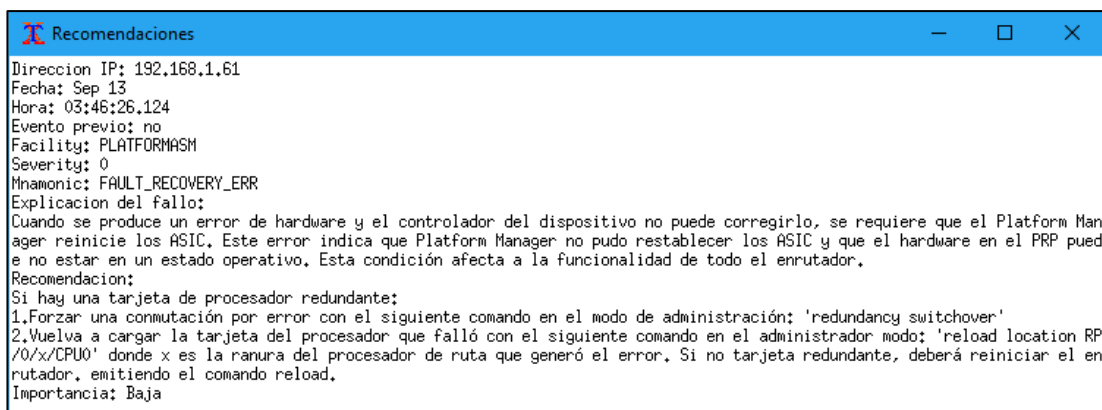


Figura 4.27.- Diagnóstico y recomendación al detectar la notificación “OSPF-4-CONFLICTING_LSAD”

4.4.1.7. Prueba de detección de eventos del tipo “PLATFORM-ASM-3-MALLOC FAILED”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “PLATFORM-ASM-3-MALLOC_FAILED”, según se muestra en la figura 4.28.

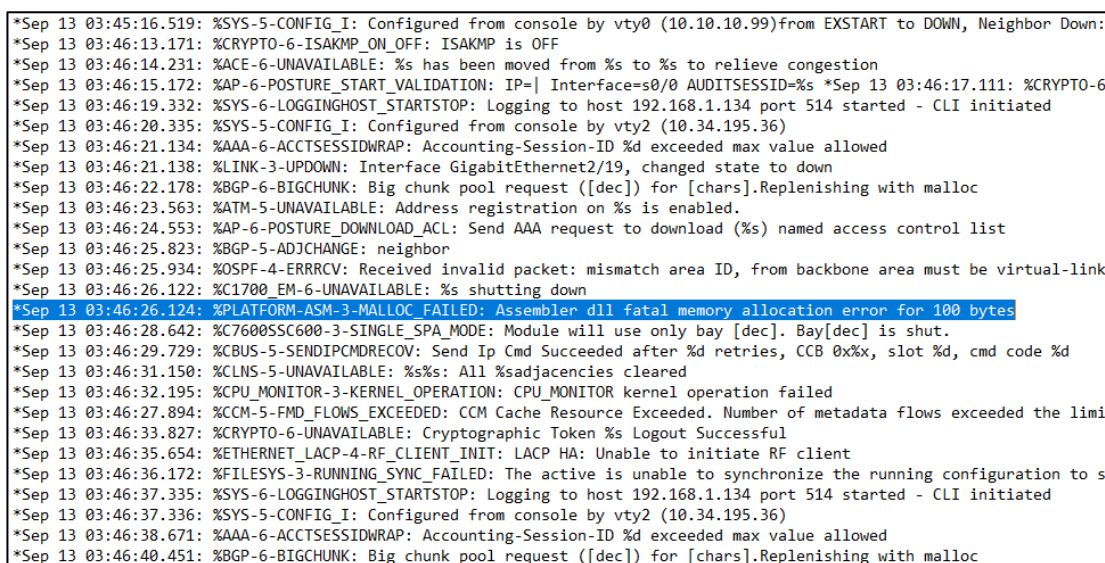


Figura 4.28.- La notificación “PLATFORM-ASM-3-MALLOC_FAILED”

Tal como muestra la figura 4.29, el softbot gestor de notificaciones extrae la notificación “PLATFORM-ASM-3-MALLOC_FAILED”, busca en su base de datos la recomendación correspondiente al evento “PLATFORM-ASM-3-MALLOC_FAILED” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

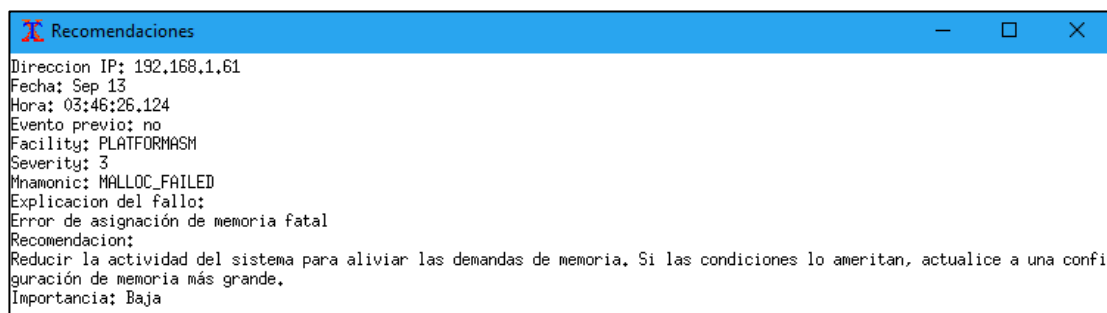


Figura 4.29.- Diagnóstico y recomendación al detectar la notificación
“PLATFORM-ASM-3-MALLOC_FAILED”

4.4.1.8. Prueba de detección de eventos del tipo “PLATFORMENVMON-0-CARD OVERTEMP SHUTDOWN”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN”, según se muestra en la figura 4.30.

```

*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Nei
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI init
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be v
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN: Slot 1/0/1 temperature 56C at or above 55C
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceed
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configu
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI init
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc

```

Figura 4.30.- La notificación “PLATFORMENVMON-0-CARD OVERTEMP SHUTDOWN”

Tal como muestra la figura 4.31, el softbot gestor de notificaciones extrae la notificación “PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN”, busca en su base de datos la recomendación correspondiente al evento “PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

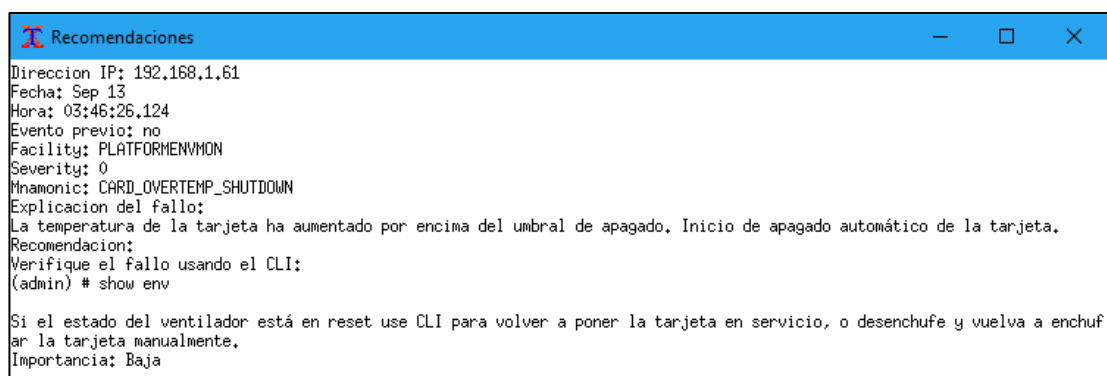


Figura 4.31.- Diagnóstico y recomendación al detectar la notificación “PLATFORMENVMON-0-CARD_OVERTEMP_SHUTDOWN”

4.4.1.9. Prueba de detección de eventos del tipo “PLATFORMENVMON-2 CHASSIS OVERTEMP”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “PLATFORMENVMON-2-CHASSIS_OVERTEMP”, según se muestra en la figura 4.32.

```
*Sep 13 03:45:16.519: %SYS-5-CONFIG I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Neighbor Down: Inter
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111: %CRYPTO-6-GDOI
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:20.335: %SYS-5-CONFIG I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be virtual-link but r
*Sep 13 03:46:26.122: %C1700 EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %PLATFORMENVMON-2-CHASSIS_OVERTEMP: Chassis ambient temperature 58C at or above rising T3 threshold 56C
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut.
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDCRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code %d
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the limit for
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configuration to standby
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:37.336: %SYS-5-CONFIG I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
```

Figura 4.32.- La notificación “PLATFORMENVMON-2-CHASSIS_OVERTEMP”

Tal como muestra la figura 4.33, el softbot gestor de notificaciones extrae la notificación “PLATFORMENVMON-2-CHASSIS_OVERTEMP”, busca en su base de datos la recomendación correspondiente al evento “PLATFORMENVMON-2-CHASSIS_OVERTEMP” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

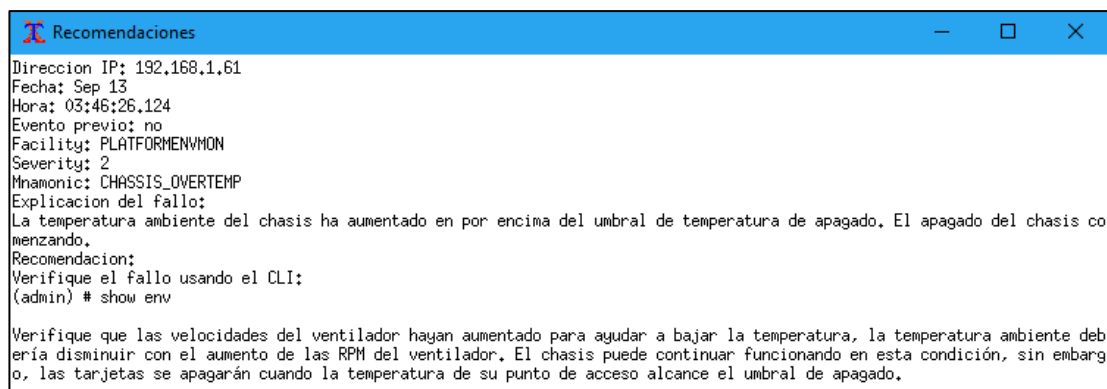


Figura 4.33.- Diagnóstico y recomendación al detectar la notificación
 “PLATFORMENVMON-2-CHASSIS_OVERTEMP”

4.4.1.10. Prueba de detección de eventos del tipo “PLATFORM-ENVMON-2-FAN FAIL”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “PLATFORM-ENVMON-2-FAN_FAIL”, según se muestra en la figura 4.34.

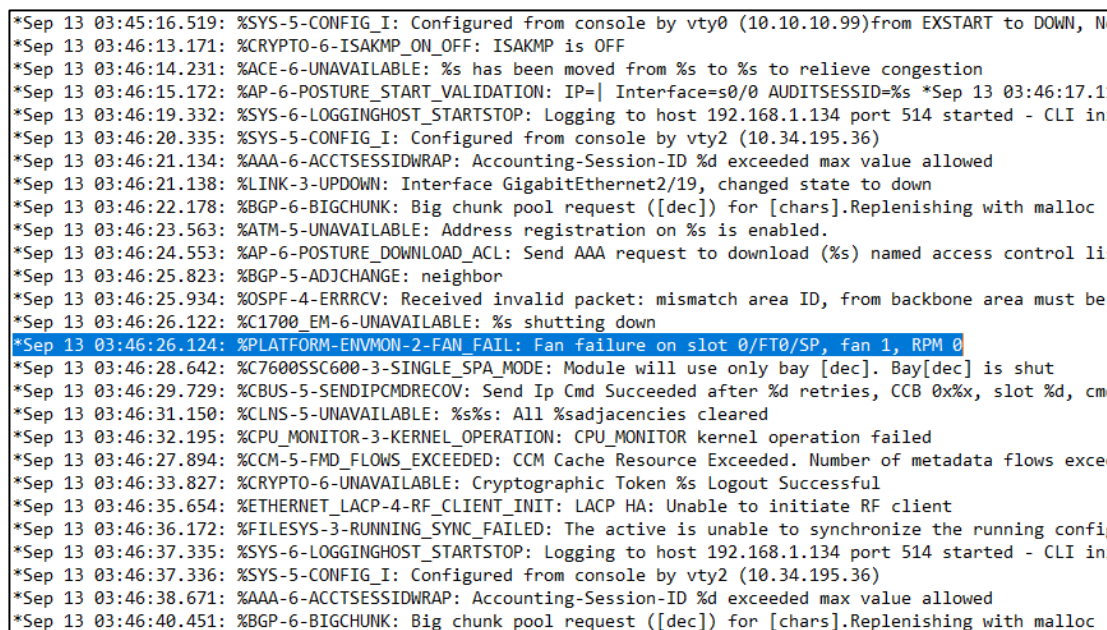


Figura 4.34.- La notificación “PLATFORM-ENVMON-2-FAN_FAIL”

Tal como muestra la figura 4.35, el softbot gestor de notificaciones extrae la notificación “PLATFORM-ENVMON-2-

FAN_FAIL”, busca en su base de datos la recomendación correspondiente al evento “PLATFORM-ENVMON-2-FAN_FAIL” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

```

Direccion IP: 192.168.1.61
Fecha: Sep 13
Hora: 03:46:26.124
Evento previo: no
Facility: PLATFORMENVMON
Severity: 2
Mnemonic: FAN_FAIL
Explicacion del fallo:
Fallo en arreglo de ventiladores.
Recomendacion:
Verifique la falla, use el CLI:

(admin) # show env

Verifique la salida del ventilador Si la falla del ventilador no se soluciona después de 30 minutos,, llame a su representante de soporte técnico para la reparación o reemplazo de la bandeja de ventiladores.

```

Figura 4.35.- Diagnóstico y recomendación al detectar la notificación “PLATFORM-ENVMON-2-FAN_FAIL”

4.4.1.11. Prueba de detección de eventos del tipo “PLATFORM-ENVMON-2”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “PLATFORM-ENVMON-2-FAN”, según se muestra en la figura 4.36.

```

*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Neighbor Down
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111: %CRYPTO
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be virtual-l
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %PLATFORM-ENVMON-2-FAN: The fan array 1 has reached max level
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code %d
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:32.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the l
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configuration to
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc

```

Figura 4.36.- La notificación “PLATFORM-ENVMON-2-FAN”

Tal como muestra la figura 4.37, el softbot gestor de notificaciones extrae la notificación “PLATFORM-ENVMON-2-FAN”, busca en su base de datos la recomendación correspondiente al evento “PLATFORM-ENVMON-2-FAN” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

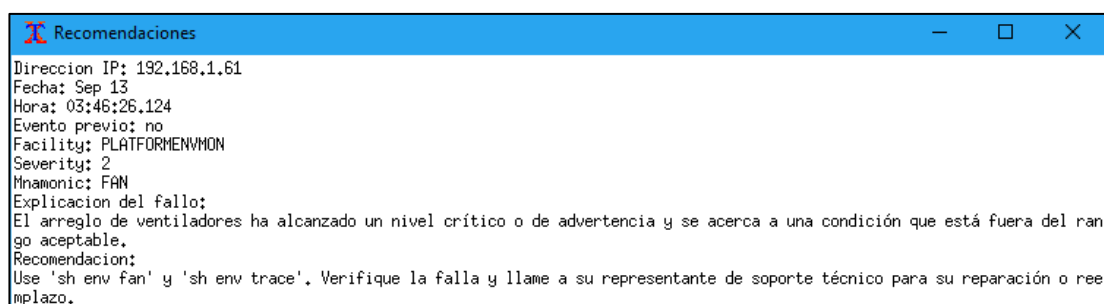


Figura 4.37.- Diagnóstico y recomendación al detectar la notificación “PLATFORM-ENVMON-2-FAN”

4.4.1.12. Prueba de detección de eventos del tipo “OSPF-5-ADJCHG”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “OSPF-5-ADJCHG”, según se muestra en la figura 4.38.

```

*Sep 13 03:45:16.519: %SYS-5-CONFIG I: Configured from console by vty0 (10.10.10.99) from EXSTART to DOWN, Neighbor Down:
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111: %CRYPTO-6
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:20.335: %SYS-5-CONFIG I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars]. Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be virtual-link
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %OSPF-5-ADJCHG: Process 1, Nbr 10.0.0.1 on GigabitEthernet5/15 from LOADING to FULL, Loading Done
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code %d
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the limit
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configuration to s
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:37.336: %SYS-5-CONFIG I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars]. Replenishing with malloc

```

Figura 4.38.- La notificación “OSPF-5-ADJCHG”

Tal como muestra la figura 4.39, el softbot gestor de notificaciones extrae la notificación “OSPF-5-ADJCHG”, busca en su base de datos la recomendación correspondiente al evento “OSPF-5-ADJCHG” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

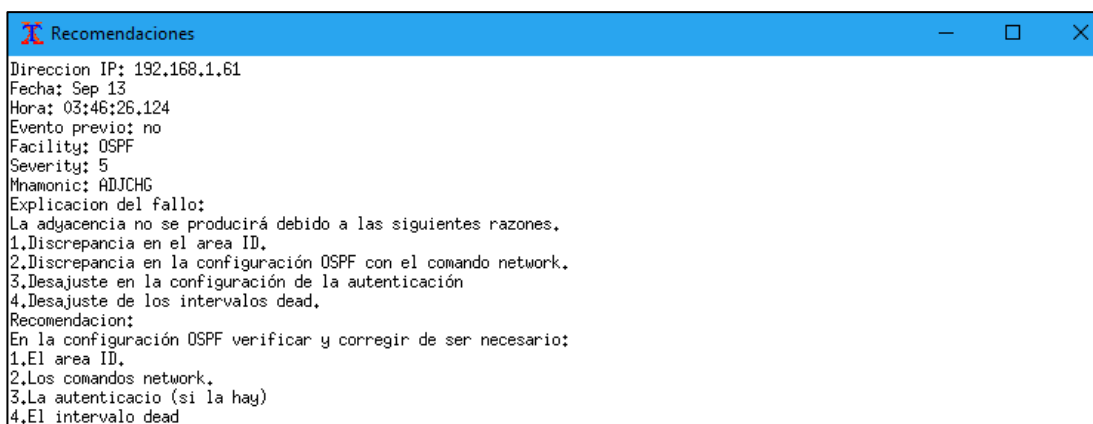


Figura 4.39.- Diagnóstico y recomendación al detectar la notificación “OSPF-5-ADJCHG”

4.4.1.13. Prueba de detección de eventos del tipo “LINK-3-UPDOWN”

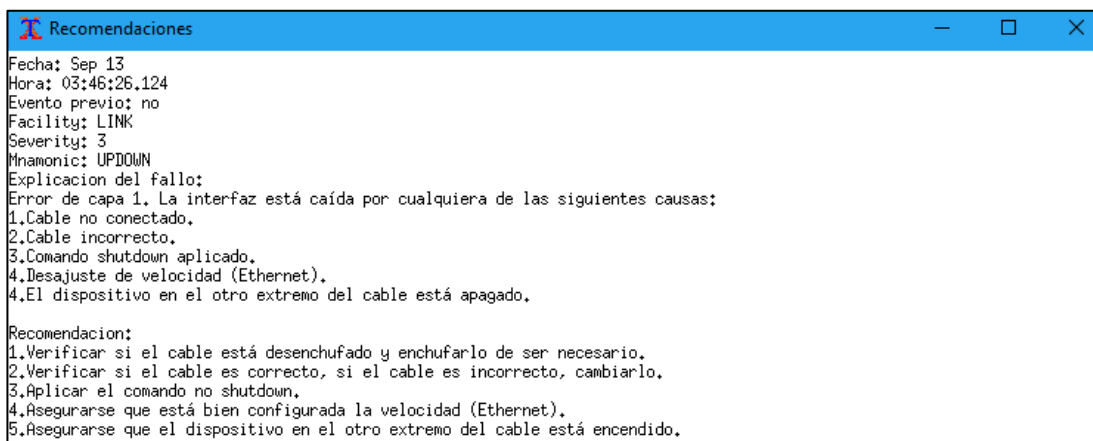
Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una

notificación importante llamado “LINK-3-UPDOWN”, según se muestra en la figura 4.40.

```
*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Nei
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSIONID=%s *Sep 13 03:46:17.111
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI init
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be v
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceed
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configu
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI init
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
```

Figura 4.40.- La notificación “LINK-3-UPDOWN”

Tal como muestra la figura 4.41, el softbot gestor de notificaciones extrae la notificación “LINK-3-UPDOWN”, busca en su base de datos la recomendación correspondiente al evento “LINK-3-UPDOWN” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.



Recomendaciones

Fecha: Sep 13
 Hora: 03:46:26.124
 Evento previo: no
 Facility: LINK
 Severity: 3
 Mnemonic: UPDOWN
 Explicacion del fallo:
 Error de capa 1. La interfaz está caída por cualquiera de las siguientes causas:
 1.Cable no conectado.
 2.Cable incorrecto.
 3.Comando shutdown aplicado.
 4.Desajuste de velocidad (Ethernet).
 4.El dispositivo en el otro extremo del cable está apagado.

Recomendacion:
 1.Verificar si el cable está desenchufado y enchufarlo de ser necesario.
 2.Verificar si el cable es correcto, si el cable es incorrecto, cambiarlo.
 3.Aplicar el comando no shutdown.
 4.Asegurarse que está bien configurada la velocidad (Ethernet).
 5.Asegurarse que el dispositivo en el otro extremo del cable está encendido.

Figura 4.41.- Diagnóstico y recomendación al detectar la notificación “LINK-3-UPDOWN”

4.4.1.14. Prueba de detección de eventos del tipo “LINEPROTO-5-UPDOWN”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “LINEPROTO-5-UPDOWN”, según se muestra en la figura 4.42.

```
*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99) from EXSTART to DOWN, Neighbor
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111: %C
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiate
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars]. Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be virtu
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/28, changed state to down
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded t
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configurati
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiate
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars]. Replenishing with malloc
```

Figura 4.42.- La notificación “LINEPROTO-5-UPDOWN”

Tal como muestra la figura 4.43, el softbot gestor de notificaciones extrae la notificación “LINEPROTO-5-UPDOWN”, busca en su base de datos la recomendación correspondiente al evento “LINEPROTO-5-UPDOWN” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

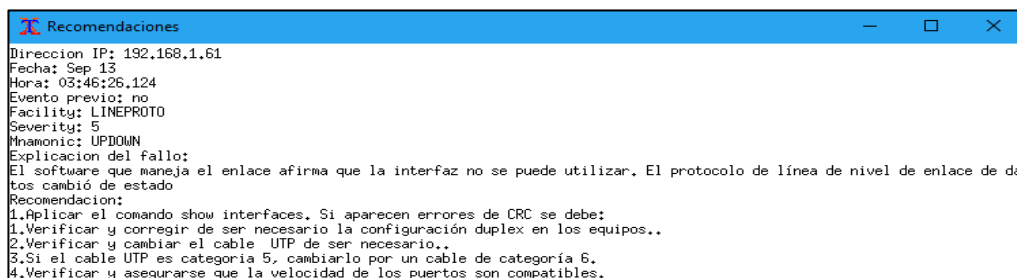


Figura 4.43.- Diagnóstico y recomendación al detectar la notificación “LINEPROTO-5-UPDOWN”

4.4.1.15. Prueba de detección de eventos del tipo “LINK-3-LINK_FAULT”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “LINK-3-LINK_FAULT”, según se muestra en la figura 4.44.

```
*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Neighbor U
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.111: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be virtual
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %LINK-3-LINK_FAULT: Lost signal clock Interface Serial0/14
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd code %d
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceeded the
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configuration
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI initiated
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
```

Figura 4.44.- La notificación “LINK-3-LINK_FAULT”

Tal como muestra la figura 4.45, el softbot gestor de notificaciones extrae la notificación “LINK-3-LINK_FAULT”, busca en su base de datos la recomendación correspondiente al evento “LINK-3-LINK_FAULT” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

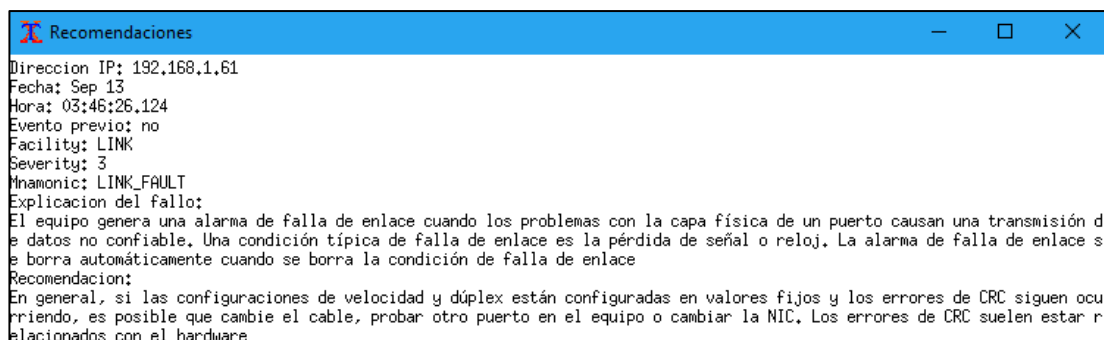


Figura 4.45.- Diagnóstico y recomendación al detectar la notificación “LINK-3-LINK_FAULT”

4.4.1.16. Prueba de detección de eventos del tipo “LINK-3-FCS_ERROR”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “LINK-3-FCS_ERROR”, según se muestra en la figura 4.46.

```
*Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DOWN, Ne
*Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
*Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
*Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSID=%s *Sep 13 03:46:17.11
*Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI init
*Sep 13 03:46:20.335: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:21.138: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
*Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
*Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s is enabled.
*Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access control list
*Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor
*Sep 13 03:46:25.934: %OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must be v
*Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down
*Sep 13 03:46:26.124: %LINK-3-FCS_ERROR: FCS error frames Interface Serial0/14
*Sep 13 03:46:28.642: %C7600SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
*Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d, cmd
*Sep 13 03:46:31.150: %CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
*Sep 13 03:46:32.195: %CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
*Sep 13 03:46:27.894: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows exceed
*Sep 13 03:46:33.827: %CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
*Sep 13 03:46:35.654: %ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
*Sep 13 03:46:36.172: %FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running configu
*Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - CLI init
*Sep 13 03:46:37.336: %SYS-5-CONFIG_I: Configured from console by vty2 (10.34.195.36)
*Sep 13 03:46:38.671: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %d exceeded max value allowed
*Sep 13 03:46:40.451: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with malloc
```

Figura 4.46.- La notificación “LINK-3-FCS_ERROR”

Tal como muestra la figura 4.47, el softbot gestor de notificaciones extrae la notificación “LINK-3-FCS_ERROR”, busca en su base de datos la recomendación correspondiente al

evento “LINK-3-FCS_ERROR” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

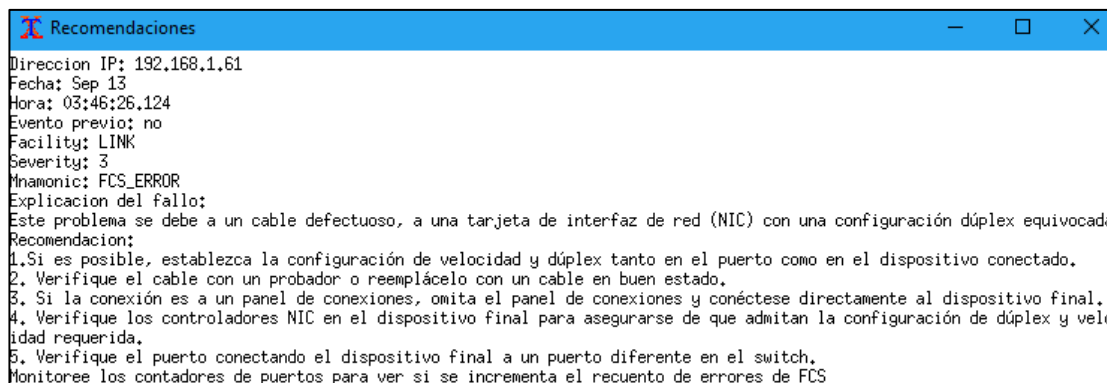


Figura 4.47.- Diagnóstico y recomendación al detectar la notificación “LINK-3-FCS_ERROR”

4.4.1.17. Prueba de detección de eventos del tipo “LINK-3-TOOSMALL”

Se envían al softbot gestor de notificaciones dos mil notificaciones de prueba y entre ellos se ha insertado una notificación importante llamado “LINK-3-TOOSMALL”, según se muestra en la figura 4.48.

```

10.10.10.11:Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)from EXSTART to DO
10.10.10.18:Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFF
10.10.10.21:Sep 13 03:46:14.231:%ACE-6-UNAVAILABLE: %s has been moved from %s to %s to relieve congestion
10.10.10.43:Sep 13 03:46:15.172:%AP-6-POSTURE_START_VALIDATION: IP=| Interface=s0/0 AUDITSESSIONID=%s
10.10.10.98:Sep 13 03:46:17.111: %CRYPTO-6-GDOI_ON_OFF: GDOI is OFF OFF
10.10.10.21:Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - C
10.10.10.19:Sep 13 03:46:21.134:%AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.11:Sep 13 03:46:21.137:%LINK-3-TOOSMALL: Interface POS3/0, Output runt packet of 0 bytes
10.10.10.75:Sep 13 03:46:22.178:%BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with mal
10.10.10.34:Sep 13 03:46:23.563:%ATM-5-UNAVAILABLE: Address registration on %s is enabled.
10.10.10.67:Sep 13 03:46:24.553:%AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request to download (%s) named access contro
10.10.10.91:Sep 13 03:46:25.823:%BGP-5-ADJCHANGE: neighbor
10.10.10.23:Sep 13 03:46:26.122:%C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.43:Sep 13 03:46:26.124: %LINK-3-UPDOWN: Interface GigabitEthernet2/19, changed state to down
10.10.10.53:Sep 13 03:46:27.894:%CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows
10.10.10.64:Sep 13 03:46:28.642:%C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
10.10.10.90:Sep 13 03:46:29.729:%CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded after %d retries, CCB 0x%x, slot %d
10.10.10.78:Sep 13 03:46:31.150:%CLNS-5-UNAVAILABLE: %s%s: All %sadjacencies cleared
10.10.10.17:Sep 13 03:46:32.195:%CPU_MONITOR-3-KERNEL_OPERATION: CPU_MONITOR kernel operation failed
10.10.10.88:Sep 13 03:46:33.827:%CRYPTO-6-UNAVAILABLE: Cryptographic Token %s Logout Successful
10.10.10.45:Sep 13 03:46:35.654:%ETHERNET_LACP-4-RF_CLIENT_INIT: LACP HA: Unable to initiate RF client
10.10.10.18:Sep 13 03:46:26.172:%FILESYS-3-RUNNING_SYNC_FAILED: The active is unable to synchronize the running c
10.10.10.76:Sep 13 03:46:37.335: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.1.134 port 514 started - C
10.10.10.45:Sep 13 03:46:38.671:%AAA-6-ACCTSESSIONWRAP: Accounting-Session-ID %d exceeded max value allowed
10.10.10.81:Sep 13 03:46:40.451:%BGP-6-BIGCHUNK: Big chunk pool request ([dec]) for [chars].Replenishing with mal
10.10.10.11:Sep 13 03:46:41.903:%C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.58:Sep 13 03:46:42.112:%CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows
10.10.10.37:Sep 13 03:46:43.179:%C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut
10.10.10.29:Sep 13 03:46:44.620:%C1700_EM-6-UNAVAILABLE: %s shutting down
10.10.10.76:Sep 13 03:46:46.167:%CCM-5-FMD_FLOWS_EXCEEDED: CCM Cache Resource Exceeded. Number of metadata flows
10.10.10.71:Sep 13 03:46:47.972:%C7600_SSC600-3-SINGLE_SPA_MODE: Module will use only bay [dec]. Bay[dec] is shut

```

Figura 4.48.- La notificación “LINK-3-TOOSMALL”

Tal como muestra la figura 4.49, el softbot gestor de notificaciones extrae la notificación “LINK-3-TOOSMALL”, busca en su base de datos la recomendación correspondiente al evento “LINK-3-TOOSMALL” y muestra a través de una ventana gráfica, la información de la notificación, la explicación del fallo y la recomendación para solucionar dicho fallo.

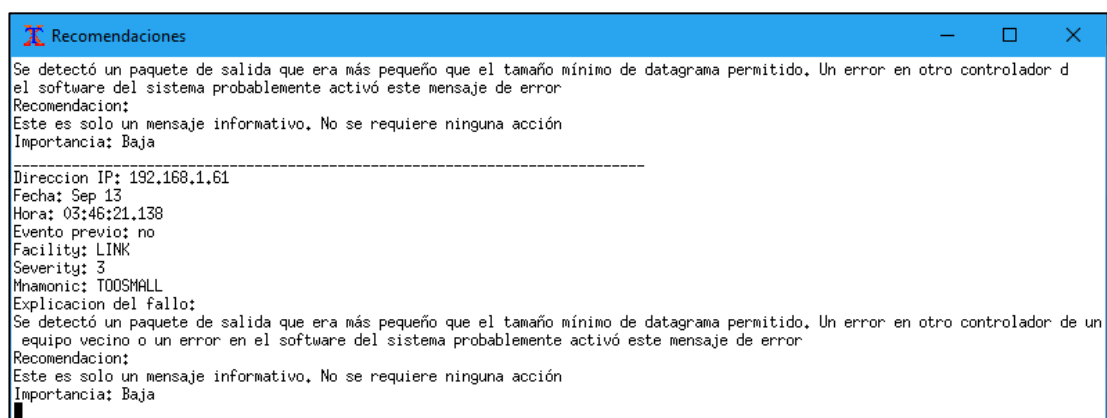


Figura 4.49.- Diagnóstico y recomendación al detectar la notificación “LINK-3-TOOSMALL”

4.4.2. Detección y procesamiento de eventos correlacionados a través de “susscribers”

4.4.2.1. Introducción

El softbot gestor de notificaciones multiagente dispone de un conjunto de clases en JAVA de la librería Esper llamadas “susscribers” que están “escuchando” notificaciones de eventos relacionados provenientes de la red. Si alguna secuencia de eventos recibidos en el tiempo cumple con los patrones establecidos en la base de datos de hechos, el softbot gestor de notificaciones alerta al operador del NOC a través de una ventana gráfica de eventos. Las pruebas de los “susscribers” consisten en emplear una red virtualizada GNS3 de enrutadores Cisco 7200, y mediante comandos se deshabilitan administrativamente interfaces. Las interfaces deshabilitadas generan un conjunto de eventos en el tiempo, que incluyen eventos a nivel de capa física, de enlace y de red. El softbot gestor de notificaciones multiagente es capaz de realizar correlación de eventos en el tiempo e identifica y extrae patrones importantes de eventos del “ruido” y alerta al personal operador del NOC de lo sucedido, indicando las acciones a realizar para solucionar el problema. Los eventos importantes se muestran en una ventana gráfica llamada “recomendaciones” y las acciones tomadas se muestran en una ventana gráfica llamada “acciones”.

4.4.2.2. Prueba de “flapping”

La prueba consiste generar intermitencia o “flapping” en la red (anuncios continuos de cambios de estado de enlace OSPF) manualmente, es decir, en apagar y encender la interface fastethernet1/0 del enrutador “AREQUIPA” más de cuatro veces seguidas en un minuto. Para ello se siguen los siguientes pasos:

a) Acceso a enrutador “Arequipa” a través de terminal remoto: Mediante consola SSH se accede al enrutador “Arequipa”, tal como muestra la figura 4.50.

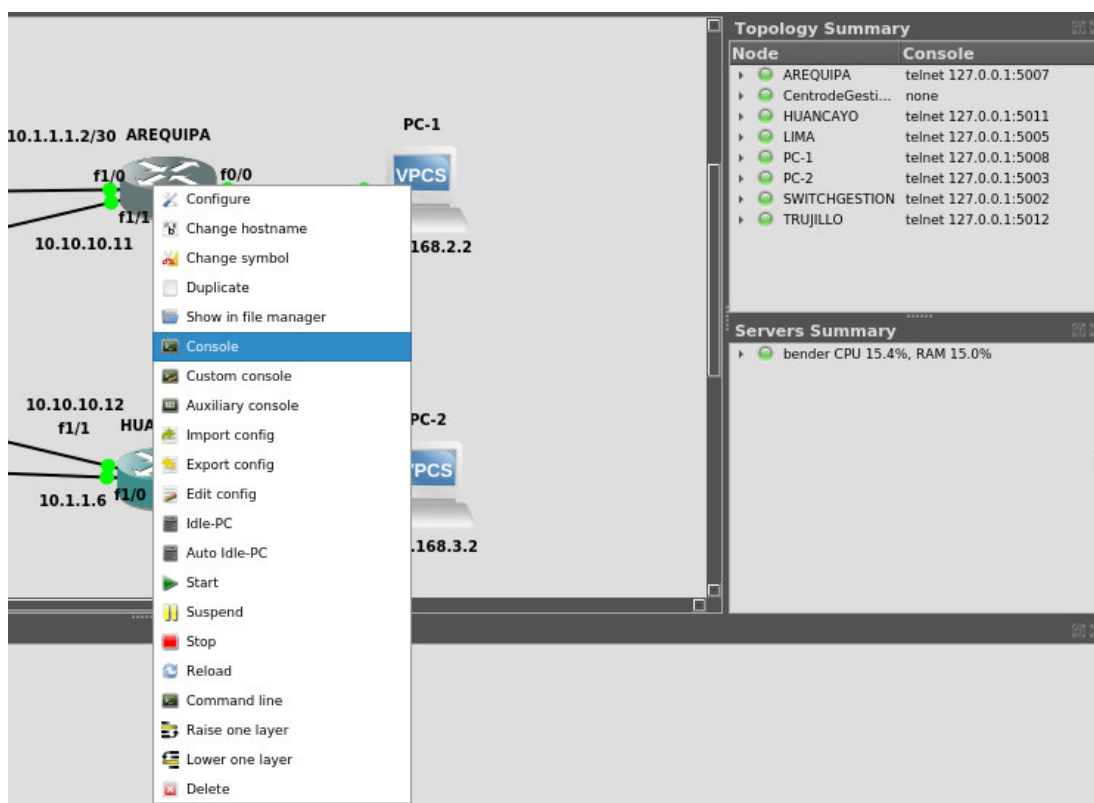


Figura 4.50.- Acceso a enrutador “Arequipa”

b) Apagado de interfaz del enrutador “Arequipa”: Se accede a la interfaz “FastEthernet 1/0” del enrutador “Arequipa”. Se apaga la interfaz con un comando “shutdown” y se reactiva nuevamente con un comando “no shutdown” tal como muestra la figura 4.51 para que los equipos generen una secuencia de eventos que son recolectados por el softbot gestor de notificaciones.

```

AREQUIPA
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#
AREQUIPA(config-if)#interface FastEthernet 1/0
AREQUIPA(config-if)#
AREQUIPA(config-if)#shutdown
AREQUIPA(config-if)#
*Oct  8 16:11:55.811: %OSPF-5-ADJCHG: Process 1, Nbr 10.10.10.10 on FastEthernet
1/0 from FULL to DOWN, Neighbor Down: Interface down or detached
AREQUIPA(config-if)#
*Oct  8 16:11:57.783: %LINK-5-CHANGED: Interface FastEthernet1/0, changed state
to administratively down
*Oct  8 16:11:58.783: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthern
et1/0, changed state to down
AREQUIPA(config-if)#
AREQUIPA(config-if)#

```

Fig 4.51.- Apagado de interfaz F0/1 del enrutador AREQUIPA

Se observa que el softbot gestor de notificaciones detecta y registra las notificaciones, tal como se muestra en la figura 4.52.

```

*Oct  8 16:04:46.099: %OSPF-5-ADJCHG: Process 1, Nbr 10.10.10.10 on FastEthernet1/0 from FULL to DOWN, Neighbor Down: Interface down or detached
*Oct  8 16:04:48.075: %LINK-5-CHANGED: Interface FastEthernet1/0, changed state to administratively down
*Oct  8 16:04:49.075: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to down
*Oct  8 16:04:49.691: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10.99)

```

Figura 4.52.- Registro de las notificaciones recibidas por el softbot (red inestable)

El “Agente_Grafico” muestra el posible problema ocurrido y recomienda al operador del NOC tomar una serie de acciones, como se observa en las figuras 4.53, 4.54 y 4.55.

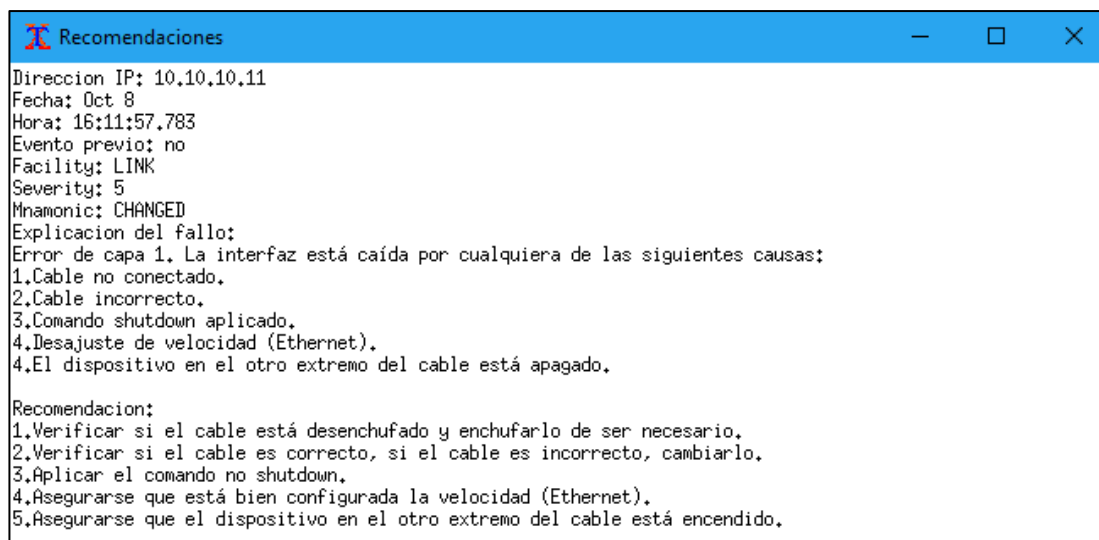


Figura 4.53.- Diagnóstico y recomendación ante apagado de interfaz de red de enrutador “Arequipa” a nivel de capa 1

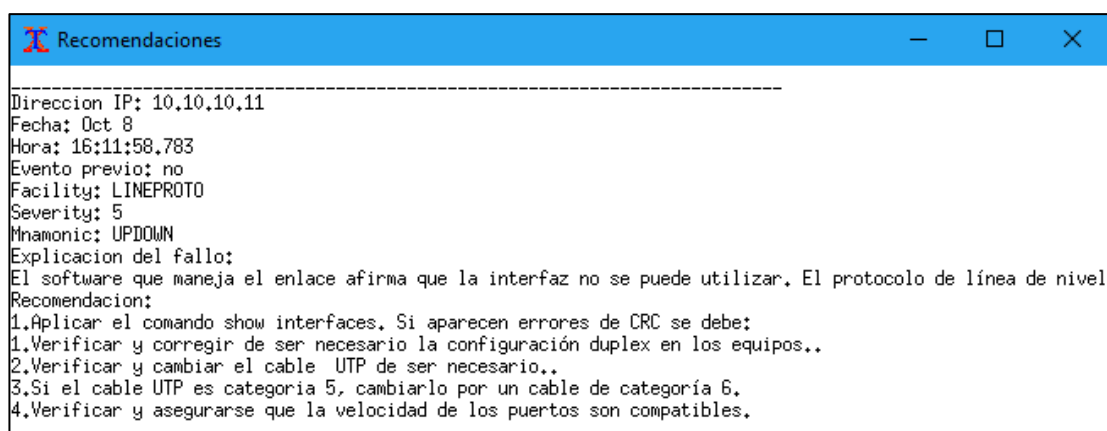


Figura 4.54.- Diagnóstico y recomendación ante apagado de interfaz de red de enrutador “Arequipa” a nivel de capa 2

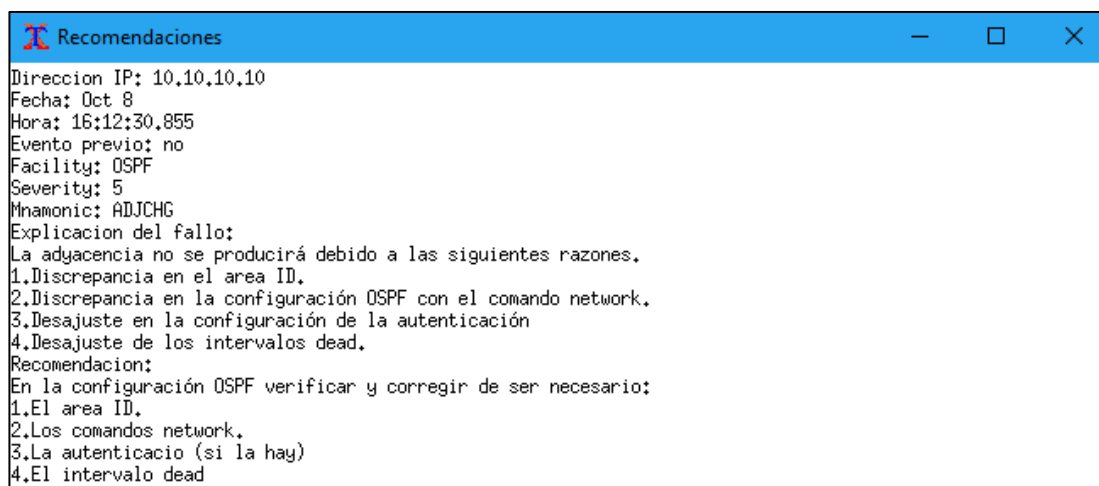
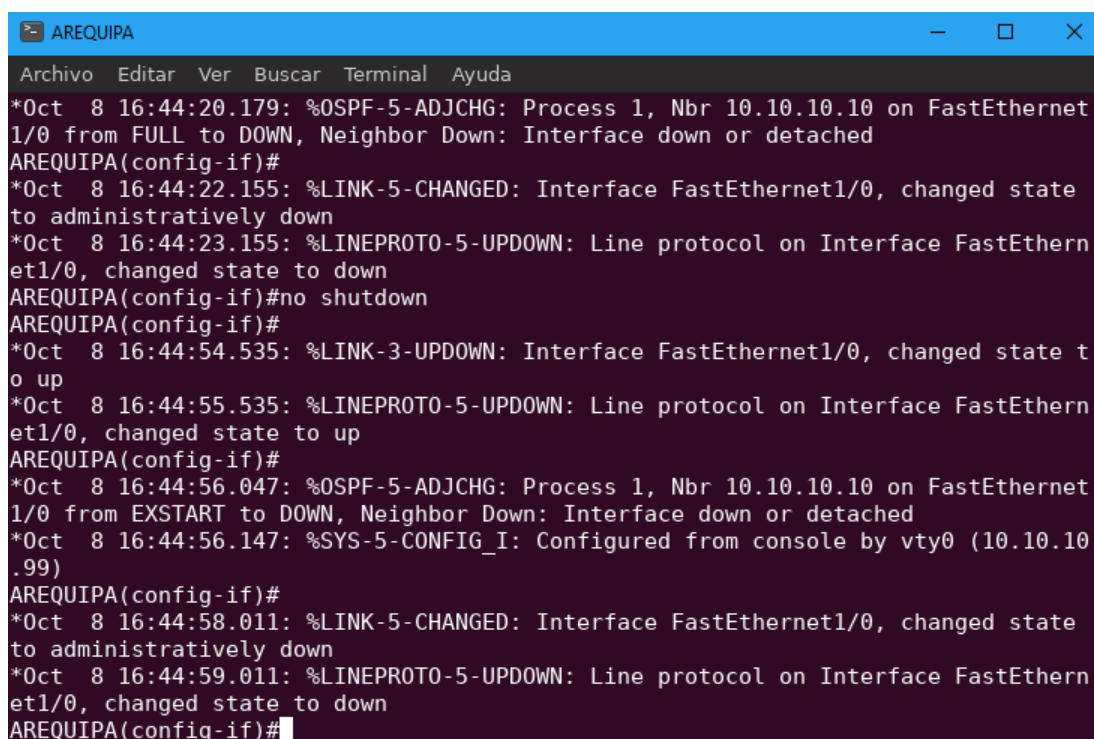


Figura 4.55.- Diagnóstico y recomendación ante apagado de interfaz de red de enrutador “Arequipa” a nivel de capa 3

c) Repetir el apagado y encendido de interfaz para generar “flapping”: Se apaga y enciende cuatro veces en un minuto la interfaz “FastEthernet 1/0” del enrutador “Arequipa”, generando secuencias de notificaciones, tal como se observa en la figura 4.56.



```

AREQUIPA
Archivo Editar Ver Buscar Terminal Ayuda
*Oct 8 16:44:20.179: %OSPF-5-ADJCHG: Process 1, Nbr 10.10.10.10 on FastEthernet
1/0 from FULL to DOWN, Neighbor Down: Interface down or detached
AREQUIPA(config-if)#
*Oct 8 16:44:22.155: %LINK-5-CHANGED: Interface FastEthernet1/0, changed state
to administratively down
*Oct 8 16:44:23.155: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthern
et1/0, changed state to down
AREQUIPA(config-if)#no shutdown
AREQUIPA(config-if)#
*Oct 8 16:44:54.535: %LINK-3-UPDOWN: Interface FastEthernet1/0, changed state t
o up
*Oct 8 16:44:55.535: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthern
et1/0, changed state to up
AREQUIPA(config-if)#
*Oct 8 16:44:56.047: %OSPF-5-ADJCHG: Process 1, Nbr 10.10.10.10 on FastEthernet
1/0 from EXSTART to DOWN, Neighbor Down: Interface down or detached
*Oct 8 16:44:56.147: %SYS-5-CONFIG_I: Configured from console by vty0 (10.10.10
.99)
AREQUIPA(config-if)#
*Oct 8 16:44:58.011: %LINK-5-CHANGED: Interface FastEthernet1/0, changed state
to administratively down
*Oct 8 16:44:59.011: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthern
et1/0, changed state to down
AREQUIPA(config-if)#

```

Figura 4.56.- Flapping en la interfaz “FastEthernet 1/0” del enrutador “Arequipa”

d) Correlación de notificaciones: El softbot gestor de notificaciones correlaciona las notificaciones recibidas y lanza un cuadro visual emergente que indica el origen del problema, tal como se observa en la figura 4.57.

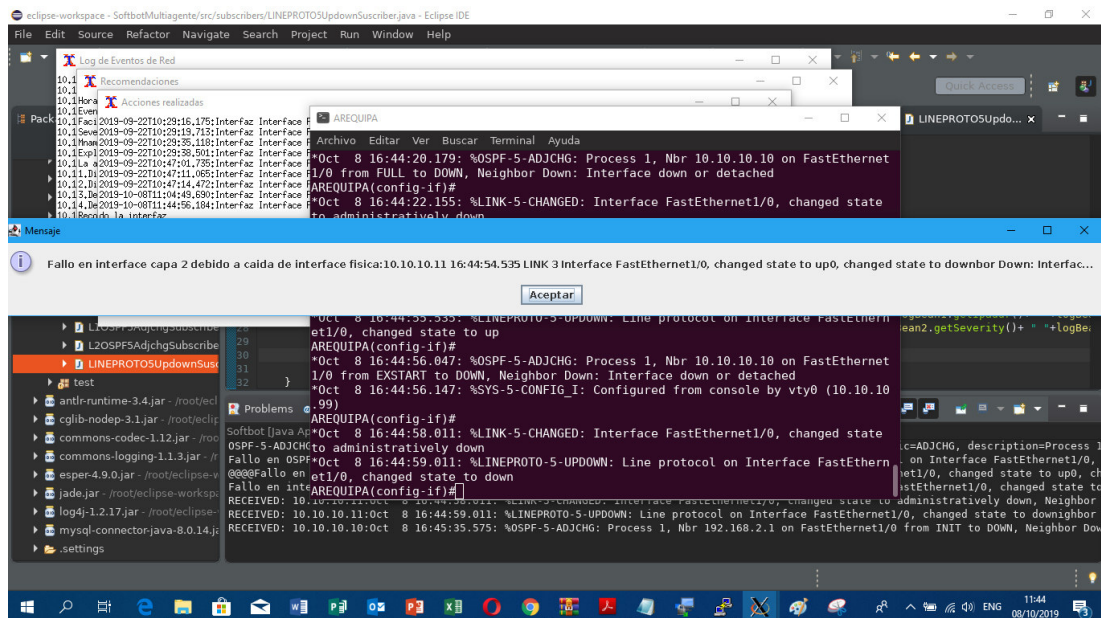


Figura 4.57.- Alerta generada por el softbot cuando correlaciona “flapping”

e) Apagado de interface en flapping: El softbot gestor de notificaciones procede a apagar en forma desatendida la interface que está generando “flapping”, e informa mediante una notificación al operador del NOC, tal como se observa en la figura 4.58.

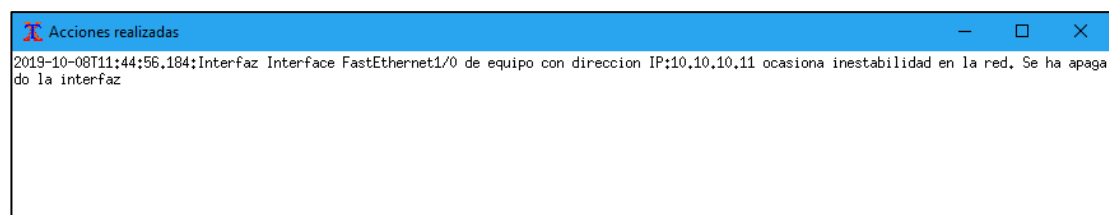


Figura 4.58.- Notificación de la acción “apagado de interfaz” generada por el softbot cuando correlaciona “flapping”

4.5. Demostración de la hipótesis

4.5.1. Demostración 1.

El gestor de notificaciones disminuye el tiempo empleado en el proceso de diagnóstico de fallos.

Resultado preliminar: Discriminar gran cantidad de notificaciones recibidas para priorizar solo las notificaciones relevantes:

Demuestra hipótesis: Identifica entre dos mil notificaciones una notificación importante y extrae conocimiento para la solución.

- Tamaño de los datos de muestra en dos mil notificaciones (entrada): 196KB
- Tamaño de la información del resultado (salida): 1KB

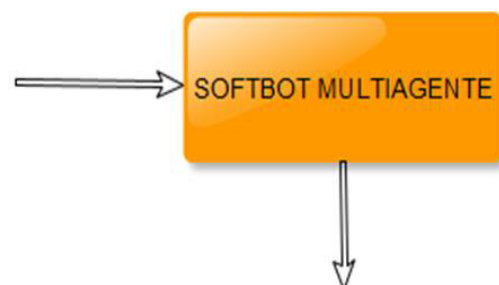
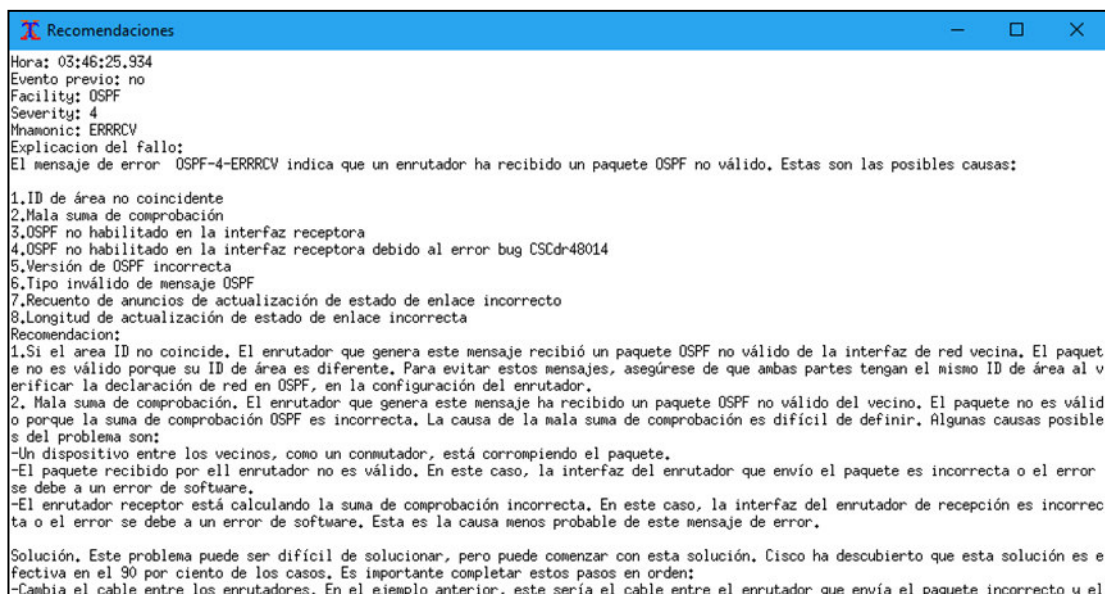
“El operador de red ya no debe analizar 196KB de datos. En vez de ello recibe 1KB de conocimiento.”



```

192.168.1.61:Sep 13 03:46:44.620: %C1700_EM-6-UNAVAILABLE: %s shutting down u$
192.168.1.61:Sep 13 03:46:46.167: %CCM-5-FMD_FLOWS_EXCEEDED: CCM Cac$
192.168.1.61:Sep 13 03:46:47.972: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will u$
192.168.1.61:Sep 22 15:47:13.067: %LINK-5-CHANGED: Interface FastEthernet1/0, c$
192.168.1.61:Sep 13 03:45:16.519: %SYS-5-CONFIG_I: Configured from console by v$
192.168.1.61:Sep 13 03:46:13.171: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is OFFle by v$
192.168.1.61:Sep 13 03:46:14.231: %ACE-6-UNAVAILABLE: %s has been moved from %s $
192.168.1.61:Sep 13 03:46:15.172: %AP-6-POSTURE_START_VALIDATION: IP=| Interface$
192.168.1.61:Sep 13 03:46:17.111: %CRYPTO-6-GDOI_ON_OFF: GDOI is OFF OFFterface$
192.168.1.61:Sep 13 03:46:19.332: %SYS-6-LOGGINGHOST_STARTSTOP: Logging to host$
192.168.1.61:Sep 13 03:46:21.134: %LINK-3-LINK_FAULT: Interface FastEthernet1/0$
192.168.1.61:Sep 13 03:46:21.134: %AAA-6-ACCTSESSIDWRAP: Accounting-Session-ID %$
192.168.1.61:Sep 13 03:46:22.178: %BGP-6-BIGCHUNK: Big chunk pool request ([dec]$
192.168.1.61:Sep 13 03:46:23.563: %ATM-5-UNAVAILABLE: Address registration on %s$
192.168.1.61:Sep 13 03:46:24.553: %AP-6-POSTURE_DOWNLOAD_ACL: Send AAA request t$
192.168.1.61:Sep 13 03:46:25.823: %BGP-5-ADJCHANGE: neighbor Send AAA request t$
192.168.1.61:Sep 13 03:46:26.122: %C1700_EM-6-UNAVAILABLE: %s shutting down st t$
192.168.1.61:Sep 13 03:46:28.642: %C7600_SSC600-3-SINGLE_SPA_MODE: Module will u$
192.168.1.61:Sep 13 03:46:29.729: %CBUS-5-SENDIPCMDRECOV: Send Ip Cmd Succeeded $
  
```

2000 eventos generados por la red

```

Recomendaciones
Hora: 03:46:25.934
Evento previo: no
Facility: OSPF
Severity: 4
Mnemonic: ERRRCV
Explicación del fallo:
El mensaje de error OSPF-4-ERRRCV indica que un enrutador ha recibido un paquete OSPF no válido. Estas son las posibles causas:
1.ID de área no coincidente
2.Mala suma de comprobación
3.OSPF no habilitado en la interfaz receptora
4.OSPF no habilitado en la interfaz receptora debido al error bug CSCdr48014
5.Versión de OSPF incorrecta
6.Tipo inválido de mensaje OSPF
7.Recuento de anuncios de actualización de estado de enlace incorrecto
8.Longitud de actualización de estado de enlace incorrecta
Recomendación:
1.Si el area ID no coincide, El enrutador que genera este mensaje recibió un paquete OSPF no válido de la interfaz de red vecina. El paquete no es válido porque su ID de área es diferente. Para evitar estos mensajes, asegúrese de que ambas partes tengan el mismo ID de área al verificar la declaración de red en OSPF, en la configuración del enrutador.
2. Mala suma de comprobación, El enrutador que genera este mensaje ha recibido un paquete OSPF no válido del vecino. El paquete no es válido porque la suma de comprobación OSPF es incorrecta. La causa de la mala suma de comprobación es difícil de definir. Algunas causas posibles del problema son:
-Un dispositivo entre los vecinos, como un conmutador, está corrompiendo el paquete.
-El paquete recibido por el enrutador no es válido. En este caso, la interfaz del enrutador que envió el paquete es incorrecta o el error se debe a un error de software.
-El enrutador receptor está calculando la suma de comprobación incorrecta. En este caso, la interfaz del enrutador de recepción es incorrecta o el error se debe a un error de software. Esta es la causa menos probable de este mensaje de error.
Solución. Este problema puede ser difícil de solucionar, pero puede comenzar con esta solución. Cisco ha descubierto que esta solución es efectiva en el 90 por ciento de los casos. Es importante completar estos pasos en orden:
-Cambia el cable entre los enrutadores. En el ejemplo anterior, este sería el cable entre el enrutador que envía el paquete incorrecto y el
  
```

Figura 4.59.- Generación de conocimiento a partir de datos

Resultado final: mejorar el tiempo empleado en el proceso de diagnóstico de fallos en redes IP

Del resultado 1, se definen los tiempos de identificación y planteamiento de solución en forma manual y a través del softbot multiagente:

- Tiempo promedio de identificación y planteamiento de solución manual ante una notificación importante entre dos mil notificaciones recibidas para un administrador poco experto: TPM
- Tiempo promedio de identificación y planteamiento de solución por parte del softbot multiagente ante una notificación importante entre dos mil notificaciones recibidas: TPS
- Teniendo como resultado el siguiente cuadro que demuestra la hipótesis:

TPM en minutos	TPS en minutos
15	0.05

- Tamaño de los datos de muestra en dos mil notificaciones (entrada): 196KB
- Tamaño de la información del resultado (salida):1KB

“El operador de red ya no debe analizar 196KB de datos en 15 minutos. En vez de ello recibe 1KB de conocimiento en 0.05 de minuto.”

4.5.2. Demostración 2

El gestor de notificaciones disminuye el tiempo empleado en el proceso de mitigación de fallos de inestabilidad en la red (flapping OSPF)

Resultado final: Mejorar el tiempo empleado en el proceso de mitigación manual de problemas de inestabilidad en redes IP

El softbot mitiga el problema de inestabilidad apagando la interface de red intermitente

TPSM en minutos	TPSS en minutos
19	0.09

- Tamaño de los datos de muestra en 2000 notificaciones (entrada): 196KB
- Tamaño de la información del resultado (salida): 1KB

“El operador de red ya no debe analizar 196KB de datos en 19 minutos para solucionar el problema de flapping OSPF de una interfaz de la red. En vez de ello recibe 1KB de conocimiento y se mitiga el problema de flapping OSPF en 0.09 de minuto.”

4.6. Resumen y discusión final

Para la realización de pruebas se empleó un computador con los suficientes recursos para albergar el entorno de desarrollo en lenguaje de programación JAVA con Eclipse, JADE y ESPER, el softbot gestor de notificaciones ejecutándose y la red de enrutadores virtualizada de pruebas con GNS3. Las pruebas del módulo gestor de notificaciones se dividieron en dos tipos:

a) Detección de un evento relevante entre muchos eventos

poco importantes: consistió en el envío de dos mil eventos (ruido) y un evento importante insertado en forma aleatoria. Las pruebas fueron realizadas con cada uno de los siguientes eventos:

- 1) LINK-3-UPDOWN,
- 2) LINEPROTO-5-UPDOWN,
- 3) LINK-5-CHANGED,
- 4) OSPF-5-ADJCHG,
- 5) OSPF-4-CONFLICTING_LSAID,
- 6) SFF8472-5-THRESHOLD_VIOLATION,
- 7) OSPF-4-ERRRCV,
- 8) PLATFORM-ASM-0-FAULT_RECOVERY,
- 9) PLATFORM-ASM-3- MALLOC_FAILED,
- 10) MALLOC_FAILED,

- 11) PLATFORM-ENVMON-0-CARD,
- 12) PLATFORM-ENVMON-2-CHASSIS_OVERTEMP,
- 13) PLATFORM-ENVMON-2-FAN,
- 14) PLATFORM-ENVMON-2-FAN,
- 15) LINK-3-LINK_FAULT,
- 16) LINK-3-FCS_ERROR,
- 17) LINK-3-TOOSMALL,
- 18) CDP-4-NATIVE_VLAN_MISMATCH,
- 19) EQUIPMENT-5-DOWN,
- 20) EQUIPMENT-5-UP.

En cada una de las pruebas efectuadas, el gestor de notificaciones fue capaz de identificar y extraer el evento importante del “ruido”, y alertar del evento sucedido y las acciones a realizar para solucionar el problema.

- b) Detección de patrones entre muchos eventos poco importantes:** consistió en pruebas de apagado y encendido de una interfaz de red de un enrutador de la red elegido al azar, para la generación de eventos importantes (LINEPROTO-5-UPDOWN y OSPF-5-ADJCHG). La generación de “flapping” en la red se hizo de tal forma que, cuatro intermitencias consecutivas en un minuto determinen el “flapping”. El gestor de notificaciones no solo fue capaz de identificar y correlacionar la secuencia temporal de eventos del “ruido”, sino también pudo determinar el “flapping” en la red, alertar del evento sucedido y apagar la interfaz en “flapping” en forma desatendida para mitigar el problema.

Se ha observado claramente que el tiempo que tomó softbot gestor de notificaciones en mitigar el problema fue casi en tiempo real, tiempo que contrasta con lo que demora un operador en detectar las notificaciones, analizarlas y tomar acción; y es más crítico aún, porque en muchos casos el operador no se da cuenta del evento

sucedido hasta que se observan las repercusiones negativas en la red.

Se tiene que indicar que, si bien la implementación del trabajo se realizó en un ambiente virtualizado con una cantidad de enrutadores limitada (virtualizar cada sistema operativo del enrutador consume recursos de procesamiento del equipo de pruebas). En la realidad, la red gestionada que genera notificaciones hacia el softbot gestor de notificaciones puede ser de tamaño mucho más grande, y no necesariamente estar compuesta por solo enrutadores. El softbot gestor de notificaciones puede extenderse para que procese notificaciones generadas por cualquier infraestructura de red, teniendo solamente implementar sensores que traduzcan las notificaciones particulares de cada equipo de comunicaciones a un formato de eventos estándar.

Si bien se encogió un conjunto de eventos limitado, el softbot gestión de eventos críticos a través de la librería CEP ESPER, tiene la capacidad de procesar dos mil notificaciones generadas por la red. La capacidad de procesamiento nuevamente depende de las prestaciones de hardware del equipo donde está instalado el softbot gestor de notificaciones. El softbot gestor de notificaciones se puede extender para que procese toda clase de eventos simples o correlacione notificaciones complejas con solo agregar más “listeners” y “suscribers”, por lo tanto, es extensible.

CONCLUSIONES

1. El sistema gestor de notificaciones desarrollado tiene la capacidad para discriminar y priorizar solo las notificaciones relevantes. En las pruebas realizadas, el sistema identificó entre dos mil notificaciones, una notificación importante y extrajo conocimiento para la solución de problemas de red.
2. Se ha mejorado sustancialmente el tiempo empleado en el proceso de diagnóstico de fallos en redes IP. Los tiempos promedio de identificación y planteamiento de solución en forma manual para dos mil notificaciones fue de 15 minutos, en cambio, el tiempo de identificación y planteamiento de solución realizado por el softbot gestor de notificaciones fue de 0.05 de minuto.
3. El softbot gestor de notificaciones mejoro el tiempo empleado en el proceso de mitigación de problemas de inestabilidad en redes IP con flapping OSPF. El tiempo promedio de identificación y mitigación manual ante un conjunto de notificaciones de intermitencia importantes entre dos mil notificaciones fue de 19 minutos, mientras que el softbot gestor de notificaciones realizo la misma tarea de mitigación en 0.09 de minuto.
4. El presente trabajo ha abordado el problema de la gestión de fallos en redes de comunicaciones desde la perspectiva de agentes de software. A partir de este trabajo, se ha propuesto demostrar que el desarrollo de sistemas de gestión de fallos empleando las ventajas que proporciona la ingeniería de agentes de software es factible y que, además, en el ámbito académico existen varias metodologías para el diseño de sistemas empleando agentes de software, que, en este caso particular se ha verificado que es válido desarrollar este tipo de aplicaciones empleando la Metodología Orientada a Agentes INGENIAS.

Como resultado adicional del trabajo realizado podemos indicar en forma general lo siguiente:

1. Los estudios académicos que relacionan la gestión de redes y los sistemas multiagente abarcan algunos aspectos puntuales de la gestión en redes de área local pequeñas, quedando mucho campo por explorar.
2. No existe consenso y menos aún un estándar, en cuanto al uso de las metodologías de desarrollo de software orientado a agentes. Debido a ello, el desarrollo de sistemas orientados a agentes no se ha visto desarrollado e impulsado masivamente en la creación de soluciones comerciales, como sí sucede en el desarrollo de software orientado a objetos. Debemos de tener en cuenta que las características del desarrollo de sistemas multiagente es mucho más complejo que el desarrollo de orientado a objetos, por cuanto el agente debe tener autonomía, habilidad social, comunicación con su entorno, etc, en comparación del objeto que es simplemente reactivo.

RECOMENDACIONES

1. Se recomienda seguir desarrollando el softbot gestor de notificaciones para gestionar un conjunto mucho mayor de eventos complejos, la solución es lo suficientemente escalable para permitir incrementar su capacidad sin impactar en su complejidad.
2. Se recomienda incrementar la capacidad que tiene los agentes para moverse a través de la red, interactuar con el ser humano, razonar y actuar en forma desatendida resulta especialmente interesante y deseable en la solución de problemas de red. En este trabajo de investigación no se ha considerado la movilidad de los agentes a través la red, característica recomendable que resulta interesante de investigar para redes muy grandes.
3. Es recomendable el uso procesamiento complejo de eventos (CEP) como herramienta muy potente para detección y correlación de eventos en tiempo real, que junto a los sistemas multiagente forman un campo interesante para futuras investigaciones no solo en el campo de la gestión de redes, también en la seguridad de la información, procesos de toma de decisiones en tiempo real, expertos que razonan en forma automática sin intervención humana, controles industriales, scada, etc.

REFERENCIAS BIBLIOGRÁFICAS

1. (Grimes, 2018). Why you need centralized logging and event log management. Recuperado del 01 de octubre de 2019. Sitio web:
<https://www.csoononline.com/article/3280123/why-you-need-centralized-logging-and-event-log-management.html>
2. (PSC, 2020) Progress Software Corporation. Recuperado el 10 de septiembre de 2019. Sitio web:
<https://www.whatsupgold.com/best-practices/event-log-management>
3. (Kent, Souppaya, 2003). Guide for Computer Security Log Management. Recuperado el 10 de septiembre de 2019. Sitio web:
<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-92.pdf>
4. (Sematext, 2019) The Complete Guide to Log Analysis: What it is, How it Works, Use Cases & Tools. Recuperado el 10 de septiembre de 2019. Sitio web: <https://sematext.com/blog/log-analysis/#toc-reduce-problem-diagnosis-and-resolution-time-2>
5. Pressman, Roger. (2010). Ingeniería de Software. Un Enfoque Práctico (7ma ed.). Ciudad de México, México: Editorial McGraw-Hill Interamericana Editores S.A.
6. Ramón Jesús Millán Tejedor (2004). Revista Comunicaciones World N° 189, IDG Communications S.A., pp. 54.
7. Ramón Jesús Millán Tejedor (1999). Revista Windows NT 2000 Actual N° 12, Prensa Técnica S.A.
8. Belén Acosta Maza (2016). Implementación de un sistema de gestión de redes en RTVA. Ingeniería de Telecomunicación. Departamento de Ingeniería de Sistemas y Automática. Área de Ingeniería Telemática. Universidad de Sevilla.
9. Umesh Hodeghatta Rao (2011). Challenges of Implementing Network Management Solution. International Journal of Distributed and Parallel Systems (IJDPS) Vol.2, No.5, pp. 67, India.
10. Antonio Cuesta García (2016). Sistema de monitorización y correlación de eventos en gestión de redes. Trabajo Fin de Grado en Ingeniería de las Tecnologías de las Telecomunicaciones. Departamento de Ingeniería Telemática. Escuela Técnica Superior de Ingeniería Universidad de Sevilla, pp. 2.
11. Cisco Systems. Network Management System: Best Practices White Paper. Recuperado del 01 de octubre de 2019. Sitio web:
<https://www.cisco.com/c/en/us/support/docs/availability/high-availability/15114-NMS-bestpractice.html>
12. Carlos Villarrubia Jiménez (2000). Líneas de investigación en informática. Universidad Castilla de la Mancha, pp. 180.

13. Maes P. (1989) "Situated Agents Can Have Goals", Designing Autonomous Agents: Theory and Practice From Biology to Engineering and Back, pp 49-71, Maes, Pattie, (Ed), 1989.
14. R.A. Brooks. A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation, pp. 14-23, 1986.
15. Ignacio Herrero Reder. (2015). Arquitectura de Comportamientos Reactivos para Agentes Robóticos basada en CBR. Tesis Doctoral. Escuela Técnica Superior de Ingeniería de Telecomunicación. Málaga.
16. Iván Palomares Carrascosa (2010). Desarrollo de un Sistema Multi-Agente para automatizar Procesos de Consenso en Problemas de Toma de Decisión en Grupo. Trabajo Tutelado de Iniciación a la Investigación. Universidad de Jaén.
17. I.A. Ferguson. Towards an architecture for adaptive, rational, mobile agents. Proceedings of the Third European Workshop on Modelling Autonomous Agents and Multi-Agent World, pp. 249-262, 1991.
18. Ramirez. (2006). El software libre y la independencia tecnológica. Recuperado el 22 de febrero de 2019. Sitio web: <https://www.aporrea.org/imprime/a20073.html>
19. Rendueles & Dreher. (2007). La Epistemología y los Sistemas de Información Basados en la Inteligencia Artificial. Revista Telematique, Volumen 6, Número 1.
20. González, Rodrigo . (2011). Máquinas sin engranajes y cuerpos sin mentes. ¿Cuán dualista es el Funcionalismo de Máquina de Turing?. Revista de Filosofía Universidad de Chile, Volumen 67, p. 184.
21. Turing (1951). Can Digital Computers Think?. Recuperado el 13 de febrero de 2019. Sitio web: <https://aperiodical.com/wp-content/uploads/2018/01/Turing-Can-Computers-Think.pdf>
22. Bond A.H. y Gasser L. (1988). Reading in Distributed Artificial Intelligence. San Mateo USA: Morgan Kaufmann.
23. Durfee & Rosenschein. (1994). Distributed Problem Solving and Multi-Agent Systems. Recuperado el 23 de febrero de 2019 de la Association for the Advancement of Artificial Intelligence (AAAI), sitio web: <http://www.aaai.org/Library/Workshops/1994/ws94-02-004.php>
24. Akinyokun, O. C., Ekuewa, J. B., and Arekete, S. A. (2017). Development of agent-based system for monitoring software resources in a network environment, Artificial Intelligence Re-search, Publicado por Sciedu Press, Vol. 3, No. 3.
25. Akinyokun, O. C., Ekuewa, J. B., and Arekete, S. A. (2014). Development of agent-based system for monitoring software resources in a network environment, Artificial Intelligence Re-search, Publicado por Sciedu Press, Vol. 3, No. 3.
26. Karina Raya-Díaz, Carelia Gaxiola-Pacheco and Manuel Castañón-Puga. (2014). Network Management based on Multi-Agent. Memorias del 7mo. Congreso Internacional en Dos milicias Computacionales, CiComp, pp. 613, México.

27. Gustavo Isaza, Maria H. Mejía, Luis Castillo, Adriana Morales, Nestor Duque (2012). Network Management using Multi-Agents System. Artículo en Advances in Distributed Computing and Artificial Intelligence Journal.
28. Escuela Superior de Ingenieros de Telecomunicación. Universidad Politécnica de Madrid (2019). Vinton Cerf. Recuperado el 24 de febrero de 2019, sitio web: <https://www.etsit.upm.es/escuela/informacion-general/honoris-causa/vinton-cerf.html>
29. Fall & Stevens. (2012). TCP/IP Illustrated, Volume 1. Michigan, Estados Unidos: Pearson Education, Inc.
30. J. Postel. (1981). Transmission Control Protocol. Darpa Internet Program. Protocol Specification. Recuperado el 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF), sitio web: <https://tools.ietf.org/html/rfc793>
31. IETF 791. (1981). Internet Protocol. Darpa Internet Program. Protocol Specification. Recuperado el 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF), sitio web: <https://tools.ietf.org/html/rfc791>
32. J. Postel. (1980). User Datagram Protocol. Recuperado el 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF) Sitio web: <https://www.ietf.org/rfc/rfc768.txt>
33. IETF 793. (1981). Transmission Control Protocol. Recuperado el 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF) Sitio web: <https://www.ietf.org/rfc/rfc793.txt>
34. J. Postel, J. Reynolds (1983). Telnet Protocol Specification. Recuperado el 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF) Sitio web: <https://www.ietf.org/rfc/rfc854.txt>
35. J. Postel, J. Reynolds (1985). File Transfer Protocol. Recuperado el 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF) Sitio web: <https://www.ietf.org/rfc/rfc959.txt>
36. J. Klesin (2008). Simple Mail Transfer Protocol. Recuperado el 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF) Sitio web: <https://www.ietf.org/rfc/rfc5321.txt>
37. Margaret Rouse (2019). Gestión de la configuración de red, NCM. Recuperado el 24 de febrero de 2019, de SearchDatacenter. Sitio web: <https://searchdatacenter.techtarget.com/es/definicion/Gestion-de-la-configuracion-de-red-NCM>
38. W. Richard Stevens. (1994). TCP/IP Illustrated, Vol. 1: The Protocols. Estados Unidos: Addison-Wesley Professional.
39. R. Gerhards. (2009). The Syslog Protocol. 24 de febrero de 2019, de Grupo de Trabajo de Ingeniería de Internet (IETF) Sitio web: <https://tools.ietf.org/html/rfc5424>
40. Cisco. (2008). System Message Logging. Recuperado el 24 de febrero de 2019, de Cisco Systems Sitio web: <https://www.cisco.com/c/en/us/td/docs/enrutadores/access/wireless/software/guide/SysMsgLogging.html>
41. Antonio Barba Martí. (1999). Gestión de Red. Universidad Politécnica de Catalunya: Ediciones Universidad Politécnica de Catalunya.

42. Russell & Norvig . (2008). Inteligencia Artificial. Un Enfoque Moderno. Madrid, España: Pearson.
43. Burguillo, Juan Carlos. (2009). Redes y Sistemas Inteligentes. Recuperado el 2 de marzo de 2019 del site de la Asignatura del Máster Universitario en Ingeniería Telemática de Universidad de Vigo Sitio web: <https://sites.google.com/site/jcburgui/redes-y-sistemas-inteligentes>
44. Torres Leyva, Joel. (2012). Arquitecturas Multiagentes. Recuperado el 2 de marzo de 2019, de Universidad Autónoma de Guerrero Unidad Académica de Ingeniería Sitio web: <https://es.scribd.com/document/112387080/Arquitecturas-Multiagentes>
45. Sycara, Katia P. (1998). Multiagent Systems. Revista AI Magazine, Vol 19 Number 2, p. 79-92.
46. Wooldridge Michael & Jennings, Nicholas R.. (1995). Intelligent agents: theory and practice. The Knowledge Engineering Review, Volume 10, Issue 2 , pp. 115-152.
47. Nwana, H (1996). Software Agents: An Overview. The Knowledge Engineering Review. Cambridge University Press. v.3, p.1-40.
48. Ana Mas; María Victoria Belmonte; Francisco Garijo; Julián Pavón Mestras; José Luis Pérez de la Cruz; et al. (2005). Agentes Software y Sistemas Multiagente. Madrid, España: Prentice Hall.
49. Mallcu Morales, Juan Wilson (2008). Agentes y Sistemas Multi-Agente. RITS, 2008, no.1, p.110-114. ISSN 1997-4044
50. Dignum V. y Dignum F. (2005). Task and Social Coordination in Agent Organization. Recuperado el 3 de marzo de 2018 de Institute of Information and Computing Sdos milces, Utrecht University Sitio web: <http://www.cs.uu.nl/research/techreps/UU-CS-2005-015.html>
51. Chen W. y Decker K. (2004). Managing Multi-Agent Coordination, Planning, and Scheduling. Recuperado el 3 de marzo de 2018 de Computer & Information Sdos milces, University of Delaware Sitio web: <https://dl.acm.org/citation.cfm?id=1018942>
52. Liviu Panai y Sean Luke (2005). Cooperative Multi-Agent Learning: The State of the Art. Journal Autonomous Agents and Multi-Agent Systems , Volume 11 , 3.
53. Jennings N., Faratin P., Lomuscio A., Parsons S., Sierra C., and Wooldridge M (2001). Automated negotiation: prospects, methods and challenges. Int. J. of Group Decision and Negotiation, pp. 199–215.
54. The Foundation for Intelligent Physical Agents. (2002). FIPA. Recuperado el 3 de marzo de 2018, de The Foundation for Intelligent Physical Agents Sitio web: <http://www.fipa.org/specs/fipa00001/SC00001L.html>
55. The Foundation for Intelligent Physical Agents. (2019). FIPA. Recuperado el 3 de marzo de 2018, de The Foundation for Intelligent Physical Agents Sitio web: <http://www.fipa.org/>
56. The Foundation for Intelligent Physical Agents. (2012). FIPA. Recuperado el 17 de marzo de 2018, de The Foundation for Intelligent Physical Agents Sitio web: <http://fipa.org/specs/fipa00097/SC00097B.pdf>

57. Fabio Luigi Bellifemine, Giovanni Caire, Dominic Greenwood. (2007). Developing Multi-Agent Systems with JADE, pp. 17. John Wiley and Sons Ltd.
58. M. Amor, L. Fuentes, M. Pinto. (2003). Interoperabilidad entre Plataformas de Agentes FIPA: Una Aproximación Basada en Componentes. Tercer congreso iberoamericano de telemática. Montevideo, Uruguay, 30 y 31 de octubre de 2003.
59. Vijayan Sugumaran (2009). Tetodological Advancements in Intelligent Information Technologies. Information Sdos milce Reference, Oakland University, USA, pp. 114.
60. Jean Vaucher, Ambroise Ncho (2003). Dep. d'informatique Université de Montréal. Recuperado el 4 de marzo de 2019, de Universidad de Montreal, sitio web: <https://www.iro.umontreal.ca/~vaucher/Agents/Jade/JadePrimer.html>
61. Francisco Javier Arias M.Sc, Germán Darío Zapata M.Sc., Leonardo Vasquez M.Sc & Demetrio Arturo Ovalle. (2011). Inteligencia Artificial Distribuida: Conceptos y Aplicaciones. Recuperado el 4 de marzo de 2019, de Universidad Nacional de Colombia Sitio web: <http://grupotyt.medellin.unal.edu.co/grupo-tyt/produccion-academica>
62. Corchado J. M. (2005). Modelos y Arquitecturas de Agente. Recuperado el 05 de marzo de 2019 de Researchgate, sitio web: https://www.researchgate.net/publication/273446104_Modelos_y_Arquitecturas_de_Agente_-140_2_Modelos_y_Arquitecturas_de_Agente
63. Gómez Sanz. (2003). Metodologías para el diseño de sistemas multi-agente. Revista Iberoamericana de Inteligencia Artificial, Vol 7 Nro 18, 51-64.
64. Gómez Sanz, Jorge. (2002). Tesis Doctoral, Modelado de Sistemas Multiagente. Recuperado el 5 de marzo de 2019, del Grupo de investigación en Aplicaciones Sociales e Interdisciplinarias basadas en Agentes (GRASIA), sitio web: <http://grasia.fdi.ucm.es/jorge/tesis.pdf>
65. Telecom Italia Lab. (2014). JAVA Agent DEvelopment Framework. Recuperado el 05 de marzo de 2019, de Telecom Italia, sitio web: <http://jade.tilab.com/>
66. Michael Eckert and Francois Bry. (2009). Complex Event Processing (CEP). Recuperado el 07 de marzo de 2019, de Institut fur Informatik, Ludwig-Maximilians-Universitat Munchen Sitio web: https://www.researchgate.net/publication/225618561_Complex_Event_Processing_CEP
67. Ignacio G.R. Gavilán. (2018). Tecnología para la digitalización de procesos (V). COMPLEX EVENT PROCESSING (CEP). Recuperado el 07 de marzo de 2019, de Reingeniería Digital. Sitio web: <http://www.reingenieriadigital.es/tecnologia-para-la-digitalizacion-de-procesos-v-complex-event-processing-cep/>
68. Sánchez Martín, Irene. (2014). Proyecto de fin de carrera. Herramienta Web para el Modelado de Consultas CEP (Complex Event Processing). Recuperado el 07 de marzo de 2019, de Universidad Politécnica de Madrid. Sitio web: <http://oa.upm.es/39255/>

69. Espertech (2018). Esper Reference printed version 8.1.0. Recuperado el 07 de marzo de 2018, de Espertech Inc. Sitio web:
<http://www.espertech.com/esper/esper-documentation/>
70. Cisco. (2008-II). Cisco IOS System Message Guide, 15 SY. Recuperado el 12 de marzo de 2019, de Cisco Systems Sitio web:
https://www.cisco.com/c/en/us/td/docs/ios/15_0sy/system/messages/15s_ysmg/sm15syovr.pdf
71. Cisco. (2012). System Messages for Cisco IOS Release 15.x SY. Recuperado el 12 de marzo de 2019, de Cisco Systems Sitio web:
https://www.cisco.com/c/en/us/td/docs/ios/15_0sy/system/messages/15s_ysmg.pdf
72. Rodrigo Emiliano, Mário Antunes "Automatic network configuration in virtualized environment using GNS3", in proc 10th International Conference on Computer Sdos milce & Education ,July 2015,publicado en IEEE Journal, paginass 25-30, 2015.
73. Chris Welsh (2013). GNS3 Network Simulation Guide. Packt Publishing Ltd. ISBN 9781782160816.
74. Piatinni M. (1996). Análisis y diseño detallado de aplicaciones informáticas de gestión. Rama Madrid.
75. Howtoinstall. (2019). Cómo instalar expect en Ubuntu 14.04 (Trusty Tahr) Recuperado el 12 de marzo de 2019, de Cisco Systems Sitio web:
<https://www.howtoinstall.co/es/ubuntu/trusty/expect>

ANEXO A

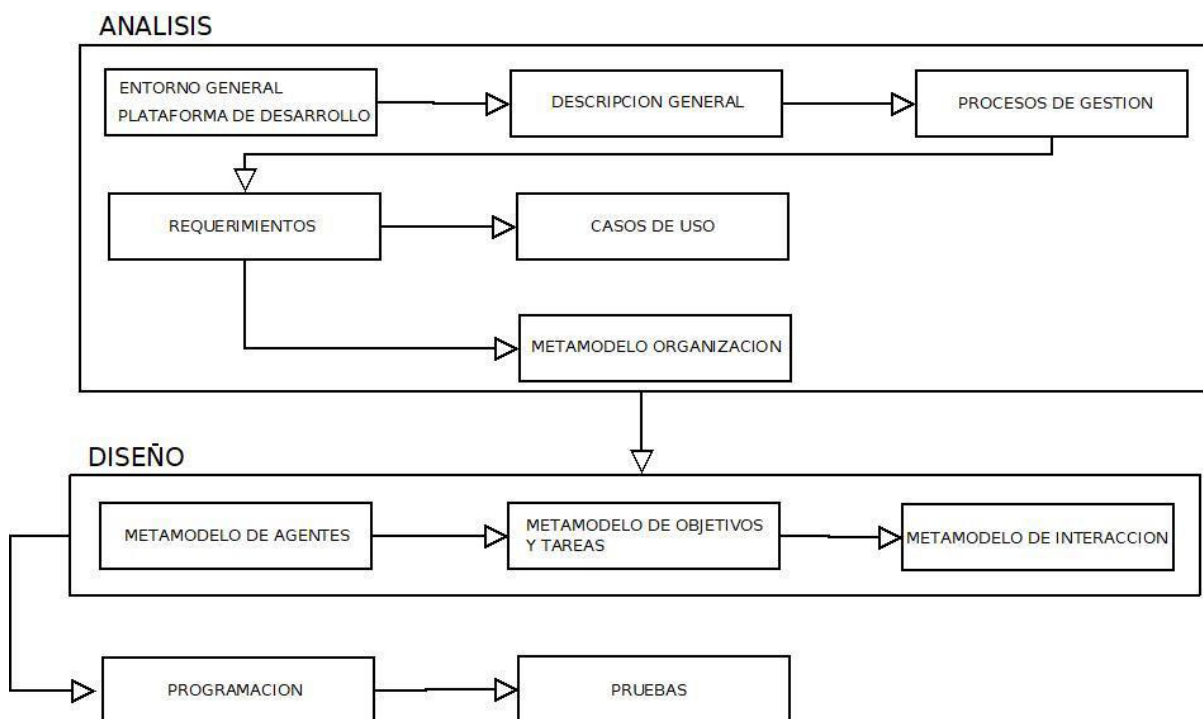
GLOSARIO

Término	Significado
Agentes Inteligentes	Es una entidad capaz de percibir su entorno, procesar tales percepciones y responder o actuar en su entorno de manera racional, es decir, de manera correcta y tendiendo a maximizar un resultado esperado.
Disponibilidad	Indica cuánto tiempo está disponible un equipo o sistema respecto de la duración total durante la que se hubiese deseado que funcionase.
Flapping	En las redes de computadoras y telecomunicaciones, el flapping de enrutamiento ocurre cuando un enrutador anuncia alternativamente una red de destino a través de una ruta y luego otra (o como no disponible, y luego disponible nuevamente) en secuencia rápida. Este fenómeno ocasiona inestabilidad en la red.
Network Management System (NMS)	Aplicación o conjunto de aplicaciones que permite a los administradores de red administrar los componentes independientes de una red dentro de un marco de administración de red más grande.
Network Operation Center (NOC)	El Network Operation Center (NOC) o Centro de Operaciones de Red, es uno o más sitios desde los cuales se efectúa el control de las redes.
Protocolo de Internet	Es un protocolo de comunicación de datos digitales

(IP)	clasificado funcionalmente en la capa de red según el modelo TCP/IP.
Redes de acceso	La red de acceso hace mención a aquella parte de la red de comunicaciones que conecta a los usuarios finales con algún proveedor de servicios.
Redes de distribución o agregación	La red de agregación representa el tramo en el que convergen los tráficos de diferentes usuarios.
Simple Network Management Protocol (SNMP)	Es un protocolo de la capa de aplicación que facilita el intercambio de información de administración entre dispositivos de red.
Sistemas Multiagente	Un sistema compuesto por múltiples agentes inteligentes que interactúan entre ellos.
Softbot	Los softbots o software robots son aquellos agentes inteligentes que hacen uso de las mismas herramientas y utilidades de que disponen los usuarios, es un agente que ayuda al usuario en el tratamiento de información. Es un representante del usuario en aquellas tareas donde el usuario haya aceptado ceder el control.

ANEXO B

DIAGRAMA DE PROCESOS DE ETAPAS DE ANALISIS Y DISEÑO DEL SISTEMA



ANEXO C

MATRIZ DE CONSISTENCIA DE

IMPACTO DE UN GESTOR DE NOTIFICACIONES MULTIAGENTE EN LA MEJORA LOS TIEMPOS DE DIAGNÓSTICO Y MITIGACIÓN DE FALLOS EN REDES IP

PROBLEMAS	OBJETIVOS	HIPOTESIS	METODOLOGIA
1. Problema Principal En los Centros de Operaciones de Redes (Networking Operations Center – NOC), la gran cantidad de notificaciones recibidas hacen que los tiempos de diagnóstico y mitigación manual de fallos sean altos. Se presentan dos problemas: a. ¿Cómo reducir el tiempo empleado en el proceso de diagnóstico manual de fallos en redes IP? b. ¿Cómo reducir el tiempo empleado en el proceso de mitigación manual de problemas de inestabilidad en redes IP?	1. Objetivo General Desarrollar un sistema multiagente gestor de notificaciones recibidas de una red, que impacte en forma positiva en la mejora de los tiempos empleados en: a.El proceso de diagnóstico de las fallas en redes IP. b.El proceso de mitigación de problemas de inestabilidad en redes IP. 2. Objetivos Específicos Desarrollar un sistema multiagente que clasifique eventos provenientes de la red para el diagnóstico y la mitigación de problemas en la red, para ello se desea lograr los siguientes objetivos específicos: a) Desarrollar un agente clasificador que periódicamente reciba, clasifique y procese mensajes de eventos SYSLOG proveniente de equipos de telecomunicaciones acerca de alarmas o fallos que están sucediendo en dichos equipos. La información recibida por los equipos es convertida en lenguaje de	1. Hipótesis General Los sistemas multiagente mejoran el tiempo empleado en los procesos de diagnóstico y mitigación de fallos en redes IP. 2.Hipótesis específicas En base a las notificaciones recibidas de la red IP: a.Los sistemas multiagente, haciendo uso de la inteligencia artificial, mejoran el tiempo empleado en el proceso de diagnóstico de fallos. b.Los sistemas multiagente, haciendo uso de la inteligencia artificial, mejoran el tiempo empleado en el proceso de mitigación de fallos de inestabilidad en la red (flapping OSPF).	1. Tipo y diseño de investigación Experimental a través de un prototipo funcional que será la una implementación de la arquitectura propuesta. El trabajo usa la Metodología INGENIAS para el análisis y desarrollo del sistema. 2. Unidad de análisis • Notificaciones provenientes de una red IP. • Tiempo de identificación y planteamiento de solución manual ante una notificación importante entre un conjunto de notificaciones recibidas. • Tiempo de identificación y de solución manual ante un conjunto de notificaciones de intermitencia importantes entre un conjunto de notificaciones recibidas. • Datos en Kilobytes. 3. Población de estudio Una red virtual de laboratorio basada en

	comunicaciones de agentes. b) Desarrollar un agente experto que efectúa diagnóstico de fallos a través de razonamiento complejo de eventos en base a la información pre procesada y obtenida del agente clasificador. El agente obtiene conclusiones y hace recomendaciones que posteriormente serán mostradas al usuario humano. c) Desarrollar un módulo de software (mitigación) proactivo-solucionador que reciba órdenes del agente experto y que actuará en forma desatendida sobre el equipo de red tratando de solucionar un problema presentado, si es que la solución del problema está al alcance del agente. d) Desarrollar un agente interfaz, que muestre los eventos recibidos, las recomendaciones y/o acciones realizadas por el sistema al usuario humano.	la tecnología IP, configurada con el protocolo de enrutamiento dinámico OSPF que genera mensajes o notificaciones. La red virtual está compuesta por cinco enrutadores o nodos que emulan una red WAN (Trujillo, Arequipa, Piura, Cuzco y Lima) y un software desarrollado en forma personal que envía notificaciones en forma programada desde uno hasta un millón de notificaciones de prueba. 4. Tamaño de la muestra Dos mil notificaciones provenientes de la red poco importantes con una notificación importante. 5. Variables Variable independiente: cantidad de notificaciones recibidas. Variables dependientes: El tiempo que demora el operador en realizar el diagnóstico de un conjunto fallos de red y el tiempo que demora el operador en mitigar ciertos problemas de inestabilidad en la red. 6. Selección de la Muestra Tamaño de los datos de muestra en 2000 notificaciones (entrada): 196KB. Las notificaciones son
--	--	---

			generadas por la red de prueba. 7. Técnica de Recolección de datos La técnica empleada para coleccionar los datos es la recolección de notificaciones proporcionadas por el protocolo SYSLOG. 8. Análisis e interpretación de la información • Identifica entre 2000 notificaciones una notificación importante y extrae conocimiento para la solución. • Se usa el tiempo promedio de identificación y planteamiento de solución manual ante una notificación importante entre dos mil notificaciones recibidas para un administrador poco experto: TPM. • Tiempo promedio de identificación y de solución manual ante un conjunto de notificaciones de intermitencia importantes entre dos mil notificaciones recibidas para un administrador poco experto: TPSM • Objetivos: reducir TPM y TPSM.
--	--	--	--