

# Propuesta de aplicación del Aprendizaje Simbólico y la Minería de Datos por Dominios en Sistemas de Supervisión Orientados a Evento (en Osmius)

En el creciente mundo de la interconectividad de usuarios y aplicaciones, cobra cada vez más significado la orientación a servicio y el acuerdo mediante contratos con usuarios y clientes sobre el rendimiento y la disponibilidad de los servicios ofrecidos/contratados. Diagnosticar antes y mejorar los problemas presentados en los sistemas que soportan estos servicios puede diferenciar a unos sistemas y proveedores de otros. Se propone en este artículo el uso y aplicación de la Inteligencia Artificial para la consecución de objetivos claros en un campo en el que puede concretarse con resultados medibles y beneficiosos.

La división por dominios de abstracción tanto en los algoritmos de correlación de eventos como en los actores implicados, el uso de código libre en una plataforma que aplica muchos de los paradigmas de la programación Orientada a Objeto (Osmius) como el concepto de corrientes de datos o "Streams" en el patrón "Pipes and Filters" y la aplicación de Minería de Datos a las bases de datos de eventos, son las principales propuestas que se exponen en el artículo. Se proponen además líneas de futuro ambiciosas en el ámbito de la predicción de sucesos y de capacidades de los recursos de las entidades supervisadas.

## Contenido

Introducción.....	3
Supervisión de Sistemas.....	4
Correlación de Eventos y Aprendizaje Automático.....	8
Introducción.....	8
Situación Actual.....	8
Minería de Datos y Eventos.....	11
Propuesta.....	11
Futuro.....	19
Conclusiones.....	20
Referencias.....	21

## Introducción

Las aplicaciones que son objeto de este artículo, se basan en el paradigma de de comunicación basada en eventos o, como también es conocido, publicación/subscripción de notificación por eventos. El rango de aplicaciones es extenso y va desde aplicaciones de escritorio, pasando por software de tiempo real, control de tráfico (incluyendo tráfico aéreo y ferroviario), hasta la gestión de transacciones en bolsa y el comercio electrónico.

La aceptación de este paradigma está ampliamente extendida como podemos comprobar en su incorporación a estándares como CORBA (Common Object Request Broker Architecture), SOA (Service Oriented Architecture) y JMS (API de servicios de mensajería de Java) y a sistemas comerciales como TIBCO. Un dominio muy importante de aplicaciones basadas en eventos en el de la Monitorización de Sistemas y Redes. Dentro de este dominio nos encontramos aplicaciones orientadas a la supervisión de sistemas de comunicaciones, servicios y sistemas informáticos, y los más orientados al mundo industrial como pueden ser los sistemas SCADA (Supervisory Control and Data Acquisition – Supervisión de Control y Adquisición de Datos).

Como ejemplo de aplicación práctica utilizaremos Osmius, un software para la recolección de eventos de sistemas distribuidos en red utilizado para la supervisión de sistemas. Osmius es software de código abierto y está basado en plataformas abiertas basadas en frameworks y patrones de diseño. En concreto utiliza el entorno de trabajo ADAPTIVE Communication Environment [ACE] que permite la reutilización de código para múltiples plataformas con un rendimiento excelente incluso en tiempo real, además de ser código abierto muy usado en la comunidad internacional académica y de mercado y fructífero en artículos de investigación.

Inicialmente la supervisión y monitorización estuvo orientada a los dispositivos concretos que formaban nuestra red. En los últimos años se observa un cambio de orientación hacia los Servicios que suministra el sistema monitorizado y que dependen de los dispositivos. La revolución que ha supuesto Internet y los altos niveles de conectividad necesarios para la comunicación constante entre usuarios y empresas, han endurecido las exigencias de rendimiento y disponibilidad de muchos proveedores que se plasman en contratos de Acuerdos de Nivel de Servicio (ANS) con penalizaciones en caso de no cumplimiento. En este entorno es fundamental contar con sistemas de monitorización preparados para el mantenimiento reactivo y proactivo de los sistemas y dispositivos y servicios.

Estos sistemas se basan en gran medida en su interacción con un humano experto, que es quien atiende los eventos y alarmas finales en una consola centralizada global u organizada por dominios. En ocasiones se generan gran cantidad de eventos que el administrador debe filtrar para así poder identificar la causa origen del problema cuanto antes, y ejecutar las acciones necesarias para restablecer el servicio o resolver la degradación del mismo.

Es en este contexto donde encaja perfectamente el aprendizaje automático: la mejora de rendimiento del proceso de identificación de causas reales.

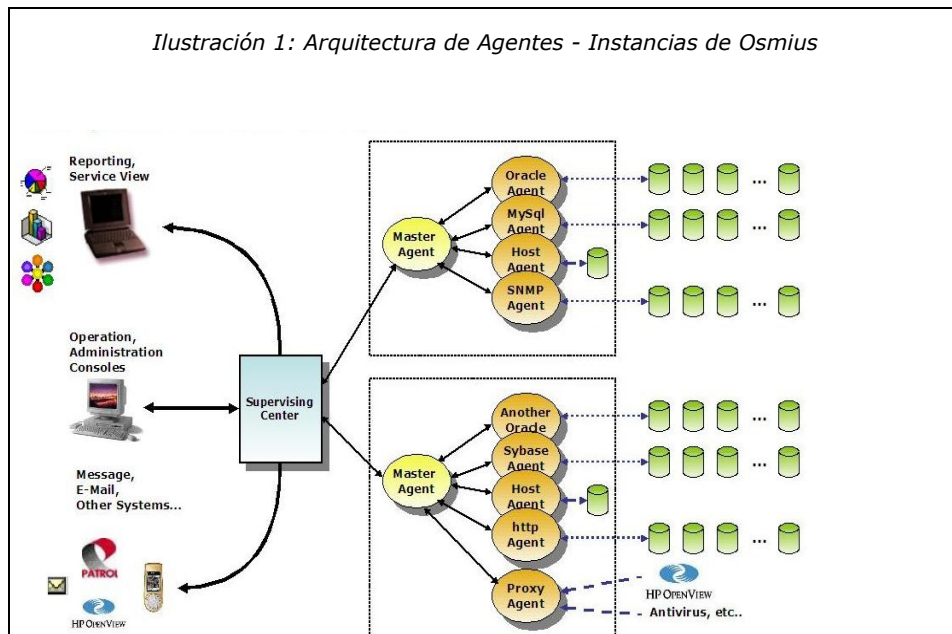
En el resto del artículo explicaremos en qué consiste la correlación de eventos y cuáles son las técnicas más utilizadas junto con sus principales ventajas y desventajas. Veremos que existen aplicaciones disponibles en el mercado junto con sus inconvenientes. Explicaremos los conceptos básicos dentro de la supervisión de sistemas orientados a servicio basándonos en la arquitectura software de Osmius, y propondremos un sistema para la mejora del proceso de identificación de causas reales y respuesta adecuada en sistemas de supervisión. Este sistema lo basaremos en la distinción de dominios de experiencia y en el uso de algoritmos predictivos y la minería de datos

## Supervisión de Sistemas

La supervisión de sistemas está cobrando un gran protagonismo ya que los servicios que se prestan ahora desde los Centros de Procesamiento de Datos (CPD), tienden a ser más exigentes desde el punto de vista de la disponibilidad, provocado sobretodo por los nuevos servicios a los usuarios de Internet como pueda ser el Comercio-Electrónico.

En este dominio de aplicaciones podemos hablar de una amplia aplicación del **Paradigma Gestor-Agente** en el que se distribuyen agentes en la infraestructura a monitorizar que se encargan de recolectar eventos y enviarlos a un gestor central para su presentación y/o procesamiento. Es verdad que dicha arquitectura puede presentar problemas de rendimiento en entornos con una gran cantidad de elementos y pueden presentarse cuellos de botella en el servidor central, como también es cierto que un buen diseño de la arquitectura de despliegue y del número y calidad de información de los eventos a recolectar puede paliar dicha problemática.

En algunos grupos [JPMARTIN2004\_01] se plantea distribuir la correlación de eventos y sus filtros entre los agentes de los sistemas dando lugar a lo que llaman **Sistemas Autogestionados**. De esta manera dejan a los sistemas la responsabilidad de filtrar lo que no consideren importante para el supervisor central. Los agentes sería más inteligentes pero más exigentes y complicados y, más intrusivos en el sistema a monitorizar.



Inicialmente los sistemas de supervisión estaban muy orientados al mundo de las comunicaciones y las redes, y tenían como principal protagonista a los dispositivos que conformaban la arquitectura. Este tipo de sistemas estaba orientado a usuarios muy técnicos con una formación muy específica para comprender la información presentada y actuar en consecuencia.

Hoy en día un sistema de monitorización debe estar enfocado a servicio en contraposición a aquellos orientados a dispositivo.

Osmius propone una arquitectura modular para poder encontrar un equilibrio entre ambos enfoques quedándose con lo mejor de cada uno.

Para Osmius cualquier posible origen de un evento recibe el nombre de **Instancia**, sea ésta un dispositivo hardware final, un servidor, una base de datos, un servicio ftp o un sensor de presión o temperatura. Cada tipo de instancia tiene asociado un tipo de agente, que es el que sabe cómo preguntar por eventos a sus instancias y se integra perfectamente en el resto de la arquitectura Osmius para enviar los eventos y recibir órdenes e indicaciones sobre los eventos. Los agentes están integrados en la arquitectura a través de **Agentes Maestros** que además se encargan de recoger y enviar el **Servidor Central** los eventos y de recibir comandos y pasarlos a cada uno de los agentes.

De esta manera se consigue independizar la arquitectura del sistema de monitorización de las instancias a monitorizar. Los agentes son poco intrusivos y al tener un interfaz muy definido es muy fácil desarrollar e incorporar nuevos tipos de instancias a monitorizar.

Hay quienes distinguen entre monitorización y correlación de eventos orientadas a dispositivo y orientadas a servicio [AHANEMANN2004\_01]. Este artículo propone un sistema capaz de abstraerse de dicha distinción para poder reutilizar el aprendizaje y el motor de correlación. Instancia o Servicio son sólo dos tipos de orígenes de los eventos.

**Instancia:**

Origen de un evento. Unidad fundamental sobre la que está construida un sistema que deseamos supervisar.

**Servicio:**

Conjunto de funcionalidades ofrecidas a un cliente o usuario mediante una funcionalidad pactada. Puede verse como una agrupación de instancias y aplicaciones.

**Proceso de Negocio:**

Wikipedia: Conjunto de tareas relacionadas lógicamente llevadas a cabo para lograr un resultado de negocio definido. En el contexto de la monitorización supone una agrupación de servicios y disponibilidades necesarias para dicho proceso .

**Acuerdo de Nivel de Servicio (ANS):**

Contrato con el cliente o usuario respecto al rendimiento y/o disponibilidad de una aplicación, servicio o proceso de negocio.

En Osmius toda la organización lógica reside en el **modelo de datos** del Servidor Central lo que nos permite que la orientación más técnica – Dispositivo o Instancia – o más orientada a Usuario – Servicio o Proceso de Negocio – sea una cuestión de presentación según el perfil del usuario conectado.

En la Tabla-1 se muestra el tipo de vista según el tipo de usuario. Los intereses sobre la información a mostrar no son los mismos en el caso de un administrador de red interesado en un parámetro específico de un modelo, que en el caso de un directivo que sólo desea saber si el proceso de facturación funciona como es debido o no.

<b>Perfil de Usuario</b>	<b>Elemento de Interés</b>
Administrador de Red	Instancia: Interfaz, Router, Servidor...
Administrador de Base de Datos	Instancia: BD producción, BD desarrollo,...
Usuario de Aplicación	Aplicación: Intranet (BD + Servidor Web + ...)
Cliente	Servicio: Aplicación o conjunto de Aplicaciones
Dirección	Proceso de Negocio: Conjunto de Servicios

Osmius opta por una arquitectura Gestor-Agente con agentes ligeros, y por agrupaciones lógicas por perfil para cumplir con las expectativas de orientación técnica para los usuarios del sistema y de orientación a negocio para los clientes o dirigentes de negocio.

El tipo de usuario condiciona también el tipo de perspectiva ante la aparición de un problema.

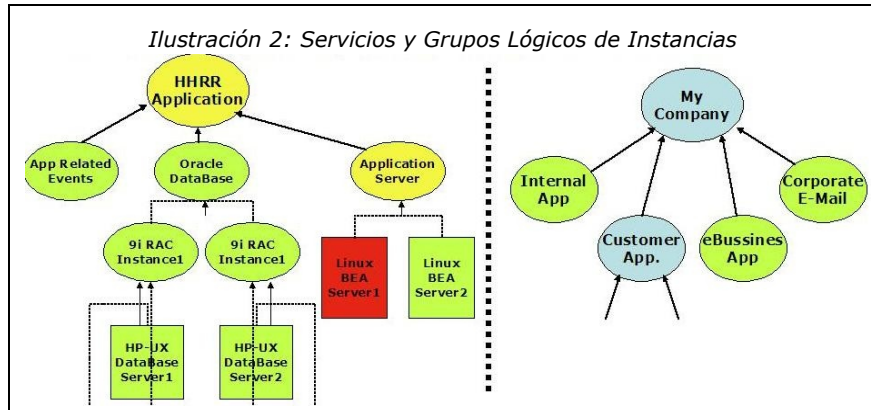
- Top-Down : (De arriba a abajo)

Falla un servicio. ¿Por qué está fallando? ¿Cuál es el elemento que está causando la caída o degradación?

- Botton-Up : (De abajo a arriba)

Falla una instancia. ¿Qué servicios se ven afectados y en qué medida? ¿Qué clientes/usuarios sufren las consecuencias ?

■



Si añadimos los contratos que suponen los ANS con los usuarios de los sistemas objeto, cobra todo su sentido el uso de sistemas de monitorización ágiles, capaces de mostrar diferentes perspectivas del modelo y que sean capaces de reducir el tiempo de identificación de causas reales a problemas ayudándose de la correlación de eventos y de técnicas de aprendizaje automático adecuadas.

<b><i>Tipos de Dispositivos o Instancias</i></b>
Elemento de Red: Router, FireWall, Switch,...
Servidor: Linux, Windows, Solaris, ...
Base de Datos: Oracle, MySql, Sybase,...
Servicio Internet: http, ftp, DNS, ...
Aplicación: CRM, Serv. Web, a medida...
Industria: Sensor Temperatura, Vávula P
Domótica: Interruptores, Electrodomésticos,.
Otros.....

# Correlación de Eventos y Aprendizaje Automático

## *Introducción*

La correlación de eventos es un proceso automatizado que permite a un operador, administrador o usuario de un sistema orientado a evento, encontrar entre muchos eventos aquéllos que son realmente críticos en el ámbito buscado.

Debe ser un proceso automatizado dada la gran cantidad de información que se nos puede presentar. La correlación de eventos permite reducir grandes cantidades de eventos a conjuntos más reducidos y con más significado sobre su causa y – por tanto – su posible solución.

Este artículo propone utilizar las mismas técnicas de correlación para eventos orientados a dispositivo y eventos orientados a servicio al contrario de lo propuesto por otros autores. Se pretende aplicar los mismos mecanismos de correlación a los diferentes dominios formados por agrupaciones con cada vez mayor nivel de abstracción.

De esta forma cada experto o usuario puede validar las reglas de correlación de su campo de especialización y conseguimos una escalabilidad lineal del rendimiento, además de la reusabilidad obvia de los algoritmos de aprendizaje y minería de datos. Mismos algoritmos; diferentes dominios. Sigue siendo importante que los algoritmos de correlación mantengan una complejidad baja ( $O(n)$  o  $O(\log(n))$ ), dado el gran número de reglas que podemos llegar a manejar y tener que aplicar a cantidades ingentes de eventos.

Lograr los objetivos de correlación inteligente que ayude a detectar antes y mejor los problemas surgidos en la red supervisada, cobra una importancia fundamental en el actual mundo de sistemas interconectados, sean de la naturaleza que sean, y con cada vez mayores exigencias de disponibilidad y rendimiento.

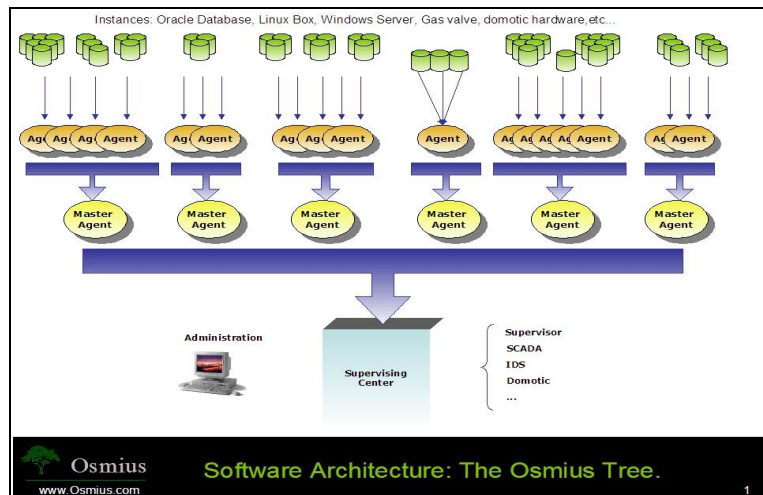
## *Situación Actual*

Existen diferentes estrategias de correlación en función del **lugar dentro de la arquitectura** en la que se lleva a cabo esta función.

- Central. En el servidor central al que llegan todos los eventos.
- Agentes Locales.
- En el propio elemento de Red.

El tomar una estrategia más centralizada o distribuida tiene ventajas e inconvenientes, y en este artículo se propone un acercamiento mixto y configurable en tiempo de ejecución.

<b>Estrategia</b>	<b>Ventajas</b>	<b>Desventaja</b>
Centralizada	Impacto controlado en los servicios monitorizados. Agentes más sencillos.	Exceso de carga en servidor central.
Distribuida	Menor flujo de eventos hacia el servidor central.	Agentes más complicados. Impacto en el rendimiento del elemento monitorizado.
Autogestión	Las de un sistema distribuido en extremo.	Poca abstracción del elemento final y poca reusabilidad. Impacto en el elemento. Poca transparencia.



Las técnicas utilizadas para llevar a cabo la correlación de eventos son variadas:

- Máquinas de Estado Finito.
- Razonamiento Basado en Reglas.
- Razonamiento Basado en Casos.
- Razonamiento Basado en Modelos.
- Redes Bayesianas.
- Redes Neuronales.

Podemos hablar de dos grandes grupos de acercamientos al problema de la correlación.

Por un lado aquéllos que necesitan de información sobre la arquitectura del sistema completo, en la forma de un diagrama de componentes y sus relaciones que luego puede ser traducido a una matriz de correlación. Este diagrama puede ser suministrado directamente al sistema de correlación, o bien, se pueden utilizar técnicas de aprendizaje que lo infieren de los datos recogidos.

Por otro lado están los sistemas que no necesitan previamente información sobre los componentes y sus relaciones. Normalmente están basados en reglas

de la forma "SI ocurre esto ENTONCES haz lo otro". Reciben los eventos y aplican dichas reglas, aunque también pueden aprender de los datos para inferir nuevas reglas.

Pasemos a describir algunas de las más utilizadas.

### **Razonamiento Basado en Modelos** (Model Based Reasoning – MBR):

Cada componente (instancia según la nomenclatura Osmius) se modela respecto a sus atributos, comportamiento y relación con otros modelos y componentes.

La correlación surge como el resultado de la colaboración entre modelos.

### **Razonamiento Basado en Reglas** (Rule Based Reasoning – RBR):

Se usa un conjunto de reglas que se aplican a los eventos recibidos – normalmente en una ventana determinada de tiempo – para correlacionar los eventos.

El algoritmo de RETE, diseñado por Charles L. Forgy de la Universidad de Carnegie Mellon, es uno de los más eficientes en búsquedas de patrones para implementar sistemas expertos basados en reglas.

En la práctica los conjuntos de reglas a aplicar pueden llegar a ser muy grandes y hacen difícil el mantenimiento del sistema.

Además si ocurre algo no previsto por el conjunto de reglas suministrado el sistema falla en su cometido. Muchos sistemas RBR se basan el reconocimiento de expresiones regulares [RVAARANDI2002\_01].

### **Razonamiento Libro de Códigos** (CodeBook Approach):

Por libro de códigos se entiende un libro de recetas en dónde se detalla paso a paso como seguir un procedimiento.

Al igual que RBR se basa en un algoritmo de correlación. Utiliza como entrada un gráfico de dependencias con eventos y causas raíz como nodos, y flechas dirigidas que los interconectan. Una vez construido en gráfico puede optimizarse eliminando información redundante o sin añadido, y se transforma en una matriz de correlación.

La matriz tiene como columnas las causas raíces y como filas los eventos. En la forma más simple cada celda tiene un valor binario (correlaciona o no correlaciona).

Permite manejarse en situaciones con combinaciones desconocidas de eventos. RBR es más flexible.

### **Razonamiento Basado en Casos** (Case Based Reasoning – CBR):

No necesita conocimiento previo sobre la infraestructura. Contamos con una base de datos de casos que han ocurrido antes, junto con las causas

raíces asociadas.

Podemos aplicar aprendizaje sobre los casos ayudando a identificar causas reales.

## **Minería de Datos y Eventos**

Todos las estrategias comentadas pueden complementarse con técnicas de **Minería de Datos**, para identificar reglas, gráficos de dependencia o conclusiones sobre el sistema supervisado.

[GUPTA] utiliza algoritmos para **Data-Mining** que usan datos sobre rendimiento para obtener dependencias entre componentes basadas en probabilidad.

De esta forma podríamos generar el gráfico de dependencias tratando un conjunto de datos en lugar de hacerlo a mano. Con dicho gráfico se pueden encontrar causas raíces de problemas concretos.

[ENSEL] sugiere utilizar redes neuronales para generar automáticamente gráficos de dependencia dinámica entre servidores, aunque no hay datos experimentales sobre la precisión de estos métodos. Además este acercamiento no detecta causalidad, sólo correlación.

**La Minería de Datos** utiliza una mezcla de técnicas estadísticas y de manejo de datos, que nos proporciona una forma de agrupar los datos de los eventos recibidos, para encontrar combinaciones interesantes [JLHELLERSTEIN1999\_01].

De esta forma se pueden identificar patrones para luego generar reglas de correlación.

En el caso de los eventos es muy importante la variable tiempo, ya que se debe tener en cuenta la secuencia de los eventos más que su sola ocurrencia

## **Propuesta**

Este artículo propone aplicar técnicas mixtas de aprendizaje automático y minería de datos sobre eventos de un sistema de supervisión de sistemas llamado Osmius.

Osmius implementa muchos de los paradigmas investigados actualmente en las ciencias de la computación, y siendo su código de libre acceso y modificación permite su uso intensivo en la investigación dentro de la Inteligencia Artificial como en otros campos relativos a la Tecnología y la Información.

El campo de las aplicaciones de supervisión de sistemas en red tiene un interés manifiesto dentro del creciente marco actual de proveedor de servicio – cliente y ANS. La pronta identificación de problemas raíces cobra una importancia capital y, la Inteligencia Artificial encaja como metodología y en este campo cuenta con indicadores sencillos y objetivos para medir sus resultados.

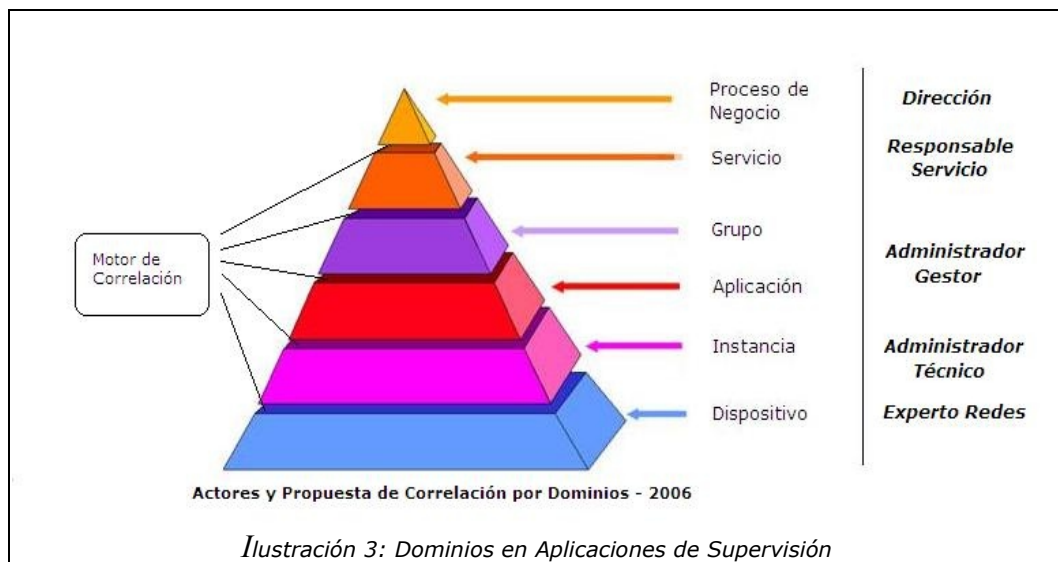
Parece además éste un campo que promete resultados para la Inteligencia

Artificial, ya que el problema tantas veces mencionado de la falta de datos, en este caso, no es tal. Es relativamente fácil hacerse con una buena base de datos de eventos de un sistema de monitorización ya implantado y aplicar algoritmos de aprendizaje o técnicas de minería de datos.

Las principales ideas a aplicar en la propuesta son:

- **Dominios de Correlación:**

Se propone dividir los eventos por dominios de correlación, de forma que las reglas a aplicar y el usuario o sistema supervisor puedan variar sin influir un conjunto en el otro. Es diferente la visión de un cliente ante un problema que la de un administrador de red. Esta división en dominios permite un acercamiento y reducción de eventos progresivo según se avanza en capacidad de abstracción, y la reutilización de algoritmos.



- **Motor Basado en Reglas Abierto:**

Un motor básico y ligero basado en atributos genéricos de cualquier sistema de eventos. Estándares, estructura de los eventos, y código y algoritmos abiertos, nos permitirán utilizar el motor de forma centralizada sólo, distribuida pura, o en un modo mixto, y en función del tipo de sistema supervisado adaptarse por parametrización.

Puede descargarse el código y su documentación en la web de soporte a desarrollos basados en software libre de Source-Forge en:

<http://osmius.sourceforge.net>

- **Osmius - Dominio productivo para la Investigación:**

Osmius seguirá su andadura orientado al mercado práctico y económico de los sistemas de supervisión. Estando "bien hecho" y al ser un sistema plenamente modular permitirá usar directamente los resultados de tesis e

investigaciones, si están bien enfocadas, hacia una utilidad final para determinados usuarios y/o mercados. Si se cumplen estas condiciones puede ser el entorno ideal para la implementación y observación en entornos reales de pensamientos y técnicas novedosas, que en otros contextos es tan difícil y frustrante, respecto a salidas de las líneas investigadas por una universidad o centro de investigación.

- Reglas de Correlación Dinámicas – Metodología de Minería de Datos:  
Se propone la existencia a priori de reglas cuya utilidad ya se ha probado en los sistemas de monitorización, y la utilización de la minería de datos junto con algoritmos de aprendizaje, para presentar nuevas reglas a cada uno de los expertos para su aprobación y puesta en marcha.
- La Minería de Datos tanto automática como asistida, se propone como principal herramienta en un caso, metodología y consultoría en el otro, para lograr como resultado, nuevas reglas con las que alimentar el motor de correlación o informes y patrones que utilizar para posibles mejoras tanto del software como de la gestión de recursos y humana de los sistemas supervisados.

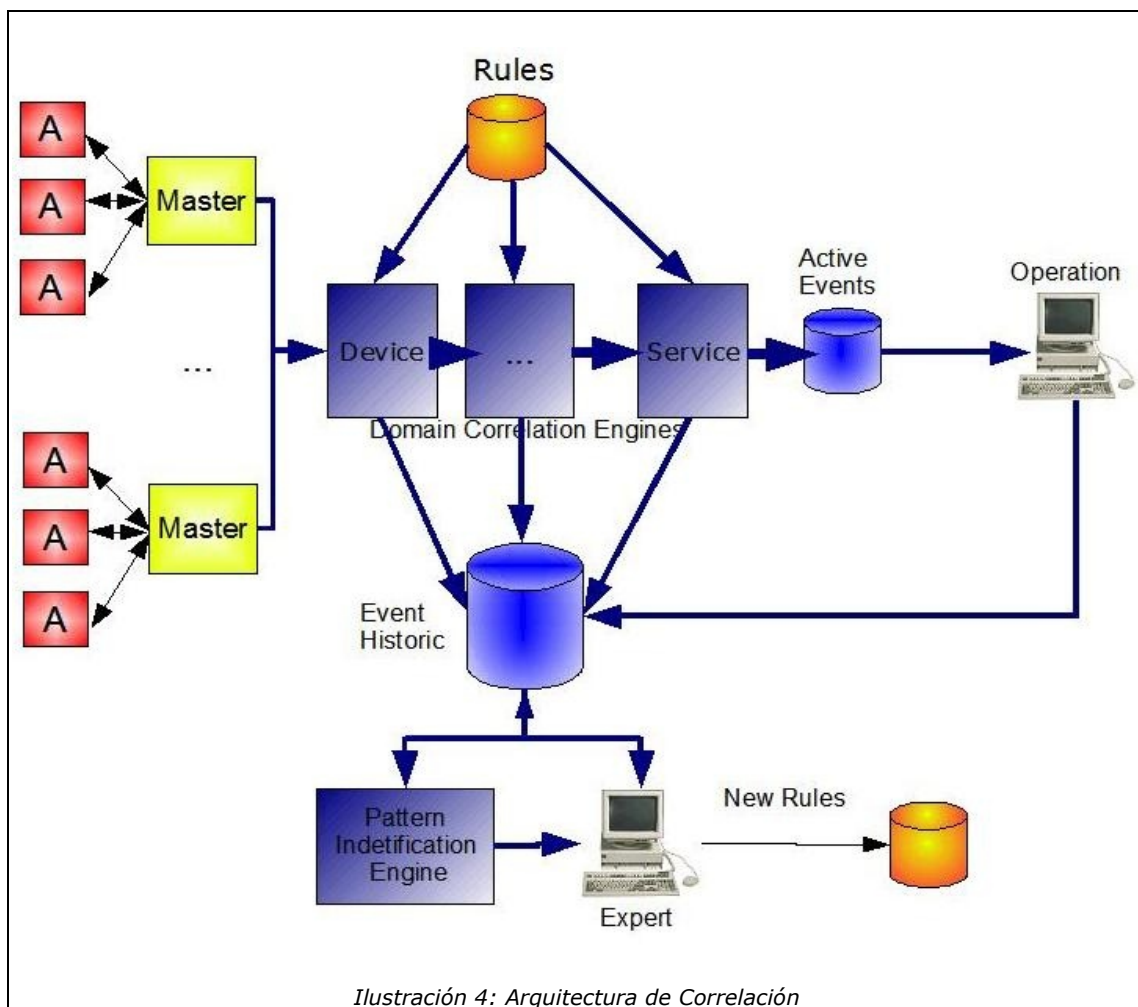
Para definir el problema en el campo del aprendizaje automático necesitamos definir el problema. En este caso es fundamental la relación de atributos de los que consta cada evento en Osmius. Su adecuada descripción y comprensión ayudan a plantear las bases del problema a resolver por los algoritmos de aprendizaje automático y las técnicas de Data-Mining.

<b>Atributo</b>	<b>Descripción</b>
TYP_MESSAGE	Tipo del Mensaje o Evento. Normal – Interno – Error
COD_MASTER	Agente Maestro del que proviene. Informa sobre la arquitectura del sistema de supervisión más que sobre el sistema monitorizado.
COD_AGENT	Agente que origina el mensaje. Idem que el campo anterior.
COD_MESSAGE	Identificador del tipo evento que origina el mensaje (Porcentaje de Disco ocupado, segundos en responder a petición http, temperatura del núcleo,...)
DAT_INIEVENT	Fecha y hora con precisión de microsegundos en que se comienza la pregunta del evento.
DAT_FINEVENT	Fecha y hora con precisión de microsegundos en que se recibe respuesta a la pregunta del evento (podemos identificar preguntas o eventos que tardan "demasiado").
COD_INSTANCE	Identificador único de la instancia o dispositivo de dónde proviene el mensaje (Nombre de la base de datos, número de serie de la válvula,...).
TYP_INSTANCE	Tipo de la Instancia (Una base de datos, una válvula, un tostador, ...).
VALUE	Valor numérico de evento (En osmius se debe reducir toda respuesta a un número entero)

Según los Agentes Maestros vayan enviando los mensajes (eventos) producidos por los agentes que están a su cargo monitorizando las diferentes instancias, se introducirán en la cadena de correlación. En dicha cadena tenemos los diferentes motores en una configuración basada en el patrón de diseño "Pipes and Filters". Ver *Ilustración 4 – Arquitectura de Correlación*.

Cada uno de los procesos es en realidad clónico de otro de tal forma que sólo varían las reglas de correlación a aplicar. Además, y en tiempo de ejecución, podremos activar o parar cada uno de los motores consiguiendo mejorar la adaptabilidad a los diferentes sistemas que pueden ser supervisados por la plataforma Osmius. Si uno de los procesos encuentra posibilidad de correlación en sus datos de entrada, la aplica pasando la información redundante al histórico de eventos recibidos, y el resultado de la correlación al siguiente proceso en la cadena o, si es el último, a la base de datos de eventos activos para que sean los operadores o administradores del sistema los que se encargan de su análisis y paso manual, si cabe, al histórico de eventos.

En el histórico de eventos se encuentra el punto de información en el que se basará toda la problemática de análisis y minería de datos con el objetivo de identificar patrones de ocurrencia y generación de nuevas reglas para los diferentes dominios.



Un motor propondrá a los expertos reglas basadas en los patrones descubiertos, que organizarán la información para poder llegar a más conclusiones y serán los encargados finales de aceptar nuevas reglas e introducirlas en la base de datos de reglas, para que puedan ser aplicadas por

los motores de correlación por dominios.

El ciclo de vida de un mensaje o evento es el siguiente:

- Un agente con capacidades para recuperar datos de un tipo determinado de instancias (TYP\_INSTANCE) lee de un fichero de configuración.
- Recupera un nombre de instancia en concreto (COD\_INSTANCE).
- Para esa instancia lee un tipo específico de evento o mensaje a monitorizar (COD\_MESSAGE).
- El agente inicia la acción para recuperar un valor asociado (VALUE) al tipo de evento o mensaje y un texto explicativo (TXT\_VALUE) en un momento determinado (DAT\_FINEVENT).
- El agente compone un mensaje con los campos apropiados y los envía al Agente Maestro.
- El Agente Maestro envía el mensaje al Centro de Supervisión.
- El Centro de Supervisión inserta el mensaje en la tabla de **Mensajes Activos**.
- El mensaje pasará a la tabla de **Histórico de Mensajes** por dos vía principales:
  - Uno de los motores de correlación lo elimina de Activos junto con otros mensajes o no, y lo pasa al Histórico al aplicar determinadas reglas.
  - Uno de los usuarios con permisos sobre el mensaje lo "reconoce" y de esta forma es borrado del Activo y pasa al Histórico.

Los mensajes al ser pasados el histórico almacenan la fecha y la hora de dicho paso para poder analizar también las posible correlación entre eventos según su fecha de reconocimiento además de la fecha de ocurrencia del evento.

Dentro de los atributos asociados a cada mensaje hay algunos que no resultan relevantes para la correlación de los eventos como pueda ser el código del agente maestro que hace de *proxy* para que el mensaje llegue adecuadamente.

Los atributos relevantes de cada mensaje son pues:

<b>Atributo</b>	<b>Ejemplo /Formato</b>
Tipo de Instancia	Base de datos Oracle [ORA01] , Base de Datos MySql [MYSQL], Servidor FTP [FTP01], Calefacción Domótica [SYM01], Termómetro industrial [TER01], etc...
Código de Instancia	INST01, INST01, ...
Código de Mensaje	Para una base de datos Oracle: Número de usuarios conectados [NUMUSU] Porcentaje de Procesos máximos utilizado [NUMPRC] Número de ficheros de "redo log" que faltan por archivar [REDLOG] Porcentaje de ocupación de "Tablespace" más utilizado [FRETBS] Ratio de acierto en el buffer de caché de datos [BUFRAT] etc
Fecha del Evento	20060707145959 / YYYYMMDDHHMISS
Valor del Evento	Numérico
Texto del Evento	Este campo es sólo descriptivo y será útil en un análisis detallado de los mensajes recibidos y para comprobar la utilidad de una correlación y hacer más entendible el contexto en el que se producen los mensajes. "El Tablespace mas ocupado es [RBS1] con:82 por ciento.[90,95]"

En la tabla de datos adjunto podemos ver un conjunto de datos reales extraídos a primera hora de la jornada laboral de un entorno con más de 40 instancias de bases de datos Oracle dando servicios productivos. Este sería el tipo de datos a analizar con los motores de correlación y algoritmos de aprendizaje automáticos adecuados.

En trabajo a realizar consiste en este caso en:

- Determinar periodos de tiempo en los que suben los valores de determinados tipos de mensaje para todas las instancias en general. Por ejemplo el número de usuarios conectados a primera hora de la mañana o en las horas asignadas para el almuerzo. Sube la carga de la base de datos por las noches al lanzarse los trabajos de procesamiento por lotes y las copias de seguridad.
- Determinar si para una misma instancia hay tendencias de los valores de diferentes tipos de mensaje que correlacionan. Por ejemplo al subir o bajar el número de usuarios también lo hace el porcentaje de procesos utilizados.
- Determinar si hay valores de tipos de eventos o mensajes en diferentes instancias que sólo se dan juntos o correlacionan fuertemente. Por ejemplo siempre que pierdo la conexión a A (CONNECT = 1) también la pierdo con B porque están en el mismo servidor y segmento de red.
- Analizar si valores altos relativos en un parámetro durante determinada ventana de tiempo provoca que otros parámetros también suban o bajen su valor. Por ejemplo si el porcentaje de procesos utilizados se mantiene alto finalmente baja la tasa de acierto en caché.

Todos estas situaciones pueden analizarse al margen de los expertos en los tipos de instancia y parámetros observados, y posteriormente se deberían interpretar en común para darle el sentido técnico final en caso de que lo hubiera.

DAT_FINEVENT	TYP_INSTANCE	COD_INSTANCE	COD_MESSAGE	VALUE	TXT_VALUE
20060626083835	ORACLE01	CAMBP	CONNECT1	0	Conexión correcta
			NUMUSU01	5	Cantidad de Usuarios Actual.
			NUMPRC01	10	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
	GASBP	CAMBP	CONNECT1	0	Conexión correcta
			NUMUSU01	20	Cantidad de Usuarios Actual.
			NUMPRC01	12	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
	HTBP	CAMBP	CONNECT1	0	Conexión correcta
			NUMUSU01	25	Cantidad de Usuarios Actual.
			NUMPRC01	41	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
ATBD	CAMBP	FREOBJ	0	Objetos sin capacidad para extenderse.	
		BUFRAT	86	Porcentaje de Acierto Cache de Datos.	
		CONNECT1	0	Conexión correcta	
		NUMUSU01	30	Cantidad de Usuarios Actual.	
20060626084439	ORACLE01	CAMBP	NUMPRC01	44	El porcentaje de Procesos usados.
			CONNECT1	0	Conexión correcta
			NUMPRC01	10	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
	GASBP	CAMBP	NUMLCK	0	Número de Bloqueos entre procesos.
			CONNECT1	0	Conexión correcta
			NUMLCK	0	Número de Bloqueos entre procesos.
			NUMPRC01	13	El porcentaje de Procesos usados.
	HTBP	CAMBP	REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
			CONNECT1	0	Conexión correcta
			NUMLCK	0	Número de Bloqueos entre procesos.
			NUMPRC01	40	El porcentaje de Procesos usados.
ATBD	CAMBP	REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]	
		FREOBJ	0	Objetos sin capacidad para extenderse.	
		BUFRAT	86	Porcentaje de Acierto Cache de Datos.	
		CONNECT1	0	Conexión correcta	
20060626090323	ORACLE01	CAMBP	NUMUSU01	29	Cantidad de Usuarios Actual.
			NUMPRC01	43	El porcentaje de Procesos usados.
			CONNECT1	0	Conexión correcta
			NUMUSU01	5	Cantidad de Usuarios Actual.
	GASBP	CAMBP	NUMPRC01	10	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
			CONNECT1	0	Conexión correcta
			NUMUSU01	19	Cantidad de Usuarios Actual.
	HTBP	CAMBP	NUMPRC01	12	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
			CONNECT1	0	Conexión correcta
			NUMUSU01	23	Cantidad de Usuarios Actual.
ATBD	CAMBP	NUMPRC01	39	El porcentaje de Procesos usados.	
		REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]	
		FREOBJ	0	Objetos sin capacidad para extenderse.	
		BUFRAT	86	Porcentaje de Acierto Cache de Datos.	
20060626094133	ORACLE01	CAMBP	CONNECT1	0	Conexión correcta
			NUMUSU01	5	Cantidad de Usuarios Actual.
			NUMPRC01	10	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
	GASBP	CAMBP	CONNECT1	0	Conexión correcta
			NUMUSU01	24	Cantidad de Usuarios Actual.
			NUMPRC01	14	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
	HTBP	CAMBP	CONNECT1	0	Conexión correcta
			NUMUSU01	25	Cantidad de Usuarios Actual.
			NUMPRC01	41	El porcentaje de Procesos usados.
			REDLOG	1	Ficheros de Redo que faltan por Archivar:[1] [7,10]
ATBD	CAMBP	FREOBJ	0	Objetos sin capacidad para extenderse.	
		BUFRAT	86	Porcentaje de Acierto Cache de Datos.	
		CONNECT1	0	Conexión correcta	
		NUMUSU01	33	Cantidad de Usuarios Actual.	
ATBD	CAMBP	NUMPRC01	47	El porcentaje de Procesos usados.	

Es la minería de datos aplicada a la generación de reglas por dominio, junto con el uso de patrones de diseño de la arquitectura software, la que dotan al modelo de una gran adaptabilidad y flexibilidad para alcanzar el objetivo de mejorar la identificación de problemas y la respuesta para solucionarlos.

### **Minería de datos en el contexto de las Bases de Datos de Eventos.**

La minería de datos se propone inicialmente para su uso sobre los datos del carrito de la compra en supermercados. En objetivo era responder a preguntas del tipo: ¿La ocurrencia de **compra A** indica que **compra B** también? (Regla de Asociación)

En este caso el algoritmo *a priori* propuesto por Agrawal proporciona buenos resultados en este tipo de situaciones y sobretodo puede darnos la medida, basándose en parámetros llamados *soporte y confianza* de una reglas, de lo buena o fuerte que es la asociación entre dos eventos.

[Hellerstein] ya avanza resultados de aplicar la minería de datos a sistemas de monitorización basados en eventos, en este caso de sistemas de comunicaciones. Estos resultados ponen de manifiesto patrones, entendidos como formas de comportamiento y agrupación de los datos, observados en la mayoría de sistemas.

Podemos explicar la similitud entre el problema de "la cesta de la compra" y los grupos de eventos, agrupando éstos dentro de una ventana de tiempo y formado de esta forma las cestas a analizar. Es muy importante también seleccionar los atributos por los que agrupar los diferentes eventos.

Algunos de los patrones que se repiten:

- **Patrón de tormenta de eventos:**

Se produce cuando falla algo importante para una parte o todo el conjunto del sistema, y no cuenta con la deseada (en principio) redundancia. Un ejemplo claro es el fallo del servidor de nombres en una red: Ninguna de las preguntas de los servidores sobre ellos mismos u otros de su red podrá ser respondida y se generan una gran cantidad de eventos de fallo en la resolución de nombres y direcciones. También puede seguirse un patrón en cascada en el que el fallo de un elemento se va propagando a otros elementos al estilo de una reacción en cadena que provoca la tormenta de eventos.

Mediante la Minería de Datos podemos buscar:

Periodos en los que hay más tormentas que un determinado umbral (recordemos las cestas)

Estudiar a fondo y reconocer patrones durante esos periodos.

- **Periodicidades:**

Ocurrencias repetidas del mismo evento o conjunto de eventos. Según [Hellerstein], este tipo de repeticiones es muy común en los sistemas de supervisión de red y se dan en un 50% o 60% de los eventos. Dicho de otra forma, hasta el 60% de los eventos pueden asociarse con un patrón

repetitivo con determinada periodicidad. Interesante y lógico, si los eventos de por sí se están monitorizando en periodos fijos, como suele ser el caso.

En este caso pueden entrar en juego tanto los algoritmos de asociación de reglas que tienen en cuenta sólo la dependencia con respecto a la frecuencia (en este caso de aparición), como otros modelos que tengan también en cuenta la variable tiempo.

- **Dependencias mutuas:**

Grupos de eventos que tienden a ocurrir juntos en lugar de hacerlo de forma separada e independiente.

Con esto queda patente que la minería de datos, con mayor o menor grado de asistencia – **cosultoría experta** versus **algoritmos automáticos** de reconocimiento de patrones – permite llegar a resultados en cuanto a alcanzar objetivos de mejorar el rendimiento de los sistemas de supervisión.

Estos algoritmos automáticos se orientarán en Osmius a la generación de reglas que bajo supervisión “experta” en cada dominio, se incorporarán o no al conjunto de reglas iniciales y previas, del motor de correlación correspondiente. *Ver Ilustración 1.*

## Futuro

No cabe duda de que el interés en sistemas capaces de supervisar aplicaciones o procesos de negocio de manera que se mejore el servicio prestado seguirá en auge en los próximos años.

Según vaya madurando la implantación de sistemas mixtos y con resultados patentes como se propone en este artículo, se abordarán nuevos retos algunos de los cuales se trata de avanzar es este apartado.

En este área cobrarán mayor protagonismo e interés los **Algoritmos Predictivos**. La propuesta en este artículo es aplicar este tipo de algoritmos para detectar problemas en un servicio, y actuar corrigiéndolos antes incluso de que afecten a los contratos firmados con los clientes o usuarios de los sistemas.

Estos algoritmos deberán ser capaces de reponder de manera proactiva no sólo ante problemas si no a previsiones de capacidad de rendimiento, o de cualquier tipo de recurso.

Preguntas como:

- ¿Cuál será el uso de CPU o disco de este servidor durante el próximo mes?
- ¿Y durante el próximo minuto?
- ¿Puedo predecir si este router va a fallar en la próxima media hora?

Y en general del tipo:

- ¿Cuál será la evolución de la variable X en el recurso Y?

, serán objetivo muy interesante de los sistemas de supervisión en los que

tendrá aplicación la investigación sobre aprendizaje automático y minería de datos en sistemas orientados a evento.

Los nuevos sistemas y algoritmos podrán utilizar los datos de cada uno de los tipos de evento (variables) recibidos de las diferentes instancias para realizar las predicciones, y utilizar las posibles correlaciones con otras variables para variar las estimaciones.

Deberemos utilizar algoritmos de Análisis de Series Temporales para realizar predicciones a corto o largo plazo. En este caso parecen encajar los algoritmos de asociación que tienen en cuenta la variable tiempo como puedan ser los *Modelos Ocultos de Markov*.

No parece descabellado que según se vayan programando y usando algoritmos para estos tipos de aprendizaje, aparezcan **patrones de diseño** específicos en el aprendizaje automático.

También será apasionante la identificación de **patrones de metodología de trabajo** y su posterior documentación que surgirán de la experiencia del análisis de las bases de datos de eventos para responder a las preguntas antes citadas, y que servirán a otros expertos en minería de datos. A este respecto podrá documentarse y modelarse una **metodología** para las diferentes fases de la implantación de la supervisión de un sistema cualquiera que sea su naturaleza. Estarían las fases de definición de requisitos tanto técnicos y como de negocio y ANS, la definición de los requisitos de no-funcionales de calidad tanto del servicio como de la monitorización, así como las fases que definen el comportamiento del sistema y su refinamiento respecto a resultados, generación de nuevas reglas y presentación de informes a los diferentes agentes dentro de cada dominio. En la fase de refinamiento cobrará sentido una **metodología específica para el aprendizaje automático** en el contexto de asociación de eventos y generación de reglas.

## Conclusiones

Hemos visto que los sistemas que ayudan a supervisar y monitorizar otros sistemas en red toman una gran importancia para ayudar a estos últimos a cumplir con una calidad de servicio cada vez más exigente y que se plasma en contratos con los usuarios y/o clientes a través de los Acuerdos de Nivel de Servicio.

Mejorar la productividad de los sistemas y de los grupos de operación encargados de supervisarlos es importante, y aunque dicha productividad ha mejorado sigue siendo problemático qué reglas de correlación escribir y aplicar en cada entorno.

Además, tener en cuenta que dentro de cada sistema existen diferentes visiones, lejos de complicar el problema, ayuda al facilitar su comprensión, y por tanto en el diseño de las posibles soluciones. No podemos tratar de igual forma las necesidades de información en la supervisión de un sistema de un usuario técnico muy especializado en los dispositivos finales, que la visión del cliente preocupado por el rendimiento de sus transacciones de comercio

electrónico, pongamos por caso.

Distinguiendo las diferentes formas de ver un mismo problema, podrá optarse por las ventajas de cada una, y dejaremos que sea el sistema el que pueda definir la estrategia que mejor se adapta a sus características de rendimiento, número de eventos y necesidad de información en el recolector central.

Las herramientas comerciales para la correlación de eventos de las que se tiene noticia antes de escribir el artículo presentan ciertos problemas como son:

- La **elevada complejidad** de configuración, puesta en marcha y mantenimiento. Esto impide la división en dominios por capacidad de abstracción y al acceso a usuarios con conocimientos útiles sobre su dominio pero no excesivamente técnicos.
- Suelen ser **dependientes de plataforma**. Los motores están desarrollados para sólo ser usados con el formato propietario de la herramienta de supervisión. Además, como usuario, sólo dispones de los binarios compilados para un número reducido de plataformas.
- **Caras**.

Una buena herramienta de Código Abierto capaz de correlacionar eventos en diferentes formatos y lo suficientemente robusta para funcionar en entornos productivos exigentes, cubre un hueco evidente en el mercado académico y comercial actual.

## Referencias

### [DCSCHMIDT1993\_01]

The ADAPTIVE Communication Environment An Object-Oriented Network Programming Toolkit for Developing Communication Software.

**Douglas C. Schmidt** - 1993  
**11<sup>th</sup> and 12<sup>th</sup> Sun user group conferences. California**

### [DCSCHMIDT1999\_01]

An architectural overview of the ACE framework. A Case Study of Successful Cross-Platform Systems Software Reuse

**Douglas C. Schmidt** - 1999  
**USENIX**

### [JPMARTIN2004\_01]

Distributed Event Correlation and Self-Managed Systems

**Jean-Philippe Martin-Flatin** - 2004  
**Proc. Int WorkShop on Self-\* Properties in Complex Inf. Systems**

### [RVAARANDI2002\_01]

Platform Independent Event Correlation Tool for Network Management.

**Risto Vaarandi**

**- 2002**

**2002 IEEE/IFIP Network Operations & Management Symposium**

**[AHANEMANN2004\_01]**

Assured Service Quality by Improved Fault Management. ServiceOriented Event Correlation.

**Andreas Hanemann et al**

**- 2004**

**ICSOC'04 New York, USA**

**[JLHELLERSTEIN\_01]**

Minig event data for actionable patterns.

**Joseph L. Hellerstein and S. Ma - 1999**