ARQUITECTURA DEL SISTEMA DE MONITOREO Y GENERACIÓN DE NOTIFICACIONES EN GNU/LINUX MEDIANTE AGENTES PARA UN SISTEMA DE GESTIÓN DIGITAL

Rafaela Blanca Silva López
Universidad Autónoma Metropolitana
r.silva@correo.ler.uam.mx

Viridiana Matías Hernández
Universidad Autónoma Metropolitana
vmatiashdez@gmail.com

Israel Isaac Montes de Oca Solis
Universidad Autónoma Metropolitana
isaac.i.montes@gmail.com

Hugo Pablo Leyva
Universidad Autónoma Metropolitana
hpl@correo.azc.uam.mx

Resumen

Es prioritario mejorar el desempeño y la eficiencia de los procesos administrativos de las Instituciones de Educación Superior (IES), para lo cual se requiere de Sistemas de Gestión Digital (SGD) que apoyen las funciones sustantivas de la IES y ofrezcan mecanismos para mantener la disponibilidad de los servicios. El objetivo fue diseñar la Arquitectura del Sistema de Monitoreo y Generación de Notificaciones (SMGN) basada en agentes. La función de los agentes fue el monitoreo de servidores y servicios en tiempo real, integrando el envío de notificaciones y generación de estadísticas. La implementación del SMGN se probó con los servidores donde se localizan los componentes del SGD, los resultados obtenidos son satisfactorios, se obtuvieron mejores tiempos de

respuesta al atender las fallas antes de que el usuario se percate de que el servicio no está activo. Por tanto, el SMGN se encarga de mejorar la eficiencia y disponibilidad de los servicios del SGD, mejorando la productividad, los tiempos de respuesta y la atención para la comunidad universitaria, beneficiando hasta a 3960 usuarios de la IES con una inversión mínima [1].

Palabra(s) Clave(s): agentes, automatización de procesos, gestión de procesos, gestión del rendimiento, monitoreo de servicios.

1. Introducción

Fomentar la competitividad en las Instituciones de Educación Superior (IES), es un mandato prioritario, por tal razón la gestión educativa debe transformarse y adoptar nuevos paradigmas que integren la gestión de los procesos, las personas y los recursos. Existen proveedores (SAP, Oracle, IBM y Microsoft., entro otros) que proporcionan soluciones para la gestión del rendimiento. Sin embargo, son muy costosas, no se ajustan a las necesidades de una IES y no incluyen mecanismos de monitoreo que garanticen la disponibilidad permanente de los servicios.

La eficiencia de los centros educativos ha sido una de las principales preocupaciones de teóricos y prácticos en educación [2,3]. Es fundamental en una Institución de Educación Superior mejorar el desempeño y la eficiencia de los procesos administrativos en beneficio de la comunidad universitaria, lo que demanda en una Institución innovadora repensar las formas de gestión [4]; cuando hablamos de gestión nos referimos a los procesos educativos, administrativos, sociales, laborales, pedagógicos y tecnológicos [5], orientados al cumplimiento de la misión Institucional [6]. Lo anterior vislumbra la necesidad de contar con un Sistema de Gestión Digital para las IES. Al hablar de la gestión digital nos referimos a la automatización de procesos clave que facilitan la toma de decisiones estratégicas al interior de una Institución tomando como base el modelado de procesos [1].

Desde el punto de vista tecnológico, existen herramientas comerciales y open source dedicadas a los servicios de monitoreo. Los proveedores ofertan desde

servicios básicos hasta la infraestructura completa, podemos mencionar a Openview [7], PRTG Network Monitor [8], Nagios [9], Logstash [10] y Kibana [11], entre otros, si bien son de utilidad resultan ser complejas para configurar, requieren de muchos recursos y tienen un costo elevado, por lo que no están al alcance de las IES. El objetivo de este trabajo fue diseñar la arquitectura del sistema de monitoreo y generación de notificaciones de los servicios ofrecidos por un Sistema de Gestión Digital (SGD), mediante agentes.

El Sistema de Monitoreo y Generación de Notificaciones (SMGN) evalúa los eventos que se presentan en los servidores donde se encuentra instalado el SGD, cuando se detecta una falla se envía una notificación al administrador para que resuelva el problema antes de que el usuario perciba la falla. Garantizando la disponibilidad de los servicios que ofrecen los servidores donde se encuentran instalados los componentes del SGD, lo que provoca mayor disponibilidad de los servicios. El SGD mejora la productividad, los tiempos de respuesta y la atención para la Comunidad Universitaria. En otras investigaciones se ha concluido que el SGD beneficia a 3960 usuarios al interior de la IES con una inversión mínima que permitirá mejorar la productividad y los tiempos de respuesta [1].

El Sistema de Gestión Digital

La arquitectura del Sistema de Gestión Digital se basa en la propuesta del meta-modelo de procesos clave departamental de la Universidad Autónoma Metropolitana (UAM) [1]. El modelado de procesos se categoriza en tres niveles: procesos estratégicos, operativos y de apoyo, lo que permite concebir la gestión como un sistema de componentes interrelacionados [12].

En la meta-modelo de procesos del Sistema de Gestión Digital de la UAM, los procesos se clasifican en tres niveles (figura 1). En el nivel principal se encuentran los procesos estratégicos donde se ubica la planeación, la presupuestación y la vinculación (generadores de ingresos). Seguido por el nivel de procesos sustantivos en los que se incluye la docencia, la investigación, la preservación y difusión de la cultura (funciones sustantivas de la Universidad), también se incorporan los procesos asociados con el apoyo a los alumnos a través de becas y

la gestión de proyectos patrocinados. Por último los procesos de apoyo integran los procesos de contratación, prestaciones, recursos financieros, servicios, editorial y comunicación.

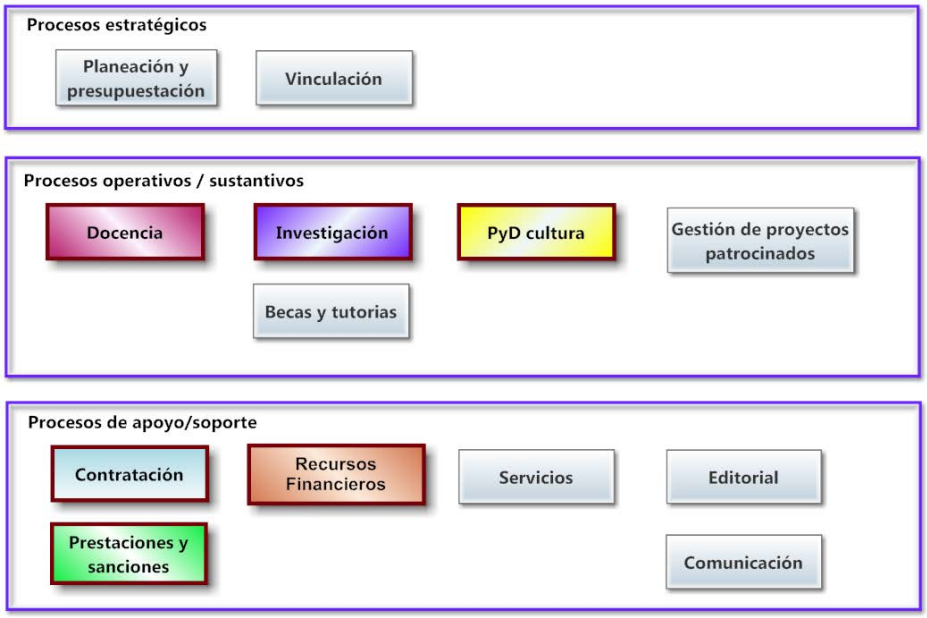

Figura 1 Meta-modelo de procesos del Sistema de Gestión Digital de la UAM.

Arquitectura organizacional del Sistema de Gestión Digital

Como caso aplicativo se ha decidido trabajar con la Unidad Lerma, dado que es el campus de más reciente creación de la UAM, cuenta con recursos limitados lo que provoca una necesidad de sistemas informáticos que apoyen al personal administrativo para la realización de los procesos que las otras Unidades aplican sin problema por tener el personal necesario para cada actividad.

Las actividades administrativas involucran: autoridades, profesores-investigadores, personal administrativo, alumnos, y a otros actores de la comunidad universitaria. Esto requiere de la continuidad del servicio las 24 horas del día de los 365 días del año.

Dos profesores-investigadores están a cargo de la administración de los servidores dónde se alojan las bases de datos, herramientas y aplicaciones, debido a la diversidad de responsabilidades en cuanto a docencia, investigación y gestión, no pueden estar monitoreando de manera permanente los servicios que

soportan el SGD. Por tanto, para garantizar un buen servicio, es necesario contar con mecanismos automatizados para monitorear los servicios y notificar fallas detectadas mediante correo electrónico al responsable y realice las acciones correctivas correspondientes para resolver el problema antes de que sea notificado por el usuario final.

Bajo este contexto, se propone la arquitectura del sistema de monitoreo y generación de notificaciones en GNU/Linux [13]. El desarrollo de los sistemas que soportan las actividades clave de la Unidad, consideran las recomendaciones del modelo de referencia de arquitectura empresarial de: The Open Group Architecture Framework (TOGAF) [14].

TOGAF propone un modelo a capas en las que integra la arquitectura tecnológica, datos, aplicaciones y negocios. En la arquitectura tecnológica o de infraestructura considera los requerimientos de hardware y software, así como lo relacionado con el uso de la red. Mientras que la arquitectura de datos considera la información que se va a resguardar, administrar y analizar. Por otro lado, la arquitectura de aplicaciones incorpora los servicios automatizados que requiere el negocio (en nuestro caso la Institución). Finalmente en la arquitectura de negocios se realiza el mapeo de procesos de negocio, integra la estructura organizacional y a las personas involucradas (figura 2).

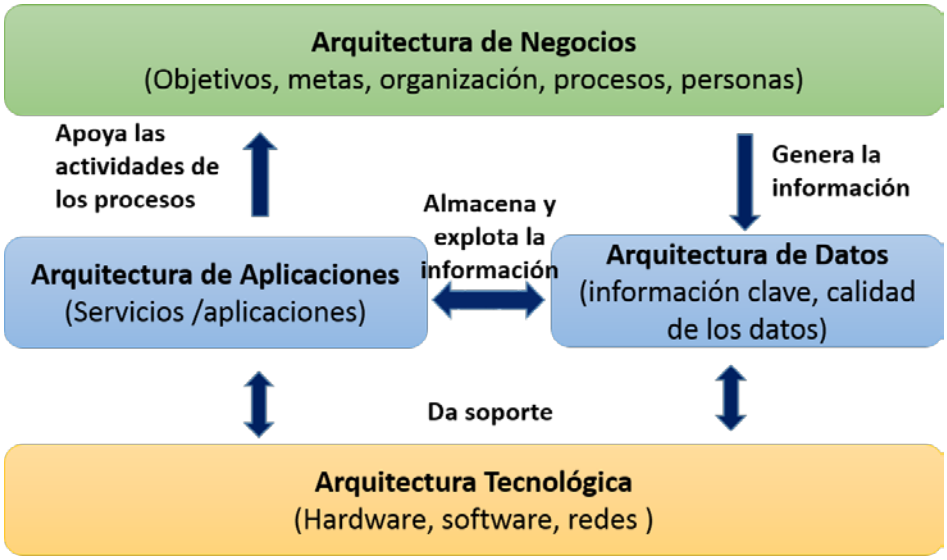

Figura 2 Arquitecturas que considera TOGAF 9.1.

En el Sistema de Gestión Digital se trabajó en la arquitectura de negocios obteniendo como resultados el meta-modelo de procesos del Sistema de Gestión Digital. Mientras que en la arquitectura de las aplicaciones encargadas de servicios de docencia (gestión de tutores y manejo de becas, por ejemplo), se diseñó una base de datos que integra todo lo que requieren los sistemas que apoyan las actividades de los diferentes procesos. Por último, en la arquitectura tecnológica se incorpora el Sistema de Monitoreo y Notificaciones cuyo objetivo es detectar fallas y promover la disponibilidad de servicios que la Institución requiere.

Arquitectura tecnológica del Sistema de Gestión Digital

Vigilar el funcionamiento de servicios, aplicaciones e inclusive los propios servidores que conforman la infraestructura tecnológica de toda organización, es una tarea esencial. Por lo que se requiere revisar periódicamente las bitácoras de los servicios para detectar alguna anomalía, y realizar las acciones correctivas correspondientes para mantener los servicios funcionando.

Existen diversas herramientas comerciales y open source que ofrecen servicio de monitoreo. A continuación se describen tres de los más usados:

Nagios [9] es una solución open source ampliamente utilizada, provee un servicio de monitoreo por medio de una interfaz web, muestra la topología de la red, permite la programación de plugins adicionales a los que ya incluye la plataforma. Sin embargo su implementación es complicada, requiere de conocimientos técnicos, herramientas adicionales y una inversión de tiempo considerable.

Logstash [10] es una herramienta que extrae datos de las bitácoras sobre eventos que se presentan en los servicios o aplicaciones que están ejecutándose. Para configurarlo se requiere de conocimientos avanzados del sistema operativo, no cuenta con una interfaz gráfica para la configuración, lo que hace que se vuelva compleja.

Openview [7] es una solución comercial de HP, ofrece una arquitectura de tipo consola-agente. Incluye paquetes de software (agentes) para los principales sistemas operativos. Los agentes son independientes de la consola central, se encargan de informar a la consola las excepciones que se presenten, además

monitorean archivos de bitácoras, permite programar tareas, capturar eventos con el protocolo SNMP4 [15], entre otras. Proporciona un mapa lógico de los componentes de la infraestructura y del estado de los servicios monitoreados. Favorece la escalabilidad e integralidad al manejar una gran cantidad de dispositivos. Tiene un costo muy elevado, por lo que es una solución inaccesible para una IES. Estas herramientas son complejas para configurar, requieren de muchos recursos o son costosas, por lo que no están al alcance de las IES.

Metodologías para el diseño de la arquitectura del Sistema de Monitoreo y Notificaciones

Se utilizó la metodología GAIA [16,17] para realizar el diseño de la arquitectura del Sistema de Monitoreo y Notificaciones basado en agentes, se definieron:

- Los requisitos iniciales.
- La organización de los roles y sus relaciones contemplando las responsabilidades del agente, los recursos que utiliza, las tareas realizadas y las interacciones.
- Se determina el tipo de agente, el número de instancias y el modelo de servicios (funciones del agente) asociados a cada rol.
- Los enlaces de comunicaciones que existen entre los agentes.

Por último, para realizar la implementación se aplicó SCRUM, un proceso que contempla un conjunto de buenas prácticas para trabajar colaborativamente. Integra el ciclo de vida de desarrollo de software iterativo incremental en el que se van desarrollando módulos o componentes que se integran al producto final [18].

Implementación con agentes reactivos

El sistema para el monitoreo se implementó mediante el uso de agentes reactivos. Dado que los Sistemas Multi-Agentes (SMA) son herramientas de software que facilitan el monitoreo de múltiples sucesos y enviar notificaciones, o tomar decisiones, los componentes de software (agentes) pueden correr en diferentes equipos de manera concurrente, facilitan la solución de problemas

complejos mediante el uso de un grupo de agentes especializados en acciones sencillas que en su conjunto resuelven un problema complejo [19]. Dada la autonomía de un agente, es posible modelarlos de manera independiente; por tanto, debido a la naturaleza autónoma y proactiva de un agente y las características de manejo de información distribuida, coordinación y comunicación entre agentes, se convierten en la alternativa primordial para la implementación de un sistema de monitoreo de servicios en servidores GNU/Linux.

Se localizaron trabajos cuyo enfoque se asocia con el uso de SMA en el ámbito de las redes y comunicaciones. Su enfoque está asociado a la comunicación entre protocolos [20], la comunicación entre agentes [21] y el monitoreo de redes de computadoras [22].

2. Desarrollo

La solución propuesta ofrece el monitoreo en tiempo real de los servicios que se proporcionan en los servidores de la red a través de una aplicación web. Brinda una configuración simple para la implementación del monitoreo y notificación. Integra un módulo de envío de notificaciones vía correo electrónico en caso de detectar fallos. Además permite la realización de consultas en períodos de tiempo especificados por el usuario sobre datos históricos y actividades monitoreadas. Utiliza software libre, por lo que no genera un gasto fijo de licenciamiento.

Los componentes del SMGN sobre Servidores Linux integran seis módulos:

- Automatización de extracción de las bitácoras.
- Automatización del almacenamiento de las bitácoras.
- Monitoreo de esclavos en servidores.
- Generación de estadísticas.
- Monitoreo de servicios en tiempo real.
- Envío de notificaciones. Los módulos implementados mediante agentes, se interconectan a través de la red con 2 subsistemas.

El subsistema A encargado de la recopilación de información de las bitácoras tal como timestamp (fecha y hora en que ocurre el evento), nombre de evento,

descripción o método para el caso de peticiones de los web servers, IP origen, recurso al que se intentó acceder, el usuario que intentó el acceso, entre otras; y el subsistema B enfocado en el análisis de la información, la generación de estadísticas y la notificación de las fallas que se presenten en los servicios de los servidores involucrados (figura 3).

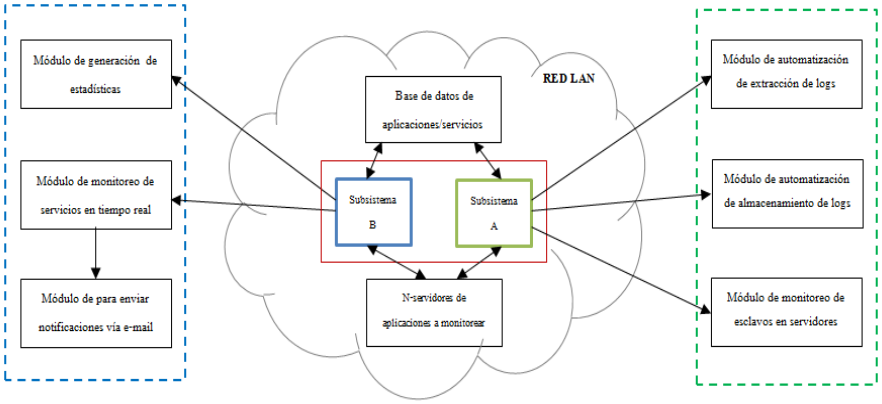

Figura 3 Componentes del SMGN sobre Servidores Linux.

Arquitectura del Sistema de Monitoreo y Generación de Notificaciones sobre servidores GNU/Linux

La arquitectura del Sistema de Monitoreo y Generación de Notificaciones (SMGN) se compone de 6 agentes, 3 asociados al monitoreo de eventos y automatización, extracción y almacenamiento de bitácoras y 3 relacionados con el monitoreo en tiempo real, la generación de notificaciones y estadísticas. Las relaciones entre los agentes requieren de una infraestructura tecnológica para su funcionamiento: una base de datos, un servidor de aplicaciones y la interfaz gráfica (figura 4).

Arquitectura tecnológica del módulo monitoreo de servicios en tiempo real

Este módulo es el encargado de monitorear en tiempo real el estado de los servicios que se encuentran en cada uno de los servidores de la red local. Informa al servidor web de los eventos ocurridos. Para la realización de este módulo se implementó la comunicación con sockets bajo el protocolo TCP/IP en python siguiendo una arquitectura cliente-servidor.

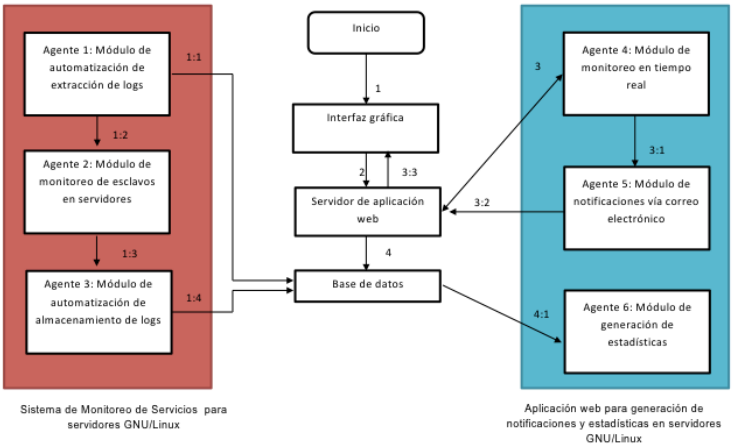

Figura 4 Arquitectura del SMGN.

La arquitectura utiliza tecnologías como Node JS, manejador de base de datos (MongoDB NoSQL, PostgreSQL), servidor Web (Apache Web Server, NGINX), servidor de aplicaciones (Apache TOMCAT), compiladores (Python, entre otros), y sistemas operativos (Linux: Debian, y Centos). Los productos y marcas son propiedad de las empresas correspondientes.

En éste módulo se propone un agente encargado de monitorear las bitácoras de diversos servidores en los que se alojan aplicaciones asociadas con el Sistema de Gestión Digital (agente 4). Además, se diseña un agente encargado de la extracción de información de las bitácoras, en los servidores que se encuentran en producción (agente 3). Por último se integra el agente encargado de la generación de estadísticas (figura 5).

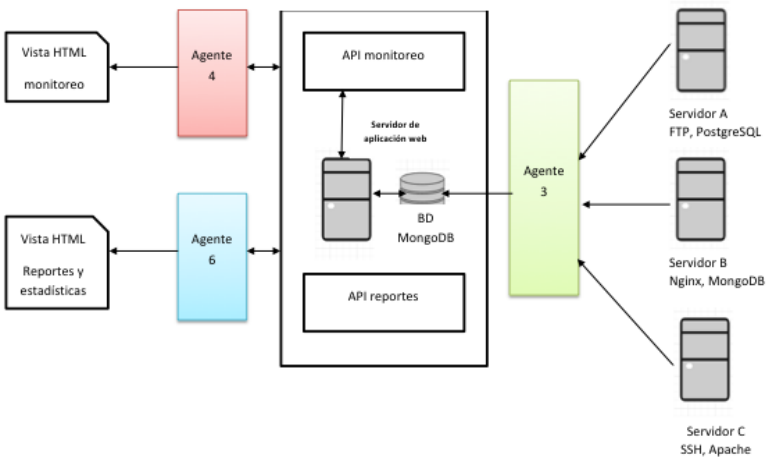

Figura 5 Arquitectura del módulo de servicios en tiempo real.

Arquitectura del módulo encargado de la generación de notificaciones y estadísticas

Este módulo extrae los datos que son almacenados en la base de datos por el sistema de monitoreo de servicios en sistemas GNU/Linux [20]. Obtiene la información en un rango de tiempo determinado (fecha actual, la última semana o una fecha específica) sobre el estado de los servicios de los servidores que son monitoreados, mostrando una gráfica con el número de eventos exitosos y fallidos del servicio seleccionado.

Arquitectura del módulo de extracción de datos de las bitácoras

La arquitectura tecnológica de éste módulo contempla la extracción de los datos de las bitácoras asociadas a los servicios que ofrece cada servidor (agentes 1 y 2), además integra los agentes encargados del procesamiento de información extraída de las bitácoras (agente 3). Por último, contempla un agente de control encargado de la periodicidad con la que se procesa la información (agente 7). La interacción entre los agentes y los servidores que alojan los servicios requeridos por el Sistema de Gestión Digital que se deben monitorear se incluyen en este módulo (figura 6).

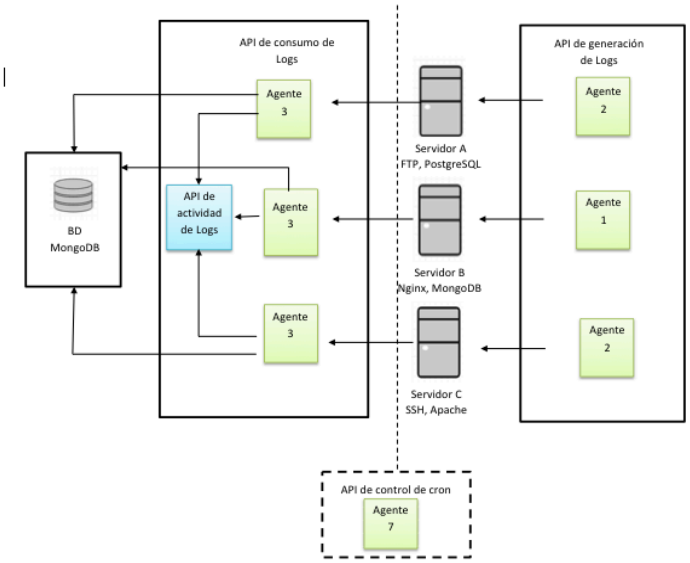

Figura 6 Arquitectura del módulo de extracción de datos de las bitácoras.

Por otro lado, el módulo integra un agente que genera la interfaz gráfica desde la cual se pueden visualizar las alertas y notificaciones del estado de los servicios en los servidores monitoreados. Adicionalmente envía un correo electrónico al detectar alguna falla o estado incorrecto de alguno de los servicios.

La arquitectura del módulo encargado de la generación de notificaciones y estadísticas interactúa con los datos filtrados por los agentes que se encargan de la extracción y procesamiento de datos almacenados en las bitácoras de los servidores que se están monitoreando (figura 7).

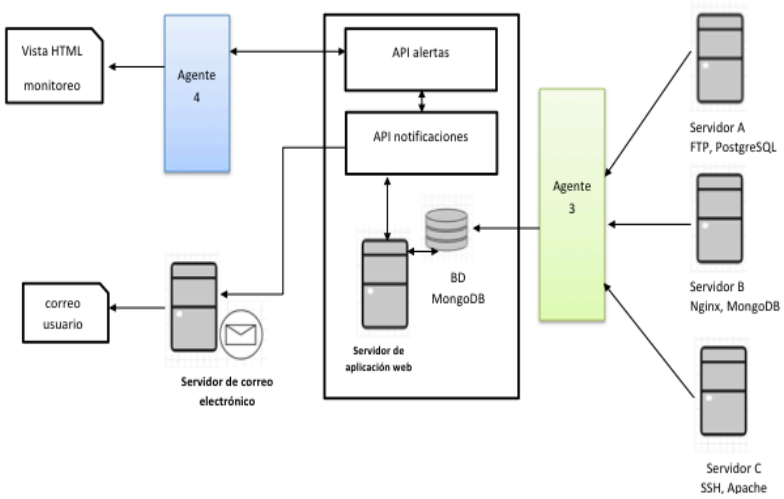

Figura 7 Arquitectura del módulo de notificaciones.

Monitoreo de Servidores distribuidos en la red

Los servidores en los que se alojan los servicios que soportan al SGD cuentan con sistema operativo Centos versión 6.7, contemplando tanto servidores normales (Servidor Dell con 18 GB RAM, 2 Cores, 2 TB de HD), como servidores virtualizados, bajo una implementación de Cloud Computing. La infraestructura tecnológica con la que se cuenta incluye 4 servidores Blade HP Proliant BL 465c G7, cada uno con 24 microprocesadores AMD Opteron 6174 de 2.199 GHz, con licencia VMware vSphere 5 Desktop y 16 GB de memoria RAM. Utilizan el software VMware vSphere 5 para llevar a cabo la virtualización de los recursos distribuidos de manera dinámica para optimizar los recursos existentes.

Actividades realizadas por los agentes

Para el monitoreo de servicios en tiempo real se utilizaron dos algoritmos para realizar las acciones de estos agentes, uno para las actividades del esclavo (instalado en cada servidor que se monitorea) y otro para las actividades del maestro.

Esclavo: en cada servidor que se monitorea se instaló un cliente que se encargará de ejecutar una serie de comandos para determinar el estatus de cada aplicación y servicio que se monitorea. Una vez que recaba los datos, los envía a la base de datos del servidor maestro. Estos pasos los realiza de manera permanente con un delay de tiempo. Maestro: abre conexión al puerto del socket por el cual cada uno de los esclavos enviará la información de los estatus recabados, se mantiene siempre en escucha. Envía la información que recibe de cada uno de los esclavos a la aplicación web. Los datos que se muestran en la aplicación se obtienen en períodos de tiempo no mayores a 10 segundos.

Para la generación de estadísticas, el algoritmo contempla un componente implementado con servicios REST que realiza la búsqueda de información y separa los datos en errores y aciertos, los datos obtenidos se reflejan en gráficos de la aplicación.

Por último, para el módulo de notificaciones vía correo electrónico, el algoritmo integra la búsqueda en cada uno de los registros obtenidos por los agentes, si se cumplen las restricciones que indican un posible error se toman los datos del registro y se envían por correo electrónico y los registra en la base de datos para poder ser consultados posteriormente desde la aplicación.

3. Resultados

Los resultados muestran que el SMGN proporciona un mecanismo eficiente de monitoreo, permitiendo una mayor disponibilidad de los servicios que apoyan el Sistema de Gestión Digital. Cabe destacar que las pruebas de extracción de todas las bitácoras se hicieron en servicios bajo el sistema operativo Centos 6.7, en servidores normales y servidores virtualizados. El monitoreo de los eventos según

los servicios monitoreados en cada servidor, presenta los eventos sin error para el servicio de apache monitoreado en los diferentes servidores (figura 8).

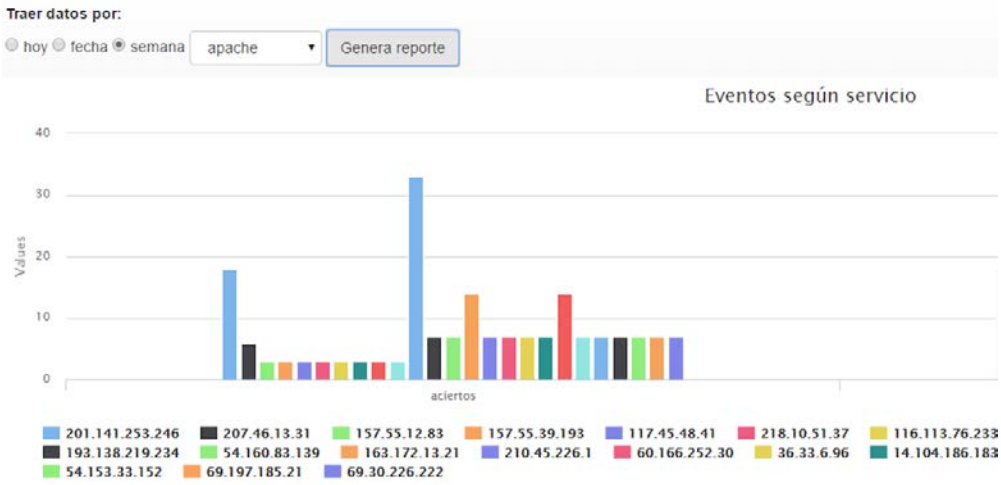

Figura 8 Vista del monitoreo de servicios en los diferentes servidores.

Las pruebas se realizaron sobre distribuciones de Linux; Debian 7, Debian 8 y Centos 6.2 encontrando diferencias en la ejecución de los clientes que realizan el monitoreo, por lo que se hizo una versión de cliente para Debian (7 y 8) y otra para Centos. Los servicios monitoreados fueron nginx, FTP, mongoDB, postgreSQL, apache Web Server, Apache Tomcat y SSH en diversos servidores. Los resultados de la extracción de las bitácoras, muestran las direcciones IP de los servidores, los servicios monitoreados y su estado (figura 9).

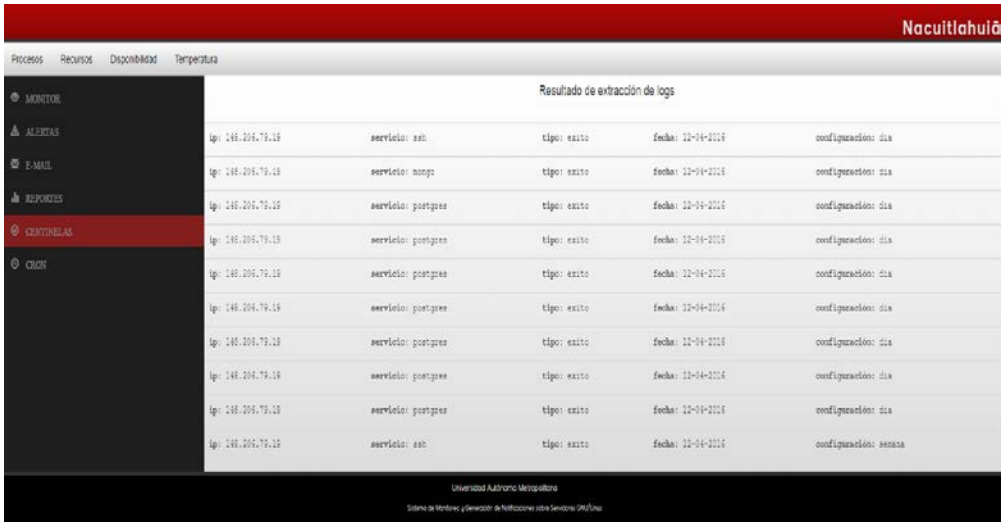

Figura 9 Resultados obtenidos de la extracción de las bitácoras.

En los resultados obtenidos durante el monitoreo de los servidores, se observó que la tasa de eventos por error es relativamente baja (menor al 5%), la mayoría de los errores obtenidos fueron generados de manera intencional para probar el funcionamiento de los módulos y el desplegado de alertas (figura 10).

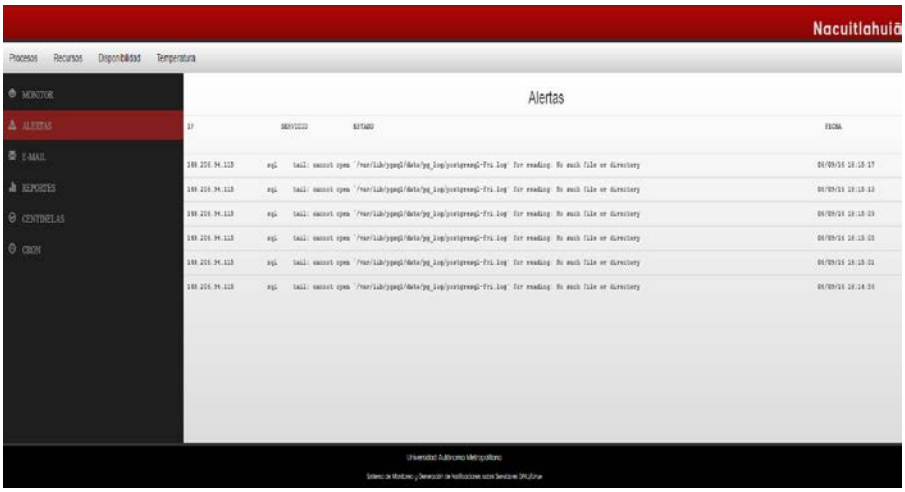

Figura 10 Desplegado de alertas al realizar pruebas de monitoreo.

4. Discusión

El monitoreo de servidores que deben dar servicio las 24 horas los 365 días del año, deben ser monitoreados en tiempo real para garantizar la disponibilidad del servicio. Para lograrlo se implementaron dos algoritmos que realizan las acciones de los agentes, uno para las actividades del proceso esclavo instalado en cada servidor que se monitorea; en el que se instaló un cliente encargado de ejecutar una serie de comandos para determinar el estado de cada aplicación. El otro (proceso maestro) envía la información que recibe de cada uno de los esclavos a la aplicación web, en periodos de tiempo no mayor a 10 segundos.

Por otro lado, al realizar las pruebas se encontraron diferencias en la ejecución de los clientes que realizan el monitoreo dependiendo de las distribuciones de Linux que tenía instalado en el servidor: Debian 7, Debian 8 y Centos 6.2.; por lo que fue necesario desarrollar un agente distinto para el cliente de Debian (7 y 8) y otro para Centos.

La arquitectura del SMGN es muy genérica, sencilla y escalable, lo que permite el monitoreo de cualquier servidor que ofrezca servicios similares, por lo que puede

extrapolarse su uso. Además la integración de nuevos módulos con otras funcionalidades es factible con una inversión mínima.

5. Conclusiones

Nuestros resultados demuestran que el diseño y construcción del Sistema de Monitoreo y Generación de Notificaciones (SMGN) para servidores GNU/Linux, garantiza la disponibilidad de servicios que soportan el Sistema de Gestión Digital (SGD) de manera semi-automatizada al implementarlo con agentes reactivos. Los resultados obtenidos son satisfactorios, ya que durante las pruebas se obtuvieron mejores tiempos de respuesta al atender las fallas antes de que el usuario se percate de que el servicio no está activo.

El desarrollo de la arquitectura del SMGN y su implementación, brindan al administrador de los servidores, un mecanismo eficaz para un monitoreo básico de servicios sin la necesidad de estar observando continuamente lo que ocurre en los servidores. Lo anterior le permite dedicar tiempo a otras actividades y estar informado en caso de que se presente algún error inesperado.

Como trabajo futuro se espera integrar módulos que realicen las notificaciones hacia dispositivos móviles, así como componentes que automaticen de manera integral la atención a fallas.

6. Bibliografía y Referencias

[1] R. Silva, E. Cruz, I. Méndez, J. A. Hernández, “Sistema de Gestión Digital para mejorar los procesos administrativos de Instituciones de Educación Superior: Caso de estudio en la Universidad Autónoma Metropolitana”. Perspectiva Educacional, Vol.52. N°2. 2013. Pp.104-134.

[2] J. L. Muñoz, D. Rodríguez-Gómez, A. Barrera-Corominas, “Herramientas para la mejora de las organizaciones educativas y su relación con el entorno”. Perspectiva educacional. Vol. 52. N° 1. 2013. Pp. 97-123.

[3] A. Chase, N.J. Aquilano, Dirección y Administración de la Producción y de las Operaciones. Sexta Edición. 1995. Editorial Irwin. México.

[4] A. Villela, “Gestión en Educación Superior: Funciones, capacidades y necesidades directivas”. Revista Akademeia. Vol. 2. N° 1. 2011.

[5] La educación y la sociedad del conocimiento y de la información. http://www.pedagogica.edu.co/storage/rce/articulos/rce36-37_09controv.pdf. Febrero 2016.

[6] I. Álvarez, E. Iturbe, Los estudios de caso como estrategia para la formación en gestión. 2005. Ed. Taller Abierto. México.

[7] OpenView. http://searchdatacenter.techtarget.com/definition/HP-OpenView. Abril 2016.

[8] PRTG Network Monitor. https://www.es.paessler.com/prtg. Abril 2016.

[9] Nagios. https://www.nagios.org/. Abril 2016.

[10] Logstash. https://www.elastic.co/products/logstash. Abril 2016.

[11] Kibana. https://www.elastic.co/guide/kibana/en/kibana/current/index.html. Abril 2016.

[12] A. Manrique, Sistematización de modelos de gestión educativa de los gobiernos regionales de San Martín, Arequipa y la Libertad. 2011. Proyecto USAID/ PERU/SUMA. Lima, Perú.

[13] GNU/Linux. http://www.gnu.org/gnu/linux-and-gnu.es.html. Abril 2016.

[14] TOGAF. http://pubs.opengroup.org/architecture/togaf9-doc/arch/. Abril 2016.

[15] SNMP. https://tools.ietf.org/html/rfc3413. Abril 2016.

[16] M. Wooldridge, N.R. Jennings, and D. Kinny, The Gaia Methodology for Agent-Oriented Analysis and Design, Journal of Autonomous Agents and Multi-Agent Systems, vol.15. 2000.

[17] F. Zambonelly, M. Wooldridge, N.R. Jennings, “Organisational Rules as an Abstraction for the Analysis and Design of Multi-Agent Systems”. International Journal of Software Engineering and knowledge Engineering. 2000.

[18] SCRUM. https://www.google.com.mx/#q=SCRUM&gws_rd=cr. Abril 2016.

[19] J. Cruz, A. I. Hernández, A. Beuchée, S. Wong, G. Passariello, F. Mora, G. Carrault, “Sistema Multiagentes para el Monitoreo Inteligente”. V Latin

American Congress on Biomedical Engineering 2007, Bioengineering Solutions for Latin America Health. Springer. Vol.18. 2007. Pp. 501-505.

[20] M. Bravo, J. Velazquez, “Measurng Heterogeneity between Web-Based Agent Communication Protocols”. On the Move to Meaningful Internet Systems: OTM 2008 Workshops. Vol. 5333. 2008. Pp. 656-665.

[21] A. F. Moreira, R. Vieira, R.H. Bordini, “Extending the Operational Semantics of a BDI Agent-Oriented Programming Language for Introducing Speech-Act Based Communication”. Declarative Agent Languages and Technologies. Vol. 2990. 2004. Pp. 135-154.

[22] A. De la Hoz Manotas, “Diseño de un sistema multi-agente para monitoreo de redes utilizando JADE y JPCAP”. INGE CUC. Vol. 6. N° 1. 2010. Pp. 227-236.

7. Autores

Dra. Rafaela Blanca Silva López es Profesor-Investigador de la Universidad Autónoma Metropolitana Unidad Azcapotzalco, Ingeniero en Electrónica, concluyó la Maestría en Ciencias de la Computación en la Universidad Autónoma Metropolitana Unidad Azcapotzalco. Concluyo sus estudios de Doctorado en Sistemas y Ambientes Educativos en la Universidad de Guadalajara UDGVirtual en 2016.

Ing. Viridiana Matías Hernández, egresada de la carrera de Ingeniería en Computación por la Universidad Autónoma Metropolitana Unidad Azcapotzalco con área de concentración de Seguridad y Redes de Computadoras. Desarrolladora de sistemas y aplicaciones web, actualmente laboro en el área de middleware en AT&T.

Ing. Israel Isaac Montes de Oca Solis, egresado de la carrera en Ingeniería en Computación por la Universidad Autónoma Metropolitana Unidad Azcapotzalco con área de concentración de Seguridad y Redes de Computadoras.

M.C. Hugo Pablo Leyva es Profesor-Investigador de la Universidad Autónoma Metropolitana Unidad Azcapotzalco. Obtuvo su título de Maestría en Ciencias de la Computación en la Universidad Autónoma Metropolitana Unidad Azcapotzalco.