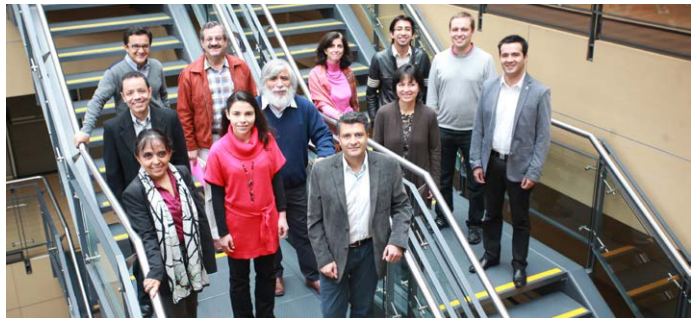
Arquitecturas para escalar en la nube

Proveedores, implementadores y académicos estudiosos de las posibilidades que ofrece el *cloud computing* a las organizaciones se reunieron el 17 de junio del 2015 en la Universidad de los Andes. Discutieron acerca de las estrategias sobre cómo sacarle provecho a este recurso informático.



El tema de las arquitecturas *cloud* unió a los grupos de investigación de Uniandes COMIT (foto arriba), que se centra en infraestructura, y TICSw (foto abajo), que se ocupa del software, porque su actividad es complementaria.

Escalar para que las aplicaciones atiendan a más usuarios y ser elásticos con el fin de crecer o contraerse según la demanda es el reto de las organizaciones en un mundo signado por las tecnologías de información. En ese ámbito, los desarrolladores enfrentan desafíos para determinar cuál es la arquitectura conveniente para irse a la nube o mantenerse en escenarios *on-premise* (instalados en los servidores del usuario).

Mediante arquitecturas escalables y elásticas de los sitios web, ellos deben garantizar que los usuarios puedan navegar o hacer transacciones y consultas seguras y sin contratiempos en días normales. También tienen que lidiar con picos de demanda como los que ocurren en Estados Unidos cuando se celebra el *Black Friday* o con los *Cyber Mondays* (o ciberlunes en Colombia), días en los que los potenciales compradores aumentan considerablemente atraídos por las rebajas de los comerciantes. Y el reto no es solo crear aplicaciones desde cero, sino encontrar la manera de escalar a la nube las que ya existen.

Sobre esa temática giró el 9.º Foro Isis de *Cloud Computing* "Arquitectura *cloud:* estrategias para hacer aplicaciones escalables en la nube" que se llevó a cabo en la Universidad de los Andes el 17 de junio de 2015, organizado por el Departamento de Ingeniería de Sistemas y Computación (DISC).

> "Uno de los mayores desafíos en desarrollo de software por demanda es lograr aplicaciones suficientemente robustas para las necesidades de los clientes".
>
> **Armando Quiñones y Giovanny Saray, Intergrupo**

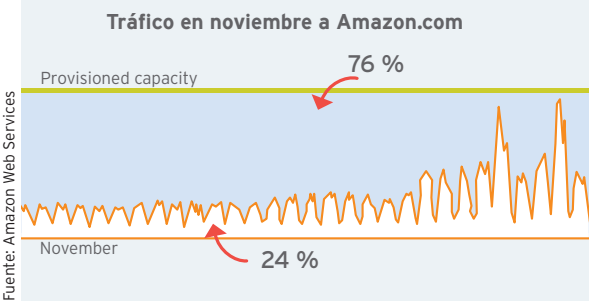
Para tratar el tema, invitaron a los proveedores Henry Alvarado y Julio Faerman, de Amazon Web Services (AWS); Germán Ramírez, de Grupodot en representación de Google; Nicolás Jiménez, de Azure de Microsoft, y David Córdoba, de Oracle. También participaron los implementadores o integradores Yuji Kiriki Rodríguez, de Seven4N; Santiago Gil, de Heinsohn, y Armando Quiñones y Giovanny Saray, de Intergrupo. A su vez, el profesor Harold Castro, del DISC, y Santiago Gil, de Heinsohn, presentaron el trabajo de la academia con un avance de los resultados de un proyecto de investigación que comenzó hace año y medio.

Recomendaciones de Amazon Web Services (AWS)

Henry Alvarado, arquitecto de soluciones de AWS para Latinoamérica, centró su intervención en las soluciones y herramientas

para escalar una aplicación hasta atender a 10 millones de usuarios. Resaltó que el *auto scaling* es una herramienta útil, pero no soluciona todos los problemas de escalabilidad y formuló, entre otras, las siguientes recomendaciones:

- Decidir si se va a usar una base de datos con lenguaje SQL (*Structured Query Language)* o NoSQL. Lo mejor es comenzar por una SQL porque es una tecnología conocida que no se va derrumbar fácilmente con 10 millones de usuarios, "a menos que, realmente, estén haciendo algo fuera de lo común o tengan cantidades masivas de datos (del orden de 5 terabytes) almacenados en ese servicio". El NoSQL se aconseja cuando las aplicaciones son de muy baja latencia (retardos o demoras en la transmisión de paquetes en la red), con búsquedas de datos altamente no relacionales, o cuando hay gestión de datos.
- Usar varias zonas de disponibilidad (*data centers* físicos) para crear una réplica esclava de las instancias web o aplicaciones y poder hacer *failover,* es decir, cambiar automáticamente el procesamiento de un componente defectuoso.
- Tener en cuenta el *performance* y la eficiencia. Por ejemplo, se puede llevar todo el contenido estático del sitio a un almacenamiento que sea nativamente *on line,* con alta disponibilidad, de manera que se saca de la aplicación, con lo cual se optimizan recursos y se disminuyen costos. También puede usarse memoria *caché* para las consultas (*queries)* más comunes de la base de datos.
- Usar métricas para que el sistema crezca o disminuya su tamaño automáticamente sin desperdiciar capacidad de cómputo. El *auto scaling* permite ser costo-eficiente. Además, deben instalarse alarmas de monitoreo para detectar problemas con el *performance.*
- Automatizar el análisis de los *logs* (archivos de texto plano que se usan en el software para llevar la traza detallada de lo que está sucediendo en el sistema).
- Prestar atención a lo que dicen los clientes en las redes sociales para atacar pronto el problema.
- Para alta escalabilidad, usar SOA (*Service Oriented Architecture*), con el fin de que haya arquitecturas independientes basadas en servicios (por ejemplo, una para el motor de búsqueda, otra para los foros, otra para productos...). Esto facilita desacoplar componentes y quitarle cargas innecesarias a la arquitectura del sistema, tener escalabilidad independiente y lograr mayor control y flexibilidad. "Usen las herramientas disponibles. No intenten recrear algo que ya existe".



La gráfica muestra el comportamiento del tráfico hacia Amazon en noviembre. La mayor parte del tiempo es regular, pero tiene dos picos muy fuertes de demanda por el *Black Friday* y el *Cyber Monday.* Si el operador aprovisiona 10 o 15 por encima de los picos, la mayor parte del tiempo solo está usando el 24 de la capacidad (desperdicio del 76). Con el *auto scaling* puede crecer y contraerse según la demanda, para ser costo-eficiente y escalar automáticamente.

Los ejemplos de Google

German Ramírez, director técnico del Grupodot, en representación de Google, mencionó que autoescalar en la nube aplicaciones que ya existen supone retos técnicos y económicos. Es indispensable saber moverse para responder a la presión del mercado y no perder el espacio ganado, pues es necesario ofrecerles un diferencial a los clientes. Agregó que para lograr la escalabilidad es importante quitarles el estado a las aplicaciones, de manera que los componentes puedan ir y venir según las necesidades.

Como ejemplo de procesamiento en tiempo real de flujos de datos, mencionó un proyecto con el IDEAM para procesar la información de las cerca de 2400 estaciones meteorológicas automáticas y semiautomáticas instaladas en el país. El número de sensores en cada una varía entre 8, 16 y 24 y el reto está en la cantidad de registros, pues cada mes se generan miles de millones de estos.

Los consejos de Oracle


David Córdoba

David Córdoba, arquitecto principal para *cloud* en Oracle Colombia, habló de "La nube como habilitador de la evolución de las organizaciones". Citó cifras según las cuales el 80 % de los líderes de negocio creen que hacer una inversión estratégica en TI tiene un impacto positivo, pero 66 % de ella se destina a la ejecución, 20 % a atenderlo y solo 14 % a transformarlo.

Mencionó que entre los motivos por los cuales una empresa quisiera migrar aplicaciones a la nube pública están aumentar el desempeño del negocio, reducir el *time to market,* ser ágiles usando elasticidad y autoaprovisionamiento y mejorar la productividad de empleados y *partners* con herramientas colaborativas de nube,

como software como servicio (SaaS). También buscan optimizar los costos con un modelo de bajas inversiones, que aproveche sus características de movilidad, autoservicio y capacidad ilimitada.

Algunas recomendaciones de Córdoba son:

- Mirar cuál es la situación actual del negocio: si las sedes están distribuidas geográficamente, los aspectos regulatorios, las metas (si quiere agilidad, si va sacar nuevos productos, si la estrategia es de movilidad), así como el balance riesgo-beneficio de llevar la aplicación a la nube y cuáles procesos o áreas del negocio se van a impactar.
- Tener en cuenta que el futuro digital requiere proveer soluciones innovadoras, acordes con las megatendencias, y cumplir con requisitos de seguridad, de integración, con aplicaciones que se tengan *on-premise* (instaladas en los servidores del usuario), o en las nubes privadas para migrarlas a la nube pública. Es decir, hay que tener una oferta completa que facilite las migraciones transparentes y que garantice la autoescalabilidad y el balanceo entre ambas nubes.
- Si hay disponibilidad y presupuesto, pero no tiempo para desarrollar una nueva aplicación, lo recomendable es tener una tipo SaaS (software como servicio) que pueda integrarse en el futuro. Además, es importante contar con el *backup* semanal o mensual como característica del sistema de administración.
- Para hacer aplicaciones escalables en la nube, se debe usar PaaS (*Platform as a Service*). Esto permite consolidar e integrar los datos de forma sincronizada y se aconseja usar bases de datos SQL.
- Usar tecnología que escale horizontal y verticalmente, desde el *on-premise* al *cloud,* apalancándose en las ventajas de la nube privada o de la nube pública.
- Seleccionar el software de base de datos según el tipo de aplicativo. Si son aplicaciones con foco de consulta solamente, pueden utilizar base de datos NoSQL pero si son transaccionales y misionales, mejor optar por bases de datos relacionales.

Microsoft y la elasticidad


Nicolás Jiménez

Nicolás Jiménez, consultor *senior,* arquitecto de nube en Microsoft, dijo que en su empresa prefieren hablar de elasticidad y no de escalabilidad, porque no solamente es crecer sino volver a un estado pequeño y pagar por lo que se consume.

Recordó que los clientes manejan infraestructura, servidores, espacio físico, pagos por servicios y administración, parcheo, instalación de las aplicaciones y demás, y las plataformas en la nube pueden aliviar los temas operativos para centrarse en el negocio.

Comparó el empaquetamiento de los componentes y su escalabilidad con un juego de Lego cuyas fichas sirven para determinados propósitos y hay que saber usarlas porque hacen parte de rompecabezas exactos.

Como ejemplo de lo que pueden hacer en escalamiento citó el manejo de las estadísticas y medallería de los Juegos Olímpicos de Invierno de Sochi (Rusia), una carga de más de 100 millones de visitantes de todo el mundo que solo puede gestionarse con automatización. La ventaja es que el sistema es elástico y cuando se empieza a perder carga, se puede desaprovisionar para pagar solo por el consumo.

Entre sus recomendaciones está desacoplar procesos en las arquitecturas para mejorar la velocidad y ganar tiempo. También llamó la atención sobre cómo las cadenas se rompen por el lazo más débil y la arquitectura de las aplicaciones suele tener mala definición en temas de identidad. Por eso, aconsejó optar por tener directorios activos que se sincronicen como un servicio.

La posición de los integradores


Yuji Kiriki Rodríguez

Yuji Kiriki Rodríguez, líder técnico de Seven4N, señaló que las aplicaciones se dividen en dos: las que tienen estado y las que no lo tienen. Estas últimas corresponden a la matemática pura, como 1 + 1 = 2, mientras que las primeras tienen un *side effect*, o efecto de borde, lo que significa que el dato se guarda en una base de datos (por ejemplo, el resultado de la suma se almacena en la memoria de la calculadora, o el carrito de compras se conserva para que el manejo de los datos sea eficiente).

Precisó que esa diferencia se traduce en que es fácil escalar las aplicaciones sin estado, pero no sucede igual con las otras, pues no hay certeza de que van a poderse soportar los datos, pese a que los proveedores prometen que la nube permite escalabilidad de cargas dinámicas.

Kiriki agregó que un problema se deriva de que en la nube no se puede mantener simultáneamente 100 % de disponibilidad y consistencia cuando se escalan horizontalmente aplicaciones con estado o cuando se busca elasticidad. Es decir, hay que disminuir la exigencia en una de las dos condiciones o empezar a graduarlas para encontrar un punto intermedio.

Además, hay que tener en cuenta que las posibilidades de falla en un centro de datos son múltiples y la situación se complica al tratar de garantizar la consistencia entre las

> "Al desacoplar procesos en las arquitecturas se puede mejorar la velocidad y ganar tiempo".
>
> **Nicolás Jiménez, Microsoft**

réplicas, pues es difícil mantener el control. Explicó que cuando se cae el líder del sistema, que es el que está recibiendo las peticiones, cualquiera de las réplicas lo remplaza mientras se restablece su función, pero surge la dificultad de cómo pasar al que se cayó la información que siguió recogiendo el sistema mientras duró la falla. Esto obedece a que en los sistemas distribuidos no existe la noción de tiempo, no hay un orden absoluto, sino parcial.

Para solucionarlo, analizaron el funcionamiento de aplicaciones exitosas como el contador *Like* de Facebook o la plataforma de Googledocs (que permite a varios usuarios modificar simultáneamente un documento). En ellas hay un común denominador: son estructuras matemáticas asociativas que conservan un orden parcial dado, propiedades que pueden usarse para hacer *merges* automáticos y deterministas que garanticen que la última versión que saldrá de esa convergencia es correcta y, por ende, consistente.

Intergrupo y las pruebas de carga

Armando Quiñones y Giovanny Saray, arquitectos de soluciones de Intergrupo, se centraron en los retos y dificultades de hacer las pruebas de carga de aplicaciones que se deben desplegar en la nube soportando muchos usuarios versus hacerlas en ambientes *on-premise*. En ambos casos, uno de los mayores desafíos en desarrollo de software por demanda es lograr aplicaciones suficientemente robustas para las necesidades de los clientes.

Armando Quiñones

De su experiencia *on-premise* resaltaron lo siguiente:

- La infraestructura de algunos clientes sigue siendo física, no está virtualizada. Así, puede haber demoras para conseguir las máquinas en que se van a hacer las pruebas de carga, aunque el equipo de desarrollo esté listo.
- No siempre es posible aislar el ambiente de pruebas para dedicarlo exclusivamente a esa tarea, lo que puede ocasionar que, como destacó el ingeniero Castro, uno de los propósitos de las organizaciones cuando usan la nube es crecer su negocio.

Para la investigación y dada la complejidad del tema, decidieron centrase en cómo sería la aplicación para dos servicios básicos: 1. generar los parámetros para el plan de pagos con un tiempo de respuesta muy corto y 2. almacenar ese plan de pagos en la nube para que pueda consultarse en el futuro.

Para ello, analizaron la forma en que grandes empresas como Netflix y LinkedIn, que tienen millones de usuarios en todo el mundo, enfrentan los problemas de arquitectura (costos, escalabilidad y elasticidad, entre otros) y encontraron que el denominador común es el microservicio. Es decir, hay que desacoplar funcionalidad y desarrollar no una sino muchas aplicaciones monolíticas o de único repositorio.

En su exposición, el profesor Castro mostró los distintos tipos de arquitectura propuestos para cada uno de los dos servicios escogidos con las instancias probadas y los efectos de latencia en cada una, los cambios en el *framework,* los lenguajes usados, el tipo de balanceadores de carga, los costos, los beneficios y otros aspectos técnicos. Además, mencionó que las están probando con distintos proveedores como Heroku, Amazon y Google.

Avances de una investigación academia-industria

Santiago Gil, gerente del área de arquitectura de Heinsohn, y Harold Castro, director del DISC de Uniandes, presentaron los primeros avances de un proyecto conjunto de investigación sobre arquitectura orientada a microservicios.

Heinsonh es una empresa colombiana con 35 años en el mercado que provee servicios de *cloud computing* a terceros y se interesa por desarrollar tecnologías maduras en ese sector, razón por la cual se alió con Los Andes en esta iniciativa (ver revista Foros ISIS # 5, págs. 46-52 https://sistemas.uniandes.edu.co/es/revista-foros-isis-5).

La investigación responde a la tendencia empresarial de desentenderse del manejo y administración de la infraestructura de TI y busca ofrecer a las organizaciones respuestas sobre cómo pasar del ámbito *on-premise* a la nube. El estudio se enmarca en el software como servicio (SaaS) y busca desarrollar una arquitectura para modelar créditos y simular los planes de pago. En el modelo de negocio propuesto ya no se va a vender la licencia para usar ese software, ni su instalación, sino que se va a cobrar por el uso.

De ahí que una pregunta central era cómo hacer que el cliente pueda pagarle al proveedor lo mínimo posible, al tiempo que atiende al mayor número de usuarios. Y además, pretenden que su solución sea apta para atender no solo al mercado colombiano, sino el de Latinoamérica, por-

Santiago Gil, gerente del área de arquitectura de Heinsohn, señaló que la alianza con universidades les da robustez a los productos y servicios de la compañía.

Harold Castro, director del DISC, explicó que el estudio que adelantan con Heinsonh busca desarrollar una arquitectura para modelar créditos y simular los planes de pago.


Giovanny Saray

sionar ruidos y distorsionar los resultados.

- Las pruebas son cíclicas: se lanza la carga, se toman los resultados, se miden, se hacen ajustes y se repite la operación. A veces se necesitan permisos para hacer los despliegues, monitorear, instalar las herramientas o analizar los resultados y no siempre es fácil conseguir las autorizaciones con el área de seguridad.
- No todas las máquinas requeridas tienen la potencia o la conectividad suficientes para simular un número *x* de usuarios.

Quiñones y Saray explicaron que usan la nube como infraestructura como servicio (IaaS, *Infrastructure as a Service*) para generar escenarios de pruebas de carga sin las limitaciones de los ambientes físicos y locales. En ella se pueden montar ambientes robustos de manera sencilla y rápida al identificar grandes balanceadores, recursos y conexiones de redes virtuales y servidores de dominio. También es posible usar las plantillas dinámicas virtuales que ofrecen los proveedores de producto, que incluyen software base y sistemas operativos diversos. Y es posible utilizar canales de redes mixtos o recursos compartidos. Esta sumatoria permite el escalamiento horizontal y vertical con la ventaja de pagar solo por lo consumido, lo cual significa un buen costo-beneficio. Además, las mediciones de aspectos como memoria, procesamiento y consumo de red son más acertadas, incluso cuando se hacen pruebas de estrés para establecer el máximo de carga que podrá soportar esa arquitectura.

También insistieron en que es importante entender y determinar el uso esperado mediante un análisis previo para definir parámetros sobre aspectos como número de usuarios y cantidad de peticiones y respuestas en un lapso preciso.

Con el fin de asegurar la escalabilidad recomendaron usar herramientas para automatizar la escalabilidad. Para ello basta seguir normas sencillas como usar la menor cantidad de peticiones posible, ajustar el tamaño de las imágenes para que pesen poco sin perder calidad o utilizar solo uno o dos archivos de *script* que los combinen todos.

Como recomendación final, Armando Quiñones y Giovanny Saray dijeron que el diseño debe estar pensando en fallas porque en la nube no se puede garantizar que todo funcionará correctamente. ■

Recomendaciones de los expertos para **escalar y ser elásticos**

¿Queda un usuario amarrado para siempre a un proveedor cuando se pasa a la nube? ¿Cuál sería la primera pregunta de los implementadores al cliente que quiere pasar sus aplicaciones a *cloud*? ¿Qué debe enseñárseles a los desarrolladores para que sus aplicaciones sean escalables y flexibles? Estas fueron algunas preguntas del público en el 9.º Foro de *Cloud Computing*. En el panel también intervino Julio Faerman, de Amazon.

Revista Foros ISIS recoge algunas respuestas a la preguntas del público en el panel del encuentro "Arquitectura *cloud:* estrategias para hacer aplicaciones escalables en la nube".

Sobre dependencia e independencia de los proveedores

- Los proveedores de servicios en la nube ofrecen múltiples opciones que facilitan tener aplicaciones independientes. Los usuarios deben escoger si el beneficio compensa el costo. Muchas de las librerías están disponibles en casi todos los lenguajes de programación para que los equipos de desarrollo sigan sacándole provecho a su inversión sin necesidad de aprender nuevas tecnologías.
  Julio Faerman, Amazon, y Nicolás Jiménez, Microsoft
- Tener dependencia o independencia no es de por sí positivo o negativo. Lo importante es pensar qué valor agrega al negocio y cómo sacarle beneficios.
  Julio Faerman, Amazon

Sobre la pregunta de los implementadores a los clientes

- ¿La aplicación que tienen en un ambiente *on-premise* está diseñada para desplegarse en la nube? Para obtener elasticidad y escalabilidad automatizadas, la aplicación tiene que tolerar fallos y sus componentes deben estar lo más desacoplados posible. Si no, habría que ajustar la arquitectura para obtener costo-beneficio.
  Giovanny Saray, Intergrupo
- ¿Qué le duele, cuál es la razón de fondo para querer el cambio? Usualmente responden que el costo, que su infraestructura no es tan veloz como quisieran y casi