Auditoría de procesos de negocio en la nube: persistencia mediante almacenes no relacionales

M. Cruz, B. Bernárdez, M. Resinas, A. Durán

Departamento de Lenguajes y Sistemas Informáticos,
Universidad de Sevilla
{cruz, beat, resinas, amador}@us.es

Abstract. Cada día crece el número de aplicaciones y servicios *basados en la nube* ofertados por proveedores tales como Amazon, Google o Sun entre otros. Además de ofrecerse el software como servicio (*SaaS, Software as a Service*), destacan los sistemas de procesos de negocio que se ofrecen a los clientes como servicio, denominados PRaaS (*Process as a Service*).

Uno de los problemas que conlleva la computación en la nube (*cloud computing*) es la pérdida de control sobre los datos y procesos que hace necesario la realización de auditorías que permitan además verificar el grado de cumplimiento de los procedimientos y regulaciones establecidas por la organización. Como resultado de este proceso de auditoría, es habitual que el volumen de datos se incremente considerablemente en poco tiempo por lo que la escalabilidad será uno de los requisitos a exigir al sistema de almacenamiento subyacente.

En este trabajo se plantea una posible línea de investigación basada en el uso de sistemas de bases de datos no relacionales para dotar de persistencia a los sistemas PRaaS persiguiendo el principal objetivo de mejorar la escalabilidad de los mismos.

Keywords: PRaaS, bases de datos no relacionales, cloud computing, compliance management, escalabilidad

1 Introducción

Actualmente está aumentando el número de organizaciones que aprovechan la tecnología de computación en la nube. Esta tecnología permite utilizar únicamente los recursos necesarios en cada momento lo que produce una reducción de los costes ya que únicamente se paga por los recursos que se utilizan.

Los servicios y aplicaciones ofrecidos para su contratación mediante computación en la nube son variados, por ejemplo, software como servicio, infraestructura como servicio, etc. Actualmente además las empresas pueden contratar la gestión de los procesos de negocio, lo que se conoce como PRaaS (*Process as a Service*).

Uno de los problemas de la computación en la nube es la pérdida de control que se produce sobre los datos y ejecuciones [7]. Centrándonos en el PRaaS ese control es fundamental cuando las organizaciones se someten a auditorías de la conformidad de

A. Ruíz, L. Iribarne (Eds.): JISBD'2012, pp. 711-716, ISBN:978-84-15487-28-9.
Jornadas SISTEDES'2012, Almería 17-19 sept. 2012, Universidad de Almería.

los procesos desplegados en su sistema [1]. El propósito de estas auditorías de conformidad es determinar si el auditado actúa, o ha actuado, de acuerdo a los procedimientos y regulaciones establecidas por una autoridad, por ejemplo: leyes (del tipo de Sarbanes-Oxley en EEUU [21], que regula las funciones financieras contables y de auditoría, o la LOPD en España), buenas prácticas, o recomendaciones (como ITIL [22] o EFQM [10]) [3].

El resto del artículo se organiza de la siguiente manera. En la sección 2 se identifican los principales problemas de la conformidad en los PRaaS. En la sección 3 se introducen algunos conceptos y en la sección 4 se describe el enfoque de la investigación que se pretende abordar.

2 Principales aspectos por resolver en la conformidad de los PRaaS

Aunque existe una importante cantidad de trabajo realizado en el estudio de la conformidad [2, 9,12], aún queda trabajo por hacer al respecto:

Por un lado, la mayoría de las aproximaciones para la comprobación de la conformidad se centran en un tipo concreto de regla de conformidad, por ejemplo las relacionadas con flujo de control, y con un momento concreto de comprobación de la conformidad: antes [2, 8], durante [4] y después [11] de la ejecución de un proceso de negocio. Sin embargo, hasta donde sabemos, no se ha definido aún un sistema de gestión de la conformidad que de soporte al ciclo de vida completo de los procesos de negocio [6].

Por otro, para hacer posible las auditorías de conformidad es necesario dejar un registro de cómo transcurren los procesos de negocio generando evidencias que permitan verificar la conformidad posteriormente. Las evidencias son habitualmente recogidas en forma de *logs* de eventos y de datos manejados y obtenidos como resultado de las distintas actividades del proceso de negocio.

Hasta el momento para dotar de persistencia a los datos generados, lo más habitual es utilizar sistemas de gestión de bases de datos relacionales para hacer frente a una gran cantidad de datos generados [5]. Los sistemas de procesos de computación en la nube deben soportar un gran número de usuarios y alta carga de transacciones lo que puede provocar que haya muchas instancias con un gran volumen de datos ejecutándose simultáneamente que hacen necesaria la escalabilidad.

Los sistemas relacionales presentan el problema de no escalar fácilmente por lo que al aumentar la carga de los mismos el rendimiento puede disminuir de forma exponencial. Habría que estudiar la manera más adecuada para almacenar los datos persiguiendo una implementación fácilmente escalable.

Desde hace unos años han surgido los sistemas NoSQL [20] como un nuevo paradigma en la gestión de datos para dar respuesta a las nuevas necesidades de almacenamiento de datos para sistemas web, redes sociales, juegos online, computación en la nube, etc. Dado que una de sus principales características es la escalabilidad parece que se podrían adaptar bien a los sistemas de conformidad de los PRaaS.

Existen ya trabajos [23] en los que se aplican bases de datos NoSQL para mejorar la escalabilidad en sistemas que utilizaban sistemas relacionales. En dicho trabajo se propone una solución de persistencia mediante una base de datos NoSQL orientada a documento para sistemas de desarrollo software basado en modelos (MDE).

3 Background

En esta sección se definen los principales conceptos necesarios para comenzar a abordar la investigación y se resumen las propiedades de las bases de datos NoSQL.

3.1 Escalabilidad

La escalabilidad es la capacidad de un sistema de atender un número creciente de peticiones y seguir cumpliendo con los requisitos impuestos. La forma de conseguir la escalabilidad puede ser vertical u horizontal [18]. Para conseguir escalabilidad vertical es necesario añadir recursos tales como CPU, memoria, etc. al servidor pero el coste va aumentando de forma exponencial y llega un momento en el que ya no puede seguir escalando. La escalabilidad horizontal consiste en añadir más servidores en paralelo para distribuir los datos. Algunos sistemas proporcionan a la vez escalabilidad horizontal y vertical.

Los sistemas de bases de datos relacionales almacenan los datos en tablas interrelacionadas; para las consultas necesitan combinar tablas que pueden estar en distintos servidores [19]. Los sistemas de NoSQL no presentan este problema ya que los datos no se almacenan normalizados y pueden estar distribuidos en varios servidores según el valor de la clave. Estos sistemas escalan con facilidad en forma horizontal.

3.2 Propiedades ACID

Otra diferencia importante entre los sistemas relacionales y los sistemas de almacenamiento NoSQL es relativa al cumplimiento de las propiedades ACID.

Las BD relacionales soportan el concepto ACID [19]. Sus transacciones tienen que cumplir los requisitos de **A**tomicidad, **C**onsistencia, a**I**slamiento y **D**urabilidad.

Sin embargo, los almacenamientos de datos NoSQL se caracterizan por su escalabilidad y disponibilidad. Estos dos objetivos se consiguen mediante partición (datos distribuidos en varios servidores para facilitar la escalabilidad) y replicación (datos duplicación en varios nodos o servidores para conseguir que el sistema esté siempre disponible)[15].

Como consecuencia de las necesidades de escalabilidad y disponibilidad se hace necesario relajar las condiciones ACID [18].

Eric Brewer presenta en el año 2000 el teorema CAP [14, 13], en el que se propone que para un sistema distribuido, sólo dos de las siguientes tres propiedades pueden ser satisfechas a la vez:

- Consistencia *(**C**onsistence)*. Cuando un dato esté replicado en varios nodos, su valor debe ser el mismo en todos los nodos en que esté replicado. Es equivalente a la C de ACID. Afectará a la disponibilidad.

- Disponibilidad *(**A**vailability)*. El sistema siempre es capaz de responder. Si se aumenta la disponibilidad disminuye la consistencia.
- Tolerancia a fallos *(**P**artition-tolerance)*. El sistema debe seguir funcionando aunque se pierdan de forma arbitraria algunos mensajes debido a fallos de alguna parte del sistema.

Cuando se produce una actualización de datos, para conseguir que haya consistencia, será necesaria su propagación al resto de las réplicas, por lo que la disponibilidad puede verse afectada temporalmente. En muchas de las implementaciones de NoSQL lo que se abandona es la consistencia como tal, garantizando que con el tiempo la base de datos sí será consistente *(eventually consistent)*. Implica que con el tiempo los datos estarán actualizados en todos los nodos [17].

Aunque la mayoría de las aplicaciones necesitan transacciones que cumplan ACID, hay sistemas como por ejemplo redes sociales en las que no es crítico que algún mensaje no se visualice en forma instantánea.

3.3 Características de los sistemas NoSQL

Una posible clasificación de los sistemas NoSQL junto con algunas de sus características se muestra en la tabla 1.

Tipo de BD NoSQL	Características de Almacenamiento	Principales ventajas	Implementaciones
BD clave-valor (*Key-value*)	Se almacenan parejas clave: valor; valores identificados por una clave.	Muy rápido para lecturas.	Dynamo (Amazon) [15], Voldemort, Tokyo Cabinet
BD de columnas (*Column store*)	Almacena los datos en columnas en lugar de filas.	Consultas de datos por columnas y agregados.	BigTable (Google) [16], Cassandra (Facebook)y HBase
BD orientadas a documentos (*Document Store*)	Similar al clave-valor pero el valor es un documento que la BD puede interpretar.	Consultas complejas sobre campos del documento que se almacena.	Apache CouchDB, MongoDB
BD orientadas a grafos (*Graph databases*)	La información se guarda en forma de grafo con nodos, arcos (conexiones) y propiedades de arcos y nodos.	Consultas sobre nodos y conexiones.	Neo4j

Tabla 1. Clasificación de las principales bases de datos NoSQL

Estas bases de datos se caracterizan por su escalabilidad y disponibilidad por lo que responden bien en situaciones de gran volumen de datos, no tienen esquema fijo de datos y la mayoría de las implementaciones son de código abierto [19].

4 Hipótesis de investigación y plan de trabajo

Tras identificar los problemas de escalabilidad de la gestión de evidencias en PRaaS, la hipótesis de partida de la investigación que se pretende someter a estudio es "Puede ser beneficioso usar una base de datos NoSQL para mejorar la escalabilidad de sistemas de gestión de evidencias en procesos que se ejecutan en la nube".

El objetivo que se pretende alcanzar es probar que al usar una base de datos NoSQL se obtienen mejoras respecto a las bases de datos relacionales en relación a variables como escalabilidad y disponibilidad.

Para comenzar la investigación una posibilidad es plantear un caso de estudio consistente en las siguientes tareas:

1. Definir una batería de procesos de negocio para su gestión en la nube.
2. Definir indicadores que permitan medir la escalabilidad, disponibilidad y otros parámetros relativos al rendimiento que se identificarán durante la investigación.
3. Definir una primera aproximación de conformidad o normas a cumplir por los procesos de negocio desplegados en la nube.
4. Registrar las evidencias que se generan durante la instanciación del proceso de negocio. La idea esencial es desplegar los procesos de negocio en una base de datos NoSQL y en un sistema de bases de datos relacional.
5. Medir y comparar el comportamiento de ambas bases de datos respecto a las variables identificadas.

Se espera que los resultados del caso de estudio proporcionen realimentación para seguir avanzando. En concreto para poder estudiar qué base de datos NoSQL es más conveniente según las características de los procesos de negocio o la naturaleza de las reglas de conformidad fijadas por la auditoría.

Referencias

1. R. Accorsi. Business Process as a Service. Chances for Remote Auditing. IEEE 35th Annual Computer Software and Applications Conference Workshops, pag. 398–403. IEEE-2011.
2. A. Awad, G. Decker, and M. Weske. Efficient compliance checking using bpmn-q and temporal logic. In BPM, pag. 326–341, 2008.
3. J. Bace and C. Rozwell. Understanding the components of compliance. Gartner Research Paper, 2006.
4. A. Birukou, V. D'Andrea, F. Leymann, J. Serafinski, P. Silveira, S. Strauch, and M. Tluczek. An integrated solution for runtime compliance governance in soa. In ICSOC, pag. 122–136, 2010.
5. J. Dean and S. Ghemawat. MapReduce. Communications of the ACM, 51(1):107, 2008.
6. C. Cabanillas, M. Resinas, and A. R. Cortés. Exploring features of a full-coverage integrated solution for business process compliance. In C. Salinesi and O. Pastor, editors, CAiSE Workshops, volume 83 of Lecture Notes in Business Information Processing, pag. 218–227. Springer, 2011.

7. R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina. Controlling data in the cloud: outsourcing computation without outsourcing control. In Proceedings of the 2009 ACM workshop on Cloud computing security, pages 85–90. ACM, 2009.
8. Y. Liu, S. Müller, and K. Xu. A static compliance-checking framework for business process models. IBM Systems Journal, 46(2):335–362, 2007.
9. K. Namiri and N. Stojanovic. Using control patterns in business processes compliance. InWeb Information Systems Engineering - WISE 2007Workshops, pag. 178–190, 2007.
10. S. Rinderle-Ma, L. T. Ly, and P. Dadam. Business process compliance. EMISA Forum, 28(2):24–29, 2008.
11. A. Rozinat and W. M. P. van der Aalst. Conformance checking of processes based on monitoring real behavior. Information Systems, 33(1):64–95, 2008.
12. S. Sadiq, G. Governatori, and K. Namiri. Modeling control objectives for business process compliance. In Business Process Management, Volume 4714, pag. 149–164, 2007.
13. S. Gilbert, N. Lynch. Brewer's. Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services. ACM SIGACT News, 33(2):51-59, 2002.
14. E. Brewer. Towards Robust Distributed Systems, Proceedings of Principles of Distributed Computing (PODC 2000), 2000.
15. G. DeCandia y otros. Dynamo. Amazon's Highly Available Key-value Store. Proceedings of symposium on Operating systems principles (SOSP), pag. 205-220, 2007.
16. F. Chang y otros. Bigtable. A Distributed Storage System for Structured Data. Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation, 2006.
17. http://www.allthingsdistributed.com/2008/12/eventually_consistent.html, 2012.
18. R. Cattell. Scalable SQL and NoSQL Data Stores. SIGMOD Rec. 39(4):12-27, 2010.
19. N. Leavitt. Will NoSQL Databases Live Up to Their Promise?. IEEE Computer Society 43(2):12-14, 2010.
20. D. McCreary. The CIO's Guide to NoSQL, 2011. https://s3.amazonaws.com/dataversity/documents/CIO_Guide_NoSQL_v4.pdf, 2012.
21. S. Bainbridge. Complete Guide to Sarbanes-Oxley : Understanding How Sarbanes-Oxley Affects Your Business, 2007.
22. L. Klosterboer. ITIL Capacity Management. IBM Press, 2011.
23. J. Espinazo, J. Sánchez, J. García. Morsa: A Scalable Approach for Persisting and Accessing Large Model http://www.springerlink.com/content/8264815854324251/fulltext.pdf