



Arquitecturas de microservicios: Accounting y Transaccionalidad

📅 18/12/2015 (<http://blog.gfi.es/arquitecturas-de-microservicios-accounting-y-transaccionalidad/>) 👤 Elías Grande Rubio (<http://blog.gfi.es/author/egrande/>) ➦
Big Data (<http://blog.gfi.es/category/big-data/>), Metodología (<http://blog.gfi.es/category/metodologia-2/>), SOA (<http://blog.gfi.es/category/soa/>)

★★★★★ (2 votos, promedio: 5,00 de 5)



([https://www.facebook.com/sharer/sharer.php?s=100&p\[url\]=http://blog.gfi.es/arquitecturas-de-microservicios-accounting-y-transaccionalidad/](https://www.facebook.com/sharer/sharer.php?s=100&p[url]=http://blog.gfi.es/arquitecturas-de-microservicios-accounting-y-transaccionalidad/))

Twitter

Guardar (<https://es.pinterest.com/pin/create/button/?guid=5EmXoDw9pk9h-1&url=http%3A%2F%2Fblog.gfi.es%2Farquitecturas-de-microservicios-accounting-y-transaccionalidad%2F&media=http%3A%2F%2Fblog.gfi.es%2Fwp-content%2Fuploads%2F2015%2F11%2FArquitecturas.jpg&description=Arquitecturas%2Bde%2Bmicroservicios%3A%2BAccounting%2B%2BTransaccionalidad>)

Si eres nuevo por aquí, puede que quieras suscribirte al RSS feed (http://feeds.feedburner.com/GFI_Blog). ¡Gracias por visitarnos!

Imagen: Bald Eagle by Luis Alberto Fernández Arnanz (<https://500px.com/photo/14712607/bald-eagle-by-luis-alberto-fern%C3%A1ndez-arnanz>)

Cuando nos enfrentamos a la implantación de una arquitectura orientada a microservicios, debemos tener en cuenta dos de los principales problemas inherentes dentro de este tipo de arquitecturas: la gestión de log y la transaccionalidad en las operaciones.

Como ya hemos visto con anterioridad, los microservicios cumplen con el principio de responsabilidad única (<http://blog.gfi.es/microservicios-y-esb-amigos-o-enemigos/>), lo cual implica que dentro de nuestra infraestructura dispondríamos de una cantidad bastante considerable de este tipo de aplicativos diferentes generando log. Esta problemática unida al posible tiempo de vida de los distintos microservicios y a su gestión entre los distintos entornos (desarrollo, preproducción y producción), convierte la gestión de log en algo prioritario dentro de estas arquitecturas.

Por otro lado, los servicios por definición deberían ser *stateless* (sin estado), y por consiguiente, los microservicios también. Esto permite que la escalabilidad horizontal de nuestros microservicios sea fácil de llevar a cabo. Sin embargo, la escalabilidad horizontal en servicios sin estado plantea otro problema mayor que la centralización de log: la transaccionalidad en las operaciones y la consistencia de los datos. ¿Quién me asegura que no existan condiciones de carrera entre los microservicios que acceden a bases de datos y me provoquen inconsistencias en los datos? Y si eso pasa, ¿Cómo gestiono la transaccionalidad de mis operaciones si mi arquitectura de microservicios está planteada *stateless* para que sea altamente escalable?

Accounting (Centralización de logs)

Por cada entorno (desarrollo, preproducción y producción) nos encontramos con una fuente de log inagotable: generado por los aplicativos, generado por los contenedores ligeros y JVMs, y del propio servidor, ya sea físico o virtualizado. Además, tenemos que recordar que nuestra arquitectura está dividida en microservicios, por lo que una misma petición puede requerir múltiples de estos servicios para ser completada.

Para comenzar a dar solución a este problema, lo primero que se debería de hacer sería definir un mensaje estándar dentro de nuestra infraestructura que contuviera los metadatos necesarios para identificarlo dentro del ámbito de una operación que pertenece a un usuario. De esta forma, aunque nuestra plataforma de microservicios sea *stateless*, los mensajes en vuelo dentro de ella irían identificados en todo momento por el usuario que realiza la solicitud y un identificador autogenerado de dicha operación en curso. Un ejemplo de solicitud bastante simple podría ser la que se muestra en la siguiente figura:

```
{
  "requestMetadata": {
    "userId": "egrande",
    "operationId": "e14a3ff5b5e67ede599cac94358e1028"
  },
  "requestBody": {}
}
```

(<http://blog.gfi.es/wp-content/uploads/2015/11/requestMetadata.png>)

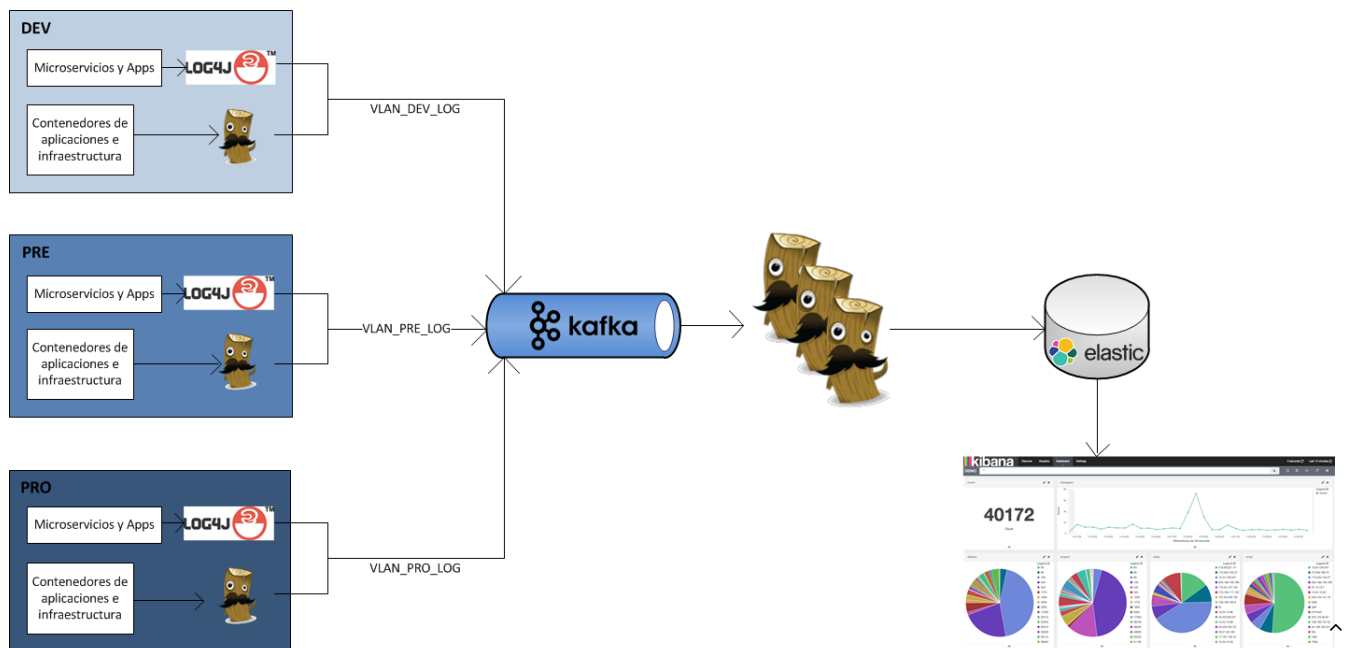
Una vez disponemos de mensajes auto-identificados dentro de nuestra plataforma, podemos apoyarnos en Log4j2 (<http://logging.apache.org/log4j/2.x/>) y Logstash (<https://www.elastic.co/products/logstash>) para generar un mensaje estándar de log en formato JSON idéntico para todos nuestros entornos. Una buena práctica sería incluir dicho mensaje de log, además de los datos identificativos del mensaje en vuelo, el entorno desde el que se genera el mensaje de log, la IP de la máquina y el tipo/nombre de microservicio.

Una vez definido nuestro mensaje de log estándar dentro de nuestra plataforma, el siguiente paso sería realizar una gestión de log centralizada de todos los entornos. Para llevar a cabo esta centralización, tendríamos que tener en cuenta que no debemos provocar un DoS en el sistema que lo centralice, ni tampoco provocar que las redes que dan servicio se saturen de mensajes de log, evitando de este modo, que nuestra plataforma proporcione servicio útil a nuestros usuarios.

Con vistas a evitar la saturación de nuestras redes de servicio, una buena práctica sería crear distintas VLANs (virtual LAN) con el fin de disponer de redes lógicas independientes de las de servicio a través de las cuales sólo se enviarán mensajes de log. De este modo, se evitará ruido dentro de las redes de servicios, y por otro lado, se podrán identificar mucho más fácilmente patrones de mensajes anómalos en las VLANs de log, ya que todo el tráfico que fluye por dichas redes debería seguir el estándar de mensaje unificado de log definido con anterioridad.

Finalmente, y una vez hemos definido el medio por el cual viajarán los mensajes de log, faltaría solventar el problema del DoS en el sistema que centralice los log de toda nuestra infraestructura. Para ello, una de las múltiples opciones posibles sería la de apoyarnos en una gestión de mensajes asíncrona basada en colas. Es en este punto cuando Apache Kafka (<http://blog.gfi.es/flume-kafka-spark-y-storm-un-nuevo-ejercicio-apache/>) sería la mejor opción para centralizar todos estos mensajes en un sistema único, con vistas a ser consumido en un flujo constante por el agente (o agentes) encargado de persistir toda esta información de log en cualquier base de datos NoSQL.

Un diseño a alto nivel de la infraestructura planteada de centralización de log sería la que se muestra en la siguiente figura. En ella, se ha dibujado como consumidores unos indexers de Logstash que vuelcan los datos en un Elasticsearch y que, posteriormente, se visualizan de manera estática en Kibana (<https://www.elastic.co/products/kibana>). Uniendo Kibana y Shield (<https://www.elastic.co/products/shield>) podríamos disponer de una visualización de los distintos dashboard en función de los entornos y de los roles que tienen permiso de visualización de dichos dashboard, evitando así por ejemplo, que los log de producción sean visualizados por personas que no deberían.



(<http://blog.gfi.es/wp-content/uploads/2015/11/AccountingMicroservices.png>)

Si quisiéramos dar un paso más allá, podríamos jugar con los “*consumer group*” de Apache Kafka para que los mensajes fueran simultáneamente consumidos, tanto por los indexers de Logstash con el fin que ya hemos comentado, así como por sistemas de Fast-Data como Apache Storm (<http://blog.gfi.es/flume-kafka-spark-y-storm-un-nuevo-ejercito-apache/>) con el fin de procesar en tiempo real toda esa información de log centralizada.

Transaccionalidad en las operaciones y consistencia de los datos

Cuando por necesidad de nuestro negocio necesitamos transaccionalidad en las operaciones dentro de nuestra arquitectura de microservicios, al ser éstos servicios *stateless*, se nos plantea un problema bastante serio que debemos resolver en tiempo de diseño. Dicho problema se divide a su vez en dos partes: cómo gestionar la transaccionalidad de las operaciones y cómo evitar los problemas devenidos del paralelismo sobre la capa de persistencia, como por ejemplo, el problema de las lecturas sucias y las escrituras perdidas.

En primer lugar, hay que identificar claramente que microservicios estarían afectados por el arco de la transaccionalidad. Este arco nunca debería abarcar más microservicios de los imprescindiblemente necesarios excluyendo, por ejemplo, invocaciones a servicios de consulta que no modifican datos o servicios de cómputo o enrutamiento.

Una vez disponemos de la información de qué microservicios estarán involucrados en el arco de la transaccionalidad de cada operación en concreto, llega el momento de decidir cómo gestionar dicha transaccionalidad. Para ello se pueden emplear muchas técnicas y herramientas, entre ellas, las siguientes tres posibles soluciones que pueden adecuarse mejor o peor dependiendo de la problemática y necesidades de cada caso.

- La primera opción es la más obvia y consistiría en apoyarse en cualquier motor de orquestación que gestione transaccionalidad utilizando, por ejemplo, BPEL o BPM. Esta opción plantea romper con la filosofía de los microservicios ya que dichos motores de BPEL/BPM se apoyan en esquemas en base de datos y no suelen ser tan fácilmente escalables horizontalmente como los propios microservicios. Sin embargo, al apoyarse dichos motores sobre bases de datos, son capaces de gestionar el estado de las peticiones en vuelo más allá de los reinicios del servidor, pudiendo retomar el flujo por donde se quedase, una vez vuelva a estar el servidor arriba. Esto permite gestionar las acciones de compensación o rollback de las operaciones que hayamos definido con mucha facilidad.
- La segunda opción sería la de utilizar el patrón de ESB o los propios microservicios apoyados en una gestión de estado asíncrona basada en colas. El ESB y los microservicios, por definición son *stateless*, lo cual plantea la necesidad de persistir su estado de peticiones en vuelo en una capa de persistencia mucho más rápida que las bases de datos. Obviamente, se debería mantener la información de por dónde ha pasado y que métodos debería ejecutar la compensación en caso de fallo en el propio mensaje en vuelo de tal forma que éste fuera auto contenido (como se ha comentado en el punto de centralización de log).

Aunque esta solución se adecúa más a las arquitecturas de microservicios, al ser más escalable, plantea un mayor conocimiento y entendimiento del concepto de asincronía basada en colas y de la correlación de mensajes, a la vez que un mayor esfuerzo en desarrollo, comparado con la solución planteada en el primer punto.

- Para terminar, como tercera opción, se podría plantear un desarrollo a medida basado en cachés en memoria como pudiera ser Redis (<http://redis.io/>), Coherence (<http://www.oracle.com/technetwork/middleware/coherence/overview/index.html>), Infinispan (<http://infinispan.org/>), o directamente apoyarnos en bases de datos relacionales en memoria con gestión de transaccionalidad, como por ejemplo VoltDB (<https://voltDB.com/>).

Mientras que en las cachés en memoria habría que realizar un gran desarrollo para solventar el problema de mantener en caché los datos de las peticiones en vuelo a través de nuestra plataforma de microservicios y simular la transaccionalidad, con VoltDB podemos gestionar mediante una base de datos SQL en memoria que permite procedimientos almacenados desarrollados en Java, toda la gestión de transaccionalidad para posteriormente, volcarlo de manera desatendida en nuestra capa de persistencia real.

El principio de esta opción sería algo así como el funcionamiento de Apache Cassandra (<http://cassandra.apache.org/>), es decir, en memoria tenemos siempre el dato actualizado, habiendo almacenado la operación en la lista de tareas pendientes para que la capa de persistencia la lleve a cabo cuando pueda.

Obviamente esta opción implica un mayor desarrollo que las anteriores, pero sin embargo, el rendimiento al estar todo gestionado en memoria, es mucho mayor y permite una arquitectura de microservicios mucho más escalable que las dos anteriores.

Conclusiones

Debido a que la automatización y la escalabilidad horizontal están encauzando a las empresas hacia los entornos IaaS, PaaS y SaaS con vistas a abaratar costes y mejorar el “*time to market*”, la importancia de los mensajes/peticiones comienza a premiar por encima de los propios microservicios. Es por este motivo que se hace tan importante una correcta definición de los metadatos que acompañan e identifican unívocamente al mensaje dentro de nuestra infraestructura, y por consiguiente, es casi igual de importante que dicha traza dentro de nuestro entorno Cloud esté centralizada correctamente a la hora de buscar por dónde ha pasado o qué ha hecho un mensaje dentro de nuestro sistema.

Para terminar, hay que tener en cuenta que en el momento que nuestro negocio requiera de transaccionalidad en las operaciones y consistencia en los datos, las arquitecturas de microservicios van a plantear un reto muy interesante para ávidos arquitectos que quieran definir cuál deberá ser la mejor forma de solventar los problemas de paralelismo y gestión de transaccionalidad que dichas arquitecturas presentan. Para ello, se ha expuesto una serie de ideas que seguramente podrán servir de base para llevarlo a cabo, aunque como se ha comentado en dicho punto, cada opción planteada deberá adecuarse a la casuística y problemática concreta de cada uno, por lo que recordad que cada problema es un mundo y que en la imaginación está el límite, así que usadla con moderación.





Elías Grande Rubio

Arquitecto SOA/BPM y Consultor de Seguridad TIC pragmático e investigador. Apasionado de la seguridad, las tecnologías middleware y de la computación de altas prestaciones. Diseñador de ideas revolucionarias que me hubieran hecho rico... pero las guardé en Megaupload y ahora escribo blogs.

Elías Grande Rubio (<http://blog.gfi.es/author/egrande/>) ha escrito 7 entradas




(<http://www.linkedin.com/in/eliasgrande>)



(mailto:blog@gfi.es)


Contenidos relacionados



(<http://blog.gfi.es/flume-kafka-spark-y-storm-un-nuevo-ejercito-apache/>)


Política de cookies

Flume, Kafka, Spark y Storm: ¿un nuevo ejército Apache? (http://blog.gfi.es/flume-kafka-spark-y-storm-un-nuevo-ejercito-apache/)




(<http://blog.gfi.es/opinion-expertos-ofrece-docker-tan-popular-la-actualidad/>)

Opinión de Expertos: ¿Qué ofrece Docker para ser tan popular en la actualidad? (http://blog.gfi.es/opinion-expertos-ofrece-docker-tan-popular-la-actualidad/)




(<http://blog.gfi.es/opinion-expertos-machine-learning-la-disciplina-de-moda/>)

Opinión de Expertos: Machine Learning, la disciplina de moda (http://blog.gfi.es/opinion-expertos-machine-learning-la-disciplina-de-moda/)




(<http://blog.gfi.es/opinion-expertos-la-transformacion-la-sanidad-siglo-xxi-ya-una-realidad/>)

Opinión de Expertos: La transformación de la sanidad en el siglo XXI ¿Es ya una realidad? (http://blog.gfi.es/opinion-expertos-la-transformacion-la-sanidad-siglo-xxi-ya-una-realidad/)




(<http://blog.gfi.es/opinion-expertos-adaptarse-al-testing-agile/>)

Opinión de Expertos: ¿Cómo adaptarse al Testing Agile? (http://blog.gfi.es/opinion-expertos-adaptarse-al-testing-agile/)




(<http://blog.gfi.es/opinion-expertos-impulsar-negocio-traves-la-mejora-operacional/>)

Opinión de Expertos: ¿Cómo impulsar su negocio a través de la Mejora Operacional? (http://blog.gfi.es/opinion-expertos-impulsar-negocio-traves-la-mejora-operacional/)




(<http://blog.gfi.es/microservicios-y-esb-amigos-o-enemigos/>)

Microservicios y ESB, ¿amigos o enemigos? (http://blog.gfi.es/microservicios-y-esb-amigos-o-enemigos/)




(<http://blog.gfi.es/vision-16/>)

VISION 16 (http://blog.gfi.es/vision-16/)




(<http://blog.gfi.es/opinion-expertos-supone-implementar-big-data-partir-business-intelligence/>)

Opinión de Expertos: ¿Qué supone implementar Big Data, a partir de Business Intelligence? (http://blog.gfi.es/opinion-expertos-supone-implementar-big-data-partir-business-intelligence/)




(<http://blog.gfi.es/samur-meeting-2016/>)

SAMUR MEETING 2016 (http://blog.gfi.es/samur-meeting-2016/)



(<http://blog.gfi.es/opinion-expertos-modernizar-los-sostenible-sistema-sanitario-espanol/>)

Opinión de Expertos: Modernizar los SI para hacer sostenible el Sistema Sanitario Español (http://blog.gfi.es/opinion-expertos-modernizar-los-sostenible-sistema-sanitario-espanol/)



(<http://blog.gfi.es/opinion-expertos-se-encuentran-los-sistemas-procesamiento-eventos-realtime/>)

Opinión de Expertos: ¿Cómo se encuentran los sistemas de procesamiento de eventos en realtime? Parte I (http://blog.gfi.es/opinion-expertos-se-encuentran-los-sistemas-procesamiento-eventos-realtime/)

Load more posts

[ELK \(http://blog.gfi.es/tag/elk/\)](http://blog.gfi.es/tag/elk/) [Kafka \(http://blog.gfi.es/tag/kafka/\)](http://blog.gfi.es/tag/kafka/) [Microservicios \(http://blog.gfi.es/tag/microservicios/\)](http://blog.gfi.es/tag/microservicios/) [Storm \(http://blog.gfi.es/tag/storm/\)](http://blog.gfi.es/tag/storm/)
[Transaccionalidad \(http://blog.gfi.es/tag/transaccionalidad/\)](http://blog.gfi.es/tag/transaccionalidad/) [VoltDB \(http://blog.gfi.es/tag/voltdb/\)](http://blog.gfi.es/tag/voltdb/)

DEJA UN COMENTARIO

Tu dirección de correo electrónico no será publicada. Los campos obligatorios están marcados con *

Comentario

Nombre *

Correo electrónico *

Web

PUBLICAR COMENTARIO

☐ Notifíqueme de nuevos comentarios vía e-mail







◀ Microservicios y ESB, ¿amigos o enemigos? (<http://blog.gfi.es/microservicios-y-esb-amigos-o-enemigos/>)

LEAN-IT-IL-IZANDO ▶ (<http://blog.gfi.es/lean-il-izando/>)

Buscar...



VOTACIÓN GFI POST AWARDS 2

-  Smart Working: una forma inteligente de teletrabajar (<http://blog.gfi.es/smart-working-una-forma-inteligente-teletrabajar/>)
4.96/5 (271)
-  Plataforma G-ONCE para la relación del ciudadano y la administración (<http://blog.gfi.es/plataforma-g-once-para-la-relacion-del-ciudadano-y-la-administracion/>)
4.94/5 (167)
-  Consejos para ser un buen Diseñador UX (<http://blog.gfi.es/consejos-para-ser-un-buen-disenador-ux/>)
4.98/5 (59)
-  UX - Card Sorting (<http://blog.gfi.es/ux-card-sorting/>)
5/5 (29)
-  Estrategias de movilidad para el mundo sanitario (<http://blog.gfi.es/estrategias-de-movilidad-para-el-mundo-sanitario/>)
4.93/5 (29)
-  Smart Tourist para incrementar el gasto por turista (<http://blog.gfi.es/smart-tourist-para-incrementar-el-gasto-por-turista/>)
4.75/5 (12)

SIGUE A GFI



http://feeds.feedburner.com/GFI_Blog<http://www.linkedin.com/company/gfi-inform-tica>http://twitter.com/GFI_Informatica<http://www.facebook.com/gfiinformatica><http://plus.google.com/u/0/106686879548311513003><http://www.youtube.com/user/GFIInformatica>https://www.instagram.com/gfi_informatica/http://www.slideshare.net/GFI_Informatica

¿SABÍAS QUE?

Las herramientas de ticketing permiten registrar las tareas de los equipos y mejorarlas con la metodología Agile.

— David Esteve,
Jefe de Proyectos

ACCESO

Nombre de usuario

Contraseña

☒ Recuérdame

INICIAR SESIÓN →

¿Perdiste la contraseña? (<http://blog.gfi.es/wp-login.php?action=lostpassword>)

ENLACES CORPORATIVOS

Gfi España (<http://www.gfi.es>)

Gfi Informatique (<http://www.gfi.world>)

Infojobs | Ofertas de empleo (<http://gfi.trabajo.infojobs.net/>)

Tecnoempleo | Ofertas de empleo (<https://www.tecnoempleo.com/gfi-trabajo/>)

SUSCRIBIRME

Suscríbete a nuestra newsletter y te mantendremos informado de las últimas actualizaciones:

Email

SUSCRIBIRSE

ARCHIVOS

Elegir mes ▼



enero 2018

L	M	X	J	V	S	D
1	2	3 (http://blog.gfi.es/2018/01/03/)	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

« Dic (<http://blog.gfi.es/2017/12/>)

CATEGORÍAS

- [Administración \(<http://blog.gfi.es/category/administracion/>\)](http://blog.gfi.es/category/administracion/)
- [Banca \(<http://blog.gfi.es/category/banca/>\)](http://blog.gfi.es/category/banca/)
- [Base de datos \(<http://blog.gfi.es/category/base-de-datos/>\)](http://blog.gfi.es/category/base-de-datos/)
- [BI \(<http://blog.gfi.es/category/bi/>\)](http://blog.gfi.es/category/bi/)
- [Big Data \(<http://blog.gfi.es/category/big-data/>\)](http://blog.gfi.es/category/big-data/)
- [BPM \(<http://blog.gfi.es/category/bpm/>\)](http://blog.gfi.es/category/bpm/)
- [DevOps \(<http://blog.gfi.es/category/devops/>\)](http://blog.gfi.es/category/devops/)
- [Diseño web \(<http://blog.gfi.es/category/disenio-web/>\)](http://blog.gfi.es/category/disenio-web/)
- [eCommerce \(<http://blog.gfi.es/category/ecommerce/>\)](http://blog.gfi.es/category/ecommerce/)
- [Eventos \(<http://blog.gfi.es/category/eventos/>\)](http://blog.gfi.es/category/eventos/)
- [General \(<http://blog.gfi.es/category/general/>\)](http://blog.gfi.es/category/general/)
- [Gfi Post Awards II \(<http://blog.gfi.es/category/gfi-post-awards/>\)](http://blog.gfi.es/category/gfi-post-awards/)
- [ITIL \(<http://blog.gfi.es/category/itil/>\)](http://blog.gfi.es/category/itil/)
- [Metodología \(<http://blog.gfi.es/category/metodologia-2/>\)](http://blog.gfi.es/category/metodologia-2/)
- [Movilidad \(<http://blog.gfi.es/category/movilidad/>\)](http://blog.gfi.es/category/movilidad/)
- [Opinión de Expertos \(<http://blog.gfi.es/category/opinion-de-expertos/>\)](http://blog.gfi.es/category/opinion-de-expertos/)
- [Productos \(<http://blog.gfi.es/category/productos/>\)](http://blog.gfi.es/category/productos/)
- [Sanidad \(<http://blog.gfi.es/category/sanidad-2/>\)](http://blog.gfi.es/category/sanidad-2/)
- [SOA \(<http://blog.gfi.es/category/soa/>\)](http://blog.gfi.es/category/soa/)
- [SQA \(<http://blog.gfi.es/category/sqa/>\)](http://blog.gfi.es/category/sqa/)
- [Tecnología \(<http://blog.gfi.es/category/tecnologia/>\)](http://blog.gfi.es/category/tecnologia/)

ENTRADAS MAS VALORADAS

- UX – Card Sorting (<http://blog.gfi.es/ux-card-sorting/>) ★★★★★ (5,00 de 5)
- Diez consejos para optimizar el diseño UX en aplicaciones móviles (<http://blog.gfi.es/diez-consejos-optimizar-diseno-ux-aplicaciones-moviles/>) ★★★★★ (5,00 de 5)
- UX – Creando Personas (<http://blog.gfi.es/ux-creando-personas/>) ★★★★★ (5,00 de 5)
- Estrategias de movilidad para el mundo sanitario (<http://blog.gfi.es/estrategias-de-movilidad-para-el-mundo-sanitario/>) ★★★★★ (5,00 de 5) ^
- Arquitecturas de microservicios: Accounting y Transaccionalidad (<http://blog.gfi.es/arquitecturas-de-microservicios-accounting-y-transaccionalidad/>) ★★★★★ (5,00 de 5)

LEAN-IT-IL-IZANDO (<http://blog.gfi.es/lean-il-izando/>) ★★★★★ (5,00 de 5)

Smart Tourist para incrementar el gasto por turista (<http://blog.gfi.es/smart-tourist-para-incrementar-el-gasto-por-turista/>) ★★★★★ (5,00 de 5)

Diseño Web – Qué hacer y qué no hacer (<http://blog.gfi.es/disenio-web-que-hacer-y-que-no-hacer/>) ★★★★★ (5,00 de 5)

Arrancan las votaciones de los Gfi Post Awards 2 (<http://blog.gfi.es/arrancan-las-votaciones-los-gfi-post-awards-2/>) ★★★★★ (5,00 de 5)

Opinión de Expertos: ¿Cuál es el presente y futuro del diseño de experiencia de usuario? (<http://blog.gfi.es/opinion-de-expertos-cual-es-el-presente-y-futuro-del-diseno-de-experiencia-de-usuario/>) ★★★★★ (5,00 de 5)

NUBE DE ETIQUETAS

seguridad (<http://blog.gfi.es/tag/seguridad/>) mainframe (<http://blog.gfi.es/tag/mainframe/>) ciudad2020 (<http://blog.gfi.es/tag/ciudad2020/>) **Agile** (<http://blog.gfi.es/tag/agile/>) Apache (<http://blog.gfi.es/tag/apache/>) post awards (<http://blog.gfi.es/tag/post-awards/>) Graylog (<http://blog.gfi.es/tag/graylog/>) UX (<http://blog.gfi.es/tag/ux/>) Docker (<http://blog.gfi.es/tag/docker/>) smart contract (<http://blog.gfi.es/tag/smart-contract/>) machine learning (<http://blog.gfi.es/tag/machine-learning/>) IoT (<http://blog.gfi.es/tag/iot/>) open source (<http://blog.gfi.es/tag/open-source/>) salud (<http://blog.gfi.es/tag/salud/>) ITIL (<http://blog.gfi.es/tag/itil/>) Selenium (<http://blog.gfi.es/tag/selenium/>) APIs (<http://blog.gfi.es/tag/apis/>) sanidad (<http://blog.gfi.es/tag/sanidad/>) ITIL-UTIL (<http://blog.gfi.es/tag/itil-util/>) OAuth (<http://blog.gfi.es/tag/oauth/>) movilidad (<http://blog.gfi.es/tag/movilidad-2/>) AAPP (<http://blog.gfi.es/tag/aapp/>) BBDD (<http://blog.gfi.es/tag/bbdd/>) UX Design (<http://blog.gfi.es/tag/ux-design/>) big data (<http://blog.gfi.es/tag/big-data-2/>) svn (<http://blog.gfi.es/tag/svn/>) claves (<http://blog.gfi.es/tag/claves/>) **Oracle** (<http://blog.gfi.es/tag/oracle/>) BPM (<http://blog.gfi.es/tag/bpm/>) UX/UI (<http://blog.gfi.es/tag/uxui/>) itSMF (<http://blog.gfi.es/tag/itsmf/>) DevOps (<http://blog.gfi.es/tag/devops/>) scrum (<http://blog.gfi.es/tag/scrum/>) UXDesign (<http://blog.gfi.es/tag/uxdesign/>) magento (<http://blog.gfi.es/tag/magento/>) utilities (<http://blog.gfi.es/tag/utilities/>) concurso (<http://blog.gfi.es/tag/concurso/>) PSD2 (<http://blog.gfi.es/tag/psd2/>) calidad (<http://blog.gfi.es/tag/calidad/>) lean (<http://blog.gfi.es/tag/lean/>) energia (<http://blog.gfi.es/tag/energia/>) paciente (<http://blog.gfi.es/tag/paciente/>) WSO2 (<http://blog.gfi.es/tag/wso2/>) Blockchain (<http://blog.gfi.es/tag/blockchain/>) ehealth (<http://blog.gfi.es/tag/ehealth/>)

POPULARES



WSO2 como plataforma SOA open source (<http://blog.gfi.es/wso2-como-plataforma-soa-open-source/>)
29 Ene , 2013



Crea tus servicios de datos sin una línea de código con WSO2 Data Services Server (<http://blog.gfi.es/crea-tus-servicios-de-datos-sin-una-linea-de-codigo-con-wso2-data-services-server/>)
16 Sep , 2014



Flume, Kafka, Spark y Storm, ¿un nuevo ejército Apache? (<http://blog.gfi.es/flume-kafka-spark-y-storm-un-nuevo-ejercito-apache/>)
27 Jun , 2014



DevOps 2.0 o BizDevOps para pensar en transformación digital (<http://blog.gfi.es/devops-bizdevops-transformacion-digital/>)
03 May , 2017



La gestión del tiempo de trabajo, ¿una necesidad para las empresas? (<http://blog.gfi.es/la-gestion-del-tiempo-de-trabajo-una-necesidad-para-las-empresas/>)
09 May , 2017

ENTRADAS RECIENTES

¿Recursos?... o Personas (<http://blog.gfi.es/recursos-o-personas/>)

Feliz Navidad y próspero año 2018 (<http://blog.gfi.es/feliz-navidad-prospero-ano-2018/>)