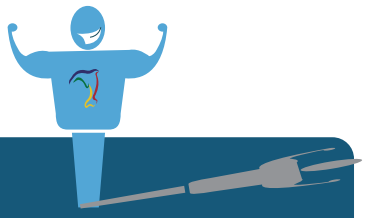
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

# ¿**Qué ofrece** Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

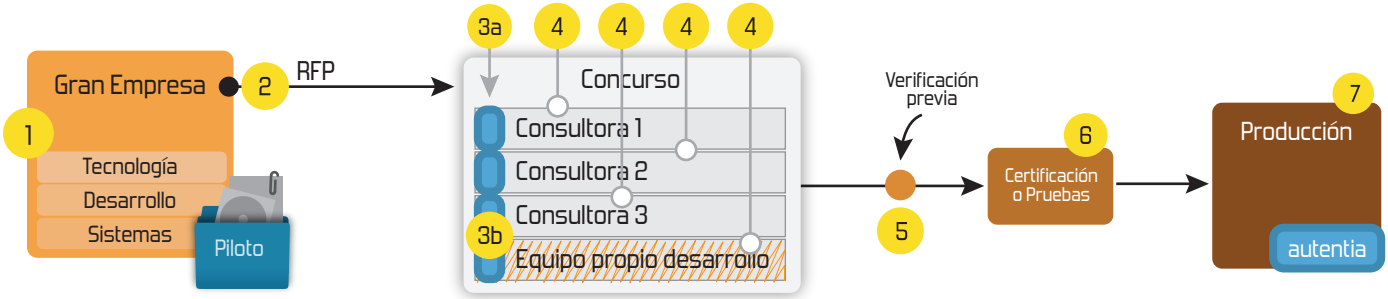
## 1. Desarrollo de componentes y proyectos a medida



## 2. Auditoría de código y recomendaciones de mejora

## 3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.




## 4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces, HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
Gestor de contenidos (Alfresco)
Aplicaciones híbridas

Control de autenticación y acceso (Spring Security)
UDDI
Web Services
Rest Services
Social SSO
SSO (Cas)

JPA-Hibernate, MyBatis
Motor de búsqueda empresarial (Solr)
ETL (Talend)

Dirección de Proyectos Informáticos.
Metodologías ágiles
Patrones de diseño
TDD

Tareas programadas (Quartz)
Gestor documental (Alfresco)
Inversión de control (Spring)

BPM (jBPM o Bonita)
Generación de informes (JasperReport)
ESB (Open ESB)

Compartimos nuestro conocimiento en:
**www.adictosaltrabajo.com**

Para más información visítenos en:
**www.autentia.com**


E-mail:
Contraseña:
Entrar
Deseo regist
He olvidado mis datos de acceso



DESARROLLADO POR:
Jose Manuel Sánchez Suárez

Consultor tecnológico de desarrollo de proyectos informáticos.

Puedes encontrarme en Autentia: Ofrecemos servicios de soporte a desarrollo, factoría y formación

Somos expertos en Java/J2EE



**Fecha de publicación del tutorial: 2010-09-10**

65



# AppDynamics Lite, encontrar problemas de rendimiento en aplicaciones Java en un entorno de producción.

## 0. Índice de contenidos.

## 1. Introducción

En este tutorial vamos a ver cómo llevar a cabo la instalación y unos primeros pasos con una herramienta gratuita para monitorizar y detectar problemas de rendimiento en aplicaciones Java en producción AppDynamics Lite. No se trata de una herramienta de profiling, está pensada para detectar respuestas lentas a peticiones web, SQLs con bajo rendimiento y errores en las peticiones; y está orientada a un entorno de producción, donde la monitorización no produzca un impacto en el rendimiento.

En palabras de los creadores: es una herramienta pensada para usar en la primera línea de fuego, se instala en 2 minutos y puedes llegar a detectar errores en muy poco tiempo, 15 minutos o menos.

Como comentaba, es gratuita, en su versión Lite, y existe una versión de pago. La versión Lite tiene una serie de limitaciones:

- solo monitoriza lo sucedido en las últimas dos horas,
- no monitoriza las llamadas a servicios web,
- no detecta Memory Leaks, Deadlocks y Payload/Input Data related errors,
- no dispone de alertas y notificaciones ni una cola de incidencias.

Con la versión Lite, en un principio, nos basta, no somos un departamento de QA.

Solo os digo que si tenéis la ocasión de probarla os va a sorprender su sencillez y el nivel de detalle que es capaz de proporcionar sobre nuestro sistema y un problema de rendimiento concreto.

## 2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 17' (2.93 GHz Intel Core 2 Duo, 4GB DDR3 SDRAM).
- Sistema Operativo: Mac OS X Snow Leopard 10.6.1
- AppAgentLiteBeta 0.91
- Una aplicación web empresarial con la siguiente arquitectura que corre bajo un Apache Tomcat 6.0.20 y tiene como base de datos un MySQL 5.0.88:
  - Spring 2.5.6
  - Hibernate 3
  - JSF 1.2
  - Apache Tomahawk 1.9

## 3. Instalación.

La instalación es realmente sencilla, es como hudson o sonar, que vienen listos para usar.

Lo primero es acceder a la url de descarga: http://www.appdynamics.com/lite.php

XI Charla Autentia - Mule

Histórico de NOTICIAS

Últimos Tutoriales

VMWare Workstation, instalación en un host Microsoft Windows

VMWare Workstation, ideal para SOHO

Cacoo, herramienta colaborativa para hacer diagramas, incluso prototipado de pantallas o UML

Introducción a CAS

Gestión de eventos en el cliente con el soporte de Ajax de RichFaces

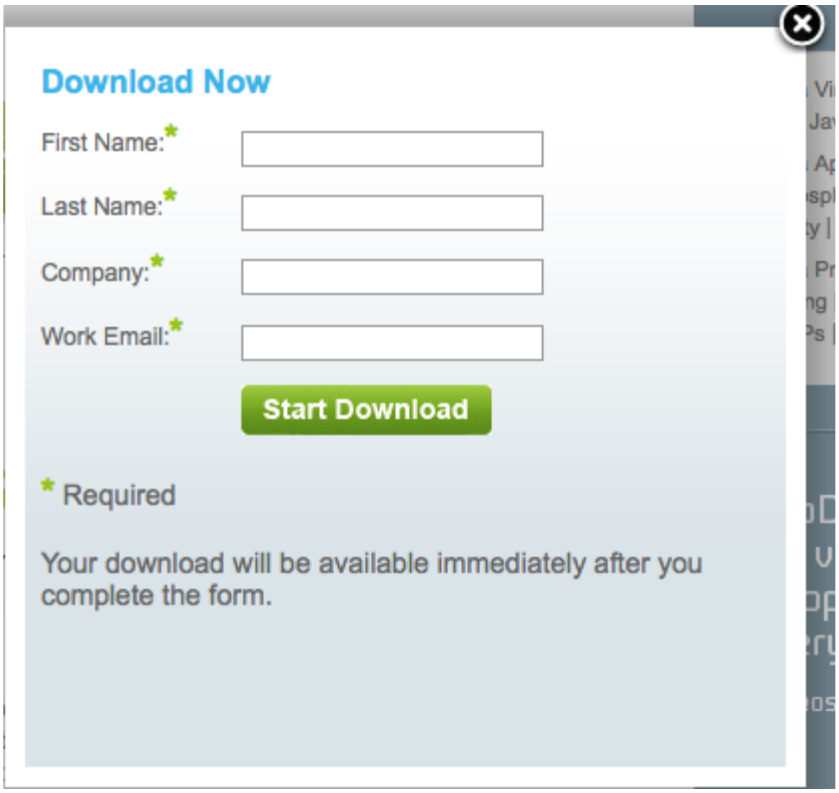
Últimos Tutoriales del Autor

Gestión de eventos en el cliente con el soporte de Ajax de RichFaces

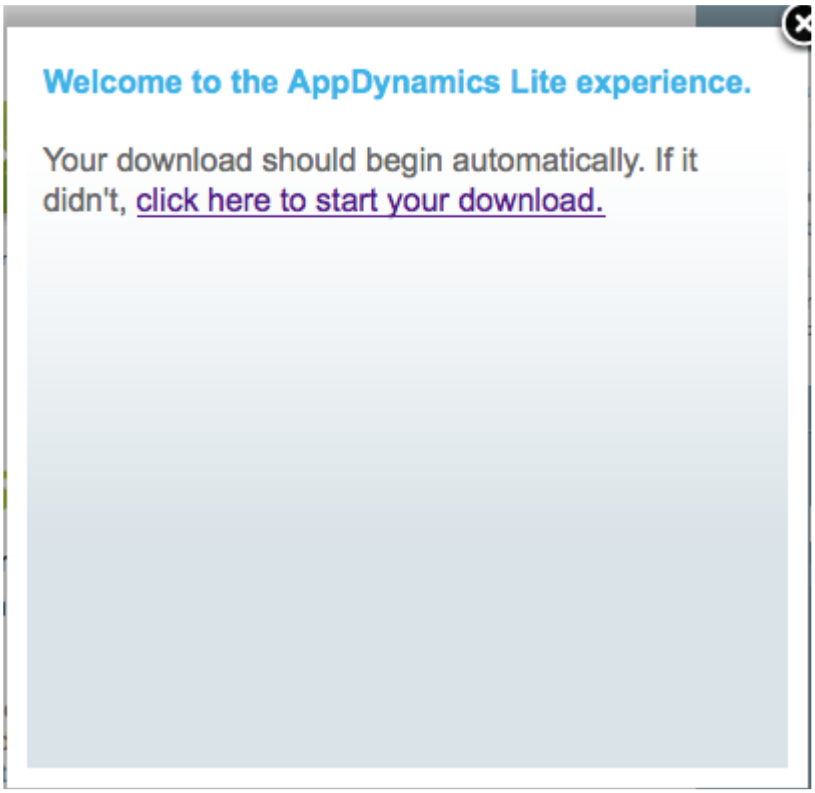
Envío de correo electrónico con el soporte de Jboss Seam.

Creación de servicios web RESTful con el soporte de RESTeasy de Jboss Seam.

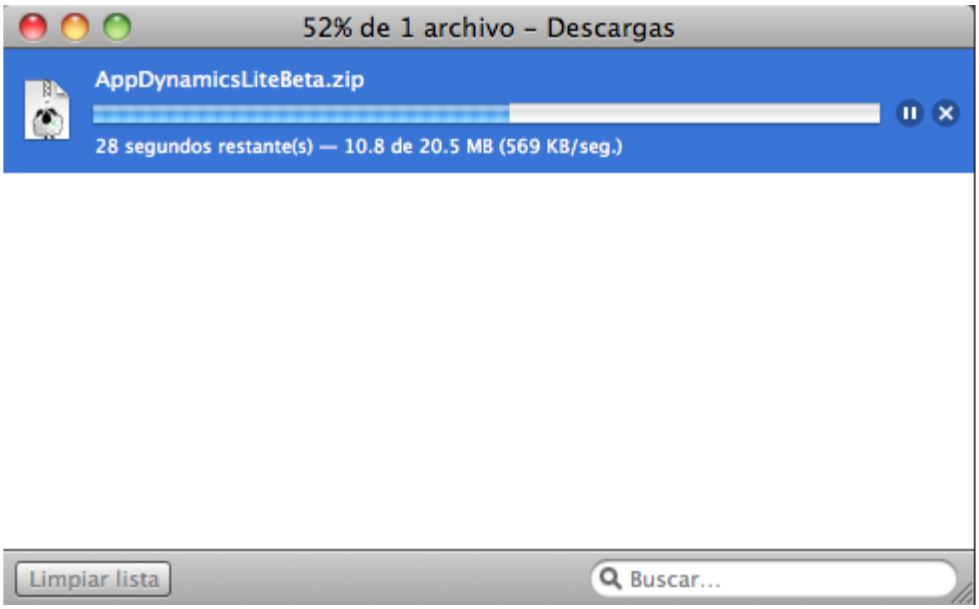

Pulsamos sobre el botón "Download" y se mostrará la siguiente ventana modal:



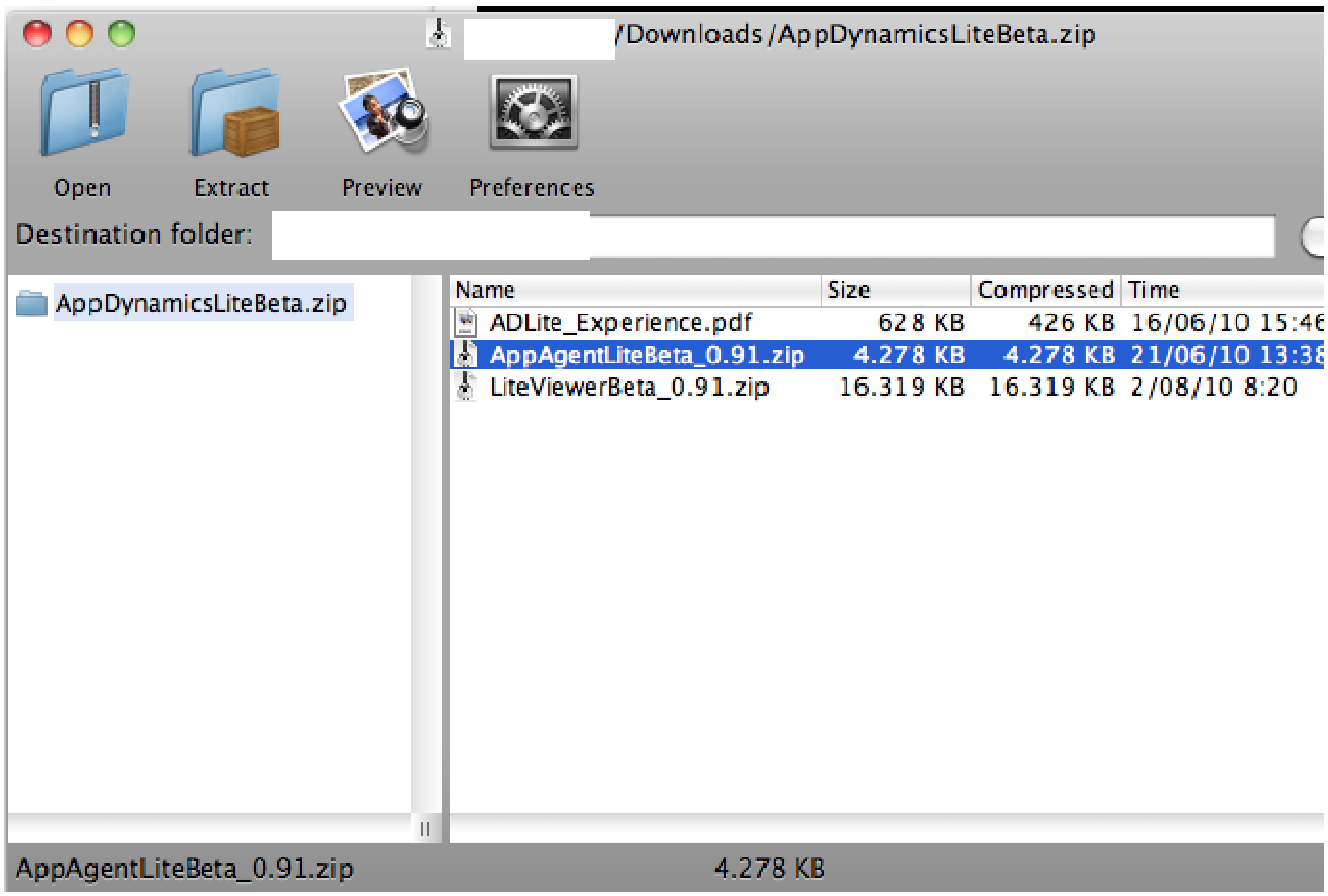

Tras instroducir los datos (el email debe ser de empresa), se mostrará una ventana como la que sigue:



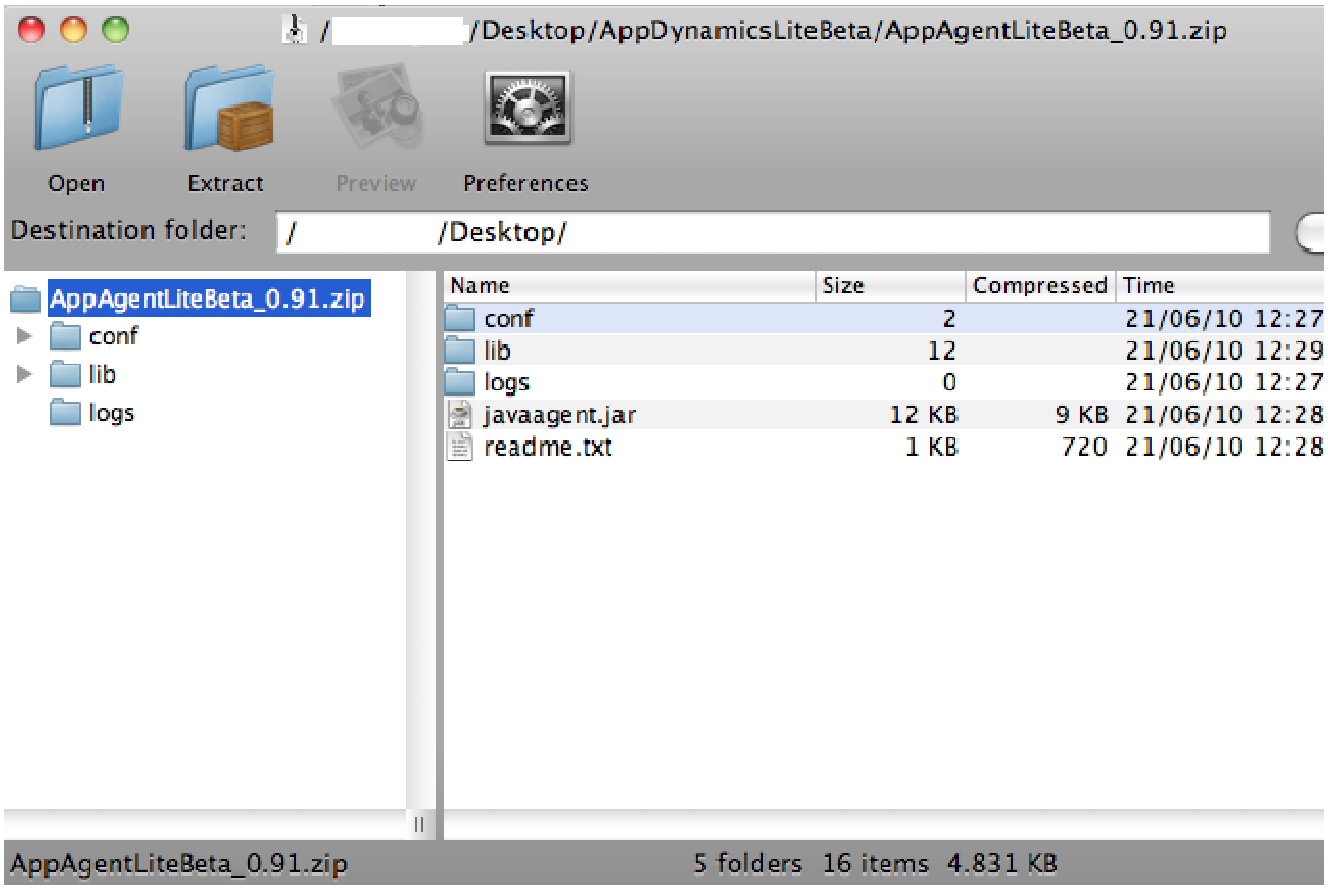

La descarga viene dada o la podemos forzar pulsando en el link, ahora solo depende del tipo de conexión del que dispongamos:



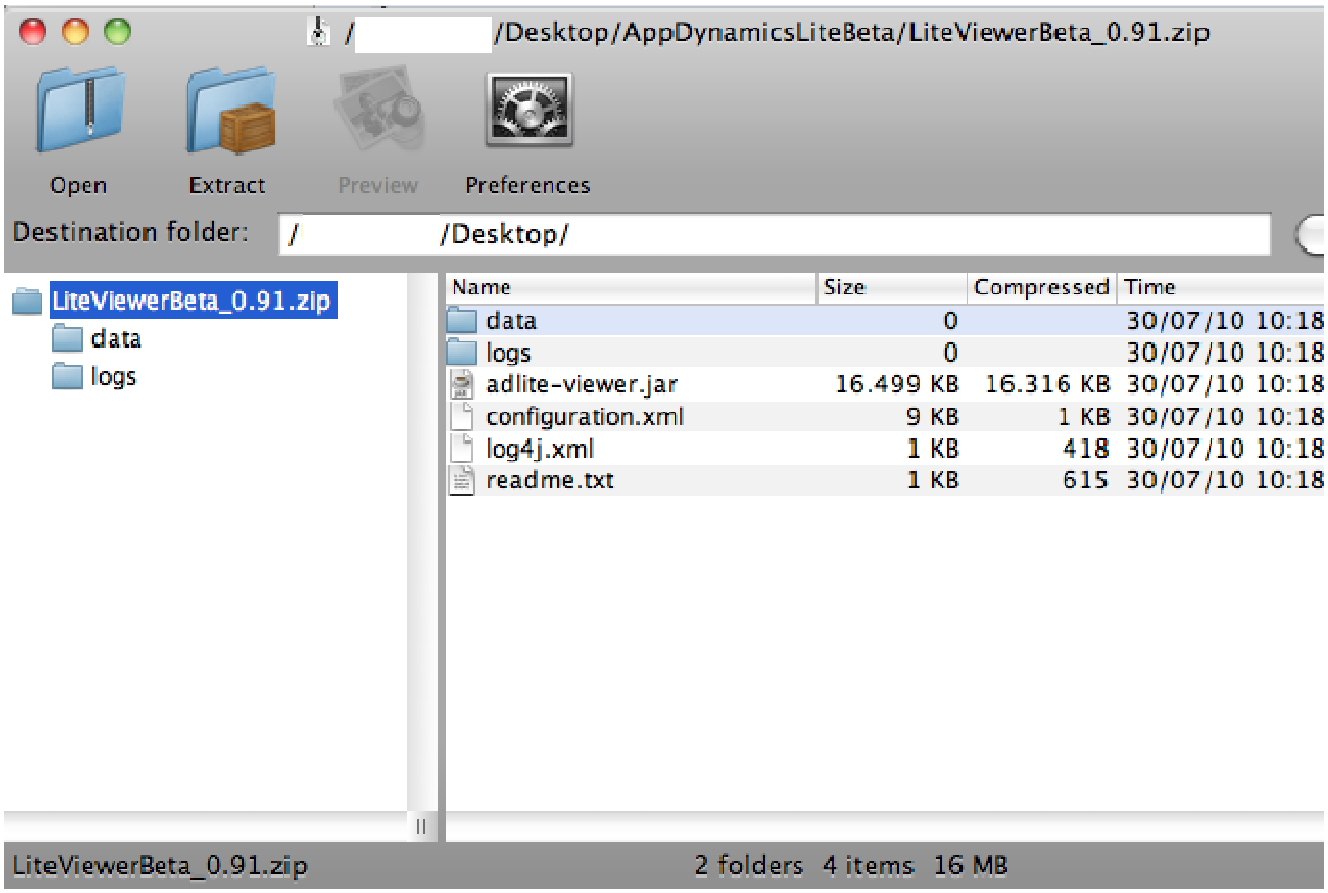

Una vez descargado solo hay que descomprimirlo con tu programa favorito: contiene un pdf con una fact sheet sobre el producto y dos ficheros comprimidos:




El fichero AppAgentLiteBeta_xx.x.zip tiene el siguiente contenido, el objetivo es descomprimirlo en una ruta local.



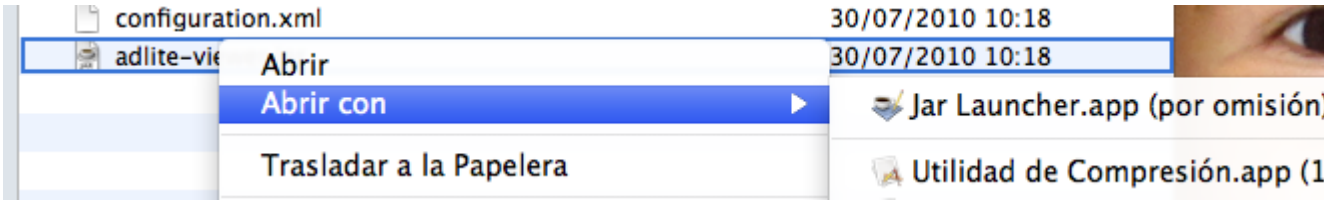

El fichero LiteViewerBeta_xx.x.zip tiene el siguiente contenido, el objetivo es descomprimirlo en una ruta local.



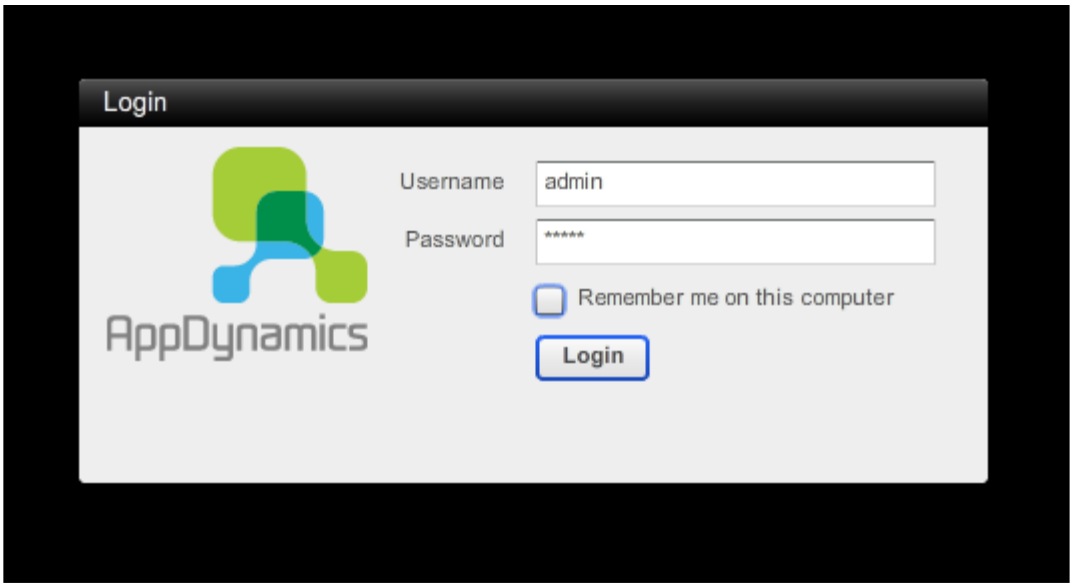

Una vez descomprimido solo falta arrancar ambas piezas:

el viewer es un visor (una aplicación web con un micronavegador embebido) que recibe y almacena los eventos de monitorización del agente; y
- el javaagent es la pieza que se engancha con la JVM del servidor de aplicaciones e instrumentaliza el código para lanzar los eventos correspondientes: de invocación, tiempo consumido, errores,... Un agente es un interceptor que se pone delante de la invocación a los métodos de las clases que monitoriza y es un estandar a partir de la JDK 5.

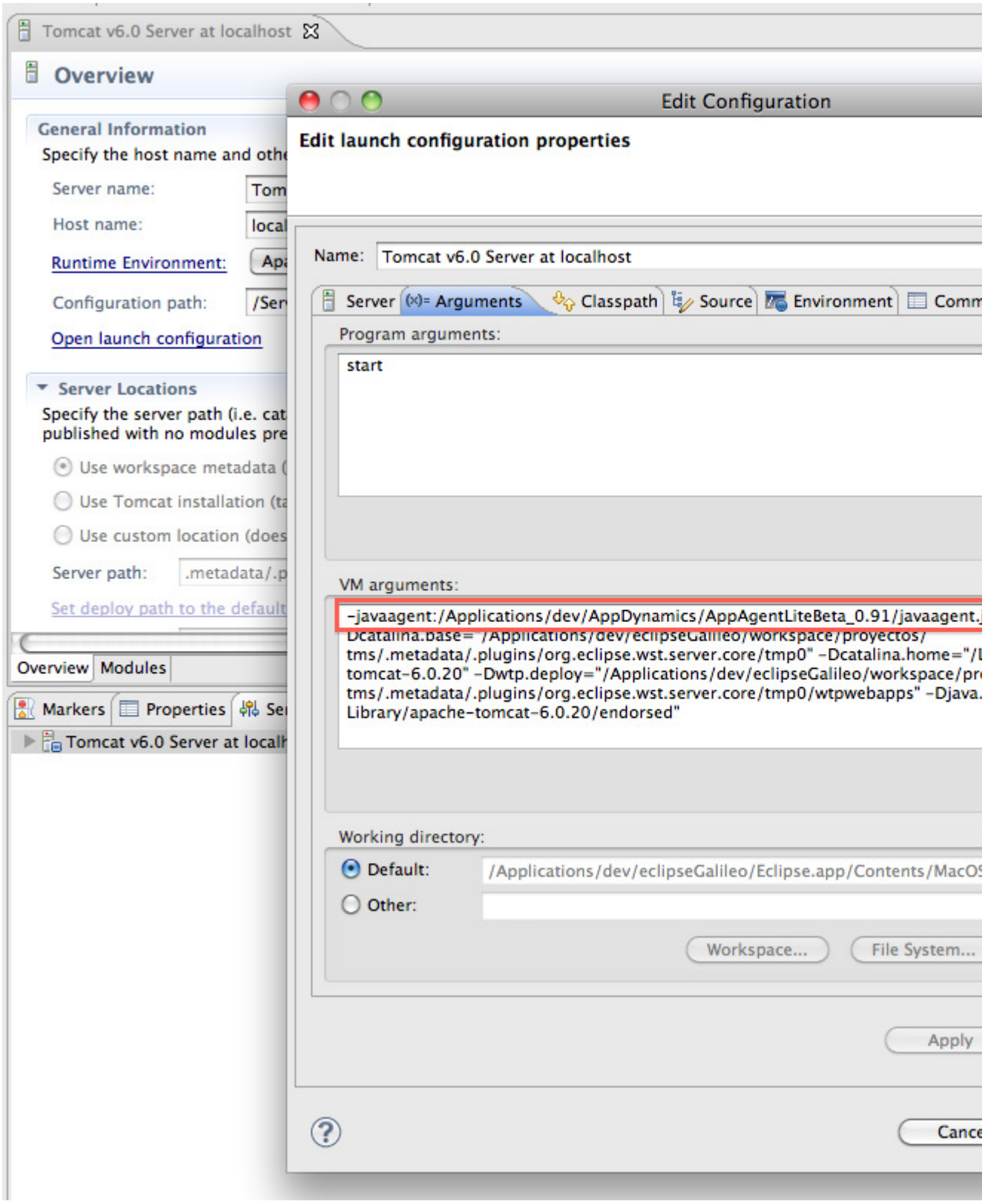
Para arrancar el visor basta con ejecutar el jar adlite-viewer.jar, como se muestra en la siguiente imagen, marcándolo como ejecutable en otros SO o por líena de comandos: `java -jar adlite-viewer.jar`.




Por defecto se accede a través del puerto 8990, pero se puede parametrizar y para acceder basta con un navegador y la siguiente url: `http://localhost:8990/`.




El usuario y contraseña es admin:admin y si accedemos sin tener arrancado el agente te informa sobre ello. De momento lo dejamos en segundo plano hasta que hagamos lo siguiente con el agente.




Para registrar el agente basta con introducir el parámetro -javaagent con la ruta al fichero javaagent.jar en los parámetros de arranque de la máquina virtual. En la imagen anterior se muestra como añadirlo a los parámetros de la VM en la configuración del servidor desde Eclipse; pero se puede añadir al script de arranque del servidor sin problema alguno.

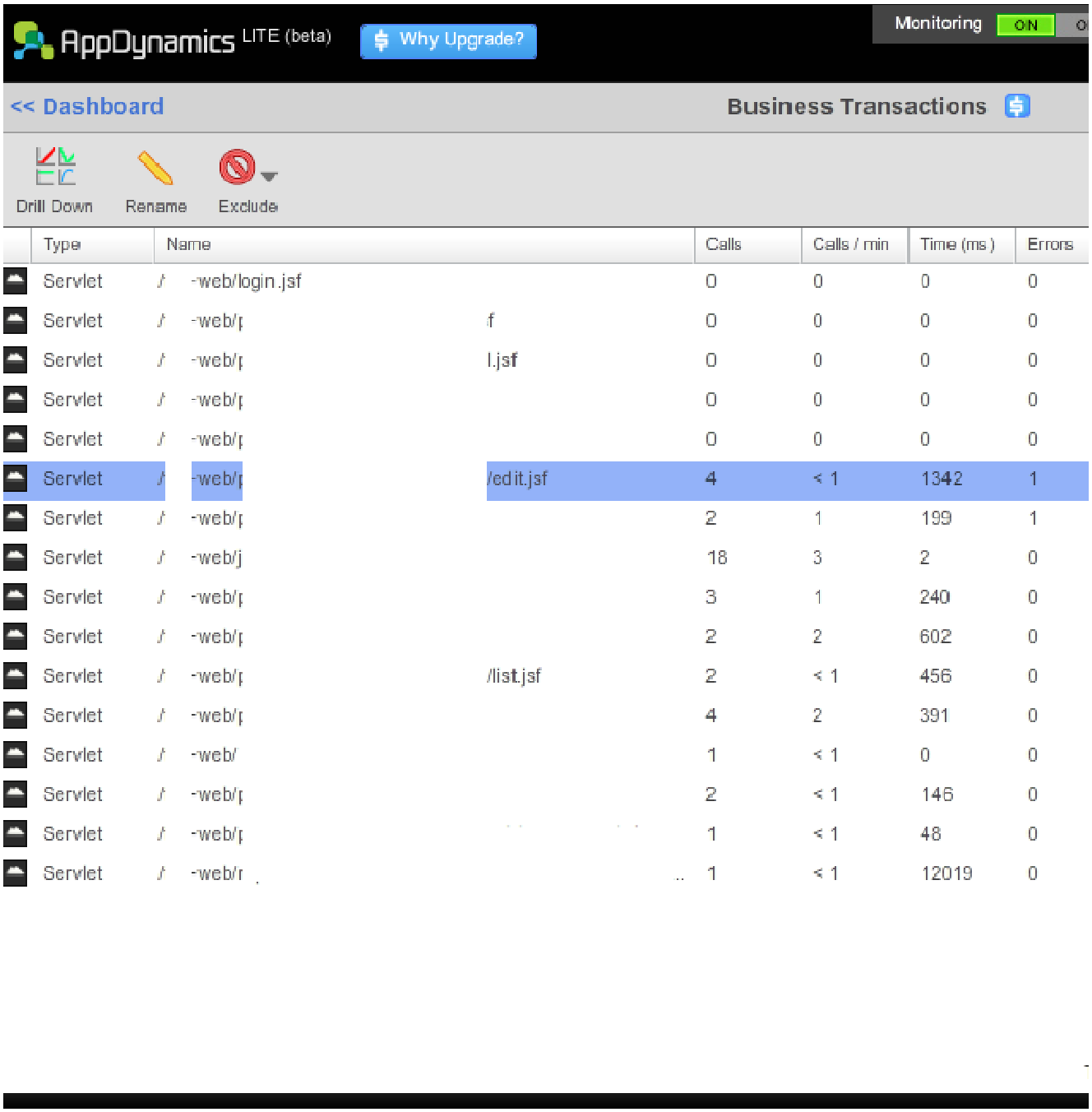
A continuación se muestra la salida por consola del agente nada más arrancar el servidor (como veis en la captura también lo he probado con una aplicación que corre bajo un Jboss Server 4.2.3).


## 4. Primeros pasos.

Una vez instalado solo queda acceder a la aplicación a monitorizar y navegar un poco por ella. Si sospechamos que alguna parte de la misma adolece de problemas de rendimiento esta es la oportunidad para que salgan a la luz.

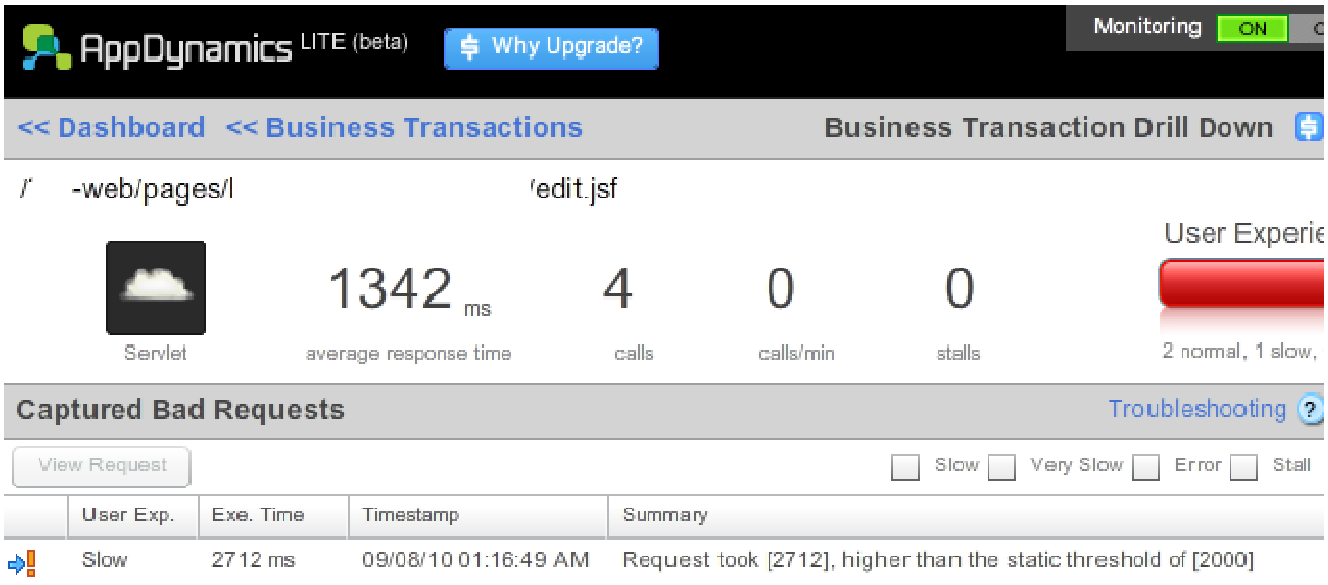
Después de generar algo de tráfico, accedemos a la url antes comentada con el usuario y contraseña ya dichos y se mostrará una pantalla similar a la siguiente, ¿no es genial? (en las capturas siguientes he borrado URLs e información sensible de la aplicación sobre la que he realizado las pruebas).


Tenemos un dashboard o cuadro de mando con un esquema de las piezas implicadas en la parte central (servidor, base de datos, invocaciones a servicios externos o internos, a colas de mensajes JMS a Web Services), en la parte derecha, una pila de las llamadas agrupadas por peticiones y ordenadas por rendimiento, y, en la parte inferior, una gráfica de tiempo de carga y respuesta en relación a un periodo de tiempo determinado. Pulsando sobre (maximize) en la parte derecha accedemos a más información sobre la pila de llamadas, lo que denominan transacciones de negocio. Se mostrará una pantalla como la que sigue:



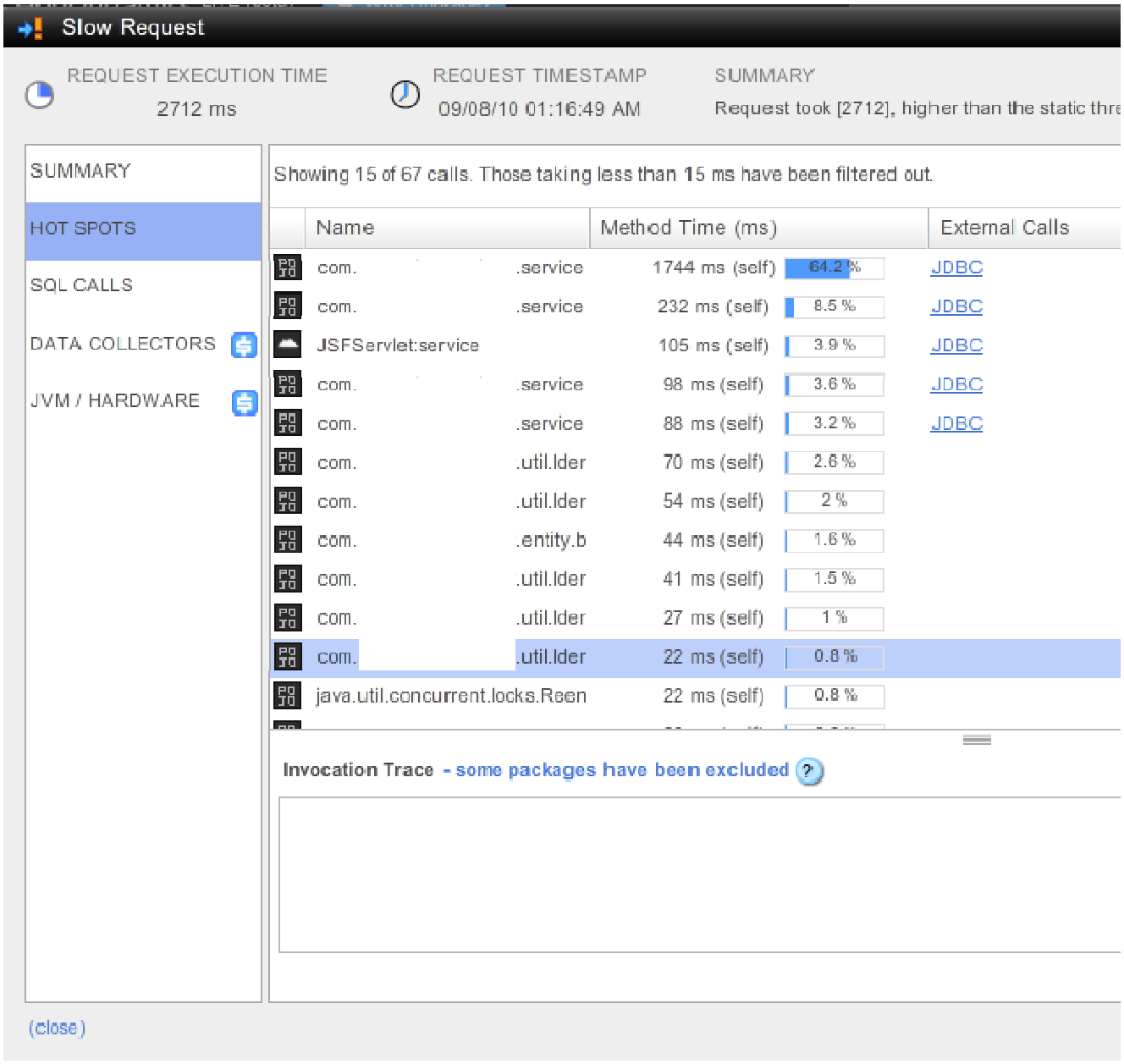

| Type | Name | Calls | Calls / min | Time (ms) | Errors |
|---|---|---|---|---|---|
| Servlet | /*-web/login.jsf | 0 | 0 | 0 | 0 |
| Servlet | /*-web/p f | 0 | 0 | 0 | 0 |
| Servlet | /*-web/p l.jsf | 0 | 0 | 0 | 0 |
| Servlet | /*-web/p | 0 | 0 | 0 | 0 |
| Servlet | /*-web/p | 0 | 0 | 0 | 0 |
| Servlet | /*-web/p /edit.jsf | 4 | < 1 | 1342 | 1 |
| Servlet | /*-web/p | 2 | 1 | 199 | 1 |
| Servlet | /*-web/j | 18 | 3 | 2 | 0 |
| Servlet | /*-web/p | 3 | 1 | 240 | 0 |
| Servlet | /*-web/p | 2 | 2 | 602 | 0 |
| Servlet | /*-web/p /list.jsf | 2 | < 1 | 456 | 0 |
| Servlet | /*-web/p | 4 | 2 | 391 | 0 |
| Servlet | /*-web/ | 1 | < 1 | 0 | 0 |
| Servlet | /*-web/p | 2 | < 1 | 146 | 0 |
| Servlet | /*-web/p | 1 | < 1 | 48 | 0 |
| Servlet | /*-web/r | 1 | < 1 | 12019 | 0 |

Haciendo un doble click sobre una fila o pulsando sobre Drill Down, se accede al detalle de la petición, mostrándose una pantalla como la que sigue:



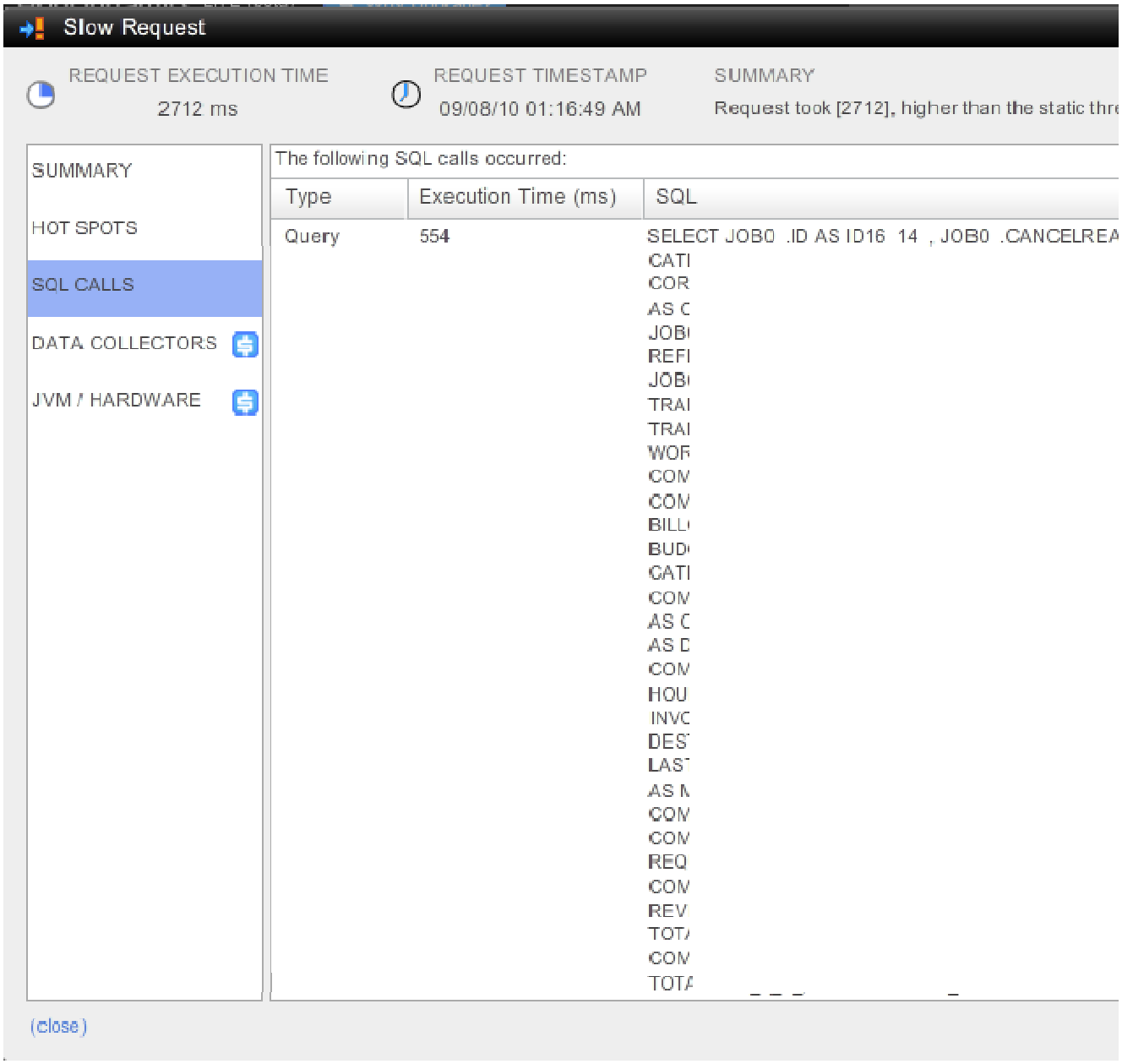

En el ejemplo anterior tenemos una petición que ha tardado más de 2 minutos en responder, que la marca con naranja; las peticiones que tardan más de 5 minutos las marca en rojo, así como los errores. Los tiempos son configurables.

Haciendo doble click sobre la fila se accede a una ventana modal con más información sobre la petición, en la que podemos ver los puntos calientes (realizando un filtrado por tiempo consumido).

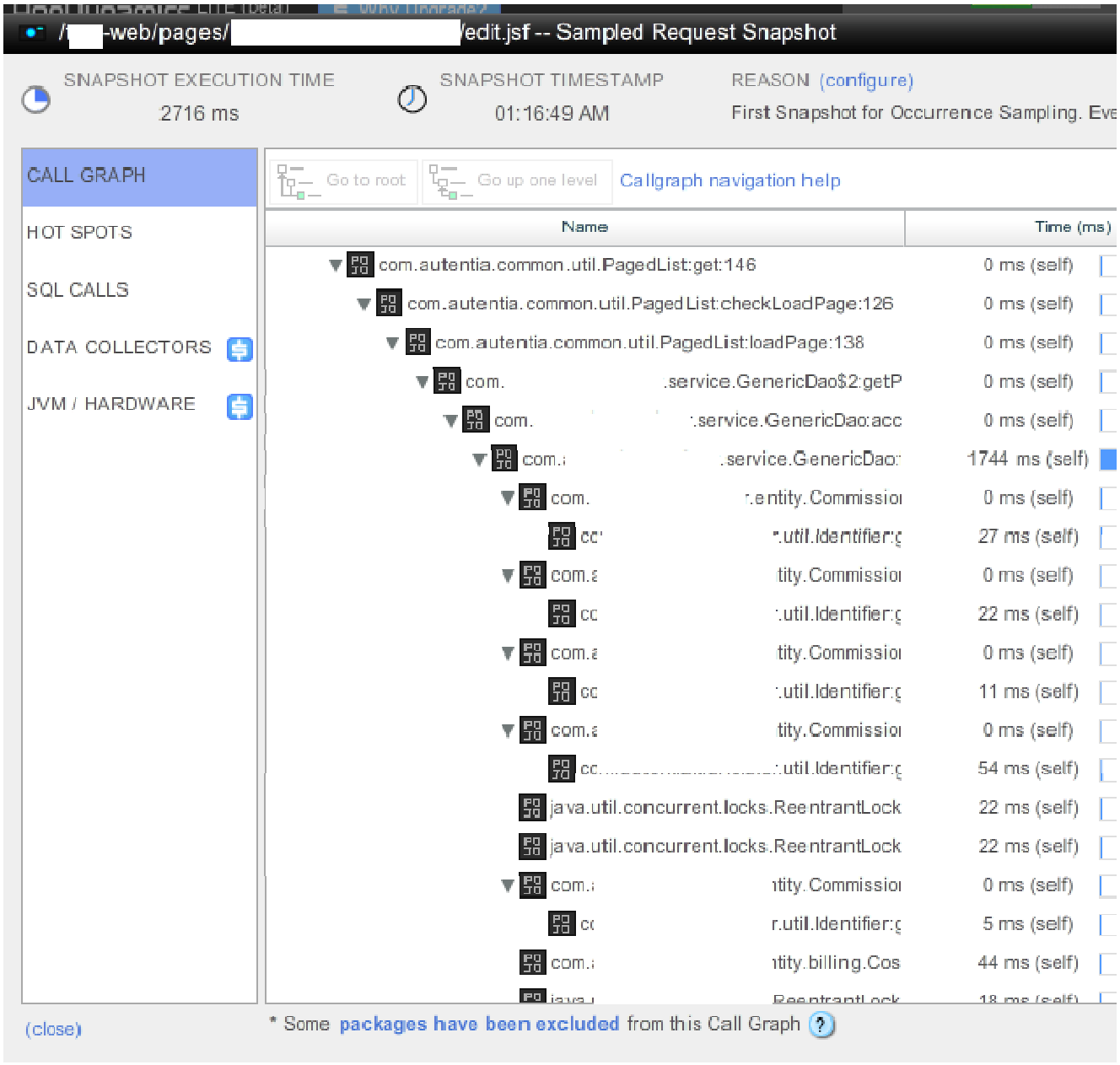



El nivel de detalle es el de la invocación a un método de una clase, y si el mismo ha implicado la ejecución de una sentencia SQL se puede acceder al contenido de la misma desde un enlace que muestra una ventana emergente.

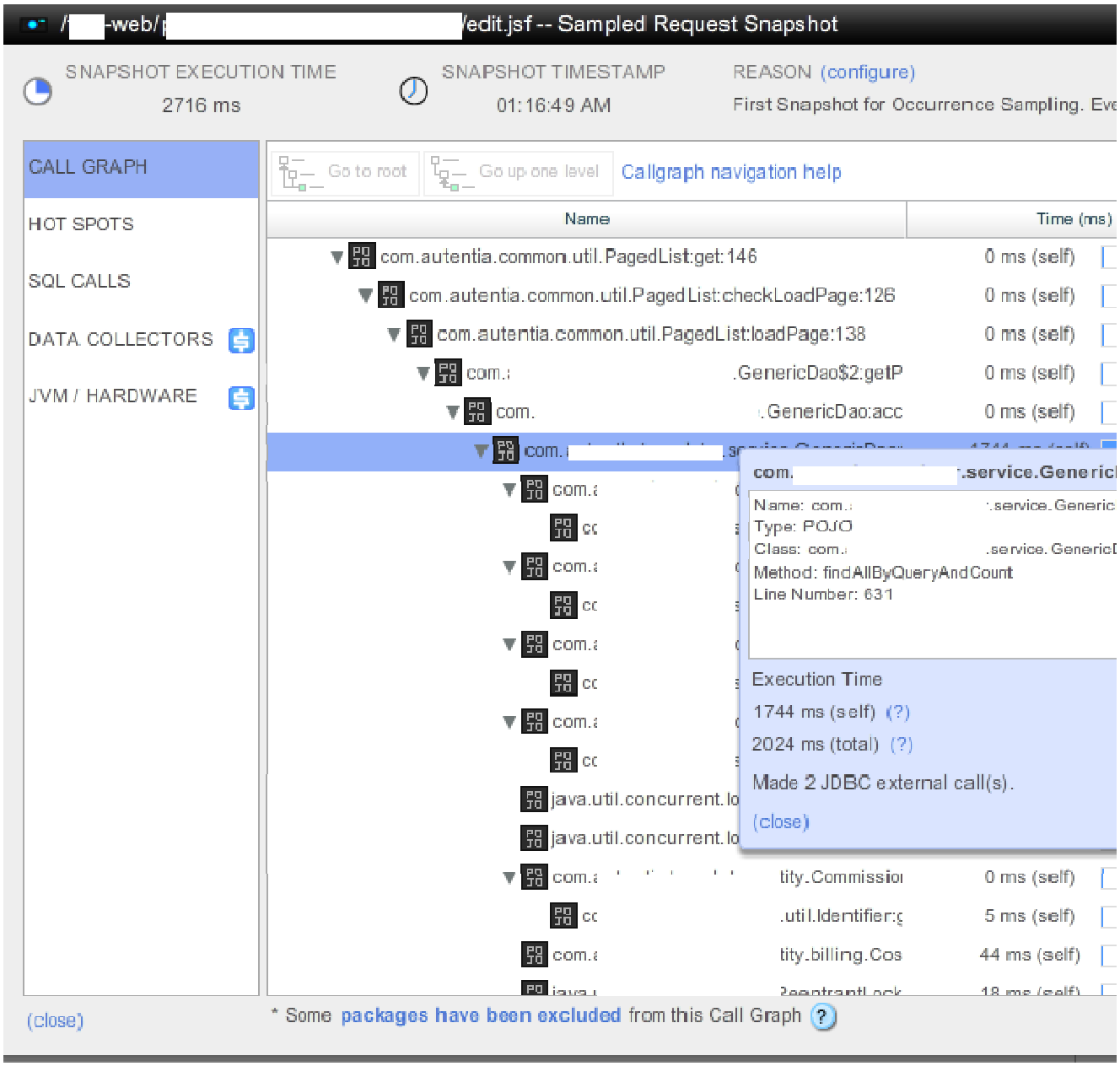
Como se muestra en la siguiente captura también se puede acceder a la pila de sentencias SQL ejecutadas durante la petición, pudiendo visualizar el contenido de las mismas.




Por último, desde la parte inferior derecha de la ventana de detalle de la petición (sampled snapshot) se puede acceder al detalle de una serie de peticiones de muestreo, pudiendo visualizar, como se muestra a continuación, la pila de ejecución de la invocación a un servicio, con un ratio del tiempo consumido en cada uno de los métodos.

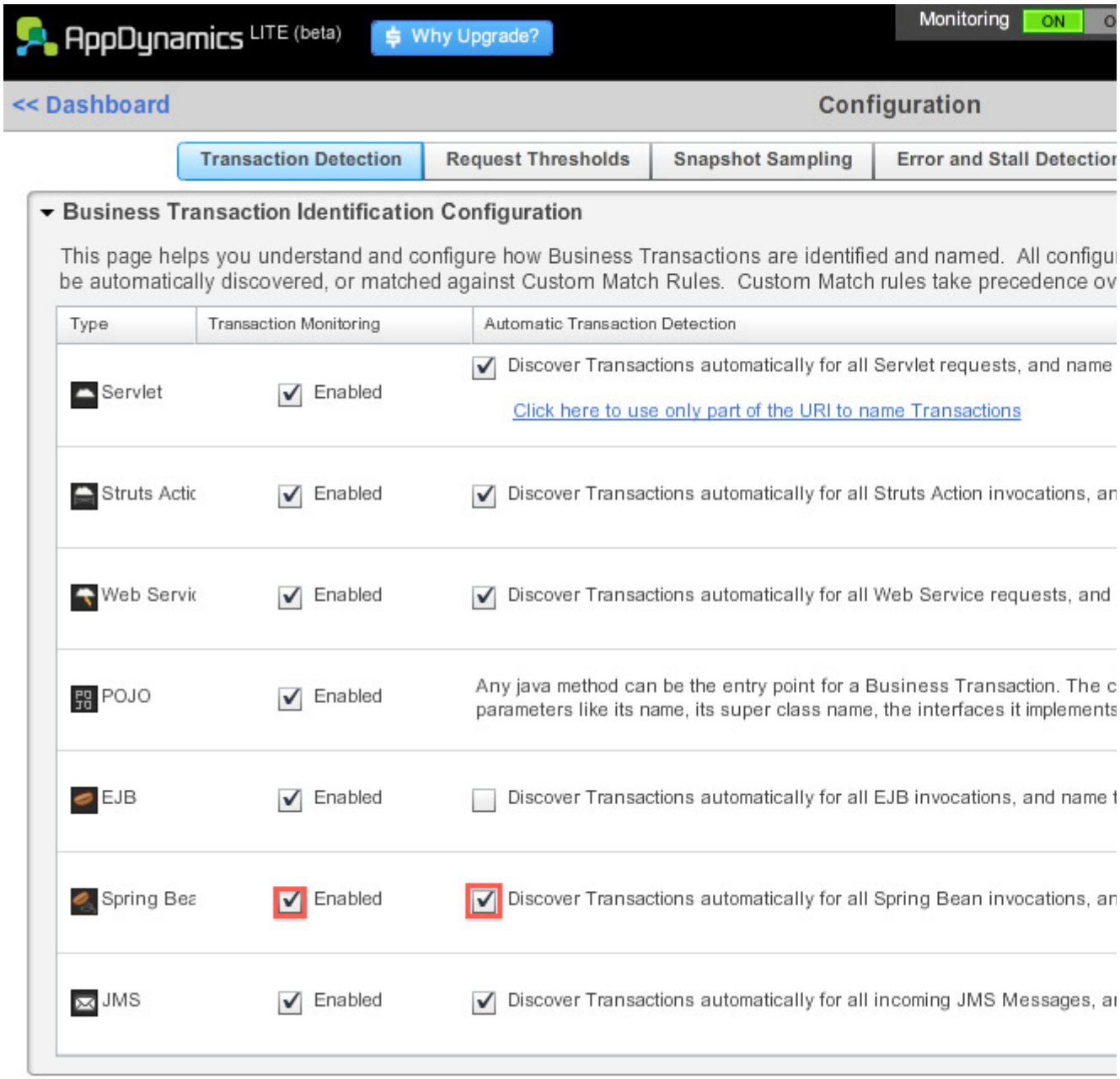



Pulsando sobre una fila se puede acceder a más detalle sobre la invocación a un método.




Sorprende que con tan poco configurado se tenga accesible tanta información y funcionando a la primera.

Se puede pinchar la invocación a los siguientes componentes, los beans de Spring no vienen por defecto seleccionados.




Y se puede configurar el nivel de detalle de las SQL, si necesitamos conocer el valor de los parámetros de las sentencias o queremos que lo filtre con ?.


## 5. Referencias.

- http://blog.codecentric.de/en/2010/08/troubleshoot-java-in-production-introducing-appdynamics-lite/
- http://blog.codecentric.de/en/2010/08/easy-performance-analysis-with-appdynamics-lite/
- http://www.appdynamics.com/lite.php

## 6. Conclusiones.

Para mi, una herramienta a tener en cuenta en caso de emergencia. Sabemos que en el entorno de desarrollo no se suele disponer del mismo juego de datos que en el

entorno de producción y nos enfrentamos con probemas de rendimiento más veces de las que desearíamos. Poder disponer de una herramienta como esta nos puede sacar de un apuro y nos ayudará a la realización de una primera auditoría de rendimiento.

Como decía, existen otras herramientas para realizar pruebas de carga, no es el objetivo de AppDynamics Lite; pero ante un error puntual de la aplicación o una deficiencia en el rendimiento, podemos pinchar directamente en producción y llevarnos una información muy valiosa sobre el comportamiento real del sistema, con una configuración mínima.

Espero que lo disfrutéis.

Un saludo.

Jose

jmsanchez@autentia.com

Anímate y coméntanos lo que pienses sobre este **TUTORIAL:**

Puedes opinar o comentar cualquier sugerencia que quieras comunicarnos sobre este tutorial; con tu ayuda, podemos ofrecerte un mejor servicio.

Enviar comentario

(Sólo para usuarios registrados)

» **Registrate** y accede a esta y otras ventajas «

# COMENTARIOS

***franferri*** *2010-09-10 - 09:36:01*

*Se parece mucho al wily introscope para java (t.b. .Net)*
*Es como una modernización de la parte de medición de rendimiento.*
*Muy útil.*


XHTML 1.0
CSS
RSS
ATOM