# [OJO VERSIÓN ENTREGADA INCLUYE CABECERA / VER OTRA CARPETA]

# Aplicaciones Ruby para generación de informes de rendimiento

El rendimiento de los sistemas informáticos de la UNED es crítico para su imagen, la utilización adecuada de los recursos, y la satisfacción de los usuarios. Para controlar y mejorar ese rendimiento es imprescindible disponer de estadísticas fiables, generadas con suficiente frecuencia para detectar problemas y realizar un seguimiento adecuado.

Actualmente se ha desarrollado tres aplicaciones para procesar tres fuentes de información:

* Logs de las aplicaciones Java / GAIA
* Tabla de información de tiempos de respuesta de procedimientos PL/SQL
* Estadísticas de Oracle

Se ha desarrollado tres aplicaciones para procesar estas fuentes de información:

* Logs - log\_utils\proc\_parser.rb
* Tabla de tiempos - log\_procesos\db\_parser.rb
* Estadísticas - oracle\_stat\_parser\parse.rb

Las tres aplicaciones generan ficheros Excel de salida, para su posterior análisis.

## Instalación

Las aplicaciones están desarrolladas en el lenguaje Ruby, elegido por su facilidad de desarrollo y sus características que permiten un tratamiento sencillo y rápido de ficheros de texto.

### Entorno

La ejecución de la aplicación requiere instalar Ruby, y una librería adicional para la generación de ficheros Excel. La aplicación se ha probado con las versiones 1.9.2 y 1.9.3 de Ruby.

**Instalación Ruby**

1. Descarga Ruby del URL: <http://rubyinstaller.org/downloads> e instalarlo:
   1. Elige un directorio de instalación. Se aconseja el uso de un directorio sencillo, sin espacios en el nombre, por ejemplo c:\ruby
   2. Asocia las extensiones .rb y .rbw con Ruby.
   3. No es necesario incluir Ruby en el PATH de la máquina, aunque facilitará la instalación de la librería de Excel.
2. Descarga el DevKit del mismo URL de Ruby e extraelo en el mismo directorio que la instalación:
   1. C:\ruby\DevKit (hay que añadir el directorio “DevKit”, no se genera solo)

**Instalación librería generación Excel**

1. Abre una ventana DOS (cmd.exe)
2. Si Ruby no está en el PATH de la máquina será necesario incluirlo de forma temporal:

set path=%path%;<directorio instalación>\bin

1. Ejecutar (desde DOS) c:\ruby\DevKit\devkitvars.bat
2. Instalar la librería:

gem install simple\_xlsx\_writer

**Instalación librería acceso Oracle**

Para la conectividad con Oracle son necesarios los siguientes pasos:

1. Instalar el cliente Oracle, incluyendo SQLPlus y los componentes SDK
2. Instalar el devkit de Ruby
3. Configurar el variable de entorno ORACLE\_HOME
4. Configurar el path para que incluya la ruta a SQLPlus
5. Instalar la librería de Ruby:

gem install -r ruby-oci8

### Aplicaciones

Las aplicaciones Ruby no requieren instalación, es suficiente copiar los ficheros a una carpeta.

# Aplicación de procesamiento de ficheros log

## Introducción

Los ficheros logs de los servidores de aplicaciones Java utilizado en la UNED se componen de entradas ordenadas en el tiempo, y que siguen un patrón definidos por la configuración del motor de log (Log4J[[1]](#footnote-1)) y los estándares de desarrollo seguidos en la UNED.

Se ha desarrollado una aplicación que permite extraer información de los logs, utilizando los patrones establecidos para sacar información acerca de:

1. Tiempos de ejecución de procedimientos
2. Números de errores reportados
3. Número de entradas

La información obtenida está procesada, y a continuación escrita en un fichero Excel para su análisis posterior. Este análisis sale de ámbito del presente documento.

### Definición de datos

La utilización de los patrones implica una dependencia fuerte sobre la configuración del sistema y el desarrollo realizado: cualquier modificación implicará una modificación en el programa.

El formato esperado es el siguiente, cada elemento está separado del siguiente con espacios:

[Fecha] [Hora] [Nivel] [ID Hilo] [Clase Java]===> [Detalles]

|  |  |  |
| --- | --- | --- |
| Concepto | Descripción | Ejemplo |
| Fecha | Fecha de la llamada (yyyy-mm-dd) | 2013-02-15 |
| Hora | Hora de la llamada (hh:mm:ss.sss) | 10:30:24.520 |
| Nivel | Nivel del mensaje | INFO |
| ID Hilo | Identificador del hilo. Los hilos están asociados a una petición de un cliente. Peticiones y los hilos asociados pueden estar entrelazados en el log.  Debe empezar con “AJPRequestHandler-ApplicationServerThread-“ | AJPRequestHandler-ApplicationServerThread-7 |
| Clase Java | Clase Java responsable del mensaje en el log. Siempre terminado con “===>” | es.uned. portal.gaia.datospersonalespas. portlet.GestionarDatosAlumnoPortlet. renderBody===> |
| Detalles | Mensaje.  Se ha identificado varios patrones de los mensajes que se utilizan para sacar información, sobre todo permiten identificar el inicio y el final de una llamada a un procedimiento PL/SQL. | \*\*\* Inicio Procedimiento --> {call FECHASEXA.PK\_ACCESOS. datosper\_usuario(?,?)}(PAS00227) \*\*\* |
| \*\*\* FIN PROCEDIMIENTO \*\*\* |

Se descartan las líneas que no incluyen la fecha y el identificador de hilo. No se está procesando, por lo tanto, las posibles trazas de excepciones. (Generalmente no es problemático, ya que disponemos del log original).

Se utiliza el hilo y la clase Java para identificar una transacción (en el sentido más amplio de la palabra), y a continuación el “inicio procedimiento” y “fin procedimiento” para obtener la información del tiempo de ejecución.

Respeto a los tiempos calculados de procedimientos, hay dos posibles problemas con esa forma de procesar los datos:

1. Los nombres de los procedimientos (de formato FECHASEXA.PK\_ACCESOS. datosper\_usuario) deben ser únicos y corresponder al mismo procedimiento. Sería posible configurar varias aplicaciones Java en el mismo servidor, compartiendo el mismo log, pero cada una con su fuente de datos, y estas fuentes de datos acceden a distintas bases de datos con solapamiento de nombres.
2. La publicación del mismo procedimiento con dos nombres distintos (synonyms) dará lugar a dos líneas en el fichero de salida, sin asociación entre sí.

De estos dos problemas el segundo parece más real, pero actualmente no disponemos de la información necesaria para identificar y cuantificar el impacto real.

## Obtención datos

Para obtener los datos es necesario entrar con un cliente SFTP (por ejemplo Filezilla[[2]](#footnote-2)) al servidor de aplicaciones:

1. Usuario: según proporcionado por la UNED
2. URL: sftp://<usuario>@kraken.csi.uned.es

Los logs están en:

/var/log/REMOTE\_LOGS/marte1/OC4J\_Grados/gaia.log

/var/log/REMOTE\_LOGS/marte2/OC4J\_Grados/gaia.log

/var/log/REMOTE\_LOGS/marte3/OC4J\_Grados/gaia.log

**Aunque en teoría los logs están *rotados* diariamente, en la actualidad este proceso no funciona correctamente y los logs de días anteriores están malformados o incompletos.** Para obtener datos fiables de un día concreto es preciso copiar los logs al final de la tarde.

Hay que copiar los logs a un único directorio; evidentemente, será necesario renombrar los ficheros para que no sobrescriban uno al otro.

Para simplificar el proceso de copia de los ficheros se aconseja la creación de un directorio en el servidor, y utilizar vínculos simbólicos para evitar el problema:

|  |
| --- |
| $ **cd ~**  $ **mkdir logs**  $ **cd logs**  $ **ln –s /var/log/REMOTE\_LOGS/marte1/OC4J\_Grados/gaia.log marte1.log**  $ **ln –s /var/log/REMOTE\_LOGS/marte2/OC4J\_Grados/gaia.log marte2.log**  $ **ln –s /var/log/REMOTE\_LOGS/marte3/OC4J\_Grados/gaia.log marte3.log** |

## Ejecución

Se ejecuta desde la línea de comandos:

1. Abre una ventana DOS (cmd.exe)
2. Ejecuta el siguiente comando:

<ruta ficheros aplicación>\proc\_parser.rb <carpeta ficheros log>

### Ejemplo

|  |
| --- |
| C:\Users\j.cheesman\ruby\_scripts\log\_utils\testdata>..\proc\_parser.rb 180213  Processing: marte\_1.log  556629 lines processed.  1466 errors found, rate=0.00263.  Processing: marte\_2.log  443709 lines processed.  1348 errors found, rate=0.00304.  Processing: marte\_3.log  575678 lines processed.  1504 errors found, rate=0.00261.  Writing llamadas: marte\_1  calls total avg best worst name  5148 112632 21 1 19636 es.uned.portal.gaia.core.bd.sql.Usuar…  10969 249456 22 1 5124 es.uned.portal.gaia.core.bd.sql.AlumnoDao.obt…  8110 469072 57 3 6815 es.uned.portal.gaia.core.bd.sql.UsuarioAut…  …  Writing calls: marte\_3  Result file:  log\_180213\_2013219-856.xlsx |

La salida a consola muestra la siguiente información:

1. Número de líneas de log
2. Número de errores y la tasa de errores (número errores / número líneas)
3. Datos resumen de los procedimientos
4. La última línea que sale en la consola es el nombre del fichero Excel generado.

La información mostrada está también incluida en el fichero Excel.

## Datos salida

La información grabada en la hoja Excel incluye 4 pestañas:

1. Datos
2. Errores
3. Llamadas
4. Malformada

A continuación una explicación de cada una.

### Datos

La pestaña de datos incluye estadísticas de los tiempos de respuesta de las invocaciones a procedimientos PL/SQL:

|  |  |
| --- | --- |
| Columna | Descripción |
| Nodo | El nodo corresponde al nombre del fichero log de origen, sin la extensión. Por ejemplo, el nodo “marte1” corresponde a los datos del fichero “marte1.log”. |
| Clase Java | Clase Java que invocó al procedimiento |
| Metodo | Método de la clase Java |
| Nombre proc | Nombre del procedimiento |
| Num Llamadas | Número de veces que se ha llamado el procedimiento desde una clase y método concreto en un nodo |
| Total (ms) | Total tiempo de todas las invocaciones en el nodo |
| Medio (ms) | Tiempo medio de las invocaciones |
| Peor (ms) | Tiempo de la peor invocación (peor = más lento) |
| Mejor (ms) | Tiempo de la mejor invocación (mejor = más rápido) |
| Datos Peor Llamada | Datos reportados para la invocación. |
| Hora Peor Llamada | Tiempo de la peor invocación |
| Datos Mejor Llamada | Datos reportados para la mejor invocación |
| Hora Mejor Llamada | Hora de la mejor invocación |
| Fecha | Fecha del log. Si el log corresponde a varios días se tomará la fecha de la última vez que aparece el procedimiento |

La clave de los tiempos está en la combinación nodo + clase + método + procedimiento; el mismo procedimiento generalmente aparecerá varias veces al invocarse desde otras clases y nodos.

### Errores

Esta pestaña incluye una relación de todos los mensajes de nivel [ERROR] reportados en el log.

|  |  |
| --- | --- |
| Columna | Descripción |
| Nodo | El nodo que ha reportado el error. |
| Hora | Hora del error. |
| Clase Java | Clase Java que reportó el error. |
| Metodo | Método de la clase Java |
| Codigo | Código “ORA”, si tiene. |
| Descripcion | Mensaje reportado del error. |
| Fecha | Fecha del error. |

La inmensa mayoría de los errores no son errores como tal, sino mensajes informando de una falta de permisos, que el estado de la matrícula no permite modificaciones, etc.

### Llamadas

La pestaña de llamadas incluye el número de entradas en el log – normales y errores – en intervalos de 15 minutos.

|  |  |
| --- | --- |
| Columna | Descripción |
| Nodo | El nodo. |
| Hora | Inicio del intervalo de tiempo. |
| Llamadas | Número de entradas en el log durante el intervalo. |
| Errores | Número de mensajes tipo “[ERROR]” durante el intervalo. |
| Fecha | Fecha del intervalo. |

### Malformado

Esta pestaña incluye una lista de todos los procedimientos encontrados con un inicio, pero sin reportar el fin. Esta información se pasa al equipo de GAIA para su inclusión en el desarrollo, y con el tiempo se anticipa que el contenido de esta pestaña desaparecerá.

|  |  |
| --- | --- |
| Columna | Descripción |
| Nodo | El nodo |
| Clase Java | La clase y método (no se han separado) |
| PL/SQL | El procedimiento reportado |
| Fecha | Fecha |

# Aplicación para procesar la tabla de rendimiento

## Introducción

La aplicación log\_procesos permite la generación de datos estadísticos a partir de la tabla posgrado\_pro.log\_procesos. Esta tabla contiene trazas de la ejecución de determinados procedimientos, incluyendo los parámetros de entrada y salida, la hora de ejecución, y el tiempo transcurrido.

La aplicación parsea la tabla, generando datos de los mejores y peores ejecuciones, y promedios.

Permite la generación de estadísticas de las fuentes de datos configurados en el fichero config.yaml.

## Configuración

El fichero config.yaml tiene el siguiente aspecto:

|  |
| --- |
| database\_pre:  user: GAIA\_MANT\_PRO  pwd: xxxxxxxx  SID: ORC5PRE  database\_pro:  user: GAIA\_MANT\_PRO  pwd: xxxxxxxx  SID: ORC5 |

Se puede definir tantas fuentes como sean necesarias, todos deben cumplir con la nomenclatura database\_xxxxx, siendo la cadena xxxxx el parámetro pasado a la aplicación cuando se ejecuta.

## Ejecución

Se ejecuta desde la línea de comandos:

1. Abre una ventana DOS (cmd.exe)
2. Ejecuta el siguiente comando:

<ruta ficheros aplicación>\db\_parser.rb <XXXXX> <dd/mm/yyyy>

**XXXXX** – sufijo del nombre de la fuente de datos configurada en config.yaml

**dd/mm/yyyy** – fecha de interés. Los días y meses con números inferiores a 10 deben incluir el cero a la izquierda.

### Ejemplo

Para generar los datos de preproducción, para el día 3 de mayo:

* db\_parser.rb PRE 03/05/2013

## Datos salida

1. Log4J: Ver http://logging.apache.org/log4j/1.2/ [↑](#footnote-ref-1)
2. Filezilla: http://filezilla-project.org/ [↑](#footnote-ref-2)