MONITOR DEL SISTEMA
Trabajo práctico integrador

TSSI Edwin Alexis Abot
Programación III y Laboratorio de Computación III

23 de mayo de 2017

1. Introducción

Este documento es una presentación del TP final. En conjunto se les presentará un diagrama de clases con el que podrán comenzar el desarrollo del sistema. También se les proveerá de un proyecto NetBeans con todas las librerías que van a necesitar y con la estructura a seguir. En las clases siguientes se presentarán los diagramas del resto de los componentes.

En este trabajo práctico integrador vamos a generar una aplicación que nos permita obtener detalles concretos de nuestra PC. Vamos a observar el sistema operativo y nuestro hardware. También tendremos que generar datos observando el consumo de memoria y CPU además de los datos provistos por sensores de la máquina. Todos estos datos serán persistidos en archivos con formato JSON e implementaremos los mecanismos para exportar los datos en formato CSV y así poder analizarlos en software de planillas de cálculo.

2. Descripción de la aplicación

El objetivo central de esta aplicación es *recolectar datos de la plataforma en la que se ejecuta y persistirlos en un archivo JSON*. Tendrá dos modos de operación: CLI[1] y GUI[2].

2.1. Modo CLI

En el modo CLI recolecta un conjunto de datos inicial (OS, memoria total, etc.) y los almacena en un archivo de texto JSON. Luego captura datos de sensor y carga del sistema con una tasa de refresco en milisegundos. La captura de datos tiene una duración total indicada en segundos. Al terminar el tiempo vuelca los datos recolectados al archivo JSON y termina su ejecución.

La ruta del archivo, el tiempo de toma de datos y la tasa de refresco se indicarán por parámetros. A continuación se listan:

[1] "Command line interface" (interfaz de línea de comandos) son aplicaciones que corren por consola, reciben parámetros de ejecución y no tienen una interfaz gráfica interactiva

[2] "Graphical user interface" (interfaz gráfica de usuario) son aplicaciones que tienen ventanas y que son interactivas

- refresco: es la tasa de refresco en milisegundos (ej: refresco=200)
- duracion: es el tiempo en segundos durante el cual la aplicación capturará datos (ej: duracion=60)
- rutaJSON: es el path donde se escribirá el archivo JSON (ej linux: rutaJSON=/home/usuario/losdatosrecolectados.json)
- rutaCSV (opcional): es el path donde se escribirá el archivo CSV (ej windows: rutaCSV=C:\losdatosrecolectados.csv)

Hay un parámetro más que indica una ruta para un archivo CSV. Si este parámetro está presente se exportará a la ruta indicada los datos de sensor y carga en formato CSV.

2.2. Modo GUI

En el modo GUI extiende el modo CLI agregando una interfaz gráfica que permite ver de forma más amigable los datos del sistema en tiempo real.

Para ejecutar la aplicación en modo GUI se agrega un parámetro más y éste será 'gui' (ej: gui=true). La interfaz deberá contar con un mecanismo que permita variar la tasa de refresco indicada en el parámetro 'refresco'.

3. Descripción de los módulos

A continuación les presento la descripción delos componentes del sistema. habrán cuatro paquetes "Interfaz gráfica", "Persistencia", "Monitor" y "Aplicación"

3.1. Interfaz gráfica

- Una ventana que muestre la siguiente información de la plataforma:
  - Familia del SO
  - Versión del SO
  - Memoria física total
  - Memoria de intercambio total
  - Marca, modelo, arquitectura, cantidad de núcleos físicos y lógicos del CPU
  - Marca y modelo del motherboard
  - Nombre, IP (v4), MAC address y bitrate de cada NIC
- En la misma ventana se debe mostrar datos de los sensores y carga del sistema que se actualizarán con una frecuencia variable.
  - Temperatura del CPU
  - Voltaje del CPU
  - Uso del CPU

- Memoria física y de intercambio en uso
- Promedio de bits transmitidos y recibidos por segundo de cada NIC
- Se debe poder ajustar la frecuencia de actualización de los datos con una resolución de milisegundos

3.2. Monitor

El monitor es el origen de nuestros datos del sistema. Es el que tiene las herramientas para captar los datos de la PC.

Vamos a implementar clases que actúen como contenedores de datos que representarán cada parte monitoreada. Es decir que tendremos un conjunto de superclases que representan el CPU, NIC, Memoria, etc. A su vez estas clases tendrán que ser serializables en JSON y CSV.

Luego extenderemos las superclases en clases más específicas con imlementaciones para Windows y Linux. Luego de analizar la librería OSHI verán que en nuestras superclases vamos a implementar la mayoría de los métodos y que sus subclases tendrán como responsabilidad instanciar los componentes de OSHI correspondientes al OS en el que corre la aplicación.

3.3. Persistencia de datos

Aquí tendremos las clases para trabajar con archivos de texto en los cuales almacenaremos datos con JSON o CSV.

Para la serialización utilizaremos las librerías Apache Commons CSV y org.json para CSV y JSON respectivamente. Éstas nos facilitarán la transformación nuestros objetos en representaciones JSON o CSV.

3.4. Aplicación

Este será el punto de entra de nuestra aplicación. Aquí tendremos nuestra clase “MonitorDelSistema” con el método main(...).

Tendremos que trabajar en el modo de empacar la aplicación de modo que podamos ejecutarla desde la línea de comandos. También haremos uso, por primera vez, del parámetro ‘args’ del método main().

De ‘args’ tomaremos los valores de los parámetros que condicionarán la ejecución de la aplicación.