Informe de la aplicación callcenter

El objetivo del documento es describir lo más preciso posible y de una manera detallada la forma en que se fue construida la aplicación y las herramientas que se utilizó.

# Herramienta de desarrollo

* La aplicación fue construido por el IDE Netbeans versión 8.2, se creó utilizando arquetipos de Spring MVC con la herramienta Maven en la version 3.0.5.
* La aplicación está desplegada en Jboss EAP 7 versión empresarial.
* El compilado y empaquetado es un war call-center-1.0-SNAPSHOT.war en la version java JDK1.8
* El framework a utilizar fueron spring version 4 y apis Junit y Log4j.
* Para herramienta de versionamiento y repositorio git hub.

# Funcionalidades de la aplicación

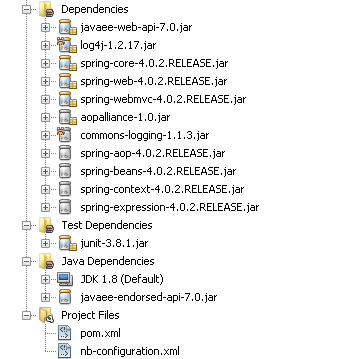
* Es un aplicación web implementando el framework spring web mvc, consta de una vista de entrada “callcenter.jsp” que contiene un formulario con los siguientes campos:
  + Numero de llamada a atender en el callcenter
  + Numero empleados Operadores
  + Numero empleados Supervisores
  + Numero empleados Directores
* Con la información anterior se construye el callcenter y se procesa la información mediante un frontController llamada “**Dispatcher.java**” quien será el encargado de atender la solicitud de peticiones y respuesta a la vista.
* El recurso que recibe la solicitud de la vista en el método “processDispatcher” que valida los datos requeridos del formulario.
* El método inicia con la construcción de una lista que define su dimensión acorde al número de empleados, cada empleado de la lista se especifica la disponibilidad y el tipo de empleado.
* Este método invoca un método llamado “**dispatchCall**” que contiene la lógica principal de la aplicación, su función es atender la lista de llamadas implementando un pool de 10 hilos concurrentes, ejemplo: si son 30 llamadas a procesar, 10 llamadas de estas se ejecutaran al mismo tiempo.
* Antes de lanzar la ejecución valida la disponibilidad de un empleado y lo asigna en un estado ocupado a la llamada para ser procesada.
* Cada llamada puede variar entre 5 a 10 segundos mediante un numero aleatorio, en este caso se simula la duración de la llamada con dormir en hilo de ejecución en el lapso de tiempo asignado.
* Una vez terminada la llamada el empleado que atendió vuelve a un estado disponible para atender posteriores llamadas.
* Cuando se procesa cada llamada en el caso que no encuentre ningún empleado disponible la llamada entra en espera de 5 segundos durmiendo el hilo a ejecutar.
* Terminados los 5 segundos vuelve a consultar disponibilidad en caso que no encuentre se repite el paso anterior, se establece 4 intentos que son 20 segundos si persiste la no disponibilidad, se rompe todo el ciclo de ejecución informando que “la disponibilidad de empleados para atender las llamadas en nula”.
* Terminado la cola de llamadas se apaga el pool de hilos y se informa el tiempo total de procesamiento en segundos.
* Se realiza trazabilidad mediante logs en toda la ejecución de proceso para evidenciar la concurrencia en los hilos y tiempos de ejecución.
* Esta traza queda en log del servidor de aplicaciones JBoss\jboss-eap-7.0\standalone\log\ server.log.
* La aplicación se implementó Junit para pruebas unitarias, en el Test Packages contiene un clase DispatcherTest.java que valida un caso de prueba “**testDispatchCall**”
* Se considera que los puntos extras del ejercicio se han cubrido en el detalle con los puntos explicados anteriormente.

# Instalación y despliegue de la aplicación

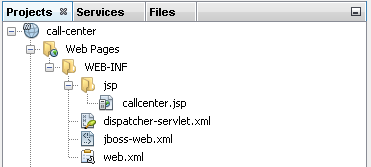
* Aplicación se encuentra en repositorio publico github en la siguiente ruta [**https://github.com/wilmercado87/callcenter**](https://github.com/wilmercado87/callcenter)
* Se descarga callcenter-master.zip, descomprime callcenter-master y se abre en Netbeans 8.2
* El IDE debe tener configurado maven 3.0.5, WildFly Application Server (Jboss EAP 7) y JDK 1.8
* Una vez se tiene la aplicación en el IDE call-center se procede a compilar y generar el artefacto en la ruta **call-center\target\ call-center-1.0-SNAPSHOT.war**
* El war debe ser desplegado en el servidor web Jboss EAP7 en la ruta **\JBoss\jboss-eap-7.0\standalone\deployments\ call-center-1.0-SNAPSHOT.war.deployed** e iniciar el servidor.
* Ejecutar la url **localhost:8080/call-center/viewCallcenter**

# Modelado y Diseño de la aplicación

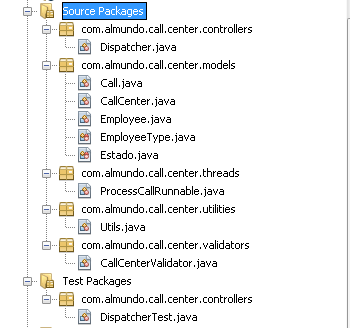
* Librería de dependencia de la aplicación



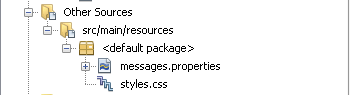
* Componente front callcenter.jsp y archivos de configuración



* Componentes backEnd
  + Controller
    - Dispatcher
  + Models
    - Call
    - CallCenter
    - Employee
    - EmployeeType
    - Estado
  + Threads
    - ProcessCallRunable
  + Utilities
    - Utils
  + Validators
    - CallCenterValidator
  + Test Packages
    - DispatcherTest

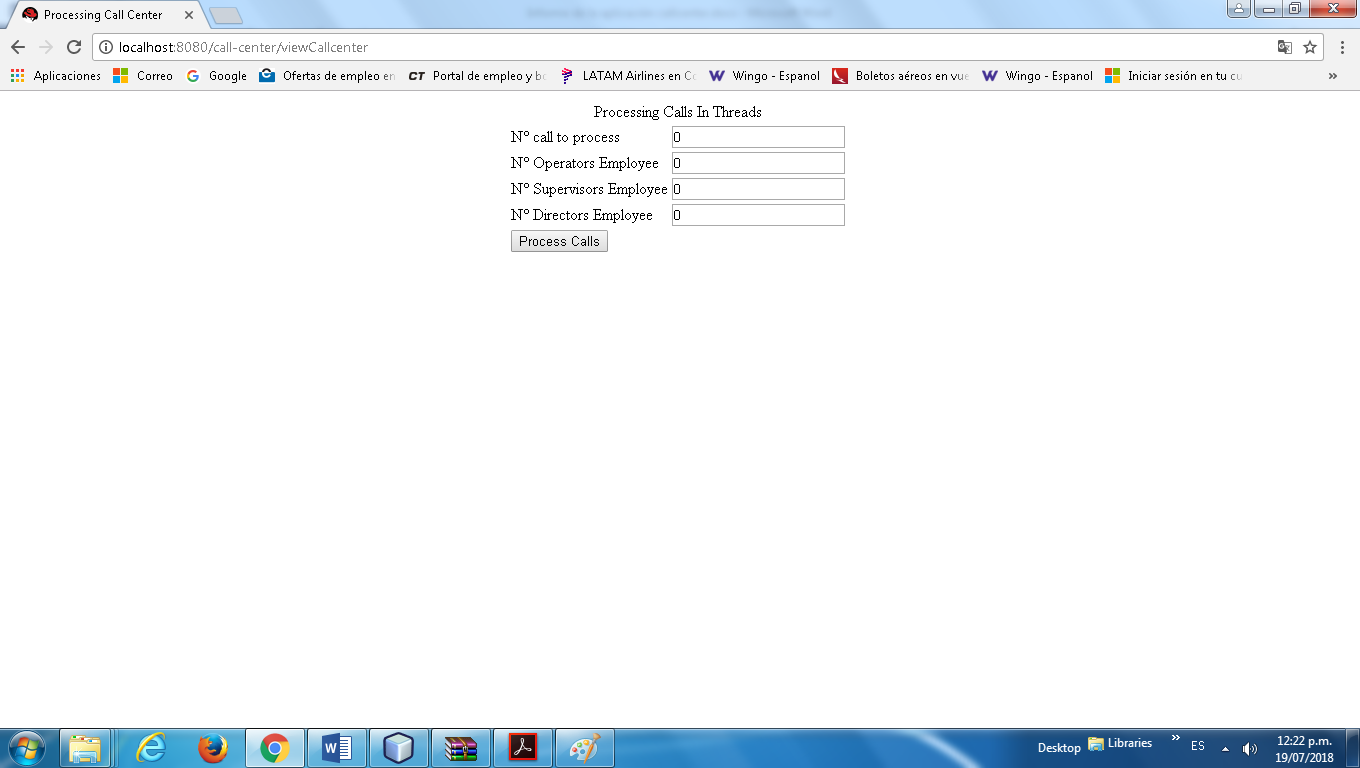


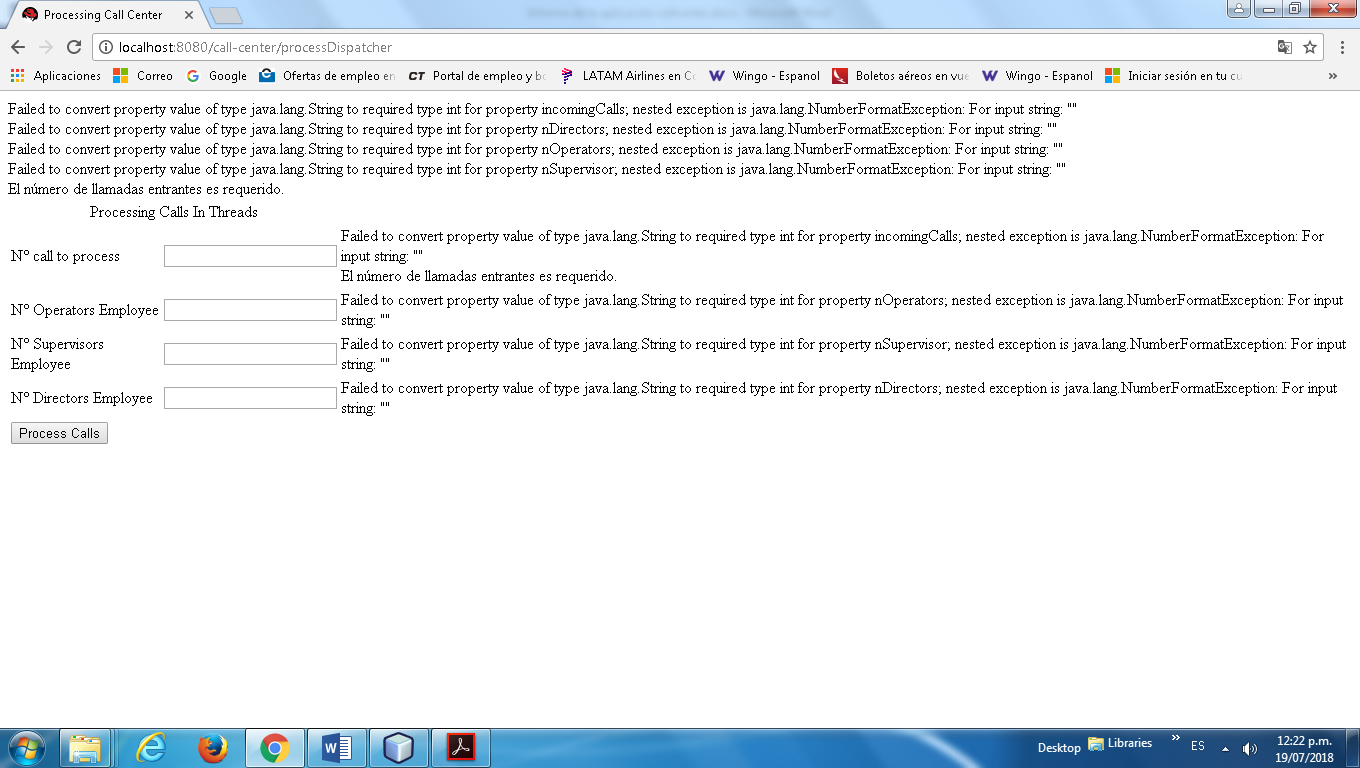
* Otros recursos de la aplicación
  + Resources
    - Messages.properties
    - Styless.css

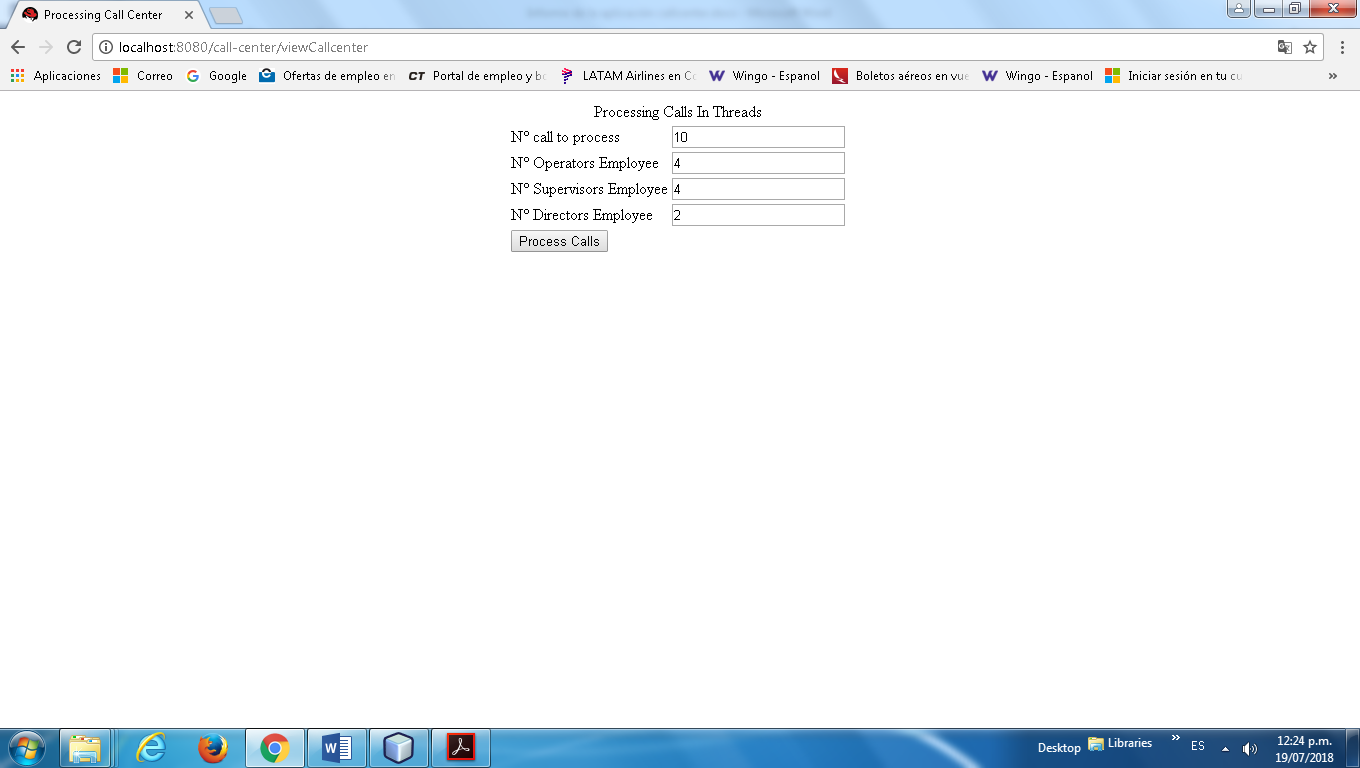


# Evidencia de la aplicación

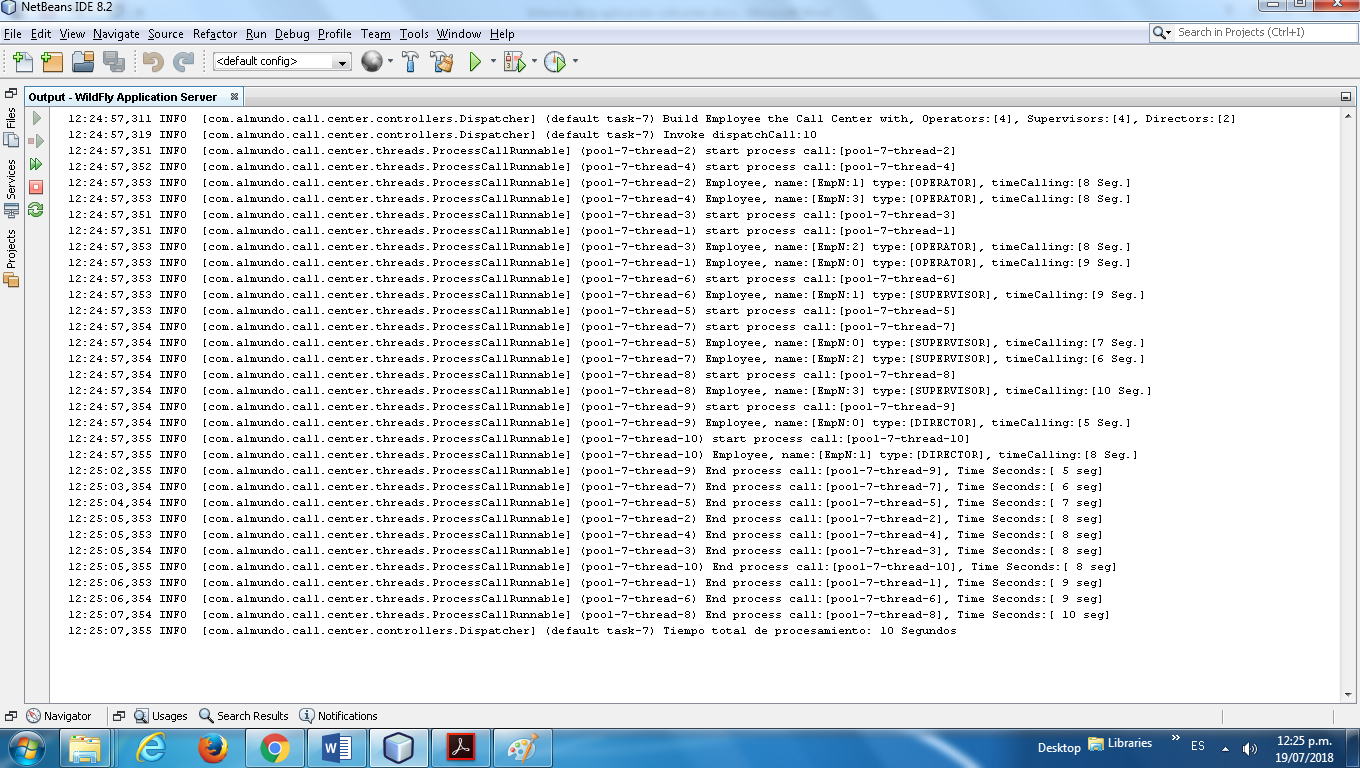
* Se recomienda no colocar números exageradamente grandes puesto que la aplicación trabaja con hilos y bucles, dependiendo estas entradas será el tiempo y procesamiento de estas.



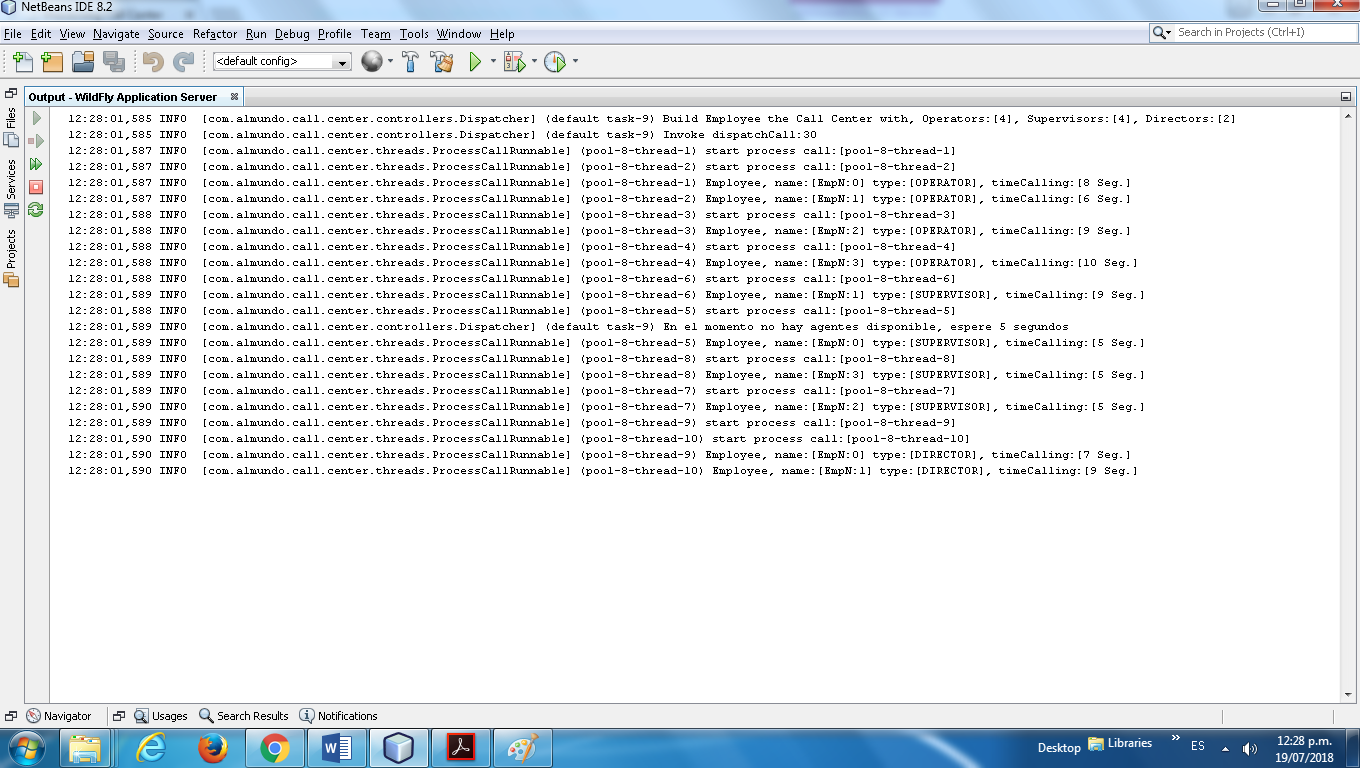




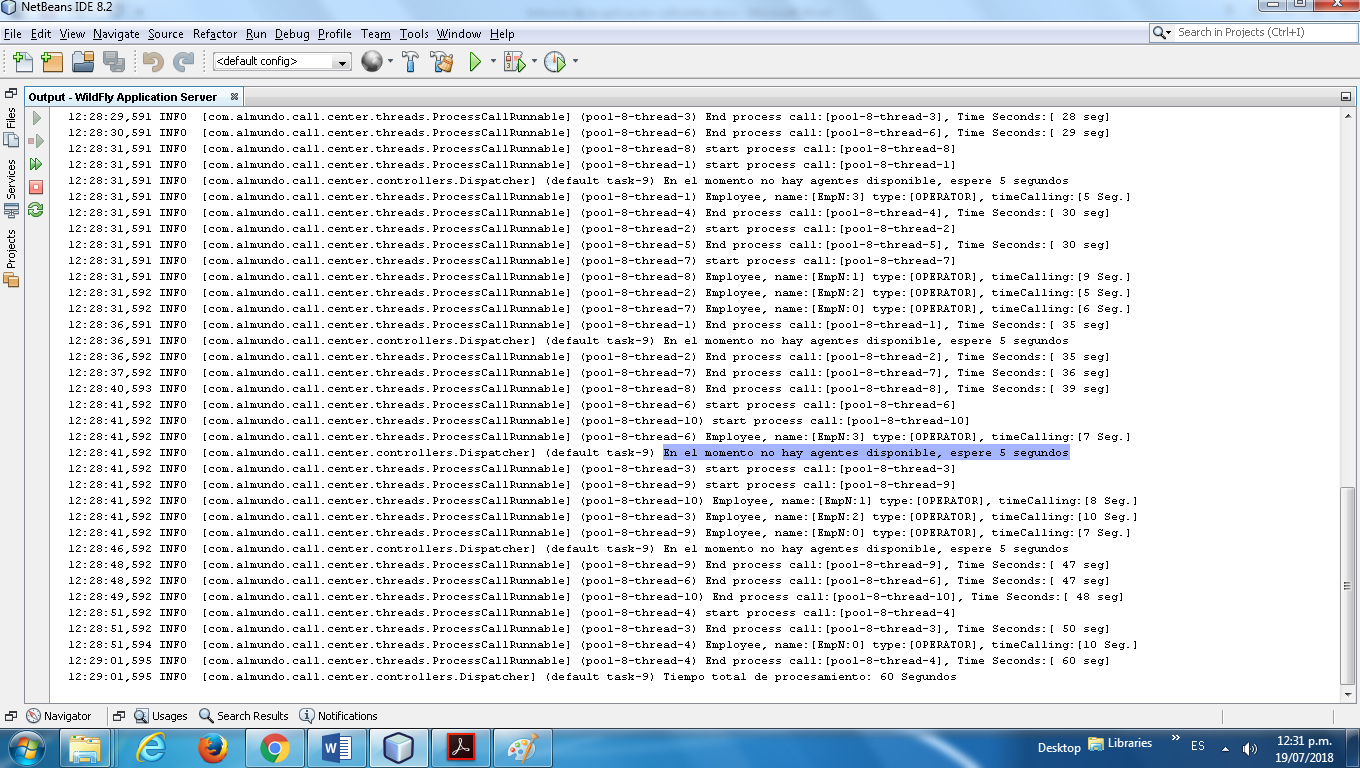
* Procesamiento de 10 llamadas por un pool de 10 hilos concurrentes



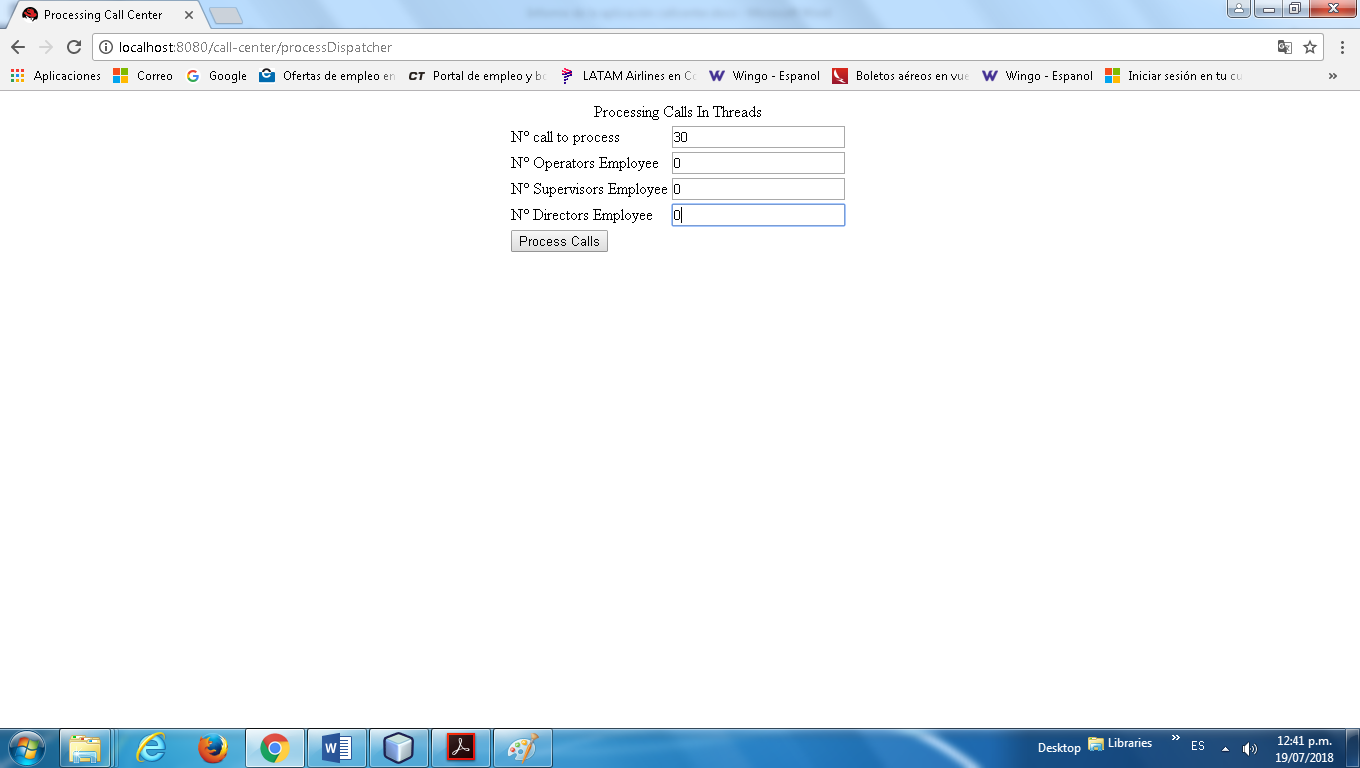
* Procesamiento de 30 llamadas por un pool de 10 llamadas concurrentes con 10 empleados disponibles.



* Evidencia en el momento en que los 10 empleados están ocupados, colocando en espera la llamada, se informa la duración de cada llamada “timeInCalling”, se detecta que algunas llamadas el tiempo es superior a la duración de la llamada, esto se da porque esa llamada quedo en espera por indisponibilidad de empleados, al final informa un tiempo de procesamiento total de 60 segundos.



* Evidencia cuando no hay ningún empleado disponible



* La aplicación realiza 4 intentos en lapso de 5 segundos para un total de 20 segundos, pasado este tiempo rompe todo el ciclo de ejecución.

