# DOCUMENTACION TECNICA

PROYECTO DE CONVIVENCIA GENERALI

4/MAR/2016

# OBJETIVOS

El objetivo del presente documento es indicar el propósito desde el punto de vista de arquitectura de los proyectos y clases principales que se han construidos para la prueba de concepto del proyecto de convivencia entre struts y spring web flow /primefaces.

# PREREQUISITOS

Es necesario para entender el problema haber leído el documento

atSistemas-ConvivenciaFWs-Enfoque.pdf

donde se explica la problemática a resolver y se describe el enfoque.

# DESCRIPCION DE COMPONENTES

Se han construido tres proyectos java:

Primefacespoc y Strutspoc, son proyectos web que funcionan de forma independientes y tienen una implementación de la solución en el framework correspondiente.

seguroHogar-model, es un proyecto jar donde se encuentra definido el modelo de datos compartido por los proyectos webs. El modelo de datos definido aquí se traspasa de una web app a la otra mediante la cache.

La ejecución y las pruebas de estas webapps se han realizado usando tomcat8 como servidor, no se hace uso de ninguna característica especifica de este contenedor, pudiendo ejecutarse cada webapp en tomcats independientes.

Para la ejecución de las pruebas cruzadas entre las dos apps, es necesario una cache y se está usando Redis. Solo es necesario ejecutar el archivo redis-server.exe

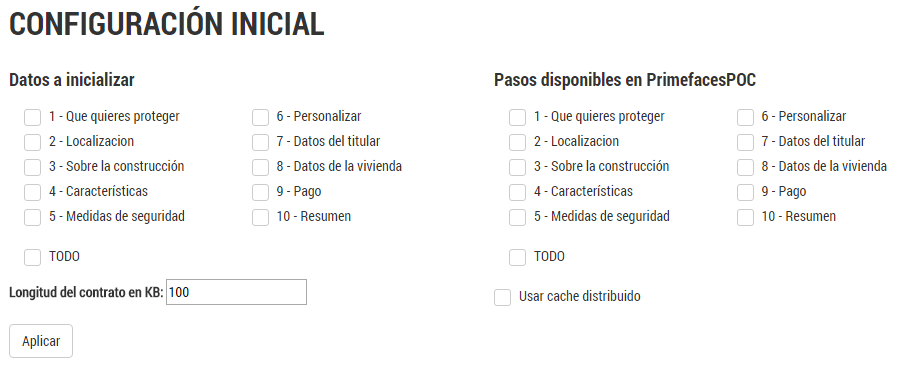
Ambas aplicaciones usan una base de datos embebida H2, de donde extrae los datos maestros que se muestran en los comboboxes como el listado de provincias o el tipo de uso de las viviendas.

# DESCRIPCION FUNCIONAL

Las dos web apps funcionan exactamente igual. Contienen un flujo de configuración de la aplicación y un flujo donde se realiza la operativa de alta de seguro de hogar.

## Flujo iniciar

La opción del menú iniciar muestra la pantalla de configuración.



El flujo de la operativa de alta de seguro cuenta con las 10 pantallas enumeradas en la configuración y que van desde 1-Que quieres proteger hasta 10-Resumen

La sección de “Datos a inicializar” permite que los datos de las pantallas correspondientes queden inicializadas y no tener que introducir esos datos ya que son obligatorios en las subsiguientes pantallas.

La sección de “Pasos disponibles en PrimefacesPOC” determina que pasos se realizaran en la web app de primfaces y cuales en la de struts.

El campo de longitud del contrato es un campo de texto generado aleatoriamente y que permite simular el paso de datos entre pantallas con un volumen de datos adicional de forma arbitraria.

El campo “Usar Cache Distribuido”, permite configurar si se quiere que se utilice la cache distribuida como implementación de la sesión en cada operación o por el contrario solo se usará para el traspaso de datos del modelo entre diferentes aplicaciones.

## Flujo Hogar

La opción hogar, permite arrancar el flujo de contratación de una póliza de hogar. Los datos y la semántica de esta operativa es orientativa y no corresponde con los datos y nomenclatura que usa Generali, solo son válidos para la prueba de concepto.

Todos los pasos excepto “5 - medidas de seguridad” y “10 - Resumen” contienen datos que son obligatorios por lo que no deja avanzar a un paso posterior sin no se introducen. El paso 10 solo visualiza los resultados del proceso. El paso “6 – Personalizar” inicializa los 3 primeros campos antes de entrar a la pantalla, lo cual permite controlar cuando se ejecuta el mismo.

# DESCRIPCION DEL PROPOSITO DE CLASES

## STRUTSPOC

**StrutsFlowAction**, permite gestionar todas las opciones del menú de navegación del menú inferior (Inicio, Anterior, Siguiente, Ultima) y la navegación por pasos según el menú superior. Gestiona además las llamadas y el paso de parámetros entre los diferentes frameworks.

**SessionCacheWebFilter**, este filtro web permite realizar un wrapper sobre las peticiones http (MyHttpServletRequestWrapper), que as su vez hace un wrapper sobre la sesión (MyHttpServletSessionWrapper) que cambia la implementación de la escritura y lectura de datos a una implementación diferente, se han proporcionado dos, un hashmap (HashMapPersistenceDataStore) en memoria y otra que usa la cache redis (RedistPersistenceDataStore).

## PRIMEFACESPOC

**GeneraliWebFlowEngine**, permite dar soporte a todas las opciones del menú de navegación del menú inferior (Inicio, Anterior, Siguiente, Ultima) y la navegación por pasos según el menú superior. Gestiona además las llamadas y el paso de parámetros entre los diferentes frameworks.

**ControlledExit**, Esta excepción se lanza cuando se quiere cortar la visualización de un view-state, durante el procesamiento de on-entry.

**ExternalExit**, Esta excepción se lanza cuando se quiere cortar la visualización de un view-state, durante el procesamiento de on-entry durante una llamada desde otro framework.

**GeneraliWebFlowListener**, Es una clase de soporte que permite vincular un beanPostProcessor (FlowVariablesControllerPostProcessor) a las variables que se definan en un flujo de forma que se inyecten una serie de clases de soporte al controlador, cuando estas extienden BaseWebFlowController.

**CustomFlowHandlerMapping**, Permite que los archivos facelets asociados con un flujo, puedan ser cargados desde el classpath solo esta soportado en JSF 2.1 y no funciona con 2.2.

# QUE HACER PARA ADAPTAR STRUTSPOC A FENIX

Hay dos aproximaciones, la primera es añadir al xml que define el flujo de FENIX un atributo que indique que si ese paso en cuestión se resolverá en el otro framework y modificar el motor para que interprete correctamente este atributo y realice una redirección a la otra app, tal cual hace la STRUTSPOC, lo que implica hacer los mapeos de datos recorriendo el modelo que define este flujo o todos los datos que tiene la sesión asociada. La otra aproximación es dentro de la PRE Action realizar el mapeo de los datos del modelo explícitamente y realizar un redirect a la app. En la action POST se debe de realizar el mapeo inverso de los datos antes de continuar con el flujo.

# JMETER

El proyecto strutspoc contiene una carpeta doc donde están este documento y una carpeta “performance” donde están los planes JMETER y un Excel con los resultados obtenidos en varias pruebas así como unos gráficos de análisis.

El proyecto primefacespoc una carpeta “performance” donde están los planes JMETER de este proyecto.

Para realizar las pruebas de rendimiento es recomendable afinar SWF y PRIMEFACES PARA UN MEJOR RENDIMIENTO.

Buscar en los fuentes la palabra PERFORMANCE

Es necesario configurar JMeter para que acepte más de 5 redirecciones.

Abrir el archivo jmeter.properties y añadir la línea siguiente

httpsampler.max\_redirects=20

# PRUEBAS DE RENDIMIENTO Y ANALISIS

**Recorrer todos los pasos del primero al último**

Peticiones tratadas por segundo

Tiempo de respuesta en milisegundos (menos es mejor)

**Recorrer todos los pasos del primero al último**

**Peticiones tratadas por segundo**

**Tiempo de respuesta en milisegundos (menos es mejor)**

Los resultados obtenidos muestran que durante el periodo de convivencia (mix) el performance cae ligeramente por debajo del escenario final en el cual las mismas aplicaciones estarían migradas completamente a primefaces.