Tecnologías para un control de flujo Web más efectivo

Technologies for a more effective Web flow control

Mabel Medina Rodriguez, Yoenia María Martinez Diaz

Universidad de las Ciencias Informáticas

mmedina, ymartinezd@uci.cu

Resumen

Los flujos Web son secuencias muy comunes en sitios que requieren de varios pasos para completar una transacción. Variadas son las tecnologías que incluyen los medios para desarrollar un flujo en un ambiente Web, cada una aportando diferentes facilidades de uso y proporcionando resultados dependientes del nivel tecnológico de cada plataforma de desarrollo. La utilización de una u otra depende del objetivo que se persiga con la aplicación a desarrollar, del nivel de complejidad que tenga la misma y los conocimientos del responsable de la implementación. *Frameworks* como Spring WebFlow y Spring MVC, ambos módulos componentes de la plataforma Spring, proveen recursos que hacen posible la implementación de un flujo Web con diferencias en cuanto al medio de almacenamiento, a la ubicación de elementos concernientes al flujo, a la dependencia de peticiones URL entre otras competencias. Precisamente estas diferencias representan los factores determinantes que inciden directamente en la decisión de uso pues condicionan el nivel de calidad de los resultados y el grado de aceptación que pueda tener por parte de los clientes.

Palabras clave: AbstractWizardFormController, diferencias, flujo Web, spring, transacción, WebFlow.

Abstract

The Web flows are sequences very common in Web sites that they require of several steps to complete a transaction. There are several technologies that include the forms to develop a flow in an ambient Web, each one contributing different ease of use and providing dependent results of the technological level of each development platform. The use of one or another depends on the objective that are persecuted with the application to develop, of the complexity level that has the same one and the knowledge of the person in charge of the implementation. Framework like Spring WebFlow and Spring MVC, both component modules of the Spring platform, provide resources that make possible the implementation of a Web flow with differences as far as the storage media, to the location of elements concerning the flow, to the request dependency URL among other competitions. Indeed these differences represent the determining factors that affect directly the decision to be used because they condition the quality level of the results and the degree of acceptance that can have on the part of the clients.

Key words*: AbstractWizardFormController, differences, spring, transaction, web flow, WebFlow.*

Introducción

Un flujo Web representa una progresión de pantallas en una aplicación cuyo resultado completa la ejecución de una transacción de negocio determinada.

Para la implementación de un flujo Web se han creado disímiles arquitecturas en el entorno de desarrollo de Java. Entre las plataformas de más repercusión de este popular lenguaje de programación se encuentra Spring, cuyos módulos integradores proveen las herramientas necesarias para llevar a cabo una secuencia de páginas por las que pasa una aplicación en función de la

conversación que mantenga con el usuario, o lo que es lo mismo, un flujo Web (2). Es preciso tener en cuenta qué usar en dependencia de las diferentes variantes y requerimientos que presente una aplicación Web.

La plataforma Spring MVC, con su controlador de alta complejidad, el *AbstractWizardFormController* y la plataforma Spring WebFlow, ambas componentes de la poderosa plataforma Spring, con técnicas particulares, se encargan de manejar los flujos Web con desiguales contextos y objetivos. Las diferencias entre ellas condicionan el uso de una u otra en cada caso que se presente.

Aspectos como el medio de almacenamiento de información, la dispersión o centralización de los elementos concernientes al flujo, el nivel de dependencia de peticiones URL y la capacidad de respuesta de la aplicación en caso de retorno a la página anterior con el botón atrás del navegador son los analizados a continuación para cada una de estas plataformas con el fin de determinar la más adecuada y apta para un desarrollo de flujo Web específico.

Materiales y métodos

Medio de almacenamiento de información

El *AbstractWizardFormController,* como medio de almacenamiento de datos utiliza la sesión, única para cada navegador, que se limpia automáticamente cuando se llega al final de un flujo implementado con este controlador (1). Hay un detalle de particular interés que es la permanencia de recursos en sesión cuando se deja el flujo sin terminar. Esto puede suceder en el caso de una aplicación que contemple como uno de sus módulos un flujo Web además de otros con características específicas y diferentes. Podemos estar corriendo el módulo representativo del flujo y recordar algún detalle que necesitemos entrar al sistema pero que requiera de otra pantalla externa al flujo para dicha operación y una vez que dejemos el flujo en un punto anterior a su término para llevar a cabo la acción citada anteriormente y regresemos posteriormente a la secuencia del flujo en proceso, los recursos asociadas a la misma aún permanecen en sesión y se vuelve a la pantalla en la que se abandonó el flujo pues ya las pasadas fueron llenadas por el usuario antes de abandonar el flujo y la información correspondiente se encuentra almacenada en la sesión. El almacenamiento de datos en sesión impide el soporte por parte de este controlador de un flujo de trabajo multipágina dentro de un mismo navegador (1), pues la sesión, como ya se ha dicho anteriormente, es única para cada navegador, y al desplegar varias ventanas en un mismo navegador se corre el riesgo de que recursos almacenados en este espacio de memoria, con igual nombre, diferente valor y pertenecientes a desarrollos individuales de las páginas abiertas, sean sobrescritos propiciando así la pérdida de información.

Por su parte *Spring WebFlow* optimiza en un gran por ciento este aspecto que introduce serias limitantes en el controlador anteriormente analizado. SWF separa espacios de memoria para cada transacción, independientes unos de otros y que se limpian automáticamente al finalizar la transacción con la que está relacionado. Estos espacios tienen el nombre de conversación y representan varios subconjuntos dentro de una misma sesión (1), dicho en otras palabras, nos brinda la posibilidad de desplegar varias páginas, con desarrollos individuales, que persiguen objetivos diferentes dentro de un mismo navegador, sin que la información de cada uno se sobrescriba o dañe. O sea, cada ventana va a contar con su medio de almacenamiento, ya sea un flujo Web, una pantalla para entrada, para actualización o para recuperación de datos. Ninguna interfiere con los recursos de las demás.

Dispersión o centralización de los elementos concernientes al flujo

En el caso del controlador para flujos Web de *Spring MVC*, todos los recursos que proveen información acerca del flujo y que contribuyen a su correcto funcionamiento, están dispersos por toda la aplicación. Esto puede provocar confusión al realizar un

cambio dentro del flujo pues sería necesario acceder a diferentes lugares para llevar a cabo una sola modificación, lo que también contribuye a la pérdida de tiempo y consecuentemente, de productividad.

Spring WebFlow centraliza todo lo concerniente al flujo en un fichero denominado definición de flujo que se ubica en una sola parte de la aplicación y que contiene la secuencia de páginas incluidas en todos los posibles caminos a seguir por el flujo (3). Aporta visibilidad pues toda esta información concentrada y estructurada en un solo fichero visualiza de forma clara y transparente todo el recorrido del flujo. Es flexible a cambios debido a su alto grado de organización de los datos y promueve la productividad gracias a la centralización de recursos, lo cual provee rapidez y efectividad en la configuración y modificación de información. Es importante destacar que en tiempo de ejecución, se puede llevar a cabo un cambio en el fichero de definición de flujo en un punto al que todavía no ha llegado la aplicación, y dicha modificación es implementada automática por el flujo en ejecución (4), lo que no sucede con el *AbstractWizardFormController* de Spring MVC.

Nivel de dependencia de peticiones URL

El *AbstractWizardFormController* de *Spring MVC* requiere de una petición URL en cada una de las páginas que componen la secuencia, con el índice correspondiente a la página que se quiere acceder. Vale aclarar que las páginas deben ser especificadas con anterioridad en el orden que deben aparecer dentro del fichero de configuración en la zona asociada con la declaración del controlador en uso. El orden de estas páginas puede ser alterado cambiando el índice en las peticiones URL. En caso de que un cambio en la secuencia esté condicionado por una especificación determinada, se puede implementar un método del *AbstractWizardFormController* mediante sentencias ***if*** que lleve a cabo un camino u otro en dependencia de una condición. Por tanto podemos concluir que el *AbstractWizardFormController* presenta un alto nivel de dependencia de las peticiones URL pues se requiere su presencia en cada una de las páginas que componen el flujo (1).

En el caso de *Spring WebFlow*, una vez que se elabore el fichero de definición de flujo que contiene los caminos y recorridos del mismo a través de las diferentes pantallas, no es necesario plasmar peticiones URL en cada una de las páginas. El flujo se auto-controla y guía por el fichero rector de su funcionamiento que es precisamente la definición del flujo (1). Sólo se requiere de una petición URL cuando se va a iniciar la ejecución del flujo para activarla, pero una vez inicializada, todas las páginas incluidas en la definición van a conformar la secuencia estructurada en el fichero guía sin necesidad de peticiones URL. Si se presentara una situación que implicara decisión, el flujo, dentro de la misma definición, puede manejar la condición con herramientas que incluye en su paquete de trabajo, demostrando una vez más su característica de contención y centralización. En fin, con *Spring WebFlow* el nivel de dependencia de peticiones URL está en cero ya que el flujo tiene la capacidad de controlar su propio funcionamiento sin depender de peticiones URL para su progreso.

Capacidad de respuesta de la aplicación en caso de retorno al paso anterior usando el botón atrás del navegador

Utilizando el complejo controlador de *Spring MVC* diseñado para el desarrollo de los flujos Web, si necesitamos volver en medio del flujo a un paso anterior usando el botón atrás del navegador, los recursos ya entrados en páginas anteriores se mantienen en la sesión como es lógico, y los campos de la pantalla que queremos re-visualizar aparecerán con el valor insertado por el usuario en el momento en que accedió a la misma en la mayoría de los casos. Un inconveniente que presenta este controlador en una acción de esta índole es el desagradable mensaje de aviso sobre una posible pérdida de la información al volver en el flujo, por lo que, evidentemente, no hay una seguridad absoluta de recuperación de datos al regresar a un paso anterior en un flujo Web implementado con *Spring MVC* (1). Esto sumado al hecho de que por navegador se tiene una sola sesión o medio de almacenamiento de datos que pueden ser alterados en otras secuencias o páginas del mismo navegador, lo cual incide negativamente en la recuperación de información de las otras ventanas del navegador.

Si usamos la plataforma *Spring WebFlow* para este fin sí lograremos un cien por ciento de seguridad en la obtención o recuperación de información debido a una funcionalidad que presenta la plataforma dentro de su extensa gama de recursos para el desarrollo de flujos, que es la presencia de continuaciones. Las continuaciones vienen siendo copias o fotos de las pantallas por las que pasa el usuario, que van almacenándose en la conversación de forma organizada, de manera que si el usuario quiere acceder a una página en un punto anterior al que está, usando el botón atrás del navegador, puede retornar y recuperar la información sin la presencia de un mensaje de aviso indeseable, pues la pantalla a la que está solicitando el acceso va a estar almacenada como una continuación más dentro de la conversación con todos los datos entrados por el usuario con anterioridad (1). La independencia de medios de almacenamientos por ejecuciones o ventanas dentro de una misma sesión y por ende, navegador, es otra característica que asegura la recuperación de información de manera satisfactoria.

Conclusiones

Ya analizados varios aspectos definitorios en cada una de estas tecnologías, somos capaces de seleccionar cuál usar en el desarrollo de un flujo Web específico. A modo de resumen y como resultado propuesto luego de un estudio de ambas plataformas, en caso de presencia de un proceso representado por un conjunto de páginas secuenciales a través de las cuales el usuario es guiado, el más adecuado a usar es el *AbstractWizardFormController* debido a que no se incluye ningún tipo de complicación que pueda influir de manera negativa en los resultados de la aplicación, mientras que si se requiere un control absoluto y el desarrollo de una compleja navegación multipágina, el más apto para ello es *Spring WebFlow* como plataforma de nueva generación que incorpora nuevas facilidades y mejoras que sigue perfeccionando en la medida en que aparecen versiones superiores que amplían las expectativas de esta joven plataforma.

Referencias Bibliográficas

1. Carrasco Montenegro, Alberto. tutoriales.php?pagina=springWebFlow. *www.adictosaltrabajo.com.* [En línea] 03 de enero de 2006. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=springWebFlow.
2. Donald, Keith, Vervaet, Erwin y Stoyanchev, Ross. reference. *static.springframework.org.* [En línea] octubre de 2007. http://static.springframework.org//spring-webflow/docs/1.0.x/reference/.
3. Ladd, Seth. *Expert Spring MVC and Web Flow.*
4. Palacio, Manuel. [En línea] http://www.manuelpalacio.net/blog/?p=49.
5. Vervaet, Erwin. index. *www.ervacon.com.* [En línea] 01 de febrero de 2007. http://www.ervacon.com/products/swf/intro/index.html.