Una experiencia práctica reutilizando aspectos

A. M. Reina, J. Torres, M. Toro, J. A. Álvarez, and J. M. Nieto

Dpto. de Lenguajes y Sistemas Informáticos,
Universidad de Sevilla,
Avda. Reina Mercedes, s/n
41012 Sevilla, España
{reinaqu, jtorres, mtoro, juan}@lsi.us.es, nulain@yahoo.es

Resumen Este artículo cuenta nuestra experiencia obtenida tras haber realizado un proyecto utilizando la tecnología de aspectos AspectJ, y plantearsenos el hecho de reutilizar los aspectos definidos en el mismo con un nuevo proyecto a desarrollar. En definitiva, comprobar de forma práctica el grado de reutilización que tienen los aspectos en distintas aplicaciones.

1. Introducción

La programación orientada a aspectos (POA) presenta una forma de modularizar los sistemas introduciendo nuevos operadores de composición. Surgió tratando de solucionar dos problemas detectados durante la construcción de software: el enmarañamiento del código y el hecho de que determinados conceptos no se localizaban bien con la programación orientada a objetos, sino que se esparcían entre distintas clases.

Creemos que las aplicaciones web se pueden beneficiar de esta nueva filosofía, centrándose en aspectos que se han vuelto críticos en el desarrollo de las mismas, como son la navegación ([4], [5]) o la seguridad [6]. Para tener experiencia con esta tecnología y probarla con aplicaciones del ámbito de la web, se decidió realizar una aplicación utilizando las herramientas ofertadas por esta comunidad de investigadores.

Tras tener una visión general de las herramientas disponibles, AspectJ [3] es el lenguaje de programación orientado a aspectos más popular, y más extendido. Esto se puede comprobar tanto por el número de mensajes que se intercambian en la lista de discusión que mantienen los creadores de AspectJ (si lo comparamos con el de otras listas tal como la de HyperJ [1]) como por el soporte y el número de versiones, apoyadas en las sugerencias que dan los usuarios, que se desarrollan para mejorar el producto. Por lo tanto, en una primera fase del trabajo es el lenguaje que se ha escogido para hacer las implementaciones de los conceptos.

La decisión de utilizar AspectJ [2] como lenguaje de desarrollo implicaba utilizar una tecnología web que se llevara bien con Java, y claramente, esta tecnología es JSP (Java Server Pages) [7]. Utilizando esta tecnología nos encontramos con algunos problemas, ya que los aspectos de AspectJ no se pueden enlazar directamente con las páginas JSP, debido a que el proceso de entrelazado del tejedor de AspectJ es en tiempo de compilación, mientras que las páginas JSP se compilan en tiempo de ejecución en el servidor web. Pero incluso si llegáramos a solventar esto, logrando que el servidor compilara con `ajc` (el compilador de AspectJ), quedarían aún problemas a los que enfrentarse. El proceso de entrelazado (*weaving*) se realiza en tiempo de compilación, lo que obliga a conocer el código fuente de todos los aspectos al compilar, mientras que las páginas JSP no necesitan saber los fuentes debido a que la compilación es en tiempo de ejecución.

Si consiguiéramos que el proceso de entrelazado fuera en tiempo de ejecución (entrelazado dinámico), se solucionaría este problema. Pero el entrelazado dinámico es más complicado

debido a la necesidad de cambio del sistema mientras la aplicación se está ejecutando. En las aplicaciones aquí desarrolladas, se ha obviado este problema y se ha realizado un tejido estático de los aspectos, dejando la influencia de los mismos a clases Java, fuera de las páginas JSP.

Este artículo se distribuye de la siguiente forma: en la sección 2, se comentan las características principales del caso de estudio 1, un sistema de gestión de prácticas, que ya se propuso como proyecto fin de carrera y que se ha implementado, dando como resultado una serie de aspectos AspectJ. En la sección 3 se comenta un segundo caso de estudio, un sistema de gestión de proyectos fin de carrera que está actualmente en desarrollo. La sección 4 explica cómo hemos podido reutilizar los aspectos definidos en el caso de estudio 1 para el caso de estudio 2. Por último, concluimos el artículo y apuntamos nuestra futura línea de investigación.

2. Caso de estudio 1: Sistemas de Gestión de Prácticas (SAGP)

Como proyecto fin de carrera, se ha realizado una aplicación web orientada a aspectos, utilizando AspectJ. Las prácticas en laboratorio siempre son un problema debido a que hay un número de plazas concreto, que depende del número de ordenadores por aula. Así que, hay que controlar el número de alumnos que acuden a cada grupo. En algunas asignaturas se implementa un sistema mediante el cual los alumnos pueden elegir el grupo de prácticas al que quieren asistir. Los alumnos se pueden apuntar a un grupo de prácticas siempre que haya hueco. Cuando el grupo de prácticas se complete, ningún alumno puede apuntarse al mismo. El procedimiento que se seguía era poner una lista en un tablón del aula, y los alumnos se iban apuntando al grupo correspondiente. Pero esto era una fuente de problemas (letra ilegible, desaparición de la lista, ...). El proyecto fin de carrera planteado pretendía, por una parte, darle una solución informatizada a este procedimiento, y por otra, probar la tecnología de aspectos para estudiar cómo se comportaba esta nueva forma de separar conceptos con una aplicación web. Los objetivos que se marcaron con este sistema se muestran en la tabla 1.

Código	Nombre	Descripción
OBJ-01	Sistema de seguridad	El sistema deberá implementar un mecanismo de identificación de usuarios y restricción de acceso a páginas según el tipo de usuario para garantizar la seguridad de la aplicación.
OBJ-02	Administración de asignaturas	El sistema deberá permitir a los usuarios la administración de las asignaturas, permitiendo crearlas, editarlas, borrarlas, obtener listado de alumnos, consultarlas, definir grupos...
OBJ-03	Administración de los grupos de prácticas	El sistema deberá permitir la gestión de los grupos de prácticas por parte de los profesores, pudiendo crear grupos, eliminar, modificar sus datos, obtener listados de alumnos y mecanismos para que los alumnos se apunten a ellos y consulten sus grupos actuales.
OBJ-04	Gestión de las noticias	El sistema deberá proporcionar un sistema de noticias referentes a las asignaturas, permitiendo publicar nuevas noticias, eliminarlas, consultarlas...
OBJ-05	Gestión de los usuarios	El sistema deberá gestionar los usuarios del sistema, permitiéndoles hacer login, registrarse, modificar sus datos...

Tabla 1. Objetivos del sistema de administración de prácticas

Analizando los objetivos obtenidos, podemos ver que los objetivos OBJ-01 y OBJ-05 requeridos pueden indicarnos en esta fase la necesidad de tratar el concepto *Seguridad* o *Autentificación*.

Objetivo	Código req.	Descripción requisito
OBJ-01	IRQ-01	Información de usuarios.
	RF-01	Login.
	RF-02	Logout.
	RF-04	Registrarse.
	RNF-01	Sistema de Seguridad.
OBJ-05	IRQ-01	Información de usuarios.
	RF-01	Login.
	RF-02	Logout.
	RF-03	Editar perfil.
	RF-04	Registrarse.
	RF-05	Registrar profesor.
	RF-06	Consultar datos usuario.
	RF-07	Eliminar usuario.

Tabla 2. Requisitos que dependen de objetivos relacionados con el concepto de seguridad

A partir de estos objetivos obtuvimos los requisitos mostrados en la tabla 2, que representa los requisitos que dependen de los objetivos OBJ-01 Y OBJ-05. La codificación que se ha utilizado en la tabla es la siguiente: los requisitos que comienzan por IRQ son requisitos de almacenamiento de información, los que comienzan por RF son requisitos funcionales, y los codificados como RNF son requisitos no funcionales.

En la implementación final, y como resultado de la traza de estos requisitos, se han obtenido directamente los aspectos *Authentication* y *AccessRestringer* (Tabla 3) . Pero, además de estos dos aspectos, en implementación hemos separado otros que no estaban previstos en los requisitos iniciales, pero que se han implementado, bien para ayudar en el proceso de desarrollo, bien para mejorar el rendimiento de la aplicación. Así, los aspectos que hemos obtenido durante la implementación han podido clasificarse en cinco categorías distintas, que se detallan a continuación:

1. Aspectos derivados de los objetivos y requisitos (CAT-01).
   Estos aspectos son los que se han podido intuir claramente desde la etapa de análisis de requisitos, y que derivan de los objetivos elegidos y de los requisitos. Principalmente, son los aspectos relacionados con la seguridad.
2. Aspectos de ayuda al desarrollo (CAT-02).
   Se han utilizado para ayudar a depurar la aplicación durante su desarrollo, pero en la fase de explotación no son necesarios. La aplicación puede funcionar perfectamente sin ellos.
3. Aspectos de mejora de rendimiento de la aplicación (CAT-03).
   Surgen para mejorar algunas cuestiones de rendimiento de la aplicación. La aplicación podría funcionar sin ellos, pero tiene un mejor rendimiento con ellos.
4. Aspectos de mejora del tratamiento de errores (CAT-04).
   Aquí se recogen una serie de aspectos, que nos sirven para asegurarnos de la comprobación de algunas condiciones que pueden dar lugar a errores posteriores.
5. Otros (CAT-05).
   Aquí se han recogido aquellos aspectos, que no hemos intuido desde la etapa de análisis de requisitos, pero que han surgido posteriormente en la aplicación.

La tabla 3 muestra la categorización de los aspectos AspectJ que se han implementado en las categorías que se han enumerado anteriormente.

Categoría	Código	Nombre	Descripción
CAT-01	ASP-11	Authentication	Implementa un mecanismo no intrusivo de autentificación en el sistema. El usuario debe identificarse al principio y luego el sistema comprobará de forma transparente la autenticidad del mismo siempre que se carga una página.
	ASP-12	AccessRestringer	Implementa el mecanismo de seguridad para restringir el acceso a las páginas dependiendo del tipo de usuario.
CAT-02	ASP-03	Logger	Es un aspecto que se utiliza durante el desarrollo de la aplicación, y escribe en un archivo de log para permitir seguir una traza cuando se ejecutan determinados métodos.
	ASP-04	Profiler	También se utiliza como ayuda a la hora del desarrollo, realizando una traza de las llamadas a métodos que se realizan en el sistema, anotando los tiempos de ejecución de cada uno.
	ASP-08	Tracer	Se utiliza durante el desarrollo, y escribe una traza de todas las ejecuciones de los métodos, indicando su profundidad.
CAT-03	ASP-01	Pooling	Es un aspecto que se ha implementado para mejorar el rendimiento y que implementa un mecanismo para mantener un *pool* de conexiones.
	ASP-02	ConnectionChecker	Se utiliza para comprobar la validez de las conexiones obtenidas con el aspecto ASP-01.
	ASP-10	CacheNews	Implementa una caché para el sistema de noticias, de forma que, evita un acceso a la base de datos si la noticia ya está en la caché.
CAT-04	ASP-10	NullChecker	Se utiliza dentro del sistema de seguridad, para comprobar que los parámetros con los que se llamen a algunos métodos no sean nulos.
	ASP-06	CodeSegregator	Se utiliza para evitar que se usen operaciones de la base de datos fuera de este paquete.
	ASP-07	ExceptionHandler	Se utiliza para tratar de forma más compacta las excepciones que se produzcan al intentar ejecutar sentencias SQL. El objetivo es manejar estos errores de manera uniforme, redireccionándolos todos a la misma página.
CAT-05	ASP-09	InfoPrinter	Sirve para homogeneizar la impresión de información de clases distintas.

Tabla 3. Clasificación de aspectos según categorías

3. Caso de estudio 2: Sistema de Gestión de Proyectos Fin de Carrera (SAGPFC)

Tras la experiencia positiva con los aspectos presentada en el caso anterior, nos surgió la necesidad de realizar otra aplicación. Desde el principio nos hemos planteados darle un enfoque orientado a aspectos para comprobar cuántos de los aspectos que surgieron en el caso de estudio presentado en la sección 2, van a poder ser reutilizados en esta nueva aplicación, que pretende cubrir la necesidad de gestionar los proyectos fin de carrera que se leen en el Departamento de Lenguajes y Sistemas Informáticos.

El nuevo proyecto surge debido a que no existe un control informatizado de los proyectos fin de carrera leídos en el departamento, y se quiere gestionar una biblioteca donde se recojan y controlen información de los mismos ya sean para localizarlos en la biblioteca del departamento, como para tener datos administrativos acerca de sus calificaciones, el tribunal que lo evaluó, el tutor, ... El conjunto de objetivos que se han fijado para el sistema se muestran en la tabla 4.

Código	Nombre	Descripción
OBJ-01	Gestión PFC presentados en el Dpto. LSI	El sistema deberá gestionar la información correspondiente a los proyectos de fin de carrera finalizados en el Departamento LSI: autores, tutores, temas tratados e información relevante del mismo.
OBJ-02	Sistema de seguridad	El sistema deberá implementar un sistema de identificación de usuarios y restringir el acceso a determinada información dependiendo del tipo de usuario.
OBJ-03	Gestión de los usuarios que tienen acceso al sistema.	El sistema deberá gestionar la información sobre los distintos usuarios que tienen acceso al sistema.
OBJ-04	Sistema de consultas	Permitir las consultas sobre datos de proyectos leídos ante un tribunal de LSI.

Tabla 4. Objetivos del sistema de gestión de PFC

4. Reutilización de aspectos.

Comenzando en el análisis de requisitos, podemos ver claramente, que las dos aplicaciones tienen algunos objetivos comunes. En concreto, aquellos que hemos visto que en el caso de estudio 1 dan lugar a los aspectos relacionados con la seguridad. El OBJ-01 (sistema de seguridad) de SAGP coincide con el objetivo OBJ-02 de SAGPFC, y de la misma forma, el objetivo OBJ-05 (gestión de los usuarios) de SAGP coincide con el objetivo OBJ-03 de SAGPFC.

Tras analizar los aspectos ASP-01 al ASP-08 de SAGP, podemos darnos cuenta que pueden incluirse en SAGPFC sin ningún tipo de modificación. Es decir, hemos podido incluir directamente y sin ninguna modificación aquellos aspectos que pertenecen a las categorías CAT-02, CAT-03 y CAT-04, las de ayuda al desarrollo, de mejora de rendimiento y de mejora del tratamiento de errores, respectivamente. En números, podemos decir que hemos podido reutilizar un 83 por ciento de los aspectos definidos en SAGP.

El hecho de poder reutilizar estos aspectos, nos sugiere la creación de una biblioteca de aspectos, que nos ayude durante la etapa de desarrollo de cualquier proyecto software.

En cuanto al aspecto de autentificación, tenemos que mirarlo con un poco más de detalle, ya que en él hay entremezclada alguna característica de navegación,ya que se incluye hacia dónde se redirigen los usuarios en caso de no estar autorizados a realizar una operación, que no es más que una forma de navegar hacia una página indicando que no hay permiso para acceder a la característica concreta.

Hay otro aspecto , el *InfoPrinter*, que no se ha podido reutilizar, porque surgió en tiempo de implementación para mejorar la modularidad del código, pero tiene que ver con cuestiones muy concretas relativas a SAGP.

5. Conclusiones

Observando estos ejemplos, podemos ver que hay conceptos, que se pueden intuir desde fases tempranas del ciclo de desarrollo del proyecto, tal como el de autentificación o seguridad; en cambio, hay otros que han surgido durante el desarrollo, bien para ayudar al propio desarrollador, bien para mejorar el rendimiento de la aplicación.

En cuanto a la reutilización de los mismos, hemos comprobado que se pueden reutilizar completamente los aspectos de ayuda al desarrollo, de mejora del rendimiento y de gestión de errores. Sin embargo el aspecto de autentificación podría necesitar alguna modificación, especialmente, porque hay un componente de navegación incluido en el mismo. Es decir, se detalla hacia dónde se va (o se navega) cuando un usuario no es autentificado. Creemos que sería necesario desligar este componente del aspecto de autentificación.

También tenemos un aspecto, que no hemos podido reutilizar, debido a que está demasiado relacionado con la funcionalidad de la aplicación, y a que surgió para tener una mejor modularización del código.

En cuanto a las líneas de investigación, uno de nuestros focos es la navegación como un aspecto. También queremos dar solución a los problemas planteados en la sección 1 acerca del tejido dinámico.

Referencias

1. IBM Corporation. Hyperj home page. web, 2002.
2. G. Kiczales, E. Hilsdale, J. Hugunin, M. Kersten, J. Palm, and W. G. Griswold. Getting started with AspectJ. *Comm. ACM*, 44(10):59–65, October 2001.
3. Xerox PARC. Aspectj home page. web, 2002.
4. A. M. Reina and J. Torres. Analysing the navigational aspect. In Pascal Costanza, Günter Kniesel, Katharina Mehner, Elke Pulvermüller, and Andreas Speck, editors, *Second Workshop on Aspect-Oriented Software Development of the German Information Society*. Institut für Informatik III, Universität Bonn, February 2002. Technical report IAI-TR-2002-1, ISSN 0944-8535.
5. A. M. Reina and J. Torres. Separating the navigational aspect. In Mehmet Akşit and Ziéd Choukair, editors, *Proc. 2nd Int'l Workshop on Aspect Oriented Programming for Distributed Computing Systems (ICDCS-2002), Vol. 2*, July 2002.
6. A. M. Reina, J. Torres, and M. Toro. Aspect-oriented web development vs. non aspect-oriented web development. In *AAOS 2003:Analysis of Aspect-Oriented Software (ECOOP 2003)*, July 2003.
7. M. Roth and E. Pelegrí-Llopart. *Java Server Pages Specificacion. Version 2.0*. Sun Microsystems, 2002.