



# Universidad de Montemorelos

## Facultad de Ingeniería y Tecnología

## Implementación de seguridad en Examen de admisión en línea de FACSA usando Spring Framework 3.

Samuel Martínez Serafín

Ignacio Cruz Domínguez

*Resumen* ..... 3
INTRODUCCIÓN ..... 3
Antecedentes ..... 3
Definición del problema. ..... 3
Justificación ..... 4
Objetivos ..... 4
Hipótesis ..... 4
Fundamentos teóricos ..... 4
Marco Teórico ..... 4
Estado del arte ..... 6
Resultados ..... 7
Metodología ..... 7
Resultados ..... 10
Conclusiones ..... 10
Discusión ..... 10
Apéndices ..... 10
referencias ..... 16

# Implementación de seguridad en Examen de admisión en línea de FACSA usando Spring Framework 3.

Martinez, Samuel.
1110475@alumno.um.edu.mx
Facultad de Ingeniería y Tecnología

***Resumen*— El uso de las tecnologías de la información ha facilitado el contacto entre el alumno y el maestro, gracias a nuevas a las nuevas tecnologías desarrolladas se ha obtenido una mejor educación. Sin embargo en el área de evaluación al estudiante hay un aspecto muy importante a considerar, la seguridad. En una aplicación que aplique exámenes a los alumnos es vital considerar los diferentes puntos de seguridad. En la Facultad de Medicina de la Universidad de Montemorelos se esta desarrollando una aplicación web que aplicará el examen de admisión a los aspirantes de dicha escuela, el desarrollo fue hecho utilizando la tecnología JSP. Esta tecnología aunque en sus inicios fue lo mejor que se tenia, actualmente no es muy popular entre los desarrolladores ya que cuenta con inconvenientes de seguridad como de código. Por lo tanto en este articulo se describe la implementación de aspectos de seguridad al proyecto utilizando Spring Framework 3, en el desarrollo del proyecto se configuraron tanto los aspectos de seguridad, como estándares o convenciones de programación, además de que se aplico el modelos de capas MVC para una mejor integración y comprensión del proyecto así como para facilitar el trabajo en las futuras modificaciones y el uso de clases con métodos genéricos para evitar la reutilización de código.**

***Índice de Términos—En linea, examen, web.***

## I. INTRODUCCIÓN

### *A. Antecedentes*

El uso de la tecnología en diferentes ámbitos de la vida diaria ha venido a dar un gran giro en la forma en la que se realizan algunos procesos. En la educación el uso de tecnologías de la información desde la mitad de los años 70´s ha aumentando y esto a favorecido a la obtención de una mejor educación.[1] Entre las herramientas que los educadores han ido adoptando encontramos el correo electrónico y las video conferencias las cuales han alterado grandemente los ambientes educacionales. Ya que los limites tradicionales como la geografía, los tiempos, etc. Han sido cruzados de manera inimaginables, con lo cual se ha alcanzado una educación mas alta. [1]

No obstante la aplicación de exámenes en línea aun es tema un poco difícil de tratar de modo en línea, ya que algunos autores señalan un examen en línea una oportunidad para hacer trampa o copiar, [2] y algunos otros ven los exámenes en línea como una perdida de tiempo por parte del maestro, ya que la creación conlleva una gran inversión del tiempo que se ve afectada cuando el alumno hace trampa. [3]Por lo tanto es esencial que al desarrollar un aplicación que genere exámenes en línea se prevean y protejan los diferentes riesgos y puntos vulnerables de seguridad, entre estos puntos se incluyen: capturas de pantalla, uso de navegadores web para consultar las preguntas, etc.

Aunque la adopción de tecnologías por parte de educadores ha favorecido a la educación en muchos aspectos: comunicación no presencial con el alumno, asesoría en línea, exámenes no presenciales, etc. Es necesario que los diferentes puntos de inseguridad en las aplicaciones que son utilizados para generar evaluaciones sean vigilados, de manera tal que al evaluar las repuestas de la evaluación, estas sean calificadas de manera eficiente a fin de que la calificación que se obtenga sea la justa al examen.

Es fundamental por lo tanto al tener un examen en línea, manejar la seguridad en las diferentes áreas que pueden ser alteradas desde un navegador web, las cuales pueden ser, modificación de los parámetros de la session web, soporte contra la inyección de sql desde el navegador y uso del navegador para la búsqueda de las respuestas.

### *B. Definición del problema.*

Este proyecto busca solucionar cómo priorizar la seguridad en el examen en línea que esta siendo desarrollado, enfocándose primordialmente en el uso del navegador por parte del usuario, ya que factores externos a la aplicación como el uso de dispositivos móviles para buscar respuestas, uso de material escrito o impreso están fuera de los limites.

Entre los malos usos que se le pueden dar al navegador podemos encontrar:

Inyección SQL: es una de las vulnerabilidades que ha sido descrita como uno de los problemas mas grandes para aplicaciones web.

Una inyección de sql se realiza cuando la información capturada por el usuario contiene un query sql, después cuando la información sea procesada esta será tratada como código sql.

Actualmente la aplicación que genera el examen de medicina, tiene muy poco software que garantice seguridad, principalmente por la tecnología por la que se opto para el desarrollo de la aplicación fue jsp´s. Aunque fue una buena tecnología en un principio, [4] que además llego a ser tomada como la tecnología estándar para desarrollar aplicaciones comerciales, [5]aunque con el paso del tiempo no dio abasto para los nuevos requerimientos que los programadores necesitaban. Aunado a esto el proceso de creación de software tiene que adecuarse a los continuos cambios de las

tecnologías, lo cual implica un mayor esfuerzo en el diseño de la aplicación, a fin de satisfacer las nuevas exigencias de las tecnologías. De esta manera las tecnologías exigen cada vez requisitos de escalabilidad, seguridad y eficiencia, los cuales no fueron cubiertas por la tecnología de jsp[6].

### *C. Justificación.*

Como lo señala en su reporte WhiteHat la capa Web es el blanco numero uno para ataques maliciosos. Porque ahí es donde esta lo mas valioso que se tiene, los datos.[7] Por lo tanto lo que se recomienda a la hora de realizar un proyecto que ocupe la capa web es evaluar los puntos de seguridad que ofrece la tecnología que será usada para el desarrollo de la aplicación y así encontrar los puntos débiles que podríamos tener [7] para después buscar la manera de prever los posibles ataques que la aplicación podría sufrir. Por otro lado la seguridad en una aplicación tiene dos objetivos principales: primero, prevenir que personal no autorizado acceda a información en un clasificación mas alta que su autorización. Segundo evitar que el personal haga publica información clasificada.

Por ello es indispensable que el examen de admisión tenga un software que avale tanto la seguridad de la información que es presentada en el examen por medio de preguntas, como la información que es recibida en le examen por parte del aspirante al resolver el examen.

Así mismo los beneficios que se obtendrán serán hacia los evaluadores, ya con un software de seguridad que monitoree el examen, se podrá prevenir posibles ataques o intentos de obtener la respuesta del examen, por métodos anteriormente comentados.

### *D. Objetivos*

Objetivo general:

Implementación de software de seguridad en examen en línea de la Escuela de Medicina de la Universidad de Montemorelos mediante el uso de Spring Framework 3.

Objetivos específicos:

- Definir puntos vulnerables de seguridad.
- Implementación de Spring 3.
- Creación de filtros y listeners.
- Reutilizar código ya existente.

### *E. Hipótesis.*

Es posible integrar los aspectos de seguridad de Spring Framework 3 a una estructura existente.

## II. Fundamentos teóricos.

### *A. Marco Teórico*

Un framework es una jerarquía de clases con un modelo de interacción entre los objetos instanciados a partir de si mismo. Además un framework invierte la idea de la reutilización de componentes. En lugar de un programador escribiendo un programa principal que llama a procedimientos reutilizables, un programador crea instancias de objetos de la jerarquía de clases del framework y luego proporciona procedimientos para el framework que desea llamar. Por tanto, un framework es un programa de aplicación genérica que un programador adapta proveyendo rutinas altamente especializadas que son llamados por el framework. [8]

Esta definición implica que un framework esta incompleto, los frameworks necesitan ser adaptados para conocer los requerimientos de la aplicación. Un framework es diferente de una librería. Las librerías proveen de funciones o funcionalidades que son llamadas por la aplicación. En contra parte un framework provee rutas de ejecución que son adaptadas por el desarrollador de la aplicación. Cada ruta de ejecución es conocido como flujo de control. Un framework al menos tiene un flujo de control abstracto. [8]

Un flujo de control abstracto es un flujo de control que no se ejecuta hasta su fin, en cambio llama una operación abstracta o envía un mensaje a un objeto no especifico. El flujo de control abstracto es un importante concepto que ayuda a distinguir entre un framework y una aplicación. La idea de tener un flujo de control abstracto que eventualmente ejecuta las instanciaciones de código de los framework es llamada inversión de control.

Los frameworks como se había mencionado necesitan ser adaptados para especificar los requerimientos de la aplicación. El resultado de adaptar un framework a los requerimiento de una aplicación en particular es llamado instanciación del código.[8]

El desarrollo de aplicaciones web aplicando frameworks ha llegado para ayudar al programador a agilizar su trabajo ya sea a la hora de emplear las funcionalidades con las que cuenta el framework o implementar convenciones para mejorar la estructura del proyecto. Algunos de estos frameworks son, Spring, Struts, Lift, Play, .Net, etc.

Spring Framework es muy demandado actualmente por empresas para el desarrollo de sus aplicaciones, ya que como lo menciona su documentación Spring es una plataforma java que proporciona una solución ligera y potente para el desarrollo de aplicaciones a medida para empresas.

Aunado a esto proporciona un amplio soporte para el desarrollo de la infraestructura de la aplicación, de esta manera Spring se centra en el manejo de la infraestructura para que así el desarrollador se concentre en la programación de la aplicación. [9]

A diferencia de otros frameworks de un solo nivel como Struts o Hibernate, Spring tiene como objetivo estructurar las

aplicación entera en una sola contante, de manera productiva, reuniendo los mejores frameworks de un solo nivel para crear una arquitectura coherente. [10]

Spring es a la vez el más popular y más ambicioso de los frameworks de peso ligero. Es el único que aborda todos los niveles de arquitectura de una aplicación típica de Java, y el único en ofrecer una gama completa de servicios, así como un contenedor ligero.

Los siguientes son los principales módulos de Spring :

- Contenedor de inversión de control(IoC): es el núcleo o contenedor de Spring y permite la gestión de configuración sofisticada por medio de POJO's.
- Framework con programación orientada a aspectos: permite un comportamiento que de otro modo se dispersa a través de diferentes métodos para ser modularizado en un solo lugar.
- Abstracción de acceso a datos: Spring fomenta una arquitectura consistente en el acceso a datos y proporciona una abstracción única y poderosa para ponerlo en practica.
- Simplificación de JDBC: Spring provee una capa de abstracción sobre JDBC que es significantemente mas simple y menos propenso a errores que usar solo el JDBC cuando es necesario el uso de base de datos relacionales SQL.
- Manejo de transacciones: Esta abstracción proporciona un modelo de programación consistente en una gama amplia de ambientes y es la base para la gestión de transacciones declarativa y programática de Spring.
- Framework MVC web: Spring proporciona un framework de desarrollo MVC web basado en peticiones. El uso de instancias compartidas multiproceso "Controllers" es similar al enfoque que usa Struts, pero Spring es mas flexible y se integra perfectamente con el contenedor IoC de Spring.
- Simplificación del trabajo con JNDI, JTA y otras api Java: Spring ayuda a eliminar gran cantidad de código verboso, repetitivo que "no hace nada".
- Ligera interacción remota: Spring proporciona soporte para la comunicación remota basada en POJO en una amplia gama de protocolos, incluyendo, RMI, IIOP, Hessian, Burlap, y otros protocolos de servicios web.
- Soporte JMS: Spring proporciona soporte para envió y recepción de mensajes JMS de una manera mucho mas simple que la manera proporcionada por el estándar Java.
- Soporte JMX: Spring soporta la configuración de objetos de la aplicación a través de JMX.
- Soporte a una estrategia integral de pruebas para desarrolladores.

Spring Framework es modular permitiendo usar solo las partes que sean necesarias, por lo tanto para el desarrollo del software de seguridad solo será necesario la implementación del modulo de seguridad conocido como Spring Security.

Spring Security es una librería java para la autenticación y autorización.[11] Spring Security ha probado que funciona bien tanto para grandes empresas como para pequeños emprendedores. Gracias a la popularidad que tiene por llevar mas de 10 años en el mercado sistemas bancarios y empresas de telecomunicación.

Por otra parte Struts framework es un framework que permite crear aplicaciones con grandes funcionalidades de manera rápida. Sin embargo solo puede ser usado para integrar el desarrollo de la aplicación, ya que para complementar adecuadamente el desarrollo de la aplicación será necesario agregar herramientas personalizadas.[12] Sin embargo Struts reduce los vínculos con el uso de especificaciones servlets, haciendo el proceso de prueba substancialmente fácil. Además permite la inyección de dependencias en muchos niveles, esto significa que tanto la capacidad de prueba y reusabilidad son mejores.

El ciclo de peticiones a la aplicación puede ser resumido de la siguiente manera, las peticiones a la aplicación son filtradas mediante interceptores e implementadas por los actions. Los actions regresan results, los cuales son ejecutados y enviados de regreso al navegador web.[12]

Bajo estándar de configuración Struts procesa todas las peticiones entrantes a la aplicación, se implementa como si fuera un filtro y se asigna a todas las peticiones.

Los interceptores son parecidos a los filtros de los servlets, pero específicamente para Struts los interceptores pueden ser configurados para la aplicación completa, un grupo de actions, solo un action o cualquier combinación de los mismos. Los interceptores proporcionan la mayor parte de la funcionalidad de Struts. La mayor parte de las cosas interesantes están en los interceptores.[12]

Los actions son clases java que pueden ser tratados como si fueran un objeto, tienen las siguiente características:

- No están ligados a las especificaciones servlet.
- No requieren el uso de cualquier constructor de Struts
- Manejan elegantemente la información y regresan simples Strings al navegador web.

Los results determinan que será enviado de regreso a navegador web, típicamente un jsp que produce HTML. No obstante Struts tiene muchos otros tipos de results además del envió estándar de un jsp. Entre los results podemos encontrar, redirection, redirection a otro action, action chaining, template de FreeMarker, un archivo, JasperReports y mucho mas. Además es posible agregar un result personalizado si fuera requerido.

En resumen las características de Struts le dan al desarrollador muchas maneras de aumentar la funcionalidad, manteniendo el desarrollo a un costo bajo y al cliente feliz.

Por otro lado la iniciativa .Net fue lanzada por Microsoft para habilitar la entrega de soluciones orientadas al usuario por parte del desarrollador. La mayor ventaja de esta iniciativa es la habilidad de proveer una solución personalizada que permite al desarrollador desplegar aplicaciones que armoniza perfectamente con los requerimientos del usuario.[13]

La iniciativa .Net surgió del cambio de desarrollo computación de escritorio a computación distribuida. En computación distribuida un numero de aplicaciones son integradas para proveer una solución. Por ejemplo si se necesita mostrar una lista de los últimos libros publicados por algún escritor, lo que se podría hacer es crear un sitio web almacena y muestra los detalles de nuevos libros.[13]

El uso del framework .Net tiene grandes ventajas como, un gran framework de programación para el desarrollo de aplicaciones para clientes Win32, una plataforma unificada de desarrollo web que provee los servicios necesarios a los desarrolladores para crear aplicaciones Web de tipo empresarial, soporte de estándares basados en XML, acceso al Active Directory por medio de código, soporte para globalización y localización de archivos, la habilidad para pasar archivos por valor o referencia entre aplicaciones distribuidas, etc. [14]

Las desventajas que se pueden encontrar son, necesidad de comprar licencias para uso, uso de ide propio, defectos en el garbage collector. [14]

Finalmente la filosofía de .Net, es ayudar a proveer un acercamiento lógico al desarrollo exitoso de soluciones empresariales. La filosofía intenta ayudar a los desarrolladores en la programación aplicaciones flexibles y escalables.

Existe otro framework llamado Play! El cual se define a si mismo como un framework web puro y full-stack en la maquina virtual de Java(JVM). Play! Ayuda al desarrollador a crear aplicaciones web utilizando tecnologías de ultima generación. [15]

Play! Ofrece a los desarrolladores una experiencia muy positiva a la hora de crear aplicaciones web. Esta experiencia viene con la visión fresca y ordenada que Play! Tiene sobre la forma en la que debe el desarrollo debe hacerse.

Uno de los beneficios que Play! Ofrece es el corto tiempo de sobrecarga al editar código y ver el resultado en una pagina web. Esto es gracias a la recarga en caliente de los archivo fuente y que los errores de compilación son mostrados en el navegador web.

Este framework fue creado con referencias del mundo web y sus historias exitosas. Algunas de ellas son Ruby en Rails o Django en python. Estos frameworks ayudan a JVM en la simplificación del trabajo, pues eliminan los boilerplate necesarios para la creación de un ambiente de desarrollo para el framework.

De hecho si definición de full-stack significa que todo esta hecho para el desarrollador, todo lo que el necesite esta en el framework.

En resumen Play! Es la forma mas rápida de crear sorprendentes aplicaciones que hacen uso de las nuevas características presentadas por la web, no obstante Play! Tiene algunos inconvenientes como, limitada presencia en la comunidad, difícil de aprender, difícil de adoptar, incompatibilidad con versiones anteriores.[15]

No obstante Play! Resuelve un problema fundamental en las aplicaciones basadas en web: la reactividad de un servidor web dentro de un ambiente distribuido.

Además de que en la versión 2 se incluyeron nuevas herramientas para facilitar su uso, entre estas herramientas están:

- Uso de scala como lenguaje principal.
- Uso de templates los cuales son renderizados desde el lado del servidor.
- Los archivos del lado del servidor, son pre compilados por Play! Usando http.
- Uso de HTML5
- Servicios externos.
- Validación de formularios.
- Recarga en caliente.
- Uso solo de dos herramientas el IDE y el navegador web.

### *B. Estado del arte*

Muchas aplicaciones están basadas en Spring Framework y además usan el modulo de seguridad entre estas podemos encontrar: Alfresco, el Catalogo de información del gobierno Chino(GIC, por sus siglas en ingles), un sistema de carga de pagos a entidades publicas.

Alfresco es un manejador de contenidos para empresas. Un manejador de contenidos para empresas como Alfresco visto desde la perspectiva de un desarrollador es una agrupación de herramientas entre la cuales se encuentran[16]:

- Manejador de contenidos: capturar, organizar y compartir archivos que generalmente son generados por software de productividad para oficinas.
- Manejador de contenidos para web: manejo de datos que serán compartidos en web.
- Manejo de activos: manejo de gráficos, video y audio.
- Manejo de historiales: Manejo de contenidos como historiales legales.
- Imágenes: capturar, etiquetar y enrutamiento imágenes de documentos desde scanners.

Desde las perspectivas de los usuarios podrían decir que es menos acerca de tecnologías y mas acerca de cómo capturar, organizar y compartir información en toda la empresa. Para el usuario es mas importante el como que el que. [16]

A grandes rasgos Alfresco es utilizado por empresas para el manejo de sus documentos desde un servidor. Para esto a

Alfresco utiliza tecnologías para seguridad, integración con aplicaciones de autoría, flujos de trabajo, personalización de las interfaces y librerías de servicio.

En años reciente el gobierno Chino a trabajo en el desarrollo del gobierno electrónico especialmente en e campo del desarrollo y utilización de fuentes de información. [17]

La función principal del GIC es recibir metadatos del sistema de catalogación, después el GIC audita y verifica la información y la importa a la base de datos. El GIC publica la

información del catalogo, para que el usuario del catalogo pueda luego consultar el GIC par localizar información deseada.

El GIC esta compuesto por tres subsistemas los cuales son:

- Subsistemas de registro y publicación de metadatos: en grandes rasgos se encarga de controlar, auditar y verificar los metadatos.
- Subsistema de manejo y mantenimiento: Manejo de usuarios, manejo de privilegios de usuario, manejo de log del sistema, configuración del sistema.
- Subsistema del servicio de catálogos: servicio de consultas al catálogo e interfaz del servicio.

La arquitectura del sistema esta basado en el estándar Java. El sistema esta hecho con las mejores practicas y construido como una aplicación web distribuida multinivel con el importante soporte de tecnologías como componentes distribuidos y servicios web.

El framework utilizado para el desarrollo de la aplicación es Appfuse framework. Appfuse es una aplicación Java de código abierto que usa un contenedor ligero(Spring Container).

Por otra parte el sistema de carga de pagos a entidades publicas se realiza de la siguiente manera:

- Se recibe un archivo de texto plano donde se mencionan las tareas realizadas durante el día en el banco.
- Luego este archivo se almacena en una ruta especial del servidor.
- Después se ejecuta una tarea a cierto día y hora.
- Después la aplicación se encargara de realizar la tarea de procesar la información y enviar mensajes informativos de manera asíncrona.
- Después la aplicación procesara la información y la grabara en la base de datos.
- Una vez procesada la información se grabaran en tablas diseñadas especialmente para pagos.

La aplicación fue desarrollada utilizando Spring framework, JMS(java message service), message bróker (apache-active-mq) y un servidor de aplicaciones que en este caso es Oracle web logic. [18]

## III. Resultados

### *A. Metodología*

Actualmente existen diferentes metodologías para el desarrollo de software, desarrollo centrado en reglas de negocio, desarrollo en cascada, metodologías agiles, etc. Estas metodologías pueden ser divididas en tradicionales y metodologías agiles, las metodologías agiles surgieron como respuesta a los problemas que generaban la metodologías tradicionales, ya que en las metodologías tradicionales no se puede trabajar en proyectos en los cuales no se sabe con exactitud los requisitos. Por otro lados las metodologías agiles surgieron como una propuesta para abordar directamente el código y no centrarse tanto en la documentación, estas metodologías son mas bien orientadas a seguir un camino donde la parte importante y primordial de la documentación es el código. [19]

Por esta razón en el desarrollo de este proyecto se utilizara Scrum la cual es una metodología ágil que a continuación será detallada.

En la figura 1 se puede observar un diagrama de cómo se empieza un proyecto con la metodología de Scrum. Todo empieza por el dueño del producto quien además del ser el encargado o dueño del proyecto, también es el responsable de levantar una lista de requerimiento que los usuarios le pedirán, esta lista es la lista de pendientes donde se priorizan las tareas a partir de su valor y tiempo estimado. La lista de pendientes puede incluir desde correcciones de errores hasta nuevas funcionalidades lógicas.

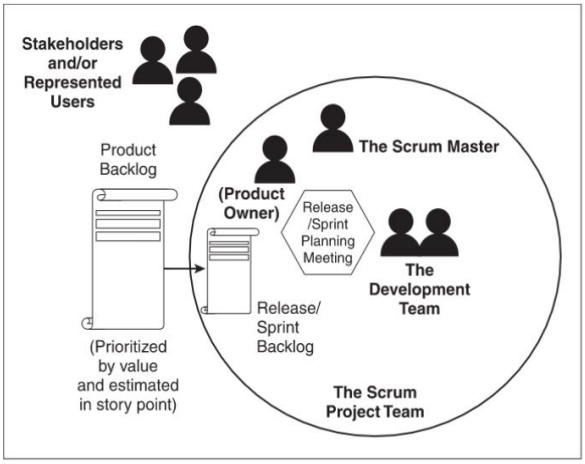


Fig. 1. Esquema esquipo de desarrollo en Scrum

En este proyecto el dueño del producto fue el asesor del proyecto, quien levanto la lista de requerimientos a partir del proyecto ya existente.

Los requerimientos que los usuarios piden deben tener una corta duración que debe ser de uno a dos días ya que estas serán priorizadas para las reuniones de planeación. En las reuniones de planeación todo el equipo del proyecto se reúne para revisar que es lo que se lleva de avance, los detalles que se encontraron y que dirección se tomara en la elección de las nuevas tareas a realizar.[20]

El maestro scrum a diferencia en otras metodologías de desarrollo de software no es un líder de proyecto absoluto que pone reglas y manda, sino un integrante mas del proyecto que se encarga de proteger al equipo de factores externos o internos que afecten el trabajo del equipo. [21]

En Scrum los Sprints son los periodos de tiempo durante los cuales se realizaran las tareas, que fueron seleccionadas para formar parte de la lista de pendientes. En la lista de pendientes se priorizan las tareas del dueños del producto.

Además de las reunionés de planeación, Scrum cuenta con cortas reuniones diarias de 15 a 30 minutos de duración que se llaman, Scrum diario. En estas reuniones el equipo comenta lo que se realizo el día anterior, si encontró algún problema y con que se trabajara ese día. El objetivo es que si algún integrante

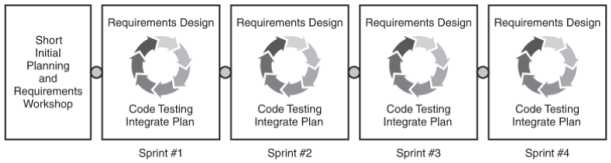


Fig. 2. Esquema flujo de trabajo Scrum

del equipo se encuentra con un detalle que otro integrante ya haya encontrado, entre los dos puedan resolver mas rápidamente el detalle y puedan continuar con el desarrollo.[22]

Durante el desarrollo del proyecto se realizaron los Scrums diarios en los cuales se verificaban que avances se habían obtenido durante el día anterior y los que se esperaban optener.

Por otra parte como se puede ver en la figura 2 el flujo de trabajo en Scrum invierte menos tiempo en entrevistas y reuniones con los usuarios, ya que solo hay un reunión grande donde se planean tareas primordiales. Aunado a esto se aprecia claramente que en cada Sprint se entrega un ejecutable, que previamente fue probado y revisado. [23]

Scrum fue aplicado en el desarrollo del proyecto. Para esto fue necesario el uso de 3 Sprints. A continuación se describe el trabajo que fue realizado en cada Sprint.

Sprint 1, durante este Sprint se creo la nueva estructura para el proyecto, este nueva estructura fue creada usando el gestor de proyectos Java Maven. El siguiente comando fue utilizado para generar la estructura del proyecto: mvn archetype:generate -DgroupId=mx.edu.um.facsa -DartifactId=examen -DarchetypeArtifactId=maven-archetype-webapp -DinteractiveMode=false. En el comando se especifican cuatro parámetros de entrada,

- *En el groupId* se especifica el paquete base donde estarán la clases Java en este caso mx.edu.um.facsa,
- En el artifactId se especifica el nombre que llevara el proyecto para este proyecto se utilizo el nombre de examen.
- En el parámetro archetypeArtifactId se especifica el prototipo de arquitectura que se usara, en este caso se utilizo el prototipo de aplicación web ya que la aplicación será utilizada en la capa web.
- Finalmente se especifica que no usara el modo interactivo.

Al ejecutar el comando en la herramienta de línea de comandos, se presentara información de lo que esta realizando Maven para crear el proyecto hasta que finalmente se imprime una línea donde especifica si la creación del proyecto fue exitosa o errónea. En la figura 3 se puede ver la salida que se presenta en la línea de comandos cuando la generación del proyecto se realizo de manera exitosa, además de esto se

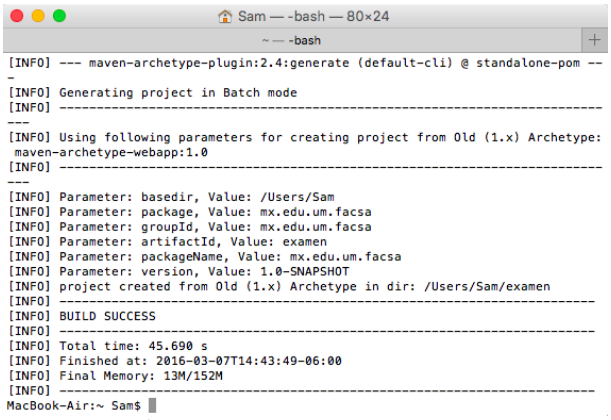


Fig. 3. Generación de proyecto exitosa

presenta otra información como en que directorio fue creado el proyecto, nombre, versión del proyecto, etc.

En la figura 4 se observa la estructura del directorio generado por el comando Maven, como se especifico en el comando crea un directorio con el nombre examen y una serie de subdirectorios que en java representan el paquete principal.

Para facilitar el acceso y edición de archivos se importo el proyecto al editor de código Eclipse.

Para utilizar Spring Framework y Hibernate en el proyecto

```
examen
|-- pom.xml
`-- src
    |-- main
    |   |-- java
    |   |   `-- mx
    |   |       `-- edu
    |   |           `-- um
    |   |               `-- facsa
    |   |                   `-- examen
    |   |-- resources
    |   |-- webapp
    |   |   `-- index.jsp
    |   |   `--WEB.INF
    |   |      `-- web.xml
```

Fig. 4. Estructura del proyecto generada con el comando.

lo primero que se realizo fue incluir las dependencias necesarias en el pom del proyecto. Las dependencias que son indispensables para Spring y Hibernate son, spring-core, spring-web, spring-webmvc, spring-security-web, spring-security-config, hibernate-core, hibernate-validator y hibernate-ehche.

Una vez que las dependencias fueron incluidas como se puede ver en el apéndice 5, se crearon los archivos de configuración. En estos archivos de configuración que son archivos XML se especifica a que base de datos se conecta la aplicación, en que paquetes Java buscar componentes de la aplicación, etc. Para crear un archivo XML en este caso se utilizo el gestor de archivos de Eclipse, para la cual se siguieron los siguientes paso, ir al menú archivo, en el menú desplegable seleccionar la opción de nuevo, en el menú

desplegable que se desplego seleccionar otro y se abrirá la ventana del gestor de archivos. Una vez en el gestor de archivos en el cuadro de búsqueda escribir xml, en la opción de xml seleccionar archivo XML. Por ultimo el gestor de archivos pedirá la especificación de la carpeta donde se guardara el archivo y el nombre del archivo. Para terminar dar clic en terminar.

Para Spring Framework en especifico para el IoC Container todo es un bean por lo tanto estos archivos de configuración son tomados como bean es los cuales se especifican configuraciones.

El primer archivo XML que fue creado es dispatcher-servlet.xml en este archivo de configuración se especifica el directorio donde se encontraran los archivos para las vistas así como el tipo de archivo. Además de mapear los recursos que fueron utilizados en la parte del front-end del proyecto. Como se puede ver el apéndice 1.

En el archivo examen.xml se especificaron las siguientes configuraciones:

- Se definió la versión de Spring Framework a utilizar.
- Se definió la ubicación del archivo de propiedades, en este archivo se declaran propiedades como usuario y contraseña del servidor de correos, usuario y contraseña de base de datos, propiedades en la conexión a la base de datos y propiedades de hibernate.
- Se declaro el paquete principal donde se encontraran alojados los Controllers y se especifico que serán identificados por el uso de las anotaciones de Spring. En este caso para identificar un controller se utilizo la anotación @Controller.
- Se declaro la conexión a la base de datos.
- Se declaro el sesión de conexión a base de datos de hibernate con la conexión de base de datos declarada anteriormente
- Se declaro el servicio para enviar correos
- Se declaro el manejador de transacciones de hibernate utilizando la session de base de datos declarada con anterioridad.
- Se declaro la fuente de mensajes que se utilizaran para el cuerpo y titulo de los correos electrónicos.
- Se declaro el método de encriptación de contraseñas que será utilizado.

En el apéndice 2 se puede la estructura del archivo examen.xml. Con este archivo se declaran las primeras convenciones de Spring Framework, además este archivo es de alta importancia ya que en el se declaro la conexión a base de datos y la interacción de esta con el proyecto.

Una vez que estos archivos fueron creados, se configuro el archivo XML en el que se especifica la seguridad, este archivo se llama security.xml.

En este archivo se especifico lo siguiente

- Primero se especifico la versión de Spring Security a utilizar.
- Se activo el uso de anotaciones para validar que se accedió al método con los roles correctos.
- Se declaro que el acceso a archivos de la capa de interacción con el usuario sea visible para todos.
- Se declararon el gestor de rutas de acceso, este gestor es el encargado de permitir el acceso con los roles determinados a cada una de las pantallas. Por ejemplo para que algún usuario pueda tener acceso a la pantalla de administración es necesario que tenga el rol de administrador, en caso de que no posea tal rol será direccionado a una pantalla que le indicara que no tiene los permisos suficientes para ingresar a tal modulo. Además en el gestor de rutas de acceso se especifican las rutas para acceder a la aplicación y para salir de la aplicación en este caso las rutas son /entrar y /salir respectivamente. En caso de que algún usuario intente acceder al sistema con credenciales invalidas el gestor de rutas enviara un mensaje a la pantalla especificando el tipo de error.
- Se declaro el servicio para los detalles del usuario, con este servicio se obtiene la información necesaria una vez que el acceso al sistema fue satisfactorio.
- Se declaro la jerarquía de roles. La jerarquía quedo dela siguiente manera: role_admin> role_user > role_authenticated > unauthenticated, empezando por el rol con mayor jerarquía hasta llegar al rol del usuario ni siquiera ha sido autenticado.
- Se declaro el manejador de acceso a web, este manejador es el encargado de que las peticiones que se hagan a la aplicación se han interceptadas adecuadamente así como dar una respuesta a la que se tiene la debida autorización.
- Se declaro el manejador de expresiones de seguridad en la capa web, este manejador es el encargado de permitir el uso de anotaciones de seguridad en la capa web esto es con la intención de solo permitir el acceso al contenido al cual se tenga autorización.

Este archivo es de vital importancia ya que en el se centra la mayor parte de la configuración necesaria para implementar Spring Security. De este archivo de configuración se espera que pueda delegar correctamente cada una de las funciones declaradas, así como dar la respuesta adecuada al usuario dependiendo de su nivel autorización.

En el apéndice 4 se puede ver la configuración creada en el archivo security.xml.

Una vez que esta configuración fue realizada, fue necesario especificar estos archivos de configuración en el archivo web.xml del proyecto. El archivo web.xml se define toda la configuración del proyecto que necesita el servidor de aplicaciones. En el apéndice se puede apreciar la versión final del archivo web.xml.

En la segunda fase del proyecto se tomaron las funcionalidades que ya existían y parte del código existente.

En esta fase se dividieron las clases java lógicamente en los siguientes paquetes:

- Model: clases java que representan un objeto.
- Dao: clases que contienen métodos de conexión a base de datos.
- Service: clases que contiene métodos con lógica del negocio.
- Controller: clases que dirigen a comunicación entre el usuario y proyecto.

Para evitar la reutilización de código se crearon clases java con métodos genéricos como crear, editar, borrar. Estas clases fueron utilizadas con herencia.

### *B. Resultados.*

Se realizo una reestructuración en general del proyecto y de la capa de seguridad, logrando un organización mas apropiada. Ya que los aspectos de seguridad fueron especificados en el nivel apropiado dentro de la implementación.

La implementación de la seguridad fue posible mediante el uso de filtros y listeners en la capa web. Gracias a esta implementación fue posible determinar a que rutas puede tener acceso el usuario dependiendo de su nivel de autorización.

Además se creo la funcionalidad de roles por usuario que anteriormente no existía, ya que previamente el acceso a las pantalla se manejaba por usuario, no por roles. Por otra parte el manejo de los datos del usuario en la sesión fue implementado mediante una clase componente para evitar la reutilización de código además de evitar riesgos de seguridad al manejar manualmente los datos de usuario. Por otra parte se implementaron clases abstractas que contienen métodos que reciben por parámetros objetos genéricos, estos métodos son, crear, actualiza, eliminar, buscar, listar. Estos métodos son básicos en cualquier catalogo. Con estas clases se evito la repetitividad de código puesto que ya contienen lo métodos básicos no es necesario reescribirlos sino solo instanciarlos.

### *C. Discusión.*

Los resultados muestran que es posible la implementación de aspectos de seguridad usando Spring Framework 3 a un proyecto existente.

El trabajo realizado en este documento muestra que los objetivos definidos anteriormente fueron obtenidos, ya que las funcionalidades y el código existente fueron reutilizados implementando los nuevos aspectos de seguridad.

Sin embargo es necesario implementar convenciones de desarrollo en el proyecto, para que el código que sea agregado en futuros desarrollos utilice e implemente correctamente los aspectos de seguridad.

## IV. Conclusiones

Utilizando Spring Framework 3 es posible implementar los aspectos de seguridad a un proyecto existente, siempre y cuando el proyecto este basado en Java y que las clases estén divididas de manera lógica, en este caso que las clases estén de acuerdo al modelo MVC. Confirmando así nuestra hipótesis inicial.

Como trabajo futuro se plantea el desarrollo de nuevas funcionalidades utilizando el modelo implementado. Además de que a las nuevas funcionalidades agregadas se especifiquen los roles de acceso mediante el uso de anotaciones de Spring Framework 3.

## V. Apéndices

### *A. Apéndice 1*

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:mvc="http://www.springframework.org/schema/mvc"

xmlns:security="http://www.springframework.org/schema/security"

xmlns:context="http://www.springframework.org/schema/context"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
    http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.1.xsd
    http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc-3.1.xsd
    http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-
3.1.xsd
    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-
3.1.xsd
    ">
  <context:annotation-config/>
  <context:component-scan base-
package="mx.edu.um.facsa.examen"/>
  <mvc:annotation-driven />
  <mvc:resources mapping="/css/**" location="/css/" />
  <mvc:resources mapping="/images/**" location="/images/" />
  <mvc:resources mapping="/img/**" location="/img/" />
  <mvc:resources mapping="/js/**" location="/js/" />
  <mvc:resources mapping="/fonts/**" location="/fonts/" />
  <mvc:interceptors>
    <ref bean="openSessionInViewInterceptor"/>
  </mvc:interceptors>
  <bean id="openSessionInViewInterceptor"
class="org.springframework.orm.hibernate4.support.OpenSessionInV
iewInterceptor">
    <property name="sessionFactory">
      <ref local="sessionFactory"/>
    </property>
  </bean>
  <bean id="viewResolver"
class="org.springframework.web.servlet.view.InternalResourceView
Resolver">
    <property name="cache" value="false" />
    <property name="viewClass"
value="org.springframework.web.servlet.view.JstlView" />
    <property name="prefix" value="/WEB-INF/jsp/" />
    <property name="suffix" value=".jsp" />
  </bean>
  <bean id="messageSource"
class="org.springframework.context.support.ResourceBundleMessag
```

```
eSource">
      <property name="basename" value="messages"/>
   </bean>
   <bean id="multipartResolver" class=
"org.springframework.web.multipart.commons.CommonsMultipartR
esolver" >
      <property name="maxUploadSize" value="1024000" />
   </bean>
</beans>
```

### *B. Apéndice 2*

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:context="http://www.springframework.org/schema/context"
      xmlns:tx="http://www.springframework.org/schema/tx"
      xsi:schemaLocation="
    http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.1.xsd
    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-
3.1.xsd
    http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-3.1.xsd
    ">
      <context:property-placeholder
      location="classpath*:ambiente/examen.properties,file:${us
er.home}/examen*.properties" />
      <context:component-scan base-
package="mx.edu.um.facsa.examen">
            <context:exclude-filter type="annotation"
      expression="org.springframework.stereotype.Controller" />
      </context:component-scan>
      <bean id="dataSource"
class="org.apache.commons.dbcp.BasicDataSource">
            <property name="driverClassName"
value="${jdbc.driverClassName}" />
            <property name="url" value="${jdbc.url}" />
            <property name="username"
value="${jdbc.username}" />
            <property name="password"
value="${jdbc.password}" />
            <property name="initialSize" value="5" />
            <property name="maxActive" value="10" />
      </bean>
      <bean id="sessionFactory"
      class="org.springframework.orm.hibernate4.LocalSessionF
actoryBean">
            <property name="dataSource" ref="dataSource"
/>
            <property name="packagesToScan"
value="mx.edu.um.facsa.examen.**.model" />
            <property name="hibernateProperties">
                  <props>
                        <prop
key="hibernate.show_sql">${hibernate.show_sql}</prop>
                        <prop
key="hibernate.hbm2ddl.auto">${hibernate.hbm2ddl.auto}</prop>
                        <prop
key="hibernate.cache.use_second_level_cache">${hibernate.cache.u
se_second_level_cache}</prop>
                        <prop
key="hibernate.cache.provider_class">${hibernate.cache.provider_cl
ass}</prop>
                  </props>
            </property>
      </bean>
      <bean id="mailSender"
class="org.springframework.mail.javamail.JavaMailSenderImpl">
            <property name="host"
value="${mail.smtp.host}" />
            <property name="port"
value="${mail.smtp.port}" />
            <property name="username"
value="${mail.smtp.user}" />
            <property name="password"
value="${mail.smtp.password}" />
            <property name="javaMailProperties">
                  <props>
                        <prop
key="mail.debug">${mail.debug}</prop>
                        <prop
key="mail.smtp.auth">${mail.smtp.auth}</prop>
                        <prop
key="mail.smtp.starttls.enable">${mail.smtp.starttls.enable}</prop>
                        <prop
key="mail.smtp.socketFactory.class">${mail.smtp.socketFactory.cla
ss}</prop>
                  </props>
            </property>
      </bean>
      <bean
class="org.springframework.orm.hibernate4.HibernateExceptionTran
slator" />
      <bean id="transactionManager"
      class="org.springframework.orm.hibernate4.HibernateTran
sactionManager">
            <property name="sessionFactory"
ref="sessionFactory" />
      </bean>
      <tx:annotation-driven />
      <bean id="messageSource"
      class="org.springframework.context.support.ResourceBund
leMessageSource">
            <property name="basename" value="messages"
/>
      </bean>
      <bean id="passwordEncoder"
      class="org.springframework.security.authentication.encodi
ng.ShaPasswordEncoder" />
</beans>
```

### *C. Apendice 3*

```
<?xml version="1.0" encoding="UTF-8"?>

<beans:beans
xmlns="http://www.springframework.org/schema/security"
      xmlns:beans="http://www.springframework.org/schema/be
ans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/sch
ema/beans http://www.springframework.org/schema/beans/spring-
beans-3.1.xsd
            http://www.springframework.org/schema/security
```

```
http://www.springframework.org/schema/security/spring-security-3.1.xsd">
	<global-method-security pre-post-annotations="enabled"
		secured-annotations="enabled">
		<expression-handler
ref="methodSecurityExpressionHandler" />
	</global-method-security>
	<http pattern="/css/**" security="none" />
	<http pattern="/images/**" security="none" />
	<http pattern="/img/**" security="none" />
	<http pattern="/js/**" security="none" />
	<http pattern="/fonts/**" security="none" />
	<http pattern="/login.jsp*" security="none" />
	<http auto-config="true" access-decision-manager-
ref="webAccessDecisionManager"
		use-expressions="true">
		<intercept-url pattern="/inicializa/**"
access="permitAll" />
		<intercept-url pattern="/articulo**"
access="hasAnyRole('ROLE_ADMIN')" />
		<intercept-url pattern="/admin/**"
access="hasAnyRole('ROLE_ADMIN')" />
		<intercept-url pattern="/**"
access="isAuthenticated()" />
		<form-login login-page="/login.jsp" login-
processing-url="/entrar"
			authentication-failure-
url="/login.jsp?error=1"
			authentication-success-handler-
ref="loginHandler" />
		<openid-login authentication-failure-
url="/login.jsp?error=1"
			authentication-success-handler-
ref="loginHandler">
			<attribute-exchange>
				<openid-attribute
name="email"
	type="http://axschema.org/contact/email" required="true"
/>
			</attribute-exchange>
		</openid-login>
		<logout logout-url="/salir" logout-success-url="/"
			invalidate-session="true" />
	</http>
	<beans:bean name="userDetailsService"

	class="mx.edu.um.facsa.examen.general.service.UserDetail
sServiceImpl" />
	<authentication-manager alias="authenticationManager">
		<authentication-provider user-service-
ref="userDetailsService">
			<password-encoder
ref="passwordEncoder">
				<salt-source user-
property="username" />
			</password-encoder>
		</authentication-provider>
	</authentication-manager>
	<beans:bean id="roleHierarchy"
	class="org.springframework.security.access.hierarchicalrol
es.RoleHierarchyImpl">
		<beans:property name="hierarchy">
			<beans:value>
				ROLE_ADMIN >
ROLE_USER
				ROLE_USER >
ROLE_AUTHENTICATED
				ROLE_AUTHENTICATED
> ROLE_UNAUTHENTICATED
			</beans:value>
		</beans:property>
	</beans:bean>
	<beans:bean id="roleHierarchyVoter"
	class="org.springframework.security.access.vote.RoleHiera
rchyVoter">
		<beans:constructor-arg ref="roleHierarchy" />
	</beans:bean>
	<!-- security:authorize tags using the url attribute will
delegate to this
		accessDecisionManager -->
	<beans:bean id="webAccessDecisionManager"
	class="org.springframework.security.access.vote.Affirmati
veBased">
		<beans:property name="decisionVoters">
			<beans:list>
				<beans:bean
	class="org.springframework.security.web.access.expression
.WebExpressionVoter">
					<beans:property
name="expressionHandler" ref="webSecurityExpressionHandler" />
				</beans:bean>
			</beans:list>
		</beans:property>
	</beans:bean>
	<beans:bean id="webSecurityExpressionHandler"
	class="org.springframework.security.web.access.expression
.DefaultWebSecurityExpressionHandler">
		<beans:property name="roleHierarchy"
ref="roleHierarchy" />
	</beans:bean>
	<beans:bean id="methodSecurityExpressionHandler"
	class="org.springframework.security.access.expression.met
hod.DefaultMethodSecurityExpressionHandler">
		<beans:property name="roleHierarchy"
ref="roleHierarchy" />
	</beans:bean>
</beans:beans>
```

### *D. Apendice 4*

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app
  version="3.0"
  xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">
  <display-name>examen</display-name>
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
      classpath:examen.xml
      classpath:security.xml
    </param-value>
  </context-param>
  <filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-
```

```
class>org.springframework.web.filter.DelegatingFilterProxy</filter-
class>
  </filter>
  <filter>
    <filter-name>sitemesh</filter-name>
    <filter-
class>org.sitemesh.config.ConfigurableSiteMeshFilter</filter-class>
  </filter>
  <filter>
    <filter-name>encodingFilter</filter-name>
    <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filte
r-class>
    <init-param>
      <param-name>encoding</param-name>
      <param-value>UTF-8</param-value>
    </init-param>
    <init-param>
      <param-name>forceEncoding</param-name>
      <param-value>true</param-value>
    </init-param>
  </filter>
  <filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <filter-mapping>
    <filter-name>sitemesh</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <filter-mapping>
    <filter-name>encodingFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <listener>
    <listener-
class>org.springframework.web.context.ContextLoaderListener</list
ener-class>
  </listener>
  <servlet>
    <servlet-name>dispatcher</servlet-name>
    <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-
class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>classpath:dispatcher-servlet.xml</param-
value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>dispatcher</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>
  <session-config>
    <session-timeout>30</session-timeout>
  </session-config>
  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

*E. Apéndice 5*

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/
4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>mx.edu.um.facsa</groupId>
    <artifactId>examen</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>war</packaging>
    <name>examen</name>
    <properties>
    <endorsed.dir>${project.build.directory}/endorsed</
endorsed.dir>
        <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.jboss.spec</groupId>
            <artifactId>jboss-javaee-web-
6.0</artifactId>
            <version>2.0.0.Final</version>
            <type>pom</type>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>commons-
logging</groupId>
            <artifactId>commons-
logging</artifactId>
            <version>1.1.1</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>jcl-over-
slf4j</artifactId>
            <version>1.6.0</version>
        </dependency>
        <dependency>

    <groupId>ch.qos.logback</groupId>
            <artifactId>logback-
classic</artifactId>
            <version>1.0.1</version>
        </dependency>
        <dependency>

    <groupId>org.springframework</groupId>
            <artifactId>spring-aop</artifactId>

    <version>3.1.2.RELEASE</version>
        </dependency>
        <dependency>

    <groupId>org.springframework</groupId>
```

```
                        <artifactId>spring-orm</artifactId>
                        <version>3.1.1.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-webmvc</artifactId>
                        <version>3.1.1.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-openid</artifactId>
                        <version>3.1.1.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-web</artifactId>
                        <version>3.1.1.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-config</artifactId>
                        <version>3.1.1.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-taglibs</artifactId>
                        <version>3.1.1.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-test</artifactId>
                        <version>3.1.1.RELEASE</version>
                        <scope>test</scope>
                </dependency>
                <dependency>
                        <groupId>org.hibernate</groupId>
                        <artifactId>hibernate-core</artifactId>
                        <version>4.1.2</version>
                </dependency>
                <dependency>
                        <groupId>org.hibernate</groupId>
                        <artifactId>hibernate-ehcache</artifactId>
                        <version>4.1.2</version>
                </dependency>
                <dependency>
                        <groupId>org.hibernate</groupId>
                        <artifactId>hibernate-validator</artifactId>
                        <version>4.2.0.Final</version>
                </dependency>
                <dependency>
                        <groupId>junit</groupId>
                        <artifactId>junit</artifactId>
                        <version>4.10</version>
                        <scope>test</scope>
                </dependency>
                <dependency>
                        <groupId>commons-dbcp</groupId>
                        <artifactId>commons-dbcp</artifactId>
                        <version>1.4</version>
                </dependency>
                <dependency>
                        <groupId>commons-pool</groupId>
                        <artifactId>commons-pool</artifactId>
                        <version>1.6</version>
                </dependency>
                <dependency>
                        <groupId>mysql</groupId>
                        <artifactId>mysql-connector-java</artifactId>
                        <version>5.1.19</version>
                        <scope>runtime</scope>
                </dependency>
                <dependency>
                        <groupId>postgresql</groupId>
                        <artifactId>postgresql</artifactId>
                        <version>9.1-901-1.jdbc4</version>
                        <scope>runtime</scope>
                </dependency>
                <dependency>
                        <groupId>javax.servlet</groupId>
                        <artifactId>jstl</artifactId>
                        <version>1.2</version>
                </dependency>
                <dependency>
                        <groupId>taglibs</groupId>
                        <artifactId>standard</artifactId>
                        <version>1.1.2</version>
                </dependency>
                <dependency>
                        <groupId>commons-fileupload</groupId>
                        <artifactId>commons-fileupload</artifactId>
                        <version>1.2.2</version>
                </dependency>
                <dependency>
                        <groupId>commons-io</groupId>
                        <artifactId>commons-io</artifactId>
```

```
                        <version>2.2</version>
                </dependency>
                <dependency>
                        <groupId>cglib</groupId>
                        <artifactId>cglib</artifactId>
                        <version>2.2.2</version>
                </dependency>
                <dependency>
                        <groupId>org.sitemesh</groupId>
                        <artifactId>sitemesh</artifactId>
                        <version>3.0-alpha-2</version>
                </dependency>
                <dependency>
                        <groupId>commons-
lang</groupId>
                        <artifactId>commons-
lang</artifactId>
                        <version>2.6</version>
                </dependency>
                <dependency>
                <groupId>org.codehaus.jackson</groupId>
                        <artifactId>jackson-mapper-
asl</artifactId>
                        <version>1.9.6</version>
                </dependency>
                <dependency>
                        <groupId>joda-time</groupId>
                        <artifactId>joda-time</artifactId>
                        <version>2.1</version>
                </dependency>
                <dependency>
                        <groupId>javax.mail</groupId>
                        <artifactId>mail</artifactId>
                        <version>1.4.5</version>
                </dependency>
        </dependencies>
</project>
```

## VI. REFERENCIAS.


[1] T. M. Harrison and T. Stephen, *Computer Networking and Scholarly Communication in the Twenty-First-Century University: An Introduction to Isaac Breuer's Philosophy of Judaism*. State University of New York Press, 1996.

[2] C. G. King, R. W. Guyette Jr, and C. Piotrowski, "Online Exams and Cheating: An Empirical Analysis of Business Students' Views," *J. Educ. Online*, vol. 6, no. 1, p. n1, 2009.

[3] Y. Levy and M. M. Ramim, "A study of online exams procrastination using data analytics techniques," *Interdiscip. J. E-Learning Learn. Objects*, vol. 8, no. 1, pp. 97–113, 2012.

[4] J. Turner, *MySQL and JSP Web applications*. Sams publishing, 2002.

[5] Q. Mahmoud, "Servlets and JSP Pages Best Practices," *Sun Dev. Netw.*, 2003.

[6] D. V. P. Colque C. Ricardo, "- INTEGRACIÓN DE TECNOLOGÍAS EN UNA PLATAFORMA J2EE DIRIGIDA POR MODELOSJF - Ingeniare. Revista Chilena de Ingeniería," vol. - 265–275, 2006.

[7] J. Grossman, "Whitehat website security statistics report," *Retrieved March*, vol. 8, p. 2010, 2007.

[8] F. Balaguer and U. of I. at Urbana-Champaign, *Using Frameworks to Extend Frameworks*. University of Illinois at Urbana-Champaign, 2008.

[9] J. Arthur and S. Azadegan, "Spring framework for rapid open source J2EE Web application development: a case study," in *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2005 and First ACIS International Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005. Sixth International Conference on*, 2005, pp. 90–95.

[10] R. Johnson, J. Höller, A. Arendsen, T. Risberg, and C. Sampaleanu, *Professional Java Development with the Spring Framework*. Wiley, 2005.

[11] P. Jagielski and J. Nabrdalik, *Instant Spring Security Starter*. Packt Publishing Ltd, 2013.

[12] D. Newton, *Apache Struts 2 Web Application Development : Design, Develop, Test, and Deploy Your Web Applications Using the Struts 2 Framework*. Birmingham, U.K.: Packt Publishing, 2009.

[13] N. Pandey and N. (Corporation), *Microsoft ASP.NET: Fast & Easy Web Development*. Premier Press, 2002.

[14] A. Chakraborti, U. Kranti, R. J. Sandhu, and Niit, *NET Framework : Professional Projects*. [Rocklin, Calif.]: Course PTR, 2002.

[15] A. Petrella, *Learning Play! Framework 2 : Start Developing Awesome Web Applications with This Friendly, Practical Guide to the Play! Framework*. Birmingham, UK: Packt Publishing, 2013.

[16] J. Potts, *Alfresco developer guide*. Packt Publishing Ltd, 2008.

[17] W. Hu, W. Yiding, and Y. Xu, "RESEARCH AND IMPLEMEN TATION OF CATALOG SYSTEM ON THE CONSTRUCTION OF THE GOVERNMENT INFORMATION CATALOG ARCHITECTURE."

[18] F. M. Santos López, "- Diseño de un módulo de carga de pagos en entidades públicas mediante mensajería con spring frameworkJF - Industrial Data," vol. - 73–79, 2012.

[19] E. Delgado Expósito, "- Metodologías de desarrollo de software. ¿Cuál es el camino?JF - Revista de Arquitectura e Ingeniería," 2008.

[20] H. F. Cervone, "Understanding agile project management methods using Scrum," *OCLC Syst. Serv. Int. Digit. Libr. Perspect.*, vol. 27, no. 1, pp. 18–22, 2011.

[21] H. Kniberg, "Scrum y XP desde las trincheras," *C4Media Inc. InfoQ*, 2007.

[22] N. B. Moe and T. Dingsøyr, "Scrum and team effectiveness: Theory and practice," in *Agile Processes in Software Engineering and Extreme Programming*, Springer, 2008, pp. 11–20.

[23] L. Rising and N. S. Janoff, "The Scrum software development process for small teams," *IEEE Softw.*, vol. 17, no. 4, p. 26, 2000.