Trabajo Fin de Grado
Ingeniería de Telecomunicación

Comunicación entre Herramientas basadas en LTI y Sistemas de Gestión de Aprendizaje

Autor: Álvaro Martín Rodríguez
Tutor: Antonio Jesús Sierra Collado

Dpto. Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2015

Trabajo Fin de Grado
Ingeniería de Telecomunicación

Comunicación entre Herramientas basadas en LTI y Sistemas de Gestión de Aprendizaje

Autor:
Álvaro Martín Rodríguez

Tutor:
Antonio Jesús Sierra Collado
Profesor titular

Dpto. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2015

Proyecto Fin de Grado: Comunicación entre Herramientas basadas en LTI y Sistemas de Gestión de Aprendizaje

Autor: Álvaro Martín Rodríguez

Tutor: Antonio Jesús Sierra Collado

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2015

El Secretario del Tribunal

A mi familia

A mis amigos

A mis maestros

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

BLOQUE 1. INTRODUCCIÓN

Objetivos

El objetivo final de este proyecto es lograr la integración de una herramienta externa, a una plataforma de gestión de aprendizaje, mediante el protocolo LTI [1], para ampliar las capacidades de esta última mediante dicha integración.

Esta amplicación de capacidades, viene a significar que siempre que podamos hacer una herramienta que lleve a cabo nuestro propósito, o satisfaga nuestra necesidad, y si la plataforma de gestión de aprendizaje (o por su siglas en inglés, LMS [2]) implementa igualmente el protocolo LTI, podremos incrustar dicha herramienta para su acceso dentro de dicho LMS. De esta manera, unas operaciones que no están al alcance de la tecnología web común, o bien unas operaciones específicas que no son comunes y por tanto no se encuentran implementadas en los LMS, acaban pudiéndose llevar a cabo.

En nuestro caso, desarrollaremos una herramienta que busca la automatización del proceso de corrección de los trabajos de programación: Práctica Final de Estructuras Dinámicas (PFED) y Práctica Final de Programación Orientada a Objetos (PFPOO) de la asignatura Fundamentos de Programación II (FP2) de 1º Grado en Ingeniería de las Tecnologías de Telecomunicación (GITT). Para ello, se propone el desarrollo de una herramienta web, que permita ampliar la capacidad de la entrega de trabajos de programación llevados a cabo por alumnos. Así, en lugar de ser únicamente entregados en formato digital al profesorado a través de la plataforma Blackboard de Enseñanza Virtual (EV) [3] de la que dispone la Universidad de Sevilla (US) [4], puedan también ser corregidos y evaluados de manera automática al realizar dicha entrega.

Esta herramienta, será accedida tanto por los alumnos como por los profesores, desde la propia Enseñanza Virtual, presentando distintas interfaces según el rol de la persona que acceda en cada caso, bien alumno bien profesor, con funciones distintas en cada caso.

Expresado el caso, resulta claro a los ojos del lector que un LMS (siendo BlackBoard Learn [5] la plataforma en la que se basa la Enseñanza Virtual) no tiene la capacidad de corregir un trabajo de un alumno de manera automática, ni mucho menos es viable que conozca como debe corregirse o evaluarse un trabajo concreto de una asignatura de una universidad u organización que emplee dicha plataforma. Esto es lo que nosotros buscamos: dotar de nuevas capacidades a nuestro sistema mediante la integración de una nueva herramienta, conectada mediante LTI.

Alcance

Este proyecto va a ser dividido en 6 bloques, donde se realizará respectivamente:

1. Una introducción al mismo y definición de su alcance, en este primer bloque.
2. En el segundo bloque, se realizará un estudio de la situación actual, analizando especialmente la tecnología LTI junto con otras posibles alternativas, algunos componentes necesarios de los que depende dicha tecnología, y un breve repaso del escenario donde se desea implantar la herramienta.
3. En tercer lugar, se pasará a explicar la aproximación elegida para desarrollar la herramienta, detallando la arquitectura diseñada y los componentes que la configuran, así como su implementación e interfaz final disponible para los usuarios.
4. A continuación, expondremos las pruebas de ejecución y funcionamiento llevadas a cabo, para asegurar el correcto funcionamiento de la herramienta implementada.
5. En el quinto bloque, expondremos las conclusiones de este proyecto, detallaremos algunas líneas

futuras viables e incluiremos la bibliografía.

6. Como sexto y último bloque, se encontrarán los diferentes anexos, donde se incluirán desde el código fuente completo de la herramienta, hasta diferentes manuales o scripts desarrollados para su despliegue.

BLOQUE 2. SITUACIÓN ACTUAL

Introducción

El objetivo final de este proyecto, como se ha comentado anteriormente, es lograr la automatización del proceso de corrección de los trabajos de programación: Práctica Final de Estructuras Dinámicas (PFED) [6] [7] y Práctica Final de Programación Orientada a Objetos (PFPOO) de la asignatura FP2.

Para abordar esta tarea, analizaremos la situación actual, extrayendo las distintas fases, componentes y actores implicados. Nos centraremos primero en LTI, para posteriormente dar un repaso a como funciona la corrección de los trabajos actualmente.

Antes de continuar, el lector querrá saber, ¿por qué LTI? ¿Existen otras opciones alternativas? ¿Cuál es su rendimiento?

Existen otras especificaciones o sistemas que permiten llevar a cabo la integración de herramientas externas en plataformas de aprendizaje, como son *Apache Wookie* [8] o *GLUE!* [9]. Cada uno de ellos, LTI incluido, presenta una arquitectura diferente que le confiere diferentes capacidades, y si el lector está interesado, puede consultar el artículo *Comparison of the main alternatives to the integration of external tools in different platforms* [10], donde se realiza una comparativa de los tres sistemas aquí mencionados. Como conlusión de este artículo, encontramos que el autor cree que LTI será el estándar de facto debido al gran apoyo que tiene, presentando Wookie una posible tendencia de mejora debido a su arquitectura basada en *widgets*.

Aunque nosotros nos centraremos directamente en LTI, dado que este proyecto se presentó con este objetivo, llevaremos a cabo primero una breve comparativa, para dotar al lector de una visión general de las distintas aproximaciones existentes a la hora buscar la integración de herramientas en los LMS.

Alternativas de Integración de Heramientas en LMS

Existen diferentes interfaces de comunicación, formatos en el intercambio de datos o incluso mecanismos de seguridad entre las diferentes herramientas. Esto provoca una dificultad añadida a la hora de intentar realizar una integración estandarizada de estas herramientas en las plataformas de enseñanza. Así, con el objetivo de mermar el costo de integraión de estas herramientas, distintas organizaciones han promovido sus propias aproximaciones, resultado las ya mecionadas anteriormente: GLUE!, Apache Wookie y LTI en dos variantes, a saber: IMS LTI y Basic LTI.

La diferencia entre estas dos versiones de LTI es que IMS LTI, también llamada comúnmente Full LTI, se encuentra basada en la integración de servicios web, mientras que Basic LTI permite una integración menos profunda de aplicaciones web, siendo la primera evolución de la segunda. Por otro lado, Apache Wookie busca mediante su arquitectura particular, ser capaz de permitir la integración de W3C Widgets [11] y herramientas que cumplan la especificación OpenSocial [12], que son aplicaciones web empaquetadas, mientras que GLUE! busca permitir la integración flexible de herramientas desarrolladas con múltiples tecnologías.

Veamos finalmente el esquema seguido en la arquitectura de cada uno de ellos, y en qué están basados estos esquemas alternativos a LTI.

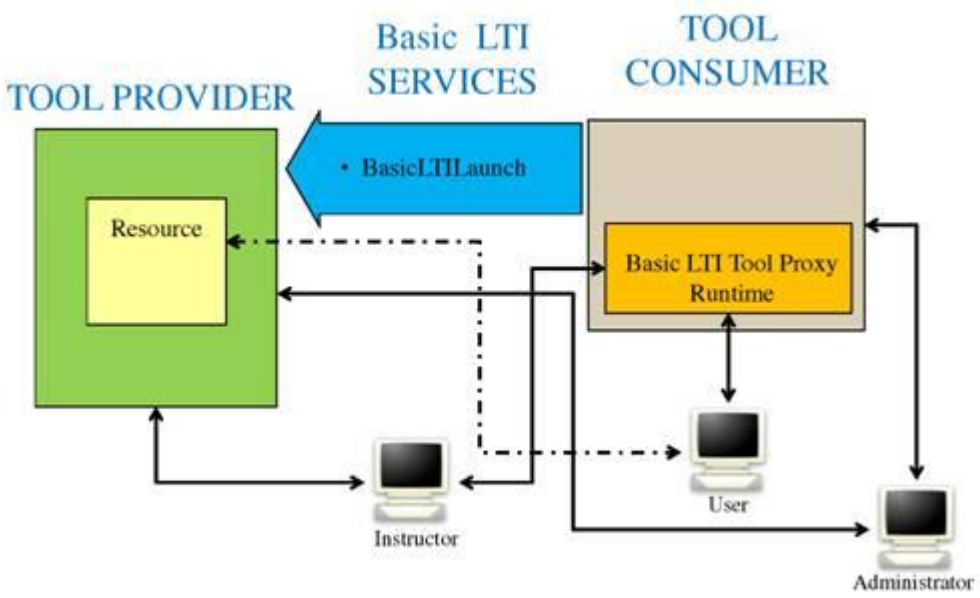

Figura 2.1. Arquitectura de Basic LTI

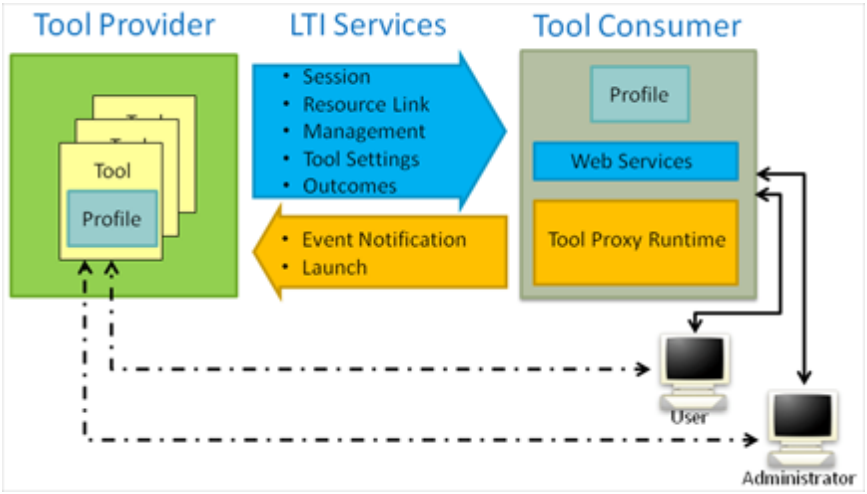

Figura 2.2 Arquitectura de Full LTI

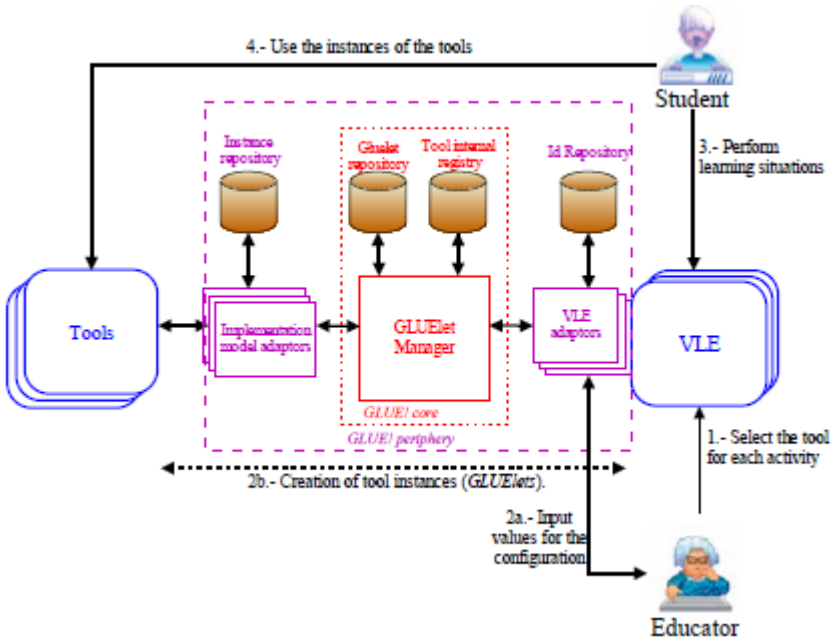

Figura 2.3. Arquitectura de GLUE!

GLUE! por su parte, está basado en REST [13], y busca una integración flexible empleando las interfaces que proporcionan las herramientas y los LMS. Esto provoca que para cada elemento que desee emplear GLUE! sea necesario elaborar un adaptador para ello.

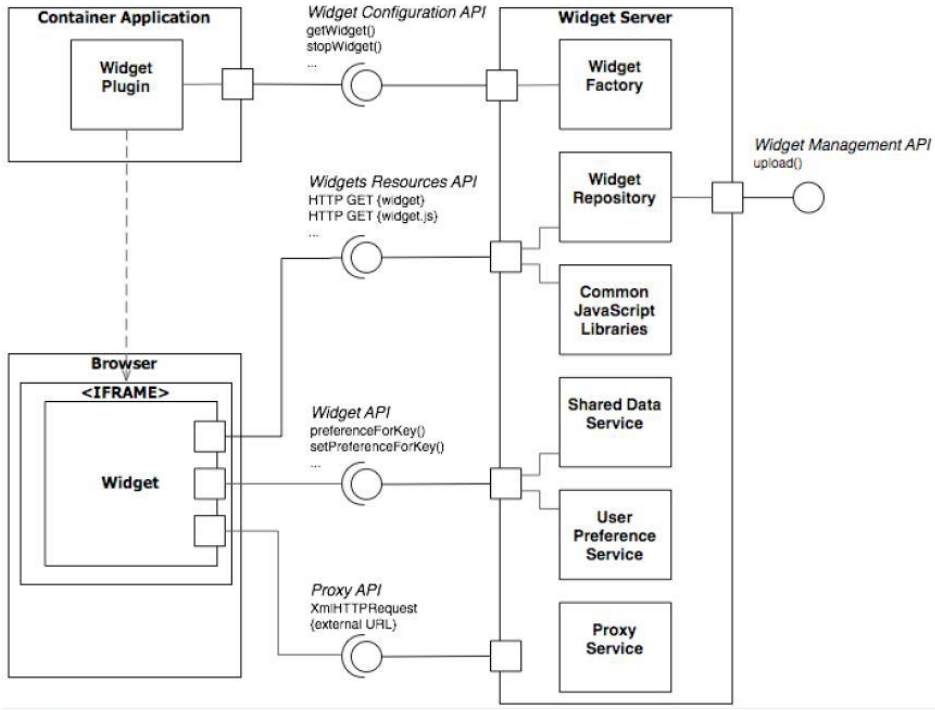

Figura 2.4. Arquitectura de Apache Wookie.

Finalmente, Apache Wookie permite el uso mediante REST de distintos widgets, permitiendo instanciarlos, controlar dichas instancias, instalar y desplegar nuevos widgets. Además, permite la gestión de una memoria compartida entre todos estos widgets, siendo esta memoria de tipo persistente. Además, el servidor Wookie puede ser ampliado con características adicionales en función de los requisitos de cada widget.

LTI

2.2.1 Introducción

Hasta ahora, la corrección de los trabajos o prácticas finales de la asignatura que nos ocupa, gozaban de un grado de automatización aceptable que facilitaba su corrección de manera sistemática al profesorado. Siendo esto así, ¿qué podemos aportar? ¿Qué línea de acción nos interesa?

En 2010, propuesto por el *IMS Global Learning Consortium* [14], LTI [15] nace con el objetivo de permitir el uso de herramientas remotas, así como otros contenidos de distinta índole, dentro de un LMS. Esta capacidad, se presenta como el primer paso de un protocolo en expansión, ya que este no es su motivo ulterior, sino que pretende ampliar sus capacidades con el paso de sus diferentes versiones.

2.2.2 Motivación

Inicialmente, esto puede parecer algo con poco fundamento, ya que se podría enlazar con hipervínculos normales a estos contenidos: nada más lejos de la realidad.

Para comprender adecuadamente las ventajas de LTI, necesitamos pararnos a analizar el escenario que acabamos de describir, al que sumaremos la terminología empleada en las recomendaciones oficiales de LTI.

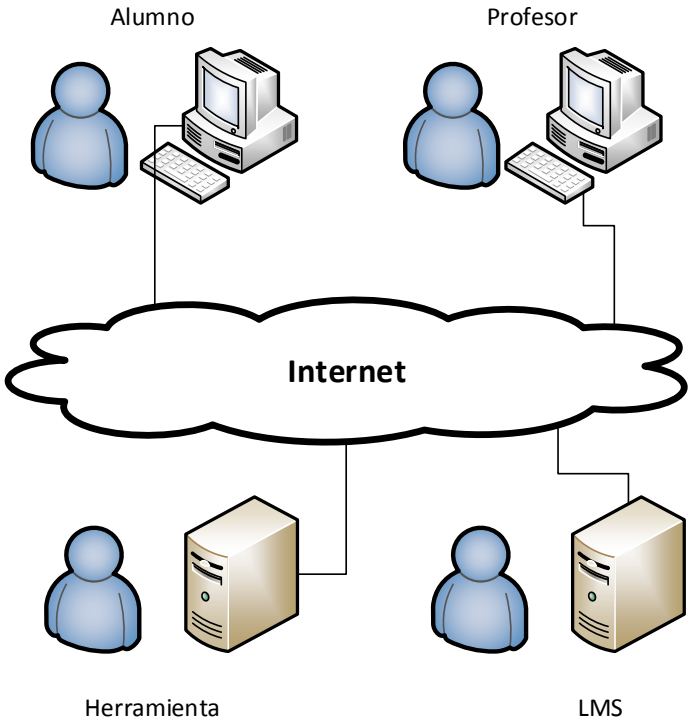

Figura 2.5. Actores y escenario LTI.

Como podemos ver en la Figura 2.6, nos encontramos ante cuatro actores principales, donde la mitad de ellos son personas (alumno y profesor), y otros dos software (herramienta externa y LMS):

- Alumno: Dispone de un equipo (PC) con el que puede acceder tanto al LMS como a la herramienta que el profesor desea que sea utilizada por los alumnos.
- Profesor: Dispone de un equipo (PC) con el que puede realizar lo mismo que un alumno, pero además tiene la capacidad de modificar los cursos bajo su tutela en el LMS.
- Herramienta: programada o no por un tercer agente. Puede acceder o poseer sus propios recursos (bases de datos, ficheros, etcétera).
- LMS: Alojado y gestionado en este caso, por la Universidad de Sevilla.

Todos los actores se encuentran conectados a Internet, de manera que los actores humanos pueden acceder sin problema a ambos actores software e interactuar con ellos.

Sin embargo, al encontrarnos en un ambiente académico, debemos tener en cuenta que nuestro LMS maneja información relevante tanto para el alumno como para el profesor: es capaz de diferenciar usuarios, sus roles dentro de la universidad, sus identificadores, sus cursos, etc. Esto no ocurre al acceder a una herramienta externa, que en nada está relacionada (o no tiene por qué estarlo) con nuestro LMS. No importa que la herramienta sea un simple contador de personas, una aplicación Java o cualquier otra cosa: el usuario accede a ella de manera anónima, como a cualquier otra página web, y como mucho puede implementar un sistema de autenticación propio, en nada relacionado con la universidad.

Recordemos que al principio hemos dicho que el objetivo con el que nace LTI, es el de permitir el uso de herramientas remotas desde un LMS. Es decir, el usuario podría acceder a la herramienta mediante un enlace que encontraría en su curso correspondiente dentro del LMS, desde ahí accedería a ella. De esta manera, siendo el LMS de alguna manera el iniciador de la comunicación entre el usuario y la herramienta, podría transmitir cierta información referente al usuario a esta última, permitiéndole comportarse de diferentes maneras según su identificador, rol, etc. Nótese que este identificador mencionado no tiene por qué ser el identificador del usuario en el LMS, sino que puede ser un identificador único generado por el mismo para las herramientas LTI, de manera que pueda identificar al usuario de manera única sin exponer su identidad fuera del sistema universitario.

Por tanto esto nos permitiría:

- Adecuación del comportamiento de la herramienta a cada usuario, rol, etc.
- Capacidad para diferenciar distintos usuarios en la herramienta, de manera única.
- Aumento de las capacidades del LMS, que pasa a poder emplear herramientas externas como si estuviesen integradas en él.

Este último punto, es la clave de la motivación de este proyecto, ya que mediante el uso de una herramienta LTI, se puede automatizar por completo la entrega y corrección de trabajos de los alumnos, cosa que queda fuera de las posibilidades de un LMS.

2.2.3 Terminología

A la hora de describir el funcionamiento de LTI, es conveniente que empleemos la nomenclatura reconocida por el *IMS Global Learning Consortium*. Así, podemos definir:

- **Tools** (Herramientas): son las aplicaciones o contenido que deseamos emplear dentro de nuestro entorno de aprendizaje.
- **Tool Provider (TP)**: es el sistema donde se aloja la herramienta que se desea utilizar. Salvo que sea necesario diferenciar entre ambos conceptos, se emplea *Tool Provider* para representar a la herramienta de manera indistinta.
- **Tool Consumer (TC)**: es el LMS o plataforma que consume la aplicación o contenido.
- **Contexto**: aquel curso/asignatura/etcétera perteneciente al LMS, bajo el que se agrupan contenidos, enlaces, etc.
- **Basic LTI**: referencia a LTI versión 1.0.
- **Basic LTI (Link)**: Se refiere a una de las dos maneras de integración de LTI. Se basa en que únicamente se expone un destino dentro del TP mediante el enlace, admitiendo una política de seguridad y sin capacidad para acceder a servicios en ejecución en el TC.
- **Full LTI**: Se refiere a la segunda manera de integración de LTI. Implica un proceso formal, negociado donde el TC y el TP llegan a un acuerdo sobre los servicios en tiempo de ejecución que se utilizarán, las políticas de seguridad empleadas (pueden ser más de una), y el conjunto de destinos que pueden ser accedidos desde el TC.
- **Common Cartridge (CC)**: conjunto de estándares del IMS Global Learning Consortium, que busca proporcionan una manera estándar de representar los materiales en los cursos digitales para su uso en sistemas de aprendizaje en línea.

- **Cartridge**: unidad básica de CC, que busca describir un material de un curso digital. Aquí se empleará para describir los enlaces LTI dentro de un curso.

2.2.4 Versiones de LTI

Si bien hasta ahora hemos remarcado la capacidad básica y fundamental de LTI para lanzar herramientas externas desde un TC, esto es propio de *LTI 1.0*, también llamado *Basic LTI* [16]. Sin embargo, en las siguientes versiones de LTI, se han ido incorporando distintas funcionalidades que complementan y amplían las capacidades de interacción entre TP y TC. Detallemos estas capacidades:

Característica	LTI 1.0	LTI 1.1	LTI 1.2	LTI 2.0	Comentarios
Lanzamiento básico	✓	✓	✓	✓	Desde LTI 2.0 se reducen mucho la necesidad de datos "opcionales" en cada lanzamiento.
Salida		✓	✓	✓	Devolver un valor numérico simple como resultado de la actividad.
Perfil de TC			✓	✓	Son metadatos que describen atributos y servicios disponibles en el TC. Se consigue mediante un servicio REST.
Tool Proxy				✓	Son metadatos que describen la negociación llevada a cabo entre un TC y un TP.
Gestión de Credenciales				✓	Intercambio automático y seguro de pares clave/secreto.
Flujo de registro				✓	El administrador del LMS inicia la inserción de herramientas, incluyendo crear TP y la gestión de credenciales.
Documentación basada en modelos			✓	✓	Documentación exhaustiva de referencia generada mediante herramientas y UML.
Servicios REST			✓	✓	Servicios REST de nivel 3 para gran variedad de tareas servidor a servidor. En LTI 1.2 solo se permiten servicios REST implementados en el TC.

Tabla 2.1. Características de las versiones existentes de LTI. [17]

Podemos observar, que LTI ha sufrido una fuerte evolución, motivada en la mejora de las capacidades de interactuación entre *Tool Provider* y *Tool Consumer*.

Como ya comentamos anteriormente, en la Enseñanza Virtual a fecha de comienzo de este proyecto, se dispone de un sistema *Blackboard Learn 9.1 SP8*. Este LMS cuenta con implementación *Basic LTI* [18], por lo que nuestra herramienta no podrá interactuar con el LMS más allá de su lanzamiento inicial.

2.2.5 Casos de Uso

Definamos por tanto los casos de uso de *Basic LTI*:

Título	**Configuración de Credenciales de Dominio de un TP**
ID	LTIv1-12
Descripción	El administrador del TC configura las credenciales de un dominio de TP
Actores	• Administrador del TC • Administrador del TP
Requisitos Previos	Ninguno
Flujo de Eventos	1. El administrador del TP crea una pareja de clave y secreto para el TC (el administrador del TC puede solicitar una clave concreta, normalmente el nombre de dominio del TC). 2. El administrador del TC recibe la pareja clave/secreto del administrador del TP. 3. El administrador del TC instala la pareja de valores y los asocia con los lanzamientos de los vínculos con destino al dominio del TP, empleando un mecanismo de configuración proporcionado por el TC.
Alternativas	A. En el paso 3, la combinación de clave/secreto puede ser aplicada en exclusiva a una URL destinada al TP, en lugar de a todas las URL con destino al dominio de dicho TP.

Tabla 2.2. Caso de uso de *Basic LTI* C1.

Título	**Establecer las Credenciales de un Enlace**
ID	LTIv1-13
Descripción	Un profesor crea un enlace LTI, y establece la clave/contraseña para dicho enlace.
Actores	• Profesor • Tool Provider (TP)
Requisitos Previos	Ninguno
Flujo de Eventos	1. El profesor contacta con TP y obtiene acceso a una herramienta de dicho TP o contenido. 2. El TP proporciona al profesor una URL o fragmento de código XML, para el lanzamiento de la aplicación LTI (o contenido), junto con la clave necesaria para acceder a él, y el secreto asociado con dicha clave. 3. El profesor introduce la terna de valores en el TC para crear el enlace Basic LTI.

Tabla 2.3. Caso de uso de *Basic LTI* C2.

Título	**Abriendo un Enlace Autorizado de Basic LTI desde un Contexto**
ID	LTIv1-14
Descripción	Un usuario no profesor, selecciona el enlace LTI dentro de un contexto en el TC.
Actores	• Usuario del TC
Requisitos Previos	El profesor ha configurado adecuadamente o importado un enlace Basic LTI, y existen credenciales apropiadas para su uso.
Flujo de Eventos	1. El usuario del TC pulsa sobre el enlace en la interfaz de usuario del TC. 2. La herramienta o contenido del TP aparece dentro de la interfaz del TC. Si JavaScript está deshabilitado, el usuario del TC tendrá que pulsar sobre un botón de "Continuar", para enviar una petición POST al TP.

Tabla 2.4. Caso de uso de *Basic LTI* C3.

Título	**Abriendo un Enlace Basic LTI Importado desde un Cartridge (con secreto)**
ID	LTIv1-15
Descripción	Un profesor importa un Common Cartridge que contiene descriptores de un enlace Basic LTI en su contexto, y los usuarios acceden al contenido.
Actores	• Creador del Cartridge • Profesor • Usuario del TC
Requisitos Previos	El administrador del TC ha recibido e instalado las credenciales necesarias para el TP al que se hace referencia en el Cartridge.
Flujo de Eventos	1. Se crea un cartridge mediante el Cartridge Creator, incluyendo uno o varios descriptores de enlaces Basic LTI. El descriptor de un enlace Basic LTI en el cartridge, contiene una URL para acceder, así como otros datos, pero no contiene claves ni secretos. 2. El profesor obtiene el Common Cartridge y lo importa en el contexto del TC. 3. Cuando un usuario del TC accede a un enlace Basic LTI importado mediante un Common Cartridge, el TC usa las credenciales preconfiguradas, que se encuentran asociadas con el dominio del TP o la URL destino. 4. En concreto, mientras que existan instaladas credenciales en el TC, el profesor no necesita realizar ninguna otra acción para lanzar el contenido, más allá de importar el cartridge.
Alternativas	A. Si los requisitos previos no se cumplen, también es posible establecer las credenciales después de importar el cartridge. Si un acceso al enlace se ha llevado a cabo antes de establecer las credenciales, será responsabilidad del TP notificar al usuario que las credenciales son necesarias.

Tabla 2.5. Caso de uso de *Basic LTI* C4.

Título	**Abriendo un Enlace Basic LTI Importado desde un Cartridge (sin secreto)**
ID	LTIv1-16
Descripción	Un profesor importa un Common Cartridge que continee descriptores de enlaces Basic LTI en su contexto, y los usuarios acceden al contenido. Este escenario es opcional, el TC y/o el TP deben decidir si los enlaces Basic LTI que son accedidos sin secretos son tratados como error o no.
Actores	• Cartridge Creator • Profesor • Usuario de TC
Requisitos Previos	Ninguno
Flujo de Eventos	1. Mediante Cartridge Creator se crea un cartridge que incluye uno o más descriptores de enlaces Basic LTI. El descriptor de un enlace Basic LTI en el cartridge, contiene una URL para acceder, así como otros datos, pero no contiene claves ni secretos. 2. El profesor obtiene el Common Cartridge y lo importa en el contexto del TC. 3. Cuando un usuario de TC accede al enlace de Basic LTI de un Common Cartridge, el TC lanza el enlace LTI sin autentificación ni información de firma.

Tabla 2.6. Caso de uso de *Basic LTI* C5.

2.2.6 Arquitectura

La arquitectura de funcionamiento de LTI 1.0 [19], puede esquematizarse como puede verse en la siguiente Figura:

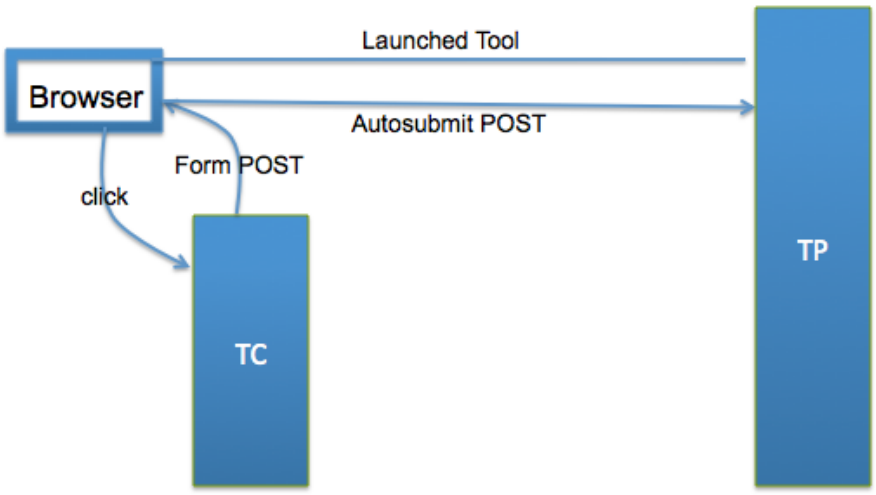

Figura 2.6. Acceso a un enlace LTI 1.0.

1. Un usuario del TC, por ejemplo un estudiante, pulsa sobre el enlace LTI, de manera que realiza una petición de lanzamiento a la herramienta LTI del TP.

2. El LMS (TC), prepara un "paquete" de lanzamiento de la aplicación, incluyendo parámetros necesarios estandarizados, así como aquellos parámetros opcionales o personalizados que estén configurados para ser enviados a la herramienta.
3. OAuth firma digitalmente los datos, para asegurar su integridad.
4. El TC envía el mensaje de vuelta al navegador del usuario, junto con un fragmento de código JavaScript para autogenerar un mensaje POST hacia el TP.
5. EL TP responde, representándose la herramienta en el navegador del usuario.

El fragmento de código JavaScript representa lo que es verdaderamente fundamental en este procedimiento, ya que asegura que la fuente del acceso a la herramienta (TP) es el usuario mediante su navegador, en lugar del TC.

Puede no quedar claro con este esquema, en una primera lectura, cómo se está enviando el mensaje. Mostremos otra Figura que ilustra una analogía entre el escenario real y los términos empleados en LTI.

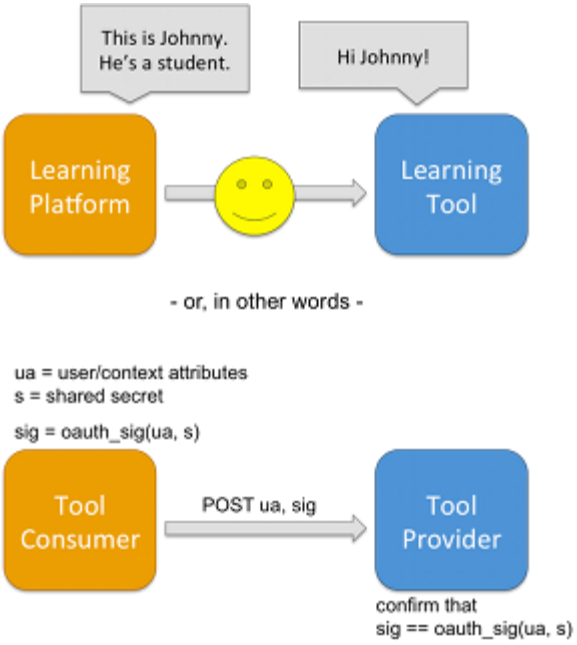

Figura 2.7. Analogía Mundo Real y LTI [20]

Como se puede observar mucho mejor en este esquema, con OAuth lo que queremos asegurar es la integridad del mensaje, que viaja sin cifrar en todo momento. Es este mensaje, junto con la firma generada del mismo mediante el secreto compartido, lo que recibe el TP.

El secreto no viaja en el mensaje, de manera que un tercer agente no podría modificar el mensaje y generar su propia firma, ya que lo único que viaja es como parte del mensaje la clave asociada al TC, que es un identificativo para que el TP sepa que secreto se ha utilizado, y por tanto lo emplee a la hora de comprobar la firma.

Este mensaje, estará compuesto de una serie de parámetros, donde deberemos atender a distintas categorías en función de la obligatoriedad de su uso en la petición POST. Enunciaremos aquí los más importantes, es decir, aquellos requeridos, y expondremos algunos de las demás categorías, pudiéndose encontrar todos ellos en la especificación de LTI 1.0 ya referenciada anteriormente.

- Parámetros requeridos
    - *Lti_message_type:* identifica el tipo de mensaje LTI. En *Basic LTI* debe tener asignado el valor *basic-lti-launch-request*.
    - *Lti_version:* identifica la versión de LTI empleada, que en *Basic LTI* será *LTI-1p0*.
    - *Resource_link_id:* identifica el enlace empleado dentro del TC para acceder a la herramienta

LTI. Si una misma herramienta tiene varios enlaces a la misma herramienta, cada uno de ellos tendrá un valor distinto de este parámetro.

- Parámetros recomendados
    - *Roles:* identifica como su nombre indica, el rol del usuario. Un alumno sería "Learner" y un profesor "Instructor", aunque existen muchos otros.
    - *User_id:* es un identificador único que referencia al usuario. Queda en manos del Tool Provider decidir qué valor se transmite en este campo cuando se use, y se recomienda que sea opaco para no desvelar información fuera de su sistema.
    - *Lis_person_name_full:* es el nombre completo del usuario que accede a la herramienta.
    - *Context_title:* es el título del contexto desde donde se accede a la herramienta. Es decir, en el caso de la Enseñanza Virtual, la asignatura o curso donde se encuentra el enlace a la herramienta.
- Parámetros opcionales
    - *Resource_link_description*: es una descripción del enlace que lanzó la aplicación.
    - *Custom_**: donde "*" es un comodín, indica que pueden configurar más parámetros opcionales, pero que deben estar definidos en el TC para poder ser empleados.

En conjunto, son toda una serie de parámetros existentes, más aquellos personalizados que se deseen crear, que nos permiten enriquecer la interacción entre TC y TP.

2.2.7 Credenciales en LTI

Hemos visto que, a la hora de conectar un TC con un TP, se emplea una pareja de valores clave/secreto para realizar la comunicación, basado en OAuth. Llamaremos por tanto a cada pareja de estos valores, **credenciales**. Además, si recordamos los casos de uso expresados anteriormente, también hemos hablado sobre como se pueden asociar estos valores a nuestra comunicación LTI, a saber:

1. Asociación de unas credenciales al dominio de un Tool Provider completo. En este caso, es el adminitrador del TC el que debe realizar la operación en el LMS, de manera que todos los accesos a dicho dominio quedan vinculados a estas credenciales.
2. Asociación de unas credenciales a una URL concreta de un Tool Provider. De nuevo es llevado a cabo por el administrador del TC, de manera que dichas credenciales son empleadas por todos los accesos a dicha URL desde el Tool Consumer.
3. Asociación de unas credenciales a una URL concreta. En este caso, esta tarea es llevada a cabo por el profesor que configura el enlace dentro de su curso.

Para un enlace LTI configurado en el Tool Consumer, puede ocurrir que haya más de una pareja de credenciales válidas configuradas, ya que las hay tanto generales (dominios) como específicas (URL). En estos casos, el Tool Consumer empleará aquella que tenga una mayor importancia para firmar la petición a la herramienta. Queda ambiguo este punto en la especificación, al no indicar si una mayor importancia significa aquellas establecidas por un usuario de mayor importancia (adminitrador sería mayor que profesor), o bien especfífico frente a general (una URL sería más concreto que un dominio).

Sin embargo, se debe mencionar que es posible que no existan credenciales asociadas al enlace LTI al que se intenta acceder. En este caso, la implementación del Tool Consumer debe decidir si desea mandar peticiones sin firmar, y de la misma forma el Tool Provider puede decidir sobre este mismo punto, es decir, si desea recibir peticiones sin firmar o no. Ambos pueden elegir su comportamiento en su implementación, permitiéndolo o tratando esta situación como un error.

2.2.8 Implementaciones

A la hora de trabajar con Basic LTI, nos veremos en la necesidad de, bien implementar el protocolo para nuestro sistema, bien encontrar una implementación sobre la que apoyarnos. Por ello, listaremos aquí aquellas

implementaciones existentes más famosas para LTI, así como sus características fundamentales y otros factores de interés en las decisiones de uso que deberemos tomar más tarde.

- **BLTI-Sandwich** [21]: donde BLTI son las siglas de *Basic LTI*, se trata de una librería implementada en Java, que permite ser usada tanto para el TC como para el TP. Desarrollada por George Kroner, el proyecto se encuentra paralizado a fecha de elaboración de este proyecto, una vez quedó implementado *Basic LTI* totalmente. Su compilación es independiente, generando un fichero JAR que puede ser importado en nuestro proyecto.
- **LTI Utilities** [22]: se trata de otra librería creada en Java por *Paul Gray*, empleando el framework Spring. Se encuentra bajo el repositorio oficial del IMS Global Learning Consortium en GitHub. Implementando *Basic LTI* actualmente, figura en los repositorios como en desarrollo actualmente. Está pensada para ser importada en un proyecto en curso, que esté siendo manipulado con Maven, de manera que se compila la librería junto con el proyecto final que desea utilizarla.
- **LtiLibrary** [23]: actualmente en su versión 1.5, se encuentra programada para .NET. Su autor es *Andy Miller*, y en esta versión encuentra implementado LTI 1.2, si bien la librería continúa en desarrollo.
- **IMS LTI** [24]: creada por *Instructure* [25], se trata de una librería escrita en Ruby, se encuentra publicada en GitHub. Implementa el protocolo LTI 1.1, y puede ser empleada tanto para TP como para TC.

Finalizaremos este apartado con una tabla comparativa de estas implementaciones, incluyendo los datos fundamentales a la hora de elegir una de ellas para el desarrollo del proyecto.

Librería	LTI	Lenguaje	Licencia
BLTI-Sandwich	1.0	Java	BSD License [26]
LTI Utilities	1.0	Java	Apache License [27]
LtiLibrary	1.2	.NET	Apache License
IMS LTI	1.1	Ruby	MIT License [28]

Tabla 2.7. Implementaciones de LTI

Donde al escribir, para la librería LtiLibrary .NET como lenguaje, en realidad son dos. Esto es así, ya que los lenguajes que funcionan sobre la plataforma NET Framework de Microsoft [29] son compatibles entre sí, siendo posible emplear una librería escrita en cualquiera de ellos, en los demás. Actualmente, estos lenguajes son **C#** y **Visual Basic .NET**.

2.2.9 OAuth

Por ultimo, hablaremos OAuth, ya que LTI presenta una dependencia functional con ello para funcionar. Nos centraremos para esta tarea en la especificación de OAuth 1.0 [30].

Al igual que pasa con LTI, en OAuth existen tres grandes actores, a saber:

- Usuario
- *Consumer*
- *Provider*

Donde el *Consumer* es la aplicación o software que invoca al servicio ofrecido por el *Provider*, bajo la orden del usuario.

Cuando el dicho procedimiento quiere llevarse a cabo:

1. *Provider* da una pareja de clave (llamada habitualmente *token*) y secreto.
2. Empleando estos valores, el *Consumer* firma los contenidos a partir de ese momento, enviando junto con el contenido firmado el *token* al *Provider*.

De esta manera, este puede saber que secreto ha sido empleado, sin necesidad de que este viaje por la red.

Como podemos comprobar, este mecanismo es exactamente el que sigue LTI, al basar la integridad de los mensajes en OAuth. En caso de desear asegurar la confidencialidad de la comunicación, haría falta emplear otros mecanismos como por ejemplo *HTTPS*.

Existen otra serie de pasos posibles a la hora de emplear OAuth, en el que el *token* para una petición se cambia por un *token* de acceso. Sin embargo, OAuth es un Framework, por lo que LTI solo lo emplea para los pasos descritos anteriormente, sin llegar a implementar todas las funcionalidades que ofrece [31].

Procedimientos de Entrega y Evaluación

A continuación, haremos un breve repaso de cómo se corrigen y evalúan los trabajos entregados por los alumnos en la asignatura FP2.

2.3.1 PFED

- El alumno entrega mediante la Plataforma de Enseñanza Virtual el código realizado, comprimido en formato zip, siguiendo las indicaciones que el profesorado especifica en dicha plataforma previamente a la entrega.
- La práctica queda almacenada en la EV, donde permanece inalterada hasta que llega la fecha de la defensa.
- En la fecha de la defensa, los alumnos son convocados a un examen presencial en la Escuela Técnica Superior de Ingeniería (ETSI) [32] de la Universidad de Sevilla. En ella, se les proponen una serie de ejercicios a realizar sobre la práctica previamente entregada, supuesto que esta funciona correctamente. Así, el alumno superará el examen siempre que se superen:
    - La primera prueba del examen, consistente en el correcto funcionamiento de práctica previamente entregada, junto con la superación de las pruebas de gestión de memoria. Estas pruebas consisten en que el alumno, al programar su código con reservas dinámicas de memoria en lenguaje C (lo cual es un requisito en dicha práctica), haya liberado toda la memoria reservada con anterioridad.
    - Una de las pruebas restantes del examen presencial, consistentes todas ellas en obtener la salida esperada en el programa del alumno, al cambiar: el fichero de entrada que se le proporciona al alumno para su programa, el fichero de configuración empleado por el mismo, y haber realizado una modificación en el comportamiento del programa según requiera el enunciado de la prueba en cuestión. En total, hay tres pruebas de este tipo en la actualidad para este examen.

De esta manera, en función del número de pruebas adicionales (de modificación) que haya sido capaz de completar el alumno con éxito, además de la prueba fundamental de funcionamiento y memoria, será asignada una nota al examen del alumno.

Podemos asignar una fórmula para dicha calificación, que nos será útil posteriormente a la hora de calificar a los alumnos con nuestra herramienta:

Sea A_1 la calificación de primera prueba o prueba fundamental, con valor 1 si ha sido superada, y 0 en caso contrario.

Sean $\{A_2, \dots, A_n\}$ las calificaciones desde la prueba dos hasta la n-ésima prueba del examen, con valor 1 en caso de haber sido superada, y 0 en caso contrario.

$$PFED = 10 \cdot \frac{A_1 \cdot (1 + \sum_{i=2}^{n} A_i)}{n}$$

Fórmula 2.1. *PFED.*

De esta manera, en el escenario actual con 4 pruebas, si el alumno supera con éxito la prueba fundamental y una de las pruebas adicionales, su calificación será:

$$PFED = 10 \cdot \frac{A_1 \cdot (1 + \sum_{i=2}^{n} A_i)}{n} = 10 \cdot \frac{1 \cdot (1 + 1)}{4} = 10 \cdot \frac{2}{4} = 5$$

Con lo que se cumple que el alumno queda aprobado con superar exclusivamente una de las pruebas adicionales. Además, todas de las pruebas tienen una ponderación equitativa igual a 2'5 puntos del examen.

2.3.2 PFPOO

- El alumno entrega mediante la Plataforma de Enseñanza Virtual el código realizado, comprimido en formato zip, siguiendo las indicaciones que el profesorado especifica en dicha plataforma previamente a la entrega.
- La práctica queda almacenada en la EV, donde permanece inalterada hasta que llega la fecha de la defensa.
- En la fecha de la defensa, los alumnos son convocados a un examen presencial en la Escuela Técnica Superior de Ingeniería de la Universidad de Sevilla. En ella, se les proponen una serie de ejercicios a realizar sobre la práctica previamente entregada, supuesto que esta funciona correctamente. Así, el alumno superará el examen siempre que se superen:
    - Una de las pruebas del examen presencial, consistentes todas ellas en obtener la salida esperada en el programa del alumno, al cambiar: el fichero de entrada que se le proporciona al alumno para su programa, el fichero de configuración empleado por el mismo (si lo hubiere), y haber realizado una modificación en el comportamiento del programa según requiera el enunciado de la prueba en cuestión. En total, hay dos pruebas de este tipo en la actualidad para este examen.

Empleando de nuevo la notación declarada para la calificación de la PFED, podemos definir en este caso:

$$PFPOO = 10 \cdot \frac{\sum_{i=1}^{n} A_i}{n}$$

Fórmula 2.2. *PFPOO.*

De esta manera, en el escenario actual con 2 pruebas, si el alumno supera con éxito una de las pruebas como se requería, su calificación será:

$$PFPOO = 10 \cdot \frac{\sum_{i=1}^{n} A_i}{n} = 10 \cdot \frac{1}{2} = 5$$

Con lo que se cumple que el alumno queda aprobado con superar una de las pruebas del examen. Además, todas de las pruebas tienen una ponderación equitativa igual a 5 puntos del examen en este caso.

Procedimientos de Corrección

Aquí analizaremos el mecanismo de corrección empleado por el profesorado para corregir la práctica entregada por los alumnos en el examen realizado.

A la hora de realizar la corrección de las prácticas entregadas por los alumnos, el profesorado cuenta con una serie de mecanismos de su propia elaboración para la comprobación de resultados en cada uno de los trabajos y diferentes apartados. De esta manera, el docente que corrige el trabajo de un alumno solo tiene que ejecutar sobre la práctica entregada por el mismo estas herramientas, y comprobar los resultados obtenidos en la misma para asignar una nota al alumno.

Además, el alumno dispone de dichas herramientas previamente, de manera que no necesita entregar a ciegas su práctica, sino que puede verificar si el comportamiento de la misma es el esperado antes de hacerlo.

2.4.1 Herramientas Empleadas

Como hemos mencionado, existen diferentes herramientas empleadas con distintos objetivos en cada práctica y según qué aspecto están centradas en corregir. Así, podemos encontrar:

- **Herramienta de Corrección de Memoria**. Se encuentra programada principalmente en C, de manera que podemos decir que es una herramienta compilada. Está diseñada para corregir la gestión de memoria de la PFED, para lo que realiza lo siguiente:
    - Duplica el código fuente entregado por el alumno.
    - Busca y reemplaza los segmentos de código fuente de gestión de memoria originales (operaciones de reserva y liberación), por unos segmentos más complejos pensados para, además de realizar estas operaciones que el alumno realizaba originalmente, llevar la cuenta del número de bytes de memoria que se encuentran reservados por el programa.
    - Compila el código fuente generado tras dichas modificaciones.
    - Ejecuta el programa generado, proporcionándole un fichero de entrada y un fichero de configuración.
    - El programa modificado genera un informe de la memoria que permanece reservada justo antes de ser finalizado.
    - Si la memoria reservada es nula (cero), el alumno ha realizado correctamente la gestión de memoria dinámica en su práctica.
    - Por último, se comprueba si la salida normal que genera el programa, en base al código escrito por el alumno, el fichero de entrada y el fichero de configuración proporcionados, coincide con la salida esperada.
    - Si ambas pruebas son correctas, la herramienta devuelve que la prueba ha sido superada.

  Esta herramienta fue desarrollada por los miembros del departamento de Ingeniería Telemática que imparten la asignatura [33] [34]. Además, la herramienta se compone también de unos ficheros makefile, mediante los que se compila y corrige el código de otros apartados.

- **Ficheros makefile**. Se trata de ficheros que compilan el código del alumno conforme a unas reglas, y ejecutan algunos comandos específicos adicionales. Existe uno que se ejecuta independientemente de los demás, por cada prueba de modificación que existe en la PFED. Realiza las siguientes operaciones:
    - Compila el código del alumno, que se encontrará separado del código de otros apartados, ya que las modificaciones realizadas sobre el mismo son distintas.
    - Ejecuta el programa resultante, proporcionándole los ficheros de entrada y configuración pertinentes.
    - La salida del programa se direcciona a un fichero concreto, nombrado de manera pertinentemente coherente con los ficheros de entrada y configuración.
    - Se compara la salida generada por el programa con la salida esperada del mismo, que también se encuentra almacenada en un fichero.
    - En caso de que ambos ficheros sean idénticos, el código del alumno ha superado la prueba con éxito.
- **Herramienta de Corrección basada en Java**. Programada en Java por un alumno en cursos anteriores como proyecto final de grado [35], muestra una pequeña interfaz mediante la que el alumno podía, en el siguiente orden, realizar las acciones:
    - Carga del trabajo al equipo servidor del profesorado. Cada apartado se carga por separado.
    - Realización de las pruebas, ejecutándose unos ficheros *makefile* en el equipo servidor. Cada prueba se ejecuta automáticamente al cargar el apartado correspondiente.
    - Se recibe en la aplicación el resultado de la ejecución y se le muestra al alumno.

Además, la herramienta era capaz de diferenciar la comprobación de funcionamiento de la práctica del alumno de manera previa al examen, de la entrega final realizada por el alumno durante el examen.

2.4.2 Uso de las Herramientas

A la hora de ser empleadas por el profesorado, estas herramientas requieren de una rigurosa metodología de uso. Esto es así, ya que el elevado número de alumnos (que supera las tres centenas normalmente), junto con la posibilidad de realizar varias entregas a lo largo del periodo de la asignatura de estas prácticas, más el número de apartados que contienen, pueden provocar que los ficheros se sobre escriban unos con otros, estropeando los resultados y el trabajo del alumno, o incluso la propia corrección, si no se manejan adecuadamente.

Para realizar este procedimiento, el profesorado emplea una estructura de directorios basada en los identificadores UVUS de los alumnos, de manera que todo el procedimiento de corrección para cada alumno se realiza dentro de su propia carpeta en el ordenador del profesor correspondiente. Este consiste en:

- Creación de la carpeta correspondiente al alumno, con nombre igual a su UVUS.
- Descarga del código entregado por el alumno en formato zip, siendo almacenado en la carpeta antes creada.
- Descompresión del fichero descargado.
- Para cada apartado a corregir:
    - Copia de la herramienta de corrección original al subdirectorio del apartado correspondiente a corregir.
    - Ejecución de la herramienta.
    - Comprobación del resultado y anotación del mismo.

La copia de la herramienta original a cada apartado correspondiente, pese a que el alumno pudiera tenerla ya incluida al haber comprobado el resultado antes de entregar el apartado en cuestión, se encuentra necesaria debido a que se debe asegurar que la herramienta no ha sufrido modificaciones para devolver un resultado que no es el esperado por el profesorado.

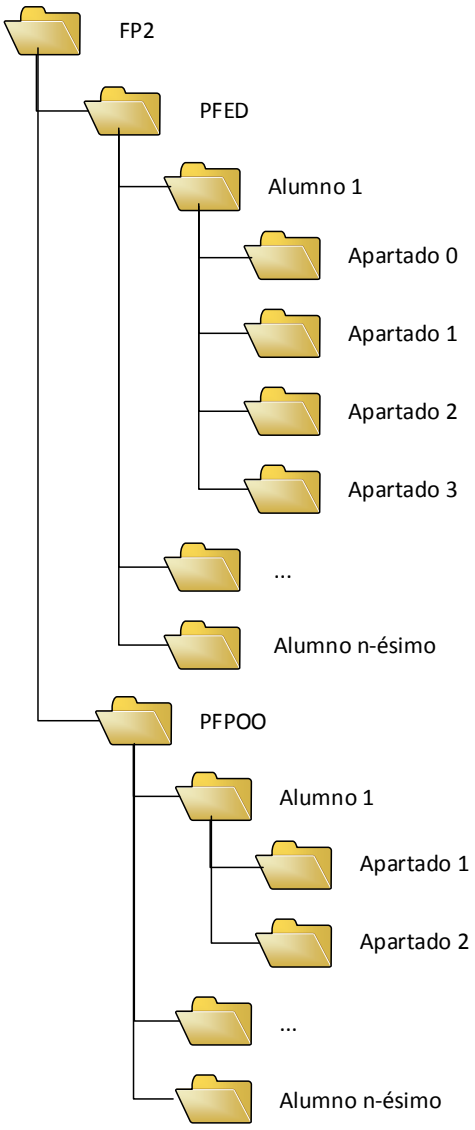

Figura 2.8. Estructura actual de directorios en la corrección

De esta manera, la estructura final de directorios que queda reflejada en el equipo del profesorado encargado de corregir la prueba es el reflejado en la Figura 2.1.

Cabe destacar, como puede ser apreciado en el esquema, que mientras la PFED comienza su numeración en el apartado número cero, la PFPOO comienza en el número 1. Esto es así por convenio del profesorado dentro de la asignatura, dado el significado antes comentado de la prueba fundamental necesaria para aprobar la PFED.

Plataforma de Enseñanza Virtual

Por último y antes de pasar a la descripción de nuestra solución, abordaremos el tema de la Plataforma de Enseñanza Virtual empleada en la US.

A la hora de comunicar alumnos y profesores, la Universidad de Sevilla decidió disponer de una plataforma centralizada de gestión del aprendizaje. Es lo que en inglés se conoce como *Learning Management System* (LMS), que consiste en una aplicación informática, que permite administrar, documentar y realizar seguimientos de las tareas de aprendizaje que realizan los alumnos de manera electrónica, organizadas según al curso al que están asociadas cada una de estas tareas. Existen variedad de sistemas de este tipo, siendo el caso

que nos ocupa un sistema Blackboard Learn 9.1 SP8 al comienzo del desarrollo de este proyecto.

En la US, cada alumno posee un usuario dentro de esta plataforma (que puede ser observada en su menú inicial en la Figura 2.2), y es suscrito automáticamente a aquellas asignaturas de las que está matriculado. Cada una de estas asignaturas se presenta en el LMS como un curso, como puede observarse en la Figura 2.3, teniendo por tanto asociado su material correspondiente para los alumnos (apuntes, temario, ejercicios, etcétera) como puede verse en la Figura 2.4, donde se muestran las carpetas creadas por el profesorado de una asignatura, junto con los documentos pertenecientes a una de ellas en la Figura 2.5; sus tareas asignadas (en las que puede tener que entregar un fichero o texto una vez completada, por ejemplo); o incluso exámenes tipo test que son automáticamente corregidos por el LMS.

Figura 2.9. Vista general de la EV

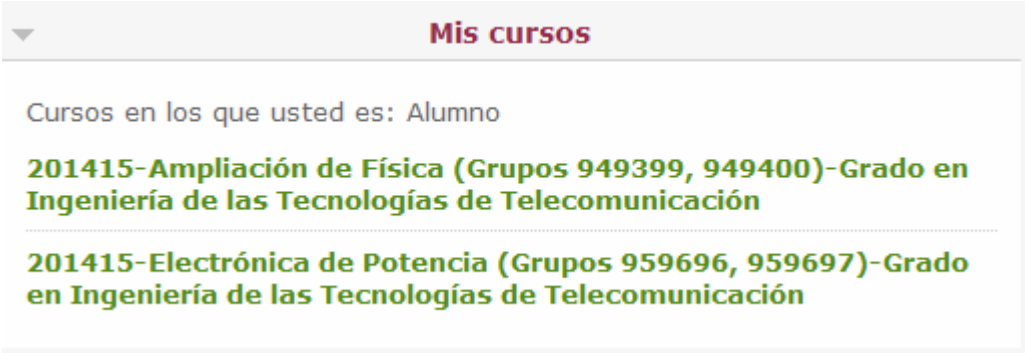

Figura 2.10. Zoom en los cursos del alumno.

Figura 2.11. Carpetas pertenecientes a un curso/asignatura.

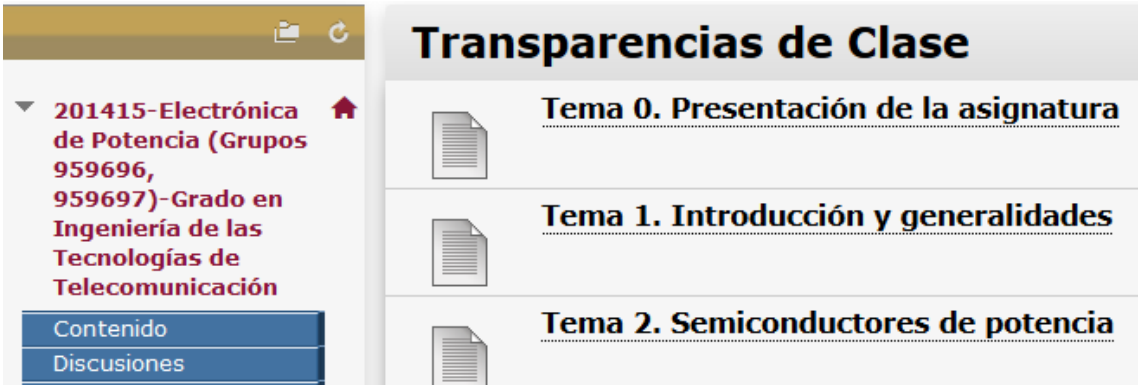

Figura 2.12. Contenidos pertenecientes a una carpeta.

Esta vista, presenta un esquema jerárquico como hemos podido comprobar, comenzando por un menú inicial para seleccionar el curso en el que queremos entrar, seguido de un menú lateral una vez dentro del mismo. A partir de aquí, el contenido del menú lateral seleccionado se presenta en el centro de la pantalla, funcionando de manera normal como los directorios del ordenador que usted use habitualmente.

Sin embargo, esta vista no es la única, sino que está compuesta por aquellos datos añadidos por el profesorado responsable de la asignatura en la plataforma, siempre que estén marcados como accesibles para los alumnos. Esto es, pueden estar almacenados datos, enlaces o contenido, al que los alumnos no pueden acceder ni, por tanto, ser visto por ellos, pero sí por los profesores. Es decir, la plataforma cuenta con *roles*, que otorgan un nivel de privilegios al usuario a la hora de interaccionar con la plataforma, tanto en su uso normal (un profesor puede ver y emplear opciones de adición/eliminación de datos, enlaces, etcétera), como en el acceso a contenidos.

Conclusiones

Tras este bloque, ya conocemos como se encuentran estructurados todos los elementos que debemos crear, remodelar e interconectar para la elaboración de nuestra herramienta. Esto nos permitirá, llegado el momento, crear una herramienta que esté modelada correctamente, lo cual implicará una arquitectura lo más sencilla y eficiente posible que permita interconectar estos elementos.

Pasaremos por tanto a realizar el diseño de nuestra herramienta en el próximo bloque, para una vez terminado dicho diseño, implementar y probar la herramienta, de manera que quede validado todo el proceso con este último paso.

BLOQUE 3. HERRAMIENTA LTI

Introducción

A continuación, empleando como base la situación actual antes descrita, y aquellas necesidades que motivan este proyecto, pasaremos a diseñar y elaborar la herramienta LTI. Para ello, describiremos las especificaciones del sistema en base a las peticiones del profesorado de la asignatura FP2 como base de trabajo.

Especificaciones del Sistema

A la hora de realizar este proyecto, el profesorado de la asignatura FP2 nos presenta una serie de requisitos, que deben ser cumplidos con mayor o menor importancia para el correcto desempeño de la aplicación, tanto por su funcionamiento interno como por el entorno en el que debe funcionar.

Recogeremos por tanto los requisitos, como se realiza comúnmente en la ingeniería de software [36], mediante una clasificación de requisitos generales, funcionales y otros requisitos no funcionales. Estos requisitos serán expresados en tablas, donde se les asignará un código identificativo, un nombre descriptivo que permita saber sobre que trata el requisito, y tres parámetros que lo acompañarán, a saber:

- **Prioridad**: con la que debe cumplirse el requisito. Siendo los posibles valores: Alta, como fundamental para el proyecto; Media, como relevante para el proyecto; y Baja, como accesorio dispensable.
- **Requisitos previos**: como aquellos requisitos que son necesarios antes de poder llevar a cabo la solución al requisito que esta siendo descrito.
- **Descripción**: del requisito en cuestión.

3.2.1 Requisitos

- Requisitos Generales

RG-01	**FUNCIONAMIENTO DEL SISTEMA**
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe calificar el trabajo entregado por el alumno, procesándolo conforme a los distintos apartados que desea evaluar el profesorado.

Tabla 3.1. Requisito General 01 “Funcionamiento del Sistema”

RG-02	INTERFAZ DE USUARIO PARA EL ALUMNO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe presentar una interfaz gráfica que permita el uso de la herramienta por parte del alumno sin necesidad de la intervención del profesorado.

Tabla 3.2. Requisito General 02 "Interfaz del Alumno"

RG-03	CARGA DEL TRABAJO DEL ALUMNO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe obtener el trabajo seleccionado por el alumno para su entrega, almacenándolo y procesándolo de manera local.

Tabla 3.3. Requisito General 03 "Carga del trabajo del alumno"

RG-04	INTERFAZ DE USUARIO PARA EL PROFESORADO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe presentar una interfaz gráfica de cara al profesorado, que le permita comprobar los valores de configuración con los que está funcionando la herramienta.

Tabla 3.4. Requisito General 04 "Interfaz del Profesor"

RG-05	CORRECCIÓN DEL TRABAJO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe corregir el trabajo entregado por el alumno, diferenciando los distintos apartados y sus resultados de evaluación.

Tabla 3.5. Requisito General 05 "Corección del Trabajo"

RG-06	ALMACENAMIENTO DE RESULTADOS
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe almacenar el resultado de las pruebas de evaluación a las son sometidos los trabajos entregados por los alumnos, tanto el resultado de los diferentes apartados como la nota final obtenida.

Tabla 3.6. Requisito General 06 "Almacenamiento de Resultados"

RG-07	ALMACENAMIENTO DEL TRABAJO DEL ALUMNO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe almacenar el trabajo entregado por el alumno, permaneciendo este disponible al profesorado una vez terminado el uso de la herramienta.

Tabla 3.7. Requisito General 07 "Almacenamiento del Trabajo"

RG-08	CONEXIÓN LTI
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe ser lanzado desde la Enseñanza Virtual de la Universidad de Sevilla, empleando el protocolo LTI.

Tabla 3.8. Requisito General 08 "Conexión LTI"

RG-09	SISTEMA OPERATIVO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe ser capaz de funcionar en Sistemas Operativos Linux.

Tabla 3.9. Requisito General 09 "Sistema Operativo"

- Requisitos Funcionales del Sistema
    - Requisitos de Información

RI-01	DATOS DE UN ALUMNO
PRIORIDAD	Alta
REQUISITOS PREVIOS	RG-08
DESCRIPCIÓN	El sistema debe trabajar con los siguientes parámetros: • UVUS del alumno, como identificador del mismo • Nota de cada apartado del trabajo • Nota final del trabajo

Tabla 3.10. Requisito de Información 01 "Datos de un Alumno"

- Requisitos de Conducta del Sistema

RC-01	PETICIONES GET DESHABILITADAS
PRIORIDAD	Alta
REQUISITOS PREVIOS	RG-08
DESCRIPCIÓN	El sistema mostrará un mensaje de imposibilidad de acceder a la herramienta mediante peticiones HTTP Get, en caso de que el TC así lo intente. Esto es así debido a las indicaciones de la recomendación de LTI.

Tabla 3.11. Requisito de Conducta 01 "Peticiones GET Deshabilitadas"

RC-02	PETICIONES POST INCORRECTAS
PRIORIDAD	Alta
REQUISITOS PREVIOS	RG-08
DESCRIPCIÓN	El sistema mostrará un mensaje de error de configuración, en caso de que la petición HTTP Post recibida no pueda ser validada como requiere LTI.

Tabla 3.12. Requisito de Conducta 02 "Peticiones POST Incorrectas"

RC-03	LÍMITE DE CORRECCIONES SIMULTÁNEAS
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema mostrará un mensaje de aviso en caso de que, al intentar corregir su trabajo un alumno, se encuentren ya en dicho proceso una cantidad considerable de alumnos. Esta cantidad será fijada de manera experimental conforme al equipo encargado de alojar la herramienta.

Tabla 3.13. Requisito de Conducta 03 "Límite de Correcciones Simultáneas"

RC-04	EJECUCIÓN CORRECTA DE LA CORRECCIÓN
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema mostrará un mensaje notificando la finalización de la corrección al alumno, indicando que el proceso ha concluido correctamente, y el número de veces que el alumno ha cargado el trabajo como nota informativa.

Tabla 3.14. Requisito de Conducta 04 "Ejecución Correcta"

RC-05	EJECUCIÓN INCORRECTA DE LA CORRECCIÓN
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema mostrará un mensaje de aviso en caso de que, al intentar corregir el trabajo del alumno, se detecte un error en alguna de sus fases, indicando al alumno que debe intentar entregar el trabajo de nuevo. Se puede acompañar este mensaje con una nota informativa si el problema es conocido.

Tabla 3.15. Requisito de Conducta 05 "Ejecución Incorrecta"

- Requisitos No Funcionales del Sistema
    - Requisitos de Seguridad

RS-01	DIFERENCIACIÓN DE USUARIOS
PRIORIDAD	Alta
REQUISITOS PREVIOS	RG-08
DESCRIPCIÓN	El sistema debe mostrar la interfaz de usuario correspondiente en función de la categoría del usuario, es decir, alumno o profesor.

Tabla 3.16. Requisito de Seguridad 01 "Diferenciación de Usuarios"

RS-02	CONTROL DE ACCESO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema solo debe permitir el acceso a aquellos alumnos matriculados.

Tabla 3.17. Requisito de Seguridad 02 "Control de Acceso"

- Requisitos de Fiabilidad

RF-01	**CONCURRENCIA**
PRIORIDAD	Alta
REQUISITOS PREVIOS	RG-01
DESCRIPCIÓN	El sistema debe permitir el uso simultáneo por parte de varios alumnos para corregir sus trabajos.

Tabla 3.18. Requisito de Fiabilidad 01 "Concurrencia"

- Requisitos de Usabilidad

RU-01	**INFORMACIÓN ESCUETA**
PRIORIDAD	Media
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe ser conciso en los mensajes mostrados al alumo, no dejando lugar a dudas en su significado y evitando texto en gran extensión en las distintas notificaciones mostradas.

Tabla 3.19. Requisito de Usabilidad 01 "Información Escueta"

- Requisitos de Mantenibilidad

RM-01	**REUTILIZACIÓN ANUAL**
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe ser reutilizable para otras convocatorias de esta asignatura.

Tabla 3.20. Requisito de Mantenibilidad 01 "Reutilización"

RM-02	**REUTILIZACIÓN ADAPTADA**
PRIORIDAD	Media
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe ser reutilizable para la corrección de otros trabajos diferentes a la PFED y la PFPOO.

Tabla 3.21. Requisito de Mantenibilidad 02 "Reutilización Adaptada"

- Requisitos de Portabilidad

RP-01	USO INDISTINTO DEL SISTEMA OPERATIVO
PRIORIDAD	Alta
REQUISITOS PREVIOS	Ninguno
DESCRIPCIÓN	El sistema debe funcionar bajo cualquier sistema operativo en lo que al alumno se refiere. Esto es, el alumno podrá acceder al sistema usando Windows, Linux, etc.

Tabla 3.22. Requisito de Portabilidad 01 "Uso Indistinto del Sistema Operativo"

3.2.2 Casos de Uso del Sistema

Separaremos los casos de uso en dos grupos, por un lado aquellos relacionados con los actores humanos, y por otro aquellos que impliquen a partes de nuestro sistema interaccionando entre sí.

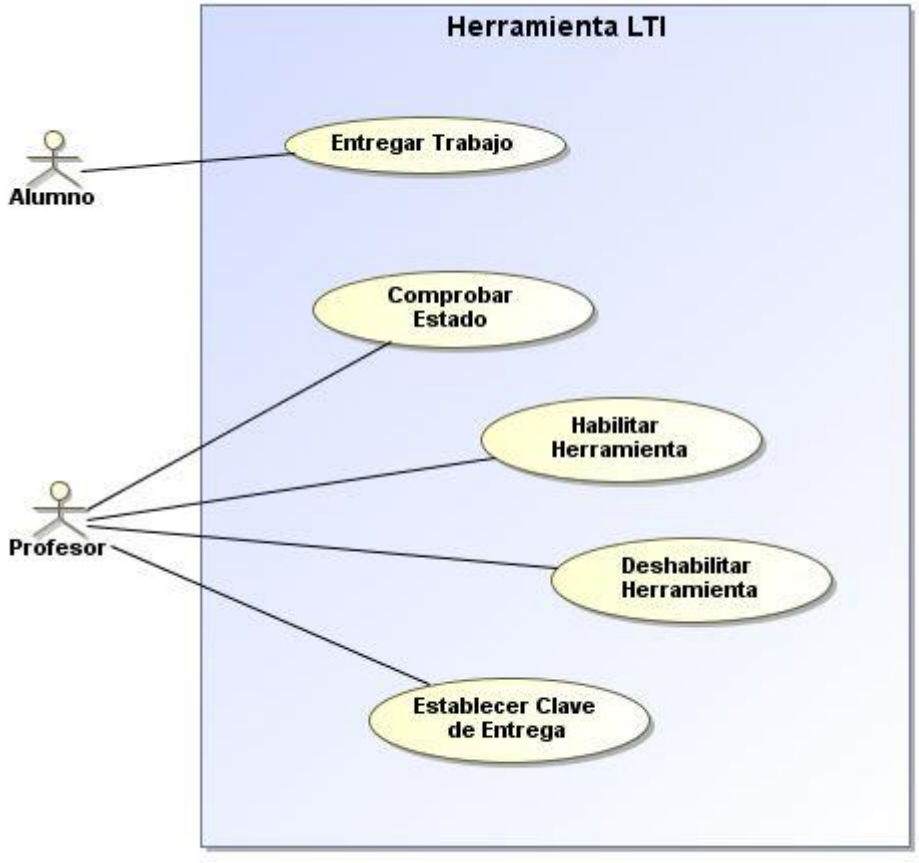

Figura 3.1. Casos de Uso Actores Humanos

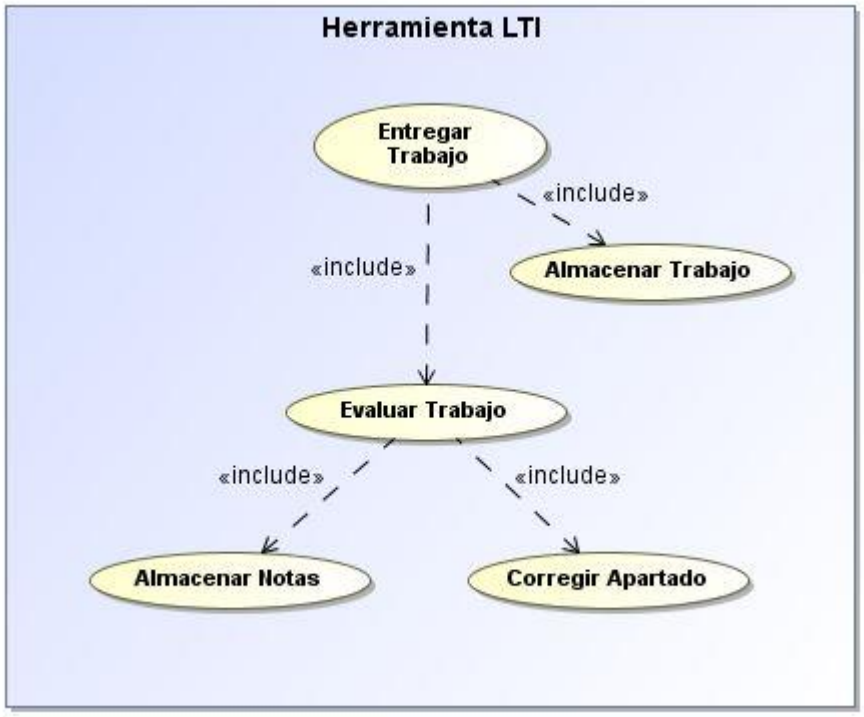

Figura 3.2. Caso de Uso "Entregar Trabajo" detallado

A continuación, vamos a especificar los casos de uso detallados en los modelos anteriores.

CS-01	**ENTREGAR TRABAJO**
PRIORIDAD	Alta
DEPENDENCIAS	Invoca a los casos de uso: • Almacenar trabajo • Evaluar trabajo
DESCRIPCIÓN	El alumno que ha accedido a la herramienta entrega su trabajo, siendo este cargado en el sistema y recibiendo el alumno un mensaje indicando si la operación ha concluido con éxito, o una notificación de error en caso contrario.

Tabla 3.23. Caso de Uso 01 "Entregar Trabajo"

CS-02	**ALMACENAR TRABAJO**
PRIORIDAD	Alta
DEPENDENCIAS	Ninguna
DESCRIPCIÓN	El fichero seleccionado por el alumno para ser entregado es cargado al sistema, que lo almacena para ser inmediatamente evaluado.

Tabla 3.24. Caso de Uso 02 "Almacenar Trabajo"

CS-03	EVALUAR TRABAJO
PRIORIDAD	Alta
DEPENDENCIAS	Invoca a los casos de uso: • Corregir apartado • Almacenar notas
DESCRIPCIÓN	El trabajo del alumno es evaluado, para lo que se corrigen los distintos apartados, se calcula la nota y se almacena.

Tabla 3.25. Caso de Uso 03 "Evaluar Trabajo"

CS-04	CORREGIR APARTADO
PRIORIDAD	Alta
DEPENDENCIAS	Ninguna
DESCRIPCIÓN	El trabajo del alumno es sometido a la corrección del apartado correspondiente, declarando dicho apartado como superado o no superado.

Tabla 3.26. Caso de Uso 04 "Corregir Apartado"

CS-05	ALMACENAR NOTAS
PRIORIDAD	Alta
DEPENDENCIAS	Ninguna
DESCRIPCIÓN	El sistema almacena el desglose de notas por apartado y nota final que el trabajo del alumno ha obtenido al ser corregido, vinculando esta información al alumno mediante su UVUS.

Tabla 3.27. Caso de Uso 05 "Almacenar Notas"

CS-06	COMPROBAR ESTADO
PRIORIDAD	Media
DEPENDENCIAS	Ninguna
DESCRIPCIÓN	El profesor accede a la herramienta, y comprueba su estado, lo que implica: • Si la entrega esta habilitada o no • Las variables configurables por el profesor que se están usando, así como su estado

Tabla 3.28. Caso de Uso 06 "Comprobar Estado"

CS-07	HABILITAR HERRAMIENTA
PRIORIDAD	Media
DEPENDENCIAS	Ninguna
DESCRIPCIÓN	El profesor accede a la herramienta, y habilita el uso de la misma por los alumnos, permitiéndoles acceder a ella.

Tabla 3.29. Caso de Uso 07 "Habilitar Herramienta"

CS-08	DESHABILITAR HERRAMIENTA
PRIORIDAD	Media
DEPENDENCIAS	Ninguna
DESCRIPCIÓN	El profesor accede a la herramienta, y deshabilita su uso, no permitiendo acceder a ella a más alumnos a partir de ese instante.

Tabla 3.30. Caso de Uso 08 "Deshabilitar Herramienta"

CS-09	ESTABLECER CLAVE DE ENTREGA
PRIORIDAD	Media
DEPENDENCIAS	Ninguna
DESCRIPCIÓN	El profesor accede a la herramienta, e introduce aquellos cambios que desee sobre la clave necesaria para que los alumnos accedan a la herramienta.

Tabla 3.31. Caso de Uso 09 "Establecer Clave de Entrega"

Diseño del Sistema

Basándonos pues, en los requisitos y casos de uso especificados anteriormente, así como en el funcionamiento y capacidades de Basic LTI, podemos pasar a diseñar la herramienta LTI. Este proceso lo dividiremos en dos fases: una fase general, en la que definiremos la arquitectura de la aplicación y otros medios relacionados necesarios para su funcionamiento; una segunda fase, en la que se detallará la estructura de la aplicación a nivel de clases, etc.

3.3.1 Arquitectura del Sistema

Atendiendo a los requisitos RC-01 y RC-02, nuestro sistema debe ser capaz de atender peticiones HTTP Post. Por tanto, nuestra herramienta será una **aplicación web**, que debe ser capaz de llevar a cabo su cometido. Si además añadimos el requisito RG-08, el siguiente paso a llevar a cabo (la elección de un lenguaje para la aplicación), se encuentra condicionada bien a la existencia de una implementación de LTI para el TP en dicho lenguaje, bien a llevar a cabo nosotros mismos una implementación del mismo.

Recordando las implementaciones antes mencionadas en el apartado correspodiente sobre LTI, y en base al requisito RG-09, necesitamos una implementación de LTI que pueda ser utilizada en un entorno Linux, por lo que las opciones se reducen a elegir una de las implementaciones Java de las que disponemos, salvo que se decida implementar en este proyecto. Dado que las implementaciones existentes cubren todos nuestros requisitos, no será necesario realizar dicha implementación por nuestra parte.

A partir a los requisitos RG-02 y RG-04, resulta claro que el sistema **presentará dos interfaces completamente distintas**, dependiendo del tipo de usuario que accede al mismo, centradas en funcionalidades distintas en cada caso y con funcionamientos independientes, pero dentro de un mismo contexto.

Por último, los requisitos RG-03, RG-05, RG-06 y RG-07 nos indican que deberemos **ser capaces de cargar**, **almacenar** y **corregir el trabajo del alumno**. Estas tareas se realizarán con distintas tecnologías para aumentar la flexibilidad de la herramienta en usos posteriores, como piden los requisitos RM-01 y RM-02.

En este punto, los requisitos restantes presentan condicionantes de funcionamiento o conducta del sistema, no condicionando así el entorno o entidad del sistema en sí mismo. Por ello, podemos decidir con lo analizado hasta ahora las bases del sistema:

- Se programará en **lenguaje Java**, ya que pretendemos emplear una implementación Java de LTI.
- Por conveniencia se empleará un **Servlet** [37], dada la facilidad de trabajo con los distintos tipos de peticiones HTTP desde él, así como el acceso al mensaje íntegro recibido en la petición por el servidor.
- Se empleará **tecnología JSP**, así como **clases Java precompiladas**, para una mayor organización del código y efectividad en el procesamiento respectivamente.
- La corrección y almacenamiento del trabajo se realizará mediante **scripts Bash**, de manera que la manipulación de ficheros sea sencilla y potente, a la vez que modificable según los requisitos de cada trabajo a corregir. Estos scripts serán en Bash debido al requisito RG-09, pudiendo ser intercambiados por ficheros de comandos por lotes de Windows, en caso de ejecutarse sobre sistema Windows la herramienta.
- Se empleará **tecnología JavaScript** y **AJAX** [38] para dotar a las distintas interfaces de las funciones deseadas, sin dejar de ser interfaces sencillas y amigables al usuario.
- Almacenamiento de las notas de los trabajos en **ficheros Excel**, debido a la imposibilidad de LTI 1.0 de devolver un resultado al LMS desde el que se lanza la aplicación.

Por tanto, y basándonos en estas indicaciones, podemos describir el sistema mediante el siguiente diagrama de componentes que se observa en la Figura 3.3.

Además, emplearemos como software y hardware del equipo destinado a hacer de *Tool Provider*:

- Sistema Operativo **Linux**.
- **Conexión a Internet**, con una velocidad recomendable de al menos 10Mbps, dado que necesita poder recibir los trabajos de varios alumnos de manera concurrente.
- **Java Runtime Environment**, al menos en su versión 7 (1.7).
- **Servidor Tomcat,** al menos en su versión 7 para evitar incompatibilidades con la versión de JRE.
- **Admisión de conexiones TCP entrantes al equipo,** ya que debe poder recibir las peticiones HTTP al puerto donde esté funcionando el servidor Tomcat (por tanto, solo a dicho puerto).

De esta forma, la interacción de los distintos actores humanos con la herramienta se podría describir mediante los siguientes diagramas de interacción, suponiendo que los actores humanos se encuentran debidamente autenticados dentro del LMS en cuestión, y que todas las operaciones de acceso a la herramienta son llevadas a cabo de manera correcta (es decir, se hará una petición POST y no GET, etcétera). En la Figura 3.4, vemos el procedimiento normal de apertura de la herramienta LTI, independientemente del tipo de usuario (alumno, o profesor) que la abre.

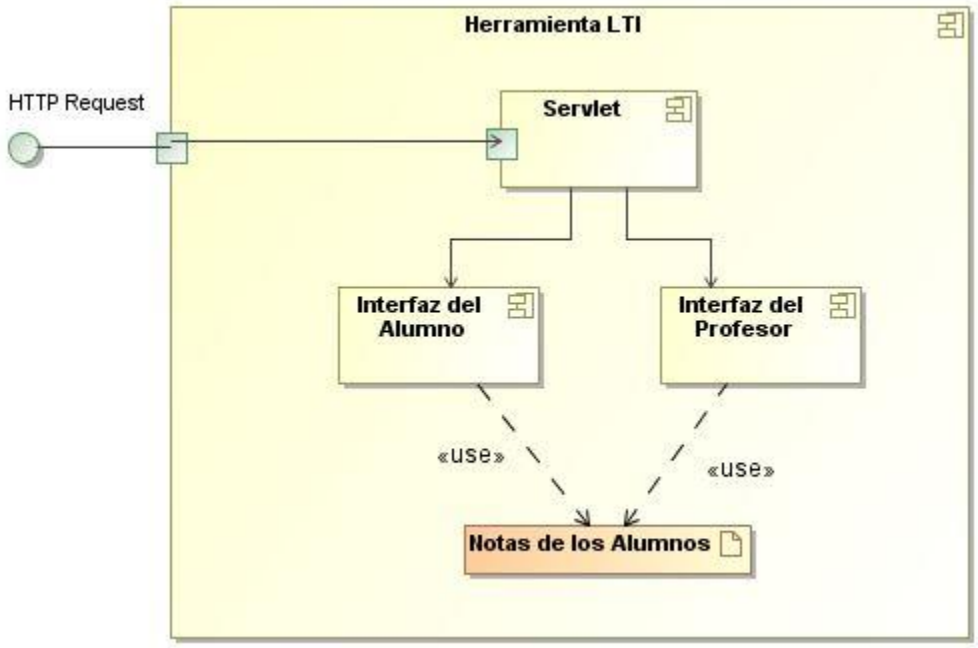

Figura 3.3. Diagrama de Componentes del Sistema

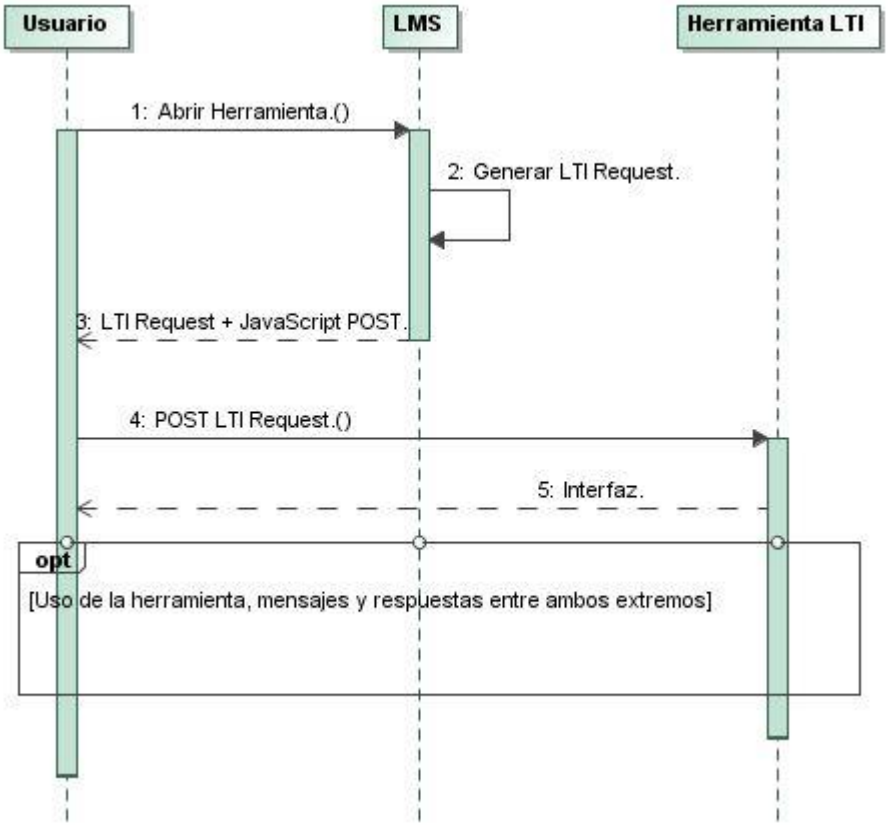

Figura 3.4. Diagrama de Interacción de Lanzamiento de la Herramienta

A continuación, en las figuras 3.5 y 3.6, observamos el uso por parte de los distintos actores humanos de la herramienta.

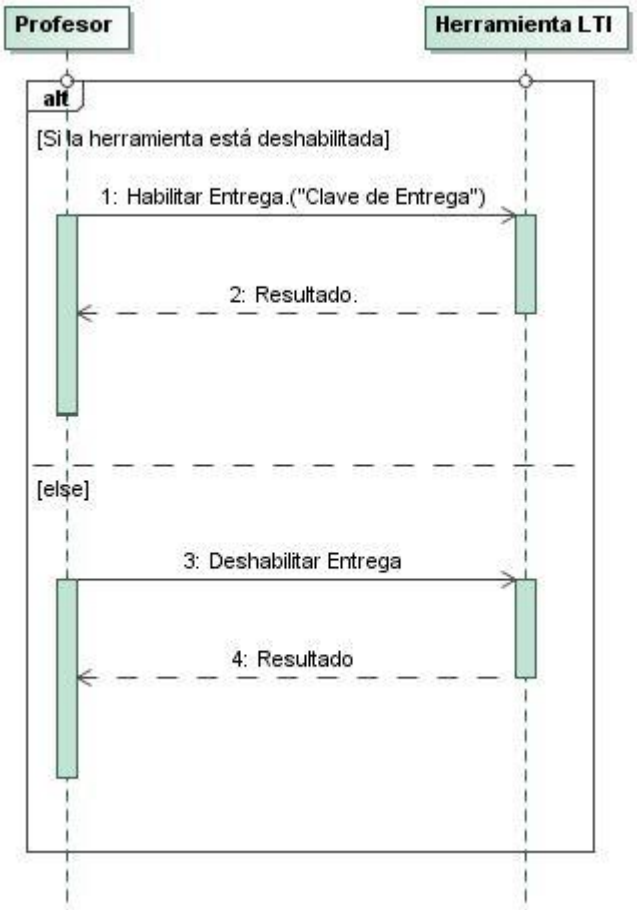

Figura 3.5. Interacción entre Profesor y Herramienta

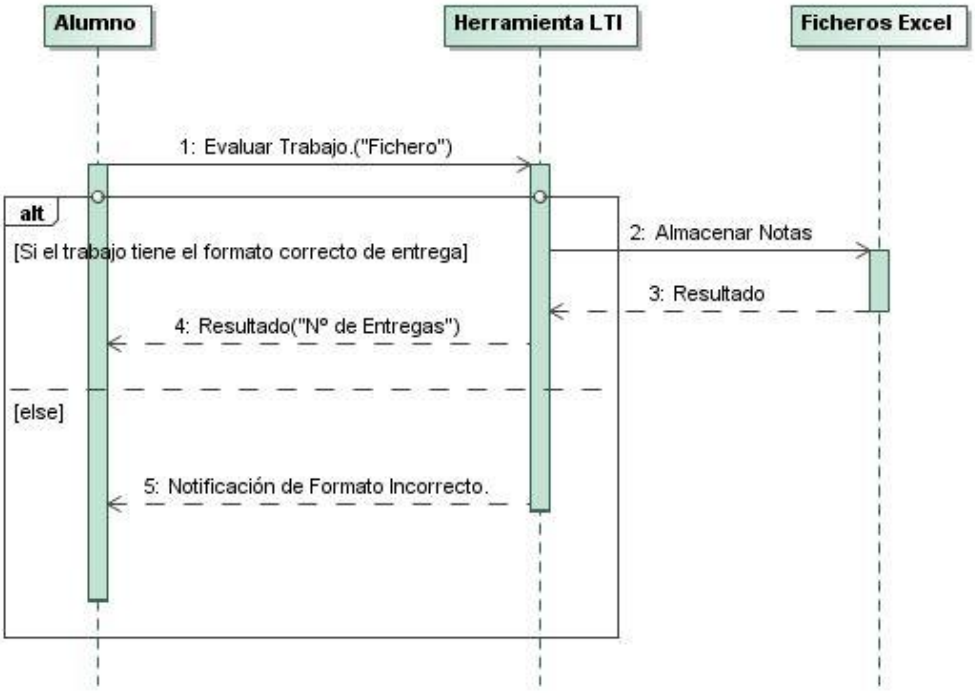

Figura 3.6. Interacción entre Alumno y Herramienta

Así pues, en caso de que la herramienta pueda corregir correctamente el trabajo entregado por el alumno, se generará un resultado informando al mismo con el número de entregas que ha realizado del trabajo hasta dicho momento. Es importante remarcar en este punto, que pese a que la herramienta almacena esta memoria de la cantidad de entregas realizadas, **la nota será exclusivamente la de la última entrega realizada**, olvidándose por completo la de entregas anteriores.

Por último, expresaremos en un diagrama de interacción los elementos de las distintas tecnologías antes mencionadas como componentes del sistema, junto con su función asociada. Esto puede observarse en la Figura 3.7.

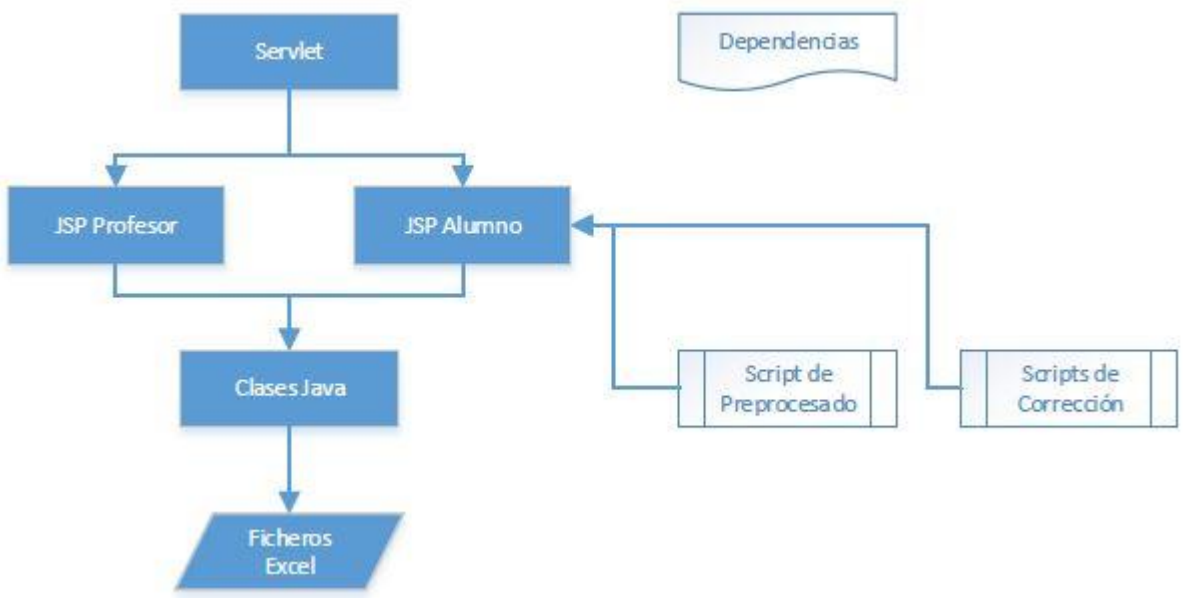

Figura 3.7. Componentes del Sistema Detallado

Implementación del Sistema

Una vez diseñado el sistema al completo, especificaremos finalmente como se implementarán los distintos subsistemas expresados hasta este momento. Basados en la Figura 3.7, especificaremos los ficheros en los que se dividirá cada bloque.

3.4.1 Servlet

El servlet es el encargado de recibir y verificar la petición LTI del usuario, y se encontrará en el fichero *ProjectEvaluation.java*, que es compilado para su despliegue en el servidor Tomcat. Como servlet, se encontrará en el subdirectorio *classes*, dentro del directorio *WEB-INF*.

En función de las distintas situaciones que puedan darse, el servlet preparará las variables de entorno necesarias para el funcionamiento de la aplicación, para posteriormente **redirigir al usuario** a la sección correspondiente. Esto puede llevarse a cabo mediante la orden *redirect*, del objeto *response* del servidor.

```
response.sendRedirect("./ruta_relativa/OtraSeccion.extension");
```

De esta manera, el servlet **configurará el UVUS** del usuario como una **variable de sesión** para el usuario que está accediendo a la herramienta, permitiendo que sea usado posteriormente por la herramienta.

```
session.setAttribute("UVUS", variable_string_UVUS);
```

Por último, se comprobará si el usuario es un profesor o un alumno, para redirigirlo a la interfaz

correspondiente. Esto puede lograrse con BLTI mediante la obtención del rol del usuario. Para ello, habremos de haber **verificado el mensaje como un mensaje LTI válido**, a través del secreto asociado a la clave recibida, y el mensaje recibido. En este punto BLTI, siguiendo el estándar LTI, presenta una dependencia con **OAuth**, en su versión 1.0 [39].

```
BLTIMessage msg = BLTIProvider.getMessage( request );
String key = msg.getKey();
String secret = null;

if (key.equals("gittfp2") == true)
{
  secret = "pol35tg7hhj1";
}

if ( BLTIProvider.isValid( msg, secret ) ) { …
```

De esta forma, al recibir la clave gittfp2, proveniente de la unión de las siglas del grado GITT con la asignatura FP2, el servlet **asocia al mensaje el secreto empleado**, generado previamente por nosotros de manera aleatoria y acordado su uso para este proyecto.

Si este paso se completa correctamente, podemos obtener del mensaje los datos del usuario que nos invoca, de donde podremos obtener si es un alumno o un profesor:

```
boolean isInstructor = msg.getUser().isInstructor();
```

Obteniendo por tanto la información necesaria para **reenviar al usuario** a la interfaz de gestión, o a la interfaz de entrega de trabajos.

Además, la configuración se ha realizado incorrectamente en el TC, y se realiza una petición GET, o bien una petición POST incorrecta, el servlet nos redirigirá al mensaje de error correspondiente con dicha situación.

3.4.2 Interfaz del Profesor

El fichero se encontrará en el fichero *Manager.jsp*, en el directorio de la aplicación web. Este fichero es al que el servlet redirige en caso de que usuario detectado es un profesor, y no un alumno.

Cuando es accedido por primera vez, si detecta que las **variables de aplicación** que emplea la herramienta aún no existen, las crea a partir de las variables de contexto de las que dispone el servlet por su despliegue: alumnos siendo atendidos, ruta de los scripts, fórmula para evaluar el trabajo, etc.

Sin embargo, su cometido principal es el de **mostrar la información de configuración** al profesorado, así como **posibilitar la habilitación y deshabilitación de la herramienta**. Para ello, la interfaz muestra datos como el nombre del proyecto que están entregando los alumnos, la ruta raíz donde está configurado que la herramienta debe crear los directorios donde trabajar con los trabajos de los alumnos, la ruta donde se encuentran los ficheros Excel con las notas de los mismos, la clave que posibilita la entrega y el estado de dicha entrega.

A la hora de generar esta interfaz, el fichero se apoya en otros ficheros, a saber:

- **Manager.css**. Como fichero de hoja de estilos en cascada, que da forma a los elementos generados en el fichero JSP.
- **AjaxManager.js**. Como fichero JavaScript, en el que se implementan las siguientes funciones:
    - **Petición AJAX de modificación**. Por la que se requiere el cambio de valor a una propiedad de la herramienta al fichero encargado de procesar estas peticiones, así como mostrar el resultado obtenido de dicha petición.
    - **changeValue/changeStyle**. Por los que se modifican los valores y estilos presentes en la información de la página, adecuándolos a los resultados actuales.
    - **Otros**. Como la alternancia entre habilitar y deshabilitar la entrega sobre un mismo botón.

A la hora de realizar un cambio sobre alguna de las propiedades de configuración, el fichero Manager.jsp invoca al fichero *ModifyStatus.jsp*, que recibiendo el nombre del parámetro y el valor que se desea fijar, realiza los cambios necesarios, para devolver el resultado a *Manager.jsp*, que es quien finalmente muestra el estado de la herramienta.

Comprobando que parámetro se desea modificar, se procede a establecer las variables de aplicación pertinentes. Además, el valor de la clave de entrega se actualiza siempre, debido a que la estructura de la interfaz no puede asegurar que no haya sido modidficada.

```
modifyName = request.getParameter("change");
if (modifyName != null)
    {
        if (modifyName.equals("deliveryEnabled"))
        {
```

Una vez nos aseguramos de que existe un parámetro que desea ser modificado, entramos en una fase de identificación del mismo, para actuar en consecuencia. Con el diseño especificado, lo único que debe ser modificado una vez la aplicación está desplegada es el estado de la herramienta, entre habilitada y deshabilitada. Cuando estas operaciones se llevan a cabo, este fichero se encarga de **hacer accesibles los objetos que representan a los ficheros Excel** de notas al resto de la herramienta, o viceversa, con el objetivo de permitir el almacenamiento de las notas de los alumnos.

Esto se realiza empleando la **librería Apache POI** [40], que es una API Java para trabajar con documentos de Microsoft Office. Con ella, podemos emplear el objeto **ExcelWorker** para representar un documento de Excel, y apartir de ahí trabajar con él como si fuese un objeto Java normal. Los cambios son llevados a cabo en memoria, teniendo que ser llevados disco cuando así se desee. Así, para hacer disponible el Excel que contiene el resumen de notas finales, que serán importadas en la EV por el profesor, realizamos:

```
ExcelWorker ewEV = null;
ewEV = new ExcelWorker(contexto.getInitParameter("evExcelPath"));
…
application.setAttribute("evExcel", ewEV);
```

Y cuando se deshabilita la herramineta, a se lleva el objeto a memoria mediante:

```
ewEV.writeFile();
```

Por último, cuando se produce un error, se apoya en el fichero *ErrorApp.jsp*, preparado para mostrar el error que le es transmitido.

3.4.3 Interfaz del Alumno

Generada por le fichero *Student.jsp*, se encuentra en el directorio de la aplicación web. Este fichero es, por el contrario al anterior, al que redirige el servlet en caso de que el usuario detectado sea un alumno.

Este fichero **comprueba inicialmente si la entrega de trabajos se encuentra habilitada**, para mostrar la interfaz correspondiente en dicho caso, o mostrar una notificación al alumno en caso contrario. Dicha interfaz consta de los elementos básicos necesarios para la entrega del trabajo, de manera que sea lo menos confusa posible para el alumno: el título del trabajo que se encuentra entregando el alumno; un control para seleccionar el fichero que debe entregar; un control preparado para insertar contraseñas, de manera que no se muestren los caracteres escritos; para acabar finalmente con **el botón de envío y evaluación del trabajo**.

Al igual que en el caso de la interfaz del profesor, el fichero se apoya en diferentes ficheros adicionales para llevar a cabo su tarea y presentar una intefaz clara y sencilla:

- **Student.css**. Como fichero de hoja de estilos en cascada, que da forma a los diferentes elementos generados en el fichero JSP. A pesar de que el estilo es similar a la intefaz del profesor, se realiza de manera separada dado que las etiquetas son distintas, permitiendo la separación y posteriores modificaciones por separado para ambas interfaces.

- **JSStudent.js**. Como fichero JavaScript, en el que se implementan las siguientes funciones.
    - **blockUI**. Como bloqueo de la entrega del trabajo, para que el alumno no pueda interactuar con la interfaz una vez entregado el trabajo, y solo deba esperar a recibir el resultado.
    - **sendWork**. Como función que valida que no se está intentando entregar el trabajo sin haber seleccionado ningún fichero en la interfaz, para pasar a invocar a las funciones AJAX correspondientes.
- **AjaxStudent.js.** Como fichero JavaScript que se encarga de realizar la petición AJAX al fichero encargado de la entrega y evaluación del trabajo, así como de presentar el resultado obtenido.

Continuando con el esquema seguido con la intefaz del profesor, cuando se entrega el trabajo lo que hace el fichero Student.jsp es invocar al fichero *UploadFile.jsp*. Este será el fichero encargado de recibir el fichero entregado por el alumno, para a continuación realizar las siguientes operaciones:

1. **Comprobación de clave**. Si la clave de entrega es correcta, se puede proceder con la entrega. Es importante, en el mensaje construido con AJAX, que la clave se situe antes en el mensaje que el fichero a adjuntar, ya que se han de tratar todos los datos adjuntados a la consulta, y se hace siguiendo dicho orden. Si se tratase antes el fichero adjunto que la clave, a la hora de entregar el fichero la clave, todavía no procesada, sería siempre incorrecta y no podríamos realizar la entrega.
2. **Creación de carpetas**. En función del UVUS del alumno, para almacenar el trabajo mientras es corregido, así como para facilitar el almacenamiento del trabajo entregado en cada una de las entregas realizadas por el alumno. Explicaremos posteriormente la estructura de directorios que se emplea y como son creadas.
3. **Carga del fichero**. Almacenándolo en la carpeta correspondiente al usuario que lo entrega. Esta operación se basa en la librería **Commons FileUpload** [41], mediante la cual, empleando las clases *ServletFileUpload*, *FileItemFactory* y *DiskFileItemFactory* podremos manejar la carga de ficheros. Para ser concretos, con el siguiente código podemos obtener una lista de los elementos recibidos en la petición, para luego poder procesar cada uno por separado:

```
FileItemFactory file_factory = new DiskFileItemFactory();
ServletFileUpload servlet_up = new ServletFileUpload(file_factory);

List items = null;
items = servlet_up.parseRequest(request);
```

4. **Preprocesado del fichero**. Empleando un script, se permiten realizar operaciones previas a la evaluación sobre el fichero entregado por el alumno. Esto es útil por ejemplo para:
    a. **Descomprimir el fichero entregado.**
    b. **Realizar una copia del fichero entregado.** Este paso es importante, ya que la implementación que se realizará, dado que debe permitir varias entregas del mismo alumno, **borra los ficheros del alumno una vez evaluado el trabajo**.
    c. **…**
5. **Copia de los ficheros de corrección.** Tanto scripts, programas como archivos auxiliares necesarios para corregir los distintos apartados del trabajo, son copiados a la carpeta del trabajo del alumno. Esto se realiza empleando la librería **Commons IO** [42], donde la clase *FileUtils* nos servirá para manejar ficheros, directorios y permisos de una manera sencilla y eficiente, además de ser una dependecia para la librería **Commons FileUpload** antes mencionada.
6. **Ejecución de la corrección.** Se corrige cada apartado y se almacena el resultado en los ficheros Excel, además de calcular la nota final y ser almacenada igualmente aplicando la fórmula de evaluación configurada. Para trabajar con los ficheros Excel, se emplea una clase creada por nosotros, que a su vez se apoya en la librería de Apache antes mencionada. Esta clase se llama *ExcelWorker*, y será explicada posteriormente. Además, de manera análoga para la ejecución de la corrección de cada apartado, emplearemos otra clase llamada *ScriptWorker*.

7. **Escritura de los ficheros Excel a disco.** Tras actualizar las notas en los ficheros, los ficheros son llevados a disco, para asegurar la persistencia de los datos frente a posibles fallos en la herramienta o en el sistema.
8. **Resultado.** Se devuelve el resultado de la corrección al alumno, indicándole cuantas entregas ha realizado hasta el momento, o la descripción del error en la ejecución de la aplicación en caso de que lo hubiera.

De estas fases aquí explicadas, debemos especificar que hace en nuestro caso la implementación realizada. Por ello, a continuación analizaremos el fichero de preprocesado elaborado para los trabajos PFED y PFPOO.

3.4.3.1 Fichero de Preprocesado

Este fichero se encontrará en la carpeta *Scripts* de la aplicación web, y será nombrado con *Preprocess_TítuloDelProyecto.extension*. La herramienta se encuentra preparada para ignorar la extensión del fichero, de manera que sea portable a Windows, o bien si se desea realizarse mediante un ejecutable que no sea un script bash. De esta manera, en nuestro caso el fichero para la PFED se nombraría: *Preprocess_PFED.sh.*

Para poder realizar correctamente su trabajo, el fichero recibe cuatro parámetros como argumento, que le permiten trabajar con el fichero y directorios en los que se encuentra. Son:

1. **Carpeta raíz**: A partir de la que se encuentran todas las carpetas de los alumnos (creadas cuando un alumno entrega su trabajo).
2. **UVUS del alumno**: para poder identificar igualmente la carpeta a la que debe acceder tras el primer paso.
3. **Título del trabajo**: dado que dentro de la carpeta de cada alumno, se trabaja dentro de una subcarpeta con el nombre del proyecto. De esta manera, **se posibilita tener en una misma estructura de directorios tener varios trabajos siendo corregidos a la vez**, por distintos despliegues de la herramienta configurada adecuadamente a cada trabajo.
4. **Nombre del fichero**: para trabajar con dicho fichero y realizar sobre él las operaciones necesarias.

Debemos hacer hincapié en que este fichero no recibe en ningún momento como parte de las rutas o nombres, las carpetas *VersionActual* o *VersionesAnteriores.* Esto se debe a que son carpetas conocidas y que son **independientes del toda la configuración de la aplicación**, por lo que a la hora de implementar este fichero se debe tener en cuenta e incluirlas en las rutas necesarias, sin esperar recibir dichos valores.

Pasemos a las dos funciones que implementa este fichero, a saber:

- **Descompresión del fichero ZIP** que entregan los alumnos.

Cuando el alumno entrega su trabajo de la PFED o PFPOO, entrega un fichero comprimido en ZIP, que contiene una serie de directorios comprimidos. Cada uno de estos directorios, corresponde a uno de los apartados que se debe evaluar, siendo autocontenidos todos ellos (es decir, no requiriendo nada para dicho apartado, de fuera de dicho directorio).

La primera función por tanto, de este script, es la descompresión de dicho fichero para la aparición de los directorios correspondientes, y los ficheros reales que deben ser analizados y evaluados. Esta descompresión se lleva a cabo en la carpeta del alumno, en el subdirectorio *VersionActual*, y para ello se emplea el comando *unzip* [43], que debe estar disponible en el equipo para su uso.

- **Copia del trabajo** entregado por el alumno.

Dado que el alumno puede realizar varias entregas, se hace imperativo que con cada una de ellas, los ficheros evaluados sean íntegramente los de dicha entrega. Para asegurar esto, la herramienta debe eliminar los ficheros tras cada corrección, en pos de evitar posibles problemas en este aspecto.

La herramienta habrá creado, dentro del directorio del alumno, una segunda carpeta denominada *VersionesAnteriores*, que una vez creada nunca es eliminada por la herramienta, y que pretende facilitar el almacenamiento de los trabajos entregados por el alumno. El segundo cometido de este script será, por tanto, realizar una copia del fichero ZIP entregado por el alumno a dicha carpeta. Además, dado que las entregas pueden sucederse, en nuestro caso el script verificará en primer lugar cuantas copias existen ya de la entrega en

dicha carpeta, para renombrar el fichero al copiarlo con un número identificativo de dicho orden al final. Esto es, si la entrega es la primera por parte del alumno, el fichero será renombrado desde su nombre original, tal que:

```
Trabajo.zip → Trabajo_0.zip
```

Para ir incrementando dicho contador con cada entrega.

3.4.3.2 Scripts de Corrección

El objetivo de nuestra aplicación LTI, es corregir el trabajo entregado por el alumno. Una vez procesado el fichero que este ha cargado a través de la herramienta, se proceden a ejecutar las pruebas de evaluación.

Estas pruebas podrán ser ficheros ejecutables de cualquier tipo (scripts [.sh], binarios de linux [.out], ficheros de órdenes por lotes [.bat] o binarios de windows [.exe]) (en su correspondiente sistema donde se encuentre la herramienta desplegada).

Estos deben ser elaborados por el profesor para evaluar el trabajo del alumno conforme a sus intereses, devolviendo un valor numérico si el apartado ha sido separado o no.

3.4.3.2.1 Scripts de un Trabajo XYZ

Los scripts o programas para evaluar los diferentes apartados de un trabajo XYZ, se sitúan en la ruta *deployPath*/Scripts/*projectName*/, conformando así la estructura de ficheros y directorios:

```
deployPath/Scripts/projectName/
deployPath/Scripts/projectName/Apartado_0.extension
deployPath/Scripts/projectName/Apartado_0/
deployPath/Scripts/projectName/Apartado_1.extension
...
```

Como se puede observar, en caso de que el apartado correspondiente necesite de recursos, programas o scripts auxiliares, estos deben ser situados en la ruta relativa:

```
./Apartado_Numero/
```

Siendo el número de la carpeta el mismo que el del apartado en cuestión. Si no es necesario, estos directorios auxiliares no tienen por qué existir.

3.4.3.2.2 Ejecución de los Scripts

A la hora de corregir los diferentes apartados, la aplicación LTI lo que hace es ejecutar los scripts encontrados en la carpeta antes indicada, en el orden de numeración ascendente (0, 1, etcétera).

Son estos scripts o programas de corrección elaborados por el profesor, los que deben trabajar con el trabajo entregado por el alumno y así determinar si la prueba es superada o no.

Recordemos que el trabajo ya ha sido preprocesado (en caso de que este preprocesado realice alguna operación sobre el fichero del alumno). Con este fin, a la hora de ejecutarlos reciben como primer y único parámetro, la ruta absoluta donde se encuentra el fichero del alumno. Esta ruta, proporcionada por la herramienta, será:

> Ruta raíz de la aplicación + uvus del alumno + titulo del trabajo + "VersionActual"

Esta última carpeta mencionada, VersionActual, es una carpeta empleada por la aplicación durante el tiempo de ejecución de la evaluación del alumno, siendo destruida una vez terminado el proceso.

Por ello, **se recomienda que en el script de preprocesado se realice una copia del trabajo del alumno al directorio**:

> Ruta raíz de la aplicación + uvus del alumno + titulo del trabajo + "VersionesAnteriores"

Si no, **el trabajo del alumno se perderá una vez evaluado**.

Como decíamos, la ruta del directorio "VersionActual" se proporciona al programa o script de corrección (a todos ellos, apartado tras apartado), de manera que este pueda saber donde se sitúa el trabajo del alumno para trabajar con él.

Por último, debe tener en cuenta que la evaluación no continua en ninguna de sus fases (para el proyecto de un alumno X) hasta que el apartado anterior correspondiente no ha sido completado, esto es, no termina su ejecución. Esto implica, entre otras cosas, tener sumo cuidado para no añadir interacciones con el usuario que pudiesen bloquear uno de estos programas, ya que esta interacción jamás llegaría.

Recomendación: a la hora de trabajar con el script o programa de corrección, se aconseja que se emplee alguna función para obtener la ruta desde la que se está ejecutando el programa, de manera que puedan utilizarse rutas absolutas construidas a partir de dicha ruta para acceder al resto de ficheros auxiliares del apartado, concatenando dicha ruta. Por ejemplo, si estamos ejecutando el Apartado_0.sh (nótese la notación de bash empleada en este caso):

```
rutaRecursoAAA="$rutaActual/Apartado_0/RecursoAAA"
```

Donde *rutaActual* es la variable donde habremos almacenado la ruta desde la que se está ejecutando el script Apartado_0.sh actualmente.

3.4.3.2.3 Resultado de los Scripts

Una vez ejecutado un apartado, la aplicación recibe el valor devuelto por el mismo, siendo los valores posibles 0 y 1.

> **0** significa que el apartado del alumno es correcto
>
> **1** significa que ha habido algún error en el apartado y por tanto el alumno no lo ha realizado correctamente

Por tanto, los scripts o programas que corrigen cada apartado, deben devolver uno de estos valores. Estos valores no deben ser devueltos a la salida estándar, sino que **deben ser devueltos como valor de la ejecución** del script o programa. Esto es, por ejemplo:

> En un programa: el valor int devuelto por la función main.
>
> En un script: valor devuelto tras la ejecución de la última sentencia, o bien uno personalizado por el autor del script empleando la orden *exit valor* (en el caso de bash).

3.4.3.2.4 Scripts para PFED y PFPOO

Una vez conocido como funcionan los scripts de corrección de la herramienta, vamos a ver como están hechos los scripts empleados en los trabajos que nos ocupan.

- **PFED**. Basados en un *makefile* empleado anteriormente en la asignatura para realizar estos apartados, se han modificado para convertirlos en un script, de manera que:
  - En caso de que se le proporcionase una ruta objetivo como parámetro al ejecutarse, trabajase con los ficheros a partir de dicha ruta en lugar de la localización del propio script.
  - Devolviese un valor como resultado de ejecución, para poder evaluar el apartado.
- **PFPOO**. Se generaron empleando de base los ficheros de corrección que existían anteriormente.
  - Invocan al make del alumno para compilar su código.
  - Ejecutan los binarios generados, para después comparar la salida generada y devolver el resultado de la ejecución según lo explicado en los apartados anteriores.

3.4.3.3 Clases Precompiladas

En la herramienta se emplean tres clases precompiladas, de manera que se evite una gran cantidad de código de procesamiento no relevante para la interfaz de usuario al JSP, y facilitando la estructura de la aplicación. Se sitúan en la carpeta *classes* de la aplicación web, bajo su árbol de directorios correspondiente al paquete en el que se las ha incluido a cada una de ellas. Veamos qué nos ofrece cada una de ellas:

- **ExcelWorker**. Esta clase se emplea para trabajar con los ficheros Excel que nos permiten almacenar las notas de los alumnos. Depende de la librería Apache POI, y consta de:
    - **Constructor**. A partir de una ruta de fichero Excel, genera un objeto de esta clase para trabajar con dicho fichero.
    - **UpdateMarks**. A partir del UVUS del alumno, un array de variables boolean con los resultados de la corrección de los apartados, y la nota final del alumno (ya calculada), se insertan estos resultados en el Excel del profesor, bien insertando al nuevo alumno, bien actualizando sus calificaciones si ya existiese. Al finalizar la operación de actualización del fichero, este se lleva de memoria virtual a disco duro.
    - **UpdateMarks [Sobrecarga]**. A partir del UVUS del alumno, y su nota final ya calculada, se insertan los resultados en el Excel empleado para introducir las notas en la EV. Para más detalles respecto a este punto, diríjase al *Anexo A – Ficheros Excel de Evaluación*.
    - **Funciones Auxiliares**. Para llevar la cuenta del número de entregas, que se refleja en el Excel del profesor, llevar el fichero a disco, o comprobar que un usuario ya existe por su UVUS en el fichero Excel.
- **ScriptWorker**. Esta clase se emplea para trabajar con los ficheros de corrección del trabajo, permitiéndonos ejecutarlos, pasándoles ciertos parámetros y devolviendo dichos resultados. Consta de:
    - **RunPreprocess**. Ejecuta el fichero de preprocesado del trabajo sobre el fichero del alumno.
    - **RunExamination**. Ejecuta los distintos ficheros de corrección, que encuentra en la carpeta del alumno, donde han sido copiados anteriormente. Son ejecutados en orden, de manera que **se comienza en el apartado nº 0**. Esta numeración es importante recordarla, y se menciona igualmente en el *Anexo B – Manual de Instalación*, a la hora de configurar la fórmula de evaluación.
    - **ExecScript**. Como función privada que permite ejecutar un único fichero de corrección, es por tanto en el que se apoya RunExamination para llevar a cabo la evaluación.
- **ErrorControlado**. Se trata de una clase que combina en su interior un valor booleano, junto con una excepción. Es empleado a lo largo de la herramienta con el objetivo de poder elevar la información de las excepciones sin necesitar capturarlas de esta forma, sino devolviendo esta clase como resultado en distintas operaciones. De esta manera, se evita tener redundancia de separación de situaciones, en las que un error puede darse por llegar a un punto del programa por una consencuencia anterior, o bien por la ejecución del código que genere una excepción. Es decir, en el siguiente fragmento, si capturásemos la excepción, habría que duplicar en el control de la excepción el mismo código que en el *else* que se muestra, o bien generar código no estructurado, cosa que con esta clase evitamos al seguir trabajando con la información de la excepción pero sin emplear el mecanismo de las excepciones.

```
If (situacion_correcta){
    // …
    // Línea que genera la excepción -> Situación Incorrecta
    //
}else{
    // Situación Incorrecta
}
```

De esta forma, dado que las excepciones que pueden producirse en la aplicación no pueden ser paliadas mediante el control de excepciones, se opta por informar al usuario de que un error ha ocurrido para que sea él quien reintente la entrega del trabajo.

3.4.4 Estructura de Directorios

Hasta el momento, hemos especificado en qué carpeta trabaja cada fichero o en que carpeta se sitúa, pero con

esto puede quedar un mapa incompleto y desclasificado de estos directorios. Por tanto, recapitularemos en este punto y estableceremos el mapa completo de ficheros de la aplicación, tanto de los propios de la aplicación, como donde se trabaja con los ficheros de los alumnos.

No se mostrará en estas estructuras de directorios la carpeta donde se encuentren los ficheros Excel, ya que el profesor puede situarlos donde prefiera sin importar su localización, y no están relacionados con el resto de la estructura de directorios.

3.4.4.1 Estructura de la Herramienta

Llegados a este punto, conocemos todos los componentes de la herramienta LTI, pero, ¿dónde se ubican? Hagamos un recorrido por los distintos niveles de la herramienta una vez se encuentra desplegada para su funcionamiento.

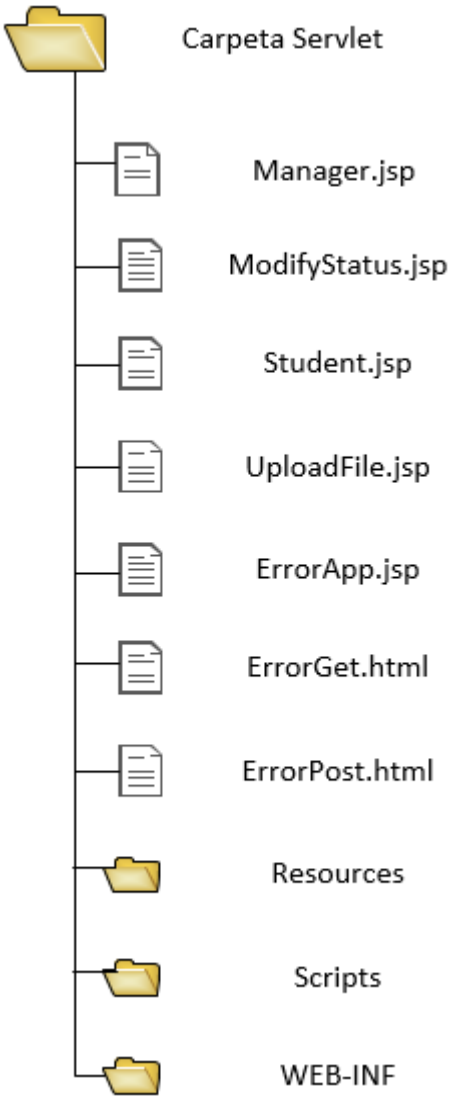

Figura 3.8. Directorios y Ficheros de la Herramienta. Nivel 1.

Como podemos observar en la Figura 3.8, en este primer nivel se sitúan todos los ficheros *jsp* y *html*, además de las diferentes carpetas que contienen el resto elementos de la aplicación web, a saber:

- **Resources**: es la carpeta en la que se almacenan el resto de ficheros necesarios de la aplicación web como tal, es decir, ficheros *css* y *JavaScript/AJAX*.

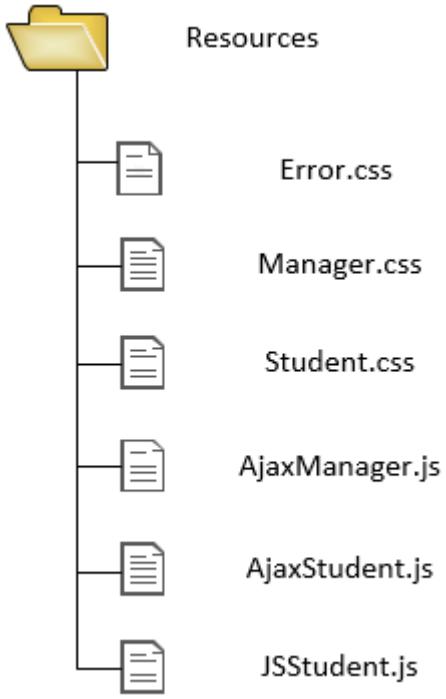

Figura 3.9. Directorio Resources Detallado.

- **Scripts**: es la carpeta donde se almacenan los ficheros de preprocesado y corrección del trabajo. El **fichero de preprocesado es ejecutado desde esta ubicación**, recibiendo los parámetros especificados en el apratado 3.4.3.1 para localizar y trabajar con el trabajo del alumno. Veamos en la Figura 3.10 el detalle de cómo quedaría para nuestra PFED.

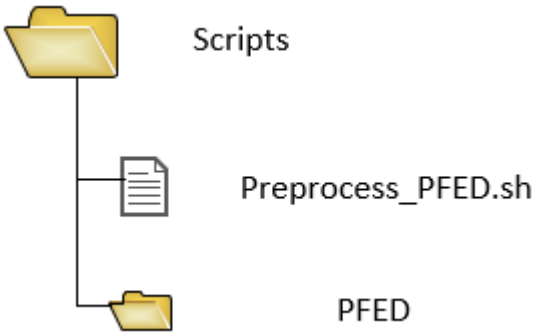

Figura 3.10. Directorio Scripts Detallado.

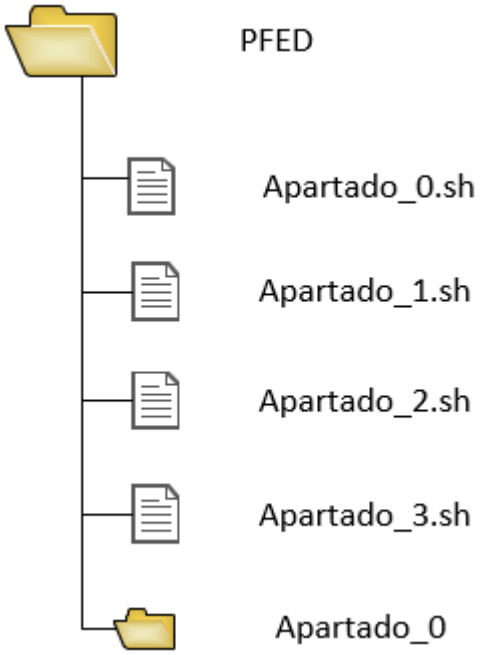

Figura 3.11. Subdirectorio de Scripts de Corrección

- **WEB-INF**: es la carpeta donde se encuentra el resto de la aplicación web, es decir, el servlet, librerías de dependencias, e incluso nuestras propias clases compiladas.

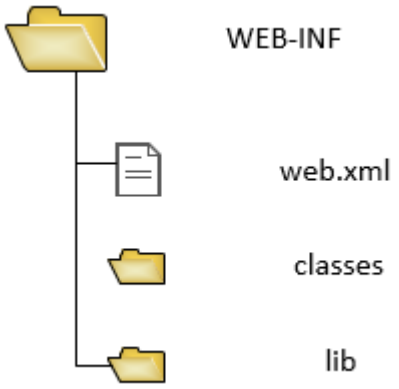

Figura 3.12. Directorio WEB-INF Detallado.

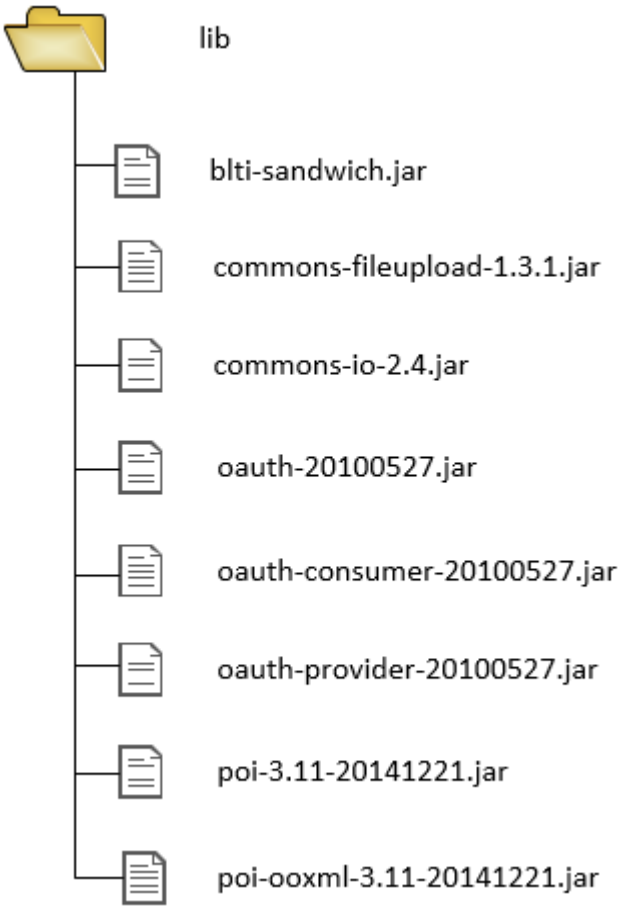

Figura 3.13. Directorio lib Detallado

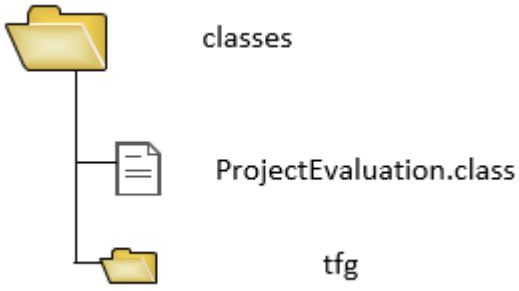

Figura 3.14. Directorio classes Detallado

Siendo el directorio *tfg* el escogido para la raíz de paquetes empleado en las clases java implementadas.

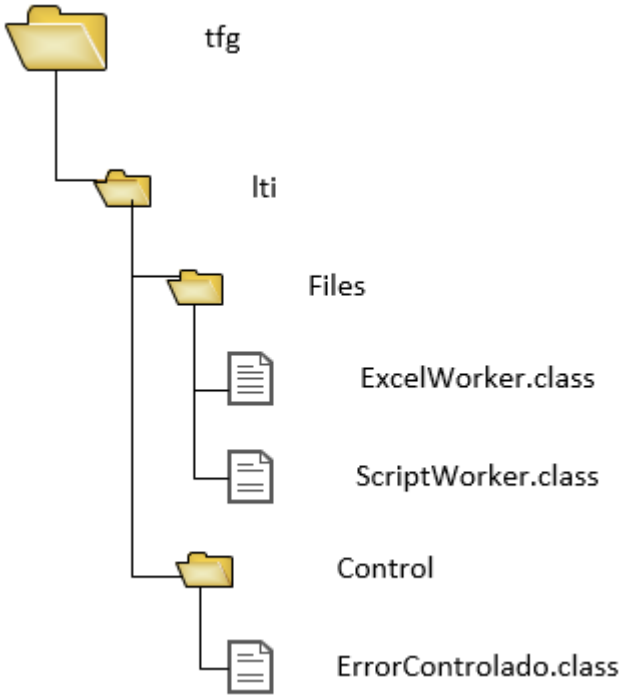

Figura 3.15. Detalle del Árbol de Directorios bajo tfg

3.4.4.2 Estructura de los Trabajos

Por otro lado, nos encontramos con la estructura de directorios donde se almacenan los trabajos de los alumnos para su corrección. Esta estructura de directorios puede ser observada en la Figura 3.16, donde se aprecia que desde la ruta inicial especificada en la configuración de la aplicación (ver *Anexo B – Manual de Instalación*), encontramos las subcarpetas correspondientes a los distitnos alumnos, siendo el nombre de cada una de ellas su UVUS.

Acto seguido, bajo la carpeta de cada usuario se encuentra la carpeta con el nombre del proyecto, para finalmente contener las carpetas *VersionActual*, *VersionesAnteriores*, y de las correcciones si existiesen. Es decir, como se ha comentado en los scripts de corrección, las posibles carpetas que acompañan a cada apartado con otros ficheros necesarios para el uso por parte del script o programa que corrige dicho apartado.

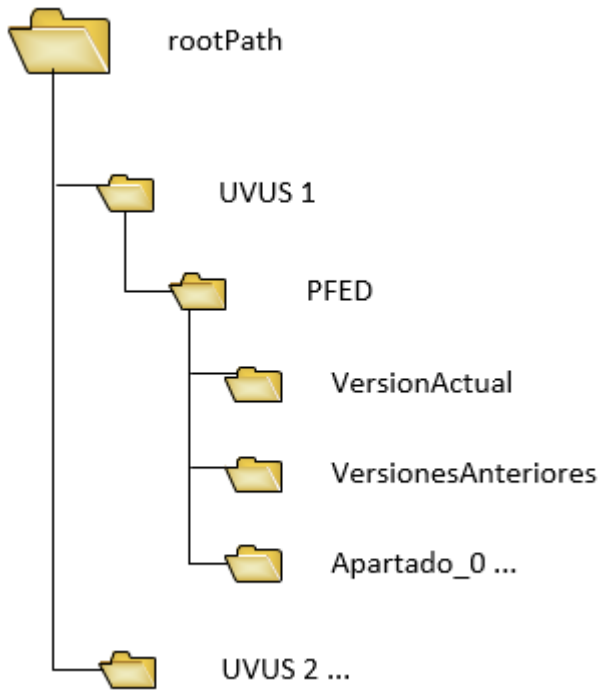

Figura 3.16. Detalle del Árbol de Directorios Bajo la Ruta Principal

Interfaces de Usuario

El cometido de las interfaces ya fue explicado anteriormente. Es por esto, que aquí nos centraremos en su aspecto, y como este está vinculado con su funcionamiento.

3.5.1 Interfaz del Profesor

Presenta dos estados posibles, que únicamente se diferencian en el estado en el que se encuentra la herramienta: bien habilitado, bien deshabilitado. En las figuras 4.1 y 4.2, podemos observar la herramienta en estos estados respectivamente, mientras se encuentra configurada con valores de ejemplo para la *PFED*.

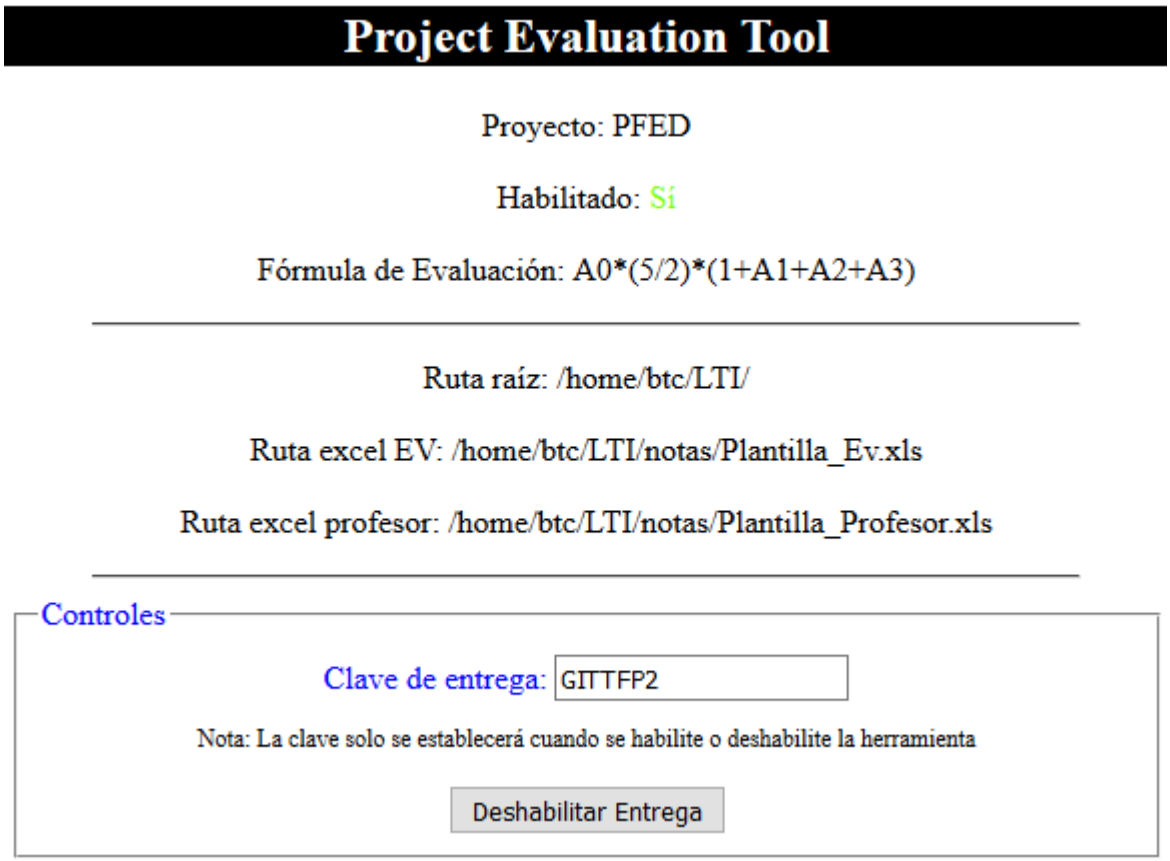

Figura 4.1. Herramienta Habilitada

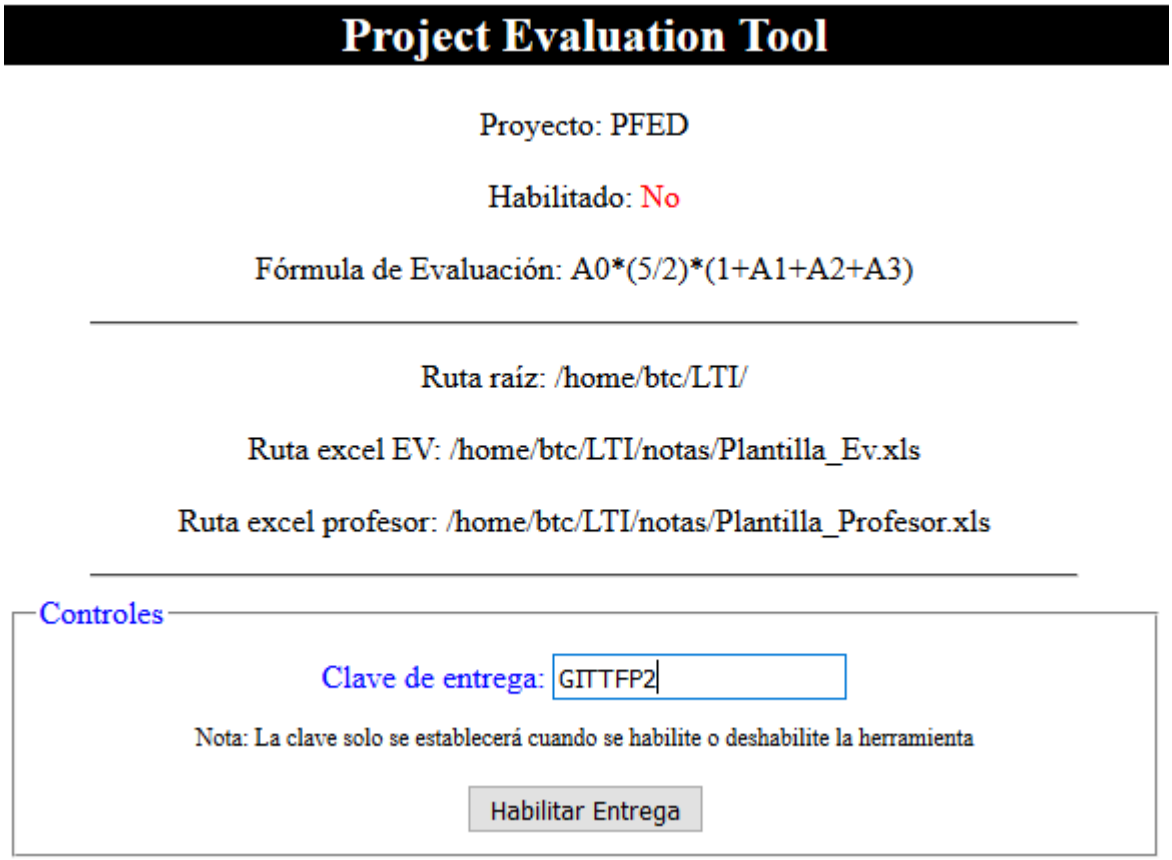

Figura 4.2. Herramienta Deshabilitada

El diseño de esta interfaz, queda marcado por tanto por los siguientes elementos:

- **Elementos Libres**. Componen la parte informativa de la interfaz.
    - Un encabezado: etiqueta *H2* en HTML. Se emplea para dar nombre a la herramienta.
    - Elementos de párrafo: etiqueta *P* en HTML. Una para cada dato que se muestra.
    - Un elemento en línea: etiqueta *SPAN* en HTML. Se emplea para poder generar en una misma línea dos elementos independientes, de manera que tengan diferentes características de estilo. De esta manera, el valor del estado de la herramienta puede ser enfatizado con un estilo que no es aplicado a la etiqueta informativa sobre qué valor se está mostrando.
    - Dos elementos separadores: etiqueta *HR* en HTML. Nos permite dividir la información de la misma clase en la interfaz, creando así dos grupos: uno para la información de uso de la herramienta (título, estado, fórmula de evaluación); y otro para la información referente al servidor (directorio raíz y ficheros Excel empleados con su ruta absoluta).
- **Elementos Agrupados**. Componen la parte organizativa de la interfaz, así como aquellos elementos con los que se puede interactuar.
    - Elemento de agrupación: etiqueta *FIELDSET* en HTML. Con él, podemos agrupar un conjunto de elementos, mostrando un nombre para dicho grupo. De esta manera, conseguimos crear el efecto panel de control, reconociendo como controles los elementos aquí recogidos.
    - Elemento formulario: etiqueta *FORM* en HTML. De uso estandarizado en tecnología web para realizar una nueva petición a la página web enviando datos contenidos en elementos anidados en él.
    - Elemento de entrada: etiqueta *INPUT* en HTML. Empleada para permitir la modificación de la clave de entrega cuando la aplicación ya está desplegada en el servidor.

3.5.2 Interfaz del Alumno

Esta interfaz, presenta cuatro estados posibles, en el flujo de trabajo que sigue un usuario que la usa, a saber: intefaz con la herramienta deshabilitada para su uso, intefaz inicial previa a la entrega; interfaz de corrección, mientras el trabajo está siendo entregado y evaluado; e interfaz de información, con la notificación resultante de la ejecución de la herramienta. Observemos en las figuras 4.3, 4.4, 4.5 y 4.6 estos estados, fijando la figura 4.6 a una entrega correcta por parte de un alumno y no a las posibles notificaciones de error que pueden aparecer.

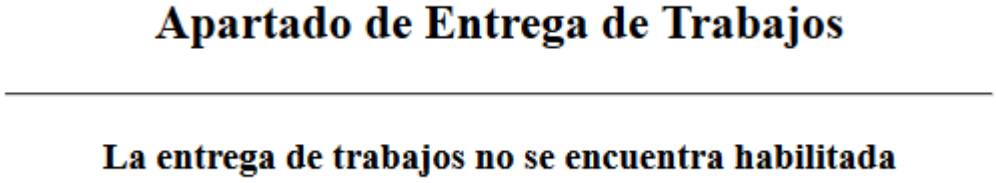

Figura 4.3. Interfaz del Alumno con la Entrega Deshabilitada

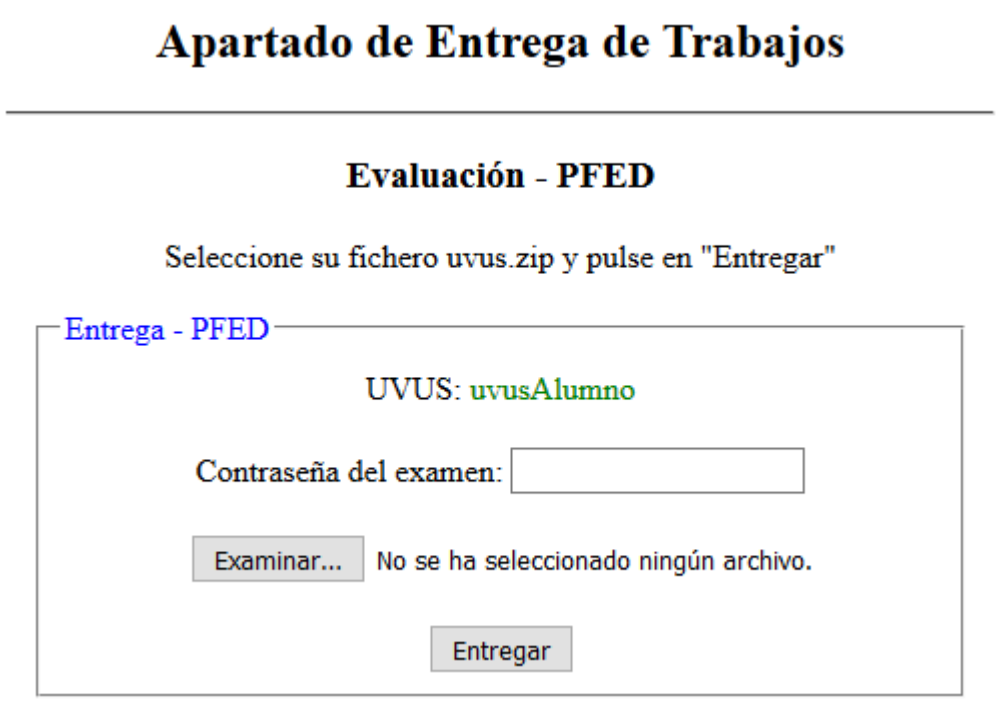

Figura 4.4. Interfaz de Alumno Previa a la Entrega

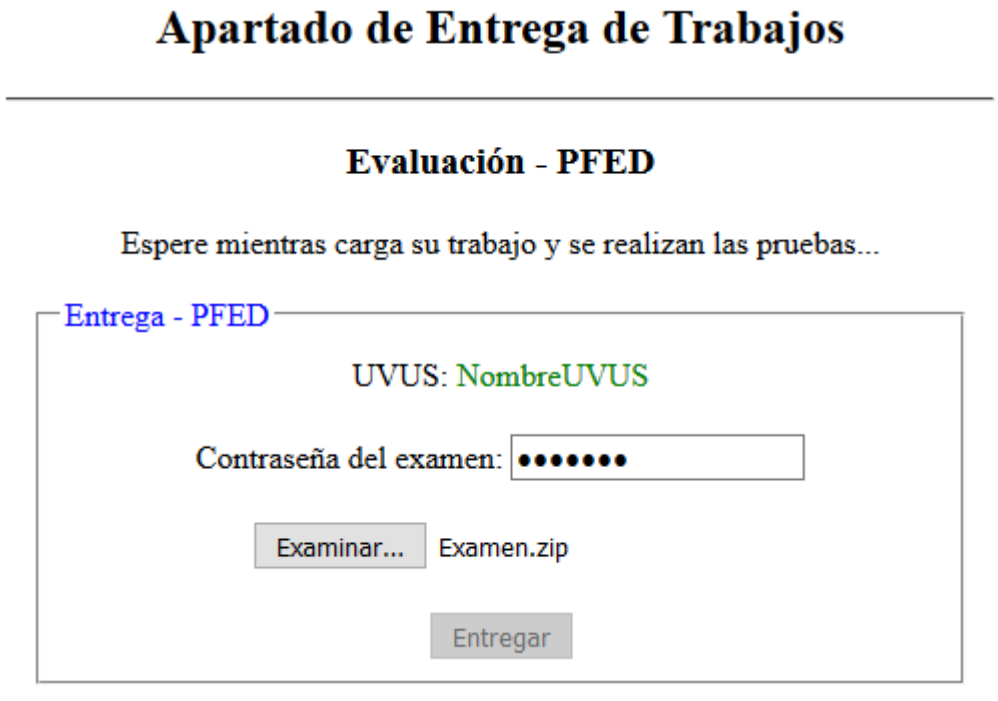

Figura 4.5. Interfaz del Alumno Durante la Entrega

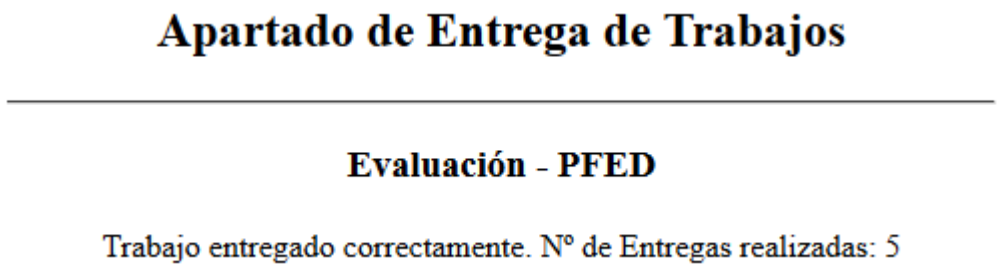

Figura 4.6. Interfaz del Alumno de Información tras la Entrega

El diseño de esta interfaz, se compone así de los siguientes elementos:

- Elementos libres. Aquellos que conforman la parte informativa de la interfaz.
    - Dos encabezados: etiquetas *H2 y H3* en HTML. Se emplea para dar nombre a la herramienta e informar al alumno del trabajo que está entregando.
    - Un elemento de párrafo: etiqueta *P* en HTML. Usado para mostrar las instrucciones de entrega al alumno.
    - Elementos en línea: etiqueta *SPAN* en HTML. Se emplean para poder generar en una misma línea dos elementos independientes, de manera que tengan diferentes características de estilo. De esta manera, ciertos campos en una línea pueden ser enfatizados con un estilo que no es aplicado al resto de la línea.
    - Un elemento separador: etiqueta *HR* en HTML. Nos permite dar énfasis al título de la herramienta, para que el alumno sea consciente del trabajo que está entregando mediante ella.
- **Elementos Agrupados**. Componen la parte organizativa de la interfaz, así como aquellos elementos con los que se puede interactuar.
    - Elemento de agrupación: etiqueta *FIELDSET* en HTML. Con él, podemos agrupar un conjunto de elementos, mostrando un nombre para dicho grupo. De esta manera, conseguimos crear el efecto panel de control, reconociendo como controles los elementos aquí recogidos.
    - Elemento formulario: etiqueta *FORM* en HTML. De uso estandarizado en tecnología web para realizar una nueva petición a la página web enviando datos contenidos en elementos anidados en él.
    - Elementos de entrada: etiqueta *INPUT* en HTML. Empleado para permitir al alumno insertar la clave de entrega, y seleccionar el fichero que debe entregar.

Cuando durante la corrección del trabajo de un alumno se produce algún error, este aparece descrito en el elemento donde, en la Figura 4.6, aparece el mensaje de entrega correcta, bajo el título del trabajo.

3.5.3 Otras Interfaces

Por otro lado, hay algunos errors que disfrutan de sus propias interfaces, dado que son situaciones especiales y estáticas. Es el caso de los errores de peticiones GET, que no están permitidas, o peticiones POST incorrectamente formuladas. Para estos casos, existen ficheros HTML especialmente diseñados para informar sobre dichos errores y sus motivos.

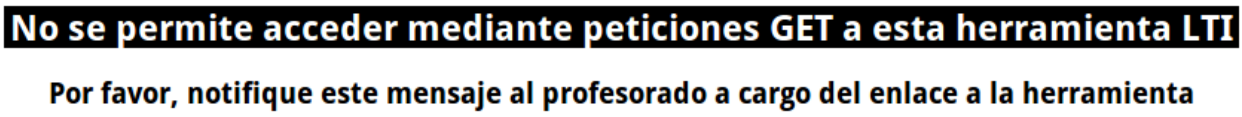

Figura 4.7. Error Petición GET

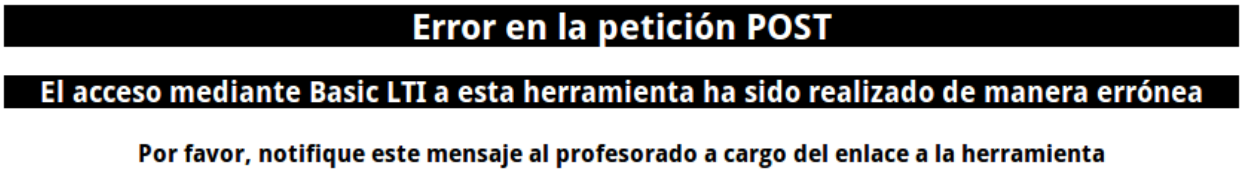

Figura 4.8. Error Petición POST Incorrecta

En ambos casos, se pide a la persona que esté intentando acceder a la herramienta que se notifique el error al profesor, que debe solucionar la configuración de lanzamiento de la herramienta, o bien escalarlo al administrador del LMS en caso de que no pueda ser resuelto por él mismo.

Conclusiones

Una vez completado este bloque, podemos decir que hemos terminado la elaboración de la herramienta que se planteaba como objetivo en este proyecto, empleando la especificación de LTI para integrarla en el LMS de nuestra elección.

Podemos pasar, por tanto, a realizar las pruebas necesarias para afirmar que esta herramienta cumple con los requisitos especificados en la fase de diseño, y así validar el trabajo llevado a cabo.

BLOQUE 4. PRUEBAS DE EJECUCIÓN

En este bloque, mostraremos las distintas pruebas de uso que se han llevado a cabo para asegurar su funcionamiento.

Pruebas de Ejecución

A la hora de hablar de las pruebas de ejecución de esta herramienta, debemos tener en cuenta las distintas partes de las que se compone, a saber, la herramienta de corrección, y la conexión LTI.

Respecto a la herramienta de corrección, se han realizado dos pruebas satisfactorias con esta parte:

- **9 Junio 2015**. En esta fecha, se realizó la defensa de las *PFED* y *PFPOO* con los alumnos de 1º GITT de la asignatura FP2 de este año. Se desplegó para esta ocasión, dos veces nuestra herramienta (no en su versión final) en el servidor del profesor, cada una de ellas configurada para uno de los trabajos.

  El resultado de la herramienta en el caso de la *PFED* fue satisfactorio, concluyendo que la parte realizada hasta el momento era correcta, esto es, el funcionamiento de la herramienta sin la conexión LTI con la EV, sino usada como una aplicación web normal, era correcto. Sin embargo, aparecieron errores puntuales de almacenamiento de las notas de los alumnos, que serían corregidos para la versión posterior de la herramienta.

  El resultado sin embargo de la *PFPOO* no fue satisfactorio, ya que aunque la herramienta funcionaba correctamente, un error en los scripts empleados para corregir este trabajo provocó que no evaluase correctamente a los alumnos. Esto se tuvo en cuenta para corregir los scripts de corrección de esta prueba, de cara a la siguiente entrega.

- **9 Septiembre 2015**. En esta fecha correspondiente a la segunda convocatoria del curso 2014/2015, se volvió a realizar un prueba de la herramienta, a la que se habían realizado diferentes modificaciones, a saber:
  - Correcciones en los scripts de corrección
  - Correcciones en el uso de la librería Apache POI para almacenar los resultados
  - Cambio en la arquitectura, que había provocado la desaparición de algunas clases precompiladas java, y la simplificación del esquema general de la aplicación, desempeñando no obstante el mismo funcionamiento.

  De nuevo, se desplegó dos veces la herramienta en el servidor, una vez para cada prueba a evaluar. En esta ocasión, el resultado de ambas pruebas fue satisfactorio, concluyendo que todo el apartado externo a LTI era correcto, incluyendo los scripts de corrección empleados.

  Queda como caso aislado de esta ocasión, el de un alumno que producía un (supuesto) bucle infinito dentro de su código. De esta manera, la herramienta, a la hora de corregir su trabajo empleando los scripts, no terminaba dicha operación nunca, y quedaba bloqueada a la espera de un resultado que no llegaba. Se recomienda por tanto, la modificación por tanto de dicho script para, después de un tiempo prudencial acorde a la ejecución del trabajo del alumno, matar el proceso si este no devuelve el resultado esperado.

Dado que no se estaba empleando con LTI, la herramienta tenía la capacidad de dejar al alumno escribir su UVUS para identificar al alumno. Esta característica, junto con el libre acceso a los apartados *Manager.jsp* y *Student.jsp* de manera libre sin pasar por el servlet, sería eliminada posteriormente para su completa implementación con LTI.

Inicialmente, en la prueba del 9 de Junio, la herramienta todavía no tenía LTI implementado, por lo que no fue posible llevar a cabo la prueba incluyendo LTI. En la siguiente sin embargo, LTI estaba listo para funcionar por parte de la herramienta.

Si se expidiese a los administradores de la Enseñanza Virtual una petición, para que habilitasen el uso de herramientas LTI del dominio *Teodosio.us.es*, correspondiente al ordenador del profesor tutor de este proyecto, equipo donde se desplegaban las aplicaciones y por tanto, ocupado el rol de Tool Provider, la herramienta podría lanzarse empleando LTI desde la EV. En la página web de BlackBoard se puede encontrar la documentación para el producto BlackBoard Learn 9.1 SP8, donde indica como realizar esta configuración [44] [45] [46]. Dicho esto, la versión final emplea LTI, por lo que no es posible emplearla si no es con dicho protocolo.

Hemos realizado un Tool Consumer que emule a la plataforma, para comprobar que, llegado el momento y si se tuviese habilitada dicha característica en la EV, la comunicación sería correcta y por tanto también nuestra implementación del Tool Provider. Es decir, una comprobación de la versión final de la herramienta.

Para ello, hemos creado una sencilla web que permite lanzar la herramienta empleando los siguientes valores en cada caso:

Nombre del Usuario	Nombre Alumno	Nombre Profesor
UVUS	uvusAlumno	uvusProfesor
Rol	Student	Instructor
Identificador de Contexto	identificadorRecurso1	identificadorRecurso2

Tabla 4.1. Parámetros de Lanzamiento del Emulador de TC

Siendo la interfaz para llevar a cabo el acceso a la herramienta la mostrada en la Figura 4.9.

Figura 4.9. Interfaz del Emulador de Tool Consumer

Al acceder a cualquiera de estos enlaces pulsando sobre las imágenes o los textos inferiores, estaremos invocando a un fichero *jsp* llamado *TC_Student.jsp* y *TC_Teacher.jsp* según proceda, donde el funcionamiento es similar en cada uno de ellos, pero empleando los diferentes parámetros especificados en la tabla 4.1.

4.1.1 Ficheros TC_Rol.jsp

Estos ficheros que acabamos de mencionar, emulan por tanto el funcionamiento de un Tool Consumer como es la Enseñanza Virtual en nuestro caso. Para ello, deben realizar las siguientes tareas:

- Representar a un usuario del LMS

- Preparar un paquete de parámetros de la aplicación
- Firmar estos parámetros con OAuth
- Enviar estos parámetros al usuario, junto con un fragmento JavaScript que fuerce un mensaje POST automático al Tool Provider.

Pero, ¿cómo se hace?

1. Representar a un usuario del LMS.

Para realizar esto, se han generado los dos ficheros TC_Rol.jsp, de manera que su única variación interna no es en funcionamiento, sino en los valores que almacenan dentro de ellos para representar a un usuario. De esta manera, acceder en nuestro servidor a uno de estos ficheros equivale a estar haciendo uso de un enlace LTI con un usuario concreto, que esta fijado dentro de dicho fichero.

2. Preparar los parámetros de la aplicación

Parte de estos parámetros, son preparados por la librería BLTI al construir el mensaje. Otros, como el usuario, tendrá que prepararlos el TC/Emulador TC. Para realizar este proceso, se debe contruir un objeto de la clase *BLTIMessage*, que representará al mensaje creado, y que es construido asociándole la clave que debe emplear el TC de la pareja clave/secreto.

```
BLTIMessage msg = new BLTIMessage( TC_Key );
```

Una vez realizado esto, debemos establecer el resto de valores, modificando el objeto *BLTIMessage*.

```
msg.getResourceLink().setId( TC_ResourceID );
msg.getUser().setId( UVUS );
msg.getUser().setFullName( name );
msg.getUser().addRole( new Role( "Instructor" ) );
```

En el código señalado, se puede observar como se configuran en los ficheros *TC_Rol.jsp* estos parámetros, con la salvedad del último de ellos, correspondiente al fichero *TC_Teacher.jsp* para ser exactos.

3. Firmar el mensaje con OAuth

Una vez tenemos le mensaje listo, debemos preparar el lanzamiento de la aplicación LTI. Para ello, empleando la clase *BLTIConsumer*, elaboraremos la petición que va a ser llevada a cabo, y la firmaremos.

```
BLTIConsumer consumer = new BLTIConsumer( "POST",
"http://teodosio.us.es:8080/ServletPFED/LTIApp", msg);

consumer.sign( TC_Shared_Secret );
```

De esta forma, ya disponemos del mensaje firmado, listo para ser enviado al Tool Provider, en este caso, nuestra herramienta LTI desplegada para el trabajo de la *PFED*.

4. Enviar parámetros y generar auto-POST.

Para cumplir con este envío, necesitaremos generar de manera dinámica un formulario con los parámetros de nuestro mensaje, de manera que estos se envíen al hacer la petición POST. Para ello, generaremos mediante el JSP una serie de elementos *input* con valor del atributo *name* el nombre de cada parámetro, y *value* el valor de dicho parámetro. Estos elementos deben estar dentro del formulario que se va a enviar.

```
List<Map.Entry<String,String>> launchParams2 = consumer.getParameters();
for (Map.Entry<String, String> entry : launchParams2)
{
  out.println("<input type=\"hidden\" name=\"" + entry.getKey()
              + "\" value=\"" + entry.getValue() + "\">");
}
```

Mientras que para realizar el auto-POST, vincularemos a este fichero *TC_Rol.jsp* un fichero JavaScript

llamado *TC.js*, donde una vez cargado por el usuario el fichero, realizaremos el envío.

```
window.onload = function()
{
  window.document.forms[0].submit();
};
```

Conclusiones

Podemos extraer por tanto de este bloque, tanto por las pruebas de campo realizadas con los alumnos, como por las posteriores pruebas sobre la conexión LTI, que la herramienta funciona correctamente en todos sus aspectos, y que por tanto es capaz de llevar a cabo su cometido, controlando de manera correcta los posibles errores asociados a su ejecución y notificándolos para su conocimiento al usuario en caso necesario.

BLOQUE 5. CONCLUSIONES Y BIBLIOGRAFÍA

Finalmente, en este bloque analizaremos la aportación de este trabajo, tanto en el campo de la corrección automática como en el uso de LTI para la ayuda a la docencia, así como algunas de las posibles líneas futuras de investigación o desarrollo.

Conclusión

A partir del estudio del estado del arte podemos afirmar, que la automatización de la corrección ha supuesto una apuesta inteligente por parte del personal docente. Ampliando la capacidad de corrección mediante software, que permite validad unos requisitos mínimos de funcionamiento en el trabajo del alumno, se consigue:

- Mayor **objetividad** en la corrección
- **Ahorro de tiempo** en el proceso

Siendo este segundo parámetro tan importante por su propio significado, como porque habilita la posibilidad de dedicar ese tiempo a la valoración de otros aspectos también importantes en el desempeño de las tareas por parte del alumno si así se desea.

Hasta ahora, el Dpto. de Ingeniería Telemática de la Universidad de Sevilla ha promovido y llevado a cabo esta automatización, enfocada en el desarrollo de software específico para esta tarea. La aportación que difirie sin embargo en este proyecto respecto a lo hecho hasta ahora es la elección de tecnología web, que permite:

- La **independencia de la plataforma de entrega** para el alumno
- **Facilidad de actualización y mantenimiento**, al ser un servicio centralizado

Estas características, se ven complementadas por las aportaciones de nuestra implementación concreta. Con la arquitectura planteada, hemos logrado:

- **Simplicidad de uso**, al tener interfaces escuetas y con pocos elementos interactivos
- **Control remoto,** al poder controlar el profesorado desde su propia interfaz el comportamiento de la herramienta
- **Familiaridad para el alumno,** al poder acceder a la herramieta desde dentro de su plataforma de gestión de aprendizaje de uso diario
- **Multidespliegue,** en el que la aplicación puede desplegarse varias veces para distintos trabajos sobre el mismo servidor, empleando la misma estructura de directorios para ambos

Además, gracias a la implementación de LTI en la herramienta, podemos beneficiarnos de todas sus ventajas, que surgen a partir de la **Integración con la plataforma de Enseñanza Virtual:**

- **Identificar** inequívocamente **al usuario**
- **Conocer su rol** en la comunidad educativa
- Conocer cualquier otro dato necesario o útil para nuestra aplicación

Estas posiblidades abren a su vez una infinidad de combinaciones de comportamiento, personalización y accesibilidad a la hora de emplear una herramienta que utiliza LTI. Posibilidades, que se emplían con cada versión de LTI, siendo estas características mencionadas representativas de *Basic LTI* o *LTI 1.0.*

Líneas Futuras

Una vez vistas las capacidades de las que dispone LTI y nuestra aplicación, podemos proponer las siguientes líneas de mejoras, siendo estas líneas las más inmediatas, ya que con las nuevas versiones de LTI las posibilidades crecen exponencialmente.

Basadas en la aplicación web:

- **Implementación de control de plagio**, para analizar si los alumnos han realizado el trabajo legítimamente.
- **Implementación de entrega sin corrección**, para recibir los trabajos que no necesiten o puedan ser corregidos automáticamente, pero consiguiendo así la disponibilidad inmediata en el equipo del profesor en lugar de la EV.
- **Perfeccionamiento de los scripts de corrección,** o bien de la herramienta, para que cuando se evalua un trabajo, si este se queda bloqueado, no devolviendo un ningún valor en su corrección (bien por un bucle infinito en el código del alumno, o por operaciones de larga duración que no deberían llevarse a cabo), se mate el proceso y se de por incorrecto el apartado.

Basadas en LTI, y por tanto dependientes de las versiones de LTI del TC y TP:

- **Vinculación de la herramienta LTI** con la asignatura FP2, dentro de la Enseñanza Virtual.
- Eliminación del fichero Excel de notas para la Enseñanza Virtual, **devolviendo** directamente a la misma **mediante LTI 1.1 la nota del alumno** en el trabajo realizado.

Bibliografía

[1] IMS Global Learning Consortium, «IMS Global Developer - LTI,» [En línea]. Available: http://developers.imsglobal.org/. [Último acceso: 5 Septiembre 2015].

[2] R. K. Ellis, «Field Guide to Learning Management Systems,» [En línea]. Available: http://www.astd.org/~/media/Files/Publications/LMS_fieldguide_20091. [Último acceso: 5 Septiembre 2015].

[3] U. d. Sevilla, «Enseñanza Virtual,» [En línea]. Available: https://ev.us.es. [Último acceso: 5 Septiembre 2015].

[4] U. d. Sevilla, «Universidad de Sevilla,» [En línea]. Available: http://www.us.es/. [Último acceso: 5 Septiembre 2015].

[5] BlackBoard, "BlackBoard Learn," [Online]. Available: http://www.blackboard.com/learning-management-system/blackboard-learn.aspx. [Accessed 10 Septiembre 2015].

[6] A. J. Sierra, T. Ariza and F. J. Fernández, "Establishment EHEA for telecommunication technologies engineering degree Global Engineering Education Conference," Marrakech, 2012.

[7] A. J. Sierra, T. Ariza and F. J. Fernández, "PBL in Programming Subjects at Engineering," *Bulletin of the IEEE Technical Committee on Learning Technology*, vol. 15, no. 2, pp. 1-4, 2013.

[8] Apache Software Foundation, "Wookie," [Online]. Available: http://wookie.apache.org/. [Accessed 10 Septiembre 2015].

[9] Intelligent & Cooperative Systems Group (GSIC), "GLUE!," [Online]. Available: http://www.gsic.uva.es/glue/. [Accessed 10 Septiembre 2015].

[10] C. Alario-Hoyos, "Comparison of the main alternatives to the integration of external tools in different platforms," [Online]. Available: http://www.researchgate.net/publication/233734142_Comparison_of_the_main_alternatives_to_the_integration_of_external_tools_in_different_platforms. [Accessed 10 Septiembre 2015].

[11] W3C, «W3C Widgets,» 27 Noviembre 2012. [En línea]. Available: http://www.w3.org/TR/widgets/. [Último acceso: 10 Septiembre 2015].

[12] OpenSocial Foundation, «OpenSocial,» [En línea]. Available: http://opensocial.org/. [Último acceso: 10 Septiembre 2015].

[13] W3C, «REST,» [En línea]. Available: http://www.w3.org/2001/sw/wiki/REST. [Último acceso: 10 Septiembre 2015].

[14] «IMS Global Learning Consortium,» [En línea]. Available: http://imsglobal.org/. [Último acceso: 5 Septiembre 2015].

[15] IMS Global Learning Consortium, «Learning Tools Interoperability,» [En línea]. Available: http://www.imsglobal.org/lti/. [Último acceso: 5 Septiembre 2015].

[16] IMS Global Learning Consortium, "Basic LTI Implementation Guide Version 1.0 Final," 17 Mayo 2010. [Online]. Available: http://www.imsglobal.org/lti/blti/bltiv1p0/ltiBLTIimgv1p0.html. [Accessed 5 Septiembre 2015].

[17] IMS Global Learning Consortium, «Comparison Chart of LTI Versions,» [En línea]. Available: http://developers.imsglobal.org/tutorials.html. [Último acceso: 5 Septiembre 2015].

[18] IMS Global Learning Consortium, «IMS Interoperability Conformance Certification Status,» [En línea]. Available: http://www.imsglobal.org/cc/statuschart.cfm. [Último acceso: 5 Septiembre 2015].

[19] IMS Global Learning Consortium, «LTI 1.0 Architecture,» [En línea]. Available: http://developers.imsglobal.org/tutorials.html. [Último acceso: 5 Septiembre 2015].

[20] Edu Apps, "Identity assertion is a one-way "handshake"," [Online]. Available: https://www.edu-apps.org/code.html. [Accessed 10 Septiembre 2015].

[21] G. Kroner, "BLTI-Sandwich," 7 Abril 2011. [Online]. Available: http://projects.oscelot.org/gf/project/blti-sandwich/. [Accessed 5 Septiembre 2015].

[22] P. Gray, "LTI Utilities," 6 Junio 2014. [Online]. Available: https://github.com/IMSGlobal/basiclti-util-java. [Accessed 5 Septiembre 2015].

[23] A. Miller, «LtiLibrary 1.5,» 1 Marzo 2015. [En línea]. Available: http://andyfmiller.com/2015/03/01/ltilibrary-1-5/. [Último acceso: 5 Septiembre 2015].

[24] Instructure, «IMS LTI,» 6 Marzo 2012. [En línea]. Available: https://github.com/instructure/ims-lti. [Último acceso: 6 Septiembre 2015].

[25] «Instructure,» [En línea]. Available: http://www.instructure.com/. [Último acceso: 6 Septiembre 2015].

[26] «BSD License,» [En línea]. Available: http://www.linfo.org/bsdlicense.html. [Último acceso: 6

Septiembre 2015].

[27] «Apache License,» [En línea]. Available: http://www.apache.org/licenses/LICENSE-2.0. [Último acceso: 6 Septiembre 2015].

[28] «MIT License,» [En línea]. Available: https://opensource.org/licenses/MIT. [Último acceso: 6 Septiembre 2015].

[29] Microsoft Corporation, ".NET Framework," [Online]. Available: https://msdn.microsoft.com/en-us/vstudio/aa496123.aspx. [Accessed 5 Septiembre 2015].

[30] Internet Engineering Task Force (IETF), Abril 2010. [Online]. Available: http://tools.ietf.org/html/rfc5849. [Accessed 10 Septiembre 2015].

[31] A. Miller, "Does LTI use OAuth?," 10 Febrero 2013. [Online]. Available: http://andyfmiller.com/2013/02/10/does-lti-use-oauth/. [Accessed 10 Septiembre 2015].

[32] U. d. Sevilla, «Escuela Técnica Superior de Ingeniería,» [En línea]. Available: http://www.etsi.us.es/. [Último acceso: 5 Septiembre 2015].

[33] A. J. Sierra, T. Ariza, F. J. Fernández y G. Madinabeitia, «TVSP: A Tool for Validation Software Projects in programming labs Global Engineering Education Conference (EDUCON),» Marrakech, 2012.

[34] A. J. Sierra, T. Ariza, F. J. Fernández y G. Madinabeitia, «Tool for Validation Software Projects in Programming Labs,» *International Journal of Engineering Pedagogy (iJEP)*, vol. 2, nº 2, 2012.

[35] A. Martínez Gotor, «HERRAMIENTAS DE EVALUACIÓN AUTOMÁTICA MEDIANTE BASES DE DATOS NoSQL APLICANDO MÉTRICA v3,» 2014. [En línea]. Available: http://bibing.us.es/proyectos/abreproy/90009.

[36] R. S. Pressman, de *Ingeniería del software. Un enfoque práctico.*, Mc Graw Hill, 1989, pp. Capítulos 4,5,6,7.

[37] Oracle, "Java Servlet Technology Overview," [Online]. Available: http://www.oracle.com/technetwork/java/overview-137084.html. [Accessed 7 Septiembre 2015].

[38] w3schools, "AJAX Description," [Online]. Available: http://www.w3schools.com/ajax/default.asp. [Accessed 7 Septiembre 2015].

[39] "OAuth," [Online]. Available: http://oauth.net/about/. [Accessed 7 Septiembre 2015].

[40] Apache Software Foundation, "Apache POI, Java API for Microsoft Documents," [Online]. Available: https://poi.apache.org/. [Accessed 7 Septiembre 2015].

[41] Apache Software Foundation, "Commons FileUpload," [Online]. Available: https://commons.apache.org/proper/commons-fileupload/. [Accessed 7 Septiembre 2015].

[42] Apache Software Foundation, "Commons IO," [Online]. Available: https://commons.apache.org/proper/commons-io/. [Accessed 7 Septiembre 2015].

[43] "Unzip Command," [Online]. Available: http://linux.about.com/od/commands/l/blcmdl1_unzip.htm.

[Accessed 8 Septiembre 2015].

[44] BlackBoard, "BlackBoard Learn 9.1. SP8 Help," [Online]. Available: https://en-us.help.blackboard.com/Learn/9.1_Older_Versions/9.1_SP_8. [Accessed 10 Septiembre 2015].

[45] BlackBoard, "Proveedores de herramientas LTI básico - Administradores," 2012. [Online]. Available: http://help-archives.blackboard.com/Blackboard-Learn/9.1/SP08/ES-ES/Admin/index_Left.htm#CSHID=_admin_app_system%2Fadmin_app_basic_lti_tool_providers.htm|StartTopic=Content%2F_admin_app_system%2Fadmin_app_basic_lti_tool_providers.htm|SkinName=Bb_i18n. [Accessed 10 Septiembre 2015].

[46] BlackBoard, "Creación de contenido - Profesores," 2012. [Online]. Available: http://help-archives.blackboard.com/Blackboard-Learn/9.1/SP08/ES-ES/Instructor/index_Left.htm#CSHID=_instructor_content%2Finstructor_content_build_content.htm|StartTopic=Content%2F_instructor_content%2Finstructor_content_build_content.htm|SkinName=Bb_i18n. [Accessed 10 Septiembre 2015].

BLOQUE 6. ANEXOS

Anexo A. Ficheros Excel de Evaluación

Como objetivo de la aplicación LTI, se encuentra la evaluación del trabajo realizado por los alumnos. Esta evaluación es almacenada en dos ficheros Excel, de manera que se pueda disponer de esta información de una manera eficiente y completa.

En un primer fichero, del que se habla durante este proyecto como fichero Excel del profesor, se almacena un desglose exhaustivo de la evaluación del alumno. Las columnas son: uvus, n columnas de los *n* apartados del trabajo, la nota final del trabajo, y el número de entregas que ha realizado el alumno mientras la aplicación ha estado activada.

En un segundo fichero, que hemos mencionado como fichero Excel de la Enseñanza Virtual, se almacenan exclusivamente dos columnas: UVUS del alumno, y nota final del trabajo.

6.1.1 Carga y Modificación de los Ficheros Excel

A la hora de trabajar con los ficheros Excel, la aplicación funciona de una manera concreta, cargando los ficheros en memoria en ciertos instantes, y volcando las modificaciones que sufren a disco en otros diferentes. Es por esto que necesitamos saber en qué estado se encuentra nuestra información en un momento dado del uso de la aplicación, para asegurarnos de no sufrir ningún incidente al respecto.

Primeramente, cuando el profesor despliega la aplicación, estos ficheros no son cargados en la memoria del servidor. Por tanto, en este momento no se trabaja con los ficheros. Una vez el profesor habilita la herramienta mediante el botón destinado a ello en la interfaz de control, los ficheros son cargados a memoria. A partir de este instante, se tiene una copia virtual de los ficheros, cargada en memoria.

Es allí donde serán modificados, cada vez que un alumno corrija su trabajo. Al finalizar la corrección del mismo, el alumno actualiza este fichero en memoria actualizando su correspondiente registro con calificaciones (o creándolo en caso de que sea necesario), y una vez terminada esta actualización, lleva el fichero a disco. Es importante destacar, que **en este proceso el fichero se lleva a disco con todo el contenido almacenado hasta el momento**, por lo que, si en algún momento no ha sido posible para algún usuario llevar dicho fichero a disco, todos los usuarios posteriores llevarán ese contenido al fichero. Esto es así para ambos ficheros Excel con los que trabaja la aplicación.

A partir de este momento, **cada vez que el profesor pulsa sobre el botón de habilitar o deshabilitar entrega, se salva o se carga el contenido de los ficheros de disco a memoria, o viceversa** respectivamente. Este comportamiento nos asegura que: en caso de que el último alumno no haya podido escribir sus cambios a disco, sus cambios se almacenan correctamente al deshabilitar el profesor la herramienta; y que no es necesario reiniciar el servidor donde se encuentra la aplicación para cambiar los ficheros de notas empleados, bastando con deshabilitar la entrega, cambiar los ficheros y habilitarla de nuevo.

Por último, y como único contra que se encuentra en este método, es que el profesor debe tener cuidado de no hacer un ciclo de deshabilitación-habilitación para no provocar un error de acceso al fichero en disco, al intentar ser este accedido por ambos a la vez.

6.1.2 Esquema de los Ficheros Excel

Como ya hemos mencionado, el hecho de que la evaluación se desglose en dos ficheros Excel es para facilitar su posterior uso por parte del profesor. Explicaremos ahora, por tanto, el esquema y su motivación para cada uno de los ficheros empleados.

El profesor debe recordar, que se trabajará siempre con la primera hoja del fichero Excel correspondiente, mientras que el resto serán ignoradas. No es necesario que la hoja tenga ningún nombre concreto ni especial.

6.1.2.1 Excel del Profesor

El objetivo de este fichero Excel, es el de proporcionar al profesor un desglose completo de la evaluación efectuada por el alumno. Sus datos son, por tanto:

Una columna inicial para el **UVUS** (identificador) del alumno

n columnas correspondientes a los diferentes **apartados** a corregir en el trabajo

Una columna para la **nota final** del alumno

Una columna para el **número de entregas** que efectúa el alumno

Estas columnas aparecerán en el fichero en el mismo orden en el que han sido aquí redactadas. El profesor tiene la oportunidad de introducir una primera fila de encabezado para dejar claros estos valores, ya que la aplicación no tratará estos campos. Sin embargo, debe tener en cuenta que en ningún caso el valor introducido en la primera columna de esta fila, debe coincidir con el posible identificador (UVUS) de un usuario (alumno), ya que podría producir un conflicto en la aplicación.

	A	B	C	D	E	F	G
1	UVUS	Apartado 0	Apartado 1	Apartado 2	Apartado 3	Nota Final	Nº Entregas
2							

Figura 6.1: Encabezado del fichero Excel del Profesor.

Por último, el profesor debe recordar que harán falta o se usarán tantas columnas de apartados, como ficheros ejecutables se encuentren en el directorio *./Server/Scripts/titulo_de_su_trabajo/* (creado por usted tras la descompresión del instalador proporcionado, y rellenado con sus herramientas de corrección) a la hora de realizar el despliegue de la aplicación.

La herramienta trabaja, como se podrá deducir del Anexo B. Scripts de Corrección, otorgando una calificación binaria (1 aprobado, 0 suspenso) a cada apartado corregido. De esta manera, la aparición del valor verdadero indica apartado superado, mientras que el valor falso indica lo contrario. En la Figura 6.2 se puede ver un ejemplo de un alumno que supera la PFED con todos los apartados realizados correctamente.

	A	B	C	D	E	F	G
1	UVUS	Apartado 0	Apartado 1	Apartado 2	Apartado 3	Nota Final	Nº Entregas
2	NombreUVUS	VERDADERO	VERDADERO	VERDADERO	VERDADERO	10	1

Figura 6.2: Alumno NombreUVUS supera todas las pruebas de la PFED

6.1.2.2 Excel de la Enseñanza Virtual

Para el profesorado ajeno a la Universidad de Sevilla (US), debemos puntualizar en este punto lo que es el UVUS. Como se ha mencionado anteriormente, es el identificador único para cada alumno de la US, (sin embargo, en el fichero anterior no era un valor crítico, aunque debía ser único, por no tener la misma finalidad que en este), y son las siglas de **Usuario Virtual de la Universidad de Sevilla**.

El objetivo de este fichero, es el de facilitar la actualización de las notas correspondientes a los alumnos en la Enseñanza Virtual de la US, es decir, el LMS empleado en la US. Por ello, el fichero es un fichero exportado directamente a través de este LMS, manteniéndose con los valores originales que poseen los campos en dicho LMS:

El **nombre de la hoja** (única)

El nombre de la columna del **UVUS**

El nombre de la columna donde se almacenará la **nota del trabajo**

Estas columnas (2º y 3º valor) aparecerán en este orden, igual que en el fichero Excel anterior, tanto por necesidad de la herramienta, como por exportación desde el LMS, ya que el UVUS del alumno es usado como clave primaria para la tabla de notas de los alumnos, y como tal aparece en el primer campo.

De esta manera, la aplicación LTI actualizará los datos en esta plantilla, que al terminar la evaluación del trabajo, el profesor puede cargar la misma plantilla descargada del LMS con los datos actualizados, pudiendo actualizar sin mayores complicaciones la nota de los alumnos sin tener que editarlas manualmente en la plataforma.

Anexo B. Manual de Instalación

A la hora de realizar la instalación de esta aplicación, hará falta realizar una serie de operaciones tanto sobre el servidor donde se aloje la aplicación, como sobre la plataforma donde se pretenda publicar.

Trataremos como desplegar la aplicación, recordando al lector que posteriormente debe vincularla en su LMS. Anteriormente se ha indicado en las referencias, el manual para realizar esto en el LMS empleado actualmente en la US.

6.2.1 Despliegue de la Aplicación en el Servidor

Para desplegar la aplicación en un servidor, necesitaremos la siguiente infraestructura:

Servidor Web compatible con Java (Ej: Tomcat)

Fichero comprimido en el que se proporciona la aplicación (LTI_App.zip)

El profesor que desee desplegar la aplicación en un servidor, deberá copiar este fichero a un directorio de su elección en dicho servidor. Una vez allí, deberá extraerse el contenido de dicho fichero, apareciendo un directorio llamado de igual manera que el fichero ZIP antes mencionado: *LTI_App*. En caso de que se esté desplegando esta aplicación en un servidor en Linux, esta descompresión se puede realizar mediante un terminal, situándose en el directorio donde se encuentra dicho fichero .ZIP, y ejecutando el comando:

```
unzip LTI_App.zip
```

Una vez extraídos los ficheros de este ZIP, encontraremos un fichero llamado *LTI_Deploy.sh*, que nos presentará un menú con las diferentes acciones necesarias para el despliegue, que se encuentra detallado en la Figura 6.3 más adelante. Asimismo, encontraremos el fichero *LTI_Deploy_Vars.txt*, un fichero de texto plano donde el profesor deberá configurar una serie de parámetros para el despliegue de la aplicación en su servidor.

Pasamos a detallar a continuación los parámetros de este fichero (*LTI_Deploy_Vars.txt*), sus posibles valores y cómo deben ser configurados. Es importante **respetar la estructura de los valores** aquí establecidos, esto es, si en este documento se especifica que una ruta a un directorio acaba con una barra "/", el profesor debe introducir su directorio acabado igualmente en dicho carácter.

6.2.1.1 Valores Configurables para el Despliegue

Ruta de despliegue de la aplicación

Su valor será el directorio donde se despliegue la aplicación. Por tanto, este directorio debe existir previamente a la ejecución del script. Normalmente, será una subcarpeta donde el servidor web empleado localiza los directorios de cada una de las webs que sirve.

Referencia: ***deployPath=***

Ejemplo: *deployPath=/var/lib/tomcat7/webapps/aplication/*

Nombre del proyecto a evaluar

Este valor se emplea para asegurar la no existencia de ficheros temporales en el directorio de scripts empleados para la corrección del trabajo, asegurando así el correcto funcionamiento de la aplicación.

Además se emplea en dos ocasiones, a saber: la creación de un directorio con dicho nombre, bajo el directorio correspondiente a cada alumno; y será el título que se muestra en la herramienta durante su uso, tanto por parte del profesorado como del alumno.

Teniendo en cuenta el primero de los usos de este valor, quedan prohibidos los espacios en blanco y los caracteres '/', '\', ':', '?', '*', '<', '>', '|' y ' " '. Asimismo, se desaconseja el uso de acentos y otros signos de puntuación. Una buena práctica sería el uso de siglas, por ejemplo en nuestro caso: *Práctica Final de Estructuras Dinámicas → PFED.*

Referencia: ***projectTitle=***

Ejemplo: *projectTitle=PFED*

Nombre del usuario que ejecuta el servidor

Contine el nombre del usuario que ejecuta el servidor (en nuestro caso, tomcat7). Es empleado para asegurar la propiedad de los ficheros para el servidor, de manera que no existan problemas a la hora de ejecutarlos.

Referencia: ***serverUser=***

Ejemplo: *serverUser=tomcat7*

Clave de entrega

Establece la contraseña que se le especifica como clave necesaria para entregar el trabajo. Esta clave deberá ser proporcionada por el profesor a los alumnos durante la entrega para permitir el uso de la herramienta.

Referencia: ***deliveryPassword=***

Ejemplo: *deliveryPassword=fp218032015*

Inicio de la estructura de directorios

Este valor indica a la aplicación, cual debe ser el inicio de la estructura de directorios sobre el que se crearán todas las carpetas con las que trabaja la aplicación.

Referencia: ***rootPath=***

Ejemplo: *rootPath=/home/usuario/nombre_asignatura/*

Rutas de los ficheros Excel

Por último, estos dos parámetros indican la ruta exacta de los ficheros Excel donde deben almacenarse los resultados de la evaluación.

Referencia 1: ***evExcelPath =***

Referencia 2: ***teacherExcelPath=***

Ejemplo:

evExcelPath=/home/usuario/notas_alumnos/asignatura/trabajo/Plantilla_Ev.xls

teacherExcelPath=/home/usuario/ notas_alumnos/asignatura/trabajo /Plantilla_Profesor.xls

Fórmula de Evaluación

Indica cómo se debe evaluar el trabajo del alumno en función de los apartados. La nota de cada apartado es An, empezando en A0... A(n-1), por ejemplo 8 apartados irían de A0 hasta A7. Pueden emplearse +,-,*,/,(,). Los apartados tomarán el valor 1 en caso de correcto, y 0 en caso de incorrecto al ser evaluados por la herramienta. Se recomienda evitar decimales, escribiendo estos números como fracciones.

Referencia: ***evaluationFormula=***

Ejemplo: *evaluationFormula= "A0*(5/2)*(1+A1+A2+A3)"*

6.2.1.2 Despliegue Mediante el Script Proporcionado

Una vez realizados los pasos anteriores, finalmente emplearemos al script antes mencionado (*LTI_Deploy.sh*) para desplegar la aplicación. Este método está pensado para sistemas UNIX, pero el usuario que desee desplegar la aplicación en Windows puede hacerlo manualmente, o bien realizando un script equivalente en fichero de comandos por lotes (.bat).

Para ejecutar dicho script, nos situaremos en un terminal de Linux, accediendo al directorio donde se encuentra el fichero mediante el comando cd. Una vez allí, ejecutaremos el script (en caso de no poder ejecutarlo, recuerde que debe asignar permisos de ejecución al fichero para el usuario o grupo con el que esté trabajando en el terminal).

```
~$ ./LTI_Deploy.sh
```

Al lanzar el script, se nos presentará un menú desde el que podremos realizar todas las tareas necesarias para el despliegue: la compilación de los ficheros funete, la copia de los ficheros a la carpeta usada por el servidor, y el cambio de la pertenencia de los mismos al usuario que controla el servidor. Una vez haya realizado los tres pasos en el orden en el que aparece en el menú, habrá desplegado la aplicación con éxito. Además, se presenta un cuarto paso, que permite iniciar el servidor, reiniciando tomcat en caso de que esté iniciado, para tener el despliegue inmediato de la aplicación. Este paso está personalizado, de manera que también se modifican las reglas del componente Netfilter del servidor, para permitir el acceso de conexiones entrantes al mismo desde cualquier dirección IP, siempre que sean al puerto 8080, en el que operará nuestro servidor tomcat.

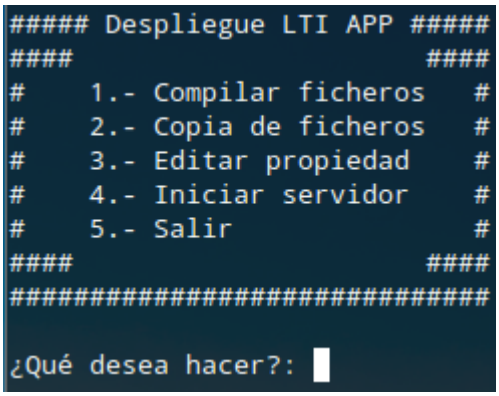

Figura 6.3. Menú del script de despliegue.

Compilación de los ficheros necesarios para el funcionamiento

Al elegir en el menú la primera opción, el script compilará todos los ficheros de la aplicación, a saber: el servlet, y las clases Java existentes.

Copia de los ficheros necesarios para el funcionamiento

Al elegir en el menú la segunda opción, el script copiará todos los ficheros de la aplicación a la carpeta correspondiente de su servidor web, que fue antes configurada en el fichero de variables. Esto omitirá los ficheros de extensión "*.java*", que aunque serán copiados, serán posteriormente eliminados por no ser necesarios, ya que se usan los ficheros compilados "*.class*" en su lugar.

Como medida de seguridad y debido a que el funcionamiento de la aplicación depende del contenido que se encuentra en algunas de sus carpetas, el contenido de la carpeta especificada se borrará como previo paso a la copia de los ficheros, para evitar ficheros residuales anteriores que pudieran generar un comportamiento indeseado en la aplicación.

Cambio de la Pertenencia de los Ficheros

Esta opción, cambia la propiedad de todos los ficheros en dicha carpeta destino, al usuario que usted ha especificado en el fichero *LTI_Deploy_Vars.txt*, el cual debe corresponder con aquel usuario que ejecuta el servidor. De esta manera, aseguramos que el programa servidor es propietario de estos ficheros y no encontraremos problemas a la hora de ejecutar la aplicación.

6.2.2 Despliegue del Emulador de TC

De manera análoga a como se despliega la herramienta LTI, el Emulador de Tool Consumer elaborado dispone de un script y parámetros de despliegue propios. En este caso, el fichero bajo el que se proporciona es denominado LTI_TC_App.zip, por lo que se podrá ejecutar la descompresión del mismo mediante:

```
unzip LTI_TC_App.zip
```

Una vez extraído, nos presentará la misma estructura que el fichero de la herramienta una vez extraido, pero

con la denominación TC en cada nombre. Pasemos por tanto a ver los paráemtros configurables para su despliegue.

6.2.2.1 Valores Configurables para el Despliegue

Ruta de despliegue de la aplicación

Su valor será el directorio donde se despliegue la aplicación. Por tanto, este directorio debe existir previamente a la ejecución del script. Normalmente, será una subcarpeta donde el servidor web empleado localiza los directorios de cada una de las webs que sirve.

Referencia: ***deployPath=***

Ejemplo: *deployPath=/var/lib/tomcat7/webapps/aplication/*

Nombre del usuario que ejecuta el servidor

Contine el nombre del usuario que ejecuta el servidor (en nuestro caso, tomcat7). Es empleado para asegurar la propiedad de los ficheros para el servidor, de manera que no existan problemas a la hora de ejecutarlos.

Referencia: ***serverUser=***

Ejemplo: *serverUser=tomcat7*

6.2.2.2 Despliegue Mediante el Script Proporcionado

En este caso, el fichero del script será *LTI_TC_Deploy.sh*, que nos mostrará el siguiente menú al ser ejecutado en el terminal mediante:

```
~$ ./LTI_TC_Deploy.sh
```

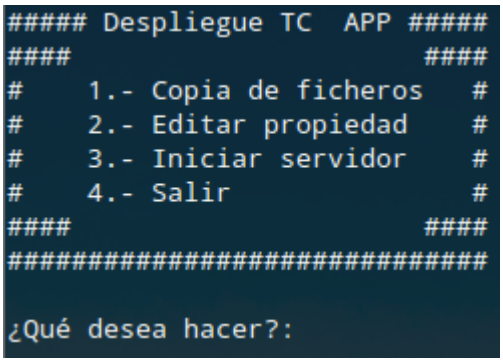

Figura 6.4. Menú del script de despliegue del Emulador de TC

Se puede observar, que el menú es similar al script anterior, con la única diferencia de que en esta ocasión no hay ficheros que deban ser compilados. Por tanto, una vez ejecutemos los distintos pasos en el orden en el que aparecen en el propio menú, habremos desplegado correctamente el emulador de TC.

Anexo C. Permisos

A la hora de desplegar la aplicación LTI, se deben tener en cuenta una serie de permisos del sistema que nos afectarán tanto en el despliegue como en el funcionamiento de la misma.

Vamos a tratar estos permisos de manera que el funcionamiento de la aplicación sea el adecuado, sin centrarnos específicamente en la aplicación de unas políticas de seguridad más concretas para entornos hostiles: esto es, supondremos un entorno de confianza.

6.3.1 Permisos en el Despliegue de la Aplicación

6.3.1.1 Permisos Dependientes del Despliegue Automático

A la hora de desplegar la aplicación, debemos tener en cuenta los siguientes permisos base:

De la carpeta objetivo del despliegue (*deployPath*)

Del usuario que ejecuta el script de despliegue (*LTI_Deploy.sh*)

Del usuario que ejecuta el servidor web

Normalmente, aunque el usuario de Linux sea administrador de su propio equipo, no suele trabajar como super-usuario, ni realiza un proceso completo estando autorizado mediante el comando sudo (y teniendo el nivel de permisos, por tanto, elevado).

Por tanto, el usuario se podría encontrar con el siguiente escenario (caso que nos ocupa):

deployPath: **pertenece a un usuario distinto al que maneja el profesor**, correspondiente al usuario que ejecuta el servidor web (*tomcat7).*

Usuario actual del terminal: *usuarioProfesor*

Usuario del servidor: *tomcat7*

Esto es así, porque la ruta *deployPath* pertenece a la estructura de directorios del servidor web, por lo que es creada por la instalación del servidor, y por tanto pertenece a dicho usuario.

En esta situación, si el usuario sin privilegios administrativos intenta escribir o borrar ficheros de la ruta donde va a ser desplegada la aplicación, es posible que se encuentre con la imposibilidad de hacerlo debido a que la carpeta puede no tener asignados estos permisos ni el grupo adecuado como para garantizar los permisos necesarios al usuario utilizado.

Por tanto, para tratar esta situación se ha trabajado de la siguiente forma:

El script ***LTI_Deploy.sh*** **debe ejecutarse con permisos de administrador** (sudo o directamente como root, según su configuración).

Esto provocará que el nuevo propietario de los ficheros que han sido copiados a su carpeta de despliegue no sea el mismo usuario que controla su servidor web. Por tanto, ahora deben modificarse para pertenecer a dicho usuario. Para ello se encuentra la segunda opción del menú del script, que **adecuará la pertenencia de los ficheros a dicho usuario**.

Por último, destacar que todo lo explicado en este apartado es **igualmente válido para el despliegue del emulador de Tool Consumer** (TC).

6.3.1.2 Permisos Dependientes del Despliegue Manual

Hasta ahora hemos hablado de los permisos a la hora de desplegar la aplicación, mientras quedaba en mano del script de despliegue. Sin embargo, hay permisos que deben ser tenidos en cuenta a la hora de realizar el despliegue manual de la aplicación, es decir, de los ficheros que deben ser desplegados a mano.

Aunque la aplicación es auto-contenida, los ficheros Excel relativos a las notas de las evaluaciones de los alumnos, deben ser colocados manualmente en la carpeta donde se le ha especificado a la aplicación que se van a encontrar.

Siendo esto así, es lógico asumir que los ficheros pertenecerán al usuario normal con el que el profesor emplea el equipo, así como a su grupo. Sin embargo, se debe tener en cuenta que el usuario que accederá a ellos posteriormente será **el que ejecute el servidor web**, de manera que este usuario **debe tener permisos tanto de lectura como de escritura**.

Para ello, se pueden garantizar estos permisos para el tercer grupo ("otros"), o bien para el segundo grupo (grupo propietario) y asegurar la pertenencia del fichero a dicho grupo.

Por ejemplo, la primera de las dos aproximaciones se puede lograr mediante:

```
chmod o=rw ficheroExcel
```

Con este comando, establecemos que **el grupo *otros* tiene permiso tanto de lectura como de escritura sobre el fichero** que se indique (no olvide que debe realizar esta operación sobre los dos ficheros Excel empleados).

Anexo D. Código de la Herramienta

En este anexo se proporciona, siguiendo el orden de la estructura de directorios, el código correspondiente a los diferentes ficheros que componen la herramienta. En el caso de los ficheros *.class* que existen en la herramienta, se muestra de manera análoga el código del fichero *.java* correspondiente.

6.4.1 LTI_Deploy.sh

```
#!/bin/bash
# Álvaro Martín Rodríguez
# Script de despligue de la aplicación LTI

function menu()
{

  clear

  echo "##### Despliegue LTI APP #####"
  echo "####                      ####"
  echo "#    1.- Compilar ficheros   #"
  echo "#    2.- Copia de ficheros   #"
  echo "#    3.- Editar propiedad    #"
  echo "#    4.- Iniciar servidor    #"
  echo "#    5.- Salir               #"
  echo "####                      ####"
  echo "################################"

  echo ''
  echo -n '¿Qué desea hacer?: '
  read O

  case $O in
    1 )
      compila
      ;;
    2 )
      copia
      ;;
    3 )
      propiedad
      ;;
    4 )
      iniciar
      ;;
    5 )
      clear
      exit
      ;;
    * )
      menu
      ;;

  esac

}

function volverMenu()
{

  echo ''
  echo -n '¿Quiere volver al menú? (S/N): '
  read O

  if [ "$O" = "S" ] || [ "$O" = "s" ]; then
    menu
  else
    clear
```

```
    #exit
  fi

}

function compila()
{

  echo ''
  echo -n '¿Quiere compilar los ficheros Java? (S/N): '
  read A

  if [ "$A" = "S" ] || [ "$A" = "s" ]; then

    #Compilamos los ficheros .java
    ROOT="./Servlet/WEB-INF/classes"
    PACKAGE="$ROOT/tfg/lti/"
    CLASSES="$ROOT"

    echo ""
    echo "1.- Compilando clases..."

    #Control
    javac -Xlint -d $CLASSES "${PACKAGE}Control/ErrorControlado.java"

    #ScriptWorker
    javac -Xlint -d $CLASSES -cp $CLASSES \
          "${PACKAGE}/Files/ScriptWorker.java"

    #ExcelWorker
    POI="$ROOT/../lib/poi-3.11-20141221.jar"
    POIXML="$ROOT/../lib/poi-ooxml-3.11-20141221.jar"
    javac -Xlint -d $CLASSES -cp "$CLASSES:$POI:$POIXML" \
          "${PACKAGE}/Files/ExcelWorker.java"

    echo "2.- Compilando servlet..."

    #Servlet
    SERVLETS="/usr/share/tomcat7/lib/servlet-api.jar"
    BLTI="$ROOT/../lib/blti-sandwich.jar"
    javac -Xlint -cp $SERVLETS:$BLTI "${ROOT}/ProjectEvaluation.java"

    echo ""
    echo "Fin."

  else
    echo 'Se ha cancelado la compilación de los ficheros .java'
  fi

  volverMenu

}

function copia()
{

  echo ''
  echo -n '¿Quiere copiar los ficheros a la carpeta? (S/N): '
  read A

  if [ "$A" = "S" ] || [ "$A" = "s" ]; then

    echo ""
    echo "1.- Recargando el fichero LTI_Deploy_Vars..."

    #Carga del fichero de variables
    source ./LTI_Deploy_Vars.txt
```

```
    echo "2.- Carpeta destino: $deployPath"

    if [ ! -d "$deployPath" ]; then

      echo "2.1.- Creando carpeta destino..."
      mkdir $deployPath &>/dev/null

    else

      echo "2.2.- Eliminando contenido previo..."
      rm -r "$deployPath"*  &>/dev/null

    fi

    # Configuramos web.xml a partir de las variables de LTI_Deploy_Vars.txt
    prepararWebXML

    # Copiamos el contenido de Server a deployPath
    echo "3.- Copiando aplicación..."
    cp -R ./Servlet/* $deployPath

    # Borramos los ficheros temporales (File.ext~)
    echo "4.- Eliminando ficheros temporales..."
    rm "$deployPath/Scripts/$projectName/*.sh~" &>/dev/null

    # Borramos los ficheros fuente del despliegue
    echo "5.- Eliminando ficheros innecesarios..."
    find $deployPath -name "*.java" -type f -delete

    echo ""
    echo "Fin."

  else
    echo 'Se ha cancelado la copia de los ficheros'
  fi

  volverMenu

}

function propiedad()
{

  echo ''
  echo -n '¿Quiere cambiar la propiedad de la carpeta? (S/N): '
  read A

  if [ "$A" = "S" ] || [ "$A" = "s" ]; then

    echo ""
    echo "1.- Cambiando propietario de la carpeta destino: $deployPath"

    #Damos permisos de ejecución a los scripts
    chown -R $serverUser $deployPath

    echo "2.- Ahora la carpeta pertenece al usuario $serverUser"

    echo ""
    echo "Fin."

  else
    echo 'Se ha cancelado el cambio de la propiedad de los ficheros'
  fi

  volverMenu

}

function iniciar()
```

```
{

  aviso="(AVISO: esto reiniciará el servidor en caso de que esté encendido)"

  echo ''
  echo -n "¿Quiere iniciar el servidor tomcat? ${aviso} (S/N): "
  read A

  if [ "$A" = "S" ] || [ "$A" = "s" ]; then

    echo ""
    echo '1.- Deteniendo servidor tomcat...'
    su -c "service $serverUser stop"

    echo '2.- Actualizando componente Netfilter...'
    iptables -t filter -I INPUT 1 -p tcp --dport 8080 -j ACCEPT

    echo '3.- Iniciando servidor tomcat...'
    su -c "service $serverUser start"

    echo ""
    echo "Fin."

  else
    echo 'Se ha cancelado el inicio del servidor Tomcat'
  fi

  volverMenu

}

function prepararWebXML()
{

  originalWeb="./Plantillas/web.xml"
  ficheroWeb="./Servlet/WEB-INF/web.xml"
  plantilla="./ Plantillas/PlantillaXML.xml"
  temp="./temp.xml"

  echo '2.3.- Reemplazando valores de LTI_Deploy_Vars.txt sobre plantilla'

  # Se elimina el fichero temp.xml si existiese
  if [ -f "$temp" ]; then

    rm $temp

  fi

  # Se toma el fichero web.xml sin las configuraciones de LTI_Deploy_Vars.txt
  if [ -f "$ficheroWeb" ]; then

    rm $ficheroWeb
    cp $originalWeb $ficheroWeb

  fi

  # Nombre del trabajo
  generarParametro "projectTitle" "$projectTitle"

  # Clave
  generarParametro "deliveryPassword" "$deliveryPassword"

  #Reemplazo de la ruta
  generarParametro "rootPath" "$rootPath"

  #Reemplazo de la ruta del excel de la EV
  generarParametro "evExcelPath" "$evExcelPath"
```

```
  #Reemplazo de la ruta del excel del profesor
  generarParametro "teacherExcelPath" "$teacherExcelPath"

  #Reemplazo de la fórmula de evaluación
  generarParametro "evaluationFormula" "$evaluationFormula"

  # Lo insertamos todo, temp.xml, en web.xml
  insertarAWebXML

}

function generarParametro()
{

  # Reemplazamos el nombre del parámetro
  sed -e "s|PARAMNAME|$1|g" $plantilla >> $temp

  # Reemplazamos el valor del parámetro
  sed -i -e "s|PARAMVALUE|$2|g" $temp

}

function insertarAWebXML()
{

  echo '2.4.- Insertando valores en web.xml'

  # Línea final
  line=$(cat "$ficheroWeb" | grep -n '</web-app>' | grep -o '^[0-9]*')

  # Vamos a insertarlo en la línea anterior
  line=$((line - 1))

  # Insertamos
  sed -i.bak -e "${line}r $temp" $ficheroWeb

  # Eliminamos el fichero temporal
  rm $temp

}

#Carga del fichero de variables
source ./LTI_Deploy_Vars.txt

#Lanzamiento del menu
menu
```

6.4.2 LTI_Deploy_Vars.txt [PFED]

```
# LTI_Deploy_Vars.txt
# Álvaro Martín Rodríguez
# Fichero de variables de despliegue PFED

# Estas lineas son comentarios

# Ruta de despliegue

deployPath=/var/lib/tomcat7/webapps/ServletPFED/

# ------------------

# Nombre del trabajo

projectTitle=PFED

# ------------------

# Nombre del usuario que ejecuta el servidor
```

```
serverUser=tomcat7

# ------------------

# A partir de aquí se configuran las variables
# pertenecientes al fichero web.xml, es decir,
# las que emplea la aplicación para funcionar.

# Los valores aquí insertados sobreescribirán a los
# de dicho fichero en caso de que se encuentren
# configurados

# Clave de entrega

# Contraseña del trabajo. Debe ser introducida por los alumnos para poder
# entregar su trabajo en la herramienta

deliveryPassword=09092015fp2

# ------------------

# Ruta raíz para la estructura de directorios

# En la ruta que se especifique en esta variable, se iniciara la estructura
# de directorios empleada por la herramienta para almacenar los trabajos
# y realizar las pruebas pertinentes

rootPath=/home/antonio/fp2_2015/

# ------------------

# Ruta de los ficheros Excel de evaluación

# Ruta de los ficheros para las notas, tanto para la enseñanza virtual
# como para el desglose personal del profesor. Nombre de los ficheros incluido.

evExcelPath=/home/antonio/fp2_2015/notas/Plantilla_Ev_PFED.xls
teacherExcelPath=/home/antonio/fp2_2015/notas/Plantilla_Profesor_PFED.xls

# ------------------

# Fórmula de evaluación

# Aquí se especifica como se deben evaluar al alumno en función de los distintos
# apartados. La nota de cada apartado es An, empezando en A0... A(n-1), por
# ejemplo 8 apartados irían de A0 hasta A7. Pueden emplearse +,-,*,/,(,).
# Los apartados tomarán el valor 1 en caso de correcto, y 0 en caso de
# incorrecto al ser evaluados por la herramienta.
# Se recomienda evitar decimales, escribiendo estos números como fracciones.

evaluationFormula="A0*(5/2)*(1+A1+A2+A3)"

# ------------------
```

6.4.3 LTI_Deploy_Vars.txt [PFPOO]

```
# LTI_Deploy_Vars.txt
# Álvaro Martín Rodríguez
# Fichero de variables de despliegue

# Estas lineas son comentarios

# Ruta de despliegue

deployPath=/var/lib/tomcat7/webapps/ServletPFPOO/
```

```
# ------------------

# Nombre del trabajo

projectTitle=PFPOO

# ------------------

# Nombre del usuario que ejecuta el servidor

serverUser=tomcat7

# ------------------

# A partir de aquí se configuran las variables
# pertenecientes al fichero web.xml, es decir,
# las que emplea la aplicación para funcionar.

# Los valores aquí insertados sobreescribirán a los
# de dicho fichero en caso de que se encuentren
# configurados

# Clave de entrega

# Contraseña del trabajo. Debe ser introducida por los alumnos para poder
# entregar su trabajo en la herramienta

deliveryPassword=09092015fp2

# ------------------

# Ruta raíz para la estructura de directorios

# En la ruta que se especifique en esta variable, se iniciara la estructura
# de directorios empleada por la herramienta para almacenar los trabajos
# y realizar las pruebas pertinentes

#rootPath=/home/btc/LTI/
rootPath=/home/antonio/fp2_2015/

# ------------------

# Ruta de los ficheros Excel de evaluación

# Ruta de los ficheros para las notas, tanto para la enseñanza virtual
# como para el desglose personal del profesor. Nombre de los ficheros incluido.

evExcelPath=/home/antonio/fp2_2015/notas/Plantilla_Ev_PFPOO.xls
teacherExcelPath=/home/antonio/fp2_2015/notas/Plantilla_Profesor_PFPOO.xls
#evExcelPath=/home/btc/LTI/notas/Plantilla_Ev.xls
#teacherExcelPath=/home/btc/LTI/notas/Plantilla_Profesor.xls

# ------------------

# Fórmula de evaluación

# Aquí se especifica como se deben evaluar al alumno en función de los distintos
# apartados. La nota de cada apartado es An, empezando en A0... A(n-1), por
# ejemplo 8 apartados irían de A0 hasta A7. Pueden emplearse +,-,*,/,(,).
# Los apartados tomarán el valor 1 en caso de correcto, y 0 en caso de
# incorrecto al ser evaluados por la herramienta.
# Se recomienda evitar decimales, escribiendo estos números como fracciones.

evaluationFormula="5*(A0+A1)"

# ------------------
```

6.4.4 PlantillaXML.xml

```
  <context-param>
    <param-name>PARAMNAME</param-name>
    <param-value>PARAMVALUE</param-value>
  </context-param>
```

6.4.5 web.xml [Plantilla]

```
<web-app>

  <servlet>
    <servlet-name>Servlet Inicial</servlet-name>
    <servlet-class>ProjectEvaluation</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>Servlet Inicial</servlet-name>
    <url-pattern>/LTIApp</url-pattern>
  </servlet-mapping>

</web-app>
```

6.4.6 ErrorApp.jsp

```
<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 Transitional//EN\">
<html>
  <head>
    <title>Project Evaluation Error</title>
    <link rel="stylesheet" href="Resources/Error.css">
    <meta charset="UTF-8">
  </head>
  <body>
    <h2 id="focusedError">Error en la herramienta LTI</h2>
    <h3 id="recommendedAction"><%= session.getAttribute("errorLTI") %></h3>
  </body>
</html>
```

6.4.7 ErrorGet.html

```
<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 Transitional//EN\">
<html>
  <head>
    <title>Project Evaluation Error</title>
    <link rel="stylesheet" href="Resources/Error.css">
    <meta charset="UTF-8">
  </head>
  <body>
    <h2 id="focusedError">
    No se permite acceder mediante peticiones GET a esta herramienta LTI</h2>
    <h3 id="recommendedAction">Por favor, notifique este mensaje al
    profesorado a cargo del enlace a la herramienta</h3>
  </body>
</html>
```

6.4.8 ErrorPost.html

```
<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 Transitional//EN\">
<html>
  <head>
    <title>Project Evaluation Error</title>
    <link rel="stylesheet" href="Resources/Error.css">
```

```
    <meta charset="UTF-8">
  </head>
  <body>
    <h2 id="focusedError">Error en la petición POST</h2>
    <h3 id ="focusedError">El acceso mediante Basic LTI a esta herramienta
    ha sido realizado de manera errónea</h3>
    <h4 id="recommendedAction">Por favor, notifique este mensaje al
    profesorado a cargo del enlace a la herramienta</h3>
  </body>
</html>
```

6.4.9 Manager.jsp

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

<%@page import="tfg.lti.Files.*"%>
<%@page import="tfg.lti.Files.ExcelWorker"%>
<%@page import="java.util.concurrent.atomic.AtomicInteger"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Apartado de Gestión</title>
  <link rel="stylesheet" type="text/css" href="Resources/Manager.css">
 <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js">
  </script>
  <script src="Resources/AjaxManager.js"></script>
</head>

<body>

  <%!

    ServletContext contexto = null;
    AtomicInteger alumnosConcurrentes = new AtomicInteger(0);

  %>

  <%

    try
    {

      // 0.- Obtenemos el contexto para trabajar con las variables
      contexto = pageContext.getServletContext();

      // 0.1.- Si los siguientes parámetros no existen en la aplicación
      // por no haber sido inicializada, deben crearse:
      // * Ruta de los Scripts - scriptsPath
      // * Alumnos siendo atendidos por la aplicación - parallelStudents
      // * Fórmula de evaluación utlizada - evaluationFormula
      if (contexto.getInitParameter("scriptsPath") == null)
      {

        application.setAttribute("scriptsPath",
                                 application.getRealPath("") + "/Scripts/");

        application.setAttribute("parallelStudents", alumnosConcurrentes);

        application.setAttribute("evaluationFormula",
                                 contexto.getInitParameter("evaluationFormula"));

        application.setAttribute("rootPath",
```

```
                                contexto.getInitParameter("rootPath"));

    }

%>

<h2>Project Evaluation Tool</h2>

<p id="project">Proyecto: <%= contexto.getInitParameter("projectTitle") %></p>

<span id="enabled">Habilitado: </span>

<%

  Boolean isEnabled = null;

  if ( application.getAttribute("deliveryEnabled") == null)
  {
    isEnabled = Boolean.parseBoolean(
                                contexto.getInitParameter("deliveryEnabled"));
    application.setAttribute("deliveryEnabled", isEnabled);
  }
  else
  {
    isEnabled = (Boolean) application.getAttribute("deliveryEnabled");
  }

  String cssClass = "falso";
  if ( isEnabled )
  {
    cssClass = "verdadero";
  }

  out.println("<span class=\"" + cssClass + "\" id=\"enableValue\">"
              + ( isEnabled ? "Sí" : "No" )
              + "</span>");

%>

<p id="formula">Fórmula de Evaluación:
                  <%= contexto.getInitParameter("evaluationFormula") %> </p>

<hr id="percentWidth">

<p id="path">Ruta raíz: <%= contexto.getInitParameter("rootPath") %> </p>

<p id="exEv">Ruta excel EV:
                           <%= contexto.getInitParameter("evExcelPath") %></p>

<p id="exTeach">Ruta excel profesor:
                    <%= contexto.getInitParameter("teacherExcelPath") %> </p>

<hr id="percentWidth">

<form id="control">

  <fieldset class="espacio">

    <legend>Controles</legend>

    <p id="pass" class="inLineStyle">Clave de entrega: </p>

    <input type="text" id="passBox" class="inLineStyle" value=<%

    String password = null;

    if ( application.getAttribute("deliveryPassword") == null)
    {
```

```
          password = contexto.getInitParameter("deliveryPassword").toString();
          application.setAttribute("deliveryPassword", password);
        }
        else
        {
          password = application.getAttribute("deliveryPassword").toString();
        }

        out.println(password);

        %> />

        <p class="info">Nota: La clave solo se establecerá cuando se habilite o
        deshabilite la herramienta</p>

        <input type="button" id="enableDisable" value=<%

        if ( isEnabled )
        {
            out.println("\"Deshabilitar Entrega\"");
        }
        else
        {
            out.println("\"Habilitar Entrega\"");
        }

        %>

        class="blockStyle">

      </fieldset>

    </form>


    <%

      }
      catch (Exception e)
      {

        // Definimos el mensaje de error
        session.setAttribute("errorLTI", e.toString() + ": " + e.getMessage());

        response.sendRedirect("./ErrorApp.jsp");

      }

      %>

  </body>

  </html>
```

6.4.10 ModifyStatus.jsp

```
<%@page import="tfg.lti.Files.ExcelWorker"%>

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<%! ServletContext contexto = null; %>

<%

    // 0.- Obtenemos el contexto de la web
    contexto = pageContext.getServletContext();
```

```
// 1.- Procesamos la cadena recibida
String modifyName = null;
String newPassword = null;
modifyName = request.getParameter("change");
newPassword = request.getParameter("password");
Boolean result = true;

// 2.- La contraseña se establece al valor existente en la caja
application.setAttribute("deliveryPassword", newPassword);

// 3.- Comprobamos qué valor se quiere cambiar y procedemos
if (modifyName != null)
{

    if (modifyName.equals("deliveryEnabled"))
    {

        Boolean enableDelivery = false;

        if ( null != application.getAttribute("deliveryEnabled"))
        {
          enableDelivery =
                    (Boolean) application.getAttribute("deliveryEnabled");
        }

        enableDelivery = !enableDelivery;
        application.setAttribute("deliveryEnabled", enableDelivery);

        // Si estamos abriendo/cerrando la entrega, toca cargar
        // o guardar los cambios producidos en el excel

        ExcelWorker ewEV = null;
        ExcelWorker ewTeach = null;

        // 3.1.- Si se ha activado
        if (enableDelivery == true)
        {

            ewEV =
                  new ExcelWorker(contexto.getInitParameter("evExcelPath"));

            ewTeach =
            new ExcelWorker (contexto.getInitParameter("teacherExcelPath"));

            application.setAttribute("evExcel", ewEV);
            application.setAttribute("teacherExcel", ewTeach);

            application.setAttribute("projectTitle",
                  (String) contexto.getInitParameter("projectTitle") );

        }
        else
        {

            // 3.2.- Si se ha desactivado

            ewEV = (ExcelWorker) application.getAttribute("evExcel");
            ewTeach =
                    (ExcelWorker) application.getAttribute("teacherExcel");

            ewEV.writeFile();
            ewTeach.writeFile();

            application.setAttribute("evExcel", null);
            application.setAttribute("teacherExcel", null);

        }
```

```
        }
        else
        {
            // No se ha modificado ninguna propiedad
            result = false;
        }

        // Devolvemos la propiedad cambiada para que se recargue
        out.print(result);

    }

%>
```

6.4.11 Student.jsp

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

<%@page import="java.util.Date"%>

<%@page import="tfg.lti.Files.*"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Apartado de Entrega de Trabajos</title>
  <link rel="stylesheet" type="text/css" href="Resources/Student.css">
 <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js">
  </script>
  <script src="Resources/AjaxStudent.js"></script>
  <script src="Resources/JSStudent.js"></script>
</head>
<body>

  <h2>Apartado de Entrega de Trabajos</h2>

  <hr id="percentWidth">

<%

  try
  {
    /*
    (String) application.getRealPath(""),
    */

    // 0.- Si no está habilitada la entrega, se notifica
    if ( (null == application.getAttribute("deliveryEnabled")) ||
         ! (Boolean) application.getAttribute("deliveryEnabled") )
    {

%>

  <h3 id="deliveredMessage">La entrega de trabajos no se encuentra
  habilitada</h3>

<%

    }
    else
    {
```

```
%>

  <h3>Evaluación - <%= application.getAttribute("projectTitle").toString() %>
  </h3>

  <p name="step" id="step">Seleccione su fichero uvus.zip y pulse en
                           "Entregar"</p>

  <form name="delivery" id="delivery" action="UploadFile.jsp"
        method="POST" enctype="multipart/form-data">

    <fieldset class="space">

      <legend>Entrega -
            <%= application.getAttribute("projectTitle").toString() %></legend>

      <span id="uvus_span" class="label">UVUS: </span>

      <span name="uvus" id="uvus" class="relevantData"><%=
         session.getAttribute("UVUS")
      %></span>

      <br><br>

      <span id="pass_span" class="label">Contraseña del examen: </span>

      <input type="password" name="pass" id="pass">

      <br><br>

      <input type="file" name="userFile" id="userFile"
             class="label" value="Examinar">

      <br><br>

      <input type="button" name="send" id="send" onClick="sendWork();"
             value="Entregar">

    </fieldset>

  </form>

<%

    }

  }
  catch (Exception e)
  {

    // Definimos el mensaje de error
    session.setAttribute("errorLTI", e.toString() + ": " + e.getMessage());

    response.sendRedirect("./ErrorApp.jsp");

  }

%>

</body>

</html>
```

6.4.12 UploadFile.jsp

```
<%@ page import="java.io.*" %>
<%@ page import="java.util.*" %>
```

```
<%@ page import="javax.script.*" %>
<%@ page import="org.apache.commons.io.*" %>
<%@ page import="org.apache.commons.fileupload.*" %>
<%@ page import="org.apache.commons.fileupload.disk.*" %>
<%@ page import="org.apache.commons.fileupload.servlet.*" %>
<%@ page import="java.util.concurrent.atomic.AtomicInteger"%>

<%@page import="tfg.lti.Files.ExcelWorker"%>
<%@page import="tfg.lti.Files.ScriptWorker"%>
<%@page import="tfg.lti.Control.ErrorControlado"%>

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<%

  // 0.- Obtenemos las variables necesarias para funcionar

  AtomicInteger alumnosConcurrentes = (AtomicInteger)
                                application.getAttribute("parallelStudents");

  String formulaNotas = (String) application.getAttribute("evaluationFormula");

  String deliveryPassword =
                        (String) application.getAttribute("deliveryPassword");

  ExcelWorker ewEV = (ExcelWorker) application.getAttribute("evExcel");

  ExcelWorker ewTeach = (ExcelWorker) application.getAttribute("teacherExcel");

  String rootPath = (String) application.getAttribute("rootPath");

  String scriptsPath = (String) application.getAttribute("scriptsPath");

  String projectTitle = (String) application.getAttribute("projectTitle");

  // 0.1.- Obtenemos los datos del alumno
  String uvus = (String) session.getAttribute("UVUS");
  String password = null;

  // 0.2.- Variables de datos
  String locationFile = "";
  String locationProject = "";
  FileItem preprocessItem = null;

  //0.3.- Variables de control
  boolean correctPassword = false;
  boolean firstPhaseExisted = false;
  boolean firstPhaseIsCorrect = false;

  // ** Fase 1 - Recepción de datos, comprobaciones y preparación **
  if (alumnosConcurrentes.get() <= 10)
  {

    alumnosConcurrentes.incrementAndGet();
    firstPhaseExisted = true;

    // 1.1.- Cargamos el fichero
    FileItemFactory file_factory = new DiskFileItemFactory();
    ServletFileUpload servlet_up = new ServletFileUpload(file_factory);

    List items;
    items = servlet_up.parseRequest(request);

    for(int i=0;i<items.size();i++)
    {

      FileItem item = (FileItem) items.get(i);
```

```
  // Separamos el fichero del resto de paremetros del request
  if (item.isFormField())
  {

    // 1.2.- Si no es el fichero, solo recibimos la contraseña
    password = item.getString();

    // Establecemos si la contraseña es correcta
    if (deliveryPassword.equals(password))
    {
      correctPassword = true;
    }

  }
  else
  {

    if (correctPassword)
    {

      // 2.- Construimos las variables
      locationProject = rootPath + uvus + "/" + projectTitle + "/";

      locationFile = rootPath + uvus + "/" + projectTitle + "/"
                              + "VersionActual/" ;

      String locationBackups = rootPath + uvus + "/" + projectTitle + "/"
                              + "VersionesAnteriores/" ;

      // 3.- Creamos la estructura de directorios del alumno
      File folder = new File(locationFile);
      folder.mkdirs();

      folder = new File(locationBackups);
      folder.mkdirs();

      File archivo_server = new File(locationFile + item.getName());
      //Finalmente lo escribimos en disco
      item.write(archivo_server);

      /*
      out.print("Nombre --> " + item.getName() );
      out.print("<br> Tipo --> " + item.getContentType());
      out.print("<br> tamaño --> " + (item.getSize()/1240)+ "KB");
      out.print("<br>");
      */

      // 4.- Establecemos que se puede continuar con la ejecución de la
      // aplicación llegados a este punto correctamente
      firstPhaseIsCorrect = true;
      preprocessItem = item;

    }
    else
    {
      out.println("Error: la contraseña introducida es incorrecta");
    }

  }

}

// ** Fase 2 - Preprocesado, procesado y resultados **
if (firstPhaseIsCorrect)
{

  // Obtención de resultados a partir de este punto -> ErrorControlado
  ErrorControlado result = null;
```

```
// 5.- Buscamos el fichero de preprocesado
ScriptWorker executer = new ScriptWorker();
String filePreprocess = "";

// Buscamos el fichero de preprocesado para el trabajo
File dir = new File(scriptsPath);
if (dir.exists())
{

  File[] ficheros = dir.listFiles();
  for(int i = 0; i < ficheros.length; i++)
  {

    // Evitamos duplicidad de ficheros de backup
    if (ficheros[i].getName().startsWith("Preprocess_" + projectTitle)
          && (ficheros[i].getName().endsWith("~") == false))
    {
      filePreprocess = ficheros[i].getAbsolutePath();
    }

  }

  // 6.- Ejecutamos el preprocesado, si es correcto continuamos
  if ((result = executer.runPreprocess(filePreprocess, rootPath, uvus,
          projectTitle, preprocessItem.getName())).getIsError() == false)
  {

    // 7.- Copiamos los ficheros y las carpetas de corrección al
    // trabajo del alumno
    File destDir = new File(locationProject);
    File eachFile = null;
    int x = 0;

    // Listamos el directorio
    dir = new File(scriptsPath + projectTitle + "/");

    if (dir.exists())
    {

      ficheros = dir.listFiles();

      for(x = 0; x < ficheros.length; x++)
      {

        eachFile = new File(destDir.toString()+"/"+ficheros[x].getName());

        // Si existe el fichero, lo borramos para usar el nuevo
        if (eachFile.exists() == true)
        {
          FileUtils.deleteQuietly(eachFile);
        }

        if ( ficheros[x].isDirectory() == false)
        {
          FileUtils.copyFileToDirectory(ficheros[x], destDir);
          eachFile.setExecutable(true);
        }
        else
        {

          eachFile = new File(destDir.toString());

          FileUtils.copyDirectoryToDirectory(ficheros[x], eachFile);

          // Aseguramos permisos de ejecución a los subficheros
          File newDirectory = new File(eachFile.toString() + "/"
                                              + ficheros[x].getName());
          //out.println(newDirectory.toString());
```

```
      File[] otherScripts = newDirectory.listFiles();
      for (File otherScript : otherScripts)
      {
        otherScript.setExecutable(true);
      }

    }

  }

}

// 8.- Ejecutamos la corrección. Los ficheros se encuentran en la
// carpeta de del trabajo del alumno
ArrayList<Boolean> results = executer.runExamination(locationProject,
                                    rootPath, uvus, projectTitle);

// 9.- Eliminamos los ficheros de la versión actual
FileUtils.deleteDirectory(new File(locationFile));

// 10.- Pasamos los resultados a los Excel
// 10.1.- Fichero Excel del Profesor: Apartados + Nota final
ScriptEngine engine =
           new ScriptEngineManager().getEngineByName("JavaScript");
Map<String, Object> vars = new HashMap<String, Object>();

Boolean[] marks = new Boolean[results.size()];

for (x = 0; x < results.size(); x++)
{

  marks[x] = results.get(x);

  // Preparamos el valor para la fórmula
  vars.put("A" + x, (marks[x]) ? 1 : 0);

}

// 10.1.1.- Cálculo de la nota del aluno
float nota = Float.parseFloat(
engine.eval(formulaNotas, new SimpleBindings(vars)).toString());

if ((result = ewTeach.UpdateMarks(uvus, marks, nota)).getIsError()
                                                          == false)
{

  // 10.2.- Fichero Excel Enseñanza Virtual
  if (( result = ewEV.UpdateMarks(uvus, nota) ).getIsError() == false)
  {

    int deliveriesCount = ewTeach.deliveriesCount(uvus,marks.length);

    out.println("Trabajo entregado correctamente. N° de Entregas "
                + "realizadas: " + deliveriesCount);

  }
  else
  {

    out.println("Error al insertar las notas en el excel EV<br>");
    if (result.getException() != null)
    {
      out.println(result.getException().toString() + ": " +
                  result.getException().getMessage());
    }

  }
```

```
          }
          else
          {

            out.println("Error al insertar las notas en el excel del Profesor");
            if (result.getException() != null)
            {

              out.println("<br>" + result.getException().toString() + ": " +
                          result.getException().getMessage());
            }

          }

        }
        else
        {
          out.println("No se ha podido procesar el trabajo. "
                      + "Puede que el formato entregado sea incorrecto.");
        }


      }
      else
      {
        out.println("Error en la carpeta de scripts de corrección");
      }

      // Se ha procesado el trabajo, admitimos un nuevo alumno
      alumnosConcurrentes.decrementAndGet();

    }
    else if ( firstPhaseExisted )
    {
      // Si no esta bien la primera parte, pero ha existido
      alumnosConcurrentes.decrementAndGet();
    }

  }
  else
  {
    out.println("Se están corrigiendo muchos trabajos ahora mismo."
                + " Espere unos segundos e inténtalo de nuevo");
  }

%>
```

6.4.13 AjaxManager.js

```
// Variables globales
var xmlhttp=new XMLHttpRequest();
var itemModified = "";



// Preparamos la función que se ejecutará cuando acabe la petición async

xmlhttp.onreadystatechange=function()

{



  if (xmlhttp.readyState==4 && xmlhttp.status==200)

  {
```

```
    // Cargamos el mensaje que devuelve el JSP
    var response = xmlhttp.responseText;

    // Si se ha modificado correctamente el "itemModified"
    response = JSON.parse(response);

    if ( response == true)
    {
      changeValue();
      changeStyle();
    }


  }



};

function askChange(parameter)

{

  var url = "ModifyStatus.jsp";
  var params = "change=" + parameter + "&password=" + $("#passBox").val();
  xmlhttp.open("POST", url, true);

  // Preparamos la cabecera en función de los parámetros a cambiar
  xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
  xmlhttp.setRequestHeader("Content-length", params.length);
  xmlhttp.setRequestHeader("Connection", "close");


  xmlhttp.send(params);



}

function changeValue()
{

  var value = "Sí";
  if( $(itemModified).text() == "Sí" )
  {
    value = "No";
  }

  $(itemModified).text(value);

}

function changeStyle()
{

  var style_false = "falso";
  var style_true = "verdadero";

  if ($(itemModified).val() == "Sí")
  {
    $(itemModified).toggleClass(style_false + " " + style_true);
  }
  else
  {
    $(itemModified).toggleClass(style_true + " " + style_false);
  }
```

```
}

$(document).ready(function(){

  $( "#enableDisable" ).click(
    function()
    {

      if ( $(this).val() == "Deshabilitar Entrega")
      {
        $(this).prop("value", "Habilitar Entrega");
      }
      else
      {
        $(this).prop("value", "Deshabilitar Entrega");
      }

      itemModified = "#enableValue";
      askChange("deliveryEnabled");
    }
  );

});
```

6.4.14 AjaxStudent.js

```
var xmlhttp=new XMLHttpRequest();

// Preparamos la función que se ejecutará cuando acabe la petición async
xmlhttp.onreadystatechange=function()
{

  if (xmlhttp.readyState==4 && xmlhttp.status==200)
  {

    // Eliminamos el formulario de entrega
    parent = document.getElementById("delivery").parentNode;
    parent.removeChild(document.getElementById("delivery"));

    // Actualizamos el resultado
    var element = document.getElementById("step");

    // Cargamos el mensaje que devuelve el JSP
    element.innerHTML = xmlhttp.responseText;

  }

}

function loadFile()
{

  // var pUvus = document.getElementById("uvus").value;
  var pPassword = document.getElementById("pass").value;

  var file = document.getElementById("userFile");

  var formData = new FormData();
  formData.append("pass", pPassword);
  //formData.append("uvus", pUvus);
  formData.append("upload", file.files[0]);

  xmlhttp.open("POST","UploadFile.jsp",true);

  xmlhttp.send(formData);

}
```

6.4.15 Error.css

```
#focusedError
{
  color:#FFFFFF;
  background-color: #000000;
  text-align:center;
}

#recommendedAction
{
  text-align:center;
}
```

6.4.16 JSStudent.js

```
//Para IE6, IE5

if (!window.XMLHttpRequest)

{


  var infoLabel = document.getElementById("step");

  infoLabel.innerHTML = "La aplicación no es compatible con este navegador!";



  //Eliminamos el formulario

  var form = document.getElementById("delivery");



  parent = form.parentNode;

  parent.removeChild(form);



}



function blockUI()

{

  var sendButton = document.getElementById("send");
  sendButton.setAttribute("disabled", true);



  var infoLabel = document.getElementById("step");



  infoLabel.innerHTML = "Espere mientras carga su trabajo y se realizan"

                      + " las pruebas...";



}

function sendWork()
```

```
{

  var valid = true;

  if ($("#userFile").val() != '')
  {

    blockUI();
    loadFile();

  }
  else
  {

    alert("Seleccione el fichero antes de pulsar \"Entregar\"");

  }

}
```

6.4.17 Manager.css

```
body
{
  text-align: center;
  background-color: #ffffff;
}

h2
{
  background-color: #000000;
  color: white;
}

/* CLASSES */
.espacio
{
  width: 50%;
  margin: auto;
  text-align:center;
  padding-bottom: 10px;
  padding-top: 10px;
  color: blue;
}

.info
{
  font-size: 12px;
  color: black;
}

.inLineStyle
{
  display: inline;
}

.blockStyle
{
  display: block;
  margin: auto;
  margin-top: 15px;

}

.falso
{
  color:red;
}
```

```
.verdadero
{
  color:chartreuse;
}

/* IDs */
#percentWidth
{
  width: 45%;
}
```

6.4.18 Student.css

```
body
{
  text-align: center;
  background-color: #ffffff;
}

/* CLASSES */
.space
{
  width: 40%;
  margin: auto;
  text-align:center;
  padding-bottom: 10px;
  padding-top: 10px;
  color: blue;
}

.label
{
  color: black;
}

.relevantData
{
  color: green;
}

.inLineStyle
{
  display:inline;
}

/* IDs */
#percentWidth
{
  width: 45%;
}
```

6.4.19 Preprocess_PFED.sh

```
#!/bin/bash
# Álvaro Martín Rodríguez
# TFG - LTI
# Script - Preprocess_PFED
# Distribución objetivo: Guadalinex v7
#
# Este fichero incluye el código que es ejecutado para tratar
# el fichero cargado por el alumno en la entrega del trabajo.
#
# Aquí se realizan tareas como: como copias de respaldo,
# descompresión de los trabajos, y preparación del entorno.
```

```
#
# A partir de este punto, el script desarrolla el procesamiento
# necesario para la corrección del trabajo concreto que nos ocupa.
#
# Parámetros esperados:
# 1.- Carpeta Raíz
# 2.- Uvus del alumno
# 3.- Título del trabajo
# 4.- Nombre del fichero
#
# Funciones utilizadas posteriormente
# 1.- backup: se encarga de devolver la ruta exacta del fichero
#     donde debe ser almacenada la copia de seguridad
function backup()
{

  #Contamos cuantos ficheros hay en el path
  amount="$(ls -1 $1 | wc -l)"

  #Componemos el nombre del nuevo backup con el nombre del fichero
  # y su nº de entrega. Debemos quitar la extensión al final del nombre
  # y volver a concatenarla al final.
  extension='.zip'
  name=$2
  clearname="${name%.*p}"

  name="$clearname""_$amount$extension"

  REPLY="$1$name"

}

result=0

if [ $# -eq 4 ]; then

  # Paso 1 - Backup de la entrega anterior, en caso de que exista.
  rootpath=${1}
  uvus=${2}
  title=${3}
  file=${4}

  # Componemos la ruta del trabajo
  path="${rootpath}$uvus/$title/VersionActual/"
  pathbackups="${rootpath}$uvus/$title/VersionesAnteriores/"
  filepath="${path}$file"

  #Si existe el fichero del trabajo como se espera
  if [ -r $filepath ]; then

    #1.- Hacemos backup al fichero en VersionesAnteriores
    backup $pathbackups $file
    backupfile=$REPLY
    cp $filepath $backupfile

    #2.- Descomprimimos el trabajo del alumno
    unzip $filepath -d $path
    result=$?

    #Lo intentamos una segunda vez en caso de que no haya sido posible
    if [ ! $? == 0 ]; then

        unzip $filepath
        result=$?

    fi

  else
```

```
    echo 'El fichero del alumno no está disponible'
    result=1

  fi

else

  echo 'Número de parámetros incorrectos'
  result=1

fi

exit $result
```

6.4.20 Preprocess_PFPOO.sh

```
#!/bin/bash
# Álvaro Martín Rodríguez
# TFG - LTI
# Script - Preprocess_PFPOO
# Distribución objetivo: Guadalinex v7
#
# Este fichero incluye el código que es ejecutado para tratar
# el fichero cargado por el alumno en la entrega del trabajo.
#
# Aquí se realizan tareas como: como copias de respaldo,
# descompresión de los trabajos, y preparación del entorno.
#
# A partir de este punto, el script desarrolla el procesamiento
# necesario para la corrección del trabajo concreto que nos ocupa.
#
# Parámetros esperados:
# 1.- Carpeta Raíz
# 2.- Uvus del alumno
# 3.- Título del trabajo
# 4.- Nombre del fichero
#
# Funciones utilizadas posteriormente
# 1.- backup: se encarga de devolver la ruta exacta del fichero
#     donde debe ser almacenada la copia de seguridad
function backup()
{

  #Contamos cuantos ficheros hay en el path
  amount="$(ls -1 $1 | wc -l)"

  #Componemos el nombre del nuevo backup con el nombre del fichero
  # y su nº de entrega. Debemos quitar la extensión al final del nombre
  # y volver a concatenarla al final.
  extension='.zip'
  name=$2
  clearname="${name%.*p}"

  name="$clearname""_$amount$extension"

  REPLY="$1$name"

}

result=0

if [ $# -eq 4 ]; then

  # Paso 1 - Backup de la entrega anterior, en caso de que exista.
  rootpath=${1}
  uvus=${2}
  title=${3}
```

```
  file=${4}

  # Componemos la ruta del trabajo
  path="${rootpath}$uvus/$title/VersionActual/"
  pathbackups="${rootpath}$uvus/$title/VersionesAnteriores/"
  filepath="${path}$file"

  #Si existe el fichero del trabajo como se espera
  if [ -r $filepath ]; then

    #1.- Hacemos backup al fichero en VersionesAnteriores
    backup $pathbackups $file
    backupfile=$REPLY
    cp $filepath $backupfile

    #2.- Descomprimimos el trabajo del alumno
    unzip $filepath -d $path
    result=$?

    #Lo intentamos una segunda vez en caso de que no haya sido posible
    if [ ! $? == 0 ]; then

        unzip $filepath
        result=$?

    fi

  else

    echo 'El fichero del alumno no está disponible'
    result=1

  fi

else

  echo 'Número de parámetros incorrectos'
  result=1

fi

exit $result
```

6.4.21 Apartado_0.sh [PFED]

```
#!/bin/bash
# Álvaro Martín Rodríguez
# TFG - LTI
# Script - PFEDfileProcess
# Distribución objetivo: Guadalinex v7
#
# La RUTA que se recibe como parámetro será absoluta, e indicará donde está el
# apartado 0 del alumno a corregir

function path()
{

  MY_PATH="`dirname \"$0\"`"
  MY_PATH="`( cd \"$MY_PATH\" && pwd )`"

}

RUTA="."
if [ $# == 1  ]; then
  RUTA="$1/Apartado0/herramienta"
fi

path
```

```
echo $RUTA > ./FICHERO_DEBUG

/bin/bash "$MY_PATH/Apartado_0/Herramienta.sh" $RUTA
```

6.4.22 Herramienta.sh

```
#!/bin/bash

RUTA="."
result=0

function diferencia()
{

  diff $1 $2
  if [ $? != 0 ]; then
    exit 1
    fi
  return $?

}

if [ $# == 1  ]; then

  RUTA=$1

fi

echo $RUTA >> ./FICHERO_DEBUG

DIR_SALIDA=$RUTA/pruebas/misSalidas

  mkdir -p $RUTA/pruebas/SPC/
  rm -f $RUTA/ResumenMemoryReport
  rm -f $RUTA/pruebas/SPC/*.c
  rm -f $RUTA/pruebas/SPC/*.h
  #TEMPORAL
  RUTAACTUAL=$(pwd)
  cd "$RUTA"
  java -jar ./jar/CopiaFicheros.jar
  #cd $RUTAACTUAL
  #FIN TEMPORAL
  gcc -o MPC.exe $RUTA/include/_dit_memory_.o $RUTA/pruebas/SPC/*.c
  rm -f $RUTA/pruebas/SPC/*.*
  rmdir  $RUTA/pruebas/SPC
  echo "\t__ Prueba 00 ____________________________________________________________"
  $RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
                $RUTA/pruebas/entradas/entrada00  \
                $DIR_SALIDA/salida00  \
                > $DIR_SALIDA/salidaStd00 \
                2> $DIR_SALIDA/salidaErr00
  diferencia  $RUTA/pruebas/misSalidas/salida00 \
              $RUTA/pruebas/salidas/salida00
  diferencia  $RUTA/pruebas/misSalidas/salidaStd00 \
              $RUTA/pruebas/salidas/salidaStd00
  diferencia  $RUTA/pruebas/misSalidas/salidaErr00 \
              $RUTA/pruebas/salidas/salidaErr00
  mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport00
  echo "\t__ Prueba 01 ____________________________________________________________"
  $RUTA/MPC.exe $RUTA/pruebas/configuracion/config01 \
                $RUTA/pruebas/entradas/entrada00  \
                $DIR_SALIDA/salida01  \
                > $DIR_SALIDA/salidaStd01 \
                2> $DIR_SALIDA/salidaErr01
  diferencia  $RUTA/pruebas/misSalidas/salida01 \
              $RUTA/pruebas/salidas/salida01
```

```
diferencia  $RUTA/pruebas/misSalidas/salidaStd01 \
            $RUTA/pruebas/salidas/salidaStd01
diferencia  $RUTA/pruebas/misSalidas/salidaErr01 \
            $RUTA/pruebas/salidas/salidaErr01
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport01
echo "\t__ Prueba 02 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config02 \
               $RUTA/pruebas/entradas/entrada00  \
               $DIR_SALIDA/salida02  \
               > $DIR_SALIDA/salidaStd02 \
               2> $DIR_SALIDA/salidaErr02
diferencia  $RUTA/pruebas/misSalidas/salida02 \
            $RUTA/pruebas/salidas/salida02
diferencia  $RUTA/pruebas/misSalidas/salidaStd02 \
            $RUTA/pruebas/salidas/salidaStd02
diferencia  $RUTA/pruebas/misSalidas/salidaErr02 \
            $RUTA/pruebas/salidas/salidaErr02
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport02
echo "\t__ Prueba 03 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config03 \
               $RUTA/pruebas/entradas/entrada00  \
               $DIR_SALIDA/salida03  \
               > $DIR_SALIDA/salidaStd03 \
               2> $DIR_SALIDA/salidaErr03
diferencia  $RUTA/pruebas/misSalidas/salida03 \
            $RUTA/pruebas/salidas/salida03
diferencia  $RUTA/pruebas/misSalidas/salidaStd03 \
            $RUTA/pruebas/salidas/salidaStd03
diferencia  $RUTA/pruebas/misSalidas/salidaErr03 \
            $RUTA/pruebas/salidas/salidaErr03
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport03
echo "\t__ Prueba 04 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config04 \
               $RUTA/pruebas/entradas/entrada00  \
               $DIR_SALIDA/salida04  \
               > $DIR_SALIDA/salidaStd04 \
               2> $DIR_SALIDA/salidaErr04
diferencia  $RUTA/pruebas/misSalidas/salida04 \
            $RUTA/pruebas/salidas/salida04
diferencia  $RUTA/pruebas/misSalidas/salidaStd04 \
            $RUTA/pruebas/salidas/salidaStd04
diferencia  $RUTA/pruebas/misSalidas/salidaErr04 \
            $RUTA/pruebas/salidas/salidaErr04
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport04
echo "\t__ Prueba 05 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config05 \
               $RUTA/pruebas/entradas/entrada00  \
               $DIR_SALIDA/salida05  \
               > $DIR_SALIDA/salidaStd05 \
               2> $DIR_SALIDA/salidaErr05
diferencia  $RUTA/pruebas/misSalidas/salidaStd05 \
            $RUTA/pruebas/salidas/salidaStd05
diferencia  $RUTA/pruebas/misSalidas/salidaErr05 \
            $RUTA/pruebas/salidas/salidaErr05
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport05
echo "\t__ Prueba 06 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config06 \
               $RUTA/pruebas/entradas/entrada00  \
               $DIR_SALIDA/salida06  \
               > $DIR_SALIDA/salidaStd06 \
               2> $DIR_SALIDA/salidaErr06
diferencia  $RUTA/pruebas/misSalidas/salidaStd06 \
            $RUTA/pruebas/salidas/salidaStd06
diferencia  $RUTA/pruebas/misSalidas/salidaErr06 \
            $RUTA/pruebas/salidas/salidaErr06
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport06
echo "\t__ Prueba 07 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config07 \
               $RUTA/pruebas/entradas/entrada00  \
```

```
              $DIR_SALIDA/salida07  \
              > $DIR_SALIDA/salidaStd07 \
              2> $DIR_SALIDA/salidaErr07
diferencia  $RUTA/pruebas/misSalidas/salidaStd07 \
            $RUTA/pruebas/salidas/salidaStd07
diferencia  $RUTA/pruebas/misSalidas/salidaErr07 \
            $RUTA/pruebas/salidas/salidaErr07
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport07
echo "\t__ Prueba 08 ____________________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config08 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida08  \
              > $DIR_SALIDA/salidaStd08 \
              2> $DIR_SALIDA/salidaErr08
diferencia  $RUTA/pruebas/misSalidas/salidaStd08 \
            $RUTA/pruebas/salidas/salidaStd08
diferencia  $RUTA/pruebas/misSalidas/salidaErr08 \
            $RUTA/pruebas/salidas/salidaErr08
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport08
echo "\t__ Prueba 09 ____________________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config09 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida09  \
              > $DIR_SALIDA/salidaStd09 \
              2> $DIR_SALIDA/salidaErr09
diferencia  $RUTA/pruebas/misSalidas/salidaStd09 \
            $RUTA/pruebas/salidas/salidaStd09
diferencia  $RUTA/pruebas/misSalidas/salidaErr09 \
            $RUTA/pruebas/salidas/salidaErr09
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport09
echo "\t__ Prueba 10 ____________________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config10 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida10  \
              > $DIR_SALIDA/salidaStd10 \
              2> $DIR_SALIDA/salidaErr10
diferencia  $RUTA/pruebas/misSalidas/salidaStd10 \
            $RUTA/pruebas/salidas/salidaStd10
diferencia  $RUTA/pruebas/misSalidas/salidaErr10 \
            $RUTA/pruebas/salidas/salidaErr10
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport10
echo "\t__ Prueba 11 ____________________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config11 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida11  \
              > $DIR_SALIDA/salidaStd11 \
              2> $DIR_SALIDA/salidaErr11
diferencia  $RUTA/pruebas/misSalidas/salidaStd11 \
            $RUTA/pruebas/salidas/salidaStd11
diferencia  $RUTA/pruebas/misSalidas/salidaErr11 \
            $RUTA/pruebas/salidas/salidaErr11
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport11
echo "\t__ Prueba 12 ____________________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config12 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida12  \
              > $DIR_SALIDA/salidaStd12 \
              2> $DIR_SALIDA/salidaErr12
diferencia  $RUTA/pruebas/misSalidas/salidaStd12 \
            $RUTA/pruebas/salidas/salidaStd12
diferencia  $RUTA/pruebas/misSalidas/salidaErr12 \
            $RUTA/pruebas/salidas/salidaErr12
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport12
echo "\t__ Prueba 13 ____________________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config13 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida13  \
              > $DIR_SALIDA/salidaStd13 \
```

```
              2> $DIR_SALIDA/salidaErr13
diferencia  $RUTA/pruebas/misSalidas/salidaStd13 \
            $RUTA/pruebas/salidas/salidaStd13
diferencia  $RUTA/pruebas/misSalidas/salidaErr13 \
            $RUTA/pruebas/salidas/salidaErr13
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport13
echo "\t__ Prueba 14 ____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config14 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida14  \
              > $DIR_SALIDA/salidaStd14 \
              2> $DIR_SALIDA/salidaErr14
diferencia  $RUTA/pruebas/misSalidas/salidaStd14 \
            $RUTA/pruebas/salidas/salidaStd14
diferencia  $RUTA/pruebas/misSalidas/salidaErr14 \
            $RUTA/pruebas/salidas/salidaErr14
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport14
echo "\t__ Prueba 15 ____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config15 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida15  \
              > $DIR_SALIDA/salidaStd15 \
              2> $DIR_SALIDA/salidaErr15
diferencia  $RUTA/pruebas/misSalidas/salidaStd15 \
            $RUTA/pruebas/salidas/salidaStd15
diferencia  $RUTA/pruebas/misSalidas/salidaErr15 \
            $RUTA/pruebas/salidas/salidaErr15
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport15
echo "\t__ Prueba 16 ____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config16 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida16 \
              > $DIR_SALIDA/salidaStd16 \
              2> $DIR_SALIDA/salidaErr16
diferencia  $RUTA/pruebas/misSalidas/salidaStd16 \
            $RUTA/pruebas/salidas/salidaStd16
diferencia  $RUTA/pruebas/misSalidas/salidaErr16 \
            $RUTA/pruebas/salidas/salidaErr16
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport16
echo "\t__ Prueba 17 ____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config17 \
              $RUTA/pruebas/entradas/entrada00  \
              $DIR_SALIDA/salida17  \
              > $DIR_SALIDA/salidaStd17 \
              2> $DIR_SALIDA/salidaErr17
diferencia  $RUTA/pruebas/misSalidas/salidaStd17 \
            $RUTA/pruebas/salidas/salidaStd17
diferencia  $RUTA/pruebas/misSalidas/salidaErr17 \
            $RUTA/pruebas/salidas/salidaErr17
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport17
echo "\t__ Prueba 18 ____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
              $RUTA/pruebas/entradas/entrada18 \
              $DIR_SALIDA/salida18 \
              > $DIR_SALIDA/salidaStd18 \
              2> $DIR_SALIDA/salidaErr18
diferencia  $RUTA/pruebas/misSalidas/salida18 \
            $RUTA/pruebas/salidas/salida18
diferencia  $RUTA/pruebas/misSalidas/salidaStd18 \
            $RUTA/pruebas/salidas/salidaStd18
diferencia  $RUTA/pruebas/misSalidas/salidaErr18 \
            $RUTA/pruebas/salidas/salidaErr18
mv     MemoryReport                  $RUTA/pruebas/MemoryRep/MemoryReport18
echo "\t__ Prueba 19 ____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
              $RUTA/pruebas/entradas/entrada19  \
              $DIR_SALIDA/salida19 \
              > $DIR_SALIDA/salidaStd19 \
              2> $DIR_SALIDA/salidaErr19
```

```
diferencia  $RUTA/pruebas/misSalidas/salida19 \
            $RUTA/pruebas/salidas/salida19
diferencia  $RUTA/pruebas/misSalidas/salidaStd19 \
            $RUTA/pruebas/salidas/salidaStd19
diferencia  $RUTA/pruebas/misSalidas/salidaErr19 \
            $RUTA/pruebas/salidas/salidaErr19
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport19
echo "\t__ Prueba 20_____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
               $RUTA/pruebas/entradas/entrada20  \
               $DIR_SALIDA/salida20 \
               > $DIR_SALIDA/salidaStd20 \
               2> $DIR_SALIDA/salidaErr20
diferencia  $RUTA/pruebas/misSalidas/salida20 \
            $RUTA/pruebas/salidas/salida20
diferencia  $RUTA/pruebas/misSalidas/salidaStd20 \
            $RUTA/pruebas/salidas/salidaStd20
diferencia  $RUTA/pruebas/misSalidas/salidaErr20 \
            $RUTA/pruebas/salidas/salidaErr20
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport20
echo "\t__ Prueba 21 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
               $RUTA/pruebas/entradas/entrada21  \
               $DIR_SALIDA/salida21 \
               > $DIR_SALIDA/salidaStd21 \
               2> $DIR_SALIDA/salidaErr21
diferencia  $RUTA/pruebas/misSalidas/salida21 \
            $RUTA/pruebas/salidas/salida21
diferencia  $RUTA/pruebas/misSalidas/salidaStd21 \
            $RUTA/pruebas/salidas/salidaStd21
diferencia  $RUTA/pruebas/misSalidas/salidaErr21 \
            $RUTA/pruebas/salidas/salidaErr21
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport21
echo "\t__ Prueba 22_____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
               $RUTA/pruebas/entradas/entrada22 \
               $DIR_SALIDA/salida22 \
               > $DIR_SALIDA/salidaStd22 \
               2> $DIR_SALIDA/salidaErr22
diferencia  $RUTA/pruebas/misSalidas/salida22 \
            $RUTA/pruebas/salidas/salida22
diferencia  $RUTA/pruebas/misSalidas/salidaStd22 \
            $RUTA/pruebas/salidas/salidaStd22
diferencia  $RUTA/pruebas/misSalidas/salidaErr22 \
            $RUTA/pruebas/salidas/salidaErr22
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport22
echo "\t__ Prueba 23 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
               $RUTA/pruebas/entradas/entrada23  \
               $DIR_SALIDA/salida23 \
               > $DIR_SALIDA/salidaStd23 \
               2> $DIR_SALIDA/salidaErr23
diferencia  $RUTA/pruebas/misSalidas/salida23 \
            $RUTA/pruebas/salidas/salida23
diferencia  $RUTA/pruebas/misSalidas/salidaStd23 \
            $RUTA/pruebas/salidas/salidaStd23
diferencia  $RUTA/pruebas/misSalidas/salidaErr23 \
            $RUTA/pruebas/salidas/salidaErr23
mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport23
echo "\t__ Prueba 24 _____________________________________________________"
$RUTA/MPC.exe $RUTA/pruebas/configuracion/config00 \
            $RUTA/pruebas/entradas/entrada24  \
            $DIR_SALIDA/salida24 \
            > $DIR_SALIDA/salidaStd24 \
            2> $DIR_SALIDA/salidaErr24
diferencia  $RUTA/pruebas/misSalidas/salida24 \
            $RUTA/pruebas/salidas/salida24
diferencia  $RUTA/pruebas/misSalidas/salidaStd24 \
```

```
              $RUTA/pruebas/salidas/salidaStd24
  diferencia  $RUTA/pruebas/misSalidas/salidaErr24 \
              $RUTA/pruebas/salidas/salidaErr24
  mv     MemoryReport                   $RUTA/pruebas/MemoryRep/MemoryReport24

  if [ -f ResumenMemoryReport ]; then
      diferencia ResumenMemoryReport  zero
    result=$?
  else
    result=1
  fi


exit $result
```

6.4.23 Apartado_0.sh [PFPOO]

```
#!/bin/bash
#
# La RUTA que se recibe como parámetro será absoluta, e indicará donde está el
# apartado 0 del alumno a corregir

function path()
{

  MY_PATH="`dirname \"$0\"`"
  MY_PATH="`( cd \"$MY_PATH\" && pwd )`"

}

RUTA="."
OPCIONES=" -Xlint -encoding ISO-8859-1 -classpath ./bin  -cp ./bin -d ./bin "

if [ $# == 1  ]; then
  RUTA="$1/Apartado1"
fi

# Ejecutamos la función para obtener la ruta donde estamos actualmente
path

# Compilamos todo el código del alumno
cd $RUTA
make

# Compilación del código del alumno
javac $OPCIONES  ./src/fp2/poo/principal/Principal05.java

# Ejecución del código del alumno
java -classpath ./bin fp2.poo.principal.Principal05 agenda05 > prueba05

diff ./prueba05Proporcionada prueba05

if [ $? -eq 0 ]; then
  echo "     apartado 1: OK "
  resultado=0
else
  echo "     apartado 1: MAL "
  resultado=1
fi

exit $resultado
```

6.4.24 Apartado_1.sh [PFED]

```
#!/bin/bash
# Álvaro Martín Rodríguez
```

```
# TFG - LTI
# Script - PFEDfileProcess
# Distribución objetivo: Guadalinex v7

if [ $# == 1 ]; then

  cd "${1}Apartado1"

fi

resultado=1
resAlumAp1=Apartado1t1
ficheroProporcionado=Apartado1t1Proporcionado

# compila el programa del alumno
gcc -W -Wall *.c > /dev/null 2> /dev/null

# Borra el resultado anterior si lo hay
if [ -f $resAlumAp1 ]; then
   rm $resAlumAp1
fi

# si se ha generado el a.out , se ejecuta en background
# se espera 3 segundos a que termine y si no ha terminado se mata
if [ -f a.out ]; then
   #  echo "    Apartado 1 generado "
   make -B -f examenPFEDAptdo1 > /dev/null 2> /dev/null &
   # Se espera 3 segundos
   sleep 3

   # Se mata el proceso que se quedo en background
   pidof $!   # mira si esta activo el ultimo proceso que dejamos en background
   if [ $? -eq 0 ]; then
     echo "Apartado 1: Proceso activo, se procede al kill "
     kill -9 $!
   fi

   # se comprueba el resultado del apartado 1
   if [ -f $resAlumAp1 ]; then
     diff $ficheroProporcionado $resAlumAp1 > /dev/null
     if [ $? -eq 0 ]; then
       echo "     apartado 1: OK "
       resultado=0
     else
       echo "     apartado 1: MAL "
     fi
   else
     echo "     apartado 1: NO HECHO "
   fi

else
   echo "    Apartado 1: a.out NO generado "
fi

exit $resultado
```

6.4.25 Apartado_1.sh [PFPOO]

```
#!/bin/bash
#
# La RUTA que se recibe como parámetro será absoluta, e indicará donde está el
# apartado 0 del alumno a corregir

function path()
{

  MY_PATH="`dirname \"$0\"`"
```

```
  MY_PATH="`( cd \"$MY_PATH\" && pwd )`"

}

RUTA="."
OPCIONES=" -Xlint -encoding ISO-8859-1 -classpath ./bin  -cp ./bin -d ./bin "

if [ $# == 1  ]; then
  RUTA="$1/Apartado2"
fi

# Ejecutamos la función para obtener la ruta donde estamos actualmente
path

# Compilamos todo el código del alumno
cd $RUTA
make

# Compilación del código del alumno
javac $OPCIONES  ./src/fp2/poo/principal/Principal06.java

# Ejecución del código del alumno
java -classpath ./bin fp2.poo.principal.Principal06 agenda06 > prueba06

diff ./prueba06Proporcionada prueba06

if [ $? -eq 0 ]; then
  echo "     apartado 2: OK "
  resultado=0
else
  echo "     apartado 2: MAL "
  resultado=1
fi

exit $resultado
```

6.4.26 Apartado_2.sh [PFED]

```
#!/bin/bash
# Álvaro Martín Rodríguez
# TFG - LTI
# Script - PFEDfileProcess
# Distribución objetivo: Guadalinex v7

if [ $# == 1 ]; then

  cd "${1}Apartado2"

fi

resultado=1
resAlumAp2=Apartado2t1
ficheroProporcionado=Apartado2t1Proporcionado

# compila el programa del alumno
gcc -W -Wall *.c > /dev/null 2> /dev/null

# Borra el resultado anterior si lo hay
if [ -f $resAlumAp2 ]; then
   rm $resAlumAp2
fi

# si se ha generado el a.out , se ejecuta en background
# se espera 3 segundos a que termine y si no ha terminado se mata
if [ -f a.out ]; then
   #  echo "    Apartado 2 generado "
   make -B -f examenPFEDAptdo2 > /dev/null 2> /dev/null &
   # Se espera 3 segundos
```

```
    sleep 3

    # Se mata el proceso que se quedo en background
    pidof $!   # mira si esta activo el ultimo proceso que dejamos en background
    if [ $? -eq 0 ]; then
      echo "Apartado 2: Proceso activo, se procede al kill "
      kill -9 $!
    fi

    # se comprueba el resultado del Apartado 2
    if [ -f $resAlumAp2 ]; then
      diff $ficheroProporcionado $resAlumAp2 > /dev/null
      if [ $? -eq 0 ]; then
        echo "     Apartado 2: OK "
        resultado=0
      else
        echo "     Apartado 2: MAL "
      fi
    else
      echo "     Apartado 2: NO HECHO "
    fi

  else
     echo "    Apartado 2: a.out NO generado "
  fi

  exit $resultado
```

6.4.27 Apartado_3.sh [PFED]

```
#!/bin/bash
# Álvaro Martín Rodríguez
# TFG - LTI
# Script - PFEDfileProcess
# Distribución objetivo: Guadalinex v7

if [ $# == 1 ]; then

  cd "${1}Apartado3"

fi

resultado=1
resAlumAp3=Apartado3t1
ficheroProporcionado=Apartado3t1Proporcionado

# compila el programa del alumno
gcc -W -Wall *.c > /dev/null 2> /dev/null

# Borra el resultado anterior si lo hay
if [ -f $resAlumAp3 ]; then
   rm $resAlumAp3
fi

# si se ha generado el a.out , se ejecuta en background
# se espera 3 segundos a que termine y si no ha terminado se mata
if [ -f a.out ]; then
   #  echo "    Apartado 3 generado "
   make -B -f examenPFEDAptdo3 > /dev/null 2> /dev/null &
   # Se espera 3 segundos
   sleep 3

   # Se mata el proceso que se quedo en background
   pidof $!   # mira si esta activo el ultimo proceso que dejamos en background
   if [ $? -eq 0 ]; then
     echo "Apartado 3: Proceso activo, se procede al kill "
     kill -9 $!
```

```
    fi

    # se comprueba el resultado del Apartado 3
    if [ -f $resAlumAp3 ]; then
      diff $ficheroProporcionado $resAlumAp3 > /dev/null
      if [ $? -eq 0 ]; then
        echo "     Apartado 3: OK "
        resultado=0
      else
        echo "     Apartado 3: MAL "
      fi
    else
      echo "     Apartado 3: NO HECHO "
    fi

  else
     echo "    Apartado 3: a.out NO generado "
  fi

  exit $resultado
```

6.4.28 ProjectEvaluation.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpSession;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import blackboard.blti.provider.*;
import blackboard.blti.message.*;

public class ProjectEvaluation extends HttpServlet
{

  private static final long serialVersionUID = -1L;

  /**
   * Petición GET. No se admiten peticiones GET a la herramienta LTI.
   * @param request
   * @param response
   * @throws ServletException
   * @throws IOException
   */
  protected void doGet(HttpServletRequest request,
                       HttpServletResponse response) throws ServletException,
                                                            IOException {

    response.sendRedirect("./ErrorGet.html");

  }

  protected void doPost(HttpServletRequest request,
                        HttpServletResponse response) throws
                                              ServletException, IOException {

    BLTIMessage msg = BLTIProvider.getMessage( request );
    String key = msg.getKey();
    String secret = "gitt1fp2sc";

    if ( BLTIProvider.isValid( msg, secret ) )
    {

      String resourceId = msg.getResourceLink().getId();
      String user = msg.getUser().getId();
      boolean isInstructor = msg.getUser().isInstructor();
```

```
      // 0.- Se establece el UVUS como una variable a recibir por el JSP
      HttpSession session = request.getSession(true);
      session.setAttribute("UVUS", user);

      // 1.- Si es un profesor, se presenta la interfaz de gestión
      if (isInstructor)
      {

        // 2.1.- Redirigimos a la web de gestión
        response.sendRedirect("./Manager.jsp");

      }
      else
      {

        // 2.2.- Se lanza la web de corrección
        response.sendRedirect("./Student.jsp");

      }

    }
    else
    {

      // Error. El mensaje no es válido.
      response.sendRedirect("./ErrorPost.html");

    }

  }

}
```

6.4.29 ErrorControlado.java

```
package tfg.lti.Control;

/**
 * Clase empleada para devolver resultados de manera jerárquica junto con
 * posibles excepciones subyacentes a estos resultados.
 *
 * @author     Álvaro Martín Rodrí-guez
 * @version    0.2
 * @category   Control
 */
public class ErrorControlado
{

  private Boolean isError = false;
  private Exception receivedException = null;

  /**
   * @param  pError  Resultado de la operación. True=error, false=correcto
   * @param  pException  Excepción asociada al resultado
   *
   */
  public ErrorControlado(Boolean pError, Exception pException)
  {

    isError = pError;
    receivedException = pException;

  }

  /**
   * @return true = es erróneo, false = es correcto
```

```
     *
     */
    public Boolean getIsError()
    {
      return isError;
    }

    /**
     * @return La excepción asociada al resultado
     *
     */
    public Exception getException()
    {
      return receivedException;
    }

}
```

6.4.30 ExcelWorker.java

```
package tfg.lti.Files;

import tfg.lti.Control.ErrorControlado;

import java.io.IOException;
import java.io.InputStream;
import java.io.FileInputStream;
import java.io.FileOutputStream;

import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;

/**
 *
 * @author Álvaro Martín Rodrí-guez
 * @version 0.2
 * @category Excel Worker
 */
public class ExcelWorker
{

  private Workbook myWorkbook;
  private Sheet mySheet;
  private String filePath;

 /**
  * @param  filePath  Ruta completa de los directorios donde se
  * almacena el fichero, incluyendo el nombre
  * @throws Exception por trabajar con ficheros
  */
  public ExcelWorker(String pfilePath) throws Exception
  {

    //Abrimos el fichero
    InputStream input = new FileInputStream(pfilePath);
    myWorkbook = WorkbookFactory.create(input);

    //Cargamos la hoja, que es única
    mySheet = myWorkbook.getSheetAt(0);

    //Almacenamos la ruta del fichero
    filePath = pfilePath;

  }
```

```
/**
 * @param      uvus      Uvus del alumno
 * @param      marks     Boolean[] con los resultados en los apartados
 *             (true) correcto o (false) incorrecto
 * @param      mark      Nota final del alumno
 * @return     ErrorControlado Se ha insertado correctamente (false) no (true)
 * @category   Excel Library
 * @version    0.2
 */
public synchronized ErrorControlado UpdateMarks(String uvus,
                                                 Boolean[] marks, float mark)
  {

  ErrorControlado result = null;
  ErrorControlado auxError = null;
  int index = 0;

  try
  {

    // 1.- ¿Existe ya el alumno en el fichero?
    if ((index = ExistUvus(uvus)) == (-1))
    {

      // 2.1.- No existe, creamos la fila y la insertamos al final

      // Creamos una fila. Nunca en la posición 0, que están los títulos
      Row myRow = mySheet.createRow(mySheet.getLastRowNum()+1);

      myRow.createCell(0).setCellType(Cell.CELL_TYPE_STRING);
      myRow.createCell(0).setCellValue(uvus);

      // Insertamos las notas
      int i = 0;
      for (i = 0; i < marks.length; i++)
      {
        myRow.createCell(i+1).setCellType(Cell.CELL_TYPE_BOOLEAN);
        myRow.createCell(i+1).setCellValue(marks[i]);
      }

      // Le ponemos su nota final
      myRow.createCell(i+1).setCellType(Cell.CELL_TYPE_NUMERIC);
      myRow.createCell(i+1).setCellValue(mark);

      // Fijamos las entregas realizadas en 1
      myRow.createCell(marks.length + 2).setCellType(Cell.CELL_TYPE_NUMERIC);
      myRow.createCell(marks.length + 2).setCellValue(1);

    }
    else
    {

      // 2.2.- Existe, actualizamos la fila que ya existe
      Row myRow = mySheet.getRow(index);

      if (myRow != null)
      {

        // Actualizamos las notas
        for (int i = 0; i < marks.length; i++)
        {

          if (myRow.getCell(i+1) == null)
          {
            myRow.createCell(i+1).setCellType(Cell.CELL_TYPE_BOOLEAN);
          }

          myRow.getCell(i+1).setCellValue(marks[i]);
```

```
        }

        // Le ponemos la nota final. No sabemos si existe
        if (myRow.getCell(marks.length+1) == null)
        {
         myRow.createCell(marks.length+1).setCellType(Cell.CELL_TYPE_NUMERIC);
        }

        myRow.getCell(marks.length+1).setCellValue(mark);

        // Obtenemos las entregas hasta el momento
        int deliveries = deliveriesCount(uvus, marks.length);

        if (myRow.getCell(marks.length+2) == null)
        {
         myRow.createCell(marks.length+2).setCellType(Cell.CELL_TYPE_NUMERIC);
        }

        myRow.getCell(marks.length+2).setCellValue(deliveries+1);

      }

    }

    // 2.3.- Escritura en disco de los datos
    auxError = this.writeFile();
    if (auxError.getIsError())
    {
      result = auxError;
    }
    else
    {
      result = new ErrorControlado(false, null);
    }

  }
  catch (Exception e)
  {
    result = new ErrorControlado(true, e);
  }

  return result;

}

/**
 * @param      uvus  Uvus del alumno
 * @param      mark  Nota del trabajo
 * @return     ErrorControlado Se ha insertado correctamente (false) no (true)
 * @category   Excel Library
 * @version    0.3
 */
public synchronized ErrorControlado UpdateMarks(String uvus, float mark)
{

  ErrorControlado result = null;
  ErrorControlado auxError = null;
  int index = 0;

  try
  {

    //1.- ¿Existe ya el alumno en el fichero?
    if ((index = ExistUvus(uvus)) == (-1))
    {

      //2.1.- No existe, creamos la fila y la insertamos al final
```

```
      //Creamos una fila. Nunca en la posición 0, que están los títulos
      Row myRow = mySheet.createRow(mySheet.getLastRowNum()+1);

      myRow.createCell(0).setCellType(Cell.CELL_TYPE_STRING);
      myRow.createCell(0).setCellValue(uvus);

      //Insertamos las notas
      myRow.createCell(1).setCellType(Cell.CELL_TYPE_NUMERIC);
      myRow.createCell(1).setCellValue(mark);

    }
    else
    {

      //2.2.- Existe, actualizamos la fila que ya existe
      Row myRow = mySheet.getRow(index);

      if (myRow != null)
      {

        //La celda de la nota puede no existir
        if( myRow.getCell(1) == null)
        {
          myRow.createCell(1).setCellType(Cell.CELL_TYPE_NUMERIC);
          myRow.createCell(1).setCellValue(mark);
        }
        else
        {
          myRow.getCell(1).setCellValue(mark);
        }

      }

    }

    //2.3.- Escribimos el fichero
    auxError = this.writeFile();
    if (auxError.getIsError())
    {
      result = auxError;
    }
    else
    {
      result = new ErrorControlado(false, null);
    }

  }
  catch (Exception e)
  {
    result = new ErrorControlado(true, e);
  }

  return result;

}

/**
 * @param    uvus     Uvus del alumno
 * @return  -1 si no existe el usuairo, o el índice del uvus
 * @category  Excel Library
 * @version  0.2
 */
public synchronized int ExistUvus(String uvus)
{

  int result = -1;
  int i = 0;
  Cell celda = null;
```

```
  for (Row myRow : mySheet)
  {

    if ((celda = myRow.getCell(0)) != null)
    {

      //Sabemos que la columna 0 es el uvus, tipo String
      if (uvus.equals(celda.getRichStringCellValue().getString()))
      {
        result = i;
      }

    }

    i++;

  }

  return result;

}

/**
 * @param    uvus     Uvus del alumno
 * @param    markColums     Número de columnas de apartados a corregir
 * @return    0 si no existe el alumno, el número de entregas eoc
 * @category  Excel Library
 * @version   0.2
 */
public int deliveriesCount(String uvus, int markColums)
{

  int count = 0;
  int index = ExistUvus(uvus);
  Cell celda = null;

  if ( index != (-1) )
  {

    Row myRow = mySheet.getRow(index);
    if (myRow != null)
    {

      if ((celda = myRow.getCell(markColums+2)) != null)
      {

        count = (int) celda.getNumericCellValue();

      }

    }

  }

  return count;

}

/**
 *
 * @exception IOException Se produce al trabajar con el fichero
 * @return    ErrorControlado Se ha escrito bien el fichero (false) no (true)
 * @category  Excel Library
 * @version   0.2
 */
public synchronized ErrorControlado writeFile()
{
```

```
    ErrorControlado result = null;

    try
    {

      FileOutputStream fileOut = new FileOutputStream(filePath);
      myWorkbook.write(fileOut);
      fileOut.close();

      result =  new ErrorControlado(false, null);

    }
    catch (IOException e)
    {
      result = new ErrorControlado(true, e);
    }

    return result;

  }

}
```

6.4.31 ScriptWorker.java

```
package tfg.lti.Files;

import tfg.lti.Control.ErrorControlado;

import java.io.File;
import java.util.ArrayList;
import java.util.Collections;
import java.io.FilenameFilter;

/**
 *
 * @author     Álvaro Martín Rodrí-guez
 * @version     0.1
 * @category   Script Worker
 */
public class ScriptWorker {

  /**
   *
   * @param   path            Localización de los ficheros a ejecutar
   * @param   rootPath        Carpeta Raíz de la estructura de directorios
   * @param   uvus            uvus del alumno
   * @param   title           Título del trabajo
   * @return              Un Arraylist que contiene booleans para cada apartado
   */
  public ArrayList<Boolean> runExamination (String path, String rootPath,
                            String uvus, String title)// throws IOException
  {

    ArrayList<Boolean> results = new ArrayList<Boolean>();
    ArrayList<String> files = new ArrayList<String>();
    int i = 0;

    // Listamos el directorio
    File dir = new File(path);

    // Creamos un filtro para detectar solo los ficheros que nos interesan
    FilenameFilter filter = new FilenameFilter() {
        public boolean accept(File dir, String name) {
            boolean isAcceptedFile = false;

            isAcceptedFile = (name.endsWith(".sh")
```

```
                                || name.endsWith(".bat")
                                || name.endsWith(".exe")
                                || name.endsWith(".out"));

            return isAcceptedFile;
        }
    };

    if (dir.exists())
    {

      File[] ficheros = dir.listFiles(filter);

      // 1.- Vamos a obtener los ficheros
      for(i = 0; i < ficheros.length; i++)
      {

        // Si es un fichero y no un directorio
        if(ficheros[i].isFile() == true)
        {

          // Si no es un duplicado del SO, un temporal
          if(ficheros[i].getName().endsWith("~") == false)
          {
            files.add(ficheros[i].getAbsolutePath());
          }

        }

      }

      // Los ordenamos
      Collections.sort(files);

      // 2.- Los ejecutamos (pisamos las rutas con los resultados)
      for(i = 0; i < files.size(); i++)
      {
        results.add(i, execScript(files.get(i), rootPath, uvus, title));
      }

    }

    return results;

}

/**
 *
 * @param   script          Localización del fichero a ejecutar
 * @param   rootPath        Carpeta Raíz de la estructura de directorios
 * @param   uvus            uvus del alumno
 * @param   title           Título del trabajo
 * @param   file            Nombre del fichero del trabajo
 * @return                  Un boolean según la ejecución del Script
 */
public ErrorControlado runPreprocess (String script, String rootPath,
                                      String uvus, String title, String file)
{

    ErrorControlado result = null;
    Process program;

    try
    {
      program = Runtime.getRuntime().exec(new String[]{script, rootPath,
                                                       uvus, title, file});
      program.waitFor();

      if ( program.exitValue() == 0)
```

```
        {
          result = new ErrorControlado(false, null);
        }
      }
      catch (Exception e)
      {
        System.out.println(e.getMessage());
        result = new ErrorControlado(true, e);
      }

      return result;

    }


    /**
     *
     * @param   file  Localización del fichero a ejecutar
     * @param   rootPath        Carpeta Raíz de la estructura de directorios
     * @param   uvus            uvus del alumno
     * @param   title           Título del trabajo
     * @return  true si el apartado es correcto, false si suspenso
     */
    private boolean execScript (String file, String rootPath,
                                String uvus, String title)
    {

      boolean result = false;
      Process program;
      String pathOfAlumFiles = rootPath + uvus + "/" + title + "/VersionActual/";

      try
      {
        program = Runtime.getRuntime().exec(new String[]{file,
                                                         pathOfAlumFiles});
        program.waitFor();

        if ( program.exitValue() == 0)
        {
          result = true;
        }
      }
      catch (Exception e)
      {
        System.out.println(e.getMessage());
      }

      return result;

    }

  }
```

6.4.32 Ficheros de depencencias

- Correspondiente a la librería BLTI Sandwich encontramos:
  - blti-sandwich.jar
- La librería Commons FileUpload cuenta con:
  - commons-fileupload-1.3.1.jar
- Para Apache IO:
  - commons-io-2.4.jar

- OAuth se encuentra dividida en tres ficheros:
    - oauth-20100527.jar
    - oauth-consumer-20100527.jar
    - oauth-provider-20100527.jar
- Por último, Apache POI cuenta con:
    - poi-3.11-20141221.jar
    - poi-ooxml-3.11-20141221.jar

6.4.33 LTI_TC_Deploy.sh

```
#!/bin/bash
# Álvaro Martín Rodríguez
# Script de despligue de la aplicación TC LTI

function menu()
{

  clear

  echo "##### Despliegue TC  APP #####"
  echo "####                     ####"
  echo "#    1.- Copia de ficheros   #"
  echo "#    2.- Editar propiedad    #"
  echo "#    3.- Iniciar servidor    #"
  echo "#    4.- Salir               #"
  echo "####                     ####"
  echo "################################"

  echo ''
  echo -n '¿Qué desea hacer?: '
  read O

  case $O in
    1 )
      copia
      ;;
    2 )
      propiedad
      ;;
    3 )
      iniciar
      ;;
    4 )
      clear
      exit
      ;;
    * )
      menu
      ;;

  esac

}

function volverMenu()
{

  echo ''
  echo -n '¿Quiere volver al menú? (S/N): '
  read O

  if [ "$O" = "S" ] || [ "$O" = "s" ]; then
    menu
```

```
  else
    clear
    #exit
  fi

}

function copia()
{

  echo ''
  echo -n '¿Quiere copiar los ficheros a la carpeta? (S/N): '
  read A

  if [ "$A" = "S" ] || [ "$A" = "s" ]; then

    echo ""
    echo "1.- Recargando el fichero LTI_TC_Deploy_Vars..."

    #Carga del fichero de variables
    source ./LTI_TC_Deploy_Vars.txt

    echo "2.- Carpeta destino: $deployPath"

    if [ ! -d "$deployPath" ]; then

      echo "2.1.- Creando carpeta destino..."
      mkdir $deployPath &>/dev/null

    else

      echo "2.2.- Eliminando contenido previo..."
      rm -r "$deployPath"*  &>/dev/null

    fi

    # Copiamos el contenido de Server a deployPath
    echo "3.- Copiando aplicación..."
    cp -R ./TC_Files/* $deployPath

    # Borramos los ficheros temporales (File.ext~)
    echo "4.- Eliminando ficheros temporales..."
    rm "$deployPath/Scripts/$projectName/*.sh~" &>/dev/null

    echo ""
    echo "Fin."

  else
    echo 'Se ha cancelado la copia de los ficheros'
  fi

  volverMenu

}

function propiedad()
{

  echo ''
  echo -n '¿Quiere cambiar la propiedad de la carpeta? (S/N): '
  read A

  if [ "$A" = "S" ] || [ "$A" = "s" ]; then

    echo ""
    echo "1.- Cambiando propietario de la carpeta destino: $deployPath"

    #Damos permisos de ejecución a los scripts
```

```
    chown -R $serverUser $deployPath

    echo "2.- Ahora la carpeta pertenece al usuario $serverUser"

    echo ""
    echo "Fin."

  else
    echo 'Se ha cancelado el cambio de la propiedad de los ficheros'
  fi

  volverMenu

}

function iniciar()
{

  aviso='(AVISO: esto reiniciará el servidor en caso de que esté encendido)'

  echo ''
  echo -n "¿Quiere iniciar el servidor tomcat? $aviso (S/N): "
  read A

  if [ "$A" = "S" ] || [ "$A" = "s" ]; then

    echo ""
    echo '1.- Deteniendo servidor tomcat...'
    su -c "service $serverUser stop"

    echo '2.- Actualizando componente Netfilter...'
    iptables -t filter -I INPUT 1 -p tcp --dport 8080 -j ACCEPT

    echo '3.- Iniciando servidor tomcat...'
    su -c "service $serverUser start"

    echo ""
    echo "Fin."

  else
    echo 'Se ha cancelado el inicio del servidor Tomcat'
  fi

  volverMenu

}

#Carga del fichero de variables
source ./LTI_TC_Deploy_Vars.txt

#Lanzamiento del menu
menu
```

6.4.34 LTI_TC_Deploy_Vars.txt

```
# LTI_TC_Deploy_Vars.txt
# Álvaro Martín Rodríguez
# Fichero de variables de despliegue para el emulador del TC

# Estas lineas son comentarios

# Ruta de despliegue

deployPath=/var/lib/tomcat7/webapps/TC/

# ------------------

# Nombre del usuario que ejecuta el servidor
```

```
serverUser=tomcat7

# ------------------
```

6.4.35 TC.html

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Emulador LMS</title>
  <link rel="stylesheet" type="text/css" href="Resources/TC.css">
</head>

<body>

  <h2>Emulador TC - LMS</h2>

  <table class="centered_item">
    <tr>
      <td>
        <a href="./TC_Student.jsp" >
          <image class="link" src="./Resources/Tool_icon.png"/>
          <p class="text_link">Herramienta Alumno</p>
        </a>
      </td>
      <td>
        <a href="./TC_Teacher.jsp" >
          <image class="link" src="./Resources/Tool_icon.png"/>
          <p class="text_link">Herramienta Profesor</p>
        </a>
      </td>
    </tr>
  </table>

</body>

</html>
```

6.4.36 TC_Student.jsp

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

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<%@page import="java.util.Map"%>
<%@page import="java.util.List"%>
<%@page import="blackboard.blti.consumer.*"%>
<%@page import="blackboard.blti.message.*"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Emulador LMS</title>
  <script src="./Resources/TC.js"></script>
</head>
```

```
<body>

  <form action="http://192.168.1.135:8080/Servlet/LTIApp" method="POST">

  <%!

    /* Variables que se emplearán en el mensaje LTI */
    String name = "Nombre Alumno";
    String UVUS = "uvusAlumno";

    String TC_Key = "claveIdentificativa";
    String TC_Shared_Secret = "gitt1fp2sc";

    String TC_ResourceID = "identificadorRecurso1";

  %>

  <%

    BLTIMessage msg = new BLTIMessage( TC_Key );
    msg.getResourceLink().setId( TC_ResourceID );
    msg.getUser().setId( UVUS );
    msg.getUser().setFullName( name );
    msg.getUser().addRole( new Role( "Learner" ) );

    BLTIConsumer consumer = new BLTIConsumer( "POST",
                            "http://192.168.1.135:8080/Servlet/LTIApp", msg);
    consumer.sign( TC_Shared_Secret );

    List<Map.Entry<String,String>> launchParams2 = consumer.getParameters();

    for (Map.Entry<String, String> entry : launchParams2)
    {

      out.println("<input type=\"hidden\" name=\"" + entry.getKey()
                  + "\" value=\"" + entry.getValue() + "\">");

    }

  %>

  </form>

</body>

</html>
```

6.4.37 TC_Teacher.jsp

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

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<%@page import="java.util.Map"%>
<%@page import="java.util.List"%>
<%@page import="blackboard.blti.consumer.*"%>
<%@page import="blackboard.blti.message.*"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Emulador LMS</title>
  <script src="./Resources/TC.js"></script>
```

```
</head>

<body>

  <form action="http://192.168.1.135:8080/Servlet/LTIApp" method="POST">

  <%!

    /* Variables que se emplearán en el mensaje LTI */
    String name = "Nombre Profesor";
    String UVUS = "uvusProfesor";

    String TC_Key = "claveIdentificativa";
    String TC_Shared_Secret = "gitt1fp2sc";

    String TC_ResourceID = "identificadorRecurso2";

  %>

  <%

    BLTIMessage msg = new BLTIMessage( TC_Key );
    msg.getResourceLink().setId( TC_ResourceID );
    msg.getUser().setId( UVUS );
    msg.getUser().setFullName( name );
    msg.getUser().addRole( new Role( "Instructor" ) );

    BLTIConsumer consumer = new BLTIConsumer( "POST",
                            "http://192.168.1.135:8080/Servlet/LTIApp", msg);
    consumer.sign( TC_Shared_Secret );

    List<Map.Entry<String,String>> launchParams2 = consumer.getParameters();

    for (Map.Entry<String, String> entry : launchParams2)
    {

      out.println("<input type=\"hidden\" name=\"" + entry.getKey()
                  + "\" value=\"" + entry.getValue() + "\">");

    }

  %>

  </form>

</body>

</html>
```

6.4.38 TC.css

```
/* ELEMENTS */
body
{
  text-align: center;
  background-color: #ffffff;
}

h2
{
  background-color: #000000;
  color: white;
}

/* CLASSES */
.link
{
```

```
  width: 100px;
  height: 100px;
  margin: auto;
  text-align:center;
  padding-top: 10px;
}

.text_link
{
  margin: auto;
  text-align:center;
  padding-top: 10px;
}

.centered_item
{
  width: 45%;
  text-align:center;
  margin: auto;
}
```

6.4.39 TC.js

```
window.onload = function()
{
  window.document.forms[0].submit();
};
```

6.4.40 Ficheros de dependencias del TC

- Como fichero correspondiente a la librería BLTI Sandwich encontramos:
    - blti-sandwich.jar
- Por otro lado, correspondientes a OAuth encontramos los ficheros:
    - oauth-20100527.jar
    - oauth-consumer-20100527.jar
    - oauth-provider-20100527.jar