Acceso de usuario a aplicaciones web, Global Login

Wisam Alsalti

Resumen— Desde 1969, el momento en que se llevó a cabo la primera conexión entre dos ordenadores de Stanford y UCLA, y el nacimiento de ARPANET, Internet ha logrado un incremento espectacular en todos los ámbitos (número de dispositivos conectados, tráfico de datos, número de usuarios, etc.) llegando a alcanzar los 2.400 millones de usuarios en todo el mundo (2013), logrando de esta forma un incremento espectacular en pocos años. No obstante, este incremento alcanzado no es nada considerable con los datos previstos para el futuro, ya que se prevé que en 2016 habrá 3.400 millones de usuarios de Internet; es decir, aproximadamente un 45% de la población mundial.

La motivación principal de este proyecto se centra en el elevado número de usuarios que ha alcanzado y que alcanzará en un futuro Internet, tratar de proporcionar a la sociedad una herramienta compartida por distintas aplicaciones web y con la cual facilitar a los usuarios el proceso de la autenticación y de acceso. En conclusión, autenticarse una única vez y acceder a diferentes aplicaciones de forma transparente y con sus perfiles pertinentes.

Palabras claves— Arquitectura Single Sign-On, Sistema Centralizado de Autenticación, Infraestructura de Clave Pública, aplicaciones Web, JSON Web Token, entorno de desarrollo integrado.

Abstract— Since 1969, when it took place the first connection between two computers from Stanford and UCLA, and the birth of ARPANET, Internet has achieved a spectacular increase in all areas (number of connected devices, data traffic, number of users, etc.) reaching 2,400 million users worldwide (2013), achieving in this way a dramatic rise in a few years. However, this increase is not worthy of consideration compared to the amount expected for the future, since it is expected that in 2016 there will be 3,400 million Internet users; that is, about 45% of the world population.

The main motivation of this project focuses on the large number of users that Internet has reached and will reach in the future, trying to supply a tool shared by different web applications and which provides users with the process of authentication and access. In conclusion, authenticating once and accessing different applications transparently and with their corresponding profiles.

Index Terms— Single Sign-On architecture, Centralized Authentication System, Public Key Infrastructure, web applications, JSON Web Token, integrated development environment.

1 Introducción

En esta primera sección se recogerá toda la información relacionada con el contenido del documento. Los siguientes apartados introducirán al lector al propósito general y a los objetivos principales a lograr mediante este proyecto.

1.1 Contenido del artículo

Este artículo está dividido en varias secciones. En esta primera sección el lector puede apreciar la temática principal del proyecto, y se expone una breve descripción de la propuesta inicial, los límites presentes para el desarrollo, finalmente, el objetivo principal junto a los subobjetivos que lo componen.

La segunda sección muestra el estado del arte del proyecto, donde se explicará la situación actual del mercado y las soluciones que aportará el desarrollo del proyecto a la problemática planteada.

En la tercera sección del documento se mostrará la diferencia principal entre el concepto de autenticación y autorización, con la finalidad de aclarar el significado de dichos conceptos antes de seguir con la lectura de las secciones siguientes.

En la cuarta sección se profundiza más en el estudio de la posible solución a desarrollar; presentando las ventajas y desventajas que supone utilizar dicho sistema y las distintas tipologías y arquitecturas que ofrece.

En la quinta sección se puede apreciar la solución finalmente seleccionada junto con el sistema de comunicación que más se adapta a las necesidades del proyecto.

Seguidamente, en la sexta sección, se explicará la metodología principal que se ha escogido para llevar a cabo el proyecto y sus principales ventajas e inconvenientes.

En la séptima sección se mostrará de forma detallada la planificación inicial del proyecto y la asignación de las distintas tareas a realizar en las versiones pertinentes.

La octava sección detallará el proceso de desarrollo llevado a cabo; explicando desde las herramientas y lenguajes utilizados hasta el diseño final de la aplicación web.

En la novena sección se presentarán las conclusiones y

los resultados obtenidos tras la finalización y el cierre del desarrollo de la aplicación. Esta misma sección también mostrará los principales cambios realizados en la planificación inicial y sus respectivos motivos.

Finalmente, los siguientes apartados irán destinados a los agradecimientos y bibliografía del documento.

1.2 Descripción de la propuesta

La propuesta inicial que fue presentada al comienzo del proyecto trataba de lograr desarrollar una herramienta de autenticación capaz de ser utilizada de forma fácil y sencilla por distintas aplicaciones web, con el fin de externalizar el proceso de autenticación. Dicha herramienta debería parametrizar su contenido para adaptarse a las necesidades de cada aplicación.

1.3 Límites del proyecto

Los límites principales del proyecto se centran básicamente en dos aspectos.

Primeramente el equipo físico en el cual se llevará a cabo el desarrollo del proyecto, dicho equipo consta de tan solo un ordenador portátil de uso corriente que se utilizará tanto como centro de desarrollo, como servidor y como gestor de base de datos. Por lo tanto se implementará una versión modelo que remarcará las funcionalidades principales desarrolladas y el cumplimiento de los requisitos solicitados en la propuesta.

El otro principal límite afecta tanto al tiempo como al personal de desarrollo impuesto, ya que el tiempo de entrega del proyecto no admite retrasos y el personal de desarrollo es de únicamente una persona. Por lo tanto la planificación del proyecto ha de ser estrictamente detallada y se ha de seguir fielmente para lograr un resultado correcto.

1.4 Objetivo principal

El objetivo principal del proyecto es proporcionar una aplicación web que facilite al usuario final el acceso a distintas aplicaciones web con un único sistema de autenticación, es decir, una herramienta centralizada que gestione tanto el sistema de acceso como el de perfiles.

Dicho objetivo principal se divide en tres subobjetivos, que a su vez están ordenados cronológicamente a lo largo del desarrollo del proyecto.

El primer subobjetivo y más importante para asegurar una buena planificación y cumplimiento de las funcionalidades básicas, es el estudio e implementación de un sistema de comunicación seguro entre las aplicaciones web y nuestra aplicación principal. Más adelante se expondrán las diferentes opciones estudiadas y la solución finalmente escogida.

El segundo subobjetivo, y una vez escogido el sistema de comunicación más idóneo, trata de aplicar dicho sistema y desarrollar todas las funcionalidades tomadas en la lista de requisitos.

Finalmente el tercer subobjetivo trata de presentar al usuario final una interfaz agradable e intuitiva; logrando una máxima transparencia en el proceso de comunicación con las distintas aplicaciones web.

2 Estado del arte

En esta sección se hará una introducción a los diferentes conceptos técnicos que se verán a lo largo del artículo.

Los elementos más importantes presentes en la mayoría de los sistemas de autenticación de todo proveedor de servicio son el nombre de usuario y la contraseña. Dichos elementos en la mayoría de los casos varían de un servicio a otro, convirtiéndose hasta en algunos casos en obstáculos para el usuario final, ya que éste se ve incapaz de recordar todas sus credenciales necesarias.

En este caso, las soluciones más comunes a las que llegan la mayoría de los usuarios pueden variar entre, utilizar credenciales poco seguras con el fin de recordarlas fácilmente, anotar los distintas credenciales en un papel o incluso utilizar las mismas para el acceso a distintos proveedores de servicio, primando en todo momento la usabilidad ante la propia confidencialidad y seguridad [1].

La solución más eficaz y sencilla ante este escenario es realizar e implementar un sistema de autenticación centralizado: Single Sign-On (SSO). Se trata de una técnica que pretende autenticar al usuario una única vez y automáticamente darle acceso a las distintas aplicaciones suscritas si es necesario, sin la necesidad de solicitar las credenciales continuamente.

En conclusión, los SSO son sistemas que aumentan de forma considerable la usabilidad de las aplicaciones web y al mismo tiempo centralizan el manejo del sistema de autenticación.

Más adelante se explicarán las diferentes posibles arquitecturas y tipologías de SSO y finalmente el sistema escogido que mejor se adapta a las necesidades del proyecto.

3 Autenticación y autorización

Antes de seguir con los apartados siguientes, cabe destacar y aclarar la diferencia principal entre estos dos términos. [2]

Por un lado se denomina autenticación al proceso de comprobación de las credenciales que introduce el usuario con la finalidad de identificarse ante el proveedor de servicios.

Por otro lado se encuentra el concepto de autorización, proceso que generalmente tiene lugar una vez realizado de forma correcta y con éxito un intento de autenticación. La autorización trata de controlar y establecer los distintos tipos de contenidos y recursos a los que los diferentes usuarios o grupos de usuarios pueden acceder.

En líneas generales el proceso de autenticación normalmente es llevado a cabo con la introducción de credenciales (nombre de usuario y contraseña), y el proceso de autorización es gestionado mediante un sistema de

- *E-mail de contacto: wisam532@gmail.com*
- *Mención realizada: Ingeniería de Tecnologías de la Información.*
- *Trabajo tutorizado por: Dr. Joaquim Borges Ayats (Departamento de Ingeniería de la Información y Comunicación).*
- *Curso 2013/14*

perfiles que administra y controla el contenido web.

4 SINGLE SIGN-ON

Como se ha mencionado anteriormente, la función principal de los sistemas SSO es apoyar de forma automática el proceso de autenticación de los proveedores de servicio asociados, incorporando también otras funciones como el registro o el bloqueo temporal por reintentos.

En los siguientes apartados se presentarán las distintas tipologías existentes y las principales arquitecturas que implementan las aplicaciones SSO del mercado hoy en día.

4.1 Tipologías

En resumen, conceptualmente se pueden distinguir dos tipos principales de SSO:

- **Delegated**: este tipo de solución da lugar a una externalización de las funcionalidades de autenticación y control de acceso a otro proveedor de servicio preseleccionado. Por lo tanto, únicamente los servicios externos anteriormente seleccionados serán utilizados para el propósito de autenticación. Un ejemplo de este tipo de solución es delegar el sistema de acceso a aplicaciones como Facebook o Twiter.
- **Federated**: esta otra solución pretende que el usuario final utilice cualquier cuenta que tenga con el fin de autenticarse, siempre y cuando ésta sea compatible y el cliente tenga plena confianza con el proveedor de identidad. Un ejemplo que representa esta solución es OpenID Connect.

4.2 Arquitecturas

A grandes rasgos existen dos importantes arquitecturas que destacan por encima de las demás existentes en el mercado:

- **Enterprise SSO**: fundamentalmente fue diseñado para dar soporte a prácticamente la totalidad de las aplicaciones que un usuario pueda necesitar, es decir, tanto aplicaciones web como aplicaciones de escritorio, como aplicaciones móviles, etc.
  Tiene un funcionamiento no intrusivo que detecta y almacena automáticamente las credenciales una vez introducidas, con la finalidad de que la siguiente vez que el mismo usuario acceda a la aplicación, éste no tenga que introducir de nuevo dichas credenciales. Además de esta funcionalidad, algunas aplicaciones basadas en esta arquitectura implementan la posibilidad de reconocer la pantalla de cambio de contraseña y, de este modo, actualizar las credenciales anteriormente almacenadas de forma transparente al usuario.
- **Web SSO**: el planteamiento de esta segunda arquitectura difiere de la anteriormente mencionada. Esta arquitectura es enfocada principalmente a las aplicaciones basadas en la web. Las políticas de autenticación y acceso se basan en un servidor central que determina quién puede tener acceso a las diferentes aplicaciones web. Esta funcionalidad se puede realizar tanto mediante la redirección del usuario al servidor de autenticación, como mediante el uso de un servidor web proxy que identifica al usuario antes de acceder a las terceras aplicaciones.

4.3 Ventajas y desventajas

Seguidamente se presentarán las principales ventajas y desventajas que surgen al implantar un sistema de SSO.

4.3.1 Ventajas

- Minimiza de forma considerable el requisito de autenticación por parte de múltiples aplicaciones.
- El usuario final no tiene la obligación de memorizar un número elevado de credenciales. Este hecho añade usabilidad a las aplicaciones y una mejor experiencia del usuario.
- Reducción de costes por parte de las aplicaciones asociadas a la hora de la gestión y mantenimiento, ya que externalizan todo el sistema de autenticación.
- Una correcta implementación puede mejorar de forma considerable la seguridad y productividad de las terceras aplicaciones asociadas.

4.3.2 Desventajas

- El apoyo general por parte de las aplicaciones web no es favorable, ya que muchas empresas y aplicaciones prefieren tener los datos de sus usuarios en servidores propios y poder explotarlos a su antojo.
- El hecho de centralizar todo el procedimiento de autenticación puede incrementar el número de ataques por parte de *hackers* y los intentos de intrusión.
- La mala implementación o un equipo físico inadecuado puede provocar tanto problemas técnicos como de latencia.

Como conclusión a esta sección se puede afirmar que la solución y arquitectura que más se adapta a las necesidades del proyecto es la basada en Web SSO. Por otro lado la tipología que se ha utilizado para el desarrollo de la aplicación y la comunicación fue la tipología Delegated, ya que se pretendía implementar una aplicación web con control de acceso, accesible desde distintos servicios web y que además fuera parametrizable para adaptarse al contenido solicitado por las aplicaciones terceras.

Finalmente cabe destacar que actualmente existen diferentes servicios de SSO, tanto de libre licencia como de pago, pero finalmente se ha optado a implementar un pequeño sistema de comunicación propio basado en tokens que simula el funcionamiento de éstos.

5 SISTEMA DE COMUNICACIÓN ESCOGIDO

Como se ha dicho a modo de resumen en el apartado

anterior, la solución adaptada e implementada es la basada en un sistema de Web SSO de tipología Delegated. En esta sección se presentarán los distintos sistemas de comunicación entre dominios estudiados y finalmente el sistema de comunicación escogido.
Además se mostrará un ejemplo del funcionamiento general del proceso de autenticación resultante.

5.1 Basado en cookies

Una cookie es una variable que los servidores de aplicaciones web envían al navegador del usuario con la finalidad de almacenar información. Dicha información habitualmente sirve para diferenciar los usuarios, proporcionarles distintos recursos y personalizar el aspecto de la página web.

Por lo tanto son variables que se guardan en el cliente y no en el servidor. Este hecho tiene la desventaja de que el cliente puede alterar su contenido a su antojo e incluso borrarlos o configurar su navegador para no aceptarlos.

Otra desventaja que presentan los cookies es el hecho de que únicamente pueden almacenar cadenas de caracteres, de tamaño limitado, que además viajan en la cabecera de la comunicación entre servidor y cliente. Esto dificulta almacenar en ellas grandes cantidades de datos y ralentiza la comunicación entre servidor y cliente.

Todas estas desventajas no presentan un obstáculo muy grave; sin embargo el principal motivo por el cual estas variables fueron descartadas para su uso en la comunicación es la imposibilidad de compartir dichas cookies entre distintos dominios.

5.2 Basado en variables de sesión

Este otro medio es ampliamente utilizado para la gestión de las sesiones de usuarios. Como en el primer caso, las variables de sesión también tienen la funcionalidad de almacenar información, no obstante, estas variables se guardan en el servidor (en el navegador del cliente se almacena un identificador de dicha sesión mediante cookies). Este hecho tiene la ventaja de que se tiene controlado en todo momento el contenido de las variables; sin embargo, un mal uso de dichas variables implica un exceso de almacenaje en el servidor.

A primera vista esta opción parece ser la idónea, sin embargo, estas variables pertenecen a una relación única entre servidor y cliente y no es posible compartir dicha información entre diferentes dominios.

5.3 Basado en tokens

Esta última opción es la que presenta una solución más flexible y adaptable a las necesidades del proyecto. Dichos tokens pueden almacenar toda la información necesaria para la comunicación entre dominios, sin embargo carecen de sistema de seguridad propio, pueden ser interceptados por terceras personas y utilizados para fines maliciosos.

Como solución al problema de seguridad, todos los token generados serán firmados y encriptados mediante un sistema criptográfico de clave pública [3]. Más adelante se mostrará un ejemplo de token donde se podrá apreciar la estructura principal de un mensaje de éxito de autenticación.

Los tokens se realizan mediante JSON (JavaScript Object Notation) [4], un formato ligero para el almacenamiento y la compartición de datos.

La ventaja principal de este formato es su simplicidad, tanto en su manejo como en su generación, respecto a otros formatos como XML; esto ha dado lugar a la generalización de su uso en casi cualquier ámbito.

Un ejemplo de la estructura de un token de respuesta de autenticación exitosa es el siguiente:

Header
{"alg":"RS256","cty":"text\/plain"}
Payload
{"jti":"0581…0780","iat":140...1981,"exp":140...2567,"usr":"userName","prf":"13"}

Los tokens generados son formados por dos elementos esenciales [5], que contienen como mínimo los atributos explicados:

- Header: contiene la información del algoritmo utilizado para la encriptación y el tipo de datos almacenados en el payload.
- Payload: en este campo es donde se transmite la parte esencial de la comunicación. Primeramente un identificador del propio token, en segundo lugar el tiempo de creación y el tiempo de expiración, y finalmente el nombre del usuario y perfil externo de éste en la aplicación solicitante.

Cabe destacar que una vez generado el contenido del token, éste es encriptado mediante un sistema criptográfico de clave pública.

5.4 Ejemplo de comunicación

En esta sección se expondrá un pequeño ejemplo del funcionamiento de la comunicación entre las distintas aplicaciones y la interacción resultante con el usuario.

En la Figura 1 se puede observar dos intentos de acceso a distintas aplicaciones web por parte de Alice. El primer intento se realiza para la App1 sin haber logrado una autenticación exitosa previamente ①; Alice es redirigida a nuestra aplicación, Global Login, con la finalidad de verificar si se ha autenticado previamente ②.

Como se ha dicho anteriormente Alice no se había autenticado previamente, por lo tanto es redirigida nuevamente a la página principal de *login* ③. Ttras autenticarse correctamente y tras la validación de la sesión por parte de Global Login ④, Alice es redirigida de nuevo a la aplicación de donde ha venido con un token, que validará su autenticación ⑤. Una posible ampliación del proceso explicado anteriormente sería solicitar por parte de App1 un mensaje de confirmación del proceso a Global Login.

El segundo intento de acceso por parte de Alice se realiza a la aplicación App2 ⑥; Alice es redirigida a Global Login para verificar su estado de autenticación ⑦. Como anteriormente ya se había logrado un intento exitoso de autenticación, Global Login redirige de forma automática a Alice con un token, que validará su autenticación, a App2 ⑧. Como se ha explicado en el primer intento, App2 también tiene la posibilidad de enviar un mensaje

de confirmación a Global Login por tal de validar la información proveniente del usuario.

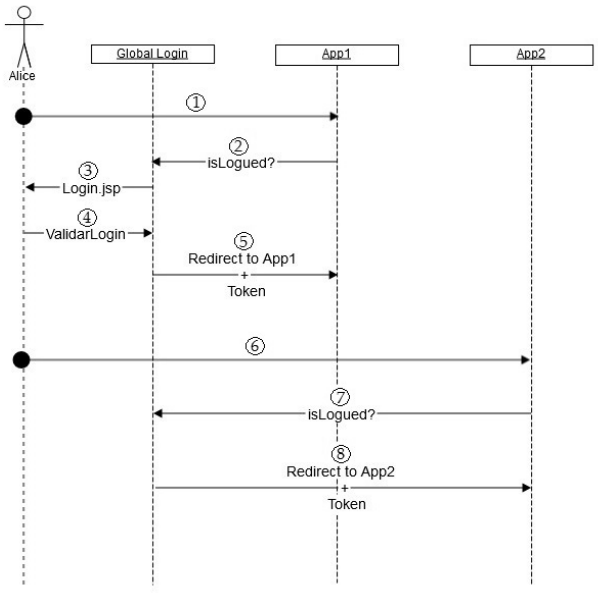


Figura 1. Comunicación básica entre aplicaciones, usuario y Global Login.

Cabe destacar que el usuario final únicamente puede apreciar el redireccionamiento que sufre en el caso de que no se haya autenticado previamente (el paso ③).

6 METODOLOGÍA

En este apartado se mostrarán las principales metodologías estudiadas para llevar a cabo el desarrollo del proyecto y la finalmente escogida.

6.1 Waterfall

Se trata de un modelo tradicional que se caracteriza en que el comienzo de cada nueva etapa debe esperar la finalización de la inmediata anterior.

Las principales ventajas que presenta esta metodología son:

- Simple y fácil de entender y de usar.
- Cada fase se completa una única vez.
- Tiene un buen funcionamiento para pequeños proyectos donde los requerimientos están muy claros.

Por otro lado tiene como inconvenientes las siguientes características:

- La metodología no permite un retroceso a alguna etapa ya previamente finalizada.
- La usabilidad de la aplicación resultante no es visible hasta el final del ciclo de vida del proyecto.

Como conclusión se ha optado por descartar esta opción dada su rigidez a la hora de realizar cambios en la planificación o en los requisitos, y el alto riesgo sumado a la incertidumbre que genera el hecho de no ver ciertos resultados de la aplicación hasta el final del ciclo del proyecto.

6.2 Scrum

Es una metodología de carácter ágil, flexible e iterativo que últimamente se utiliza de forma muy frecuente para el desarrollo de proyectos relativamente pequeños. Este hecho fue el más llamativo a la hora de considerar esta opción.

Sin embargo, tras estudiar con más profundidad la propia metodología y sus inconvenientes se ha llegado a la conclusión de que no se adapta a las necesidades del proyecto por las siguientes razones:

- Es una metodología pensada para el trabajo en equipo.
- Plantea un problema esencial con respecto a los entregables, ya que el proyecto está restringido por una fecha de entrega final que no es posible modificar.

6.3 Incremental

Se trata de una metodología que ofrece un término medio entre el modelo de Waterfall y el modelo Scrum respecto la flexibilidad que presenta.

Básicamente es una evolución del primer modelo, y que pretende solucionar los problemas y carencias que el anticuado modelo no logra superar.

Dicho modelo pretende comenzar la implementación del proyecto con una funcionalidad básica y mínimamente funcional; iterativamente mejorar y añadir nuevas funcionalidades.

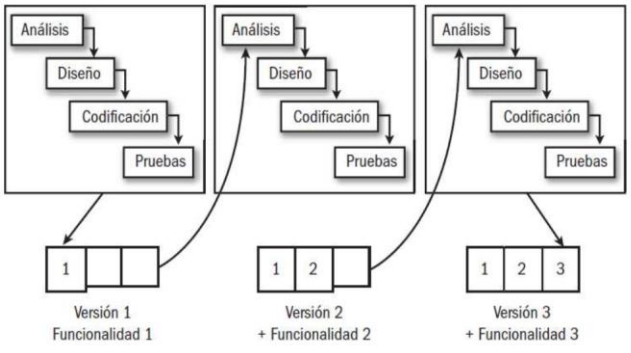


Figura 2. Representación gráfica del modelo incremental.

En conclusión, esta metodología representa un término medio respecto las dos anteriormente citadas y ha sido escogida por los motivos siguientes:

- Visualización de resultados que representan las funcionalidades más importantes al poco tiempo de comenzar la implementación.
- El hecho de comenzar las primeras iteraciones con los objetivos más críticos favorece desarrollar una aplicación más consistente, ya que se les realizan más pruebas.
- Las posibilidades de riesgo de fracasar o no alcanzar ciertas entregas son bajas.

Como apunte final a esta sección, cabe destacar que durante el transcurso de cada versión se llevarán a cabo las siguientes tareas: Actualización de la visión general, análisis de requisitos, controles de seguimiento y planifi-

cación, implementación y finalmente las pruebas pertinentes.

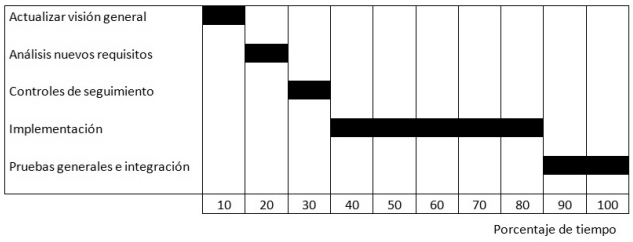


Figura 3. Desglose general de las tareas en cada versión.

7 ORGANIZACIÓN Y PLANIFICACIÓN

En esta sección se mostrará la identificación y subdivisión en objetivos funcionales y no funcionales llevada a cabo durante la primera etapa del proyecto, y la planificación temporal diseñada para lograr el cumplimiento de todas sus tareas.

Después de la entrevista con el cliente, rol que representaba el tutor del proyecto, la toma de requisitos y funcionalidades que ha de cumplir la aplicación final, se llegó a un acuerdo que incluía los objetivos funcionales y no funcionales que se describen en las siguientes secciones.

7.1 Objetivos funcionales

Los objetivos funcionales se han dividido según los entregables a producir, es decir, según las páginas web finales que se han acordado realizar.

Como apunte, cabe destacar que algunas funcionalidades fueron modificadas o rechazadas tras su pertinente estudio de viabilidad. Dichos cambios, junto a las razones que los ocasionaron, se explicarán en el apartado de Resultados.

7.1.1 Pantalla de *login*

Esta pantalla ha de incluir las funcionalidades siguientes:

- A.1 La posibilidad de recordar la contraseña del usuario en el navegador. (Funcionalidad modificada)
- A.2 Incluir un apartado de *Hint* para ayudar al usuario a recordar sus credenciales en caso de necesidad.
- A.3 Implementar un apartado de envío de usuario/contraseña vía email en caso de que el usuario lo necesite. (Funcionalidad modificada)
- A.4 Implementar un sistema de bloqueo temporal parametrizable (donde las distintas aplicaciones pueden escoger activarlo o no), que se dispare después de un número de intentos fallidos por parte del usuario. (Tanto el número de intentos, como el tiempo que permanece el usuario bloqueado es personalizable con los parámetros que la aplicación envía).
- A.5 Enviar automáticamente un aviso mediante email informando de la situación de bloqueo.

7.1.2 Pantalla de configuración de cuenta de usuario

Esta pantalla únicamente tiene la siguiente funcionalidad:

- B.1 Implementar las funcionalidades de cambiar contraseña y *hint* del usuario.

7.1.3 Pantalla de administrador

Esta pantalla será accesible únicamente por usuarios con perfil interno administrador. Más adelante se explicará el sistema de perfiles, tanto interno como externo, que se ha llevado a cabo.

- C.1 Desarrollar un sistema de asignación de perfiles a los distintos usuarios.
- C.2 Implementar un sistema que permita modificar las suscripciones de los usuarios, bloquear dichas suscripciones y suscribir manualmente usuarios a aplicaciones que lo requieran. Esta última funcionalidad es sumamente importante para las aplicaciones con registro cerrado, ya que es accesible únicamente vía administrador.

7.1.4 Pantalla de registro

- D.1 Incluir el sistema de registro en la pantalla de *login*. Las distintas aplicaciones pueden escoger si dan a sus usuarios la capacidad de poder registrarse libremente o no.

7.1.5 Funcionalidades extra

Seguidamente se exponen las funcionalidades extra que se han añadido para mejorar la usabilidad y la apariencia de la aplicación y que no estaban descritas en la toma de requisitos inicial:

- E.1 En la pantalla del administrador se ha añadido una tabla donde se representa un listado de todos los usuarios de la aplicación con la información vital que pueda necesitar el administrador. Dicha tabla se puede ordenar tanto por columnas como por multi-columna (manteniendo pulsada la tecla *shift* y seleccionando las columnas).
- E.2 En la pantalla de administrador se ha añadido un formulario de filtro, que permite al administrador gestionar de forma más simple y rápida los datos de los usuarios y aplicaciones.
- E.3 Se ha implementado una pantalla *logued*, de bienvenida al usuario una vez haya conseguido autenticarse de forma correcta. En esta pantalla se muestran las distintas aplicaciones web a las que el usuario está suscrito.
- E.4 En la pantalla de administrador se ha añadido la funcionalidad de bloquear temporalmente a usuarios. Además se ha añadido la opción de borrar suscripción del usuario o eliminar por completo al usuario y sus suscripciones.

Seguidamente se mostrará una tabla que recoge la prioridad asignada a cada objetivo funcional. Las funcionalidades extra se realizaron al final del desarrollo de la aplicación como un aporte opcional más, por lo tanto, no tienen asignadas ningún tipo de prioridad.

Crítico	Prioritario	Secundario
A.2, A.3, D.1	A.1, B.1, C.1, C.2, D1	A.4, A.5

Tabla 1. Asignación de prioridades a los objetivos funcionales.

La implementación y desarrollo de las distintas funcionalidades del proyecto se llevarán a cabo en un orden, que favorecerá la finalización de las tareas más prioritarias. Por lo tanto primeramente se llevarán a cabo las actividades críticas, en segundo lugar las prioritarias y finalmente las tareas secundarias.

Una vez realizadas todas las funcionalidades exigidas en la toma de requisitos por el cliente, se llevaron a cabo las funcionalidades extra, para mejorar el aspecto y la experiencia final del usuario.

7.2 Objetivos no funcionales

En este apartado se exponen los objetivos y tareas que no producen de forma directa un entregable, por lo tanto, se velará durante el transcurso de todo proyecto su cumplimiento:

- Resistencia contra las vulnerabilidades más comunes de la web, realizando distintas pruebas de intrusión con el fin de evitar ataques como Cross-Site Scripting, Inyecciones SQL, Inyección de código, etc. Esta tarea se llevará a cabo durante la etapa de pruebas e integración de cada versión a entregar.
- Viabilidad legal del proyecto, cumpliendo con las distintas leyes relativas a Internet en general y a las del ámbito web en particular; entre ellas, y más importante, se encuentra la ley Orgánica de Protección de Datos. [6]
- Aunque la propuesta del proyecto se centra en la parte más técnica y funcional de la aplicación, se intentará implementar una interfaz agradable y sencilla para el usuario final.

7.3 Asignación a versiones y planificación temporal

Seguidamente se expone la subdivisión versiones, las cuales han de generar los productos entregables al final de cada etapa, siguiendo en todo momento la metodología incremental descrita anteriormente:

- **Versión 0.1**: Primera versión mínimamente funcional que incluye la implementación de una base de datos, la pantalla de registro y una pantalla de *login* con las características más básicas. (2 semanas)
- **Versión 0.2**: Añadir a la pantalla de *login* la funcionalidad de *hint;* la posibilidad de recuperar la contraseña y el nombre de usuario vía email. (1,5 semana)
- **Versión 0.3**: Incluir en la pantalla de login la funcionalidad de recordar la sesión del usuario, opción de recordar credenciales. (1,5 semana)
- **Versión 0.4**: Implementar la pantalla de administrador y desarrollar el sistema de gestión de perfiles. (2 semanas)
- **Versión 0.5**: Implementar un sistema de bloqueo temporal y enviar una notificación al usuario pertinente informando de dicho bloqueo. (1,5 semana)

A parte de las versiones entregables de la propia aplicación, se ha establecido un tiempo extra para la realización de las siguientes tareas:

- **Investigación y diseño previo**: donde se realizó un estudio previo de los distintos lenguajes de programación web y el diseño básico de la base de datos necesaria. (1 semana)
- **Pruebas y correcciones finales**: mientras va avanzando el proyecto se realizarán las distintas pruebas necesarias para cada versión finalizada, no obstante, se reserva un intervalo de tiempo para la realización de pruebas de intrusión más específicas y la corrección de fallos en caso de necesidad. (1,5 semana)
- **Maquetación y diseño**: Entre las versiones 0.3 y 0.4 se han llevado a cabo las principales tareas de maquetación y diseño web. (1 semana)
- **Documentación y reserva**: La documentación del proyecto se realizará de forma gradual mientras vaya avanzando el proyecto, aun así, se reserva un intervalo de tiempo extra al final del proyecto como recurso temporal de contingencia en caso de extrema necesidad. (+1 semana)

En el apéndice A1 y A2 se muestran, de forma resumida la planificación inicial establecida en las etapas iniciales del proyecto, y la planificación final resultante.

Como resultado de todos los apartados anteriores, se puede calcular que la duración final del proyecto es de 13 semanas laborables, donde en cada semana se dedicará aproximadamente 15 horas de trabajo, dando como resultado final 195 horas totales.

En conclusión, la fecha de inicio de la implementación del proyecto se marcó para el **24 de febrero de 2014**, y la fecha de la entrega final es el **6 de junio de 2014**.

8 DESARROLLO

Esta sección mostrará los conceptos principales del proceso de desarrollo del proyecto; analizando las distintas herramientas escogidas y sus conceptos más importantes.

8.1 Herramientas y lenguajes utilizados

Este apartado se ha dividido en dos partes. La primera explicará todo lo relacionado con la base de datos utilizada y la segunda el entorno de desarrollo escogido junto con sus ventajas principales.

8.1.1 Base de datos

El lenguaje principal utilizado para la gestión y creación de la base de datos es SQL [7], bajo la gestión de la herramienta MySQL Workbench, que presenta un entorno visual para todo el proceso de diseño y desarrollo.

En la figura siguiente se presenta el diagrama final del modelo entidad-relación.

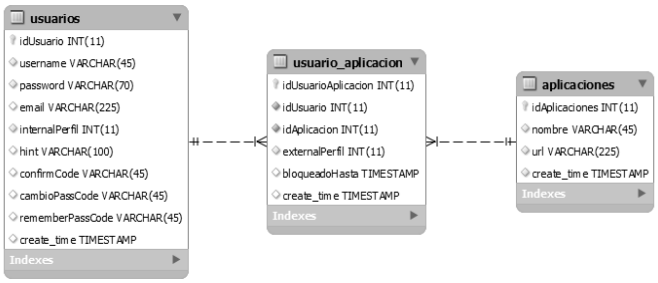


Figura 4. Modelo entidad-relación de la base de datos.

Dicho diagrama representa tres principales tablas, que se explicarán brevemente a continuación:

- **Usuarios**: Esta tabla almacenará principalmente la información relacionada con el usuario (identificador, nombre de usuario, contraseña, email, perfil interno, *hint*, código de confirmación enviado por email en el momento de registro, código de cambio de contraseña enviado por email y finalmente el código que permite gestionar la funcionalidad de recordar las credenciales en el navegador).
- **Aplicaciones**: En esta tabla se almacena la información principal de todas las aplicaciones asociadas al servicio de Global Login (identificador, nombre de la aplicación y la página web de redirección).
- **Usuarios-Aplicaciones**: Tabla que relaciona las dos anteriores mencionadas y añade dos columnas adicionales para la gestión de las funcionalidades de perfiles externos y bloqueo temporal.

Como último apunte a este apartado cabe destacar que la función hash criptográfica utilizada para el almacenaje de las contraseñas en la base de datos es SHA-256 [8] por las razones siguientes:

1. La probabilidad de colisión es sumamente baja ($4.3*10^{-60}$), que incluso puede ser considerada como 0. Este hecho asegura que sea casi imposible que dos cadenas de caracteres puedan dar el mismo resultado.
2. Dada la característica comentada anteriormente; el hecho de atacar con fuerza bruta la función hash para lograr vulnerar al sistema se vuelve una tarea sumamente costosa para los procesadores de hoy en día. Además cabe destacar que Global Login dispone de un sistema de bloqueo temporal que paraliza los intentos de acceso tras un número determinado de intentos.
3. Se trata de un algoritmo más costoso que el MD5, pero presenta una mejora muy considerable respecto la seguridad que aporta al sistema.

8.1.2 Entorno de desarrollo

Respecto al entorno de desarrollo escogido, se ha optado por realizar la implementación con NetBeans IDE 7.4 [9].

Es un entorno de desarrollo libre y gratuito sin restricciones de uso, estéticamente muy similar a eclipse, y diseñado especialmente para trabajar con Java y JSP (J2SE, web, EJB y aplicaciones móviles).

Las principales ventajas que ayudaron a escoger este IDE para la implementación son:

- Como todo software libre, este IDE dispone de una gran comunidad de soporte que facilita la tarea de resolución de errores.
- Entre sus características se encuentra un sistema de control de versiones y *refactoring*; ventaja fundamental vista la metodología utilizada.
- La configuración de la propia herramienta es automática en el momento de la instalación; de esta forma no se pierde tiempo en tareas de actualización y configuración.
- Incluye una gran variedad de *templates* que facilitan la generación de código.

Antes de pasar a la siguiente sección, cabe destacar que toda la implementación llevada a cabo y las librerías utilizadas se guardan en un repositorio remoto Bitbucket [11], de libre acceso y público.

8.2 Lenguajes

Los diferentes lenguajes que se utilizaron para la implementación son:

1. Java: Lenguaje básico para la implementación de la aplicación, dando lugar a las diferentes clases y servlets encargadas de la parte lógica de la aplicación y la conexión con la base de datos.
2. HTML5: Lenguaje utilizado para la elaboración de las distintas páginas web de la aplicación [12].
3. CSS3: Lenguaje utilizado básicamente para las tareas de maquetación y diseño web.
4. Javascript: Lenguaje de programación orientado a objetos, que se utiliza principalmente en la parte del cliente. Es lenguaje se ha utilizado mínimamente para realizar pequeñas comprobaciones de formularios.
5. SQL: Como se ha explicado anteriormente, este lenguaje se ha utilizado para el manejo y creación de la base de datos.

Aparte de estos lenguajes, se ha utilizado JavaServer Pages (JSP) para la creación de la totalidad de las páginas web. Es una tecnología que ayuda a implementar páginas web dinámicas de forma semejante a PHP, pero con la diferencia de que ésta utiliza Java como lenguaje principal. Para desplegar y ejecutar las distintas páginas web diseñadas con esta tecnología, se necesita un servidor web compatible con contenedores servlet; se ha utilizado como servidor GlassFish Server 4.0.

Como conclusión a este apartado, cabe señalar que se han utilizado diversas librerías de apoyo, tanto para la parte visual como para algunas funcionalidades básicas.

8.3 Diseño y arquitectura

En esta sección se explicará tanto la arquitectura principal del proyecto, como el diseño del sistema de perfiles implantado.

8.3.1 Arquitectura del proyecto

La arquitectura principal que fue utilizada para el desarrollo del proyecto es la basada en el patrón Modelo-Vista-Controlador. Es una arquitectura que intenta separar los datos, la lógica de la aplicación y la interfaz de usuario en distintos módulos; para ello, dicho modelo trata de construir tres diferentes componentes:

- **Modelo**: es el componente encargado de la gestión de la información que trata el sistema, tanto consultas a base de datos como actualizaciones y creación de información nueva.
- **Controlador**: normalmente tiene el rol de enlazar la vista con el modelo; por lo tanto, es el encargado de responder las peticiones del componente vista y solicitar la información al componente de modelo, tratar dichos datos y enviar el resultado de nuevo a la vista.
- **Vista**: representa la interfaz con la que el usuario interactuará con la aplicación. Técnicamente toda la información representada por la vista ha de ser extraída del componente modelo.

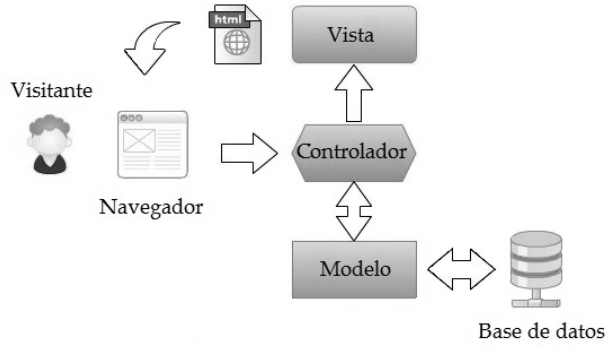


Figura 5. Arquitectura MVC.

Dicha arquitectura se ha adaptado a las necesidades del proyecto, partiendo de romper su rigidez y desarrollando un sistema menos estricto. Por ejemplo, todas las páginas web estáticas, donde el contenido no se verá modificado en ningún momento, su contenido será almacenado en la misma componente de vista por tal de facilitar y agilizar la implementación.

En la Figura 6 se pueden apreciar las diferentes particiones del código implementado en el proyecto.

Donde los componentes de la vista están almacenados en la carpeta Web Pages, que contiene todos los ficheros JSP, JSPF (fragmentos de páginas web), las imágenes utilizadas, ficheros CSS y los ficheros Javascript.

Por otro lado, todas las clases almacenadas dentro del subdirectorio Source Packages pertenecen a los componentes de modelo y controlador.

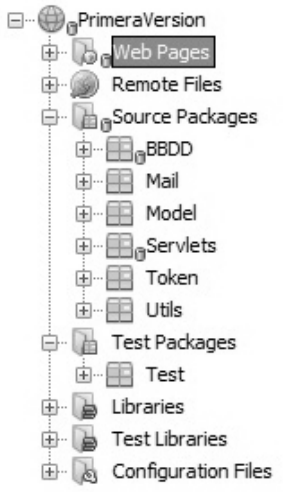


Figura 6. Representación de la división de código en el proyecto.

El componente de modelo está representado básicamente por el subdirectorio BBDD (contenedor de las funcionalidades de acceso y modificación de la base de datos) y Model (contenedor de las clases Aplicacion.java y User.java encargadas de mapear el contenido de la base de datos a objetos Java cuando se necesite).

Finalmente todas las demás carpetas pertenecen al componente controlador, conteniendo así las diferentes servlets, las funcionalidades de comunicación con tokens, el envío de cualquier email por parte de la aplicación y varías funcionalidades útiles utilizadas en distintos lugares.

8.3.2 Sistema de perfiles

El sistema actual es capaz de almacenar en la base de datos y gestionar dos tipos distintos de perfiles:

- **Perfil externo**: dicho perfil ha de ir siempre asociado a una relación usuario-aplicación, por lo tanto, si un usuario no está suscrito a ninguna aplicación, éste no tendrá asignado ningún perfil externo. Los valores de perfil se han definido de 0 a 15, siendo 0 el representante temporal de "sin perfil asignado". Dichos valores se entregan a las aplicaciones asociadas como mero soporte a su sistema de autorización.
- **Perfil interno**: por otro lado, el perfil interno se ha implementado siguiendo un sistema de gestión de perfiles por máscara, muy semejante al umask de los sistemas operativos Unix o la función de PHP [10]; por lo tanto, el sistema implementado es una representación simplificada.
  Dichos perfiles se pueden combinar entre ellos para formar nuevos perfiles hasta alcanzar los 16 como máximo.

```
1 1 1 1 → Admin        (15)
0 0 0 1 → Free         (01)
0 0 1 0 → Premium      (02)
0 1 0 0 → Application  (04)
1 0 0 0 → Reserved     (08)
```

Figura 6. Perfiles internos básicos utilizados.

9 RESULTADOS

En esta sección se expondrán todos los cambios llevados a cabo durante el transcurso del proyecto en la planificación inicial y sus correspondientes razones.

- Cambios a los objetivos funcionales A1 y A3: por un lado A1 pretendía la posibilidad de recordar la contraseña en el navegador y en segundo lugar A3 tenía el objetivo de enviar al usuario su contraseña vía email en caso de que éste la necesitara.
  Dichas funcionalidades fueron rechazadas después del pertinente análisis de requisitos ya que implican almacenar en la base de datos dichas contraseñas sin ningún tipo de sistema criptográfico; hecho que pone en grave peligro el sistema de autenticación ante una vulnerabilidad inesperada por parte de la base de datos.
  Como solución se implantó mediante cookies un sistema para recordar la sesión en sí y en segundo lugar, en vez de enviar la contraseña al usuario cuando la solicite, se ha implementado una pantalla de cambio de contraseña con un código enviado por email.
- Cambio en el sistema hash utilizado: al principio del proyecto se optó por utilizar MD5 como sistema de hash, más tarde se cambió por SHA-256 por las razones citadas anteriormente.
- Cambio en la planificación inicial: las tareas de maquetación tomaron más tiempo del inicialmente previsto, este hecho produjo un retraso a la hora de comenzar la implementación de la versión 0.4. La solución principal a este problema fue atrasar todas las tareas previstas en un plazo de una semana. Consumiendo de esta forma parte del tiempo del plan de contingencia.

10 CONCLUSIONES I LÍNEAS FUTURAS

10.1 Conclusiones

Como conclusión final, y tras la finalización del proyecto, se puede asegurar que las soluciones que presentan las diferentes arquitecturas de SSO destacan por su flexibilidad, su poco consumo de recursos (tanto económicos como computacionales) y su baja complejidad de infraestructura necesaria.

Un correcto diseño del sistema SSO puede aportar grandes beneficios para sus organizaciones, partiendo desde una reducción de costes en la administración de la seguridad, pasando por un incremento significativo de los niveles de seguridad y finalmente logrando una mayor satisfacción personal por parte del los usuarios finales.

10.2 Líneas futuras

Aparte de toda la implementación realizada y las funcionalidades cumplidas; han surgido nuevas ideas que pueden mejorar el resultado final de la aplicación:

- Utilizando el sistema de perfiles ya existente, seguir un negocio *freemium* basado en ofrecer servicios básicos a usuarios gratuitos y servicios más avanzados a usuarios de pago.
- Mejorar la seguridad del sistema de comunicación, tanto el realizado entre el cliente y servidor, como la comunicación entre el servidor y las distintas aplicaciones asociadas.

AGRADECIMIENTOS

A toda mi familia por su constante apoyo; a mi tutor del proyecto, Joaquim Borges Ayats, por guiarme en el proceso de desarrollo y documentación; y especialmente a mi pareja Miriam Alierta del Valle por darme ánimos y apoyarme en todo momento.

BIBLIOGRAFÍA

La última consulta, a modo de comprobación, de todas las referencias ha sido el 21/06/2014

[1] Ted Samson, "Study finds high rate of password reuse among users", (10 de Febrero de2011) [Online]: http://www.infoworld.com/t/data-security/study-finds-high-rate-password-reuse-among-users-188

[2] Álvaro Gómez Vieites, "Enciclopedia de la Seguridad Informática", Segunda edición actualizada, 2011, Capítulo 10. Autenticación, Autorización y Registro, pág. 327 - 341.

[3] B. Kaliski, J. Staddon, "RSA Cryptography Specifications", Verion 2.0 (Octubre de 1998) [Online]: http://www.ietf.org/rfc/rfc2437.txt

[4] T. Bray, "The JavaScript Object Notation (JSON) Data Interchange Format", (Marzo de 2014) [Online]: http://tools.ietf.org/html/rfc7159

[5] M. Jones, "JSON Web Token (JWT)", (8 de Mayo de 2012) [Online]: http://tools.ietf.org/html/draft-jones-json-web-token-09

[6] Ley Orgánica de Protección de Datos de Carácter Personal. (14 de Diciembre de 2014) [Online]: http://www.boe.es/buscar/doc.php?id=BOE-A-1999-23750

[7] E. Rivero, L. Martínez, L. Reina, J. Benavides, J. Olaizola, "Introducción al SQL para Usuarios y Programadores", Segunda edición, 2012, Capítulos 2-5, pág. 9-33.

[8] D. Eastlake, "US Secure Hash Algorithms", (Julio de 2006) [Online]: http://tools.ietf.org/html/rfc4634

[9] NetBeans IDE [Online]: https://netbeans.org/

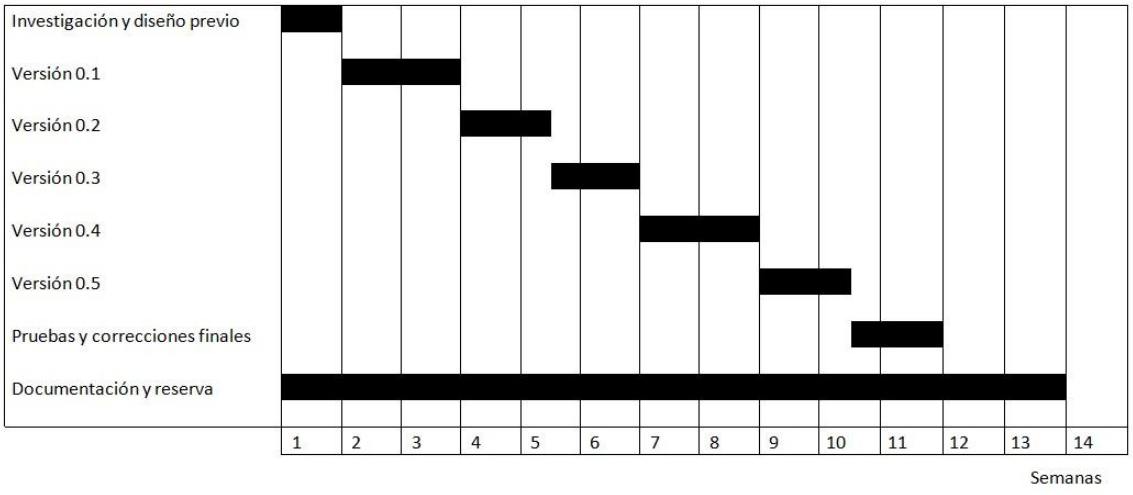
[10] The PHP Group, "Umask Definition" [Online]: http://php.net/manual/es/function.umask.php

[11] W. Alsalti, "Repositorio remoto de la aplicación" [Online]: https://bitbucket.org/wisam_alsalti/globalloginapp/commits/all

[12] Luc Van Lancker, "HTML5 y CSS3, Domine los estándares de las aplicaciones Web", Segunda edición, 2013, Capítulo 1. Presentación de HTML5, pág. 18-40.

APÉNDICE

A1. DIAGRAMA DE GANTT INICIAL



A2. DIAGRAMA DE GANTT FINAL