Escuela Politécnica Superior de Linares

Grado en Ingeniería Telemática

UNIVERSIDAD DE JAÉN

Escuela Politécnica Superior de Linares

Trabajo Fin de Grado

Desarrollo de software de red MANET (Mobile Ad-hoc Network)

Alumno: Raúl Salido Sánchez

Tutor: Prof. D. Ildefonso Ruano Ruano

Depto.: Ingeniería de Telecomunicación

Febrero, 2018

Universidad de Jaén

ESCUELA POLITÉCNICA SUPERIOR DE LINARES

Departamento de Ingeniería de Telecomunicación

CURSO ACADÉMICO

2017-2018

TRABAJO FIN DE GRADO

TÍTULO DEL TRABAJO

Desarrollo de software de red MANET (Mobile Ad-hoc Network)

Grado en Ingeniería Telemática

AUTOR: Raúl Salido Sánchez

TUTOR: D. Ildefonso Ruano Ruano

Linares, febrero de 2018

1 ÍNDICE

2 RESUMEN

Una red Ad-Hoc es un tipo de red que se crea de forma circunstancial para resolver un problema de comunicación específico y, normalmente, por un periodo de tiempo limitado. Estas redes pueden ser cableadas o inalámbricas y, en ellas, por definición no deberían existir infraestructuras de comunicaciones como puede ser un nodo central o padre que ayude al funcionamiento de las comunicaciones de red [1]. Cuando los nodos que forman la red Ad-Hoc tienen capacidad de movimiento se dice que es una red de tipo MANET (*Mobile Ad-Hoc Network*), una implicación del movimiento de los nodos es que en las redes MANET se asume que las comunicaciones se producen de forma inalámbrica. Un caso particular de red MANET lo constituyen las redes establecidas entre vehículos, más conocidas como redes VANET (*Vehicular Ad-Hoc Network*).

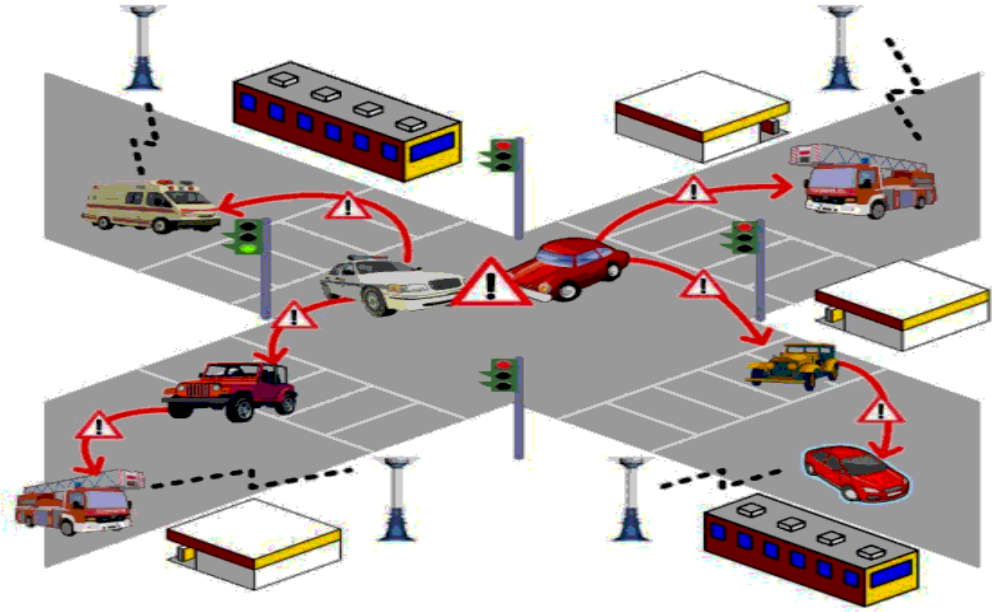


Figura 2.1 Ejemplo VANET. Fuente: https://ns2projects.org/ns2-simulation-code-for-vanet/

La temática principal de este Trabajo Fin de Grado está centrada en las redes MANET, se han desarrollado dos módulos de contenido de *e-learning* que incluyen, entre otros materiales docentes, simulaciones interactivas de redes MANET que ayudan a la comprensión de este tipo de redes.

Estos recursos se pueden utilizar en la mayoría de los Sistemas de Gestión de Aprendizaje (LMS o *Learning Management System* en inglés) [2] del mercado, como por ejemplo la plataforma institucional ILIAS [2] empleada por la Universidad de Jaén, en la que se han realizado las pruebas de funcionamiento necesarias para su depuración.

Para la obtención de estos módulos se han utilizado tecnologías y estándares abiertos, así como herramientas de uso libre. En el apartado siguiente se van a describir

brevemente a todas ellas, incluyendo también explicaciones de algunos de los conceptos incluidos en este resumen.

3 INTRODUCCIÓN

Como se ha comentado anteriormente se han creado varias aplicaciones de simulación de redes MANET. Los programas creados en el ámbito de este proyecto se han integrado como parte de contenido de *e-learning* (aprendizaje electrónico), para ello se han utilizado distintas tecnologías y lenguajes.

Para la elaboración de las simulaciones se ha utilizado el software *Easy Java Simulations* (EJS) [3] el cual ofrece las opciones de programar usando los lenguajes Java o JavaScript. En este caso se ha utilizado JavaScript por ser el lenguaje de programación Web más extendido, utilizado y compatible con la mayoría de navegadores a la hora de obtener páginas Web interactivas. El uso del lenguaje Java fue desestimado por los problemas que presentan los navegadores para la ejecución de applets Java.

Con el fin de conseguir una mayor efectividad docente se han desarrollado contenidos docentes asociados a estas simulaciones, y para extender la compatibilidad de todo el conjunto se ha utilizado el estándar SCORM (*Shared Content Object Reference Model*) [4] como contenedor de todos ellos. Los módulos, laboratorios virtuales, Laboratorios Web o WebLabs, desarrollados de esta forma puedan ser empleados en distintos entornos LMS y utilizados por los alumnos remotamente con el simple uso de un navegador Web. Los dos laboratorios virtuales desarrollados sirven para ayudar al aprendizaje sobre las redes MANET y sus protocolos de enrutamiento. Al emplear el formato SCORM, estos módulos se pueden utilizar con cualquier LMS compatible al que se quieran agregar (la mayoría del mercado, ya que SCORM es el estándar de contenido de *e-learning* más extendido [5]). Un LMS es un software que reside en un servidor y se encarga de controlar, administrar y distribuir las diferentes actividades y ejercicios de carácter formativo de una organización o institución. Las pruebas de funcionamiento de estos módulos se han realizado en el LMS ILIAS de la Universidad de Jaén.

En el desarrollo de los módulos SCORM se han usado múltiples tecnologías:

- HTML (*HyperText Markup Language*) [6], XHTML (eXtensible HyperText Markup Language) [7], DOM (*Document Object Model*) [8] y XML (*Extensible Markup Language*) [9] para la elaboración de las distintas páginas web que componen los módulos.
- JavaScript [10] y CSS (*Cascading Style Sheets*) [11] para agregar funcionalidad y estilo de las páginas, respectivamente.
- el estándar QTI (*Question & Test Interoperability*) [12], necesario para la elaboración e integración de las preguntas de tipo test en dichos módulos.

Todas estas tecnologías se explicarán más detenidamente en los siguientes sub-apartados.

También gracias al uso del estándar SCORM y a la programación realizada, los módulos de prácticas obtenidos son interactivos, personalizables y adaptables a las preferencias del alumno, de manera que al mismo tiempo monitorizan y almacenan el trabajo individualizado realizado por cada alumno para, posteriormente, poder ofrecer estos resultados a ellos mismos y a los tutores.

A continuación, se describen brevemente como están estructurados los dos WebLabs realizados, así como los contenidos más importantes que incluyen:

- **WebLab VANET:** está organizado en 5 páginas Web:
    - **Página 1. Introducción:** Proporciona información general sobre el WebLab y enmarca al WebLab dentro del programa de estudios del alumno.
    - **Página 2. Teoría**: Proporciona información teórica sobre lo que son las redes Ad-Hoc y profundiza sobre un tipo de estas llamadas redes VANET. Contiene un test inicial cuyo propósito es asegurar que se han asimilado los conceptos teóricos incluidos, con el objetivo de permitir acceso al resto del WebLab sólo cuando se haya demostrado su comprensión. De esta forma sólo podrán acceder al laboratorio (simulación), los alumnos preparados para ello.
    - **Página 3. Descripción de simulación:** En esta página se encuentra la descripción de la simulación realizada, con el objetivo de ayudar de forma gráfica a comprender el funcionamiento de la misma, y con ello, de este tipo de redes, así como su identificación.
    - **Página 4. Simulación y actividades:** Incluye la simulación JavaScript desarrollada y ofrece al alumno una serie de actividades que debe realizar en la misma para asegurar la correcta comprensión y uso. En la simulación encontrará un juego que junto con las actividades determinarán la nota de este SCO. Si se superan todas las actividades correctamente se podrá pasar a la siguiente página para realizar el test de evaluación final.
    - **Página 5. Test Final (última página):** Contiene un test final (solo aparecerá si el alumno realiza de manera exitosa las actividades previamente propuestas) relacionado con la simulación y toda la teoría explicada anteriormente que los alumnos deben superar.

- **WebLab OLSR:** La estructura que se seguirá será la siguiente:
    - **Página 1. Introducción:** Proporciona información general sobre el WebLab y se enmarca dentro de los estudios de grado.

- **Página 2. Teoría**: Proporciona información teórica sobre lo que son las redes Ad-Hoc y MANET y profundiza sobre el protocolo de enrutamiento *Optimized Link State Routing* (OLSR) [13]. Contiene un test inicial cuyo propósito es asegurar que se han asimilado los conceptos teóricos explicados antes de acceder al laboratorio, con el objetivo de afianzar los conocimientos antes de acceder al laboratorio
- **Página 3. Descripción de simulación:** En esta página se verá una descripción de una simulación que se ha realizado con el objetivo de ayudar de forma gráfica a comprender el funcionamiento de este tipo de redes así cómo identificarlas.
- **Página 4. Simulación y actividades:** Muestra la simulación desarrollada y obliga al alumno a realizar una serie de actividades para asegurar la correcta comprensión y uso de la simulación. Si se superan todas las actividades correctamente se podrá pasar a la siguiente página para realizar el test de evaluación final.
- **Página 5. Test Final (última página):** Contiene un test final (solo aparecerá si el alumno realiza de manera exitosa las actividades previamente propuestas) relacionado con toda la teoría sobre lo explicado anteriormente en el WebLab que los alumnos deben superar.

La falta de infraestructura permite que las redes MANET se puedan desplegar rápidamente en cualquier momento y lugar. Los nodos de esta red tienen capacidad de auto-organización la cual usan para transportar paquetes de datos entre un origen y un destino.

A continuación, se van a describir brevemente los distintos conceptos, tecnologías y estándares que se ha empleado para la realización de este TFG.

3.1 E-Learning

Con el gran crecimiento de las tecnologías en los últimos años, la sociedad poco a poco va adentrándose y dependiendo cada vez más de ella. Prácticamente hoy en día se aplica la tecnología a cosas cotidianas de la vida como puede ser mantener comunicación con otras personas, buscar una ubicación de un lugar al que ir, etc.

La tecnología se está expandiendo a todos los ámbitos que rigen una sociedad como es la educación, sanidad, economía, política, etc. Gracias a las posibilidades que da internet en el ámbito de la educación se está mirando a un futuro en el cual los alums no tengan que desplazarse a su colegio, o que los mismos profesores no tengan que hacerlo tampoco, esto se denomina, educación online. Este tipo de educación ya está implantada

en diversas universidades las cuales ofrecen estudios de grado o de postgrado de manera online.

Según las estadísticas del ministerio de educación, cultura y deporte [14] , para el curso pasado 2016/2017, los alumnos matriculados en enseñanzas de régimen general no universitarias son:

- Bachillerato a distancia: 44653 alumnos.
- Ciclos formativos de FP Grado Medio a Distancia: 27399 alumnos.
- Ciclos formativos de FP Grado Superior a Distancia: 50516 alumnos.

Se puede definir el e-learning como un conjunto de procesos que usan las Tecnologías de Información y Comunicación (TIC), con una finalidad de aprendizaje.

El concepto de e-learning es comparable con docencia/aprendizaje mediante el uso de la electrónica. Para comprenderlo mejor, se desglosará el término:

- **e-:** Su significado es electronic (electrónico) y refiere al medio que se utiliza para la transmisión de información (ordenadores, redes, etc.).
- **Learning:** Su significado es aprendizaje y se refiere a la obtención de habilidades y conocimientos a través del estudio y la experiencia.

Existen distintos tipos de e-learning en función al tipo de docencia que se quiera impartir: virtual, mixto y de apoyo [15].

- **Virtual (e-learning puro):** La docencia se hace totalmente de manera virtual usando para ello internet.
- **Mixto (blended-learning):** La docencia se hace 50% virtual y 50% presencial dividiendo el curso y las actividades en presenciales y virtuales.
- **De apoyo:** Internet es usado como apoyo a la docencia presencial para la información de software de utilidad para complementar lo aprendido de manera presencial.

Para la docencia universitaria, se utiliza el elemento TIC Learning Management System (LMS) [15].

Para sacarle el máximo rendimiento a la enseñanza virtual, es conveniente utilizar herramientas que permitan crear simulaciones acerca del contenido que se quiere transmitir, para ello se ha utilizado Easy Java Simulations (EJS) [3] que permite crear simulaciones gráficas de manera sencilla. EJS está diseñado para ayudar a los estudiantes de ciencias a comprender sus materias y poder hacerlo mediante la visualización gráfica de simulaciones, las cuales harán que el alumno comprenda mejor el tema que esté

tratando. EJS permite el desarrollo de aplicaciones Java independientes y multiplataforma, o *applets* que son compatibles con cualquier navegador Web, aunque la evolución de estos no ha sido la esperada y se van quedando cada vez más obsoletos cuando se habla de desarrollo web. También permite desarrollar aplicaciones con el lenguaje JavaScript, el cual ha sido escogido para la realización de este TFG debido a que se integra mejor con los navegadores y se ejecuta en el cliente por lo que no se tiene que esperar una respuesta del servidor.

Otra ventaja que ofrece EJS es que permite exportar los proyectos con formato Sharable Content Object Reference Model (SCORM) [4] . SCORM es el estándar que define la manera de crear contenido digital dedicado a la docencia para que este pueda integrarse y comunicarse perfectamente con un LMS.

Como todo, este tipo de educación tiene sus ventajas y sus desventajas. Algunas de las ventajas que puede ofrecer son las siguientes:

- **Flexibilidad:** Al no depender de un horario fijo o un sitio determinado, se puede compaginar la formación de una persona con su vida laboral.
- **Colaboraciones:** Mediante foros, chats… a través de los cuales, los mismos alumnos pueden plantear sus dudas y estas pueden ser solucionadas por otros alumnos o por el profesor.
- **Ahorro de dinero:** Tanto a la hora de adquirir material, si se necesita algún tipo de software para el desarrollo de una actividad, este puede ser proporcionado a través de la página web de la universidad o centro en el que se estén realizando los estudios, además, el ahorro en transporte y vivienda para aquellos alumnos cuyos estudios se encuentran en un centro que no está ubicado en su localidad.
- **Responsabilidad:** La formación online exige que el alumno se planifique el mismo adaptándose a sus necesidades pero que lo haga de una forma correcta y disciplinada para llevar sus estudios al día.

Algunas desventajas que se pueden encontrar son:

- **Familiarización con las tecnologías:** Para poder realizar unos estudios de este tipo es necesario tener unas nociones básicas sobre el manejo a nivel de usuario de las tecnologías.
- **Medios de difusión de la información:** Algunos centros educativos no disponen de las herramientas necesarias para hacer un uso efectivo de la docencia online, por lo que sus formaciones no tendrán el mismo nivel de calidad de aprendizaje como el de otros centros.

3.2 Redes MANET

Para hablar de redes inalámbricas hay que remontarnos a los años 70 donde Agencia de Investigación de Proyectos Avanzados de Defensa, EE.UU. (DARPA) ya comenzó a crear este tipo de redes para operar con ellas.

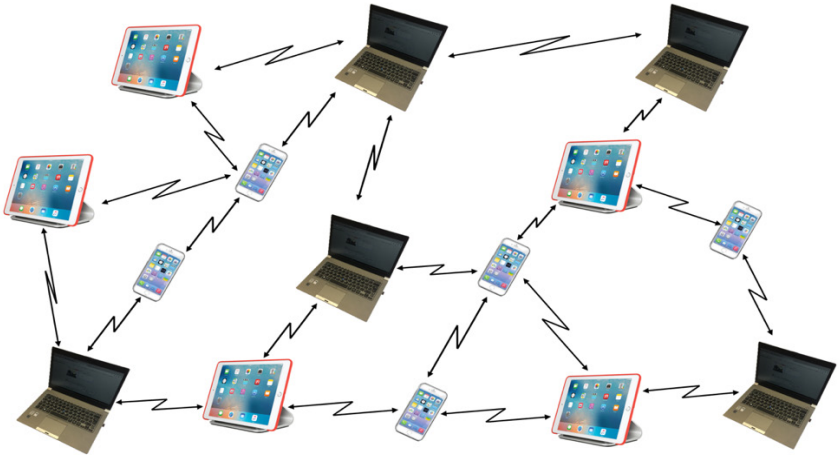


Figura 3.1 Red MANET. Fuente: Propia.

El auge de internet las y la constante evolución de las redes inalámbricas han dado relevancia a las redes ad-hoc (redes móviles). El termino ad-hoc tiene un significado literalmente *"para esto"* por lo que este tipo de redes se crean por y para una necesidad concreta durante un tiempo limitado. La principal ventaja de este tipo de redes es que no necesitan una infraestructura para poder montarse y funcionar, esto brinda un gran abanico de posibilidades en cuanto a aplicabilidad como la comunicación entre vehículos, sensores, etc.

En los años 70 el ministerio de defensa americano mostró su interés por el proyecto que se estaba llevando acabo entonces, denominado Packet Radio Networks (PRNET) el principal objetivo de este proyecto era el de establecer comunicaciones en el campo de batalla usando dispositivos de radio, en el que hubiera diferentes tipos de movimientos, de tal forma que, en cualquier momento, cualquier radio (nodo) pudiera hacer las funciones de un conmutador de paquetes.

Por tanto, se puede definir una red ad-hoc como una red formada por dispositivos de comunicación inalámbricos con un propósito específico que se conectan por periodos de duración corta sin hacer uso de ningún tipo de infraestructura.

Algunas de las aplicaciones más usuales de este tipo de redes son:

- Militares.
- Recuperaciones de desastres.

- Entornos donde no es posible montar infraestructuras.
- Entornos donde es excesivo el coste de montar una infraestructura.
- Comunicaciones entre vehículos.
- Ocio.

Las características más comunes de este tipo de redes son:

- Comunicaciones inalámbricas.
- Uso limitado a un periodo de tiempo determinado.
- Sencillas de montar.
- Flexibilidad.
- Terminales autónomos.
- Homogeneidad sin nodo central. Funcionamiento distribuido.
- Ofrecer un servicio personalizado o improvisado.
- Cuenta con protocolos personalizados que les permiten adaptarse a necesidades específicas.
- Movilidad.
- Multidifusión.

Los tipos de redes que se pueden encontrar son las siguientes:

- **Redes en Malla (Mesh Network):** Los nodos transmiten y reciben datos y además son capaces de reenviar datos procedentes del resto de nodos que conforman la red.
- **Redes Ad-Hoc Móviles (MANET):** Se trata de una red Ad-hoc cuyos nodos tiene capacidad de movimiento. Cada uno de los dispositivos que forman una red MANET tiene la capacidad de moverse libremente en cualquier sentido y dirección, esto hace que las conexiones entre los distintos dispositivos cambien de manera muy rápida.
- **Redes de sensores inalámbricas:** Wireless Sensors Networks (WSN) es una red formada por ordenadores pequeños equipados con sensores con el objetivo de realizar una tarea común
- **Redes MANET Vehiculares (VANET):** Vehicular Ad-Hoc Network (VANET), red establecida entre vehículos [1]. Los vehículos son considerados los nodos que conforman la red por lo que una red VANET es considerada como un tipo de red MANET.

Este TFG se centrará en las redes VANET y MANET.

3.2.1 *Protocolo de enrutamiento OLSR*

El Protocolo *Optimized Link State Routing Protocol* (OLSR), con RFC 3616 [16] está desarrollado para redes ad hoc móviles. Funciona como un protocolo proactivo controlado por tabla, es decir, intercambia información de topología con otros nodos de la red regularmente. Cada nodo selecciona un conjunto de sus nodos vecinos como "retransmisores multipunto" (MPR). En OLSR, solo los nodos, seleccionados como MPR, son responsables de reenviar el tráfico de control, destinado a la difusión en toda la red. Los MPR proporcionan un mecanismo eficiente para controlar el tráfico de inundaciones al reducir el número de transmisiones requeridas.

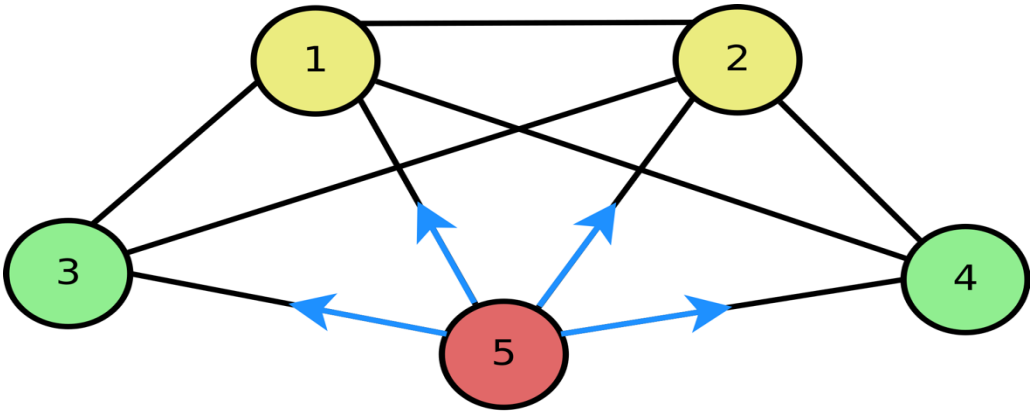


Figura 3.2 Red Ad-Hoc con protocolo OLSR. Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/f/f0/Envoie_Hello.svg/2000px-Envoie_Hello.svg.png

Los nodos, seleccionados como MPR, también tienen una responsabilidad especial al declarar información de estado del enlace en la red. De hecho, el único requisito de OLSR para proporcionar las rutas más cortas a todos los destinos es que los nodos MPR declaran información de estado de enlace para sus selectores de MPR. Se puede utilizar información de estado de enlace disponible adicional, por ejemplo, para redundancia.

Los nodos que han sido seleccionados como MPR por algunos nodos vecinos anuncian esta información periódicamente en sus mensajes de control. De este modo, un nodo anuncia a la red que tiene accesibilidad a los nodos que lo han seleccionado como MPR. En el cálculo de la ruta, los MPR se utilizan para formar la ruta desde un nodo determinado a cualquier destino en la red. Además, el protocolo usa los MPR para facilitar la inundación eficiente de los mensajes de control en la red.

Un nodo selecciona MPR de entre sus vecinos de un salto con enlaces "simétricos", es decir, bidireccionales. Por lo tanto, la selección de la ruta a través de MPR evita automáticamente los problemas asociados con la transferencia de paquetes de datos a través de enlaces unidireccionales (como el problema de no obtener reconocimientos de

capa de enlace para paquetes de datos en cada salto, para capas de enlace que emplean esta técnica para unidifusión tráfico).

OLSR está desarrollado para funcionar independientemente de otros protocolos. Del mismo modo, OLSR no hace suposiciones sobre la capa de enlace subyacente.

Un estudio ha realizado una comparativa entre los protocolos de enrutamiento ha decretado que el protocolo que mejores prestaciones ofrece a la hora de conformar una red MANET es OLSR [17].

3.3 JavaScript

JavaScript es uno de los tres lenguajes que hoy en día se usan para la programación web, junto con HyperText Markup Language (HTML) y Cascading Style Sheets (CSS). Cada uno de estos lenguajes cumplen su función dentro de una página web:

- **JavaScript:** Especifica la manera de comportarse de una página web.
- **HTML:** Usa etiquetas para organizar el contenido de una web.
- **CSS:** implementa el aspecto gráfico y la presentación que tendrá una web.

Figura 3.3 HTML, CSS y JavaScript. Fuente: https://commons.wikimedia.org/wiki/File:HTML5_logo_and_wordmark.svg

Durante los años 90 se desarrolló la programación de las páginas web, a las cuales empezaron a incluirle aplicaciones web que constaban de una complejidad mayor en cuanto a formularios integrados en las mismas. El problema surge en que el desarrollo de web complejas aumentaba, pero la velocidad de conexión no lo hacía al mismo ritmo, así fue como se pensó en lo beneficios de un lenguaje de programación que redujera el tiempo de espera cuando, a la hora de rellenar un formulario y hubiera algún error, lo notificara rápidamente, es decir un lenguaje de programación que se ejecutase en el lado de usuario en vez de en el servidor.

La estandarización de este lenguaje se produce en 1997 gracias a la empresa Netscape, la cual envió al organismo European Computer Manufacturers Association (ECMA) la especificación JavaScript 1.1. [10]

ECMA se encargó de la creación de un comité para "*estandarizar de un lenguaje de script multiplataforma e independiente de cualquier empresa*" de modo que este comité creo la primera estandarización del lenguaje ECMAScript que se denominó ECMA-262.

Para personas que ya tengan conocimientos en lenguajes como java o C les puede resultar sencillo aprender JavaScript debido a la similitud que hay entre estos lenguajes. De la misma forma, para alguien que no tenga conocimientos sobre estos lenguajes no le resultara muy difícil aprenderlo debido a que se trata de un lenguaje muy sencillo y manejable sintácticamente hablando.

El abanico de posibilidades a la hora de crear contenido visual en una web es muy grande usando JavaScript, desde carruseles y galerías de imágenes hasta juegos y gráficos 3D.

Por otra parte, por encima del núcleo de este lenguaje se han desarrollado una gran cantidad de herramientas que permiten tener acceso a una cantidad mayor de funcionalidades si a penas esforzarse, por ejemplo:

- Interfaces de Programación de Aplicaciones del Navegador (API) desarrolladas en los navegadores para ofrecer funcionalidades como establecer y crear contenido HTML y estilos CSS.
- Librerías y Marcos de trabajo para implementar y subir de manera muy rápida aplicaciones y páginas web. Unas de las librerías más usadas por los programadores web es la librería JQuery.

3.4 HTML

HyperText Markup Language (HTML) es un lenguaje de programación desarrollado por el Conseil Européen pour la Recherche Nucléaire (CERN) basado en un conjunto de etiquetas que se encargan de estructurar y definir el contenido de una página web. Surge de las etiquetas Standard Generalized Markup Language (SGML) y su objetivo principal era la de difundir información con formato de texto e incluyendo algunas imágenes.

En 1991 se publicó el primer documento de manera formal el cual tenía la descripción de HTML, dicho documento recibió el nombre de HTML Tags . Este documento, a día de hoy puede ser consultado usando cualquier navegador a través de internet [18].

En 1995 Internet Engineering Task Force (IETF) publica el estándar HTML 2.0 que pese a su nombre es el primer estándar publicado de este lenguaje.

A partir de 1996 el encargado de la publicación de los estándares de este lenguaje es un organismo denominado World Wide Web Consortium (W3C), la primera versión que publicó este organismo fue HTML 3.2 en 1997.

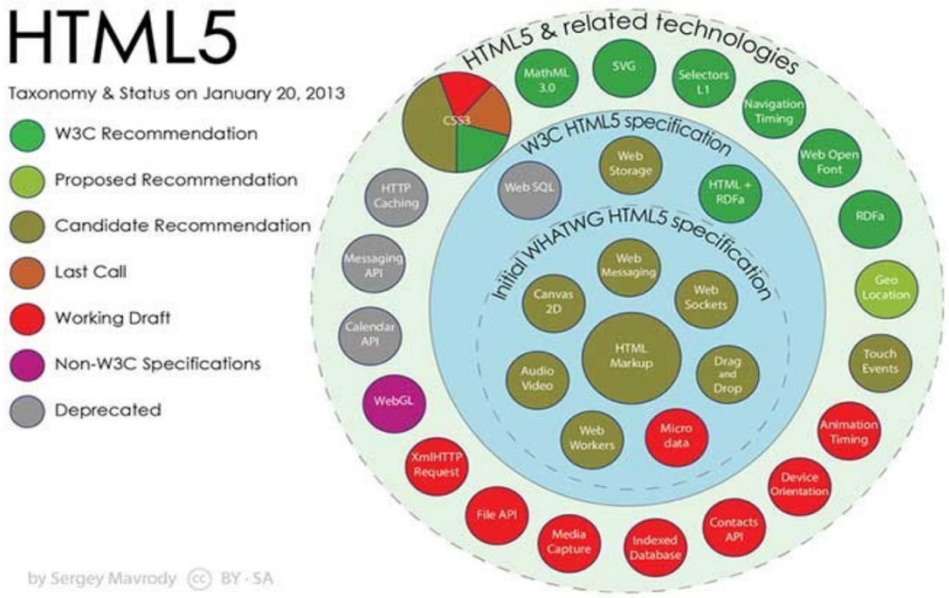


Figura 3.4 HTML5. Fuente: https://www.genbetadev.com/desarrollo-web/historia-de-html-un-lenguaje-de-marca-que-ha-marcado-historia

Fue a partir de la versión 4.0 publicada en 1998 la que supuso un gran salto al ser la que ofrecía la posibilidad de insertar a los archivos de código HTML puro, hojas de estilo CSS y scripts programados con otros lenguajes como JavaScript o PHP.

La versión 4.01 se publicó en 1999 pero a diferencia de las demás, esta no tenía funcionalidades nuevas, sino que corregía errores de la versión anterior.

En 2014 se publicó de forma oficial y tras muchos borradores, la última versión que hoy se conoce de este lenguaje HTML 5 la cual trae consigo mejoras a nivel multimedia, permite video, audio, gráficos vectoriales y creación de contenido 3D y 2D mediante canvas.

Para la correcta escritura de las etiquetas de las que consta el documento HTML se usan los caracteres especiales paréntesis angulares “<” y “>”, con esto lo que se logra es realizar un marcado estructural lo que hace que se concrete la finalidad del texto, pero sin llegar establecer como se visualice el elemento. Esto se verá complementado por el marcado presentacional, el cual se encarga de definir como se visualizará el texto.

HTML permite insertar scripts de código escritos en otros lenguajes que dotan al navegador de funcionalidades específicas al procesarlos. Los lenguajes más usados para elaborar estos scripts son JavaScript y PHP

3.4.1 Estructura básica de un documento HTML

Todos los documentos HTML tienes una estructura de inicio, en la cual algunos componentes (etiquetas) son de carácter obligatorio y otras de carácter opcional. La figura 3.5 muestra la estructura básica de un documento HTML:

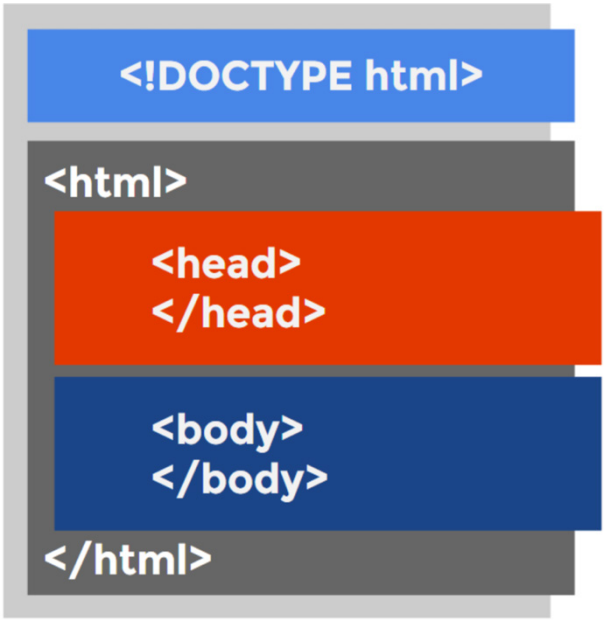


Figura 3.5 Estructura básica de un documento HTML. Fuente: https://lenguajehtml.com/p/html/introduccion/estructura-documento-html

Se diferencian 4 etiquetas:

- **<!DOCTYPE html>:** Es la primera línea del documento e indica el tipo de documento, en este caso, html. Esta etiqueta no es de carácter obligatorio, pero es muy recomendable ponerla ya que evita problemas.
- **<html></html>:** Todo documento html deberá de tener estas etiquetas, son de carácter obligatorio. <html> indica donde comienza el documento y </html> donde acaba.
- **<head></head>:** En esta sección se incluirá la información que no será visible para el usuario (metadatos) excepto el título de la página. <head> indica donde empieza la cabecera del documento y </head> donde termina.
- **<body></body>:** Esta suele ser la sección más gruesa de un documento HTML ya que contiene toda la información que se mostrará al usuario. <body> indica donde empieza el cuerpo del documento y </body> donde acaba.

3.5 XHTML

< XHTML >

XHTML (*EXtensible HyperText Markup Language*) surge en el año 2000 debido a la necesidad de mejorar y limitar las restricciones que tenía HTML a la hora de integrarse con distintas herramientas basadas en Extensible Markup Language (XML). [18]

La primera versión fue XHTML 1.0 y fue una aclimatación de HTML 4.01 al lenguaje XML. Esta versión se diferenciaba poco de HTML 4.01, las únicas novedades que incorporó fue la inserción de algunas restricciones propias del lenguaje XML.

En 2001 el W3C publicó la recomendación de la versión XHTML 1.1 con la novedad de la modularización para este lenguaje. Actualmente se está trabajando en la versión XHTML 2.0 en la que el cambio con respecto a las versiones anteriores será más significativo, con cambios en el vocabulario de elementos.

Algunas de las diferencias más significativas con HTML son:

- La etiqueta DOCTYPE es obligatoria.
- El atributo xmlns en <html> es obligatorio.
- <html>, <head>, <title> y <body> son obligatorios.
- Los elementos XHTML deben estar anidados correctamente.
- Los elementos XHTML deben de acabar siempre con su etiqueta de cierre.
- Los elementos XHTML se escriben siempre con letra minúscula.
- Los documentos XHTML deben de tener un elemento raíz [7].

3.6 CSS

Cascading Style Sheets (CSS) es un lenguaje basado en hojas de estilos que es el encargado de proporcionar estilo a los distintos elementos HTML que constituyen un documento HTML, es decir, CSS controla el aspecto y presentación que tendrá finalmente una página web [11].

Separar el contenido de un texto y su estilo de presentación es muy ventajoso ya que con esto lo que se logra es que, solo modificando la hoja de estilos, la página web se visualice de una forma totalmente distinta.

En 1996 la W3C, organismo que se encarga de definir y publicar los estándares relacionados con la web y concretamente en este caso de CSS, publicó la primera versión oficial de este lenguaje denominada CSS1. Esta recomendación incorporaba funcionalidades como:

- Atributos propios de caja como los bordes, relleno, margen y espaciado.
- Presentación de listas y propiedades de identificación.
- Colores de elementos como bordes, fondos o textos entre otros [19].

En 1998 la W3C publica la segunda estandarización de manera oficial de este lenguaje, CSS2. Esta recomendación trato de ampliar las funcionalidades existentes en CSS1 e incorporó otras nuevas como:

- Sombras y texto bidireccional.
- Soporte para las hojas de estilo auditivas.
- Funcionalidades características de los div como niveles, posicionamiento (fijo, absoluto, relativo), etc. [19]

En 2007 apareció la tercera versión de este lenguaje, CSS3 que trajo consigo el concepto de modularización, es decir, CSS3 estaba compuesto por varios documentos llamados módulos independientes. Para mantener la compatibilidad entre versiones, cada módulo añadía nuevas funcionalidades a las ya existentes de CSS2. En noviembre de 2011 se contabilizaron cincuenta módulos distintos que componían CSS3, tres de los cuales pasaron a formar parte de las recomendaciones de la W3C como:

- Selectores.
- Espacios de nombres.
- Color [19].

Navegador	Motor	CSS 1	CSS 2.1	CSS 3
Internet Explorer	Trident	Completo desde la versión 6.0	Completo desde la versión 8.0	Prácticamente nulo
Firefox	Gecko	Completo	Casi completo	Selectores, pseudo-clases y algunas propiedades
Safari	WebKit	Completo	Casi completo	Todos los selectores, pseudo-clases y muchas propiedades
Opera	Presto	Completo	Casi completo	Todos los selectores, pseudo-clases y muchas propiedades
Google Chrome	WebKit	Completo	Casi completo	Todos los selectores, pseudo-clases y muchas propiedades

Figura 3.6 Compatibilidad de CSS1, CSS2 y CSS3 con navegadores. Fuente: http://dis.um.es/~lopezquesada/documentos/IES_1213/LMSGI/curso/UT5/librosweb css/www.librosweb.es/css/capitulo1/soporte_de_css_en_los_navegadores.html

Algunas de las ventajas que CSS proporciona son:

- Gracias a la separación del código HTML y CSS se obtendrá un código más claro y limpio.
- Ahorro de esfuerzo y tiempo ya que para modificar el aspecto visual de un sitio web solo se deberá modificar el archivo de estilo CSS.
- La experiencia de usuario más homogénea.

3.7 DOM

"Document Object Model (DOM) es una interfaz de plataforma y lenguaje neutro que permitirá a los programas y scripts acceder y actualizar el contenido, estructura y estilo de los documentos de forma dinámica." [8]

Se trata de un estándar definido por W3C que a su vez define un estándar para acceder a los documentos. Es la arquitectura base sobre la que se elaboran los documentos XML y HTML por lo que las aplicaciones web y los navegadores que quieran mostrar algún documento que este elaborado bajo este formato deberán de tener esta base.

DOM sigue una estructura de árbol, es decir, de un objeto padre pueden descender varios objetos hijos y estos hijos dependerán del padre.

W3C divide el estándar DOM en tres partes distintas:

- **Core DOM (Núcleo DOM):** Estándar modelo a seguir para los documentos de todo tipo.
- **XML DOM:** Estándar modelo para los documentos de tipo XML.
- **HTML DOM:** Estándar modelo para los documentos de tipo HTML.

El DOM HTML es un estándar que permite obtener, cambiar, añadir o eliminar elementos HTML. Al comenzar el proceso de renderizado de una página web, el navegador lo primero que hace es crear una estructura DOM de la página. Una forma visual de ver esta estructura se apreciará en la figura 3.7:

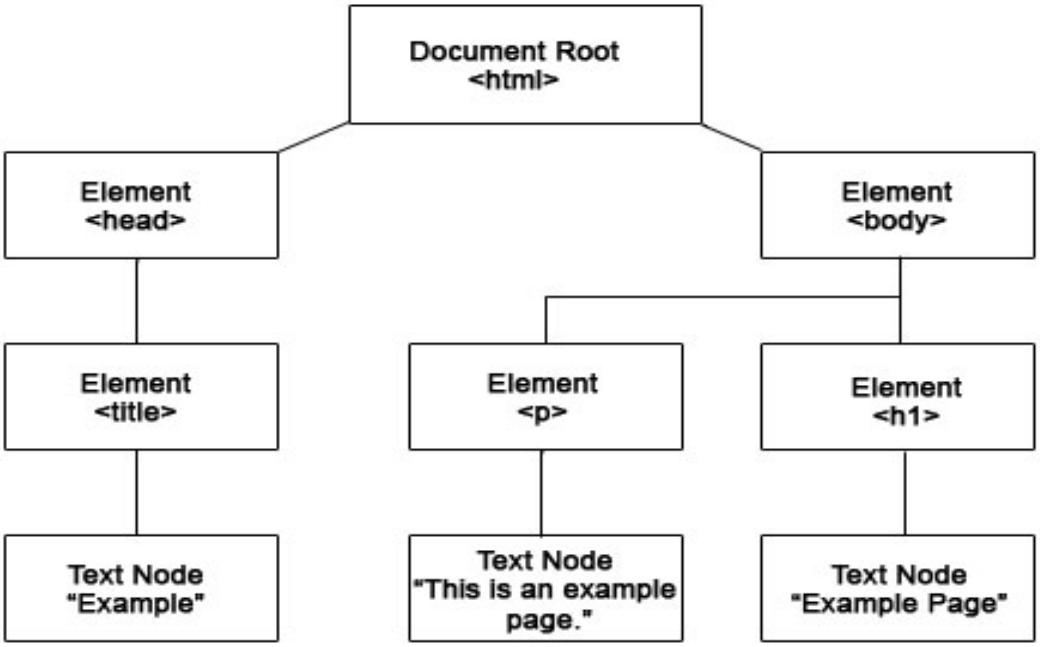


Figura 3.7 HTML DOM estructura de objetos con forma de árbol. Fuente: https://www.computerhope.com/jargon/d/dom.htm

JavaScript lo que hace es aprovecharse de este modelo para poder manipular de forma dinámica un documento o incluso para poder desarrollarlo desde cero.

Algunas de las capacidades que adquiere JavaScript gracias al DOM son:

- Puede insertar o modificar todos los elementos HTML de una página.
- Puede modificar los atributos HTML de una página.
- Puede modificar los estilos CSS de una página.
- Puede eliminar tanto elementos como atributos HTML ya creados [20].

3.8 XML

XML

XML (eXtensible Markup Language) fue creado por el W3C en el año 1998 como resultado de una versión mejorada del lenguaje SGML (Standarized generalized marckup language). Se trata de un metalenguaje por lo que su principal objetivo de poder aportar información sobre otro lenguaje, es decir, un lenguaje para hablar de otro lenguaje.

XML fue diseñado para ser humano y a la ver interpretado por máquina y puede transportar datos.

Los documentos elaborados con XML están formados por entidades las cuales almacenan datos que posteriormente serán analizados o no. [21]

Un ejemplo de un documento XML se puede ver en la figura 3.8:

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Figura 3.8 Documento XML. Fuente: https://www.w3schools.com/xml/default.asp

La mayoría de los sistemas informatizados contienen datos en formatos no compatibles por lo que el intercambio de datos entre sistemas de distintas características conlleva un consumo de tiempo e incluso grandes cantidades de datos pueden llegar a perderse [9] . XML lo que proporciona es una manera de simplificar todas estas tareas:

- Simplifica el intercambio de datos.
- Se simplifica el transporte de datos.

- Se simplifica el cambio de plataforma.
- Se simplifica la disponibilidad de datos.

3.9 QTI

Question & Test Interoperability (QTI) [12] es un modelo de datos del IMS para representar ítems (preguntas), baterías de test (exámenes) y los resultados que han obtenido los alumnos.

QTI usa XML para el intercambiar los datos entre los sistemas de creación. Las representaciones de resultados y evaluaciones son usadas con una menor frecuencia. QTI provee de innovación e interoperabilidad mediante la correcta definición de puntos los cuales encapsulan datos y permiten el uso de los mismos junto con otros elementos representables de manera directa.

Las versiones que más repercusiones ha tenido esta especificación han sido:

- **Versión 1.2:** publicada en 2002. Esta versión contemplaba desde preguntas a nivel individual como exámenes completos. Fue de las primeras versiones por lo que al poco tiempo de publicarse surgieron problemas que había que modificar, pero hacer estas modificaciones suponía perder la compatibilidad existente con las versiones anteriores publicadas.
- **Versión 2.0:** publicada en 2005. Esta versión solucionó los problemas existentes en las antiguas versiones además de incorporar la compatibilidad hacia atrás, es decir, con antiguas versiones. Por temas de tiempo y simplicidad, esta versión solo se centró en las preguntas individuales, dejando a un lado la creación completa de exámenes.
- **Versión 2.1:** Comenzó en 2006. Esta versión continua con la simplicidad y evolución, sin embargo, en esta versión si se da soporte a la creación completa de exámenes y al intercambio de resultados entre los mismos.

Los objetivos del grupo de trabajo encargado de QTI son los siguientes:

- Habilidad de suministrar bancos de preguntas o exámenes a los alumnos que estén en docencia virtual.
- La posibilidad de usar bancos de preguntas y exámenes de procedencias distintas en un mismo sistema virtual de enseñanza (VLE).
- Ayuda para las herramientas capacitadas para la creación de nuevas preguntas y exámenes.
- Realización de informes detallados de las pruebas realizadas usando este método de enseñanza.

La estructura que forma un documento QTI se puede observar en la figura 3.9.

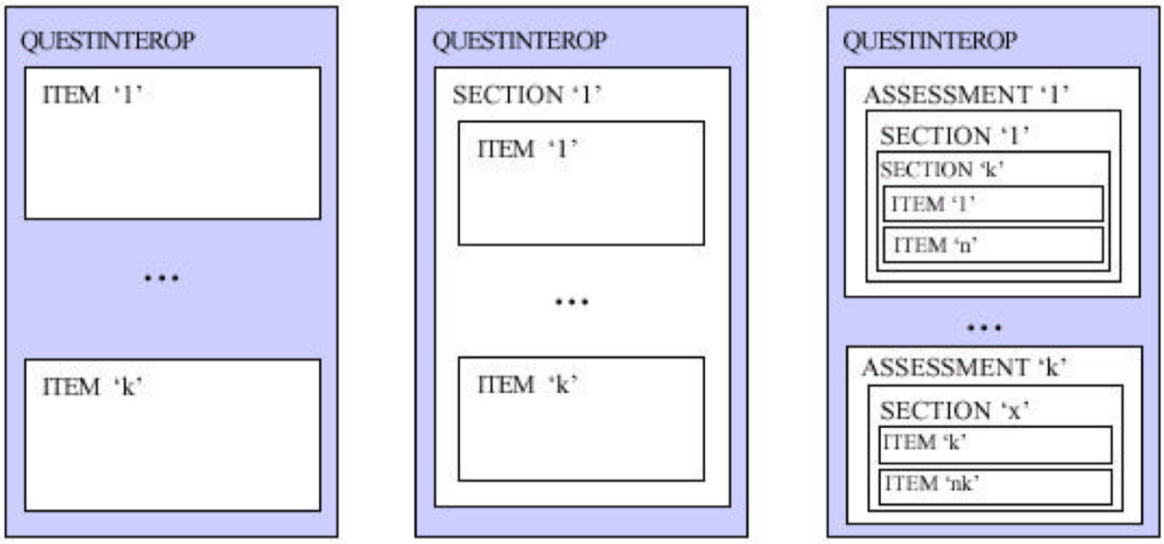


Figura 3.9 Estructura adoptada por QTI IMS. Fuente: http://pdi.topografia.upm.es/m.manso/docs/Estandar_QTI_Descripcion.pdf

En esta estructura se diferencian tres áreas:

- **ITEM:** Unidades mínimas que pueden intercambiarse usando QTI. Los Ítems no pueden, a su vez, estar formados por otro Ítems.
- **SECTIONS:** Una sección es una parte del documento QTI-XML. Un documento puede contener varias secciones y dentro de estas secciones puede haber varios Ítems.
- **ASSESSTMENT:** Un ASSESSTMENT es un examen dentro de un documento QTI-XML. Los exámenes están compuestos por al menos una sección.

Un sistema de exámenes está compuesto por:

- **Authoring System:** Procedimiento que da soporte a la edición y creación de exámenes, preguntas y secciones.
- **Assessment engine:** Procedimiento para la evaluación de las respuestas, generando las calificaciones pertinentes y refuerzos de estudio en el caso que fueran necesarios.
- **Learning Management System:** Sistema en el que recae la responsabilidad de la enseñanza y su gestión.
- **Candidate and Repository:** Base de datos que almacena las especificaciones de cada alumno (candidato).
- **ASI Repository:** Base de datos local que contiene las preguntas, exámenes y secciones.

- **Externar ASI Repository:** Base de datos externa la cual es importada usando las especificaciones del estándar QTI.

La figura 3.10 muestra un diagrama de casos de uso del sistema de exámenes.

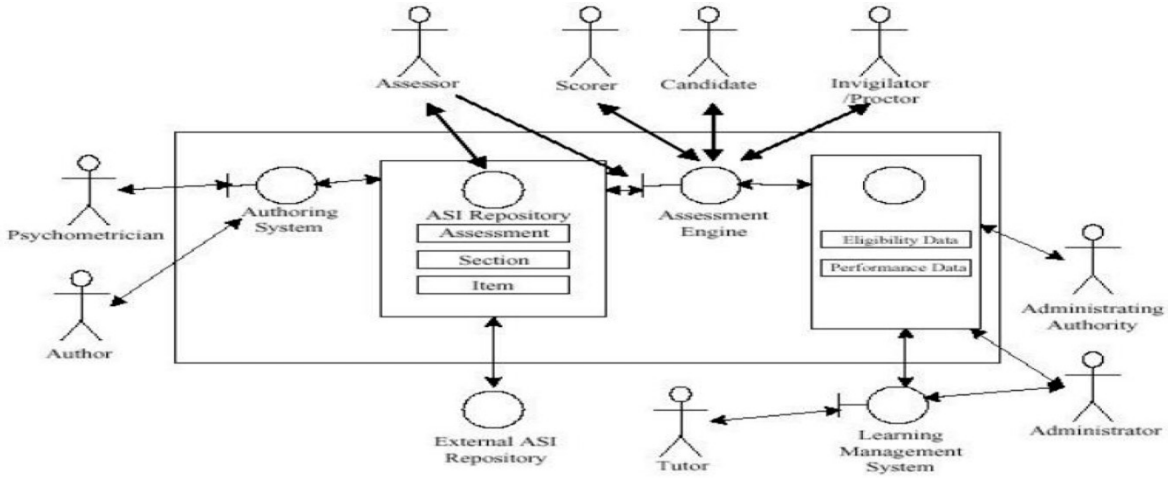


Figura 3.10 Casos de uso del sistema de exámenes. Fuente: http://pdi.topografia.upm.es/m.manso/docs/Estandar_QTI_Descripcion.pdf

3.10 SCORM

A la hora de hablar de e-learning, surgen dos ideas de manera muy rápida:

- Contenido de calidad.
- Compatibilidad con otras plataformas o LMS.

El contenido de esta forma de educación es realizado a través de software mediante programación, por lo que la calidad del mismo ha ido aumentando conforme se evolucionaban más las tecnologías que se usan para crear este tipo de formación. A día de hoy todo el mundo es consciente de la repercusión que ha cogido internet a la hora de hacer cualquier tipo de cosas que se pueden hacer en la vida cotidiana y las tecnologías han ido evolucionando para hacer más sencillos realizar estos procesos y, además, poder hacerlas de manera telemática, es decir, sin salir de casa.

Por otra parte, al hablar de compatibilidad con otros LMS entran en juego los estándares y pese a que existen acuerdos entre organismos de educación y organizaciones para establecer formatos y protocolos que sean compatibles con distintos LMS, no hay una continuidad y un acuerdo para su uso y fruto de esto, muchos acuerdos y propuestas fueron desapareciendo sin respetarse.

El afán de las empresas de mejorar su contenido las ha llevado a ignorar estos estándares y como consecuencia, el contenido creado es incompatible. El resultado de

esto fue que sus clientes (alumnos) abandonaron su contenido para buscar otro más compatible y con más capacidad de adaptación a otras plataformas o sistemas operativos.

El abandono y el descenso de clientes llevo a las empresas a crear un contenido interoperable que fuera reutilizable, duradero y portable entre sistemas diferentes. Con este tipo de contenido las empresas y los vendedores de sistemas gestión ganaban en tiempo y dinero ya que su producto puede ser utilizado en lugares distintos sin necesidad de estar adaptándolo a distintas librerías para su correcto funcionamiento en otras plataformas.

Así fue como Advanced Distributed Learning Initiative (ADL) desarrolló una serie de normas y especificaciones para estandarizar de manera oficial el contenido de aprendizaje electrónico. De este procedimiento surgió Sharable Content Object Reference Model (SCORM).

SCORM se trata de un conjunto de estándares relacionados con el e-learning que, a pesar de recibir algunas críticas en sus orígenes, a día de hoy se puede decir que es el estándar de facto que se usa en la creación de contenido e-learning. [5]

3.10.1 *Versiones de SCORM*

La primera versión fue SCORM 1.0 y se publicó en el año 2000. Como se muestra en la figura 3.11 tras la primera versión fueron surgiendo otras versiones hasta que en 2009 se publicó la última, SCORM 2004v4.

Se hablará sobre las versiones que más repercusión tuvieron en el mundo del e-learning: SCORM 1.2 y SCORM 2004v4.

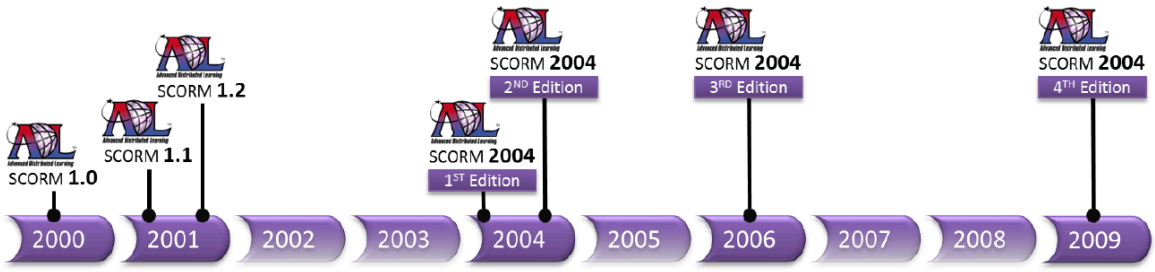


Figura 3.11 Línea temporal de versiones de SCORM. Fuente: http://ruja.ujaen.es/handle/10953/797

3.10.1.1 Versión 1.2

SCORM 1.2 incorporó todas las lecciones aprendidas de las primeras adopciones de SCORM 1.1 para crear una especificación robusta e implementable que demostrase que el contenido puede ser interoperable y portable. Los vendedores que adoptaron SCORM 1.2 se dieron cuenta de los dramáticos ahorros de costos gracias a la mayor interoperabilidad de contenido.

SCORM 1.2 contaba con algunas ambigüedades que necesitaban ser fortificadas y carecía de una especificación de secuenciación y navegación que permitiera al proveedor de contenido explicar cómo el alumno podía progresar entre los distintos SCOs.

3.10.1.2 Versión 2004

SCORM 2004 (en todos sus aspectos) incluye versiones muy maduras de los paquetes de contenido, tiempo de ejecución y libros de metadatos. Los libros compuestos por estos estándares individuales se convertirán en estándares acreditados.

SCORM 2004 trajo consigo las nuevas funcionalidades llamadas "Secuenciación y navegación". Esta especificación permite la creación de reglas para conocer cómo los usuarios pueden navegar entre los diferentes SCOs. Un ejemplo de reglas sería decir que "un alumno no puede hacer un examen final hasta que haya finalizado los contenidos previos al examen del curso". O, "si un alumno falla en la pregunta X, mándalo de vuelta a SCO Y".

El término "SCORM 2004" se usa para referirse a cualquier edición de la especificación SCORM 2004(v1, v2, v3 o v4).

3.10.2 *¿Qué es SCORM?*

SCORM es un conjunto de normas en forma de especificaciones que explica a los desarrolladores como crear un código para que este se pueda reproducir en cualquier plataforma de gestión de aprendizaje.

SCORM está compuesto por tres sub-especificaciones los cuales indican como crear un paquete SCORM con contenido e-learning. Estas especificaciones describen crear contenido y empaquetarlo, la secuenciación y navegación por estos contenidos y, por último, como el contenido puede interactuar con el LMS para el intercambio de datos.

Un paquete que haya sido creado siguiendo las especificaciones SCORM debe atender a los siguientes criterios:

- **Accesibilidad:** tener acceso a los recursos educativos desde cualquier lugar utilizando las tecnologías web.
- **Adaptabilidad:** poder modificar el recurso educativo en función de la finalidad del aprendizaje.
- **Durabilidad:** capacidad de adaptación a la evolución tecnológica sin necesidad de modificar/reescribir el código o rehacer el recurso educativo.
- **Interoperabilidad:** Integración con otros sistemas de gestión de aprendizaje o sobre otras plataformas.

- **Reusabilidad:** capacidad de integrar dividir los componentes de un recurso en componentes más pequeños compatibles con diferentes contextos y aplicaciones.

3.10.3 *Contenido del paquete (sub-especificación CAM)*

El modelo de contenido de SCORM especifica que el empaquetado tendrá formato comprimido ZIP (archivo PIF, Packet Interchange File). En el interior del PIF siempre habrá un fichero XML denominado imsmanifest.xml el cual contiene la información necesaria para que el LMS interprete el contenido de la forma adecuada.

La sub-especificación CAM, Modelo de Agregación de Contenidos de SCORM [22] describe que el archivo imsmanifest.xml divide el contenido del curso en distintos apartados denominados SCOs y a su vez contiene información de distinto tipo relativa a los distintos SCOs que componen el paquete (representación de SCO, arranque de SCO y metadatos de SCO).

3.10.4 *Navegación y secuenciación (sub-especificación SN)*

La sub-especificación SN (Secuenciación y Navegación) [23] describe cómo el autor puede administrar el contenido para que la presentación al alumno siga un camino u otro a la vez que se van almacenando los datos del progreso del alumno.

Las reglas que permiten al autor esto se definen en el archivo imsmanifest.xml y se deben de incluir haciendo referencia a los elementos a los que se le aplica. Un ejemplo de esto se puede ver en la figura 3.12

```
<title>OLSR_TFG_v2_preguntas</title>

<item identifier="SCO1" identifierref="RES1" isvisible="true">
  <title>Introduccion</title>
</item>

<item identifier="SCO2" identifierref="RES2" isvisible="true">
  <title>Teoria</title>
</item>

<item identifier="SCO3" identifierref="RES3" isvisible="true">
  <title>Descripción y conceptos</title>
</item>
<item identifier="SCO4" identifierref="RES4" isvisible="true">
  <title>Redes MANET protocolo OLSR</title>
</item>
<imsss:sequencing>
  <imsss:controlMode flow="true" choice="true"/>
  <imsss:rollupRules rollupObjectiveSatisfied="true" rollupProgressCompletion="true" objectiveMeasureWeight="0"/>
</imsss:sequencing>
</organization>
</organizations>
<resources>
<resource identifier="RES1" href="intro_OLSR.html" adlcp:scormType="sco" type="webcontent" xml:base="/">
  <file href="intro_OLSR.html" />
  <dependency identifierref="API-DEP" />
  <dependency identifierref="API-DEP1" />
</resource>
<resource identifier="RES2" href="Teoria_OLSR.html" adlcp:scormType="sco" type="webcontent" xml:base="/">
  <file href="Teoria_OLSR.xhtml" />
  <dependency identifierref="API-DEP" />
  <dependency identifierref="API-DEP1" />
</resource>
<resource identifier="RES3" href="descripcion_simulacion_OLSR.html" adlcp:scormType="sco" type="webcontent" xml:base="/">
  <file href="descripcion_simulacion_OLSR.html" />
```

Figura 3.12 Captura de archivo XML de un paquete SCORM. Fuente: Propia

Las reglas de secuenciación permiten al autor hacer cosas como las siguientes:

- Permitir que el alumno elija cualquiera de los SCOs disponibles.
- Obligar al alumno a llevar una secuencia estricta, donde cada el objetivo del SCO que se está trabajando es un prerrequisito para el siguiente SCO.
- Creación de SCOs (secciones) opcionales.
- Hacer repetir al estudiante una determinada parte del curso si no superó el examen [24] .
- Etc.

3.10.5 *Run Time (sub-especificación RTE)*

La sub-especificación RTE (Run-Time Environment) se encarga de controlar la forma en la que el LMS ofrece el contenido además de las normas que se rigen en la comunicación que se establece entre el LMS y el contenido. Una vez presentado de manera correcta, el contenido se apoya en una API proporcionada por el LMS, denominada API SCORM (Application Program Interface) para poder realizar el intercambio de datos entre el SCO y el LMS.

Esta sub-especificación controla que el LMS ofrece el contenido a través de un navegador web y que solo se lanza un SCO cada vez ya que el LMS solo puede lanzar un SCO en cada interacción.

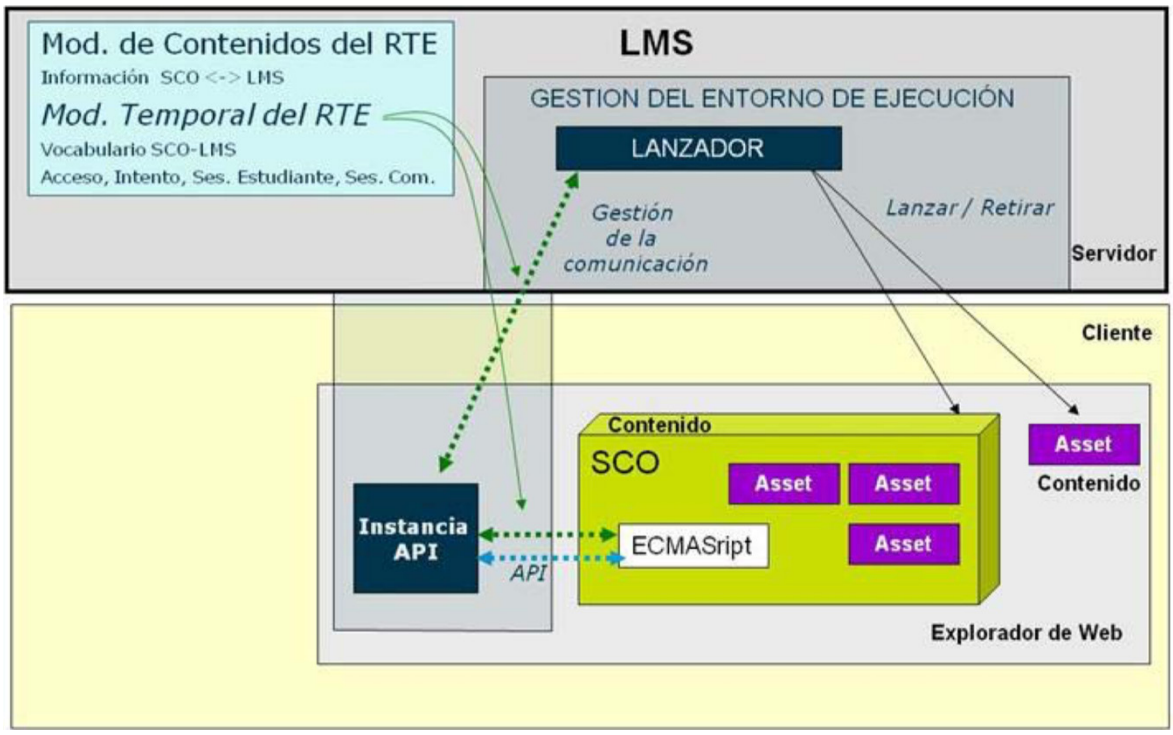


Figura 3.13 Relación entre LMS y paquete SCORM. Fuente: http://www.unpa.edu.ar/sites/default/files/descargas/Administracion_y_Apoyo/Materiales/2016/T129/SCORM_Standar.pdf

4 OBJETIVOS

El principal objetivo de este TFG es el desarrollo de dos laboratorios virtuales autónomos para que el alumno desarrolle y ponga a prueba sus conocimientos sobre redes ad-hoc. Otro de los objetivos de este TFG es no ofrecer estas simulaciones de forma aislada, sino que se presenten a los alumnos integradas en un itinerario de aprendizaje que incluya parte teórica y test de evaluación.

Para lograr estos objetivos finales existen una serie de objetivos secundarios a cumplir, los cuales son:

- Desarrollo de simulaciones
  - Estudio de lenguaje de programación JavaScript.
  - Estudio de herramienta de software usada para la creación de simulaciones EJS (*Easy Java Simulations*).
  - Estudio de redes Ad-Hoc, concretamente MANET (*Mobile Ad-Hoc Network*) y VANET (*Vehicular Ad-Hoc Network*).
  - Estudio de protocolos de enrutamiento que utilizan las redes Ad-Hoc, concretamente OLSR (*Optimized Link State Routing*).
  - Desarrollo de simulación VANET.
  - Desarrollo de simulación OLSR.

- Crear test de preguntas
  - Desarrollo de bancos de preguntas test basados en los conocimientos explicados en el WebLab y los temas correspondientes de la asignatura Redes Basadas en Dispositivos Móviles impartida en 4º curso de Ingeniería Telemática.
  - Estudio de estándar QTI.
  - Estudio y comprensión de librerías JavaScript desarrolladas para ejecutar y crear test de preguntas.

- Módulos SCORM
  - Estudio de normas y estándares SCORM.
  - Estudio y comprensión de WebLabs proporcionados, ya realizados en base a SCORM.
  - Estudio del estándar XHTML.
  - Creación y edición de páginas web usando HTML y JavaScript.
  - Modificación de páginas web usando hojas de estilo (CSS).
  - Desarrollo de WebLabs completos y adaptados a las dos versiones más usadas de SCORM (v1.2 y v2004).

- Exposición de WebLabs en un LMS, concretamente en Ilias.

  Otros objetivos

- Uso de tecnologías y herramientas gratuitas, de uso libre.
- Diseño de contenidos usando estilo similar a LMS UJA

5 MATERIALES Y MÉTODOS

En esta sección se mostrarán los lenguajes de programación usados para el desarrollo de las simulaciones así como la metodología de trabajo seguida para la creación, edición y subida de los módulos elaborados.

Se hablará del entorno de trabajo integrado y las distintas herramientas usadas para el desarrollo de los WebLabs.

5.1 Desarrollo y Metodología de Trabajo

En este apartado se expondrá la metodología de trabajo seguida a la hora de realizar algunos aspectos y funcionalidades del TFG desarrollado, concretamente en los módulos SCORM implementados.

En primer lugar, se tuvo que hacer un estudio inicial y diseño, usando para ello internet e ILIAS. Este estudio derivó entres resultados:

- Diseño de la simulación: diseño inicial de las simulaciones a implementar, estructura y contenido de las mismas.
- Teoría a incorporar: teoría que se incorporará en el módulo para que el alumno obtenga una serie de conocimientos.
- Ideas a preguntar: conjunto de preguntas de tipo test sobre la teoría explicada y los temas de la asignatura Redes Basadas en Dispositivos Móviles.

Tras esto se procede a la elaboración de la simulación y la descripción de la misma, usando para ello la herramienta EJS. Una vez realizada dicha simulación se exporta el proyecto desde EJS con formato SCORM y el resultado es un fichero ZIP.

Una vez obtenido este fichero se procede a la elaboración de las preguntas de tipo test, previamente pensadas, en un banco de preguntas perteneciente a ILIAS, el cual permite, una vez creado este banco de preguntas, exportar el mismo en formato QTI. Se Obtiene así, otro fichero ZIP en cuyo interior se encontrará el fichero con QTI que contiene las preguntas realizadas en el banco de preguntas.

Obtenido todos los archivos anteriores, se procede a la manipulación e integración de estos archivos en el paquete SCORM obtenido con EJS, además, habrá que añadir también las demás páginas que contendrá el módulo, usando para ello el IDE NetBeans.

Al añadir estos archivos se obtiene un nuevo fichero en formato ZIP que será el que contenga el módulo en su totalidad y se subirá al LMS para su verificación.

Una vez en el proceso de verificación se tendrán que controlar varios aspectos del módulo como:

- Aspecto visual.

- Test.
- Simulación.

Si se encuentra algún tipo de anomalía en la simulación o el los test, se deberá volver al proceso en el que se crean los mismos, corregirlos y volver a realizar todos los procesos siguientes.

Si, por el contrario, solo se encuentran anomalías en el aspecto visual de alguna de las páginas que integran el módulo, deberemos de ir al proceso de manipulación e integración SCOM y usando el IDE NetBeans, corregir esas anomalías. Tras esto se deberán de repetir los procesos siguientes al mencionado.

Como se puede observar, para el simple hecho de verificar si el módulo contiene algún tipo de anomalía, se necesitan realizar una gran cantidad de procesos, por lo que el trabajo de verificación puede ser bastante laborioso, dependiendo de donde encontremos la anomalía.

La figura 5.1 muestra en forma de flujograma todos los procesos y todas las posibilidades que se pueden dar.

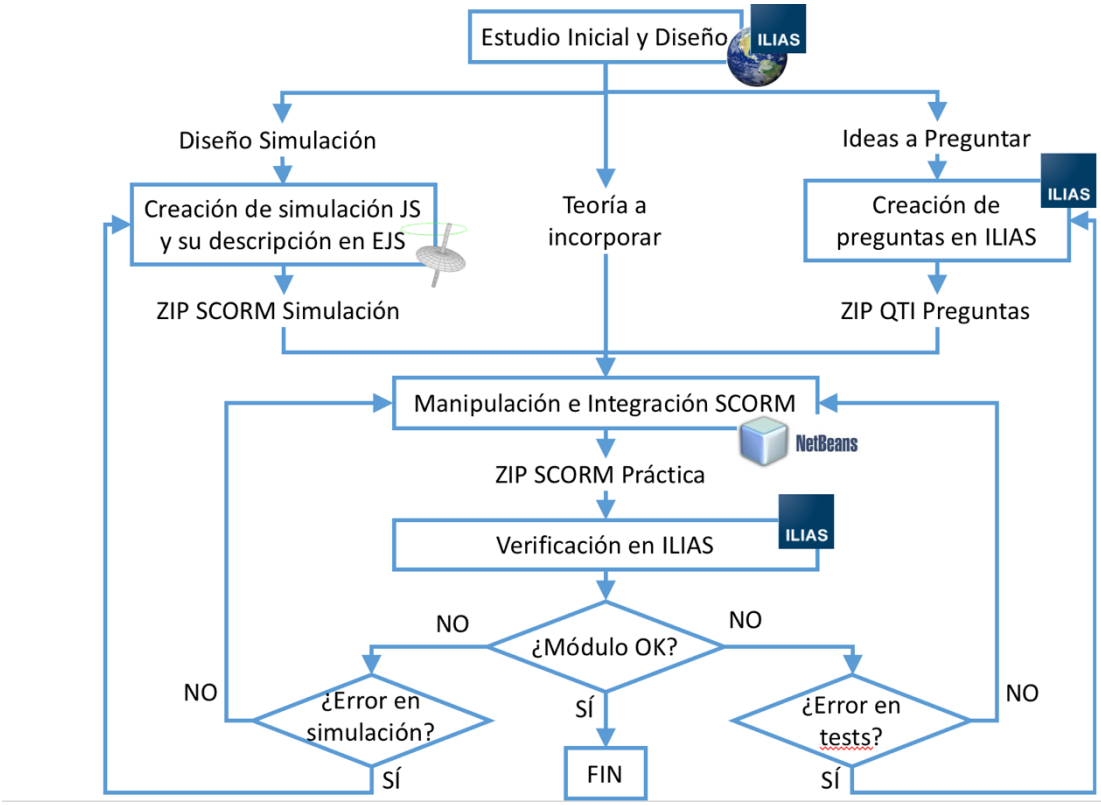


Figura 5.1 Flujograma de trabajo. Fuente: Propia.

A continuación, se explicará de forma más detallada los procesos:

- Manipulación e integración SCORM.
- Creación e inserción de test.

- Subida del proyecto al LMS (ILIAS).

5.1.1 *Manipulación e integración SCORM*

Una vez realizado todo el trabajo en EJS se ha procedido a realizar un proceso de retoque de los distintos SCOs para evaluar el comportamiento y correcto funcionamiento del módulo. Para ellos se han realizado una serie de sub-procesos para los cuales se han usado las siguientes herramientas:

- EJS.
- IDE NetBeans.
- HTML y CSS.
- JavaScript.
- Navegadores web.
- LMS (ILIAS).

Los sub-procesos se podrán ver en forma de flujograma en la figura 5.2.

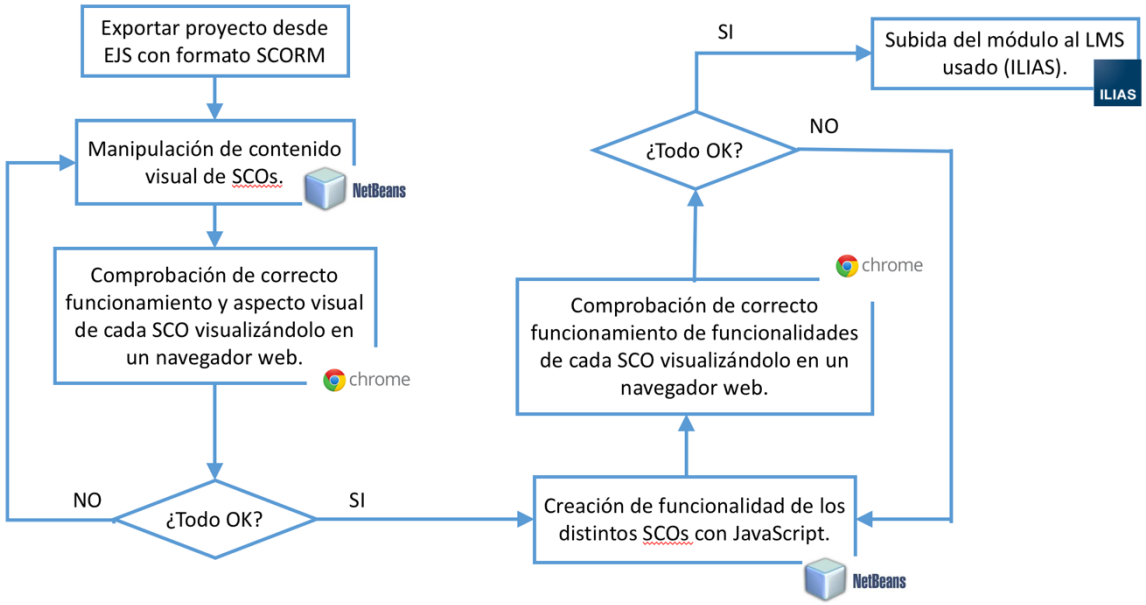


Figura 5.2. Flujograma manipulación SCOs. Fuente: Propia.

5.1.2 *Creación e inserción de test.*

Una vez comprobado que el aspecto visual, la secuenciación entre SCOs, las funcionalidades interactivas, etc. funcionan de forma correcta, se realizará la creación e inserción de los test de evaluación en los módulos. Para cumplir con este proceso se han utilizado las siguientes herramientas:

- IDE NetBeans.
- HTML y CSS.

- JavaScript.
- Navegadores web.
- LMS (ILIAS).

Los sub-procesos llevados a cabo para la creación e inserción de los test se pueden observar en forma de flujograma en la figura 5.3.

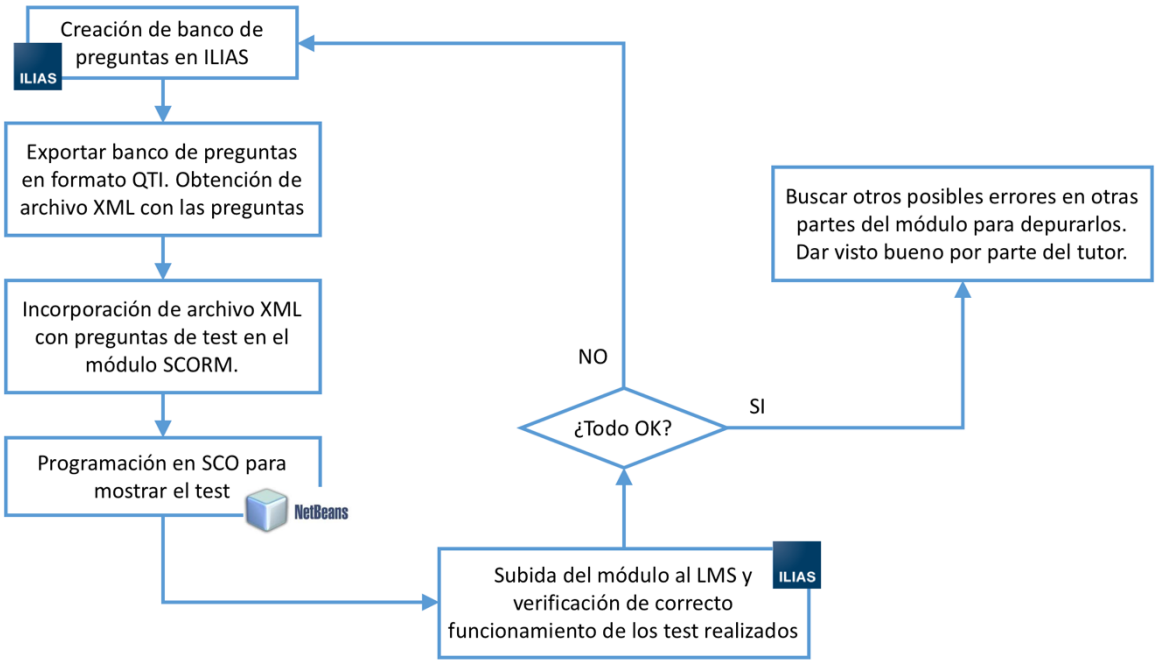


Figura 5.3. Creación e inserción de test. Fuente: Propia.

5.1.3 *Subida del proyecto al LMS (ILIAS).*

En todos los procesos anteriores, siempre se ha tenido que usar, al menos una vez, el proceso de subida al LMS utilizado, ya que una vez ahí es donde se observan mejor los errores que puedan tener los módulos y, además, gracias a debugger que incorpora el LMS, se puede observar en tiempo real las comunicaciones que se hacen con el LMS.

Para llevar a cabo el proceso de subir los módulos al LMS se han llevado a cabo una serie de sub-procesos para los cuales se han utilizado las siguientes herramientas:

- LMS (ILIAS).

Los sub-procesos llevados a cabo para la subida de los módulos se pueden observar en forma de flujograma en la figura 5.4.

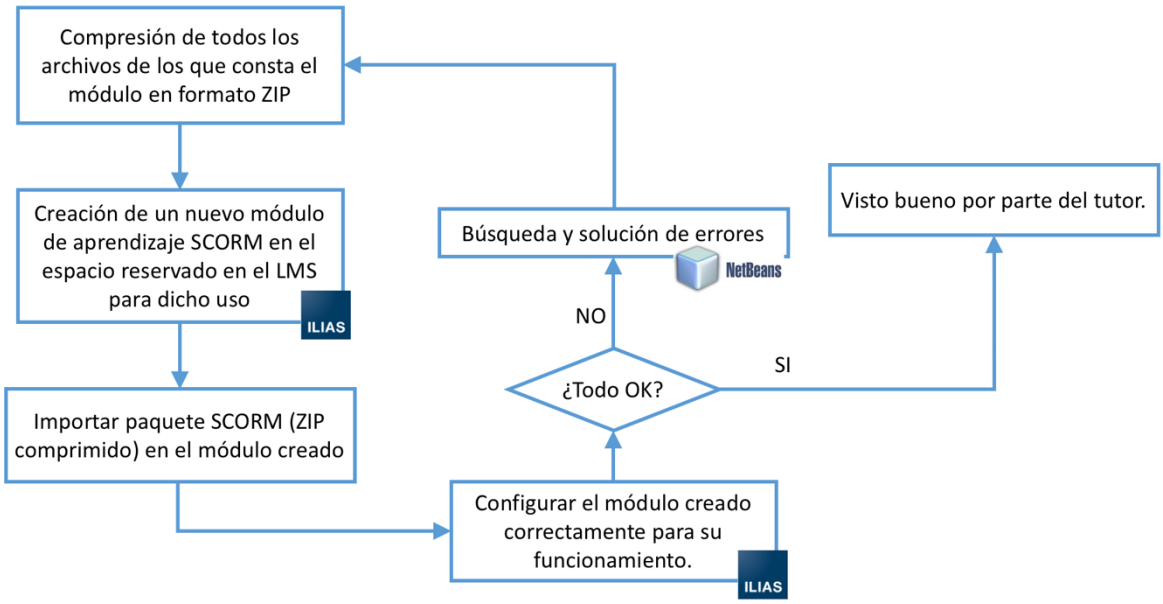


Figura 5.4. Subida de módulo a LMS. Fuente: Propia.

5.2 JavaScript

Este lenguaje ha sido elegido debido a la naturaleza del TFG y también a la mejor integración que ofrece este lenguaje en los navegadores web que es donde se visualizarán los distintos WebLabs desarrollados.

JavaScript cuenta con un amplio número de ventajas como [25]:

- Se trata de un lenguaje sencillo.
- Se ejecuta en el lado del cliente, por lo que, la ejecución de funciones tiende a ser inmediata.
- Tiene un amplio número de efectos visuales.
- Los navegadores más utilizados y famosos soportan JavaScript.
- Su versatilidad es muy útil para el desarrollo de aplicaciones y páginas web dinámicas.
- Cuenta con una funcionalidad de validación de datos en formularios.
- Se puede ejecutar en distintos sistemas operativos móviles debido a que es multiplataforma.
- Agiliza el procedimiento de desarrollo.
- Aprendizaje de manera rápida.
- Frecuente actualización y gran soporte.
- Lenguaje de scripting fiable y seguro.
- Scripts limitados en capacidad por motivos de seguridad.

5.3 Herramientas de desarrollo de software

Para el desarrollo del código que implementa las simulaciones incluidas en los WebLabs se han utilizado de manera conjunta el propio editor de texto que incorpora EJS (*Easy Java Simulations*) y el entorno de programación integrado (IDE) NetBeans, cuyos logos se aprecian en la figura 5.5.

Figura 5.5 Logos de NetBeans (izquierda) y Easy Java Simulations (derecha). Fuente Netbeans: http://www.orlandobarcia.com/wp-content/uploads/2015/03/netbeans-logo-e1426181313229.png. Fuente EJS: Easy Java Simulations software.

Se han usado conjuntamente ya que EJS es donde se tiene que almacenar el código para generar la simulación, sin embargo, el editor de texto integrado en EJS no dispone de corrección de errores en tiempo real ni de auto-completar el código, dificultando la tarea al programador. Por esto se usa NetBeans para poder realizar el código de una manera más sencilla y fluida y después pasarlo al editor de texto integrado de EJS. NetBeans cuenta con un gran abanico de posibilidades a la hora de programar ya que incluye varios de tipos de lenguajes de programación con los que se pueden crear proyectos desde cero. NetBeans es fácil de utilizar y su interfaz es muy intuitiva.

Cabe destacar que NetBeans ha sido utilizado con fines que podría suplir perfectamente un editor de texto como tales como NotePad++, SublimeText, Bloc de notas, etc. Personalmente lo elegí ya que, al ser un IDE, ofrece las funcionalidades de un editor de texto además de las propias de un IDE.

Para la edición y creación de contenido HTML, XHTML, CSS, XML se ha usado únicamente el IDE NetBeans.

5.3.1 *Easy Java Simulations (EJS)*

Para sacarle el máximo rendimiento a la enseñanza virtual, es conveniente utilizar herramientas que permitan visualizar de manera gráfica el contenido que se quiere transmitir, para ello se ha utilizado EJS (*Easy Java Simulations*) [3] que permite crear

simulaciones gráficas de manera sencilla. EJS está diseñado para ayudar a los estudiantes de ciencias a comprender sus materias y poder hacerlo mediante la visualización gráfica de simulaciones, las cuales harán que el alumno comprenda mejor el tema que esté tratando. EJS permite el desarrollo de aplicaciones Java independientes y multiplataforma, o *applets* que son compatibles con cualquier navegador Web.

Otra ventaja que ofrece EJS es que permite exportar los proyectos con formato SCORM (*Sharable Content Object Reference Model*) [4] , aunque después se han tenido que modificar los ficheros para adaptarlos a los objetivos.

La interfaz que ofrece EJS es sencillo e intuitivo, por lo que lo hace una herramienta muy fácil de entender y usar. En la figura 5.6 se puede ver esta interfaz.

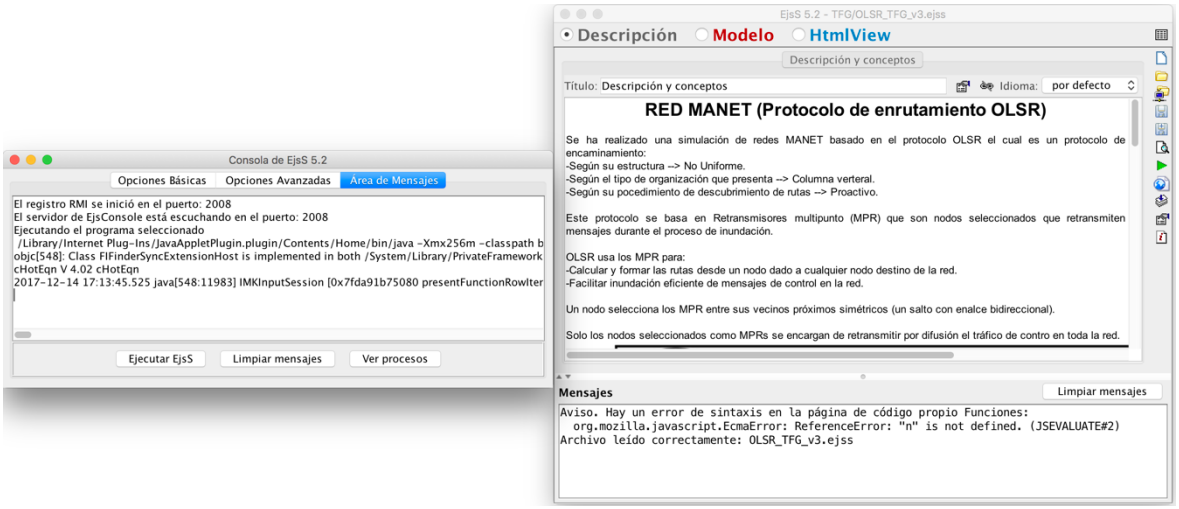


Figura 5.6 Interfaz gráfica EJS. Fuente: Propia

Como se ha podido observar, al iniciar el programa, se muestran dos ventanas distintas, a la izquierda se encuentra la consola integrada en el programa y a la derecha se encuentra la ventana donde se realizará el proyecto.

Por un lado, la consola está compuesta por tres pestañas en las que se encuentran los elementos necesarios para configurar el programa a nuestro gusto, elegir el directorio donde se almacenarán nuestros proyectos, elegir el lenguaje de programación que se usará (Java o JavaScript), etc.

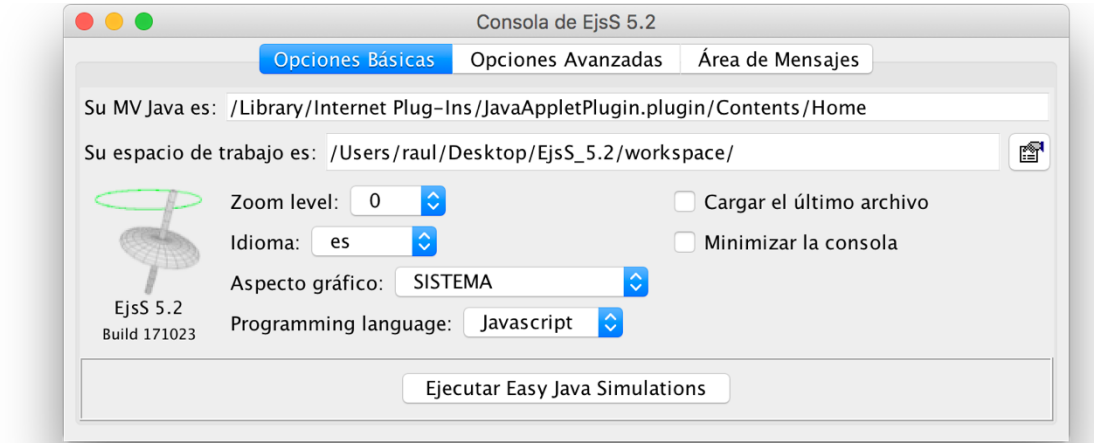


Figura 5.7 Pestaña Opciones Básicas. Fuente: Propia

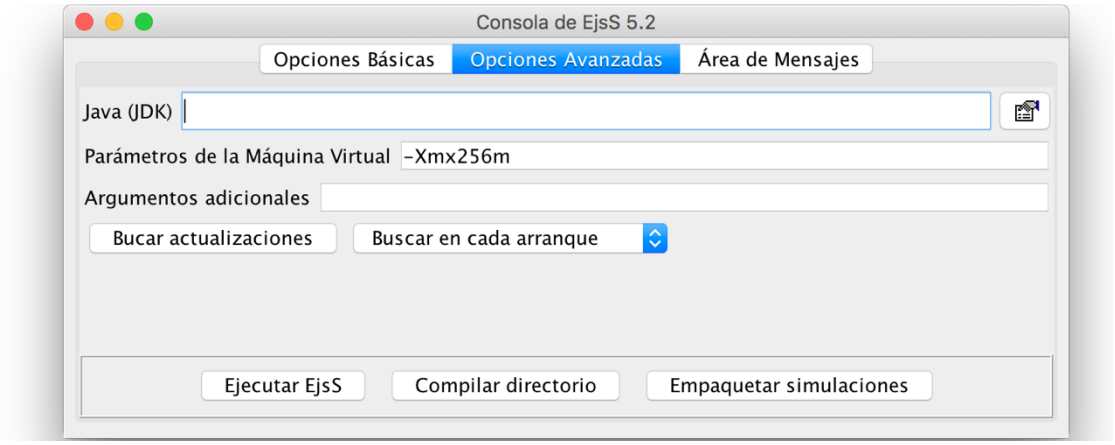


Figura 5.8 Pestaña Opciones Avanzadas. Fuente: Propia

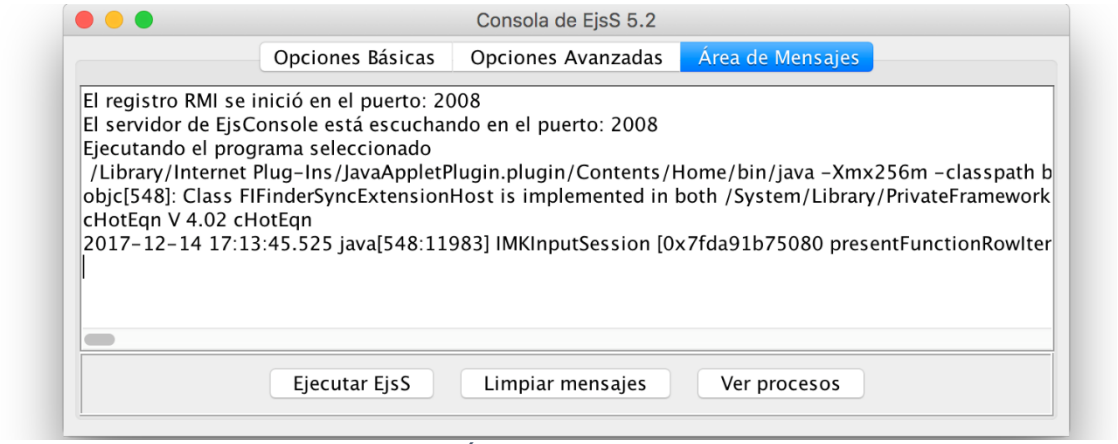


Figura 5.9 Pestaña Área de Mensajes. Fuente: Propia

Por otro lado, se podrá observar la ventana donde se realizará todo lo necesario para desarrollar el proyecto. Esta ventana contiene tres pestañas que ayudará a la realización del proyecto que son: Descripción, Modelo y HtmlView.

DESCRIPCIÓN

En esta pestaña se puede crear una descripción de nuestro proyecto que luego será convertida y mostrada en formato HTML de manera automática.

La manera de crear esta descripción es muy sencilla ya que el comportamiento que tiene es muy similar a la que pueden tener programas famosos como es Microsoft Word. Simplemente consiste en escribir de que trata nuestro proyecto como si en un documento de Word se estuviera haciendo, incluso se puede agregar imágenes, tablas, formularios, etc.

La figura 5.10 muestra el aspecto de esta ventana.

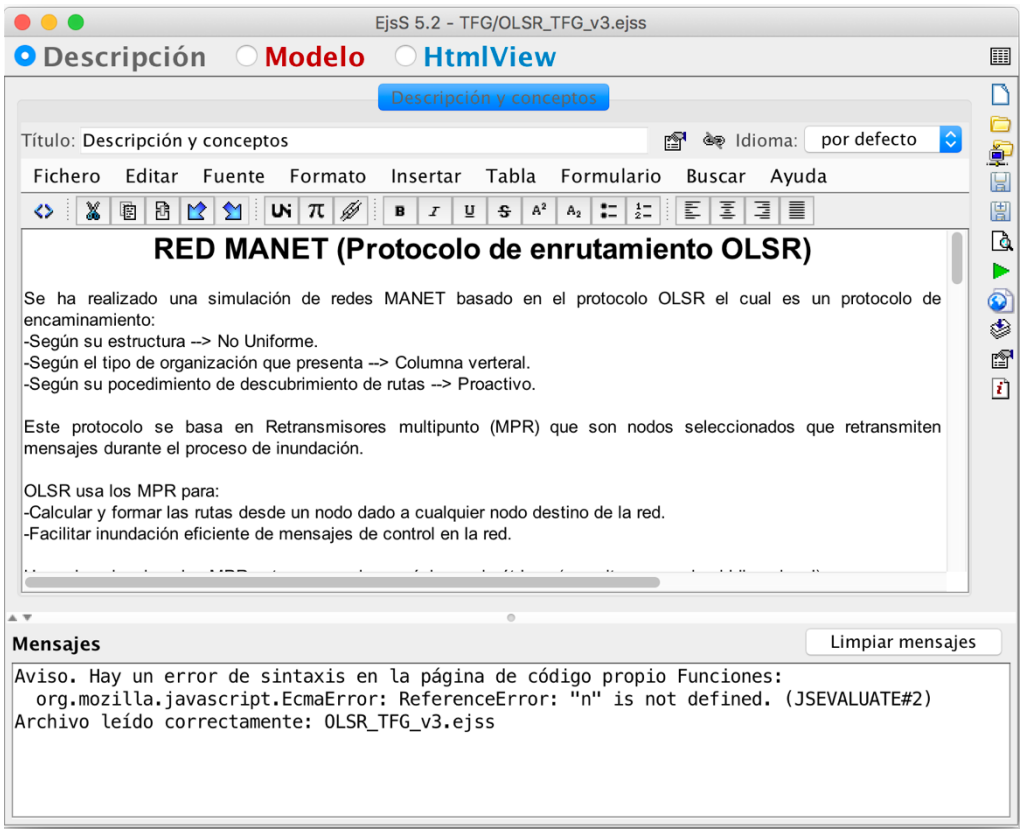


Figura 5.10 Pestaña Descripción EJS. Fuente: Propia

MODELO

Esta pestaña contendrá el grueso de nuestro proyecto, es decir, lo que en entornos de programación convencionales se denominarían “archivos de código”. Estará compuesta a su vez por sub-pestañas que organizarán en código de una manera más estructurada. En la figura 5.11 se puede ver el aspecto de esta ventana.

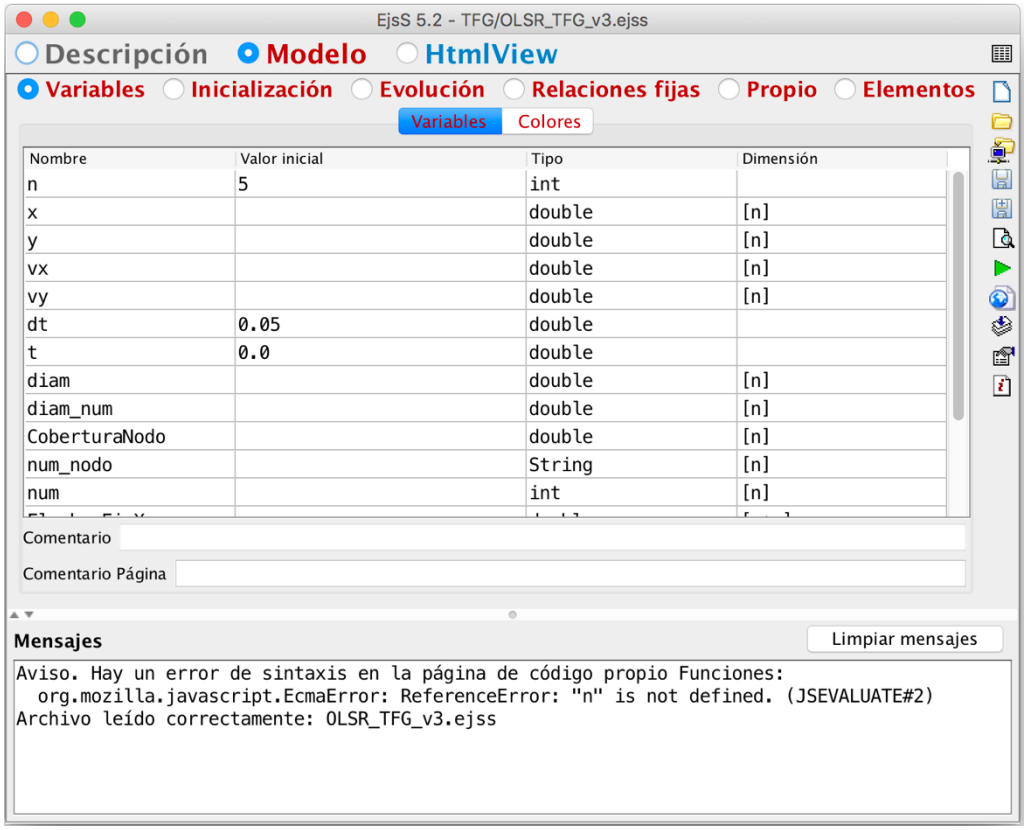


Figura 5.11 Pestaña Modelo EJS. Fuente: Propia

Como se ha podido observar las sub-pestañas de las que consta esta pestaña son las siguientes:

- **Variables:** Aquí es donde se definen las variables que se van a utilizar en las funciones que posteriormente serán implementadas (Figura 5.12).
- **Inicialización:** Contendrá el código que inicializará el programa o una llamada a la función que lo inicializa definida en la sub-pestaña “Propio” (Figura 5.13).
- **Evolución:** Contendrá el código que se ejecutará constantemente a modo de bucle o una llamada a las funciones que evolucionen con el paso de tiempo definida en la sub-pestaña “Propio” (Figura 5.14).
- **Relaciones Fijas:** Contendrá el código que a efectos del proyecto no se pueda o se deba modificar (Figura 5.15).

- **Propio:** Contendrá todo el código en forma de funciones que se usará en el proyecto (Figura 5.16).
- **Elementos:** Contendrá elementos (normalmente hardware) que se quiera utilizar en el proyecto, pero estos elementos están elaborados mediante software (Figura 5.17).

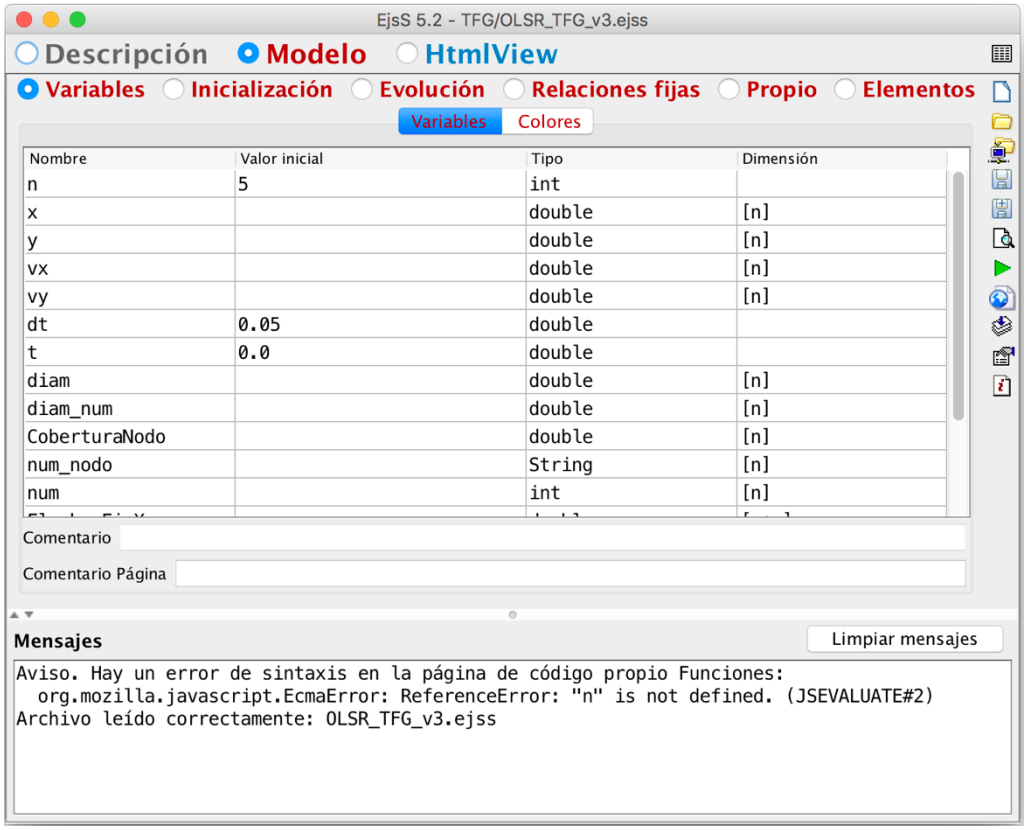


Figura 5.12 Sub-pestaña Variables EJS. Fuente: Propia

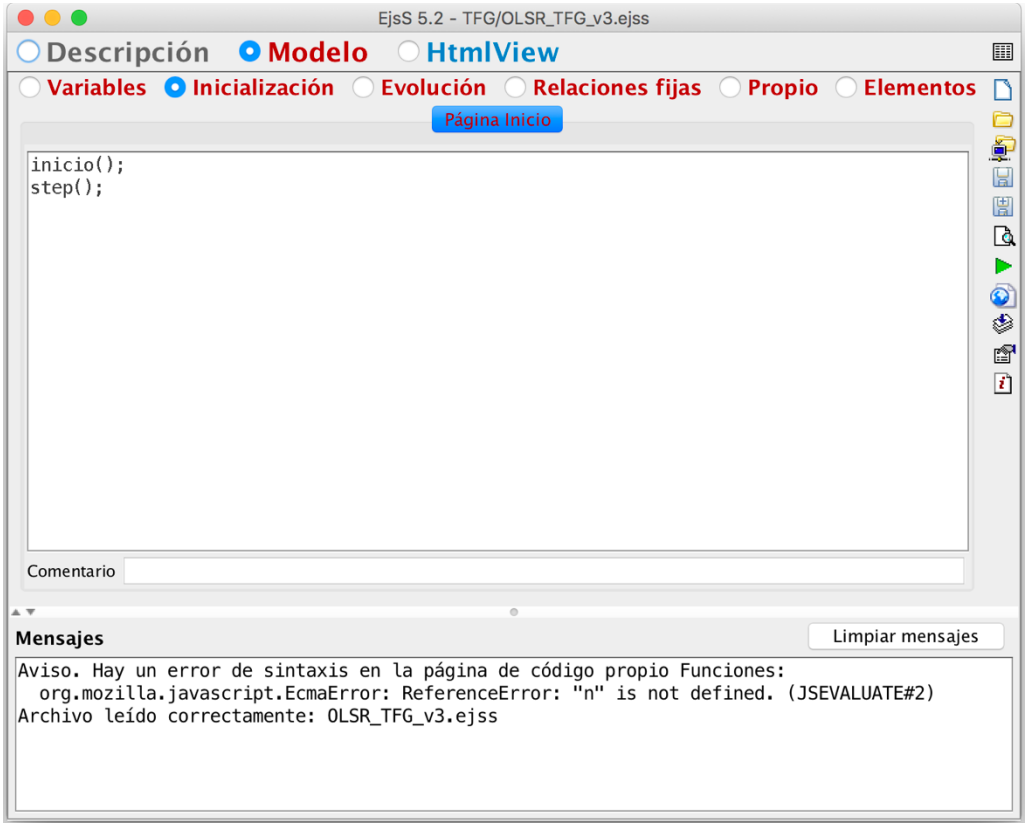


Figura 5.13 Sub-pestaña Inicialización EJS. Fuente: Propia

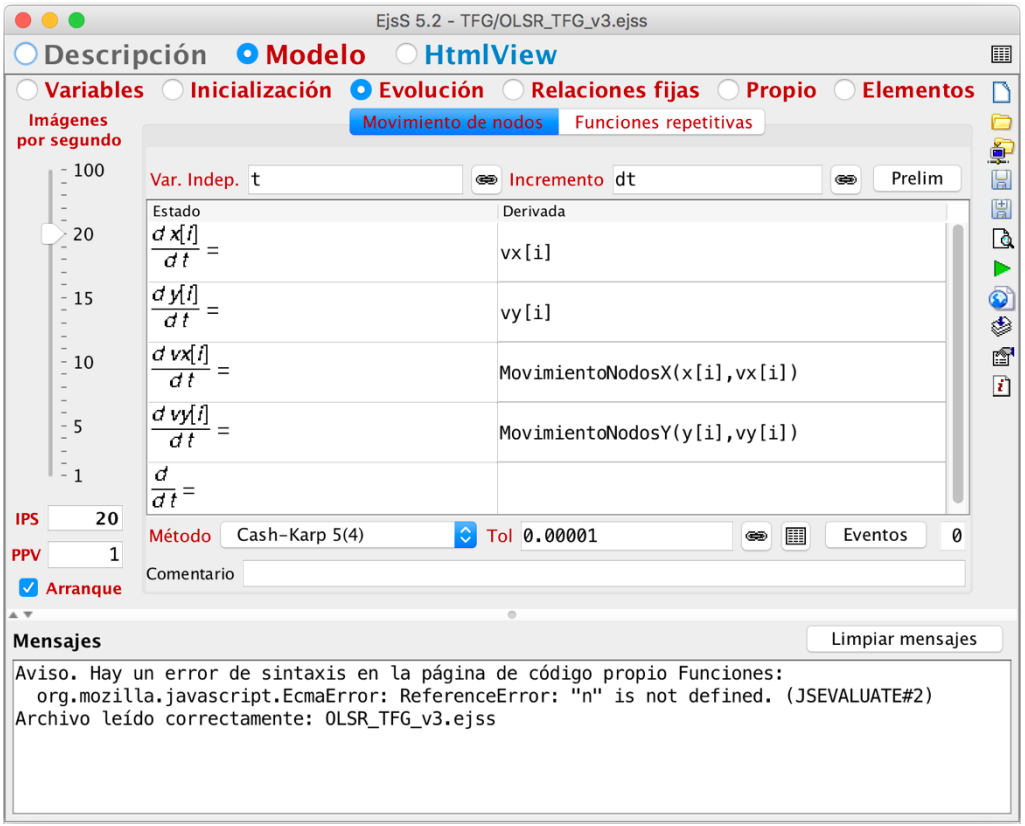


Figura 5.14 Sub-pestaña Evolución EJS. Fuente: Propia

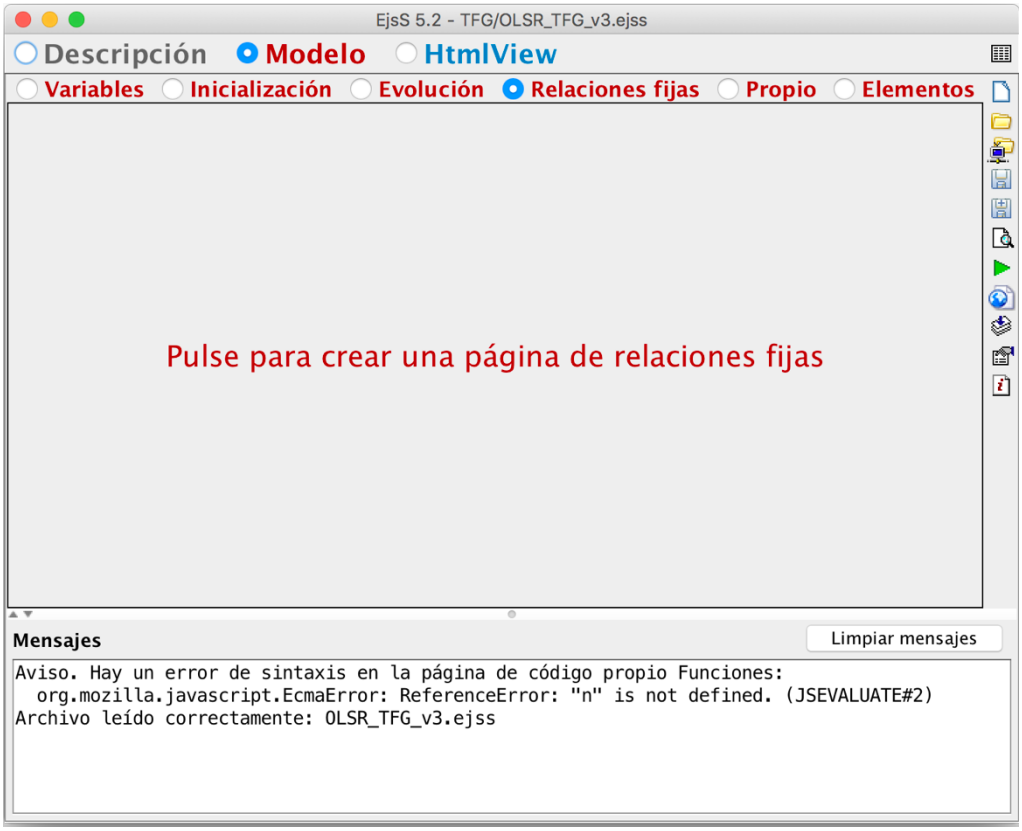


Figura 5.15 Sub-pestaña Relaciones Fijas EJS. Fuente: Propia

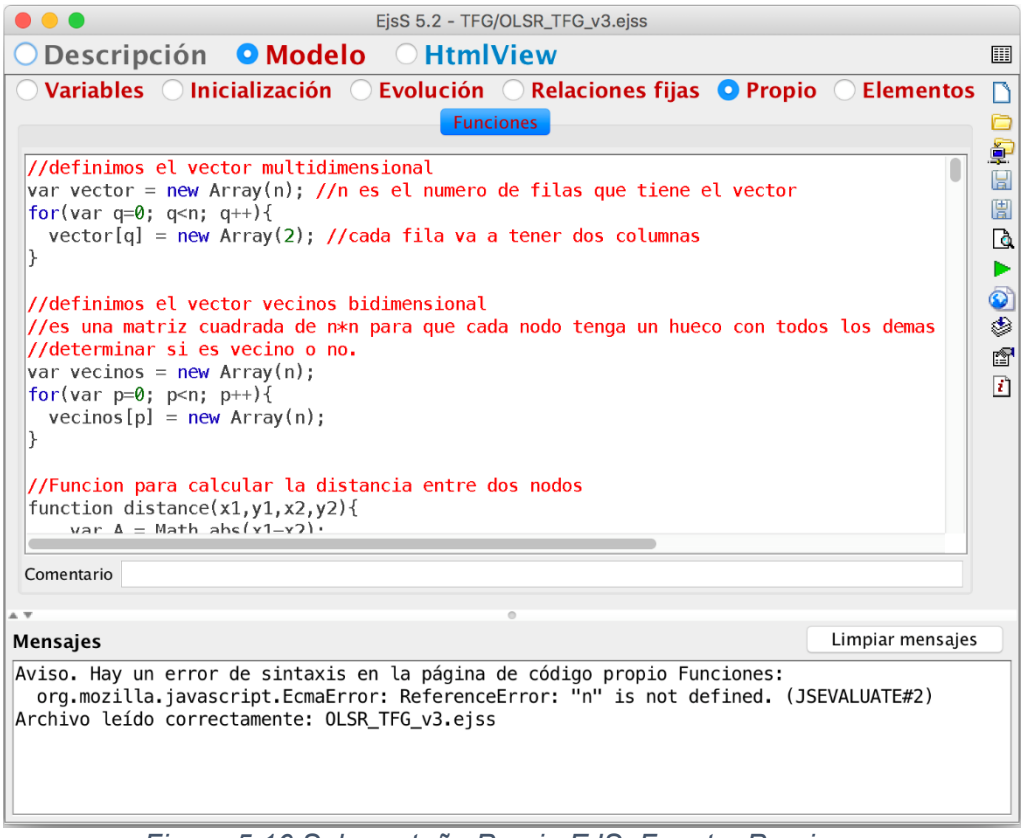


Figura 5.16 Sub-pestaña Propio EJS. Fuente: Propio

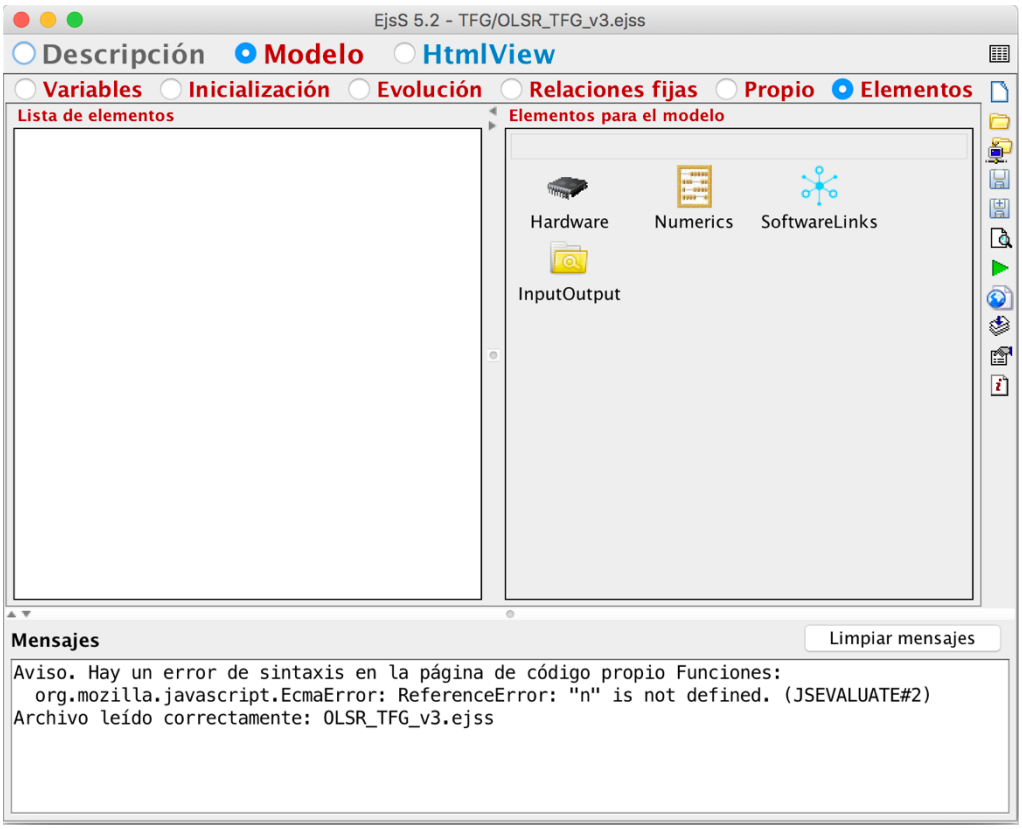


Figura 5.17 Sub-pestaña Elementos EJS. Fuente: Propia

HTMLVIEW

Esta pestaña contendrá los elementos visuales que conformarán la simulación, como se puede ver en la figura 5.14, en la parte de la derecha se verá un menú donde se puede seleccionar el tipo de objeto que se visualizará en la simulación (flechas, círculos, panel de control de botones, imágenes, etc.).

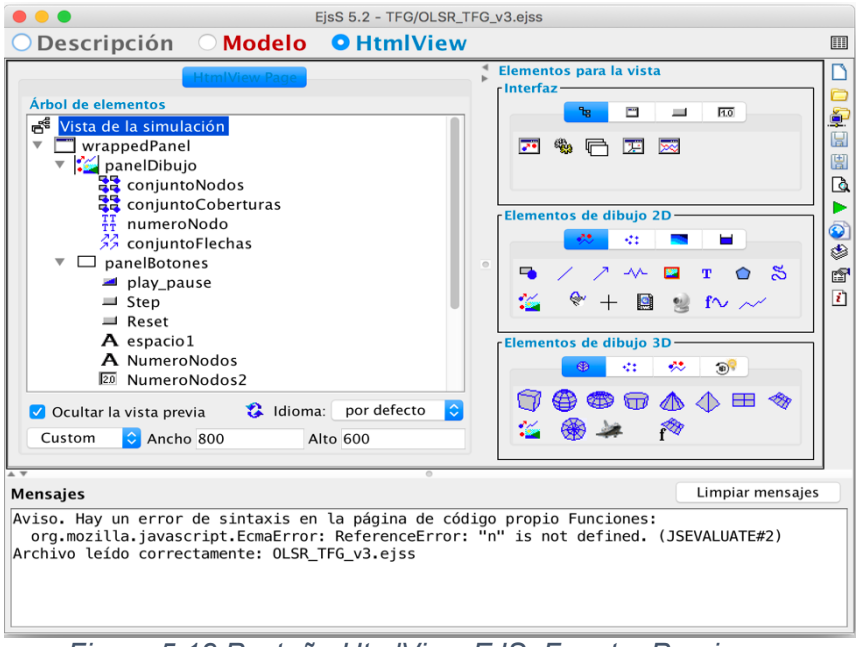


Figura 5.18 Pestaña HtmlView EJS. Fuente: Propia

5.3.2 *NetBeans*

NetBeans IDE es un entorno de desarrollo (gratis, libre y sin restricciones de uso) que sirve para compilar, escribir, ejecutar y depurar programas. Fue desarrollado en Java, pero admite cualquier otro lenguaje de programación, además cuenta con una gran cantidad de módulos (plugins) para su extensión [26] .

En la figura 5.19 se muestra el aspecto grafico de NetBeans.

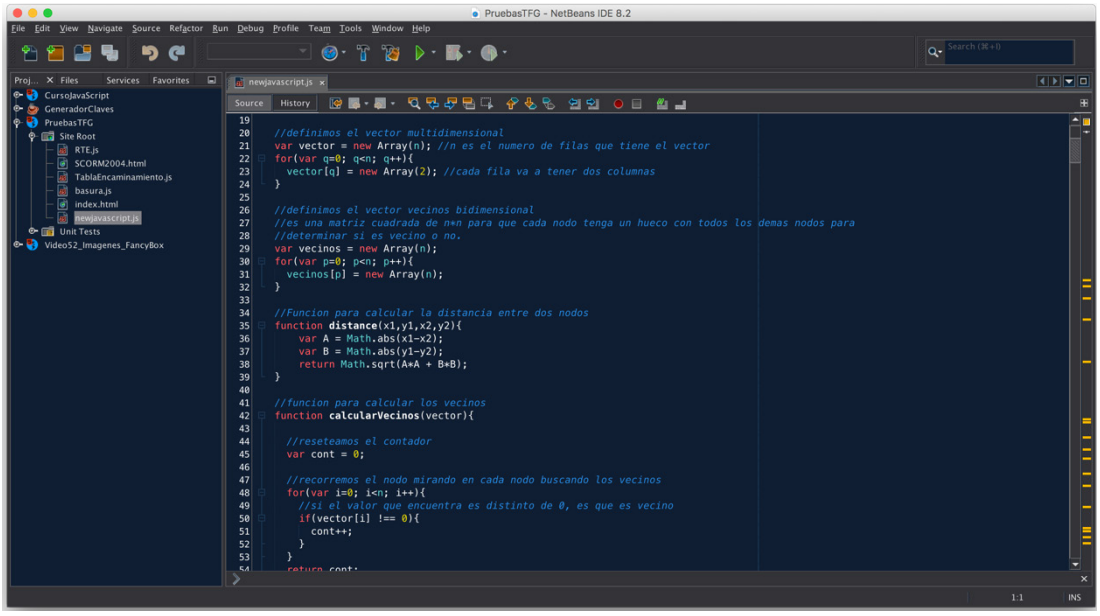


Figura 5.19 Interfaz gráfica NetBeans. Fuente: Propio

5.4 Shareable Content Objects (SCO)

Un SCO (*Shareable Content Objects*) es un conjunto de uno o más activos que representan un único recurso de aprendizaje [22]. Utiliza SCORM RTE para iniciarse y para comunicarse con un LMS. Un SCO representa el nivel más bajo de un recurso de aprendizaje que es rastreado por un LMS usando SCORM. La única diferencia entre un SCO y un activo es que el SCO se comunica con un LMS usando ECMAScript API. La figura 5.20 ilustra un ejemplo de una SCO formada por varios activos.

Un SCO podría reutilizarse en distintos entornos de aprendizaje, además, una actividad puede contener más de un SCO.

SCORM no impone ninguna restricción particular en tamaño exacto de un SCO por lo que suelen a ser unidades muy pequeñas y gracias a esto es posible la reutilización en varios contextos de aprendizaje. Los requerimientos de reutilización tendrán un impacto importante en las decisiones que definirán el tamaño del SCO.

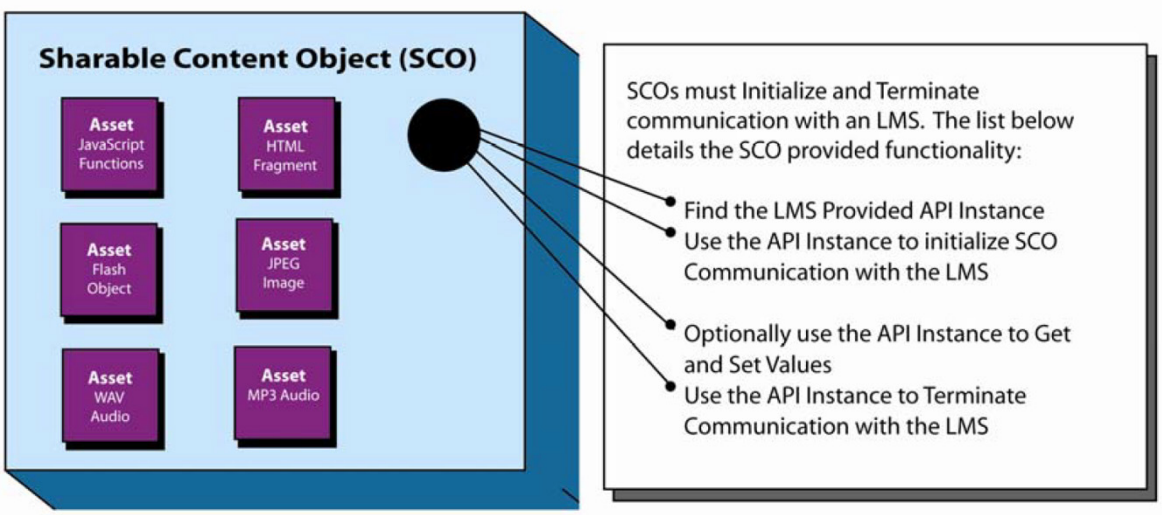


Figura 5.20 SCO formado por varios activos. Fuente: [22]

Se requiere que un SCO cumpla con los requisitos [27] . Esto implica que debe tener un medio para localizar una instancia de API proporcionada por LMS y debe invocar los métodos mínimos de API (Initialize ("") y Terminate ("")).

Los beneficios que produce que un SCO use SCORM RTE son los siguientes:

- Cualquier LMS que admita SCORM RTE puede iniciar un SCO y rastrearlos, independientemente de quién los generó.
- Cualquier LMS que admita SCORM RTE puede rastrear cualquier SCO y saber cuándo ha sido iniciado y cuando ha terminado.
- Cualquier LMS que sea compatible con SCORM RTE puede ejecutar cualquier SCO de la misma manera.

5.5 Comunicaciones entre SCO y LMS

La sub-especificación RTE (Run-Time Enviroment) [27] explica que para la comunicación entre un SCO (*Sharable Content Object*) y un LMS (*Learning Management System*) se utiliza un modelo de comunicación en forma de API denominado SCORM RTE. Esta API se encuentra en posesión del LMS de turno y consiste en una serie de funciones escritas en JavaScript para establecer la comunicación entre SCO y LMS, especificar los requerimientos para poder lanzar los objetos de contenido y para gestionar la información que puede ser intercambiada entre un SCO y el LMS.

SCORM tiene dos modelos de objetos de contenido [27] :

- **Assets o recursos:** No se pueden comunicar en tiempo de ejecución.
- **SCOs:** Se pueden comunicar en tiempo de ejecución.

El API estandarizada que regula el procedimiento de comunicación informa al LMS de la situación en la que se encuentra la comunicación entre este y el objeto de contenido correspondiente (terminada, iniciada, error) además, también se encarga del almacenamiento y recuperación de datos intercambiados entre SCO y LMS.

En la figura 5.21 se puede ver un ejemplo de comunicaciones al iniciar un SCO usando como LMS la plataforma Ilias de la Universidad de Jaén.

Figura 5.21 Comunicaciones entre SCO y LMS. Fuente: Ilias

5.5.1 *Run-Time Enviroment (RTE) Management*

A la vez el alumno está interactuando con los distintos SCOs, el LMS se encarga de evaluar la productividad del alumno y las peticiones de navegación [27].

El LMS busca e identifica alguna actividad para entregar al alumno, la cual contenga contenido asociado a él. El LMS lanzará el objeto de contenido y lo mostrará al aprendiz. La Figura 5.22 ilustra la forma de cómo la estructura de contenido del archivo imsmanifest.xml puede interpretarse con forma de árbol.

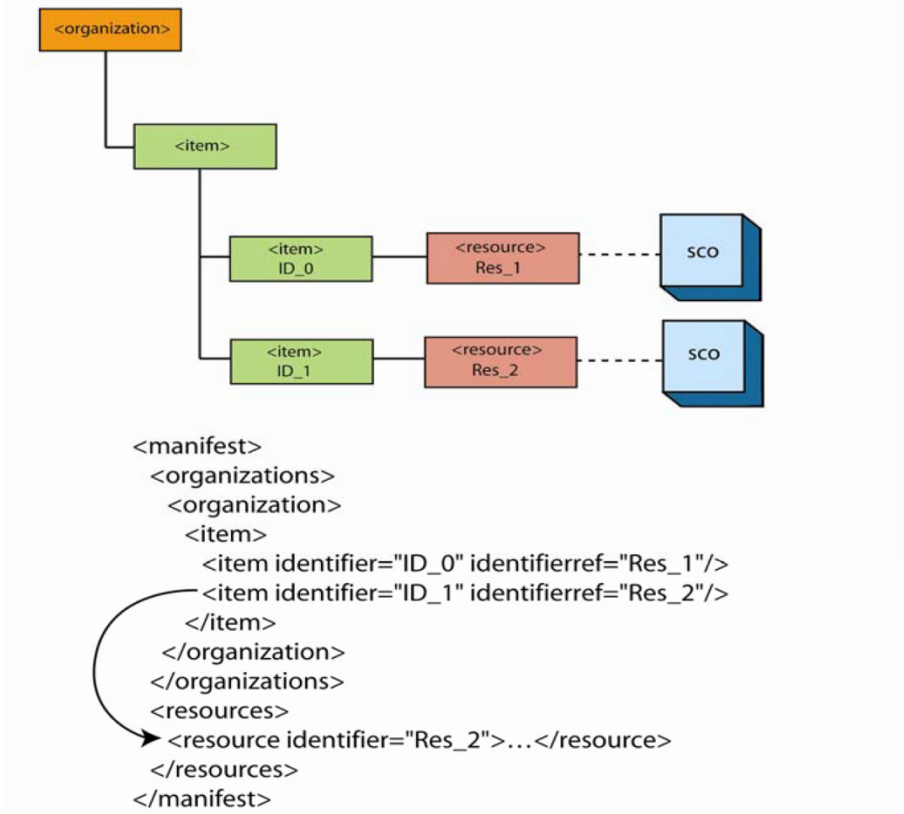


Figura 5.22 Contenido de archivo imsmanisfest.xml con forma de árbol. Fuente: [27]

Un modelo de lanzamiento común aborda la entrega de objetos de contenido habilitados para la web en forma de SCO y activos en el contexto de una experiencia de aprendizaje. Este modelo de lanzamiento permite la coherencia del comportamiento del lanzamiento del objeto de contenido en los LMS sin especificar la implementación del LMS subyacente. En este contexto, el término "LMS" se usa para describir cualquier sistema que proporcione el lanzamiento de objetos de contenido.

5.5.2 *Application Programming Interface (API)*

Las primeras versiones de SCORM, hasta e incluyendo la versión 1.2, se basaban en la funcionalidad del entorno en tiempo de ejecución. Desde entonces, el trabajo se enfocó a la estandarización de varias piezas de las Instrucciones de Interoperabilidad de la Instrucción Administrada por Computadora (CMI). SCORM describe el Estándar IEEE 1484.11.2 (denominado “API_1484_11” en SCORM 2004), se trata de un interfaz de programación de aplicaciones ECMAScript para comunicación de Content to Runtime Services como se usa en SCORM. El estándar describe la API para el contenido para la comunicación del servicio en tiempo de ejecución (RTS). Un RTS se define como el software que controla la ejecución y entrega de contenido de aprendizaje y que puede proporcionar servicios tales como asignación de recursos, programación, control de

entrada-salida y administración de datos. La API permite la comunicación de datos entre el contenido y un RTS normalmente proporcionado por un LMS a través de un conjunto común de servicios de API utilizando el lenguaje ECMAScript (comúnmente conocido como JavaScript).

El uso de una API común cumple muchos de los requisitos de alto nivel de SCORM para interoperabilidad y reutilización. Proporciona una forma estandarizada para que los SCO se comuniquen con los LMS, pero protege la implementación de comunicación particular del SCO desarrollador.

Hay varios términos que se utilizan en SCORM: API, implementación de API e instancia de API. La figura 5.23 describe los términos y sus relaciones entre sí.

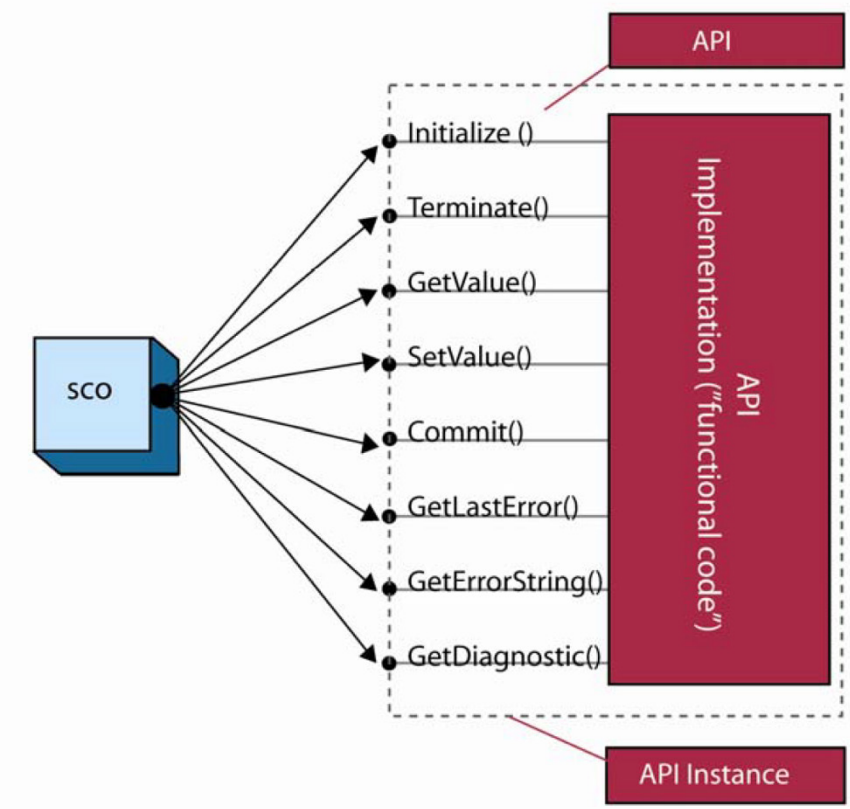


Figura 5.23 API, implementación de API e instancia de API. Fuente: [27]

API es simplemente un conjunto de funciones definidas en las que el SCO puede confiar para estar disponible.

Una implementación de API es una pieza de software funcional que implementa y expone las funciones de la API. Cómo las funciones de implementación de API no deberían importarle a un desarrollador de SCO, siempre que la implementación de API use la misma interfaz pública y se adhiera a la semántica de la interfaz. El LMS solo necesita proporcionar una implementación de API que implemente la funcionalidad de la API y exponga su interfaz pública al cliente SCO.

Una Instancia API es un contexto de ejecución individual y el estado de una Implementación API. La Instancia API representa la parte del software de ejecución con la que el SCO interactúa durante la operación de SCO.

La API SCORM cuenta con ocho métodos los cuales tienen una sintaxis diferente en función de la versión [27].

Versión 2004

- Initialize(“”) : bool
- Terminate(“”) : bool
- GetValue(element : CMIElement) : string
- SetValue(element : CMIElement, value : string) : string
- Commit(“”) : bool
- GetLastError() CMIErrorCode
- GetErrorString(errorCode : CMIErrorCode) : string
- GetDiagnosticString(errorCode : CMIErrorCode) : string

Versión 1.2

- LMSInitialize(“”) : bool
- LMSFinish(“”) : bool
- LMSGetValue(element : CMIElement) : string
- LMSSetValue(element : CMIElement, value : string) : string
- LMSCommit(“”) : bool
- LMSGetLastError() CMIErrorCode
- LMSGetErrorString(errorCode : CMIErrorCode) : string
- LMSGetDiagnosticString(errorCode : CMIErrorCode) : string

Aunque con el cambio de versión la sintaxis de los métodos sea distinta, la funcionalidad de los mismos es idéntica. A continuación, se verán más detenidamente cada uno de estos métodos [27].

Initialize / LMSInitialize

- **Sintáxis del método:** return_value = Initialize/LMSInitialize(parameter)
- **Descripción:** Función usada para iniciar la sesión de comunicación. Dota al LMS de poder para el manejo de problemas de inicialización propios del LMS.
- **Parámetro:** Se pasará como parámetro una cadena de caracteres vacía.
- **Valor de retorno:** “true” o “false”.

Terminate / LMSFinish

- **Sintáxis del método:** return_value = Terminate/LMSFinish(parameter)
- **Descripción:** La función se utiliza para finalizar la sesión de comunicación. Es utilizado por el SCO cuando el SCO ha determinado que ya no necesita comunicarse con el LMS.
- **Parámetro:** Se pasará como parámetro una cadena de caracteres vacía.
- **Valor de retorno:** “true” o “false”.

GetValue / LMSGetValue

- **Sintáxis del método:** return_value = GetValue/LMSGetValue(parameter)
- **Descripción:** Función que usa el SCO para pedir información de cualquier tipo al LMS.
- **Parámetro:** El parámetro representa la identificación completa de un modelo de datos del elemento.
- **Valor de retorno:** Por un lado, puede devolver una cadena de caracteres que contiene el valor del elemento solicitado. Por otro lado, Si se produce un error, la Instancia API establecerá un código de error a un valor específico al error y devolver una cadena de caracteres vacía.

SetValue / LMSSetValue

- **Sintáxis del método:** return_value = SetValue/LMSSetValue(parameter_1, parameter_2)
- **Descripción:** Método que se usa para asignar el valor contenido en “parameter_1” al elemento de datos especificado en “parameter_2”.
- **Parámetros:** “parameter_1” contiene la identificación completa de un elemento de modelo de datos para establecer. “parameter_2” contiene el valor al que se debe configurar el contenido de “parameter_1”.
- **Valor de retorno:** “true” o “false”.

Commit / LMSCommit

- **Sintáxis del método:** return_value = Commit/LMSCommit(parameter)
- **Descripción:** El método solicita el reenvío a los datos persistentes del SCO que la instancia de la API puede haber almacenado en caché desde la última llamada a Initialize("") o Commit(""), lo que ocurra más recientemente.
- **Parámetro:** Se pasará como parámetro una cadena de caracteres vacía.
- **Valor de retorno:** “true” o “false”.

GetLastError / LMSGetLastError

- **Sintáxis del método:** return_value = GetLastError/LMSGetLastError()
- **Descripción:** Este método solicita el código de error para el estado de error actual de la instancia de la API. Si un SCO llama a este método, la Instancia API no alterará el estado de error actual, sino que simplemente devolverá la información solicitada.
- **Parámetro:** Este método no acepta parámetros.
- **Valor de retorno:** El valor de retorno será una cadena de caracteres (convertible a un entero en el rango de 0 a 65536 inclusive) que representa el código de error del último error encontrado.

GetErrorString / LMSGetErrorString

- **Sintáxis del método:** return_value= GetErrorString/LMSGetErrorString(parameter)
- **Descripción:** La función GetErrorString() se puede utilizar para recuperar una descripción textual del estado de error actual. La función es utilizada por un SCO para solicitar la descripción textual del código de error especificado por el valor del parámetro.
- **Parámetro:** Representa la cadena de caracteres del código de error (valor entero) correspondiente a un mensaje de error.
- **Valor de retorno:** El método devolverá un mensaje textual que contiene una descripción del código de error especificado por el valor del parámetro.

GetDiagnostic / LMSGetDiagnostic

- **Sintáxis del método:** return_value= GetDiagnostic/LMSGetDiagnostic(parameter)
- **Descripción:** La función GetDiagnostic() permite al LMS definir información de diagnóstico adicional a través de la Instancia API. Esta llamada no tiene efecto en el estado de error actual; simplemente devuelve la información solicitada.
- **Parámetro:** El valor del parámetro puede ser un código de error, pero no está limitado solo a códigos de error. La longitud máxima del valor del parámetro será de 255 caracteres.
- **Valor de retorno:** La instancia de API devolverá una cadena de caracteres que represente la información de diagnóstico. La longitud máxima de la cadena de caracteres devuelta será de 255. Si el parámetro es desconocido por el LMS, se devolverá una cadena de caracteres vacía. Si el parámetro pasado es una cadena de caracteres vacía, se recomienda que la función devuelva una cadena de

caracteres que representa la información de diagnóstico sobre el último error encontrado.

5.6 Navegadores Web

Para la evaluación y corrección de errores de las páginas web creadas se han utilizado tres navegadores distintos (los tres más usados) con el fin de comprobar que para cada uno de ellos todo funciona perfectamente y la visualización es óptima y correcta. Los tres navegadores que se han usado para esto han sido: Safari, Google Chrome y Mozilla Firefox.

5.6.1 *Safari*

Uno de los navegadores más usados en el momento y más conocido. La versión utilizada de este navegador es la Versión 11.0.2. Para acceder a la consola de este navegador se deben pulsar las teclas ⌥+⌘+C (Option+command+C). La figura 5.24 muestra el aspecto de la consola.

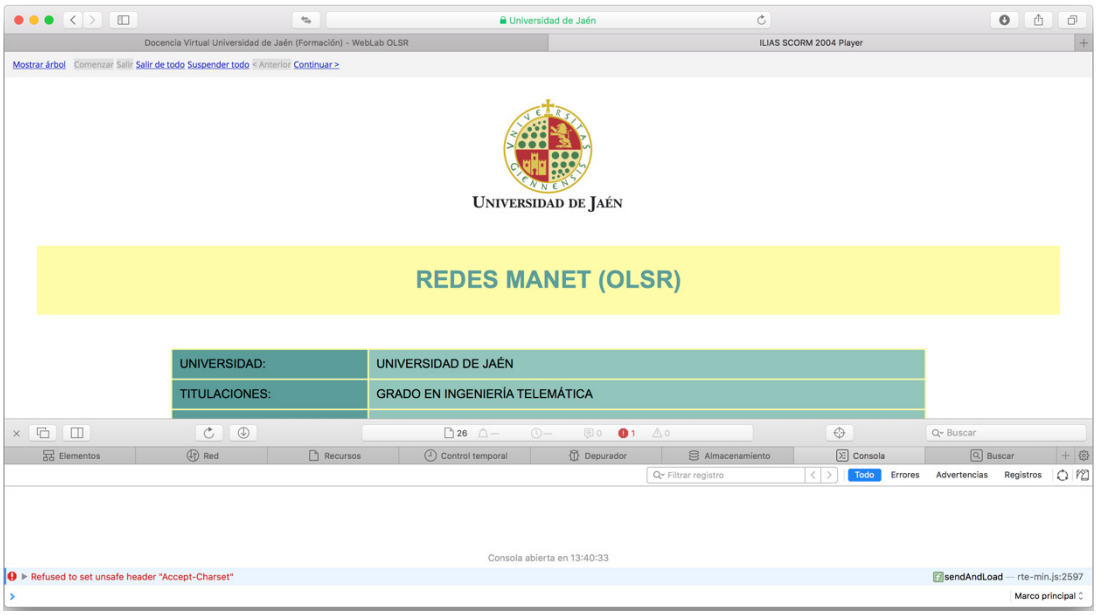


Figura 5.24 Consola Safari. Fuente: Propia

Se mostrará directamente la consola de JavaScript del navegador, pero cambiando de pestañas se puede acceder a todos los aspectos de la web como los elementos HTML, los estilos CSS, etc.

5.6.2 *Google Chrome*

Navegador con el que se ha trabajado en primera instancia por elección personal en cuestión de gustos. Los demás navegadores usados son perfectamente compatibles con el trabajo desarrollado.

Uno de los navegadores por excelencia ya que se trata de un navegador desarrollado por Google que cuenta con más de 750 millones de usuarios en el año 2013 [28]. Se ha utilizado en su versión 63.0.3239.108.

Se ha utilizado la herramienta para desarrolladores que incorpora este navegador siendo accesible pulsando las teclas ⌥+⌘+I (macOS) o la tecla F12 (Windows).

La figura 5.25 ilustra la herramienta para desarrolladores que incorpora este navegador.

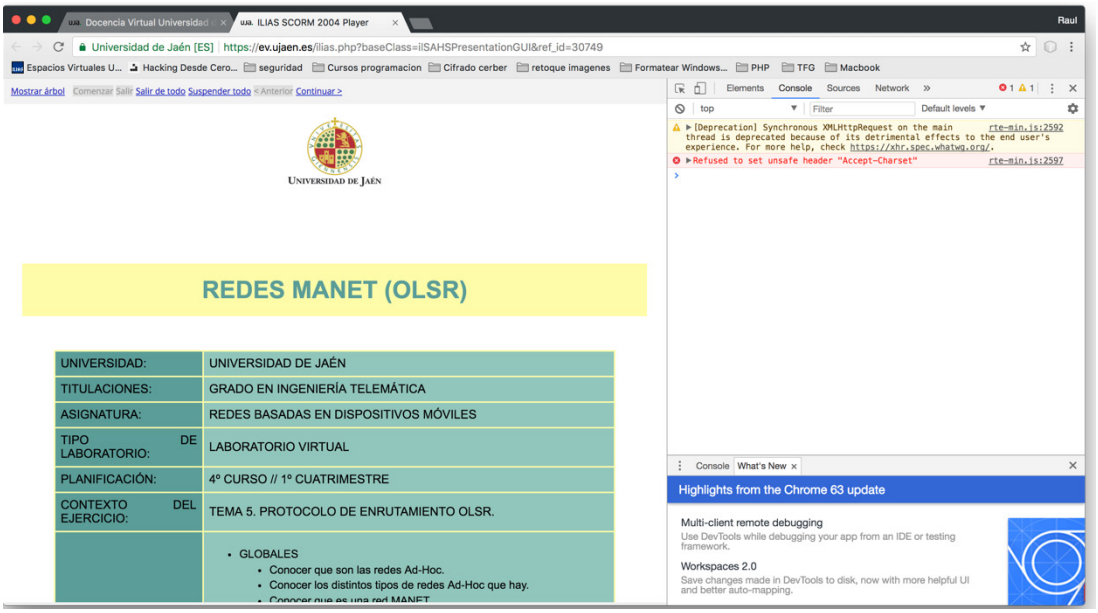


Figura 5.25 Consola navegador Google Chrome. Fuente: Propia

Al igual ocurre con Safari, se muestra en primer lugar la consola de JavaScript, pero navegando entre las distintas pestañas se puede acceder a distintos aspectos de la web como los elementos HTML, los estilos CSS, etc.

5.6.3 *Mozilla Firefox*

Otro de los navegadores más usados con 350 millones de usuarios alcanzados en el año 2010 [29]. Este navegador cuenta con una herramienta muy útil a la hora del desarrollo web, esta herramienta el Firebug, el inconveniente que se ha encontrado es que en la reciente actualización de este navegador (Firefox Quantum versión 57.0), esta herramienta pasado a ser incompatible, por lo que se ha tenido que usar la herramienta para desarrolladores que incorpora este navegador.

Para mostrar la herramienta para desarrolladores basta con pulsar las teclas ⌥+⌘+K (macOs) o las teclas Ctrl+Shift+K (Windows).

La figura 5.26 muestra el aspecto de esta herramienta.

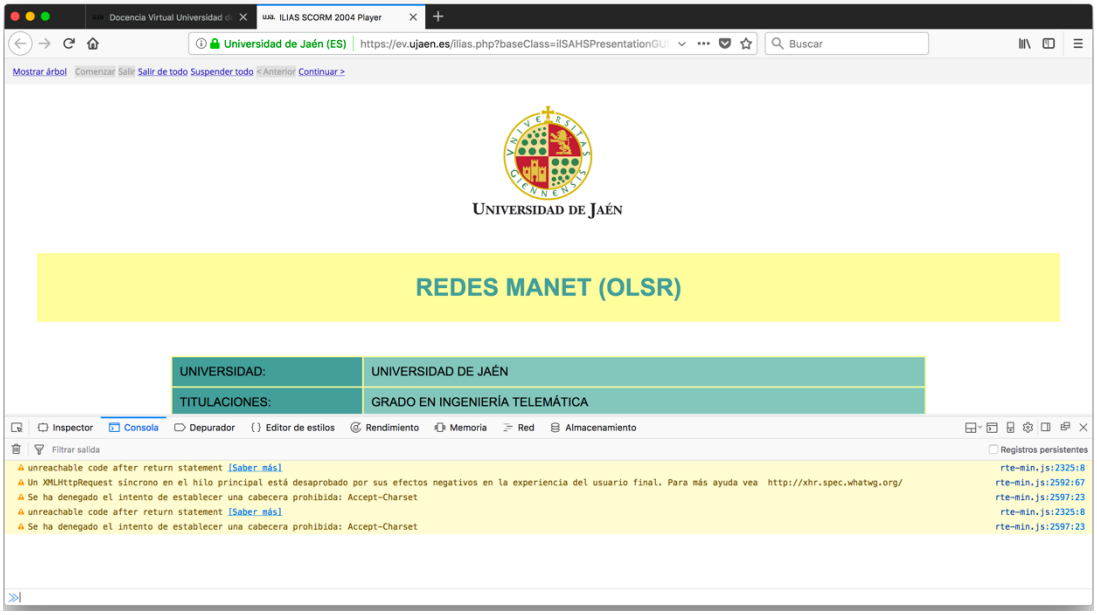


Figura 5.26 Consola navegador Mozilla Firefox. Fuente: Propia

Al igual ocurre con Safari y Google Chrome, se muestra en primer lugar la consola de JavaScript, pero navegando entre las distintas pestañas se puede acceder a distintos aspectos de la web como los elementos HTML, los estilos CSS, etc.

5.7 Interacciones

El modelo de datos de interacciones determina un acopio de respuestas del alumno que puede pasar del SCO al LMS [27]. Las interacciones pretenden ser respuestas a preguntas individuales o actividades que el diseñador de SCO desea registrar. No hay un comportamiento predeterminado de un LMS cuando se solicita que se establezcan interacciones, salvo el almacenamiento de los datos.

Las interacciones pueden valorarse como una selección de información (datos de interacción).

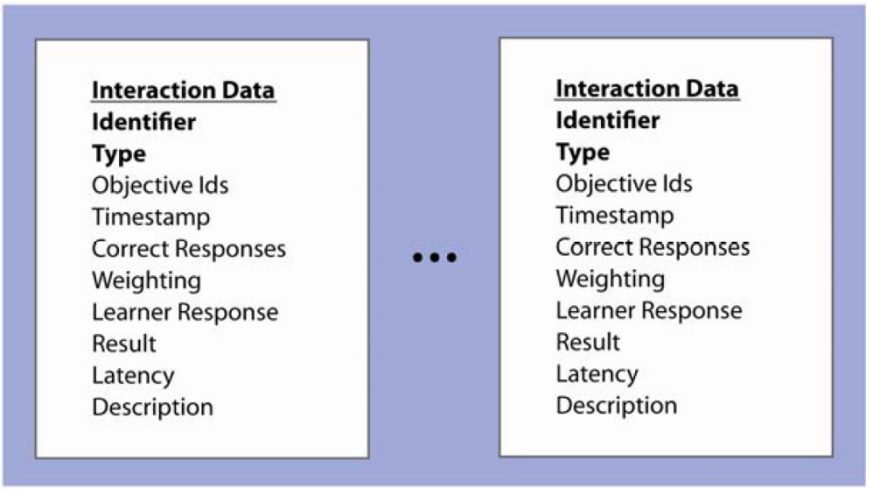


Figura 5.27 Interacción de datos. Fuente: [27]

Hay dos datos importantes que son precisados en los datos de interacción, un identificador (cmi.interactions.n.id) y el tipo de interacción (cmi.interactions.n.type). El elemento de modelo de datos cmi.interactions.n.type es de carácter obligatorio al incluir la interacción datos vinculados con la respuesta escogida por el alumno o la respuesta válida de la pregunta. Estos dos elementos de información son las que diferencian los datos de interacción de otros datos encontrados en el conjunto de interacciones. El identificador cumple la función de identificar de manera unívoca una interacción. El tipo identifica de manera unívoca el tipo de interacción (verdadero-falso, coincidencia, etc.).

Este elemento se puede utilizar mediante dos medios primarios por el SCO: diario y estado.

- **Esquema diario:** precisa que el SCO almacene los datos de interacción cada vez que el alumno se implique con la interacción (se añaden nuevos datos de interacción al conjunto de interacciones). Al aplicar este esquema para registrar interacciones, se puede coleccionar información para analizar la destreza del alumno con las interacciones halladas en el SCO.
- **Esquema de estado:** precisa que el SCO registre datos de interacción y sostenga actualizada la interacción a partir de la destreza del alumno con el SCO.

Existen ventajas y desventajas a la hora utilizar un esquema particular sobre otro. Desde el punto de vista de un LMS, el esquema diario precisa más capacidad sobre los requisitos de almacenamiento. Pero, los desarrolladores de SCO deben comprender que se requiere un LMS para admitir (es decir, almacenar) al menos 250 conjuntos de datos de

interacción. Un LMS puede optar por proporcionar soporte para más de 250; sin embargo, el requisito mínimo es admitir al menos 250 conjuntos de datos de interacción [27].

5.8 Librerías JavaScript para crear test online

Para dotar a una web de la funcionalidad de poder crear un test de preguntas online se ha utilizado una librería denominada *principal.js* cuyo autor de la misma es *Sergio Díaz Fuentes.*

Para poder usar esta librería es necesario añadirla al archivo HTML que se quiere dotar de esta funcionalidad. La importación se realizará en la cabecera del documento HTML, concretamente, dentro de las etiquetas de marca *<head></head>.* Un ejemplo de cómo se ha importado esta librería para su uso en este proyecto se puede ver en la figura 5.28.

```
<!DOCTYPE html>

<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <script type="text/javascript" src="_ejs_library/scripts/principal.js"></script>
        <script type="text/javascript" src="_ejs_library/scripts/APIWrapper.js"></script>
        <script type="text/javascript" src="_ejs_library/scripts/RTE.js"></script>
        <script type="text/javascript" src="_ejs_library/scripts/smoke.js"></script>
        <link href="_ejs_library/css/estilo.css" rel="stylesheet" type="text/css">
        <link href="_ejs_library/css/smoke.css" rel="stylesheet" type="text/css">
```

Figura 5.28 Importación de librería principal.js en archivo HTML. Fuente: Propia

Como se puede observar en la figura 5.29, en cada archivo HTML ha sido necesaria la creación de seis variables distintas en código JavaScript para la modificación de distintos aspectos del test que se creará. Estas seis variables son [21]:

- ***banco:*** con esta variable lo que se hace es referenciar el banco de preguntas que contendrá las cuestiones que se mostrarán al alumno cuando realice el test. Para seleccionar el banco de preguntas que se quiere usar se debe escribir la ruta del mismo usando una cadena de caracteres.
- ***numeroPr:*** Indica el número de preguntas que contendrá el test, este número de preguntas se le indica escribiendo un número entero.
- ***time:*** Esta variable indica el tiempo del que dispone el alumno para la realización del test en su totalidad, se indicará el tiempo escribiendo un número entero el cual será el número de minutos de los que dispondrá el alumno. También se contempla la posibilidad de no establecer un límite de tiempo, por lo que si se desea esta opción basta con escribir el número 0.
- ***prAleat:*** Esta variable da la posibilidad de que las preguntas se muestren de una manera aleatoria. Para activar o desactivar esta opción deberá escribir una cadena

de caracteres con el texto “si” si se pretende que las preguntas se muestren de forma aleatoria o, por el contrario, el texto “no” si no se quiere aleatoriedad en las preguntas.

- ***opAletal:*** Para el caso de las preguntas que sean del tipo “*opción múltiple-respuesta única”* u *“opción múltiple-respuesta múltiple”*, esta variable dará la posibilidad de mostrar de forma aleatoria las distintas opciones de respuesta que dispongan esa pregunta.
- ***comunicaciones:*** Esta variable capacita al creador del SCO, si quiere que este establezca comunicaciones con el LMS de turno o por el contrario no lo desea. Para activar o desactivar esta opción deberá de escribir una cadena de caracteres con el texto “si” si se quiere que haya comunicación o, por el contrario, el texto “no” si no se quiere.

```
<!DOCTYPE html>

<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <script type="text/javascript" src="_ejs_library/scripts/principal.js"></script>
        <script type="text/javascript" src="_ejs_library/scripts/APIWrapper.js"></script>
        <script type="text/javascript" src="_ejs_library/scripts/RTE.js"></script>
        <script type="text/javascript" src="_ejs_library/scripts/smoke.js"></script>
        <link href="_ejs_library/css/estilo.css" rel="stylesheet" type="text/css">
        <link href="_ejs_library/css/smoke.css" rel="stylesheet" type="text/css">


        <script type="text/javascript">


            var miRTE = new RTE("2004");
            var terminated = false;
            var banco = '../../_ejs_library/html/Preguntas OLSR.xml';
            var numeroPr = 5;
            var time = 0;
            var prAleat = 'si';
            var opAleat = 'si';
            var comunicaciones = 'no';
```

Figura 5.29 Variables para crear test online. Fuente: Propia

Una vez realizados los pasos de importar la librería *principal.js* y de declarar las seis variables en JavaScript (todo esto en la cabecera del documento) se puede invocar a la función *loadJsLib* definida en la librería *principal.js.*

La invocación de dicha función tiene que ser dentro de las etiquetas *<body></body>* del documento HTML. En primer lugar, se deberá que crear una tabla en HTML y después, dentro de esa tabla abrirá unas etiquetas *<script></script>* dentro de las cuales ira la llamada a esa función. En la figura 5.30 se puede ver cómo realizar la llamada de esta función.

```
151
152
153     <body>
154         <!-- Para insertar el test tenemos que crear una tabla y despues en una etiqueta script insertamos la funcion
155              loadJsLib() -->
156
            <table id="tabla" align="center"> <tr>
158             <script type="text/javascript">
159                 loadJsLib(banco, numeroPr, time, prAleat, opAleat, comunicaciones);
                </script>
161         </tr> </table>
162
163     </body>
164
```

Figura 5.30 Llamada a la función loadJsLib. Fuente: Propia

Una vez realizados estos pasos, la función *LoadJsLib* es la encargada de invocar a una librería u otra atendiendo al valor que haya sido establecido en la variable *comunicaciones* definida previamente.

A modo de resumen, la figura 5.31 muestra un diagrama de flujo con todos los pasos explicados.

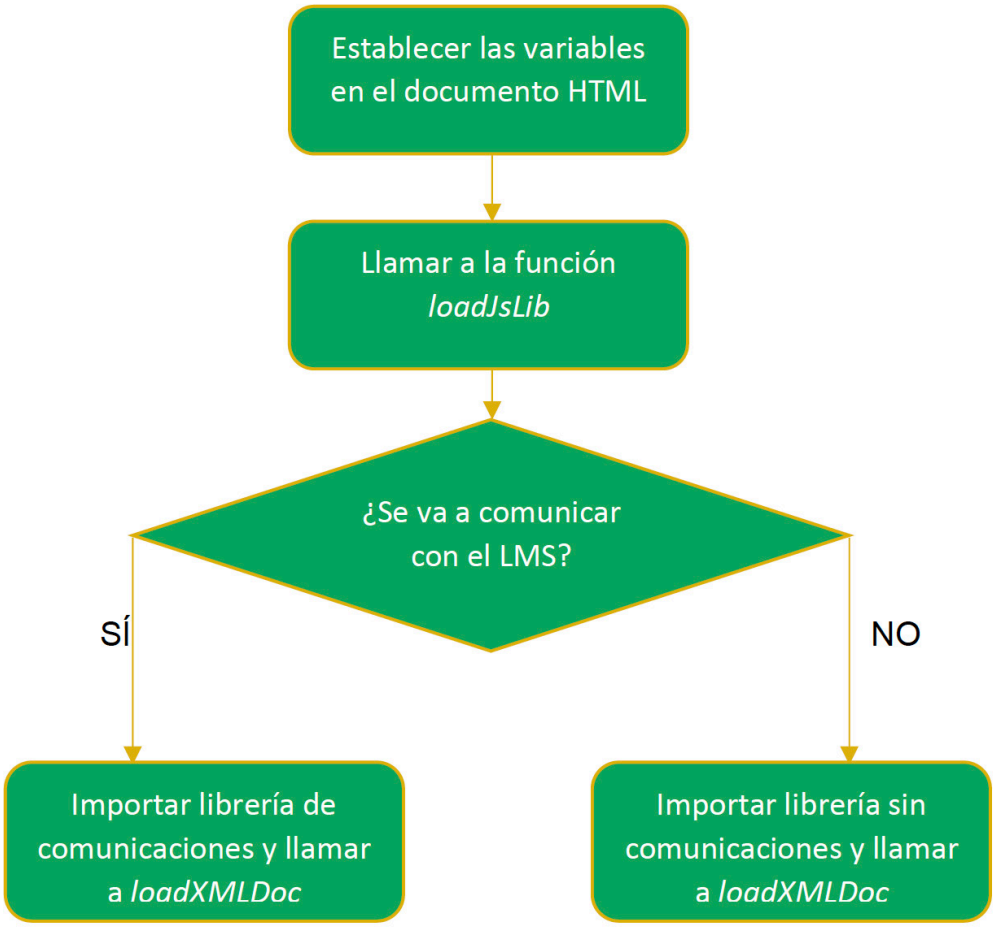


Figura 5.31 Pasos para crear un test en un documento HTML. Fuente: [21]

5.9 Interacción del alumno con el LMS

Se han usado tres librerías cuyo autor es *Sergio Díaz Fuentes* denominadas:

- creaPreguntas2004.js
- creaPreguntas12.js

Además de estas librerías se han usado otra para insertar la interactividad adicional a los test, por un lado, se ha usado la librería JavaScript APIWrapper.js cuyo autor es la ADL. Por otro lado, se ha usado la librería JavaScript RTE.js cuyo autor el *Ildefonso Ruano Ruano.* En la figura 5.32 se puede ver una captura parcial de esta librería.

```
30  }
31  var RTE = function (ver) {
32      var startDate = new Date();
33      this.startMS = startDate.getTime();
34      this.verScorm = ver; //SCORM VERSION: 2004 or 1.2
35      /* ************************** Read Only elements (constant in session)*/
36      this.learnerName; //Learner name in the LMS
37      this.learnerId; //Learner identification in the LMS
38      this.completionThreshold;
39      this.credit;
40      this.entry; //contains information that asserts whether the learner has previously accessed the SCO: "ab-initio", "resume", ""
41      this.launchData;
42      this.maxTimeAllowed;
43      this.timeLimitAction;
44      this.mode;
45      this.totalTime;
46      this.scoreChildren;
47      this.interactionsChildren;
48      this.learnerPreferenceChildren;
49      this.objectivesChildren;
50      this.startTimeStamp = timeStamp(this.verScorm);
```

Figura 5.32 captura parcial de librería RTE.js. Fuente: Propia

Estas librerías contienes muchas funciones de entre las cuales se encargar de rellenar los huecos previamente creados para el test. De forma que al iniciar el test se verán los siguientes campos:

- Preguntas del test. Se puede ver un ejemplo en la figura 5.33
- Botones de acción: Validar, Borrar, Posponer. Se puede verlos en la figura 5.34
- Puntuación del alumno. Se puede verlo en la figura 5.35

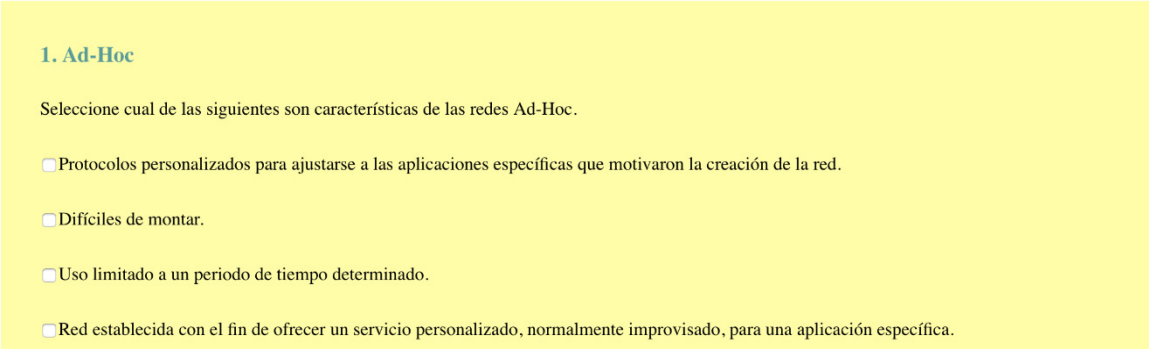


Figura 5.33 Pregunta de test mostrada en laboratorio virtual. Fuente: Propia

Figura 5.34 Botones de acción en test. Fuente: Propia

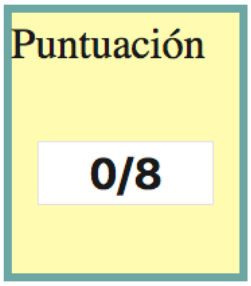


Figura 5.35 Puntuación de alumno. Fuente: Propia

Al responder una pregunta y pulsar el botón de validar, se le muestra al alumno un mensaje en que se le felicita si ha acertado la pregunta y se le anima si ha fallado. Este mensaje es un alert pero con estilo modificado por la librería usada denominada smoke.js [30].

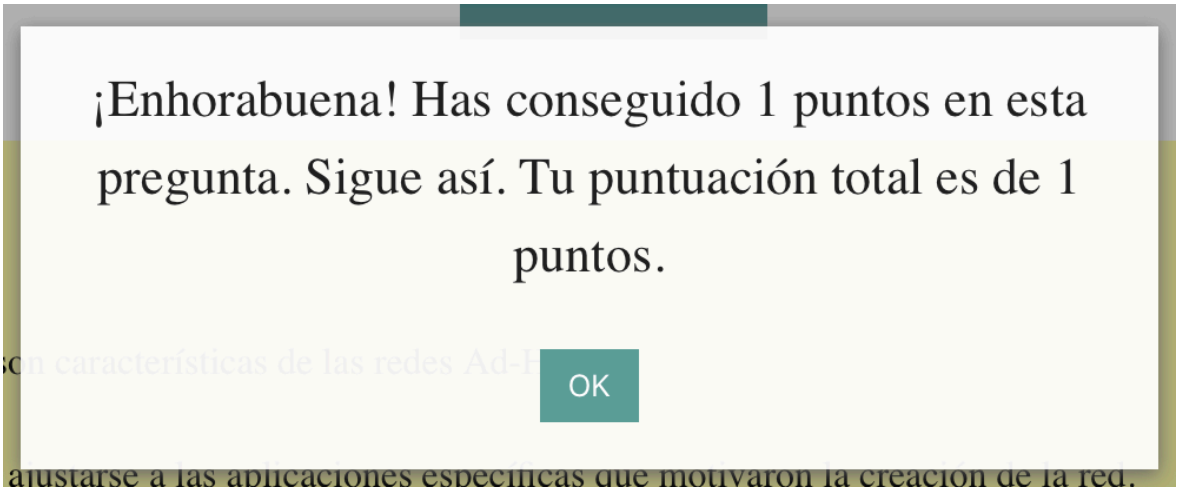


Figura 5.36 Alert smoke respuesta correcta. Fuente: Propia

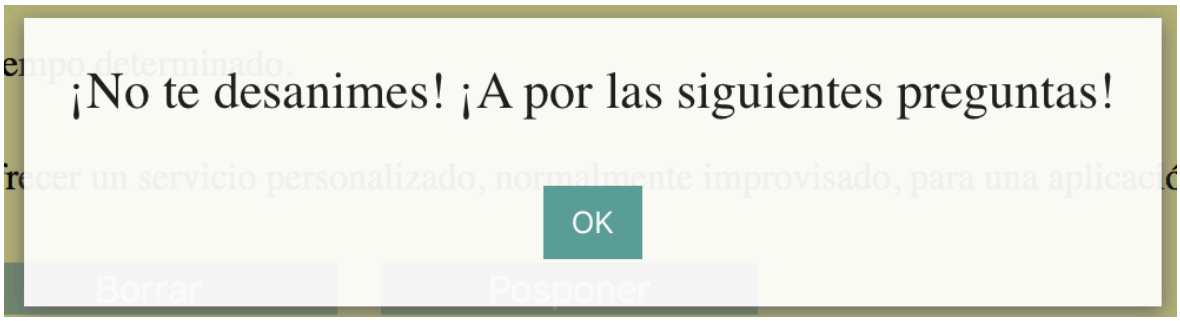


Figura 5.37 Alert smoke respuesta incorrecta. Fuente: Propia

creaPreguntas2004.js

Tras obtener las variables necesarias, esta librería invoca a las funciones *setInteractionsAddNewI* y *setInteractionsDescription* definidas en *RTE.js* (previamente se ha tenido que definir una instancia de *RTE.js*).

Para funcionar correctamente la función *setInteractionsAddNewI* debe recibir cinco parámetros que son: interacción, identificador de interacción, tipo de pregunta, peso de pregunta y respuesta válida de la pregunta.

Para obtener la repuesta válida de la pregunta hay definida una función denominada *getCorrectaInv,* esta función tiene que recibir como parámetros el tipo de pregunta y el identificador de la misma.

Para que no haya errores a la hora de que el LMS guarde la repuesta válida, se obtendrá esta repuesta en el formato que acepte el LMS. Esto ocurre igual con las respuestas marcadas por los alumnos, las cuales deberán ser pasadas también a este formato y posteriormente será comparada con la respuesta válida para ver si el alumno ha respondido de manera acertada o no.

Por otro lado, la función *setInteractionsDescription* se encarga de insertar una descripción a la interacción en la que se encuentra el programa. Esta función deberá de recibir como parámetros el número de interacción y la descripción que se quiere insertar en esa interacción.

Se puede ver estas dos funciones en uso en la figura 5.38.

```
if (contador ==0){
        patt = getCorrectaInv(id,tipo);
        miRTE.setInteractionsAddNewI(contador,id,"fill-in",1,patt.toLowerCase());
        miRTE.setInteractionsDescription(contador, titulo);
}
```

Figura 5.38 Funciones setInteractionsAddNewI y setInteractionsDesription. Fuente: Propia

Para llevar a cabo la evaluación de las preguntas se han seguido los siguientes procesos:

- En primer lugar, se obtendrá la respuesta válida usando para ello la función *getInteractionsCorrectRespPattern.*
- En segundo lugar, compara la respuesta elegida por el alumno con la respuesta obtenida anteriormente (que es la correcta). Si las dos coinciden la respuesta se dará por correcta y si no será incorrecta.
- En tercer lugar, ya que se sabe si el alumno ha fallado o ha acertado, se invoca la función *setInteractionsResponseAll,* esta función recibirá como parámetros la interacción en la que se encuentra, la respuesta elegida por el alumno y la corrección, de esta manera, en el LMS se guardará esta interacción.

creaPreguntas12.js

De la misma forma que ocurre con la librería de la versión 2004, tras obtener las variables necesarias, se invocan las funciones *setInteractionsId, setInteractionsType, setInteractionsTs, setInteractionsWeight* y *setInteractionsCorrectRespPattern* las cuales están definidas en el archivo *RTE.js* (previamente se ha tenido que definir una instancia de *RTE.js*).

El uso de todas estas funciones de manera separada es equivalente al uso de la función *setInteractionsAddNewI* perteneciente a la versión 2004. En la versión 1.2 se usan estas funciones por separada ya que la función que se usa en la versión 2004 no es soportada por la versión 1.2. Los parámetros necesarios que deben recibir estas funciones son los siguientes:

- ***setInteractionsId:*** Recibe como parámetros la interacción en la que se encuentra(número) y su identificador.
- ***setInteractionsType:*** Recibe como parámetro el tipo de pregunta (interacción) y el número de la interacción en la que se encuentra (número).
- ***setInteractionsTs:*** Recibe como parámetros la interacción en la que se encuentra (número) y el tiempo de interacción.
- ***setInteractionsWeight:*** Recibe la interacción en la que se encuentra (número) y el peso de la misma.
- ***setInteractionsCorrectRespPattern:*** Recibe como parámetros: interacción (número), “0” y la solución correcta de la pregunta.

En la figura 5.39 se puede ver estas funciones en uso.

```
232             }
233
                if (contador ==0){
235                     patt = getCorrectaInv(id,tipo);
                        miRTE.setInteractionsId(contador,id);
                        miRTE.setInteractionsType(contador,"fill-in");
                        miRTE.setInteractionsTs(contador, timeStamp("1.2"));
                        miRTE.setInteractionsWeight(contador, 1);
                        miRTE.setInteractionsCorrectRespPattern(contador, 0, patt.toLowerCase());
241             }
242
243                     flag = 1; //Este flag nos permite que no se repita la pregunta de nuevo en el div_preguntas
244                     flagE = 1;
245
246             }
```

Figura 5.39 Funciones librería creaPreguntas12.js en uso. Fuente: Propia

El procedimiento realizado para conocer la respuesta válida es idéntico que en la versión 2004, invocando a la función getCorrectaInv y tras esto el realizar el proceso de formateo para que el LMS sea capaz de entenderlas. De la misma forma, se ha usado esta misma función para la evaluación de la pregunta.

Tras haber realizado la evalucion de la pregunta, se invoca a la función *setInteractionsResponseAll* (igual que en versión 2004) para guardar esta interacción.

5.10 WebLabs SCORM desarrollados

En este punto se van a describir los dos laboratorios virtuales SCORM que se han implementado. Hay que destacar que estos dos módulos han sido desarrollados con la versión 2004 de SCORM ya que se trata de la versión más actualizada.

Para que sea posible la portabilidad de estos módulos para usar en otros LMS, estos han de ser importados en un fichero de tipo ZIP, por tanto, se han anexado junto con la memoria de este TFG los dos módulos desarrollados en formato ZIP.

El sistema de gestión de aprendizaje que se ha usado para la evaluación y comprobación de estos módulos ha sido la web de espacios virtuales perteneciente a la universidad de Jaén https://ev.ujaen.es/. Se ha creado un espacio específico para este fin donde se ha ido incluyendo todo lo necesario para la realización de estos módulos.

Hacer referencias a ubicación de docencia virtual pública para verlo.

En la figura 5.40 y 5.41 se puede ver la web y el espacio virtual creado respectivamente.

Figura 5.40 Acceso a Espacios Virtuales UJA. Fuente: Propia

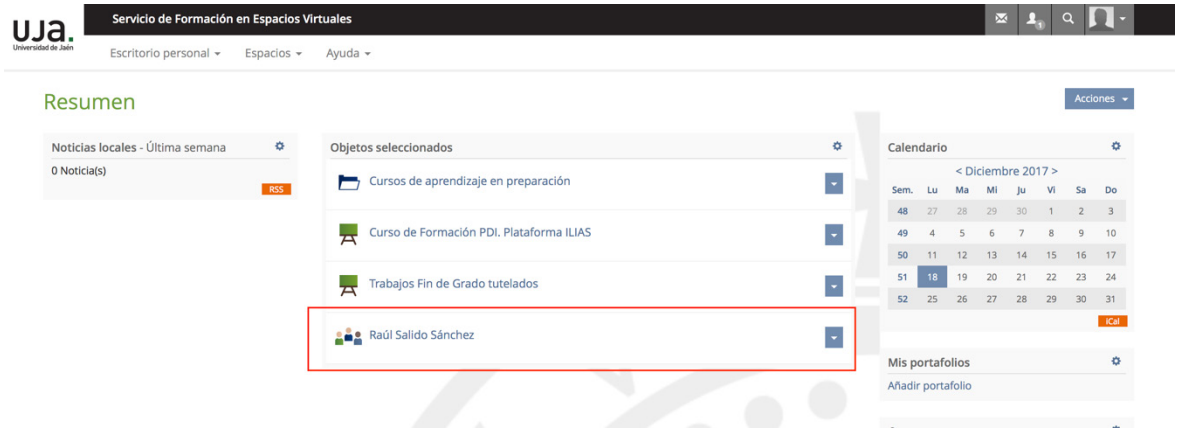


Figura 5.41 Espacio Virtual creado para TFG. Fuente: Propia

5.10.1 *WebLab SCORM VANET*

En este WebLab se puede distinguir entre 5 SCOs (páginas) cuyo objetivo es dar una pequeña introducción a las redes Ad-Hoc y dentro de estas a las VANET (*Vehicual Ad-Hoc Network*).

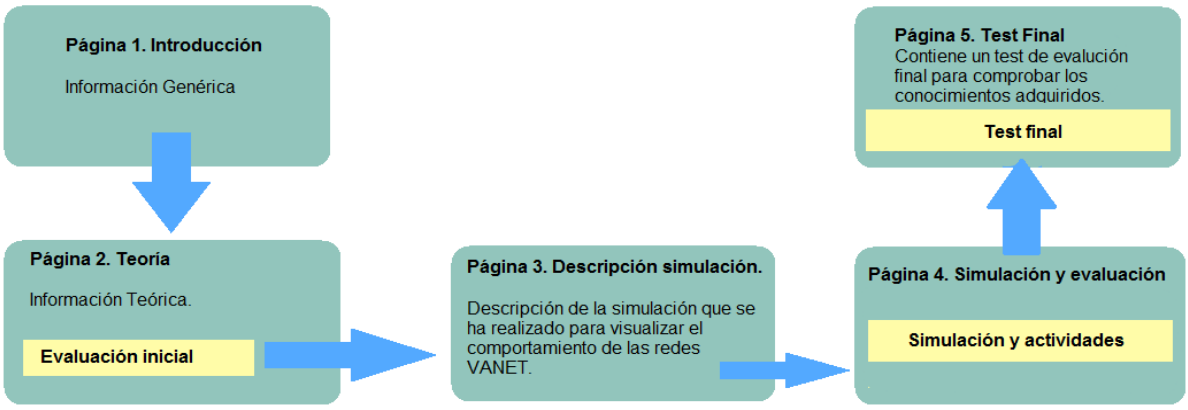


Figura 5.42 Estructura WebLab VANET. Fuente: Propia.

En la primera página, nada más iniciar el WebLab se muestra un mensaje de alerta modificado gráficamente gracias a la librería *smoke.js.* Este mensaje puede ser de dos tipos:

1. Si se trata de la primera vez que se accede a este WebLab, el mensaje que se mostrará será de bienvenida. Este mensaje es personalizado en el que aparece el nombre del usuario. Se puede ver un ejemplo de este mensaje en la figura 5.43.
2. Si no es la primera vez que se accede al WebLab se mostrará un mensaje en el cual se le dirá al usuario el número de veces que ha accedido a este SCO. Además, muestra la nota de evaluación que se ha obtenido en el último intento y realiza la pregunta de si se quiere seguir en este SCO o se pretende pasar al siguiente. Se puede ver un ejemplo en la figura 5.44.

Este primer SCO indica información relativa al WebLab como: objetivos del WebLab, estructura, etc.

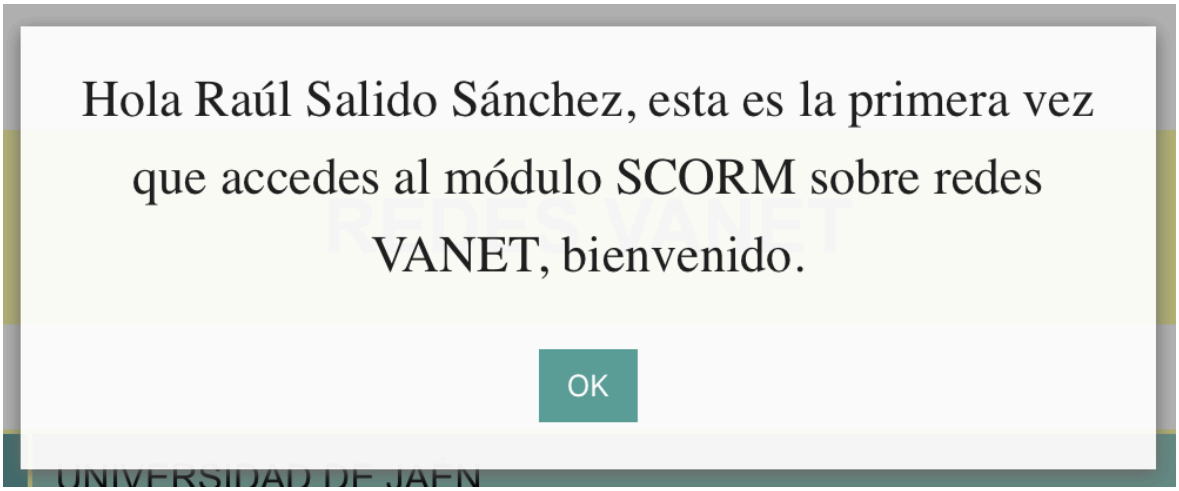


Figura 5.43 Mensaje de bienvenida a un SCO VANET. Fuente: Propia

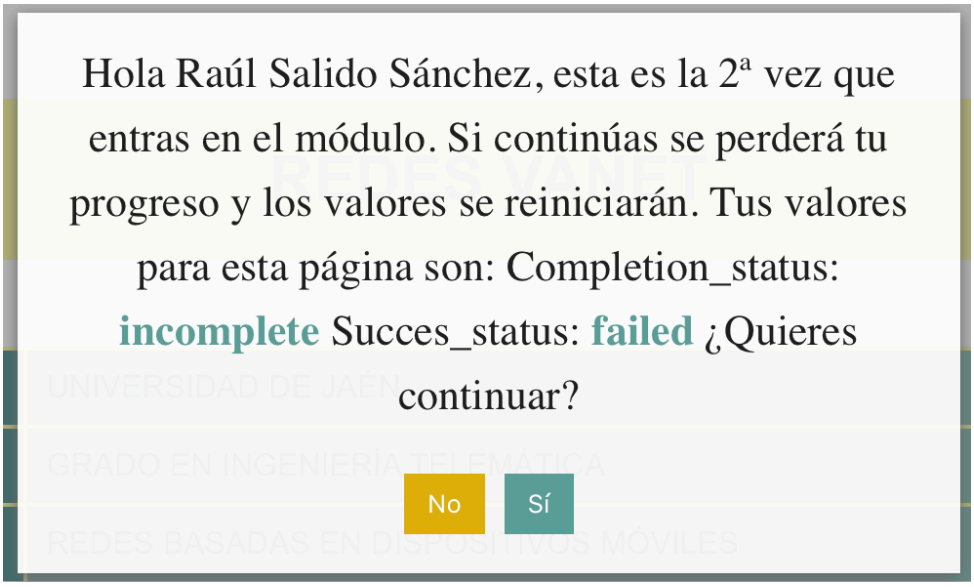


Figura 5.44 Mensaje bienvenida a un SCO repetido VANET. Fuente: Propia

La segunda página es una página específica de teoría donde se explican conceptos básicos como:

- ¿Qué es una red Ad-Hoc?
- MANET (Mobile Ad-Hoc Network).
- REDES AD-HOC vs OTRAS REDES
- VANET (Vehicular Ad-hoc Network)
- Aplicaciones más usuales de las redes Ad-Hoc.
- Características más comunes de las redes Ad-Hoc.

Al final de esta página aparecerá un botón para la realización de un test de que es de forma obligatoria para poder pasar a la siguiente página. Las características de este test son las siguientes:

- Contendrá cinco preguntas (la puntuación máxima de cada pregunta será de un punto).
- No hay un límite de tiempo para la ejecución del test.
- Las preguntas aparecerán de manera aleatoria.
- Las respuestas a las preguntas también aparecerán de forma aleatoria sin seguir ningún tipo de orden.
- Si hay comunicación con el LMS.

Una vez mostrado el test, justo debajo de este aparecerá un nuevo botón denominado “Mostrar Resultados” que al pulsarlo mostrará una tabla a modo de resumen del test realizado. Un ejemplo de esta tabla se puede ver en la figura 5.45.

Resultados del Test	
Alumno:	Raúl Salido Sánchez
Fecha de inicio:	2018-01-18T17:19:51
Fecha de finalización:	2018-01-18T17:20:24
Tiempo empleado en el test:	00:00:33
Contador de interacciones:	0
Estado de terminación:	Completado
Calificación:	Aprobado
Puntuación mínima:	-1.33 puntos
Puntuación máxima:	5 puntos
Puntuación obtenida:	5 puntos
P1. Respuesta, puntos y latencia:	Entornos donde no es posible poner infraestructuras[,]Militares || 1 puntos || 00:00:10
P2. Respuesta, puntos y latencia:	802.11p 5.9GHz || 1 puntos || 00:00:03
P3. Respuesta, puntos y latencia:	Diversos vehículos se conectan entre ellos sin el uso de ninguna infraestructura. || 1 puntos || 00:00:06
P4. Respuesta, puntos y latencia:	Red establecida con el fin de ofrecer un servicio personalizado, normalmente improvisado, para una aplicación específica.[,]Uso limitado a un periodo de tiempo determinado. [,]Protocolos personalizados para ajustarse a las aplicaciones específicas que motivaron la creación de la red. || 1 puntos || 00:00:06
P5. Respuesta, puntos y latencia:	autonomo || 1 puntos || 00:00:05

Figura 5.45 Tabla resumen test. Fuente Propia

Justo debajo de esta tabla se mostrará un nuevo botón para pasar a la siguiente página o SCO de este WebLab. Al hacer clic sobre este botón se realizan una serie de procesos que son:

- Si la puntuación obtenida al realizar el test es más alta que la mitad del total de puntos en juego en el test, el alumno ha aprobado el test y, por tanto, se comunica al LMS que el estado del test es aprobado (passed). Por el contrario, si la nota obtenida por el alumno al realizar el test es más baja que la mitad del total de puntos en juego en el test, el alumno ha suspendido el test y el estado es de suspenso (failed).
- El LMS recibe la puntuación que ha obtenido el alumno (tanto si es suspenso como aprobado) y, además, también recibe las notas, tanto máxima como mínima que se puede obtener en dicho test.
- En última instancia, se cierra la sesión y se pasa a la siguiente página o SCO.

La tercera página será una página sencilla que contendrá una descripción de cómo utilizar la simulación existente en la siguiente página (SCO cuatro).

En esta descripción se le proporcionará al alumno los conceptos necesarios para la comprensión y manipulación de la simulación realizada. Se indicará cual es el objetivo de dicha simulación, una imagen de cómo es su interfaz gráfica y una descripción de los colores y herramientas usadas.

La cuarta página contendrá la simulación. Esta simulación estará ambientada en el uso que tienen la redes VANET en la que se puede apreciar diferentes vehículos en movimientos en dos carriles distintos, en sentido contrario y el mismo sentido.

En la simulación habrá un juego interactivo en el cual los alumnos deberán de tratar de hacer pasar un mensaje de alerta por el máximo número de vehículos posibles sin que este desaparezca de la pantalla, en el momento en que el mensaje desaparezca del frame de la simulación, esta seguirá en ejecución, pero ya no se podrá visualizar este mensaje más. El mensaje solo podrá traspasarse de un vehículo a otro si estos están en cobertura entre sí y por tanto se muestra la flecha de comunicación entre ambos.

En la cabecera de esta página habrá un botón para que el alumno modifique la configuración a su gusto. Al presionar el botón aparecerá una ventana modal (figura 5.46 en la cual se podrán modificar:

- Audio.
- Idioma.
- Velocidad.
- Subtítulos.

Figura 5.46 Modal para configuración de WebLab. Fuente: Propia.

Se puede ver como cada vehículo cuenta con su propia cobertura de red, la cual al entrar en contacto con algún otro vehículo (cabe destacar que tiene que entrar en contacto la cobertura de un vehículo con otro vehículo en sí, no con la cobertura del otro vehículo) aparecerán una serie de flechas de distintos colores indicando que se ha establecido una comunicación unidireccional o bidireccional entre los vehículos. Los dos tipos de flechas que se puede ver son:

- Rojas: conexión de vehículos en el mismo carril.
- Verdes: conexión de vehículos de distintos carriles.

Una vez comprendido esto, imagine que tiene dos coches que circulan en el mismo carril y el coche más retrasado tiene una velocidad mayor que el que va por delante, pues bien, al entrar el coche que viene por detrás dentro de la cobertura del coche que hay por delante, el coche de detrás disminuirá la velocidad de manera automática y esta velocidad se igualará a la que tiene el vehículo que va por delante, evitando así un accidente.

Al tratarse de una simulación interactiva, el alumno podrá realizar varias funciones y así poder probar y ver cosas a su antojo como:

- Iniciar y pausar la simulación en cualquier momento de la misma.
- Avanzar la simulación durante un periodo de tiempo muy corto.
- Reiniciar la simulación desde cero.
- Aumentar y disminuir el área de cobertura de los coches.
- Movimiento hacia arriba del mensaje
- Movimiento hacia abajo del mensaje.
- Movimiento hacia la derecha del mensaje.
- Movimiento hacia la izquierda del mensaje.

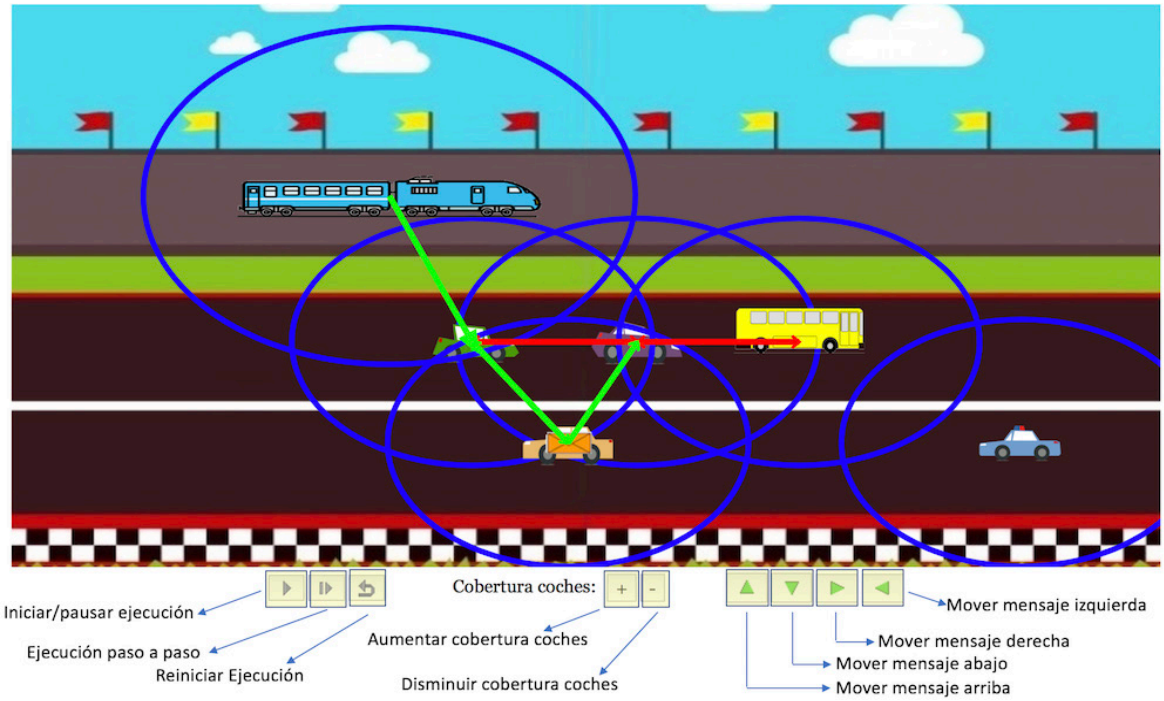


Figura 5.47 Simulación VANET. Fuente: Propia

Tras la comprensión de la simulación por parte del alumno se procederá a la realización de una serie de actividades que indicarán al alumno que realizar una serie de pasos en la simulación y tras esto, deberá de pulsar el botón de comprobar, con esto se asegura que el alumno ha realizado los pasos correctamente y sabe utilizar la simulación. Un ejemplo de actividad se muestra en la figura 5.48.

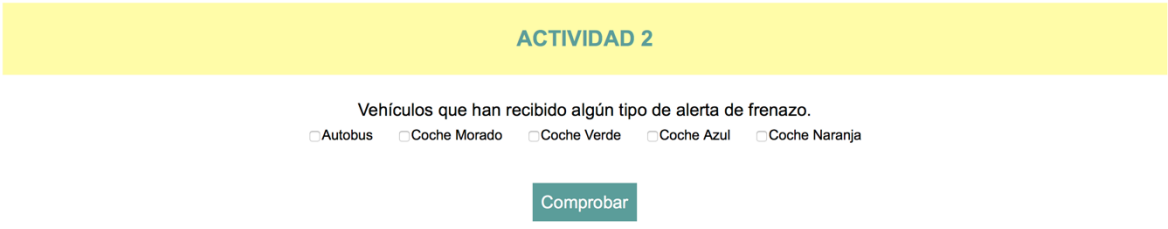


Figura 5.48 Actividad comprensión simulación. Fuente: Propia

Durante la realización de estas actividades, se irán mostrando distintos mensajes para comunicar al alumno si la actividad se ha realizado de forma correcta, o por el contrario ha cometido algún error, estos mensajes irán acompañados de dos sonidos distintos, uno para cuando aciertan, y otro para cuando fallan. Estas actividades tendrán una puntuación cada una:

- Juego mensaje: Realizada correctamente sumará 6 puntos.
- Actividad 1: Realizada correctamente sumará 2 puntos.

- Actividad 2: Realizada correctamente sumará 2 puntos.

Una vez realizadas las actividades de forma correcta, aparecerá un botón para pasar a la página 5.

Por último, la página 5 contendrá un test a modo de evaluación final que pondrá a prueba los conocimientos adquiridos, este test es de forma obligatoria para poder finalizar el WebLab. Las características de este test son las siguientes:

- Contendrá diez preguntas (la puntuación máxima de cada pregunta será de un punto).
- No hay un límite de tiempo para la ejecución del test.
- Las preguntas aparecerán de manera aleatoria.
- Las respuestas a las preguntas también aparecerán de forma aleatoria sin seguir ningún tipo de orden.
- Si hay comunicación con el LMS.

Tras la realización de este test se podrá observar su correspondiente tabla resumen de resultados y después se podrá finalizar el WebLab cerrando la sesión del mismo.

5.10.2 *WebLab SCORM OLSR*

En este WebLab seguirá la misma estructura que el anterior, por lo que podrá distinguir entre 5 SCOs (páginas) cuyo objetivo es dar una pequeña introducción al protocolo de enrutamiento *Optimized Link State Routing Protocol* (OLSR) usado en las redes MANET.

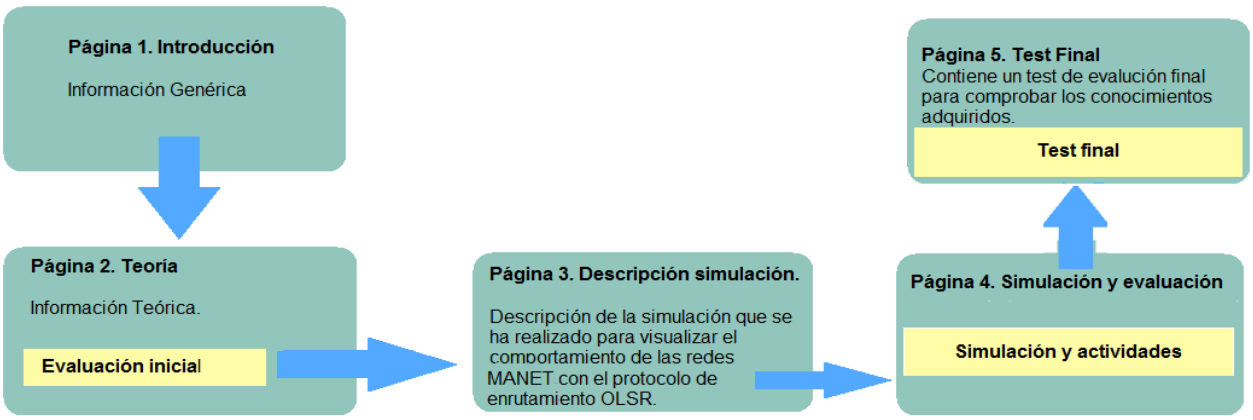


Figura 5.49 Estructura WebLab OLSR. Fuente: Propia.

En la primera página, nada más iniciar el WebLab se muestra un mensaje de alerta modificado gráficamente gracias a la librería *smoke.js.* Este mensaje puede ser de dos tipos:

1. Si se trata de la primera vez que se accede a este WebLab, el mensaje que se mostrará será de bienvenida. Este mensaje es personalizado en el que aparece el nombre del usuario. Se puede ver un ejemplo de este mensaje en la figura 5.50.
2. Si no es la primera vez que se accede al WebLab se mostrará un mensaje en el cual se le dirá al usuario el número de veces que ha accedido a este SCO. Además, muestra la nota de evaluación que se ha obtenido en el último intento y realiza la pregunta de si se quiere seguir en este SCO o si se pretende pasar al siguiente. Se puede ver un ejemplo en la figura 5.51.

Este primer SCO indicará información relativa al WebLab como: objetivos del WebLab, estructura, etc.

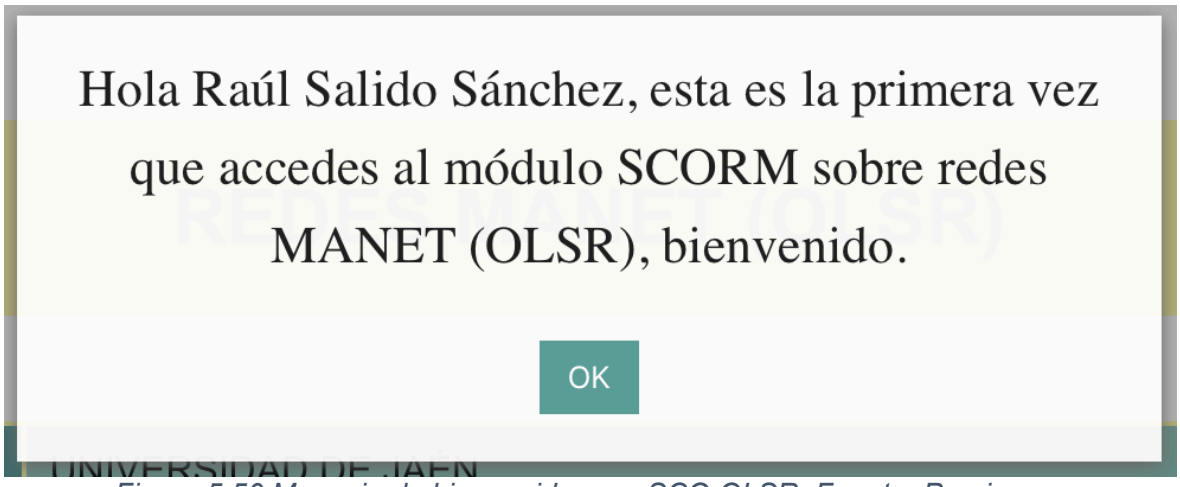


Figura 5.50 Mensaje de bienvenida a un SCO OLSR. Fuente: Propia

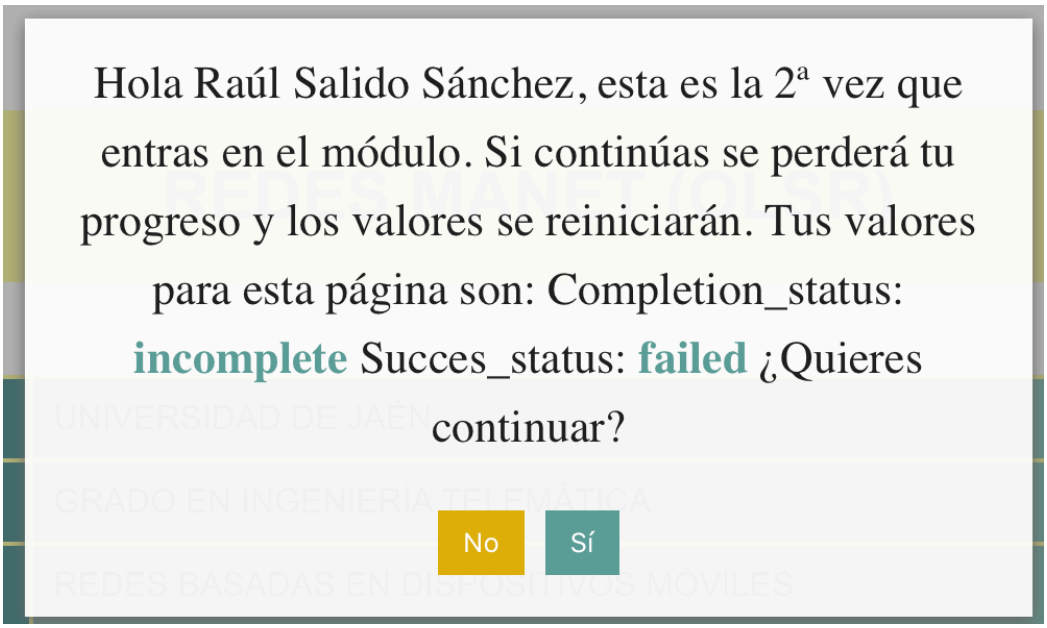


Figura 5.51 Mensaje bienvenida a un SCO repetido OLSR. Fuente: Propia

La segunda página es una página específica de teoría donde se explican conceptos básicos como:

- ¿Qué es una red Ad-Hoc?
- Aplicaciones más usuales de las redes Ad-Hoc.
- Características más comunes de las redes Ad-Hoc.
- REDES AD-HOC vs OTRAS REDES.
- ¿Qué es OLSR?
- Características de OLSR.

Al final de esta página aparecerá un botón para la realización de un test de que es de forma obligatoria para poder pasar a la siguiente página. Las características de este test son las siguientes:

- Contendrá ocho preguntas (la puntuación máxima de cada pregunta será de un punto).
- No hay un límite de tiempo para la ejecución del test.
- Las preguntas aparecerán de manera aleatoria.
- Las respuestas a las preguntas también aparecerán de forma aleatoria sin seguir ningún tipo de orden.
- Si hay comunicación con el LMS.

Una vez mostrado el test, justo debajo de este aparecerá un nuevo botón denominado “Mostrar Resultados” que al pulsarlo mostrará una tabla a modo de resumen del test realizado. Un ejemplo de esta tabla se puede apreciar en la figura 5.52.

Resultados del Test	
Alumno:	Raúl Salido Sánchez
Fecha de inicio:	2018-01-18T17:36:49
Fecha de finalización:	2018-01-18T17:39:01
Tiempo empleado en el test:	00:02:12
Contador de interacciones:	0
Estado de terminación:	Completado
Calificación:	Aprobado
Puntuación mínima:	-1.33 puntos
Puntuación máxima:	8 puntos
Puntuación obtenida:	8 puntos
P1. Respuesta, puntos y latencia:	Es un protocolo de enrutamiento proactivo basado en tablas para redes móviles ad hoc (MANET). || 1 puntos || 00:01:09
P2. Respuesta, puntos y latencia:	698 || 1 puntos || 00:00:06
P3. Respuesta, puntos y latencia:	1 || 1 puntos || 00:00:04
P4. Respuesta, puntos y latencia:	Protocolos personalizados para ajustarse a las aplicaciones específicas que motivaron la creación de la red.[,]Uso limitado a un periodo de tiempo determinado.[,]Red establecida con el fin de ofrecer un servicio personalizado, normalmente improvisado, para una aplicación específica. || 1 puntos || 00:00:07
P5. Respuesta, puntos y latencia:	lanmar || 1 puntos || 00:00:07
P6. Respuesta, puntos y latencia:	Los nodos, además de transmitir y recibir sus datos, reenvían datos del resto de nodos.[,]Es una red Ad-Hoc cuyos nodos tienen capacidad de movimiento. || 1 puntos || 00:00:11
P7. Respuesta, puntos y latencia:	Es un protocolo de enrutamiento proactivo los nodos de la red intercambian información sobre la topología regularmente. || 1 puntos || 00:00:18
P8. Respuesta, puntos y latencia:	Los nodos colaboran para enviarse información enrutando paquetes de salto a salto. || 1 puntos || 00:00:08

Figura 5.52 Tabla resumen test. Fuente: Propia

Justo debajo de esta tabla se mostrará un nuevo botón para pasar a la siguiente página o SCO de este WebLab. Al hacer clic sobre este botón se realizan una serie de procesos que son:

- Si la puntuación obtenida al realizar el test es más alta que la mitad del total de puntos en juego en el test, el alumno ha aprobado el test y, por tanto, se comunica al LMS que el estado del test es aprobado (passed). Por el contrario, si la nota obtenida por el alumno al realizar el test es más baja que la mitad del total de puntos en juego en el test, el alumno ha suspendido el test y el estado es de suspenso (failed).
- El LMS recibe la puntuación que ha obtenido el alumno (tanto si es suspenso como aprobado) y, además, también recibe las notas, tanto máxima como mínima que se puede obtener en dicho test.

- En última instancia, se cierra la sesión y se pasa a la siguiente página o SCO.

La tercera página será una página sencilla que contendrá una descripción de cómo utilizar la simulación existente en la siguiente página (SCO cuatro).

En esta descripción se le proporcionará al alumno los conceptos necesarios para la comprensión y manipulación de la simulación realizada. Se indicará cual es el objetivo de dicha simulación, una imagen de cómo es su interfaz gráfica y una descripción de los colores y herramientas usadas.

La cuarta página contendrá la simulación. Esta simulación estará ambientada en el comportamiento del protocolo de enrutamiento OLSR es una red MANET en la que se puede apreciar diferentes nodos en movimientos de forma aleatoria.

La cabecera de esta página tendrá un botón para que el alumno modifique la configuración a su gusto. Al presionar el botón aparecerá una ventana modal (figura 5.53 en la cual se podrá modificar:

- Audio.
- Idioma.
- Velocidad.
- Subtítulos.

Figura 5.53 Modal para configuración de WebLab. Fuente: Propia.

Se puede ver como cada nodo cuenta con su propia cobertura de red, la cual al entrar en contacto con algún otro nodo (cabe destacar que tiene que entrar en contacto la cobertura de un nodo con otro nodo en sí, no con la cobertura del otro nodo) aparecerán una serie de flechas indicando que se ha establecido una comunicación unidireccional o

bidireccional entre los nodos. Los distintos colores que podrá tener un nodo son los siguientes:

- Verde: no está en cobertura con ningún nodo.
- Amarillo: Está dentro de la cobertura de otro nodo.
- Rojo: Está dentro de la cobertura de uno o varios nodos y además este nodo es troncal (MPR) dentro de la red.

Al tratarse de una simulación interactiva, el alumno podrá realizar varias funciones y así poder probar y ver cosas a su antojo como:

- Iniciar y pausar la simulación en cualquier momento de la misma.
- Avanzar la simulación durante un periodo de tiempo muy corto.
- Reiniciar la simulación desde cero.
- Seleccionar el número de nodos que aparecerán en pantalla.
- Eliminar un nodo (el nodo que se eliminará será el nodo cuyo índice numérico sea el más alto).
- Aumentar y disminuir el área de cobertura de los nodos.

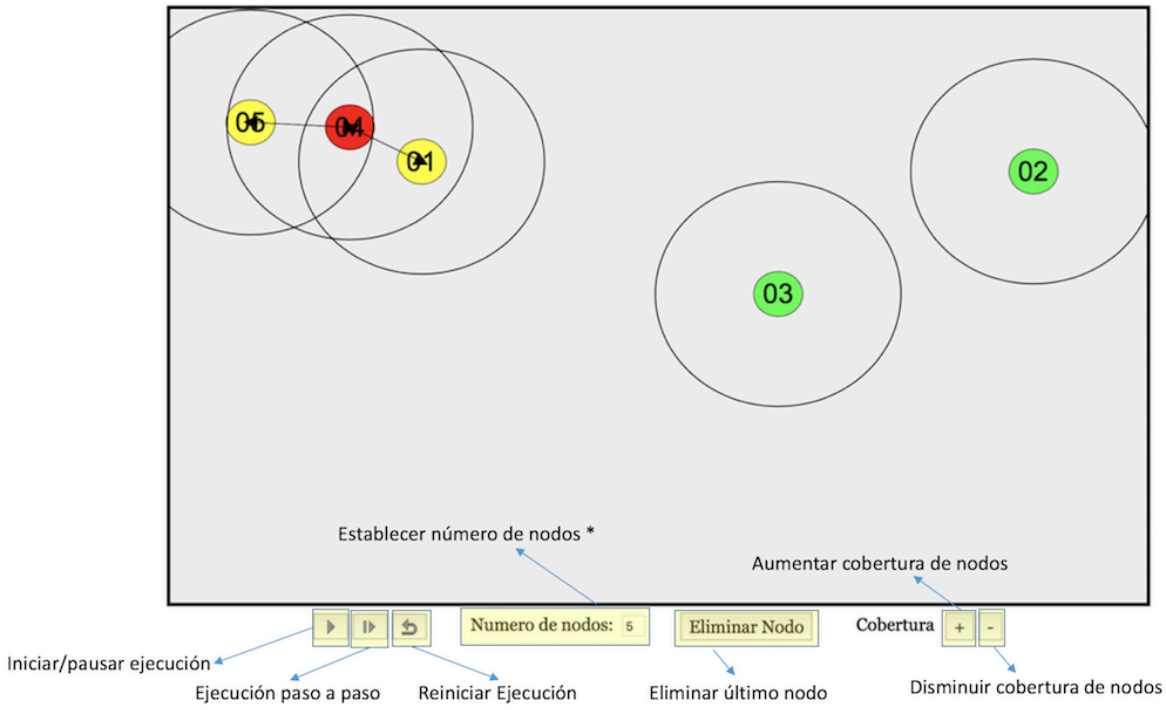


Figura 5.54 Simulación OLSR. Fuente: Propia

Tras la comprensión de la simulación por parte del alumno se procederá a la realización de una serie de actividades que indicarán al alumno que realizar una serie de pasos en la simulación y tras esto, deberá de pulsar el botón de comprobar, con esto se asegura que el alumno ha realizado los pasos correctamente y sabe utilizar la simulación. Un ejemplo de actividad se muestra en la figura 5.55.

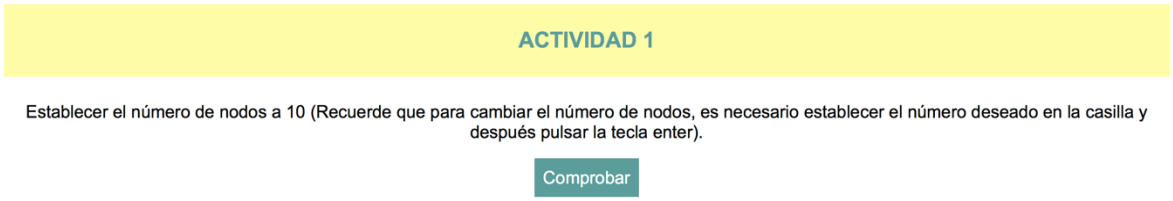


Figura 5.55 Actividad comprensión simulación. Fuente: Propia

Durante la realización de estas actividades, se irán mostrando distintos mensajes para comunicar al alumno si la actividad se ha realizado de forma correcta, o por el contrario ha cometido algún error, estos mensajes irán acompañados de dos sonidos distintos, uno para cuando aciertan, y otro para cuando fallan. Estas actividades tendrán una puntuación cada una:

- Actividad 1: Realizada correctamente sumará 2.5 puntos.
- Actividad 2: Realizada correctamente sumará 2.5 puntos.
- Actividad 3: Realizada correctamente sumará 5 puntos.

Una vez realizadas las actividades de forma correcta, aparecerá un botón para pasar a la página 5.

Por último, la página 5 contendrá un test a modo de evaluación final que pondrá a prueba los conocimientos adquiridos, este test es de forma obligatoria para poder finalizar el WebLab. Las características de este test son las siguientes:

- Contendrá diez preguntas (la puntuación máxima de cada pregunta será de un punto).
- No hay un límite de tiempo para la ejecución del test.
- Las preguntas aparecerán de manera aleatoria.
- Las respuestas a las preguntas también aparecerán de forma aleatoria sin seguir ningún tipo de orden.
- Si hay comunicación con el LMS.

Tras la realización de este test se podrá observar su correspondiente tabla resumen de resultados y después se podrá finalizar el WebLab cerrando la sesión del mismo.

Para el aspecto gráfico y estilo que se visualizan en los test se ha utilizado un archivo CSS ya implementados pero que han sido modificados en muchos aspectos por cuestiones de gusto y adaptación al estilo que tiene por defecto el LMS usado. El archivo usado ha sido *estiloIndex.css* cuyo autor es *Sergio Díaz Fuentes.*

6 DISCUSIÓN Y RESULTADOS

El resultado final de este TFG ha sido la creación de dos módulos SCORM que trabajan bajo la versión 2004 del mismo, totalmente interactivos y personalizables que realizan una evaluación automatizada para cada alumno.

Toda la información de cada alumno al realizar cualquiera de los dos módulos será almacenada por el LMS, lo que significa, que el profesor, podrá acceder a esta información para saber lo que el alumno ha hecho concretamente.

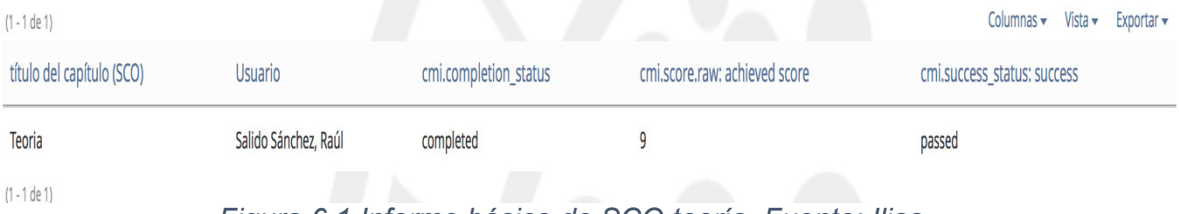
Para ver todas las posibilidades que ofrecen estos módulos SCORM se realizará el módulo que habla sobre el protocolo de encaminamiento OLSR y se visualizarán algunos aspectos de la información que estará disponible para el profesor (estos informes estarán más detallados en los anexos).

Las páginas más relevantes de este módulo son las siguientes:

- Teoría.html
- Simulación_OLSR.html
- TestFinalOLSR.html

Son las más relevantes ya que son las que más comunicaciones realizan con el LMS, debido a que estas páginas incorporan test interactivos, configuraciones de página, etc.

La página de teoría contiene un test cuya realización es de manera obligatoria para el alumno, una vez realizado el test se podrá ver desde el LMS la información mostrada en la figura 6.1.


(1 - 1 de 1) Columnas ▾ Vista ▾ Exportar ▾

título del capítulo (SCO)	Usuario	cmi.completion_status	cmi.score.raw: achieved score	cmi.success_status: success
Teoria	Salido Sánchez, Raúl	completed	9	passed

(1 - 1 de 1)

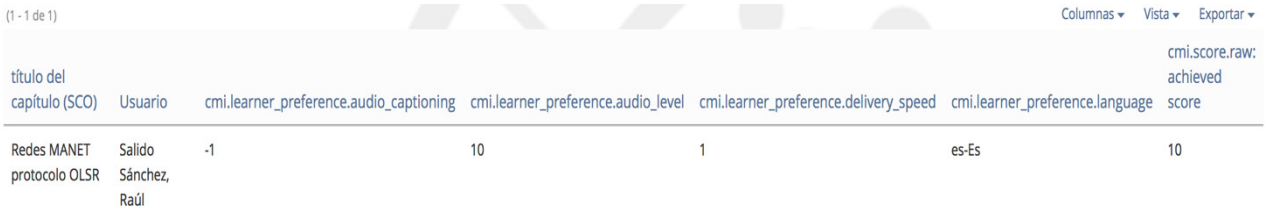
Figura 6.1 Informe básico de SCO teoría. Fuente: Ilias.

Los atributos *completion_status* y *succes_status* se actualizan de la siguiente forma:

- ***completion_status**:* Por defecto tiene el valor incompleto, pero al pulsar en el botón de pasar a la siguiente página cambia a completado.
- ***succes_status:*** Por defecto tiene el valor suspenso (failed) pero al realizar el test y dependiendo de la puntuación que se obtenga en el mismo este valor cambiará a aprobado (passed) o se mantendrá en suspenso. La puntuación del SCO será la puntuación obtenida en el test.

La siguiente página en la que fijarse es la página que contiene la simulación. En esta página el alumno podrá establecer la configuración de distintos aspectos de la misma como el audio, velocidad, idioma y subtítulos.

Los valores de *completion_status* y *succes_status* se evalúan de la misma forma que en la página de teoría, pero la puntuación del SCO vendrá determinada por la realización de las distintas actividades que se encontrarán en el SCO. En la figura 6.2 se verá un informe básico de este SCO.



(1 - 1 de 1) Columnas ▾ Vista ▾ Exportar ▾

título del capítulo (SCO)	Usuario	cmi.learner_preference.audio_captioning	cmi.learner_preference.audio_level	cmi.learner_preference.delivery_speed	cmi.learner_preference.language	cmi.score.raw: achieved score
Redes MANET protocolo OLSR	Salido Sánchez, Raúl	-1	10	1	es-Es	10

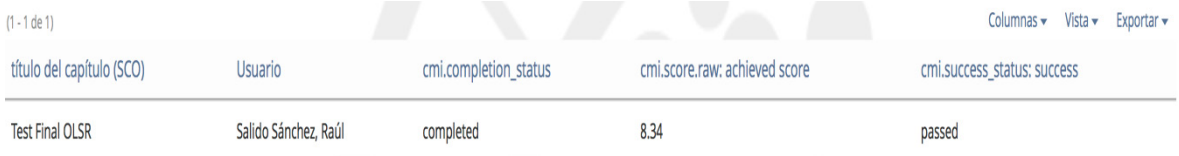
Figura 6.2 Informe básico de SCO simulación. Fuente: Ilias.

Ahí se puede ver la elección de la configuración de los atributos de la página escogida por el alumno siendo:

- **audio_captioning:** Se corresponde con los subtítulos y los valores que admite son -1 para desactivarlos y 1 para activarlos.
- **Audio_level:** Nivel del volumen del audio de la página siendo el valor 0 para audios desactivados y el valor 10 para audios activados.
- **Delivery_speed:** Velocidad de la simulación siendo el valor 1 para una velocidad baja, el valor 5 para una velocidad media y el valor 10 para una velocidad alta.
- **Preference.language:** Idioma de la página siendo el valor es-Es el valor que indica el idioma español (solo está disponible este idioma).

Por último, la última página de la simulación es la que contiene el test final para evaluar los conocimientos adquiridos del alumno. Los valores de *completion_status* y *succes_status* se evalúan de la misma forma que en la página de teoría y la puntuación final del SCO vendrá determinada por la puntuación obtenida en el test.

En la figura 6.3 se verá un informe básico de este SCO.



(1 - 1 de 1) Columnas ▾ Vista ▾ Exportar ▾

título del capítulo (SCO)	Usuario	cmi.completion_status	cmi.score.raw: achieved score	cmi.success_status: success
Test Final OLSR	Salido Sánchez, Raúl	completed	8.34	passed

Figura 6.3 Informe básico de SCO TestFinalOLSR. Fuente: Ilias.

Al finalizar este SCO se podrá finalizar el módulo y usando los filtros del LMS se podrá observar el informe básico de éxito de este módulo, que mostrará lo que hay en la figura 6.4.


(1 - 1 de 1) Columnas Vista Exportar

Título del módulo de aprendizaje	Usuario	Estado	Intentos	Número de SCO completados	Número de SCOS superados
v10	Salido Sánchez, Raúl		2	5	5

Figura 6.4 Informe básico de éxito. Fuente: Ilias.

Para el otro módulo SCORM desarrollado el funcionamiento es el mismo.

Todos los tipos de informes que ofrece el LMS se encontrarán de manera más detallada en la parte de anexos.

7 CONCLUSIONES Y LINEAS FUTURAS

Como se ha descrito en el punto anterior, el resultado de este TFG es la creación de dos módulos SCORM que trabajan bajo la versión 2004 del mismo, totalmente interactivos y personalizables para cada alumno. Estos módulos han sido probados e integrados satisfactoriamente en el LMS de una Universidad de Jaén.

Esto ayuda a la evolución de la enseñanza online quedando presente la incorporación de contenido interactivo usando simulaciones, actividades, test, etc. Esto ha sido posible utilizando herramientas que permiten crear este tipo de contenido como es EJS (*Easy Java Simulations*). Estos módulos también han sido desarrollados siguiendo el estándar QTI, lo que hace que el contenido sea portable e interoperable a los distintos LMS que hay en el mercado.

Se han completado los objetivos que fueron propuestos en los inicios de este TFG además de propinar algunas funcionalidades extra que no se contemplaron como la elaboración de un juego, lo que añade el concepto de gamificación, por lo que, los alumnos serán estimulados a intentar lograr completar el objetivo de este juego para obtener la máxima puntuación del SCO correspondiente. Concretamente, los objetivos alcanzados han sido:

- Obtención de dos módulos que ayudan a la compresión de las redes Ad-Hoc, MANET y VANET.
- Interactividad:
    - Control de secuenciación.
    - Respuesta a preguntas con respuesta de acierto/error inmediata.
    - Personalización a gusto del alumno (Audio, Idioma, velocidad, subtítulos).
    - Trabajo en simulación.
    - Ver resultados del trabajo en la misma simulación.
- Facilitar el trabajo al tutor gracias a la generación de informes.
- Sirve de ejemplo para ver posibilidades en la creación de futuros laboratorios web.
- Integración con el LMS, el cual es la principal herramienta TIC en la enseñanza universitaria.

Por otra parte, gracias a la elaboración de este TFG, se han desarrollado una serie de habilidades y competencias de las que antes, el autor del mismo, no disponía como:

- Ampliación de los conocimientos sobre el lenguaje de programación JavaScript.
- Ampliación de los conocimientos sobre el lenguaje de marcas HTML.

- Ampliación de los conocimientos sobre la elaboración de hojas de estilos usando el lenguaje de diseño gráfico CSS.
- Estudio, desarrollo y correcto uso de módulos basados en estándar SCORM.
- Uso de la herramienta de desarrolladores de distintos navegadores para la adaptación de los módulos desarrollados a estos.
- Estudio y aprendizaje de las funciones usadas para la comunicación con un LMS.
- Desarrollo de la capacidad creativa.
- Mejora de la habilidad de organización.
- Mejora de la habilidad de trabajo.
- Mejora de la capacidad a la hora de afrontar retos.
- Mejora de la capacidad de recopilación de información.
- Mejora de la capacidad de redacción.

Líneas de futuro:

- Añadir más cantidad de actividades interactivas.
- Profundizar más en los conceptos teóricos.
- Añadir mejoras a los módulos y descripción de las mismas (mayor personalización, mayor accesibilidad, etc.).
- Desarrollo de módulos de otras temáticas distintas.

8 ANEXOS

8.1 Informes proporcionados por el LMS.

En este anexo se verán todos los tipos de informes proporcionados por el LMS, en este caso se trata de ILIAS perteneciente a la universidad de Jaén. Estos informes podrá obtenerlos el profesor o administrador de los WebLabs para ver toda la información que requiera oportuna.

En primer lugar, para acceder a estos informes será necesario ir al apartado de configuración del WebLab y una vez ahí irnos a la pestaña de 'datos de seguimiento' como se muestra en la figura 8.1.

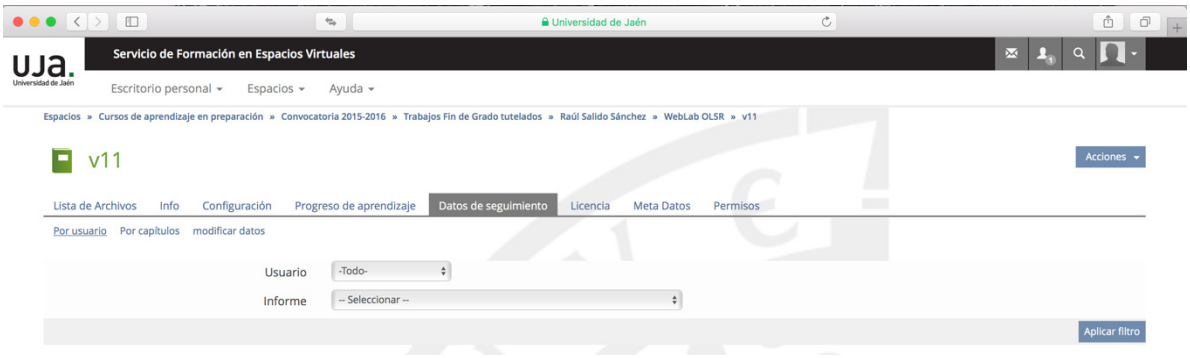


Figura 8.1 Datos de seguimiento ILIAS. Fuente: ILIAS.

Ahora se podrá distinguir dos grandes tipos de informes, los informes 'por usuario' y los informes 'por capítulos'.

Por un lado, en los informes 'por usuario' encontrará dos casillas, una para seleccionar el usuario del que se quiere obtener los informes y otro para seleccionar el tipo de informe que se quiere obtener. Los tipos de informes que se pueden obtener aquí son:

- Informe básico de éxito.
- Informe básico de capítulos (SCO).
- Informe básico de interacciones.
- Informe de objetivos.
- Informe de objetivos globales del sistema (usado en la secuenciación).

Por otro lado, en los informes 'por capítulos' encontrará dos casillas, una para seleccionar el capítulo del que se quiere los informes y otro para seleccionar el tipo de informe que se quiere obtener. Los tipos de informes que se pueden obtener aquí son:

- Informe básico de éxito.
- Informe básico de capítulos (SCO).
- Informe básico de interacciones.

- Informe de objetivos.
- Informe de objetivos globales del sistema (usado en la secuenciación).

A continuación, se verán cada uno de los informes que se podrán obtener para las dos partes.

8.1.1 *Informe básico de éxito.*

Contendrá información básica sobre el módulo, como la que se puede apreciar en la figura 8.2. En la pestaña que se puede apreciar denominada 'Columnas' se podrá acceder a otro tipo de información realizando el filtrado pertinente.


(1 - 1 de 1) Columnas ▾ Vista ▾ Exportar ▾

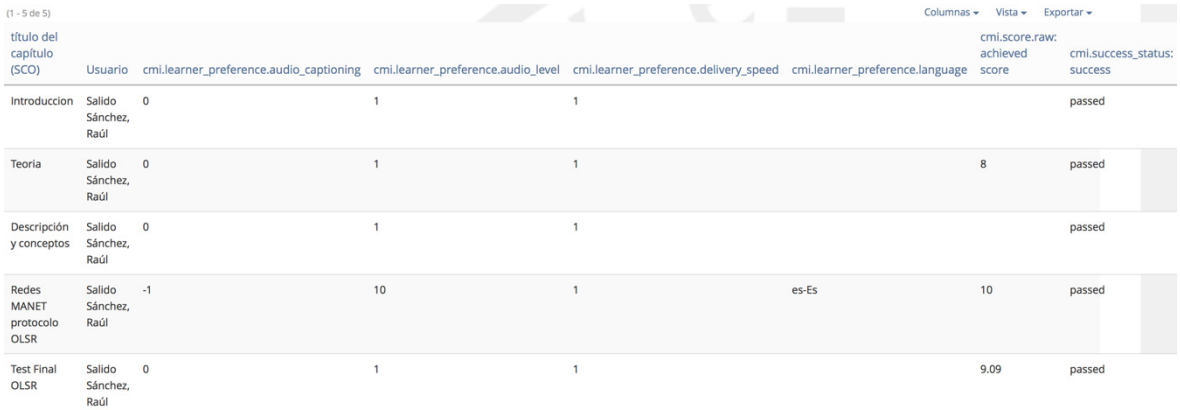
Título del módulo de aprendizaje	Usuario	Estado	Intentos	Número de SCO completados	Número de SCOS superados
v11	Salido Sánchez, Raúl		1	5	5

(1 - 1 de 1)

Figura 8.2 Informe básico de éxito. Fuente: ILIAS.

8.1.2 *Informe básico de capítulos (SCO).*

En este informe se podrán observar las diferentes opciones de configuración que se han escogido para el desarrollo de cada SCO, además de la nota obtenida en cada SCO y el estado del mismo.


(1 - 5 de 5) Columnas ▾ Vista ▾ Exportar ▾

título del capítulo (SCO)	Usuario	cmi.learner_preference.audio_captioning	cmi.learner_preference.audio_level	cmi.learner_preference.delivery_speed	cmi.learner_preference.language	cmi.score.raw: achieved score	cmi.success_status: success
Introduccion	Salido Sánchez, Raúl	0	1	1			passed
Teoria	Salido Sánchez, Raúl	0	1	1		8	passed
Descripción y conceptos	Salido Sánchez, Raúl	0	1	1			passed
Redes MANET protocolo OLSR	Salido Sánchez, Raúl	-1	10	1	es-Es	10	passed
Test Final OLSR	Salido Sánchez, Raúl	0	1	1		9.09	passed

Figura 8.3 Informe básico de capítulos (SCO). Fuente: ILIAS.

8.1.3 *Informe básico de interacciones.*

Se podrá observar las distintas interacciones que ha realizado el alumno con el módulo y ver distinta información sobre cada interacción, como la respuesta que dio el alumno, el tiempo que tardó en responder, etc.

(1 - 22 de 22) Columnas Vista Exportar

título del capítulo (SCO)	Usuario	Id	cmi.interactions.n.result	cmi.interactions.n.latency: time	cmi.interactions.n.learner_response: Response
Redes MANET protocolo OLSR	Salido Sánchez, Raúl	1			
Teoria	Salido Sánchez, Raúl	il_1950_qst_25384	correct	PT00H00M08.93S	b
Teoria	Salido Sánchez, Raúl	il_1950_qst_25201	correct	PT00H00M10.88S	a
Teoria	Salido Sánchez, Raúl	il_1950_qst_25224	correct	PT00H00M05.72S	698
Teoria	Salido Sánchez, Raúl	il_1950_qst_25385	correct	PT00H00M09.33S	a[,]b[,]c[,]d
Teoria	Salido Sánchez, Raúl	il_1950_qst_25197	correct	PT00H00M05.50S	a[,]b[,]c
Teoria	Salido Sánchez, Raúl	il_1950_qst_25223	correct	PT00H00M07.09S	b[,]d
Teoria	Salido Sánchez, Raúl	il_1950_qst_25383	correct	PT00H00M09.83S	a
Teoria	Salido Sánchez, Raúl	il_1950_qst_25202	correct	PT00H00M04.24S	1
Redes MANET protocolo OLSR	Salido Sánchez, Raúl	Actividad1	correct		1
Redes MANET protocolo OLSR	Salido Sánchez, Raúl	Actividad2	correct		1
Redes MANET protocolo OLSR	Salido Sánchez, Raúl	Actividad3	correct		0,0,0,0,0,6,7,0,0

Figura 8.4 Informe básico de interacciones. Fuente: ILIAS.

8.1.4 *Informe de objetivos.*

Se observarán los distintos objetivos propuestos para el módulo y si han sido superados o no.

(1 - 2 de 2) Columnas Vista Exportar

título del capítulo (SCO)	Usuario	Id	cmi.completion_status
Teoria	Salido Sánchez, Raúl	Teoria_OLSR	completed
Redes MANET protocolo OLSR	Salido Sánchez, Raúl	simulacion_OLSR	completed

(1 - 2 de 2)

Figura 8.5 Informe de objetivos. Fuente: ILIAS.

8.1.5 *Informe de objetivos globales del sistema (usado en la secuenciación).*

Se observará un informe global sobre el módulo como si ha sido superado o no y la satisfacción del mismo.

(1 - 1 de 1) Columnas Vista Exportar

Título del módulo de aprendizaje	Usuario	estado total	Status: satisfied
v11	Salido Sánchez, Raúl	completed	satisfied

(1 - 1 de 1)

Figura 8.6 Informe de objetivos globales del sistema. Fuente: ILIAS.

8.2 Siglas

ADL	*Advanced Distributed Learning*
AODV	*Ad hoc On-Demand Distance Vector*
API	*Interfaces de Programación de Aplicaciones*
BATMAN	*Better Approach To Mobile Adhoc Networking*
CAM	*Content Aggregation Model*
CERN	*Conseil Européen pour la Recherche Nucléaire*
CSS	*Cascading Style Sheets*
DARPA	*Agencia de Investigación de Proyectos Avanzados de Defensa*
DOM	*Document Object Model*
DSDV	*Destination Sequenced Distance Vector*
DSR	*Document Object Model*
ECMA	*European Computer Manufacturers Association*
EJS	*Easy Java Simulations*
FP	*Formación Profesional*
HTML	*HyperText Markup Language*
IDE	*Entorno de Desarrollo Integrado*
IETF	*Internet Engineering Task Force*
LMS	*Learning Management System*
MANET	*Mobile Ad-Hoc Network*
MPR	*Retransmisor Multipunto*
OLSR	*Optimized Link State Routing Protocol*
PHP	*Hypertext Preprocessor*
PIF	*Packet Interchange File*
PRNET	*Packet Radio Networks*
QTI	*Question & Test Interoperability*

RTE	*Run-Time Environment*
SCO	*Shareable Content Objects*
SCORM	*Sharable Content Object Reference Model*
SGML	*Standard Generalized Markup Language*
SN	*Secuenciación y Navegación*
TIC	*Tecnologías de Información y Comunicación*
TORA	*Temporally Ordered Routing Algorithm*
VANET	*Vehicular Ad-Hoc Network*
VLE	*Virtual Learning Environment*
WSN	*Wireless Sensor Networks*
W3C	*World Wide Web Consortium*
XHTML	*EXtensible HyperText Markup Language*
XML	*Extensible Markup Language*
ZRP	*Zone Routing Protocol*

8.3 Índice de figuras

9 REFERENCIAS BIBLIOGRÁFICAS

[1] I. Ruano Ruano, “Tema 1- Clasificaciones de redes Ad-Hoc.,” 2017.

[2] “LMS UJAEN.” [Online]. Available: https://dv.ujaen.es/login.php?target=&soap_pw=&ext_uid=&cookies=nocookies&client_id=docencia&lang=es.

[3] F. Esquembre, “Easy Java Simulations web oficial.” [Online]. Available: http://www.um.es/fem/EjsWiki/Main/HomePage.

[4] EasyLMS, “¿Qué es SCORM? Definición y significado explicado.” [Online]. Available: https://www.easy-lms.com/es/ayuda/centro-de-conocimiento-lms/que-es-scorm/item10195.

[5] A. Ruano Ruano, “INTEGRACIÓN DE LABORATORIOS ONLINE DE AUTOMÁTICA Y TELECOMUNICACIÓN EN LOS SISTEMAS DE GESTIÓN DE APRENDIZAJE MEDIANTE SCORM,” Universidad de Jaén, 2016.

[6] J. Pérez Porto and A. Gardey, “Definicion de HTML,” 2012. [Online]. Available: https://definicion.de/html/.

[7] W3schools, “HTML y XHTML.” [Online]. Available: https://www.w3schools.com/html/html_xhtml.asp.

[8] W3C, “Docu ento Object Model (DOM).” [Online]. Available: https://www.w3.org/DOM/.

[9] W3C, “Extensible Markup Language (XML),” 2016. [Online]. Available: https://www.w3.org/XML/.

[10] J. Eguiluz, “Historia JavaScript,” 2007. [Online]. Available: http://librosweb.es/libro/javascript/capitulo_1/breve_historia.html.

[11] J. Orellana, “Introduccion hisotria CSS,” 2013. [Online]. Available: https://es.slideshare.net/johnorellanaec/introduccion-historia-css.

[12] “Estandar QTI description.” [Online]. Available: http://pdi.topografia.upm.es/m.manso/docs/Estandar_QTI_Descripcion.pdf.

[13] I. Ruano Ruano, “Tema 5 - Protocolo de enrutamiento OLSR.” 2017.

[14] cultura y deporte. Ministerio de educación, “Nota : Estadística de las Enseñanzas no universitarias . Datos avance 2016 - 2017,” vol. 915. pp. 1–10, 2017.

[15] Empeloyformacion, “Introduccion al e-learning.” .

[16] T. Clausen and P. Jacquet, “Optimized Link State Routing Protocol (OLSR) RFC 3626,” *Ietf Rfc3626*, p. 75, 2003.

[17] L. Coya Rey, T. O. Ledesma Quiñones, and W. Baluja García, “Implementación De Una Manet Routing Protocol ´ S Selection for a Manet Implementation.” pp. 1–11,

2014.

[18] J. Eguiluz, “Introducción a XHTML,” 2014.

[19] A. Lopez de Aberasturi, “Historia de las hojas de estilo,” 2013. [Online]. Available: http://www.profesordeinformatica.com/css/historia.

[20] W3schools, “The HTML DOM.” [Online]. Available: https://www.w3schools.com/js/js_htmldom.asp.

[21] S. Diaz Fuentes, “CREACIÓN DE TESTS INTERACTIVOS SCORM,” 2016.

[22] Advanced Distributed Learning (ADL), *Content Aggregation Model (CAM) SCORM 2004 4th Ed. v1.1*. 2009.

[23] Advanced Distributed Learning (ADL), *Sequencing and Navigation [SN] SCORM 2004 4th Edition*. 2009.

[24] M. P. Laguna Lozano, “Introducción al Modelo de Referencia SCORM.” p. 78, 2011.

[25] “Beneficios JavaScript.” [Online]. Available: https://www.nextu.com/blog/conoce-las-ventajas-y-desventajas-de-javascript/.

[26] “NetBeans.org.” [Online]. Available: https://netbeans.org/index_es.html.

[27] Advanced Distributed Learning (ADL)., *Run-Time Environment [RTE] SCORM 2004 4th Edition*. 2009.

[28] “Chrome Blog,” 2013. [Online]. Available: https://chrome.googleblog.com/2013/05/live-from-google-io-mo-screens-mo.html.

[29] “The Mozilla Blog,” 2010. [Online]. Available: https://blog.mozilla.org/blog/2010/03/31/introducing-the-mozilla-state-of-the-internet-report/.

[30] “Smoke.js.” [Online]. Available: https://smoke-js.com.