

Aplicación web para la creación y gestión de gramáticas electrónicas

Nahuel Nillni Perera

Resumen—Los lenguajes naturales tales como el castellano y otros idiomas difieren en gran medida de los lenguajes formales como las matemáticas, debido a que su creación ha sido generada espontáneamente a lo largo de la historia con el simple propósito de comunicarse, sin hacer especial énfasis en unas reglas que deben cumplirse estrictamente. La sintaxis gramatical de estos lenguajes naturales ha sido estudiada en profundidad durante miles de años, y con estos estudios se ha podido llegar a la conclusión que la definición de estas gramáticas consiste en una jerarquía de elementos gramaticales. Actualmente, existen aplicaciones que permiten generar estas sintaxis a partir de elementos gramaticales organizados de forma jerárquica gracias a las estructuras en forma de árbol, las cuales son extensamente utilizadas en la informática. Este proyecto pretende desarrollar una de estas aplicaciones capaz de gestionar sintaxis gramaticales, pero facilitándole la interacción al usuario utilizando una interfaz gráfica y eliminando la necesidad de descarga de ningún tipo de software haciendo uso de un servicio web.

Palabras clave—Aplicación software, Entornos de programación, Gramática, Servicios web, Sistemas cliente-servidor

Abstract—Natural languages such as English and other languages differ greatly from formal languages like mathematics because its creation has been produced spontaneously throughout history with the only purpose of communicating, without making a particular emphases on certain rules that must be strictly enforced. The grammatical syntax of these natural languages has been extensively studied for thousands of years, and these studies have been able to conclude that the definition of these grammars consists of a hierarchy of grammar elements. Currently, there are applications that can generate these syntaxes from grammatical elements organized hierarchically through the tree-shaped structures, which are widely used in computing. This project aims to develop one of these applications capable of handling grammatical syntaxes, but providing a better user interaction by using a graphical interface and removing the need of downloading any kind of software through the use of a web service.

Index Terms—Application software, Client-server systems, Grammar, Programming environments, Web services



1 INTRODUCCIÓN

EL conjunto de los lenguajes naturales, formado por los idiomas que han estado en una constante modificación a lo largo de miles de años, han mantenido un único propósito durante toda su existencia: la comunicación.

Este propósito de comunicación, trasciende las reglas ya creadas en ese propio idioma y por lo tanto nos encontramos con una evolución en la cual no se mantiene ningún tipo de patrón y estos idiomas son modificados espontáneamente e incontrolablemente con cierta regularidad.

Las teorías ya extensamente estudiadas sobre la sintaxis gramatical de los lenguajes naturales han conseguido representar a ésta con estructuras jerárquicas en las que se define cada elemento gramatical que compone la sintaxis a estudiar [1]-[5].

Este trabajo pretende conseguir desarrollar una herramienta web para estudiantes y maestros

especializados en las lenguas, en la cual éstos serán capaces específicamente de crear y gestionar sus propias sintaxis gramaticales en ella sin la necesidad de descargar e instalar la aplicación en el sistema operativo, gracias a la utilización del navegador web.

A día de hoy, existen ciertas aplicaciones que permiten la creación de estas sintaxis gramaticales pero no suelen ser aplicaciones web disponibles al público ni tampoco suelen contener una interfaz gráfica amigable para el usuario, lo que tiende a asustarle en lo que se refiere a su utilización, especialmente en este grupo de usuarios que no está realmente acostumbrado al uso de aplicaciones informáticas sin interfaz gráfica.

Por esta razón, la implementación de una interfaz gráfica y acceso a la aplicación via navegador web, junto con un sistema de cuentas de usuario para poder almacenar y editar las sintaxis gramaticales creadas, permitirá a los usuarios una mejor interacción y un mayor rendimiento en su desarrollo, con mejoras de rapidez y comodidad en la creación de éstas.

Esto último comentado será el principal objetivo del proyecto y en las próximas secciones de este documento se especificará con más detalle todo el proceso realizado para

-
- E-mail de contacto: nahuel.nillni@gmail.com
 - Mención realizada: *Enginyeria de Tecnologies de la Informació*.
 - Trabajo tutorizado por: Marc Ortega Gil (*departamento de Ingeniería de la Información y de las Comunicaciones*)
 - Curso 2013/14

el desarrollo de la aplicación, las herramientas utilizadas durante el proceso y las funcionalidades implementadas en ésta.

2 BASE DE DATOS

2.1 Diseño de la base de datos

Las primeras semanas del proyecto consistieron en una investigación para determinar cómo almacenar las gramáticas en la base de datos y que otros datos serían también necesarios.

Debido a que la aplicación web como requerimiento principal debía permitir cuentas de usuario para guardar las gramáticas de éstos, era obvia la creación de una tabla de usuarios que en nuestro caso será llamada USERS, y en la cual se almacenarán datos básicos como el nombre del usuario, sus apellidos, un email el cual será utilizado para ingresar en la aplicación y una contraseña.

Por obvias razones de seguridad, actualmente es necesario cifrar la contraseña antes de ser almacenada en la base de datos, ya que almacenándola en texto plano provocaría grandes riesgos en la confidencialidad del usuario. Por esta razón la contraseña será cifrada con una función hash sha512, lo cual implicará que se requieran 128 caracteres en el campo contraseña de la base de datos.

En el momento del registro del usuario también se generará un identificador único de usuario para tener constancia de una forma sencilla qué usuario es el que está ingresado actualmente en cada una de las sesiones PHP. En otras palabras, este identificador será la clave primaria de la entrada del usuario.

Además, como requerimiento se propuso que la base de datos no pueda contener ningún grupo de usuarios con la misma dirección de correo electrónico. Esta comprobación es fácilmente implementable tanto en la parte del cliente como en la del servidor antes del registro, no obstante para una mejor integridad de los datos, este requerimiento también se comprueba en la base de datos marcando el campo como único, lo cual implica que no pueden existir dos entradas con el mismo correo electrónico en la base de datos.

Por otro lado, los usuarios también deben ser capaces de almacenar y recuperar sus propias sintaxis gramaticales, lo cual implica la creación de una nueva tabla en la base de datos nombrada GRAMMARS, que contendrá los siguientes campos: un nombre que servirá para identificar la gramática cuando el usuario desee cargarla en la interfaz gráfica de la aplicación web, una descripción resumida de la funcionalidad de esta sintaxis, la expresión que compone la sintaxis gramatical por tal de poder ser renderizada en la interfaz gráfica de la aplicación web al recuperarla de la base de datos, el identificador del usuario al que pertenece esa gramática el cual se puede recuperar desde la tabla USERS, y por último un identificador único para la propia gramática.

En este último caso, la clave primaria de la tabla será el identificador de la gramática ya que es único, mientras que el identificador de usuario podrá repetirse en múltiples entradas de la tabla en función del número de gramáticas que ese usuario haya almacenado durante su estancia en la

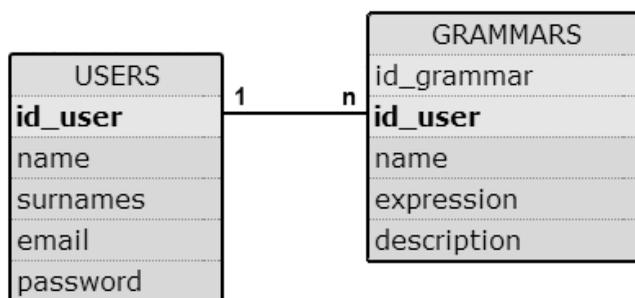


Fig. 1. Esquema de la base de datos utilizada en el proyecto, en concreto se pueden observar los campos utilizados en las tablas y la relación entre éstas.

aplicación web.

Para el principal funcionamiento de la aplicación web no ha sido necesario diseñar ninguna otra tabla en la base de datos por lo que nos quedaría un diseño como el que se muestra en la Fig. 1. No obstante, en una futura extensión sería posible la necesidad de nuevas tablas para permitir al usuario interactuar con nuevas funcionalidades de la aplicación web.

2.2 Configuración de la base de datos

Una vez diseñada la base de datos se procedió a instalar un servidor de base de datos MySQL 5.6 en el sistema. El principal motivo de la utilización de este servidor de base de datos se debe a la experiencia ya adquirida con el tipo de sintaxis que utiliza y su facilidad de configuración.

Por lo que respecta a la configuración, cabe destacar que el puerto utilizado para acceder a la base de datos se encuentra cerrado en el router del sistema, lo cual implica que los datos son totalmente inaccesibles desde una red externa, lo cual asegura la privacidad, confidencialidad e integridad de los datos a ataques por parte de organizaciones ajenas.

No obstante, se debe mencionar que en el caso de este proyecto la base de datos es compartida tanto en el entorno de producción como en el entorno de desarrollo, lo cual resulta completamente contradictorio con lo comentado justo hace un momento por lo que respecta a la integridad que se quería conseguir con el acceso de la base de datos sólo desde local, ya que modificar datos de producción desde el entorno de desarrollo es altamente peligroso.

Para una correcta integridad de los datos, el entorno de desarrollo debería conectarse a una diferente base de datos destinada solamente a ese entorno. Debido a que este trabajo se trata de un proyecto no comercial y se dispone de unos recursos limitados, no se ha sido estrictamente necesaria la creación de una segunda base de datos puramente destinada al entorno de desarrollo y se ha utilizado la misma para los dos entornos.

3 SERVIDOR WEB

El servidor web utilizado en el proyecto ha sido el conocido Apache, versión 2.0.65 para Windows 7, el cual ha sido instalado en el mismo sistema que el servidor de base de datos por motivos de comodidad. En un ambiente

más profesional, estos dos servidores deberían estar separados en diferentes sistemas por motivos de seguridad y control.

Por tal que los usuarios sean capaces de acceder a este servidor desde redes externas, el router ha sido configurado para enrutar las conexiones entrantes y salientes de la red interna desde el puerto 80.

No obstante, para el correcto desarrollo de la plataforma web también ha sido creado un servidor web virtual al cual se puede acceder por el puerto 8080 estrictamente desde redes internas, y el cual está dirigido a un diferente árbol de ficheros. Gracias a esta configuración es posible desarrollar en el árbol de ficheros de desarrollo y ver los cambios realizados a través del puerto 8080 desde la red local, mientras que por otro lado, el árbol de ficheros de producción, el cual está disponible al público, no resulta modificado de ninguna manera. En el esquema de la Fig. 2 se puede observar de una manera gráfica cómo están interconectados los diferentes elementos del sistema.

Si a esta configuración añadimos un sistema de control de versiones como Subversion, el cual se ha utilizado en este proyecto y se comentará en las próximas secciones, tenemos un entorno de programación con un gran aspecto profesional que nos permite desarrollar continuamente de una forma mucho más cómoda y a prueba de errores, manteniendo historiales de los cambios realizados y pudiendo comparar a tiempo real el código del entorno de producción con el de desarrollo.

Por otro lado, también ha sido necesaria la instalación de un módulo PHP en el servidor web por tal de ejecutar

el código en la parte del servidor. La versión utilizada para este proyecto ha sido la 5.2.6, la cual contiene las mejoras necesarias para un correcto funcionamiento de la aplicación web que se creará, por lo que no es necesario instalar las últimas versiones PHP 5.5 que tan sólo aportan mejoras de rendimiento [6].

Además, ha sido necesaria la configuración del módulo PHP para la activación de cookies y sesiones para el correcto funcionamiento de la aplicación web. Si estas funcionalidades no son activadas, no sería posible mantener sesiones con el usuario para identificarlo y recuperar sus datos y gramáticas, ni seríamos capaces de generar cookies para una mayor comodidad al navegar en la aplicación web.

Finalmente, por lo que respecta al directorio root del servidor web, se debe comentar el uso de los ficheros ".htaccess" tanto en él como en sus subcarpetas, los cuales nos permiten definir restricciones y excepciones a cada una de ellas, controlando de esta forma qué conexiones son capaces de acceder a los diferentes contenidos del servidor web. En nuestro caso, el acceso a la carpeta "Model", la cual contiene todos los métodos del modelo, está restringida a todo tipo de conexión y solamente es accesible desde local, ya que sería un gran fallo en la seguridad de nuestra aplicación si los atacantes pudiesen acceder al código de la parte del servidor.

4 SISTEMA DE CONTROL DE VERSIONES

Tal y como se ha descrito en la sección anterior, realmente

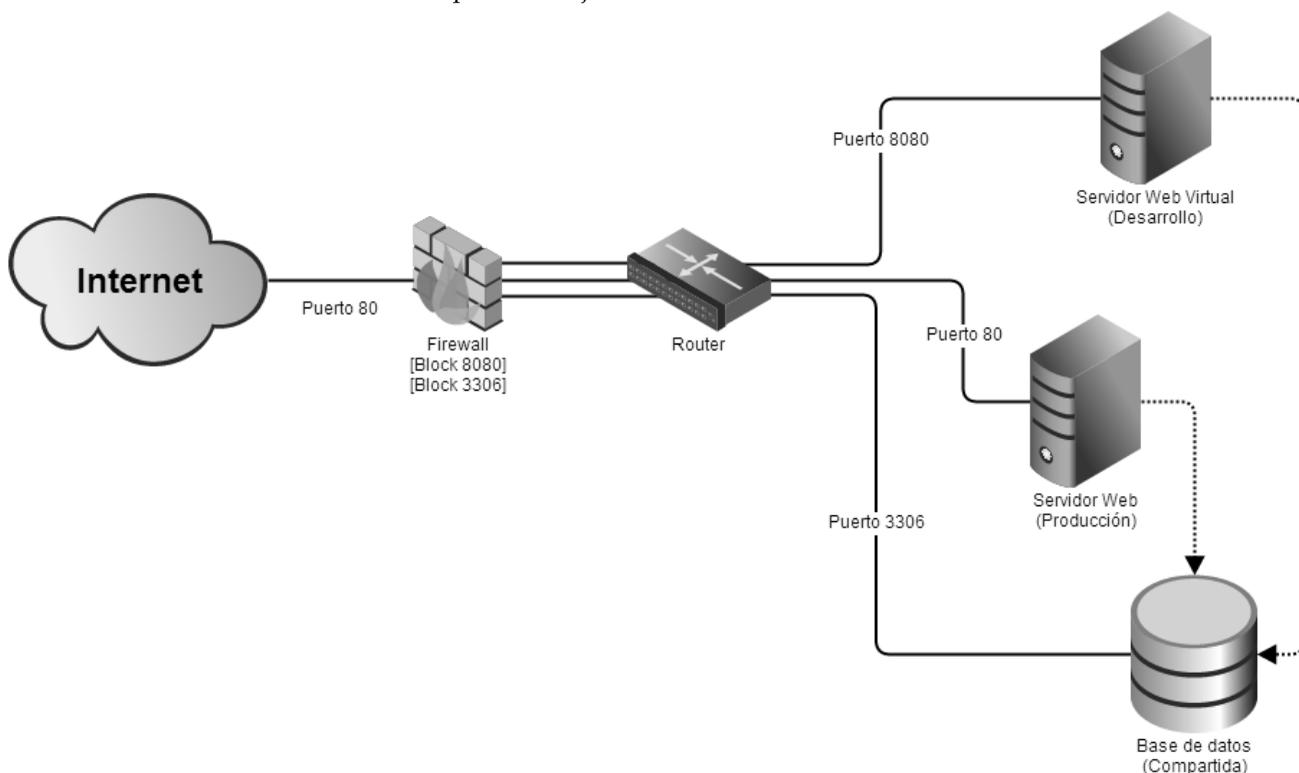


Fig. 2. Esquema de la arquitectura de los sistemas utilizados en el proyecto, concretamente mostrando la base de datos, los servidores web y las conexiones entre ellos.

han sido creados dos árboles de ficheros por tal de implementar un sistema de control de versiones. El software instalado para poder implementar este sistema ha sido Silk Subversion 1.8.8 para el sistema operativo Windows 7.

Una vez instalado este software es posible ejecutar los comandos que implementa Subversion desde la consola del sistema operativo, los cuales están disponibles en la propia página oficial de Subversion [7].

De entre todo el surtido de comandos que nos proporciona Subversion, los más importantes para este proyecto han sido los siguientes: uno que nos permite la creación de un nuevo repositorio a partir de unos ficheros ya creados (`svn import`), otro que nos permite la actualización de un repositorio (`svn update`), uno que nos permite crear una copia local de trabajo de un repositorio en concreto (`svn checkout`) y finalmente un comando que nos permite efectuar commits al repositorio (`svn commit`), el cual se ha utilizado indirectamente desde la interfaz gráfica del entorno de programación empleado en este proyecto, el cual se explicará en la próxima sección del artículo.

Inicialmente, para crear el sistema del control de versiones de este proyecto, ha sido necesario importar los ficheros iniciales mediante el comando “`svn import`”, los cuales se encuentran en una carpeta llamada “`repository`”.

Una vez generado el sistema de control de versiones se procede a generar una copia local de trabajo (`working copy`) mediante el comando “`svn checkout`”, en otra carpeta diferente la cual será nombrada “`repositoryDev`”.

Estas dos carpetas son respectivamente los directorios root de los en el servidores web de producción y de desarrollo, y mediante comandos “`svn commit`” podemos subir los cambios de la copia local de trabajo (o desarrollo) al entorno de producción.

Sin embargo, esta subida no es suficiente para poder visualizar las modificaciones realizadas en el servidor web de producción desde el navegador web. Para poder realmente generar los cambios de producción, es necesario actualizar el árbol de ficheros a la nueva revisión mediante el comando “`svn update`” en tal directorio.

Gracias a estos simples comandos que nos proporciona Subversion, se ha podido generar un sistema de control de versiones que, aún siendo muy básico, nos permite una gran eficiencia en el desarrollo de la aplicación web y un acercamiento a los métodos utilizados en ámbitos profesionales.

5 ENTORNO DE PROGRAMACIÓN

Finalmente, para concluir el apartado de configuración de los sistemas, los servidores y las aplicaciones necesarias para el desarrollo y la implementación de la aplicación web, se debe mencionar el entorno de programación utilizado para crear el código necesario para el funcionamiento de la aplicación web.

El software utilizado ha sido NetBeans IDE 8.0, un entorno de programación que permite desarrollar en todos los lenguajes web que se utilizan en este proyecto: HTML, CSS, Javascript y PHP.

Además, este software permite la compatibilidad con Subversion para generar commits desde el directorio de ficheros de desarrollo hacia el directorio de producción, así como poder recuperar el historial de cambios para filtrar errores que se hayan podido cometer durante las diferentes revisiones del código, y hacer comparaciones entre el código modificado y el que se encuentra actualmente publicado.

Se trata de un software de código libre con un gran repertorio de funcionalidades que permiten con mucha más facilidad el desarrollo de un proyecto como este.

6 DESARROLLO DE LA APLICACIÓN WEB

La aplicación web de este proyecto tiene como objetivo permitir al usuario tener una interacción con la base de datos de una forma dinámica, pudiendo registrarse y siendo capaz de almacenar y recuperar las gramáticas ya creadas previamente.

Todo el código que se ha generado durante el transcurso de este proyecto ha sido basado en el conocido patrón de arquitectura del software Modelo-Vista-Controlador, a la vez que organizado de una forma lógica y comprensible en las carpetas de nuestro árbol de ficheros, tal y como se muestra en la Fig. 3 de este artículo.

6.1 Modelo

En el modelo de nuestro código podemos encontrar los principales archivos con funciones estáticas en lenguaje PHP, las cuales pueden ser llamadas desde cualquier archivo que cargue el entorno de nuestro código que se encuentra en el fichero nombrado “`utils-LoadEnvironment.php`”.

Este archivo se requiere en todos los ficheros web PHP y su funcionalidad es cargar todos los ficheros del modelo utilizando la función `include()` de integrada en PHP.

Una vez cargado el entorno, podemos obtener los métodos del modelo, los cuales nos permiten: interactuar con la base de datos (“`Database.php`”), gestionar el usuario de la sesión (“`Users.php`”), almacenar y procesar las gramáticas de los usuarios (“`Gramaticas.php`”), utilizar ciertas constantes y funciones

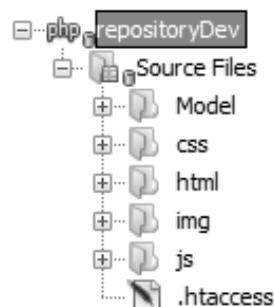


Fig. 3. Imagen mostrando la organización utilizada en el directorio del entorno de desarrollo y producción.

de tiempo (“Time.php”), ejecutar métodos para recuperar parámetros GET y POST a partir de AJAX filtrando los caracteres que pudieran ejecutar código malicioso como SQL injections o XSS, iniciar sesiones y otras funcionalidades web (“Web.php”), recuperar el título de una URL a partir de ésta (“Title.php”) y finalmente, ejecutar otras funciones básicas como comprobar si una string es un email válido a partir de expresiones regulares, cifrar contraseñas con la función hash que hemos elegido para asegurar la privacidad (sha512) o determinar si el código está siendo ejecutado en el entorno de producción o en el de desarrollo (“Utils.php”).

Cabe destacar que todo el modelo ha sido generado desde cero y no se ha utilizado ningún tipo de librería excepto las que vienen integradas con el módulo PHP 5.2.6, implantando un buen modelo de programación con el objetivo de conseguir una buena escalabilidad en los futuros métodos que sean generados por tal de intentar no replicar código innecesario.

6.2 Vista

Para la creación de la vista de nuestra aplicación se han utilizado las tecnologías HTML, CSS y también Javascript, por tal de conseguir una sensación de fluidez y de ejecución a tiempo real, característica que no suele destacar en las páginas web.

Dentro de la carpeta nombrada “html” se puede encontrar la única página web del proyecto en el archivo “index.php”.

En este archivo se crea la vista principal que necesitamos para el correcto manejo de la aplicación, la cual esta compuesta de una cabecera en el margen superior de la ventana que permitirá al usuario ingresar, registrarse o desconectarse, dependiendo de su estado actual, y también un contenedor en el resto de la página donde se encontrará la interfaz gráfica que permite la creación y edición de las sintaxis gramaticales con sus respectivas funcionalidades y será explicada en la proxima sección de este documento.

Por tal de permitir al usuario registrarse y gestionar las gramáticas en la aplicación web sin la necesidad de masacrar la página web con formularios y botones, se han utilizado formularios en capas que son cargadas en la propia página web a partir de la tecnología AJAX, las cuales aparecen sobrepuestas a la principal vista de la página web, utilizando los controladores para determinar qué información debe haber en éstas y si es posible cargarlas (e.g. no es posible cargar tus gramáticas si no estás ingresado o no tienes). En la Fig. 4 se puede observar la capa de registro utilizada en el proyecto.

Para diferenciar los ficheros que contienen estas capas de los demás ficheros PHP que se encuentran dentro de la carpeta “html”, se utiliza la terminación “...LayerAjax.php”. Este tipo de reglas nos proporciona un mejor visionamiento de dónde se encuentra cada elemento de la aplicación web en el caso de implementar alguna nueva funcionalidad.

Por otra parte, dentro de la carpeta “html” también se ha creado un fichero llamado “widgets.php”, en el cual se encuentran funciones PHP que generan la vista de

elementos en concreto, tales como los headers, los contenedores y los formularios. De esta forma obtenemos un buen estilo de programación modulando los elementos sin tener todo en un mismo fichero totalmente inentendible.

Finalmente, cabe destacar el esfuerzo que ha requerido generar una estética decente para la página web y la maquetación de ésta. Para ello se ha generado un único fichero CSS con clases generales tales como márgenes, paddings, tamaños o propiedades (hidden, overflow, etc.) entre otros. Este enfoque también permite una buena modulación a la vez que una mejor eficiencia y entendimiento del código en general evitando la replicación de éste.

6.3 Controlador

Para terminar, tenemos el controlador de nuestro entorno, el cual es el encargado de gestionar las peticiones que genera el usuario, y este se encuentra en dos apartados bien diferenciados de nuestro código.

Primeramente, tenemos el controlador de la parte del cliente en funciones Javascript dentro del fichero “functions.js”, los cuales nos permiten definir el comportamiento que debe seguir el código al ejecutar los elementos de la aplicación web por parte del usuario, dependiendo de las variables actuales, y también nos permite realizar llamadas AJAX por tal de ejecutar código en la parte del servidor y devolverlo para mostrarlo.

Por otra parte, estos ficheros en la parte del servidor son el otro apartado del controlador y se encargan de gestionar funciones del modelo como el ingreso, registro o almacenamiento de sintaxis gramaticales en función de las variables que nos proporciona el usuario al hacer la

The image shows a registration form overlay with a dark grey background. It features five input fields stacked vertically. The first three fields contain the text 'Nahuel', 'Nillni Perera', and 'nahuel@tfg.com' respectively, each with a checkmark icon on the right. The fourth field contains seven dots, representing a password, also with a checkmark icon. The fifth field contains four dots and a warning triangle icon. At the bottom of the form is a button with the text 'Empieza ya!'.

Fig. 4. Imagen mostrando la capa de registro la cual se sobrepone al plano principal de la página web al cargarse mediante AJAX.

petición AJAX. Se pueden diferenciar estos archivos del controlador del resto de ficheros ya que en su nombre siempre contienen la terminación "...Run.php" y se encuentran dentro de la carpeta "html".

En el caso del registro, se comprueba tanto por la parte del cliente como en la del servidor que el nombre contenga entre 3 y 30 caracteres, y solamente sea una palabra mientras que el apellido pueden ser entre 4 y 60 caracteres y hasta dos palabras. Además la contraseña debe tener como mínimo 8 caracteres y por la parte del cliente se comprueba que se haya escrito bien en los dos campos, y una vez en el servidor se hashea la contraseña para almacenarla en la base de datos. Por último, por lo que se refiere al correo electrónico, se ha utilizado la misma expresión regular (regex) en la parte del cliente y del servidor que comprueba que éste sea correcto y además mediante AJAX y incluso en el fichero "registerRun.php" antes del registro se comprueba que ese correo electrónico no esté asociado ya a una cuenta de usuario.

Por otro lado, en lo que se refiere al ingreso a la aplicación web, se trata de un proceso mucho más sencillo, el cual en la parte del cliente comprobamos que se haya escrito un correo electrónico correcto y una contraseña, y si es así se pasa a la parte del servidor para comprobar si esa cuenta de usuario existe para en tal caso empezar la sesión PHP.

Finalmente, nos encontramos con los controladores que permiten recuperar, almacenar y eliminar las sintaxis gramaticales. La razón por la cual estas capas son cargadas por AJAX constantemente al hacer clic en ellas, se debe a que si sólo se cargaran al iniciar la sesión no podríamos ver los cambios realizados en la base de datos sin recargar la página web, lo cual no generaría ningún tipo de sensación de fluidez por parte del usuario.

Las restricciones que existen en estas funcionalidades son: se debe ingresar antes de poder utilizarlas y no es posible eliminar ni cargar gramáticas si el usuario no tiene ninguna almacenada y no es posible guardar una gramática sin nombre o sin elementos gramaticales en la sintaxis.

Por último nos encontramos con la funcionalidad de limpiar la sintaxis gramatical que se encuentra en la interfaz gráfica, la cual no ejecuta ninguna petición AJAX ni comprueba ninguna variable excepto que se confirme la eliminación mediante un alert ejecutado por Javascript.

7 APLICACIÓN GRÁFICA

Una vez ya generado el Modelo-Vista-Controlador con los métodos necesarios para la gestión de los usuarios, la gestión de las sintaxis gramaticales y la principal vista de la aplicación, era necesario proceder a crear la interfaz gráfica que permite al usuario generar y editar sus sintaxis gramaticales.

No obstante, surgen algunas dudas en cuanto a qué tipo de lenguaje de programación a utilizar y el cómo diseñar la aplicación para facilitarle la edición y la creación de las sintaxis gramaticales a los usuarios.

7.1 Lenguaje de la aplicación gráfica

La primera cuestión que se planteó fue qué lenguaje de programación debería ser utilizado para desarrollar esta interfaz gráfica, y inicialmente fueron consideradas las tres opciones mayormente utilizadas en las aplicaciones web: Flash, Java Applet y Javascript.

Flash quedó descartado rápidamente de las otras dos opciones debido a que actualmente se encuentra en un estado bastante obsoleto y requeriría el estudio de otro lenguaje de programación en el cual no se tenía ningún tipo de experiencia. Por otra parte, se trata de un lenguaje que no está disponible en dispositivos móvil, los cuales son uno de los mayores mercados de Internet actualmente. Además, este lenguaje nos proporciona un problema añadido el cual será comentado al explicar la Applet de Java, ya que los dos lenguajes comparten el mismo inconveniente.

Después de descartar el lenguaje Flash por los motivos comentados anteriormente, hubo un intento de generar una Applet de Java cargada en la aplicación web. Mediante un código Java de prueba se compiló una Applet de Java a partir del entorno de programación Eclipse y éste fue cargado en la página web mediante el tag <applet> implementado en HTML, no obstante, rápidamente surgieron los problemas.

El primero de éstos problemas se encuentra cuando se quiere almacenar la gramática desde la Applet de Java, la cual no contiene los métodos del modelo que ya han sido creados en el código PHP y por lo tanto, habría que replicar este mismo código en el proyecto de Java que se ha creado en Eclipse, lo cual supondría un gran coste de recursos del proyecto y una eficiencia nula en lo que se refiere al código.

Por otro lado, hubo un problema más grave todavía con la Applet de Java, el cual se encontraba en la sincronización de las sesiones. En el entorno web es posible ingresar como usuario a partir de AJAX y mantener una sesión PHP, pero no es posible transmitir esta información a la Applet de Java para cambiar su funcionamiento, al menos no de una forma viable. Este es el problema que también nos encontrábamos en Flash. Debido a esta incompatibilidad, la Applet de Java ya no resultaba tan idónea como en un principio cuando se empezó a desarrollar y fue necesario un intento final con lenguaje Javascript para crear la interfaz gráfica.

Javascript permite una total sincronización con las sesiones PHP en cualquier momento a partir de llamadas AJAX a diferencia de los anteriores lenguajes, por esta razón ha sido el lenguaje utilizado en nuestro proyecto. No obstante este lenguaje también contiene sus inconvenientes con respecto a los anteriores.

Las Applet de Java y Flash proporcionan librerías gráficas muy útiles para conseguir un buen diseño y interacción de los elementos mientras que Javascript sólo permite la generación y edición de código HTML, por esta razón fue necesario algo de creatividad para poder generar una estructura de árbol y una representación comprensible por parte del usuario utilizando sólo elementos HTML y CSS.

Además, otro inconveniente que también se encuentra

en la utilización del lenguaje Javascript consiste en que se trata de un código que se encuentra en la parte del cliente a diferencia los lenguajes Flash y Java Applet, los cuales requieren la compilación del código antes de ser utilizado. Esto le permite al usuario ver e incluso modificar el código de la aplicación sin que sea posible ningún tipo de confidencialidad en éste.

Para este proyecto no es necesario un código cerrado ya que no se trata de un software comercial pero en el caso de que así fuese, sería necesario un replanteamiento sobre el lenguaje a utilizar.

7.2 Diseño de la aplicación gráfica

La segunda cuestión que se tuvo que planear fue el diseño necesario para conseguir que los usuarios se sientan cómodos con la interfaz gráfica y les resulte lo mínimamente confusa.

Esta aplicación gráfica que gestiona las sintaxis gramaticales creadas por el usuario, nos debe permitir generar una estructura que contiene los elementos gramaticales que el usuario vaya creando como nodos en forma de árbol. Además, al usuario le debe ser posible la creación de nuevos elementos con respecto al seleccionado, tanto a los lados como justo debajo, y también le debe ser posible la eliminación del elemento gramatical seleccionado.

Al empezar, la aplicación debe encontrarse en un estado inicial donde solamente aparece en el panel un botón para añadir el primer elemento a partir del cual se formará la estructura de la sintaxis gramatical que se quiere crear, y a partir de éste, se podrá ir creando toda la estructura jerárquica.

Todos estos elementos gramaticales están contruidos a partir de contenedores <div> de HTML los cuales contienen como datos un identificador y el nombre del elemento.

Dentro del código PHP del modelo existe un vector donde se encuentran los identificadores y nombres de los elementos básicos creados por defecto: Determinante, Sustantivo, Verbo, Preposición, Adjetivo, Adverbio y el elemento Vacío, que permite al usuario determinar si otro elemento gramatical es opcional de forma gráfica. Además, también es posible utilizar sintaxis gramaticales ya creadas por el usuario como elementos, esto permite una reutilización y más posibilidades para el usuario. En este caso el ID y el nombre del elemento html serán la clave primaria utilizada para almacenar esa sintaxis gramatical en la base de datos y su nombre respectivamente.

Además, por tal de generar bien la estructura jerárquica necesaria para la representación de la sintaxis gramatical, existen contenedores <div> con las clases "column" y "row", los cuales permiten representar tanto en el código como visualmente esta estructura.

Para conseguir una buena visualización del árbol sin mostrar todas las opciones disponibles para cada elemento gramatical de la interfaz gráfica, se ha creado una funcionalidad que permite mostrar las opciones disponibles para ese elemento al hacer clic encima de éste, haciendo uso de código Javascript tal y como se muestra en el esquema de la Fig. 5.



Fig. 5. Esquema de la funcionalidad Javascript que permite mostrar el menú de opciones al hacer clic en un elemento.

En el menú nos encontramos las opciones que permiten añadir un nuevo elemento en la estructura de la interfaz gráfica, y tras hacer clic en una de éstas se ejecuta un código Javascript el cual mediante una petición AJAX muestra la capa de inserción de un nuevo elemento gramatical.

Por otro lado, el botón de eliminación del elemento no ejecuta ninguna petición AJAX para mostrar ninguna capa y simplemente elimina el elemento HTML que se había seleccionado.

Aparte de este menú dinámico que aparece al seleccionar un elemento gramatical, en la interfaz gráfica de la aplicación también se encuentra un sencillo menú que nos permite ejecutar las funcionalidades principales de la aplicación web para la gestión de las gramáticas y su funcionamiento ha sido comentado en la sección anterior. Cabe destacar que estas funcionalidades están separadas en el menú para diferenciar si están disponibles sin estar ingresado en el entorno web o no.

La funcionalidad que permite limpiar la estructura creada siempre está disponible para el usuario mientras que las otras tres funcionalidades sólo están disponibles una vez ya se ha ingresado en la web como usuario. En caso de no estar ingresado y utilizar una de estas funcionalidades se genera un mensaje de error recomendando al usuario su ingreso.

8 RESULTADOS

El resultado del proyecto ha sido esencialmente el estipulado en los objetivos iniciales en la creación de la aplicación web.

Se ha conseguido desarrollar un entorno web que permite crear cuentas de usuario mediante un sencillo registro, y tras esto almacenar y gestionar sus propias gramáticas de manera amigable mediante una interfaz gráfica destinada para usuarios no acostumbrados a aplicaciones informáticas complejas, tal y como se muestra en la Fig. 6.

Indirectamente, también ha sido posible el realizar el despliegamiento de un entorno de desarrollo para una



Fig. 6. Resultado final del proyecto en el cual se muestra la vista de la aplicación web y todas sus funcionalidades, a la vez que una sintaxis gramatical dentro de la interfaz gráfica con sus respectivos elementos gramaticales. En la figura se puede observar como la gramática hace referencia a la expresión regular: *Determinante (Sustantivo + Prep Adv Sustantivo) Verbo (<E> + Adv + Prep Sustantivo)*

aplicación web, instalando y configurando los sistemas necesarios para su correcto funcionamiento tales como los servidores web de desarrollo y producción, la base de datos y un sistema de control de versiones, siendo posible así un buen mantenimiento del proyecto.

La característica más importante que se ha conseguido gracias a este proyecto ha sido la posibilidad de generar en una sola página web todo lo necesario para el funcionamiento de la aplicación a partir de la tecnología AJAX. Esta característica genera en los usuarios una sensación de fluidez y de ejecución en tiempo real la cual consigue que éstos puedan experimentar sin ningún tipo de frustración por tener que aprender un nuevo lenguaje de programación para configurar sus sintaxis gramaticales o tener que tratar con errores que genera la aplicación.

Por otro lado, se debe comentar que la aplicación ha sido desarrollada en el navegador Chrome de Google y cabe la posibilidad que la aplicación no funcione correctamente en otros navegadores. Este proyecto no tenía como objetivo principal la compatibilidad en todos los navegadores disponibles del mercado y por esta razón no se ha realizado una exhaustiva investigación en los detalles necesarios para esta funcionalidad.

En general, por lo que se refiere a la estética y la maquetación web de la aplicación, éstas podrían ser mejoradas con técnicas más avanzadas y una mayor experiencia en el campo de diseño web. Tal y como se especificó en los objetivos de este proyecto, era necesario generar un diseño amigable para los usuarios y por esta razón se ha dedicado una gran parte del tiempo a este apartado por tal de obtener un aspecto considerable en el entorno web, pero podría ser fácilmente mejorado por especialistas en este ámbito.

9 CONCLUSIÓN

Este proyecto ha consistido en un trabajo extenso en cuanto a los diferentes campos de la informática

empleados para su desarrollo, tales como la programación web y de aplicaciones, la configuración de servidores web, la configuración de redes, utilización de bases de datos, diseño web y maquetación web entre otros.

Las líneas de continuación de este proyecto son muy variadas y algunas de ellas podrían resultar muy útiles para la enseñanza.

En un primer informe inicial, como pequeña extensión para el proyecto, se había considerado la creación de un entorno de pruebas en la interfaz gráfica la cual permitiría a los usuarios testear la sintaxis gramatical actual a partir de otra almacenada anteriormente, es decir, probar si una sintaxis gramatical es correcta a partir de otra. Para esto simplemente sería necesario duplicar la interfaz gráfica en otra ventana o panel para ser capaces de crear una segunda gramática en la misma aplicación, y a partir de un parseo Javascript de los identificadores de cada elemento se podría determinar si la sintaxis es correcta o no. Debido al limitado tiempo disponible no se ha realizado esta extensión, pero no se requieren tareas complejas para su creación.

Por otro lado, una línea de continuación con una mayor dificultad consistiría en ser capaz de generar la sintaxis gramatical a partir de una oración escrita por el usuario. Para esto sería necesario la ayuda de una aplicación externa que analizara la oración por tal de determinar como está formada, y una vez hecho esto devolviera a nuestra aplicación la serie de elementos gramaticales que la compone. Aplicando esta funcionalidad, seríamos capaces de testear una oración completa en una sintaxis gramatical que hayamos creado mediante la extensión comentada en el párrafo anterior.

Finalmente, otros aspectos a considerar para la mejora de la aplicación serían un mejor diseño y vista, crear funcionalidades para gestionar la cuenta del usuario y sus datos, y ser capaz de publicar tus gramáticas en un apartado público de la web entre otras muchas funcionalidades al alcance de este proyecto.

AGRADECIMIENTOS

Se agradece la colaboración por parte de Marc Ortega Gil, tutor de este proyecto, por ayudar a comprender y fijar los objetivos de éste.

REFERENCIAS

- [1] W. Brauer, "On Minimizing Finite Automata", Bulletin of the European Association for Theoretical Computer Science, EATCS, vol. 32 , pág. 113-116, 1988.
- [2] J. E. Hopcroft, J. D. Ullman, "Introduction to Automata Theory, Languages and Computation", Addison-Wesley, Reading, Mass., 1979
- [3] M. Mohri, "Finite-state transducers in language and speech processing", Computational Linguistics, vol. 23(2) , pág. 269-311, 1997.
- [4] M. Ortega, "Teoría de Autómatas aplicada a la Lingüística Informática". Facultat de Ciències. Secció d'Enginyeria Informàtica, Universitat Autònoma de Barcelona, 1997.
- [5] M. Silberstein, "Finite State Language Processing with INTEX", en Pre-Conference Tutorial. COLING-ACL-98, Université de Montreal. Montréal, Québec, Canada
<http://www.ladl.jussieu.fr/INTEX/Tutorial.htm>, 1998.
- [6] Mehdi Achour, Friedhelm Betz, Antony Dovgal and several others (the PHP Documentation Group), "PHP Manual", 1997-2014. <http://www.php.net/manual/en/>
- [7] Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato, "Version Control with Subversion, For Subversion 1.2", 2002-2006. <http://svnbook.red-bean.com/en/1.2/svn-book.pdf>