Universidad
de La Laguna

DISEÑO AUTOMÁTICO DE HORARIO Y AGENDA SUJETO A CIERTOS CRITERIOS DE CALIDAD PARA CENTROS CON RECURSOS LIMITADOS DE ESPACIO

AUTOMATIC SCHEDULING AND TIME TABLE DESIGN SUBJECT TO CERTAIN QUALITY CRITERIA FOR CONSTRAINED RESOURCES CENTERS.

Javier Mena Mena

Departamento de Ingeniería informática y sistemas

Escuela Técnica Superior de Ingeniería Informática

Trabajo de Fin de Grado

La Laguna, 7 de Septiembre de 2014

Camino San Francisco de Paula,s/n.Campus de Anchieta 38271-La Laguna E-mail: csinfo@ull.es Tel: 922319042 Fax: 922 319085

D. Jorge Riera Ledesma, con N.I.F 43788061-V, profesor Titular de Universidad adscrito al Departamento de Ingeniería informática y sistemas de la Universidad de La Laguna

C E R T I F I C A

Que la presente memoria titulada, ***"DISEÑO AUTOMÁTICO DE HORARIO Y AGENDA SUJETO A CIERTOS CRITERIOS DE CALIDAD PARA CENTROS CON RECURSOS LIMITADOS DE ESPACIO"***, es realizada por el estudiante Javier Mena Mena con N.I.F 45728657-B, bajo la supervisión y tiene el visto bueno para ser presentada como Trabajo Fin de Grado para la Universidad de La Laguna.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firmarán la presente en La Laguna a 7 de septiembre de 2014.

Resumen

Se presenta el diseño de una aplicación informática interactiva para la planificación automática de horario y agenda de los centros. Esta aplicación, orientada a la ayuda a la decisión, tiene en cuenta ciertos criterios de calidad en lo que al diseño del horario semestral se refiere. Por ejemplo, que el número máximo de horas que un alumno puede recibir al día de una misma asignatura esté acotado, que el conjunto de las horas de teoría precedan a las de problemas, etc. También se tiene en cuenta la carga de trabajo sobre el alumno en el diseño de su agenda. Evitar un pico de carga en determinadas semanas, distribuyendo a lo largo del semestre la entrega de trabajos. Dado que en determinados centros el uso de los laboratorios y de determinadas aulas se comparte entre asignaturas e incluso entre titulaciones, el objeto de esta aplicación es compatibilizar el uso de estos recursos escasos. Una vez que la aplicación proporcione una posible solución, esta puede ser descartada o modificada dinámicamente de acuerdo a las pretensiones de los interesados.

PALABRAS CLAVE: Innovación en el uso de tecnologías de la información y comunicación (TIC), Metodología de coordinación docente, Innovación en la gestión de los centros.

Abstract

We introduce the design of an interactive computer application for automatically scheduling of timetables and agenda in faculties. This computer application, which is decision support system, has certain quality criteria into account, as far as the design of the time table concerns. For instance, the number of lectures received by a student daily from each subject is upper bounded; theoretical lectures must precede practical lectures, etc. The design of the agenda has also into account the student's work load, trying to avoid overload in rush weeks, making a rational distribution of deadlines through the semester. This computer application aims at making compatible the use of scarce resources, because degrees may share some rooms and laboratories. Once the computer application gives us a feasible solution, we can discard it; or modify it according to our objectives.

KEY WORDS: Innovation in information and communication technologies, Cordination methodology in teaching, Innovation in faculty management. Innovation in information and communication technologies,

Agradecimientos

Tras realizar este proyecto es inevitable pensar en todas aquellas personas que me han ayudado a conseguirlo y un sentimiento de agradecimiento me invade.

Muchas gracias al profesor Jorge Riera, quien me ha guiado con sus explicaciones, las cuales conseguían que algo que al principio resultaba difícil de pronto se convirtiera en algo asequible, y gracias a su indiscutible experiencia como docente.

También, tengo que agradecer a esas dos personas que siempre han estado ahí, las que más que nadie me han ayudado a conseguir este objetivo profesional y personal. Ellos son mis padres, las personas que económicamente me han ayudado a costearme la carrera ,pero mi agradecimiento es mayor cuando pienso en el apoyo moral que me han ofrecido, las horas que me han escuchado ensayar este proyecto. Porque son ello y nadie más, los que realmente valoran el trabajo que he realizado.

Y finalmente, gracias a mis amigos y compañeros de carrera que siempre con sus conversaciones conseguían que, por unos minutos olvidara el estrés y presión que supone un proyecto de tal calibre.

Índice de contenidos

Lista de figuras y tablas

Parte I. Introducción

1. Introducción

En la actualidad es imprescindible disponer de herramientas que permitan automatizar la planificación y así poder obtener resultados más efectivos en un menor tiempo. Existen empresas o instituciones que necesitan de estas herramientas para la planificación de sus actividades. Este software será utilizado por la Facultad de Ingeniería Informática de la Universidad de La Laguna para la creación de horarios y la administración de los recursos, ya que en la actualidad este proceso no se realiza de forma automática.

La planificación de un horario para la Universidad requiere de un alto esfuerzo y bastante tiempo. Cuando esta planificación tiene una gran cantidad de variables y restricciones, este problema se vuelve muy complejo. Por lo tanto, este software llevara a cabo una mejora en el tiempo y un menor esfuerzo por parte de las personas que se encargan de la elaboración.

Para la creación de horarios se necesitan varias semanas si se realiza de forma manual, en vista de que se debe considerar todas las restricciones, evitando tareas que requieran de un mismo recurso o se asigne a un mismo periodo de tiempo. Se ha de contemplar también una distribución de forma equilibrada del horario de clase, de tal forma que no haya sobrecarga de clase y tareas en tiempos cortos, tanto para estudiantes como para profesores.

1.1 Justificación

En la facultad de Ingeniería Informática se realiza el proceso de creación cada inicio de curso y se toma aproximadamente un tiempo de una semana. Esto constituye una tarea tediosa que tiene como objetivo garantizar *tiempo* y *espacio* para impartir el currículo y armonizar el uso de los recursos del centro. Tiempo, dado que cada asignatura debe impartir un número determinado de créditos teóricos, prácticos y de laboratorio, y dichos créditos deben plasmarse temporalmente en una franja horaria limitada. Y espacio, porque la impartición de estos créditos requiere del uso de aulas, laboratorios y otro tipo de recursos educacionales.

Adicionalmente, los subdirectores y vicedecanos de los centros deben velar en el diseño del horario por el cumplimiento de ciertos criterios de calidad que faciliten la conclusión de los objetivos de nuestros estudiantes.

Si bien la primera tarea requiere de encontrar soluciones que cumplan con ajustadas restricciones, la segunda, añade una dificultad adicional al reducir el número de soluciones factibles. En este sentido, el uso de las tecnologías de la información y comunicación (TIC) puede liberar al decisor de efectuar cálculos de forma que garanticen que sus propuestas cumplen con las restricciones especificadas.

1.2 Alcance

Este trabajo final de grado, tiene como alcance la creación de un software para la generación automática de horarios, además de la realización de una interfaz web, donde se especificaran los recursos y se hará uso del programa para mostrar los resultados.

Con el software se desea cubrir los si siguientes puntos:

- Reducir el tiempo de generación de horarios y quitar los posibles conflictos que puede darse entre las asignaturas, profesores y aulas.
- Hacer una asignación de horarios donde cumpla las restricciones de la facultad. Que serían: Restricciones de cubrimiento, asignación, incompatibilidad y calidad.
- Mejorar la introducción y presentación de los datos ya que el programa esta desarrollado en C++.

1.3 Objetivos

El trabajo aquí descrito persigue el diseño de una aplicación informática interactiva para la planificación automática de los horarios y agendas de los centros. Esta aplicación, orientada a la ayuda a la decisión, tendrá en cuenta ciertos criterios de calidad en lo que al diseño del horario semestral se refiere. Por ejemplo, que el número máximo de horas que un alumno puede recibir al día de una misma asignatura esté acotado; que el conjunto de las horas de teoría precedan a las de problemas; etc. También se tendrá en cuenta la carga de trabajo sobre el alumno en el diseño de su agenda. Evitar un pico de carga en determinadas semanas, distribuyendo a lo largo del semestre la entrega de trabajos. Dado que en determinados centros el uso de los laboratorios y de determinadas aulas se comparte entre asignaturas e incluso entre titulaciones, el objeto de esta aplicación es compatibilizar el uso de estos recursos escasos. Una vez que la aplicación proporcione una posible solución, esta puede ser descartada o modificada dinámicamente de acuerdo a las pretensiones de los interesados.

Los objetivos de este trabajo son:

1. Determinar la carga de cada asignatura, así como el número de grupos de teoría, problemas y prácticas de laboratorio, haciendo hincapié en las fechas de entrega de trabajos.
2. Determinar los criterios de calidad que deben tenerse en cuenta a la hora de diseñar los horarios de los centros y agendas del alumnado.
3. Elaborar un modelo matemático que describa el problema a abordar teniendo en cuenta los criterios de calidad y el uso restringido de ciertas aulas y laboratorios.
4. Implementar una aplicación interactiva basada en el modelo matemático que permita de forma flexible la introducción y modificación de datos de entrada, así como el descarte de soluciones propuestas.

Los aspectos novedosos del proyecto comienzan con la implantación de criterios de calidad en el diseño de los horarios y establecimiento de la carga de trabajo del alumnado. Hasta el momento, la aplicación de estos criterios, al menos en el centro piloto, se desconoce o es obviada.

Por otro lado, el uso de una herramienta informática que genere posibles horarios cumpliendo las restricciones de cada una de las asignaturas junto con las restricciones asociadas a los criterios de calidad es otro de los aspectos novedosos del proyecto.

Cabe destacar que aunque el proyecto va a ser aplicado al Grado en Ingeniería Informática de la Escuela Técnica Superior de Ingeniería Informática, se trata de un trabajo extrapolable a todos los centros de la universidad. Se trata de una herramienta universal cuya versatilidad viene dada por la configuración de la misma. Además, la herramienta informática puede adaptarse a la inclusión de nuevos criterios de calidad, siempre y cuando se puedan representar como operaciones lineales sobre el horario.

2. Estado del arte

2.1 Antecedentes Software

Para la elaboración de un software basado en la creación de horarios mediante un modelo matemático, que va a dar una solución optima a un problema de asignación de recursos. Es necesario investigar la información de este tema, las técnicas y las variables que le afectan.

Para ello se ha hecho una búsqueda de las webs y aplicaciones similares a la que se ha desarrollado. Se ha encontrado bastante información y aplicaciones muy interesantes, de escritorio, webs y algoritmos aplicados a estas herramientas.

Aplicaciones de Escritorio

FET [1]: Es un programa evolutivo (usa algoritmos genéticos) para crear automáticamente horarios para un colegio, instituto o universidad. Permite la entrada de datos en formato XML y a partir de la entrada la generación automática, semi-automática o manual de los horarios.

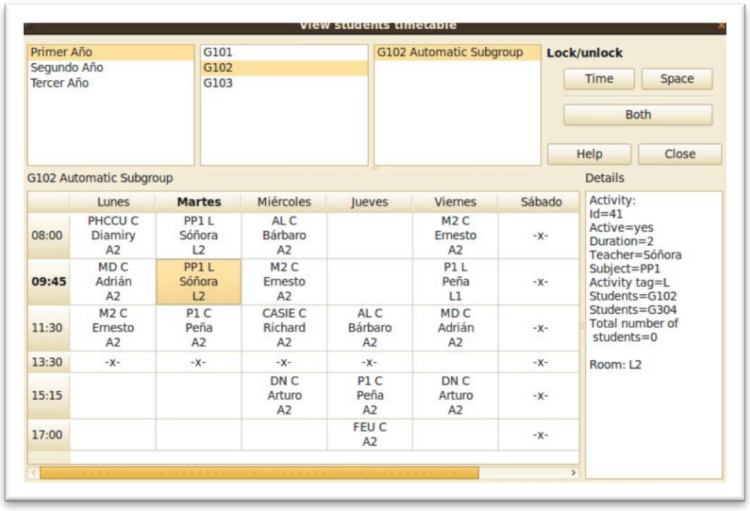

Imagen 1. horario del aplicación

Vista del estudiante en FET.

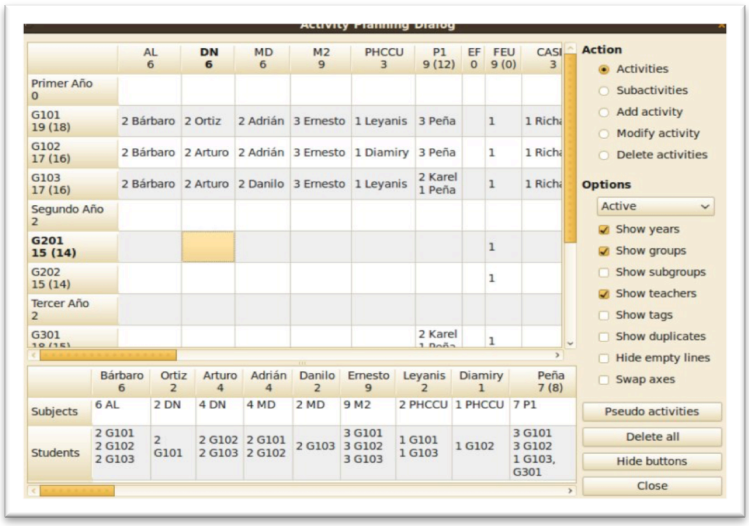

Imagen 2. Vista de la planificación de la actividad FET.

Tablix [2]: Es un generador de horarios para centros educativos. Se introducen los diferentes datos (profesorado, aulas, asignaturas, restricciones,...) y genera horarios en formato XML o en HTML (perfecto para colgarlos en internet). Tablix funciona con línea de comandos por lo que, para trabajar con él de forma amigable, es bueno instalar GTablix, su front-end.

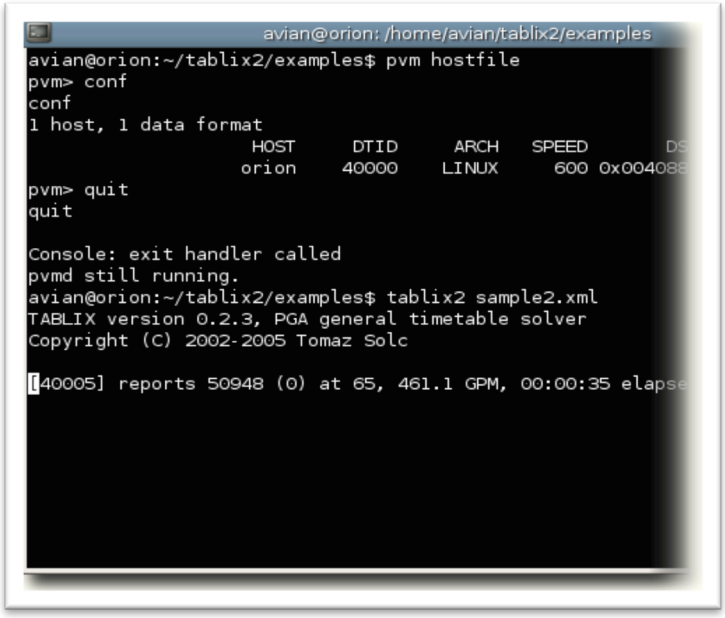

Imagen 3. Vista de la interfaz de Tablix.

Imagen 4. Vista del módulo G-Tablix para la aplicación Tablix.

aScTimeTables [3]: Es un generador de horarios, que mediante la introducción de requisitos los planificará de una forma equilibrada y con todos los criterios, para ganar la aprobación de alumnos y profesores.

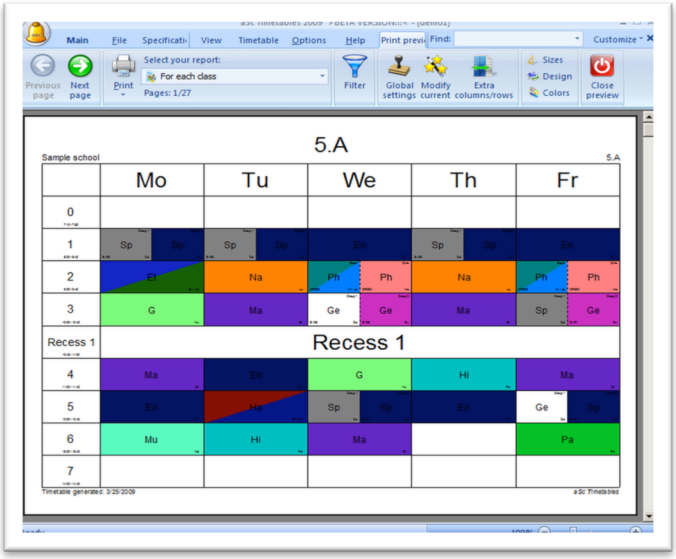

Imagen 5. Vista de ejemplo de generación de horarios en aSc TimeTables.

Aplicaciones Web

Timetable Web [4]: Herramienta online, para la generación automática de los horarios escolares. Todas las operaciones están en internet, a partir de la entrada de datos iniciales hasta la impresión final, por lo tanto, puede utilizar todos los computadores disponibles conectados a Internet.

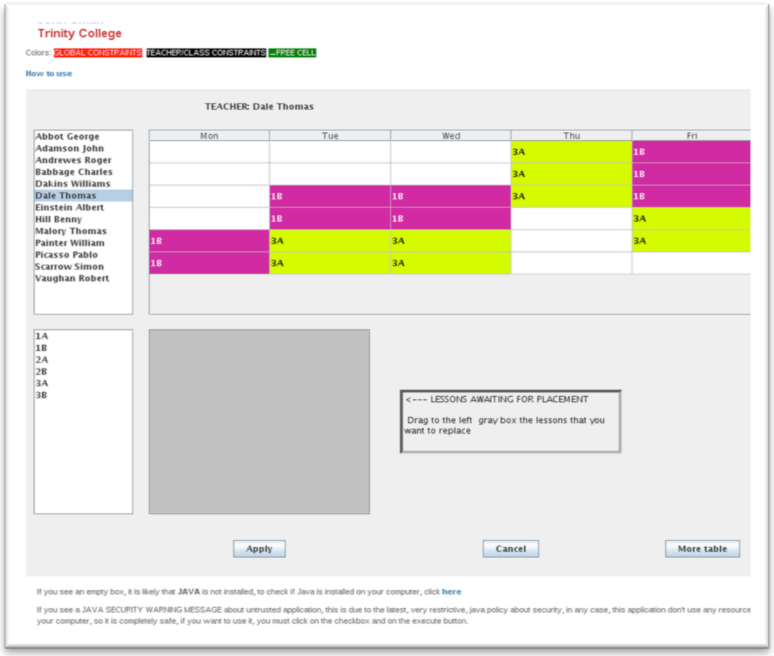

Imagen 6. Vista final del horario web con Timetable Web.

Parte II. Descripción del trabajo

3. Metodología

A continuación se detalla de forma pormenorizada el abordaje de cada uno de los objetivos a cubrir en este trabajo.

El primero de los objetivos, persigue determinar la carga de cada una de las asignaturas y los recursos educacionales necesarios para impartir cada una de las titulaciones involucradas. En este caso, hemos optado por elaborar un prototipo con el segundo cuatrimestre del primer curso del Grado en Ingeniería Informática. Se trata de cinco asignaturas de seis créditos ECTS cada una, las cuales serían:

- Algoritmos y Estructuras de Datos
- Principios de Computadores
- Optimización
- Sistemas Electrónicos Digitales
- Expresión Gráfica en Ingeniería

Los requerimientos específicos son los siguientes:

- La asignatura “Algoritmos y Estructura de Datos” requiere de dos horas de teoría a la semana en un aula que albergue la totalidad del grupo (un único grupo de teoría), una hora de problemas en un único grupo de problemas, y cinco horas de laboratorio en aulas de laboratorio de ordenadores con veinticinco puestos.
- La asignatura “Principios de Computadores” requiere de dos horas de teoría a la semana en un único grupo, una hora de problemas en dos grupos reducidos, y siete grupos en un laboratorio específico para la asignatura.
- La asignatura “Optimización” requiere de dos horas de teoría en un único grupo, dos horas de problemas, en sendos grupo, y cinco grupos de laboratorio en sala de ordenadores.
- La asignatura “Sistemas Electrónicos Digitales” requiere de dos horas de teoría para un único grupo de teoría, dos horas de problemas, para dos grupos, y siete grupos de laboratorio.
- Finalmente, la asignatura “Expresión Gráfica en Ingeniería” requiere de dos horas de teoría para un único grupo, una hora de problemas para un único grupo, y cuatro horas de laboratorio, para cuatro grupos.

Denominaremos *slot* a cada una de las entradas en el horario que requiera del uso de un recurso educacional (aula, laboratorio,…). Este estudio previo nos lleva a concluir que debemos establecer diez slots en aula de teoría, ocho slots en aulas de problemas, diez slots en salas de ordenadores, cuatro slots en sala de dibujo, y catorce slots para laboratorios de electrónica.

Asimismo, hemos establecido la posibilidad de simultanear algunos de los slots dentro del horario. Si bien aquellos slots referidos a las horas de teoría requieren de la presencia de todo

el alumnado, en el caso de algunas asignaturas, los slots de problemas y laboratorio pueden solaparse en algunas circunstancias. Es éste el caso de los grupos de problemas P1 y P2, que pueden simultanearse en asignaturas como Principios de Computadores, Sistemas Electrónicos Digitales, y Optimización. Este aspecto requiere de un estudio más pormenorizado en los grupos de laboratorio. Hay que tener en cuenta que estos son más heterogéneos, dado que hay asignaturas que requieren de cinco, otras de siete, y una de cuatro.

El segundo objetivo, persigue determinar aquellos criterios de calidad que deban ser incorporados en el diseño automático del horario. Entre ellos hemos incluido algunos de los considerados en Sancho (2001):

- Aprovechar las primeras horas de la mañana para trabajar las asignaturas más complejas que constituyen una base para el desarrollo del resto de asignaturas y que son vertebradoras del perfil profesional de los estudiantes. Habitualmente este tipo de asignaturas tienen una mayor carga horaria, por lo que es recomendable distribuirlas en las primeras franjas horarias a lo largo de toda la semana.
- Es recomendable que se dediquen al menos dos horas seguidas a la misma asignatura (una hora es poco y tres horas es mucho). Como recomendación al profesorado, indicar que no es conveniente hacer dos horas seguidas de exposición teórica sino que es aconsejable combinarla con la hora práctica.
- Las asignaturas específicas deberán distribuirse a lo largo de la semana, trabajándose una o dos cada día de la semana, sin superar las dos horas seguidas. Por otro lado, es recomendable que exista coordinación entre el profesorado que imparte clases al mismo grupo de estudiantes, con el objeto de conocer la evolución a lo largo del curso, detectar deficiencias y necesidades de refuerzo. Estos procedimientos están vinculados con el Plan de Orientación y Acción Tutorial.
- Potenciar el trabajo por Proyectos y la integración de asignaturas. El aprendizaje de los estudiantes no debe ser la suma de distintas asignaturas, sino que todos los aspectos trabajados deben interconectarse e integrarse (a modo de ejemplo, deberá funcionar como el trabajo fin de grado).

La elaboración de un modelo matemático que formule el problema de optimización asociado al diseño del horario, es el tercero de los objetivos. Con este fin debe establecerse cuáles son los datos que van a servir como entrada a este modelo:

- Asignatura. Cada asignatura está caracterizada por el número de créditos de teoría, problemas y prácticas de laboratorio. Además, cada asignatura debe tener un número de grupos de Teoría, grupos de problemas y grupos de laboratorio. Cada grupo, asimismo, tendrá asociado un recurso educacional, fundamentalmente aula de determinadas características (aforo). Con estos datos sabremos la cantidad de slots que necesitará cada asignatura dentro del horario. La asignatura vendrá caracterizada por un identificador único.
- Curso. Un curso se refiere al conjunto de asignaturas a impartir en un año determinado del grado. Se trata de una lista de asignaturas. El curso también tendrá asociado un horario, determinando los slots disponibles para su impartición.
- Incompatibilidad entre grupos. Hemos mencionado anteriormente que puede darse el caso de que dos asignaturas impartidas en el mismo curso requieran un número

diferente de grupos. Por tanto hay que establecer las incompatibilidades entre ellos. Es decir, establecer si dos grupos no pueden impartirse en el mismo slot por tener alumnos comunes.

Por tanto el modelo que refleje el problema de horarios debería contener las siguientes restricciones:

- Restricciones de cubrimiento. Es decir restricciones que fuercen que cada una de las horas de cada uno de los grupos de las asignaturas debe impartirse. No obstante, la impartición debe estar sujeta a los slots establecidos en el horario.
- Restricciones de asignación. Cada grupo debe asignarse a un recurso educacional y a un slot. Con estas restricciones se vigila por el uso adecuado de las aulas y recursos.
- Restricciones de incompatibilidad. Se refiere a aquellas restricciones que impiden que grupos conteniendo alumnos en común se impartan simultáneamente en diferentes aulas.
- Restricciones de calidad. Nos referimos a aquellas restricciones que recogen los criterios de calidad antes mencionados.

4. Tecnologías aplicadas

A continuación se va a desarrollar una descripción de las herramientas que van a ser usadas durante el desarrollo de esta aplicación.

4.1 Lenguaje de programación: C++

Para el desarrollo de este proyecto se ha decidido utilizar C++, debido al grado de soltura y la alta experiencia de la que se dispone. Otra ventaja que hizo inclinar la balanza, es la gran cantidad de información que podemos encontrar sobre este lenguaje y el poder utilizar los módulos ya desarrollados por otros programadores.

C++ es muy versátil, potente y de propósito general. Es un lenguaje de programación orientado a objetos y permite la programación estructurada, al existir compiladores de C++ para diferentes sistemas operativos, también le otorga portabilidad a los programas desarrollados.

4.2 Servicio web: GSOAP

Es un conjunto de herramientas de desarrollo software para C y C++ que permite compartir recursos computacionales y de información con las aplicaciones desarrolladas, el envío de los datos se realiza mediante un esquema SOAP/XML.

Las características principales de GSOAP son:

- **Rutinas pre-compiladas:** Las funciones de serialización están optimizadas y pre-compiladas.

- **Soporte de tipos nativos:** Las funciones pre-compiladas serializan y deserializan los tipos de datos nativos de C/C++, de esta forma se puede optimizar la colocación de datos en memoria para reducir accesos.
- **Minimización de las operaciones con memoria:** Evita copiar repetidas veces los datos.
- **Minimización del uso de la memoria:** Los ejecutables de un servidor o clientes suelen ser pequeños. Esto permite implementar clientes y servidores en sistemas empotrados muy pequeños.
- **Procesamiento eficiente del XML:** Esta librería incluye un procesador de XML muy eficiente y modificado para procesar este archivo sin tener que mantenerlo todo en memoria.
- **Integración de aplicaciones:** El uso de funciones de transformación para tipos de datos nativos de C/C++ y del usuario permite incluir aplicaciones desarrolladas en C/C++ en clientes y servicios.
- **Independencia de la plataforma:** gSOAP genera código en C/C++ independiente de la plataforma, por lo que puede obtener ejecutables para cualquier plataforma donde exista un compilador de C/C++.

4.3 Framework: Laravel

Es un poderoso framework de código abierto que se utiliza para desarrollar aplicaciones y servicios webs con el lenguaje de programación PHP 5. Este propone una forma de realizar aplicaciones web de un modo mucho más ágil. En Laravel se puede utilizar el patrón MVC (Modelo-Vista-Controlador).

Desarrollar aplicaciones en Laravel es de lo más fácil debido a su expresiva sintaxis, sus generadores de código y su ORM que se encarga de mapear datos entre el sistema de tipos utilizados en el lenguaje y la base de datos.

4.4 Base de datos MySQL

Es un sistema de administración y gestión de las base de datos relacionales, mutihilo y multiusuario. MySQL utiliza múltiples tablas para almacenar y organizar la información.

Este tipo de base de datos es muy utilizado en aplicaciones web, su popularidad esta ligada al lenguaje de programación PHP y a menudo aparece en combinación.

MySQL es un sistema muy rápido en cuanto a lectura, pero dispone de problemas de integridad en entornos de alta concurrencia o múltiples accesos, pero como nuestra aplicación va a trabajar con baja concurrencia porque la cantidad de usuarios no va a ser muy elevada, esta será la mejor opción por la que podremos optar, además tiene una licencia GNU, que garantiza la libertad de usar y modificar el código.

4.5 Interfaces: HTML 5, CSS 3, JavaScript y Bootstrap

En la actualidad el diseño web es muy importante, es una disciplina que evoluciona muy rápido y va acorde a las tendencias. A la hora de desarrollar la aplicación se ha utilizado las últimas tecnologías en cuanto al diseño web:

- El leguaje de programación **HTML (Hypertext Markup Language)**, describe el formato de las webs. Es un estándar que sirve de referencia para la elaboración de páginas, donde se define una estructura básica y un código para la definición del contenido como texto, imágenes, títulos, párrafos, etc.
- El lenguaje de programación **CSS (cascading style sheets)**, permiten definir las reglas y estilos de representación en documentos estructurados, escritos en HTML o XML. La principal idea fue la separación de la estructura y su representación.
- **JavaScript**, es un lenguaje de programación interpretado, se define como orientado a objetos, basado en prototipos, débilmente tipado y dinámico. Al ser un lenguaje interactivo, JavaScript permite realizar cambios al estilo original de la página y además sin que sea necesario volver a cargar la página del servidor.

  Este leguaje proporciona una gran cantidad de bibliotecas, la más que se utilizará en el proyecto es **jQuery**, ya que ayuda a manejar eventos, desarrollar animaciones, manipular los elementos de un documento HTML, etc.
- **Bootstrap**, es un framework que permite crear interfaces webs utilizando las tecnologías anteriormente mencionadas. Este es muy popular en internet, puesto que ha sido creado por los ingenieros de Twitter y actualmente lo utilizan en su aplicación.

  Ofrece interfaces webs adaptativas para los distintos dispositivos móviles, sus diseños son simples, limpios e intuitivos, por lo que le da agilidad a la hora de cargar las páginas.

5. Diseño

El diseño que se pretende crear para esta herramienta tiene que ofrecer una interfaz limpia adaptadas a las tecnologías webs, proporcionando facilidad y usabilidad, para convertir, un trabajo que antes era complejo en uno menos laborioso, centrándose en que la aplicación tenga un uso universal.

La otra parte del diseño serían las bases de datos, estas son muy importantes y debemos de asegurarnos de que están bien diseñadas para que tengan eficiencia y se puedan seguir utilizando durante mucho tiempo.

5.1 Diseño de la base de datos

La parte esencial del cualquier sistema es la información que proporciona los datos, y que posteriormente se van a almacenar. El resultado que la aplicación nos va

a proporcionar es información valiosa con la que podemos tomar decisiones y con la que se establecerá el horario óptimo, pero no antes sin haber realizado un análisis del resultado. Es por eso que los datos deberán estar almacenados de una forma legible, fácil y accesible.

Base de datos relacional

Para este proyecto se utilizara una base de datos relacional, esto quiere decir que la información almacenada va estar relacionada y que se va a agrupar o estructurar de acuerdo a un modelo de datos.

En este modelo, los datos están almacenados en relaciones, y cada relación es un conjunto de datos. La información puede ser recuperada o almacenada por medio de consultas lógicas.

La base de datos relacional que vamos a utilizar para el proyecto es la siguiente:

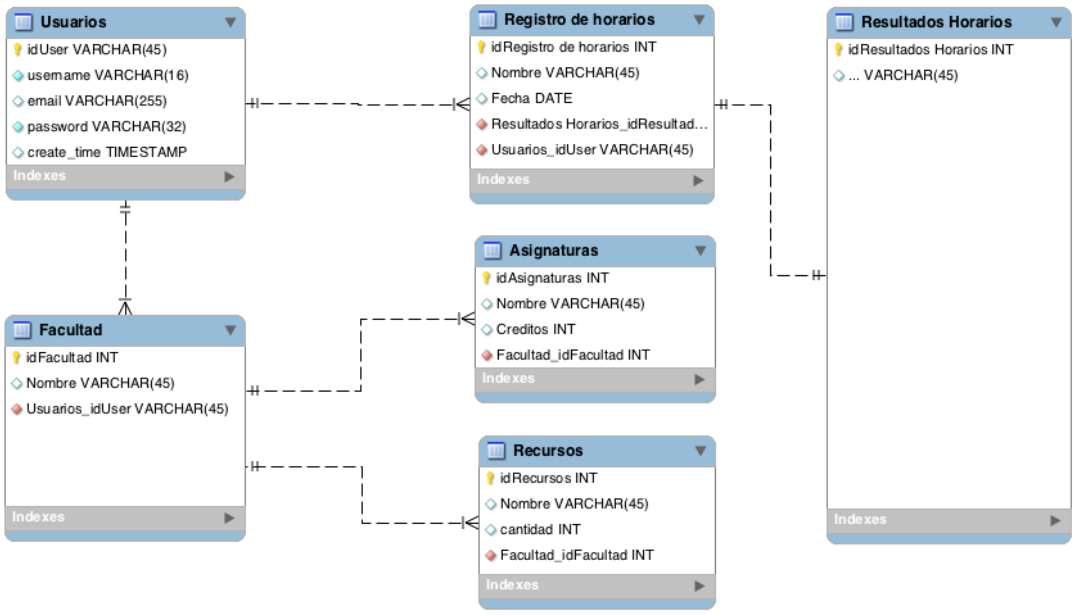

Imagen 7. Diseño de la base de datos “horarios”

Esta base de datos tiene por nombre “horarios”, como se puede observar esta compuesta por varias tablas que disponen de una relación entre ellas. A continuación, se va a especificar cada una de las tablas y la relación que se establece entre ellas:

- **Usuarios:** Tabla donde se almacena un usuario. Los campos que la componen son: ID (idUser), nombre de usuario (username), email, contraseña encriptada (password), fecha de creación (create_time) y rol. Esta tabla, establece una relación “Uno a

muchos", donde la clave primaria es idUser y la clave ajena Usuarios_idUser. ID identificativo (idRegistro)

- **Registro de Horarios:** Tabla donde se va a almacenar cada una de las salidas hechas por la aplicación. Los campos que la componen son: ID (idRegistro de horarios), nombre, fecha y Resultados Horarios_idResultados, esta última establece una relación "Uno a uno" con la tabla Resultado Horarios.
- **Resultados Horarios:** Tabla donde se almacena la salida que genera el programa.
- **Facultad:** Tabla donde se almacena la facultad a la que va dirigido el horario. Aunque el programa este hecho específicamente para la facultad de ingeniería informática de La Laguna, se ha incluido para posteriores modificaciones. Los campos que la componen son: ID (idFacultad) y el nombre. Establece una relación "Uno a muchos" con la tabla usuarios, donde un usuario puede pertenecer a varias facultades y establecer los horarios correspondientes.
- **Asignaturas:** Tabla donde se almacenan las asignaturas asociadas a cada facultad. Los campos que la componen son: ID (isAsignaturas), nombre y créditos. Establecen una relación "Uno a muchos" con la facultad ya que cada facultad dispone de varias asignaturas.
- **Recursos:** Tabla donde se almacenan los recursos específicos para cada facultad. En esta se especificarán el nombre y la cantidad de estos recursos (cantidad), todo esto asociado a un ID (idRecursos). Establece una relación "Uno a muchos", donde se le asignarán varios recursos a una facultad.

5.2 Prototipado rápido

El prototipado rápido es algo muy habitual en equipos de desarrollo web, con este trabajo se ha intentado utilizar las herramientas necesarias para que se realice de la mejor forma.

Con el programa "Balsamiq Mockups" podemos dibujar maquetas de las pantallas de un navegador web y poder plasmar sus elementos. Así podemos transmitir la idea que queremos obtener, para que a la hora de programarla resulte más fácil, ya que no se tiene que pensar en el diseño.

A continuación, se presentan las principales interfaces con las que los usuarios se van a encontrar en la plataforma de gestión de horarios:

1. **Identificación o login:** La primera interfaz con la que el usuario se encontrara es un identificador o login, donde se tendrá que poner los datos que lo identifican, es decir, una contraseña y un email que el administrador del portal le proporcionara. Una característica de este formulario de entrada es que se tiene que especificar el rol con el que actúa en la web, puede ser administrador o usuario normal.

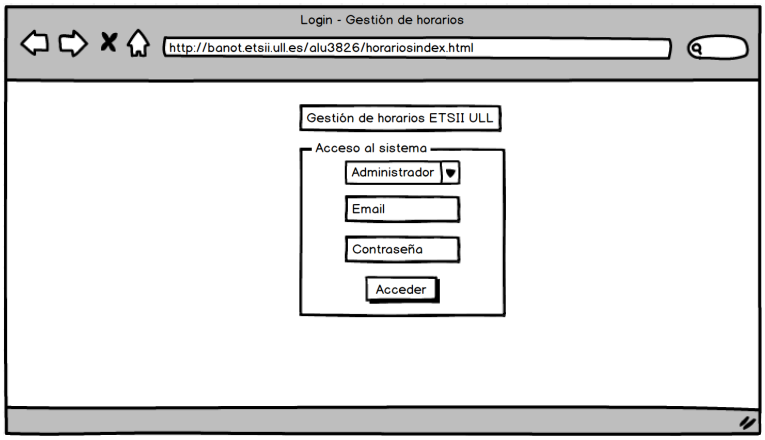

Imagen 8. Prototipo de login de usuario

2. **Formulario para ejecución de la aplicación:** En esta sección el usuario establecerá las variables y restricciones del problema, especificando un nombre, facultad a la que se realizara el horario, asignaturas correspondiente a la facultad y por ultimo especificación de los recursos, como la cantidad de grupos de teoría, de prácticas y de laboratorio.

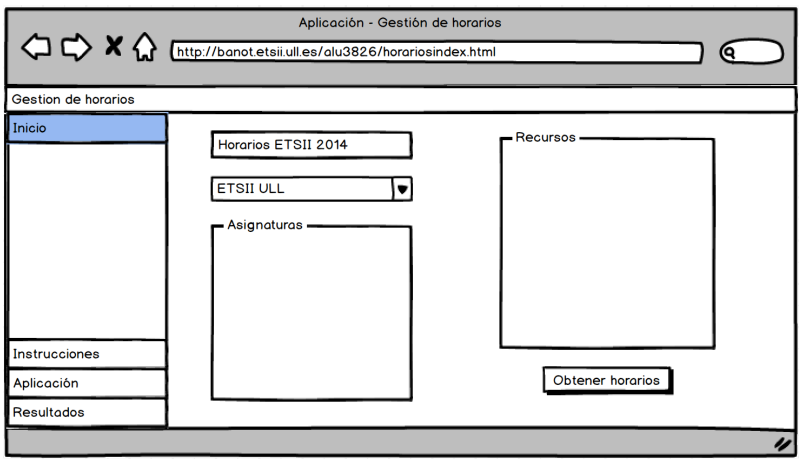

Imagen 9. Prototipo de formulario para obtención de horarios

3. **Resultado de la aplicación:** Aquí se obtendrán los resultados de la aplicación, donde se muestra un horario de lunes a viernes con las correspondientes horas,

especificando en cada casilla el nombre de la asignatura en siglas y el recurso que le corresponde para esa hora.

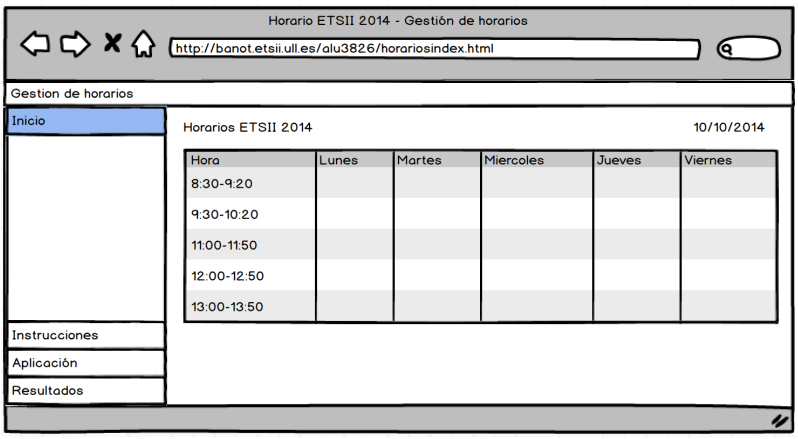

Imagen 10. Prototipo de obtención de resultados de los horarios

4. **Registro de horarios:** Se muestra un registro con acceso a la base de datos donde se pueden ver los horarios realizados con anterioridad, mostrando su número identificativo, su fecha de creación y la opción de visualizado desde la pagina web o exportarlo a PDF.

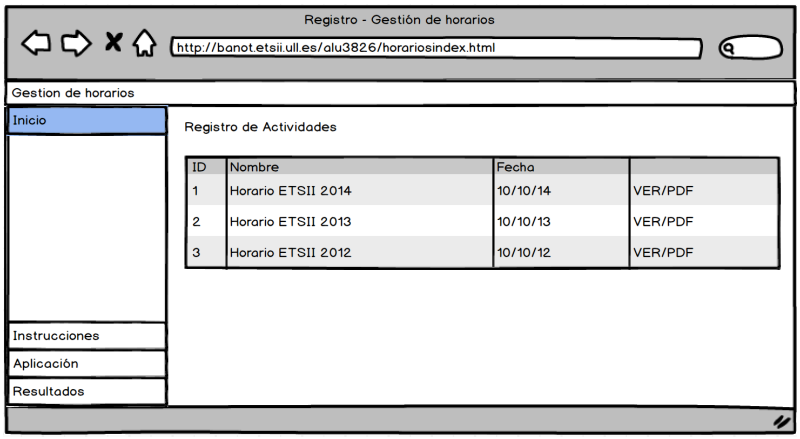

Imagen 11. Prototipo de los registro de horarios

6. Desarrollo

En esta parte del proyecto se va a detallar el diagrama de red, el código utilizado para la realización de la interfaz y comunicación entre el servidor web con el servidor de aplicaciones.

Configuración del diagrama de red

A continuación, se va a representar los componentes que hacen que la aplicación funcione correctamente, pero se va especificar a un alto nivel. Esta representación, la podemos dividir en dos partes, las que hacen las peticiones al servidor web, que la conforma a los usuarios o clientes, y la otra parte formada por alojamiento donde se encuentra nuestra aplicación, una base de datos y un servicio de ejecución de aplicaciones.

A continuación, se muestra el diagrama de red que conforma el funcionamiento del servicio:

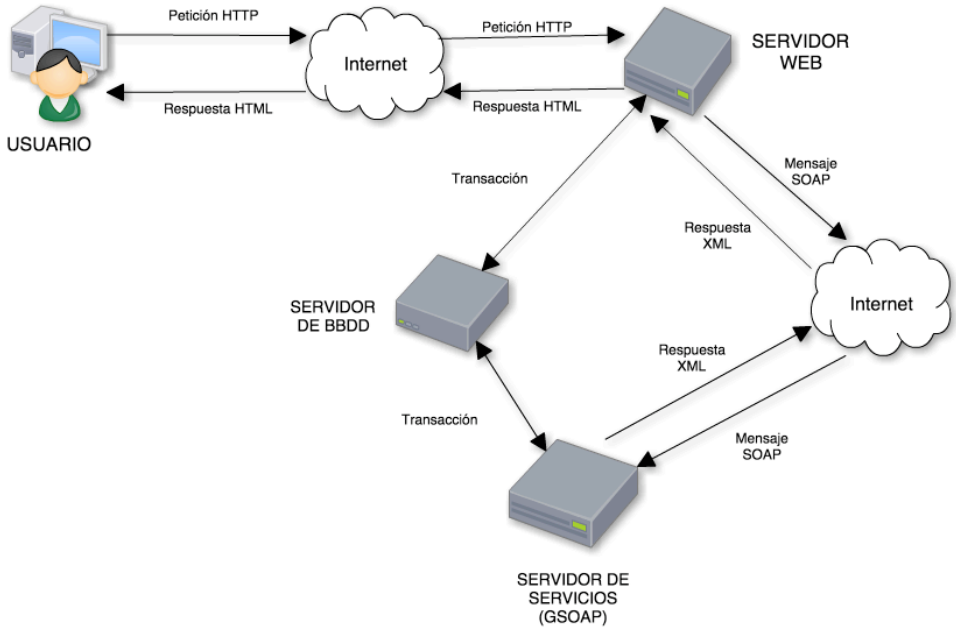

Imagen 12. Diagrama de uso de servicios web

Como se puede observar, toda la aplicación depende del servidor web. Los mensajes, respuestas y peticiones van desde el cliente al servidor web, hasta el servidor de servicios y de vuelta al cliente. La respuesta que proporciona el servicio web son páginas completas.

Configuración del servidor de servicios GSOAP

Lo siguiente que vamos a tratar es la configuración del “Servidor de servicios (GSOAP)”, así que se describirán los pasos necesarios para crear un servicio SOAP.

Lo primero de todo y necesario es descargarse el paquete gsoap-linux-2.7.tgz, seguidamente se activara un puerto al servidor por el que estará escuchando para que acepté conexiones y

así reciba o envíe cadenas a los clientes. Los pasos siguientes, se incluirán en el anexo **"Configuración del servidor de servicios GSOAP"**.

Lógica de la plataforma web con Laravel

En cuanto a la lógica de desarrollo de la aplicación web, se ha utilizado Laravel usando el paradigma de Modelo-Vista-Controlador.

En el anexo **"Configuración de Laravel"** se incluye toda la información necesaria para la instalación de Laravel en nuestro ordenador, así como la explicación de la estructura que compone este framework.

Diseños con Bootstrap

El primer prototipo que se ha desarrollado, ha sido guiándose por los primeros esbozos de la aplicación "Balsamiq Mockups". A continuación, se contempla el módulo de acceso o login que identifica al usuario y le otorga privilegios según el rol.

Imagen 13. Prototipo de pantalla de login

Tras acceder el usuario dispone de una bienvenida, donde se describe el contenido de la aplicación. No obstante, donde realmente se hace uso de esta herramienta, es en la sección "Aplicación" de la parte izquierda donde aparecerán los formularios que se han de rellenar para que se ejecute correctamente.

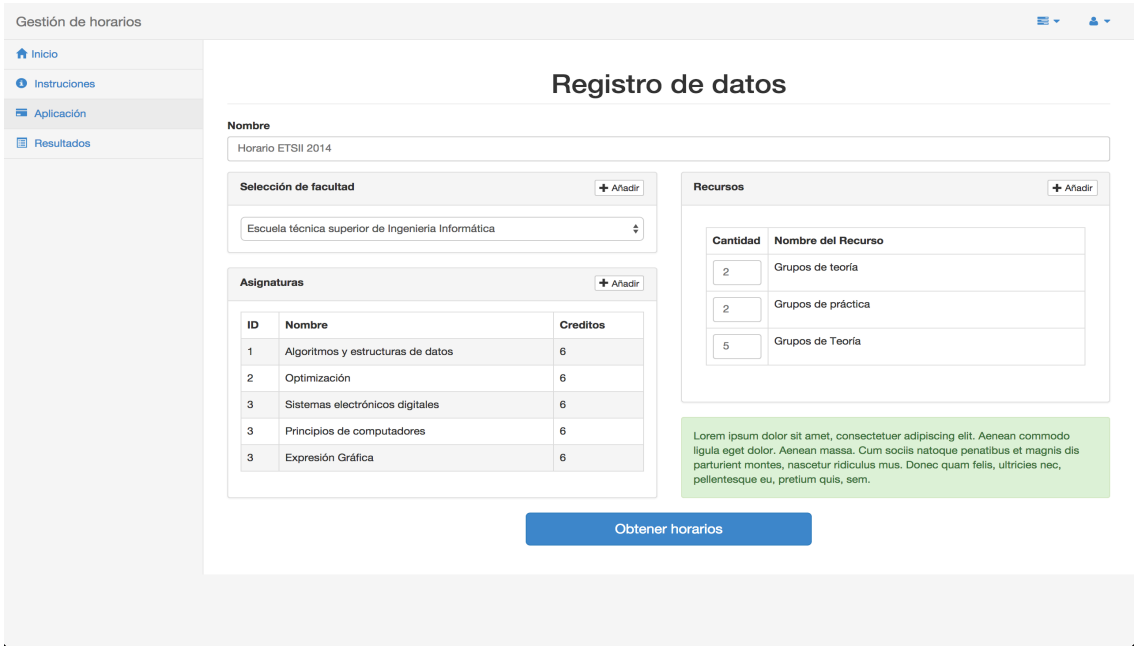

Imagen 14. Prototipo de formulario de la aplicación

Al enviar estos datos, al lado del servidor y utilizando el servicio web GSOAP, se ejecutará el programa en C++ con los datos que se han rellenado y se obtendrá una salida en un formato XML que será interpretado y almacenado por la aplicación.

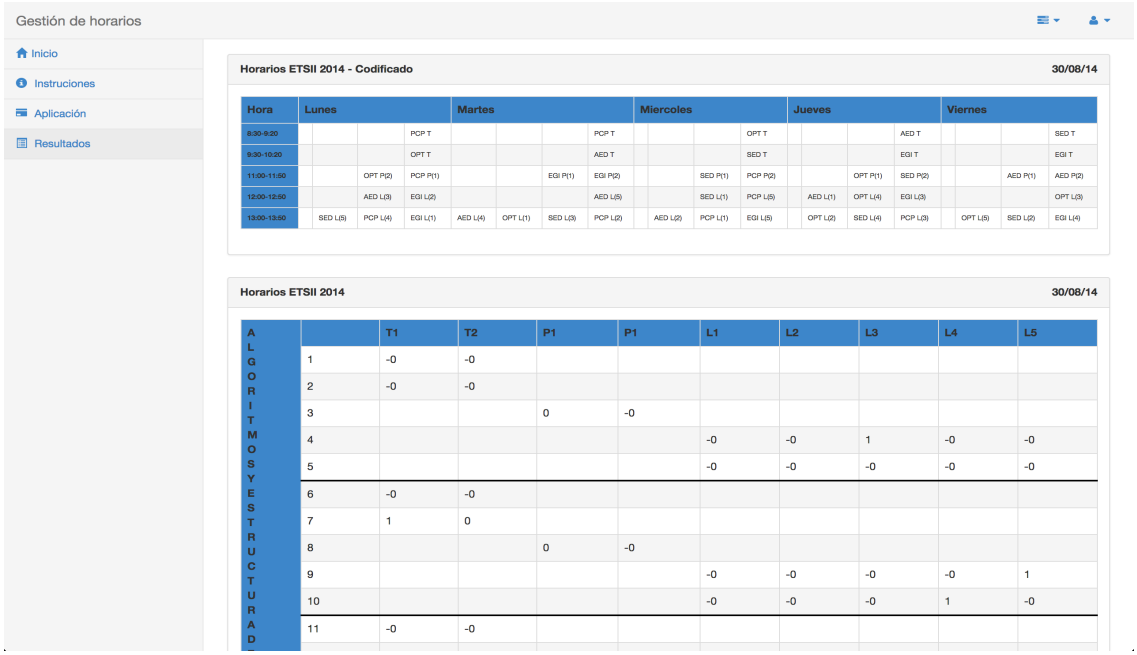

Imagen 15. Prototipo de horario completo codificado

ALGORITMOS Y ESTRUCTURA DE DATOS		T1	T2	P1	P1	L1	L2	L3	L4	L5
	1	-0	-0							
	2	-0	-0							
	3			0	-0					
	4					-0	-0	1	-0	-0
	5					-0	-0	-0	-0	-0
	6	-0	-0							
	7	1	0							
	8			0	-0					
	9					-0	-0	-0	-0	1
	10					-0	-0	-0	1	-0
	11	-0	-0							
	12	-0	-0							
	13			0	-0					
	14					-0	-0	-0	-0	-0
	15					-0	1	-0	-0	-0
	16	-0	1							
	17	-0	-0							
	18			0	-0					
	19					1	-0	-0	-0	-0
	20					-0	-0	-0	-0	-0
	21	-0	-0							
	22	-0	-0							
	23			1	1					
	24					-0	-0	-0	-0	-0
	25					-0	-0	-0	-0	-0

Imagen 16 Prototipo de horario completo sin codificar

En esta parte, se muestra un registro de los horarios que se han generado. Este tendrá acceso a la base de datos MySQL, para mostrar los resultados de una forma ordenada con propiedades de visualización y exportación a pdf. Esta exportación se hace con una librería de JavaScript llamada jsPDF que crea archivos PDF donde no se necesita ningún script del lado del servidor, ya que los puede crear del lado del cliente.

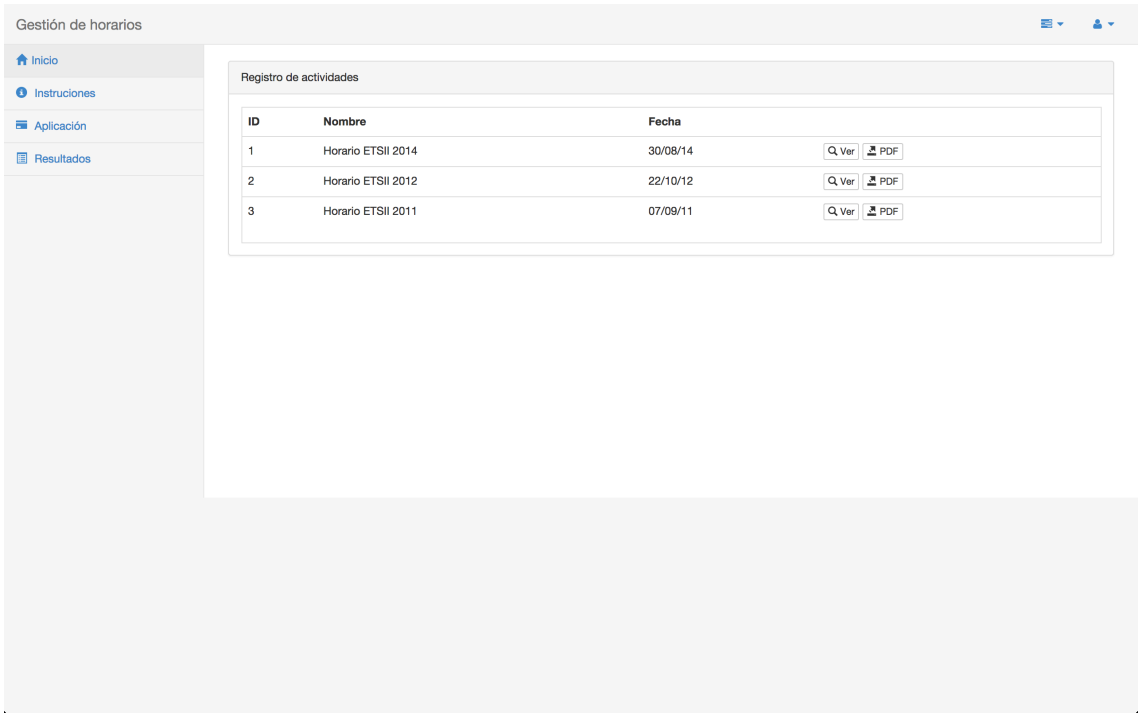

Imagen 17. Prototipo de registro de horarios

Parte III. Conclusiones y resultados

7. Conclusiones y resultados

7.1 Conclusiones

La creación de horarios siempre es un problema difícil que puede tardar varias semanas en resolverse, por eso, este proyecto tiene una gran importancia debido a que la automatización de los horarios ayuda a reducir el trabajo del docente.

La utilización de herramientas como estas, donde existe una ayuda para el diseño de horarios facilita el uso racional de los recursos y la inclusión de criterios de calidad que permitan al alumnado concluir sus objetivos. Con esa finalidad, se ha desarrollado una herramienta informática que optimiza el uso de recursos y proporciona horarios que satisfacen la docencia e incluyen algunos de los criterios de calidad. No obstante, en el momento actual del desarrollo, la aplicación ofrece una interfaz más amigable, donde el que la va a utilizar se pueda adaptar de una forma más rápida.

Esta primera versión, tiene como objetivo descubrir el potencial de este tipo de herramientas y otorgar una nueva interfaz adaptada a la web. Sin embargo, no se pretende una implantación generalizada como herramienta de ayuda a la decisión en los centros.

7.2 Resultados

En base a los datos y las restricciones anteriormente especificadas, hemos llevado a cabo la implementación de una aplicación informática que recoge dichas especificaciones y que proporciona soluciones al problema de la elaboración del horario.

Dicha elaboración ha utilizado el lenguaje de programación C++, y la biblioteca de optimización CPLEX 12.4. Las ilustraciones 15 y 16 muestra un resultado de la aplicación informática con una interfaz web desarrollada en Bootstrap. La lectura de los datos se envían a través de varios formularios rellenados por el usuario y donde se generarán varios ficheros de configuración que serán enviados a la aplicación, en un formato SOAP. Cada uno de los datos de entrada, son: características de las asignaturas, aulas, incompatibilidad entre grupo, etc.

El cómputo de cada propuesta de horario consume pocos segundos, para un problema que incluye un único curso de cinco asignaturas. La resolución de problemas más complejos, como puede ser los cuatro cursos del grado en horario de mañana consume aproximadamente siete minutos. El incremento del tiempo se justifica por el importante problema de decisión que subyace con un gran número y diversidad de asignaturas, que en algunos casos llegan a compartir horario.

7.3 Conclusions

Making schedules is always a difficult problem that can take several weeks to resolve it, therefore, this project is very important because automation helps to reduce the teachers' work.

The use of this kind of tool, helps in creating schedules' design facilitating the rational use of resources and the inclusion of quality criteria that enable students to complete their objectives. It is clear, therefore, that I developed a software tool that optimizes the use of resources and provides schedules that satisfy the teaching and include some of the quality criteria. Furthermore, at this stage of development, the application offers a friendly user interface where a user can adapt more quickly.

This first release, aims to uncover the potential of such tools and provide a new interface adapted to the web. However, is not intended the implementation as a general tool for decision support in schools.

8. Presupuesto

En este apartado se intentara explicar de manera detallada los gastos que tendrá el desarrollo e implantación de la aplicación para el "Diseño automático de horario y agenda sujeto a ciertos criterios de calidad para centros con recursos limitados de espacio".

Dado que este proyecto requiere alrededor de dos meses para su puesta a punto, se detallará el siguiente cuadro con el precio por unidad de cada elemento, así como el tiempo que se necesita de cada uno.

A continuación, se detalla dicho presupuesto en diferentes secciones:

8.1 Presupuesto de mobiliario

Para la realización de este desarrollo software y su puesta en marcha se necesitan de cierto mobiliario imprescindible.

Como es lógico, necesitamos la herramienta fundamental para este trabajo que sería un ordenador. Además, se necesita de otros accesorios como son : un ratón, una silla y una mesa.

Concepto	Unidad	Precio/Unidad	Total
Mesa	1	50 €	110 €
Silla	1	30 €	47 €
Macbook Pro Retina 13'	1	1229 €	1229 €
Magic Mouse	1	69,90 €	69,90 €

Tabla 1. Presupuesto en mobiliario

8.2 Presupuesto en suministro y similares

En esta partida, hablaremos de los gastos necesarios para la marcha del proyecto, como son gastos de alquiler, electricidad, internet, entre otros.

Por un lado, como es un proyecto que se realiza bajo la supervisión de una sola persona la cual no tiene medios económicos suficientes para poder pagar un alquiler “normal”, hemos recurrido a un cowworking, el cual minimiza los costes de alquiler al trabajar varias empresas en un mismo local, y como cabe esperar los gastos de electricidad y agua también son menores que en alquiler normal al ser compartidos.

Por otro lado, para que el desarrollo de dicha aplicación se lleve a cabo de manera eficaz y eficiente se necesita de una conexión a internet , puesto que necesitamos buscar mucha información registrada en los distintos blogs presentes en la web.

Asimismo, en esta partida el elemento más importante sería la contratación de un servicio de hospedaje o hosting. Este es necesario para abaratar los costes que supondría una instalación de servidores con todo el cableado y software necesario para poner en marcha toda la red del servicio, ya que con el hosting tendríamos todo esto con tan solo contratarlo.

Concepto	Meses	Euros/Mes	Total
Alquiler	3	200	600
Electricidad	3	15	45
Internet	3	10	30
Hosting	3	162	486

Tabla 2. Presupuesto en suministro y similares

8.3 Presupuesto en salario

Esta parte del presupuesto estará compuesto por los sueldos y salarios de aquellas personas que van a desarrollar la aplicación. En nuestro caso es un titulado en Ingeniería Informática por la Universidad de La Laguna y cobrará a razón de las horas dedicadas al proyecto, que en un principio se fijaron a 320 horas.

Empleado	Categoría	Horas	Euros/Hora	Total
Javier Mena Mena	Ingeniero Informático	320	15	4800

Tabla 3. Presupuesto en salario

8.4 Presupuesto final

En este último apartado se detallara el gasto total que va a suponer la puesta en marcha y finalización de este proyecto. Se hará la suma del "presupuesto en mobiliario" con el "presupuesto en suministro y similares" y el "presupuesto en salario". Con ello obtendremos la totalidad en euros de la aplicación.

Descripción	Total (Euros)
Presupuesto en mobiliario	1378,9
Presupuesto en suministro y similares	1161
Presupuesto en salarios	4800
TOTAL	7339,9

Tabla 4. Presupuesto final

Por lo tanto, el proyecto tendrá un gasto total de **7339,9 €**

Parte IV. Bibliografía/Referencias/Anexos

9. Bibliografías

1. **Aplicación C++**
    a. http://www.ijcaonline.org/volume12/number5/pxc3872083.pdf
    b. http://orbit.dtu.dk/files/60366101/A_Comprehensive_Study.pdf
    c. http://www.patatconference.org/patat2004/proceedings/305.pdf
    d. Sancho Gil, J. M. (2001) Docencia e investigación en la univer- sidad: una profesión, dos mundos Educar, 28, 41-60
    e. Stroustrup, Bjarne (1997) The C++ Programming Language (3a edición) Addison-Wesley
2. **Bootstrap**
    a. http://getbootstrap.com/
    b. http://startbootstrap.com/
    c. https://wrapbootstrap.com/themes
    d. https://wrapbootstrap.com/
    e. http://themeforest.net/?osr=tn&_ga=1.240456643.1154345858.1395665212
3. **HTML**
    a. http://www.w3schools.com/html/
4. **CSS**
    a. http://www.w3schools.com/css/
5. **GSOAP**
    a. http://www.cs.fsu.edu/~engelen/soap.html
    b. http://mercurio.ugr.es/pedro/docencia/irhc/4_gsoap.ppt.pdf
6. **Laravel**
    a. https://www.udemy.com/aprende-a-programar-aplicaciones-web-con-laravel-4/?dtcode=tjv8GUF1qMu6
    b. http://codehero.co/codehero-colaravel-4-desde-cero-proyecto/

10. Referencias

1. http://lalescu.ro/liviu/fet
2. http://www.tablix.org
3. http://www.asctimetables.com
4. http://www.timetableweb.com
5. http://mercurio.ugr.es/pedro/docencia/irhc/irhc/documentacion_gsoap.pdf
6. http://codehero.co/laravel-4-desde-cero-instalacion-configuracion/
7. http://codehero.co/codehero-colaravel-4-desde-cero-proyecto/
8. http://parall.ax/products/jspdf

11. Anexos

11.1 Configuración del servidor de servicios GSOAP

En esta sección se describirán en detalle los pasos necesarios para crear un servicio SOAP sencillo y el cliente asociado. Se trata del ejemplo en el que basaremos las prácticas de esta parte del curso.

Para desarrollar programas que utilicen gSOAP para llevar a cabo las comunicaciones, debem os bajar el paquete gsoap-linux-2.7.tgz disponible en:

http://sourceforge.net/projects/gsoap2/files/

Debemos descomprimirlo y entrar en el directorio que habrá creado:

tar xvfpz gsoap-linux-2.7.tgz

cd gsoap-linux-2.7

Como ejemplo para esta práctica desarrollaremos un ejemplo sencillo que constará de un servidor y varios clientes. El servidor estará escuchando cierto puerto por el que aceptará conexiones para recibir o enviar cadenas a los clientes.

Un cliente puede conectarse para enviarle una cadena o bien para solicitarla (en este caso, el servidor toma la más antigua de las que tiene almacenadas y la envía al cliente).

Para trabajar crearemos una carpeta nueva llamada **ejemplo_basico.**

Una vez que sabemos qué funcionalidad debe tener el servidor, debemos crear el fichero ***funciones.h*** que especificará las funciones a las que se puede acceder, y los parámetros que se esperan. El contenido del fichero será similar al siguiente:

```
//gsoap ns service name: funciones

//gsoap ns service style: rpc

//gsoap ns service encoding: encoded

//gsoap ns service namespace: -

//gsoap ns service location: -

//gsoap ns schema namespace: urn:func

int ns__enviar (unsigned nodo, std::string cadena, std::string &resultado);

int ns__recibir(unsigned nodo, std::string &resultado);
```

Vemos que en este ejemplo sólo permitimos dos tipos de acceso al servidor: para enviar un nuevo valor y para recibirlo.

Una vez tenemos el fichero funciones.h debemos compilarlo para que gSOAP nos genere todos los ficheros necesarios para manejar las comunicaciones:

```
../bin/soapcpp2 funciones.h
** The gSOAP Stub and Skeleton Compiler for C and C++ 2.7.8a
** Copyright (C) 2000-2006, Robert van Engelen, Genivia Inc.
** All Rights Reserved. This product is provided "as is", without any warranty.
** The gSOAP compiler is released under one of the following three licenses:
** GPL, the gSOAP public license, or the commercial license by Genivia Inc.
Saving soapStub.h
Saving soapH.h
Saving soapC.cpp
Saving soapClient.cpp
Saving soapClientLib.cpp
Saving soapServer.cpp
Saving soapServerLib.cpp
Using ns service name: funciones
Using ns service style: rpc
Using ns service encoding: encoded
Using ns service location: -
Using ns schema namespace: urn:calc
Saving funciones.wsdl Web Service description
Compliance warning: operation 'ns__enviar' is not compliant with WS-I Basic Profile 1.0a,
reason: uses SOAP encoding
Compliance warning: operation 'ns__recibir' is not compliant with WS-I Basic Profile 1.0a,
reason: uses SOAP encoding
Saving soapfuncionesProxy.h client proxy
Saving soapfuncionesObject.h server object
Saving funciones.enviar.req.xml sample SOAP/XML request
Saving funciones.enviar.res.xml sample SOAP/XML response
Saving funciones.recibir.req.xml sample SOAP/XML request
```

Saving funciones.recibir.res.xml sample SOAP/XML response

Saving funciones.nsmap namespace mapping table

Saving ns.xsd XML schema

Compilation successful

El código del proceso cliente (fichero cliente.c) es bastante sencillo, y sólo debe tener en cuenta contactar con el servidor adecuado:

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <math.h>
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <sstream>
using namespace std;
#include "soapH.h"
#include "funciones.nsmap"
// poner la              IP:PUERTO    del servidor
const char server[] = "127.0.0.1:9999";
struct soap soap;
int main(int argc, char **argv){
 if (argc < 3){
cout << "Uso:   cliente  [enviar|recibir]  cadena  nodo \n";
return 1;
 }
 char *accion=argv[1];
 string cadena=argv[2];
 unsigned nodo = atoi(argv[3]);
 soap_init(&soap);
 string resultado;
 if(accion[0]=='e'){
   soap_call_ns__enviar(&soap, server, "", nodo, cadena, resultado);
```

```
    cout <<"-> enviar devuelve = "<<resultado<<endl;
  }else{
    soap_call_ns__recibir(&soap, server, "", nodo, resultado);
    cout <<"-> recibir devuelve = "<<resultado<<endl;
  }
  soap_destroy(&soap);
  soap_end(&soap);
  soap_done(&soap);
  return 0;
}
```

Para compilar el cliente.c debemos usar la siguiente orden:

```
g++ -Wall -O2 -I..  -o cliente cliente.c soapC.cpp soapClient.cpp ../stdsoap2.cpp
```

El proceso servidor (fichero servidor.c) constará de un bucle infinito en el que se aceptan conexiones SOAP y las sirve, utilizando las dos funciones creadas al principio (enviar o recibir).

El servidor tiene un vector de cadenas de texto donde se van almacenando las cadenas que los clientes envían, y de donde extrae la más antigua para enviarla a un cliente que lo solicite:

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <math.h>
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <sstream>
using namespace std;
#include "soapH.h"
#include "funciones.nsmap"
vector<string> almacen;
unsigned accede_nodo;
int main(int argc, char **argv){
  int m, s;
```

```
  struct soap soap;
  soap_init(&soap);
  //poner la IP  y PUERTO  del servidor
  m = soap_bind(&soap, "127.0.0.1" ,  9999 , 100);
  if (m < 0){
    soap_print_fault(&soap, stderr);
    exit(-1);
  }
  for ( ; ; ){
    s = soap_accept(&soap);
    if (s < 0){
      soap_print_fault(&soap, stderr);
      exit(-1);
    }else{
if (soap_serve(&soap) != SOAP_OK) soap_print_fault(&soap, stderr);
else cout << "servido  NODO="<<accede_nodo<<"  ELEMS="<<almacen.size()<<endl;
soap_destroy(&soap);
soap_end(&soap);
    }
  }
  return 0;
}
// el cliente envia al servidor => insertar en el vector
int ns__enviar(struct soap *soap, unsigned nodo, std::string cadena, std::string &resultado){
  accede_nodo = nodo;
  almacen.push_back( cadena );
  resultado = "enviar";
  return SOAP_OK;
}
// el cliente recibe del servidor => sacar del vector y devolverlo
int ns__recibir(struct soap *soap, unsigned nodo, std::string &resultado){
  accede_nodo = nodo;
  if( almacen.size() > 0 ){
```

```
    resultado = (string)(almacen[0]);
    almacen.erase( almacen.begin(), almacen.begin()+1 );
  }else{ resultado="<vacio>"; }
  return SOAP_OK;
}
```

Para compilar el servidor.c debemos usar la siguiente orden:

```
g++ -Wall -O2 -I..  -o servidor servidor.c soapC.cpp soapServer.cpp ../stdsoap2.cpp -lm
```

Ahora, en la máquina que hará de servidor ejecutamos el programa servidor y en una o varias máquinas ejecutamos el programa cliente

11.2 Configuración de Laravel

En esta parte se incluye la total configuración del framework de desarrollo Laravel, para poder iniciar un proyecto con esta herramienta sin que falte ninguna librería. También añadiré un videotutorial el cual se ha seguido y da mucha información.

- **El link correspondiente de configuración:** http://codehero.co/laravel-4-desde-cero-instalacion-configuracion/
- **Estructura de Laravel:** http://codehero.co/codehero-colaravel-4-desde-cero-proyecto/
- **Videotutorial:** https://www.udemy.com/aprende-a-programar-aplicaciones-web-con-laravel-4/?dtcode=tjv8GUF1qMu6