TRABAJO FINAL DE CARRERA

TÍTULO: Serveis col·laboratius d'agenda, contactes i tasques

AUTOR: Cristian Morón Espadas

DIRECTOR: Roc Meseguer Pallarès

FECHA: 21 de julio de 2006

Título: Serveis col·laboratius d'agenda, contactes i tasques

Autor: Cristian Morón Espadas

Director: Roc Meseguer Pallarès

Fecha: 21 de julio de 2006

Resumen

El proyecto consta del estudio tanto en el ámbito teórico como en el ámbito práctico de la necesidad, la evolución y la creación de un sistema de calendario. Un sistema de calendario debe permitir la organización temporal de diferentes usuarios de forma simultanea y desde diferentes organizaciones o empresas permitiendo automatizar la desagradecida tarea de la planificación.

En los últimos años, múltiples desarrolladores de software han trabajado en la creación de un sistema de calendario, con el objetivo principal de estandarizar un mecanismo común de datos y su acceso a éstos, a fin de conseguir una interoperabilidad real y efectiva.

El proyecto estudia los diferentes estándares acordados y analiza algunas de las aplicaciones creadas con más éxito en el mercado, extrayendo los puntos fuertes y los puntos débiles de cada una, así como sus funcionalidades más interesantes. Así mismo se plantea la creación de una aplicación para adentrarnos más en las necesidades y las problemáticas que presenta el desarrollo de una aplicación de este tipo.

Una vez extraídas las características más importantes que debe ofrecer un sistema de calendario y adentrados en los estándares que ha de cumplir, se ha procedido a diseñar una aplicación con arquitectura cliente-servidor, cuyo cliente resultaría un cliente de calendario de los ya existentes, y el servidor cumpliría con el estándar WebDAV. Además se ofrecería un portal de gestión a los usuarios para poder acceder y gestionar los recursos.

Finalmente se ha conseguido desarrollar y poner en marcha un sistema propio de calendario que cumple con las funcionalidades definidas y que permite trabajar a diferentes usuarios de forma paralela e interoperable.

Title: Serveis col·laboratius d'agenda, contactes i tasques

Author: Cristian Morón Espadas

Director: Roc Meseguer Pallarès

Date: 21 July 2006

Overview

The main target of the project is to study the need, the evolution and the requirements to create a calendar data sharing system. A calendar system must allow share data to different users and within different organizations.

During the last years, there were different software developers working hardly on the creation of a calendar data sharing system, whose main target was to create a standard way to share the data and also to create a way to have access to it in order to achieve interoperability between the different users.

The project, also, studies the different software applications existing nowadays in the market evaluating them and extracting the most important characteristics. Therefore, we have designed and developed a data calendar sharing system to try to understand better the requirements of this kind of application.

Studying the different software applications existing in the market we can extract the most important options that an application should offer to the users. We have designed a client-server architecture application, whose client is a popular calendar client in the market and the server must be WebDav. In the same way, we have developed a Web page to make easier to the user the managing of its calendar account .

Finally, we have developed a calendar data sharing system that achieves with the targets defined, and that makes possible to work with different users.

A mis padres, porqué siempre han luchado para darme lo que ellos no tuvieron, y porqué sin tenerlo me han dado las lecciones más importantes de mi vida.
A Mari y a Dani por siempre confiar en mi y a Javi, porque siempre estuvo ahí cuando me hacia falta algo, los encabezados, los ánimos y sus siempre acertadas opiniones.
A David, porqué el teléfono sonó mientras yo entraba por esa puerta, y nunca tendré las palabras suficientes para agradecérselo.
A mi tutor Roc, por guiarme y apoyarme en la realización de este proyecto.
Y a Irene, porqué aunque aquí no se incluya, llegó sin darme cuenta, y se puso a redactar el capítulo más bonito de mi vida. Sin su alegría, sin su ilusión no habría podido llevarse a cabo este proyecto.

ÍNDICE

CAPÍTULO1. INTRODUCCIÓN

En este primer capítulo se repasan las razones y motivaciones que han llevado a la realización del proyecto, así como los objetivos del proyecto que se fijan. A su vez, también repasamos la planificación inicial del proyecto, junto con el diagrama de planificación correspondiente, describiendo cada una de las tareas en las que se divide.

En el segundo capítulo encontramos un análisis desde el punto de vista teórico de la necesidad de la existencia de una aplicación calendario y la historia de su desarrollo, así como los protocolos utilizados para éste.

El tercer capítulo es un estudio del mercado de las aplicaciones de calendario, comparándolas entre ellas y extrayendo las funcionalidades que creemos más importantes que cumpla una aplicación de este tipo.

En el cuarto capítulo se repasan las funcionalidades escogidas para nuestro sistema, y el escenario y los actores designados para llevar a cabo la funcionalidad.

EL quinto capítulo resulta ser una descripción de nuestra aplicación, ayudados por diagramas y modelados UML, se expone de forma visual el trasfondo de la aplicación creada.

El sexto capítulo se corresponde con la implementación del sistema, se muestra alguna de las interfaces gráficas creadas para permitir al usuario utilizar la aplicación. Así mismo, se explican las pruebas realizadas para verificar el correcto comportamiento y los resultados esperados.

Las conclusiones ocupan el séptimo capítulo, valorando cuales son los puntos fuertes y débiles del entorno una vez implementada la aplicación, desde un punto de vista personal. Este capítulo pretende ser también un repaso de las posibles mejoras a realizar en nuestra aplicación.

1.1. Razón y oportunidades del proyecto

En el mundo empresarial, así como en diferentes organizaciones es patente la necesidad de una aplicación para compartir información de calendario, donde puedan trabajar conjuntamente los usuarios y se pueda facilitar así la ardua tarea de planificación que tantas horas y tan poco beneficio conlleva. Esta necesidad se acentúa cuando en las rondas de planificación se involucra a un buen número de personas y más si estas pertenecen a diferentes organizaciones.

La necesidad de automatización de este proceso, es el objetivo principal de múltiples desarrolladores de software, desde hace unos años, cuyo principal objetivo es el de estandarizar un mecanismo común de datos y de acceso a éstos que permita de una vez por todas una interoperabilidad real y efectiva.

No obstante la lucha por convertirse en la aplicación de calendario de más éxito, ha llevado a una ramificación fuera de los estándares acordados luchando por convertirse en el estándar de facto.

Con el crecimiento de las compañías, la globalización de las empresas y la cooperación entre ellas crece la necesidad de una aplicación calendario. Así mismo la utilidad de una aplicación de calendario también se hace patente en proyectos entre diferentes empresas, compañías, organizaciones o asociaciones, pudiendo compenetrar fácilmente las tareas de los usuarios que ellos engloban.

Es por estos motivos que aprovechando las grandes perspectivas que posee el entorno, así como las expectativas que plantea, ha surgido la idea de realizar su estudio y una valoración, llegando a realizar una aplicación para conocer mejor su funcionamiento.

1.2 Objetivos

A grandes rasgos, los objetivos del proyecto se basan en el estudio tanto teórico como práctico de los requisitos que presenta un sistema de calendario. Detallándolos más podemos dividir en cuatro grandes bloques dichos objetivos.

En un primer momento, el primer objetivo es introducirse en las necesidades que posee un sistema de calendario, así como en su historia y evolución, estudiando de forma teórica las diferentes posibilidades y características que debe poseer. De esta forma se obtendrá una idea sólida y de base, sobre la dirección a donde se encaminan las plataformas calendario.

Un segundo objetivo es el estudio de las diferentes alternativas presentes en el mercado, extrayendo cuales son sus principales puntos fuertes y desventajas. Este paso permitirá conocer las características más importantes que debe completar un sistema de calendario y que se utilizarán más adelante para el desarrollo.

Como tercer objetivo, la elección de un sistema de calendario, y la instalación y configuración del entorno de trabajo, para poder trabajar correctamente en una aplicación distribuida de calendario.

El cuarto objetivo se corresponde con el mayor de los bloques y consiste en la creación de una aplicación calendario con arquitectura cliente servidor que permita una interoperabilidad con diferentes clientes de calendario y ofrezca diferentes de las funcionalidades extraídas en el análisis, intentado generar una aplicación atractiva a la vez que funcional.

1.3. Planificación Inicial

Seguidamente se describe la planificación realizada para llevar a cabo el desarrollo de nuestros objetivos. Para ello hemos tenido que dividir nuestro proyecto en diferentes partes, siendo éstas cuantificadas en tiempo.

1.3.1. Descripción de las tares a realizar

Hemos dividido nuestro proyecto en un total de 10 grandes tareas a realizar, que ahora se describen. En cada una de ellas se especifica el número de días dedicados, suponemos que cada día supondrá una media de 4 horas de trabajo diarias, existiendo un total de 110 días de trabajo, obtenemos que es necesario invertir un total de 440 horas en nuestro proyecto.

1.3.1.1. Estudio teórico de las aplicaciones Calendario (14 días)

Se trata del estudio de la necesidad de un sistema de calendario, así como de la historia de la evolución de un sistema de este tipo a fin de realizar las tareas deseadas. En este punto se describen los protocolos y sus características que hoy en día son utilizados en el entorno de aplicaciones calendario.

1.3.1.2. Lectura de documentación (5 días)

Para el conocimiento de los protocolos y su entendimiento es necesario leer la diferente información de estos mediante manuales y especificaciones. Así mismo estudiaremos el tipo de datos que se intercambian estas aplicaciones y que más tarde deberemos de conocer para saber sus posibilidades.

1.3.1.3. Búsqueda de software Calendario del mercado (3 días)

Consiste en la búsqueda de los sistemas de calendario que existen en el mercado y que posteriormente analizaremos. Así mismo clasificaremos éstos, según su arquitectura y el role que tengan en el sistema para posteriormente probar su funcionalidad.

1.3.1.4. Instalación y pruebas software de calendario (3 días)

Consiste en instalar las diferentes aplicaciones de calendario de libre distribución encontradas en el punto anterior en nuestra máquina y comprobar su funcionamiento. Así mismo compararemos las aplicaciones y veremos cuales son sus puntos fuertes y sus mayores debilidades.

1.3.1.5. Definir objetivos y funcionalidades para nuestra aplicación (9 días)

Una vez analizadas las diferentes aplicaciones intentaremos extraer las funcionalidades que más nos convienen para nuestro sistema intentado, analizar cual debe ser la arquitectura de nuestra plataforma y los diferentes usuarios que deberían interactuar en ella.

1.3.1.6. *Diseño del sistema a implementar (4 días)*

Esta etapa del proyecto consiste en especificar y diseñar la aplicación a realizar. El resultado esperado es un documento encargado de describir las clases utilizadas, los casos de uso y las funcionalidades, ayudados por medio del modelado UML.

1.3.1.7. *Implementación de la parte gestora de calendario (25 días)*
Utilizando las funcionalidades extraídas en las etapas anteriores crearemos una aplicación con arquitectura cliente servidor escrita en lenguaje JAVA y que constará de dos partes. Una parte de gestión que resultará un portal Web.

1.3.1.8. *Implementación de la parte lógica de calendario (19 días)*

La otra pieza a implementar se basará en la parte de lógica de calendario, que resultará una interfaz entre un cliente de calendario y un servidor instalado previamente.

1.3.1.9. Creación de documentación (9 días)
Tarea consistente en la redacción del resultado de las diferentes tareas realizadas, a fin de presentarlo como el documento de la memoria final del trabajo.

1.3.1.10. Pruebas de funcionamiento (6 días)
Una vez finalizada la aplicación será el momento de cerciorar, que esta funciona y devuelve los resultados esperados. Para ello describiremos diferentes pruebas realizadas y el resultado devuelto. Dicha tarea se realiza después de entregar la documentación al tutor, a fin de aprovechar los días en los que el tutor repase el trabajo realizado.

1.3.1.11. *Preparación de la presentación (3 días)*
Tarea consistente en la preparación de la presentación con el fin de exponer los resultados finales del trabajo.

En la siguiente imagen se puede apreciar el Diagrama Gantt estimado para llevar a cabo nuestro proyecto. Observamos como se reparten las tareas en el tiempo y como se compaginan estas para estar acorde con nuestra planificación

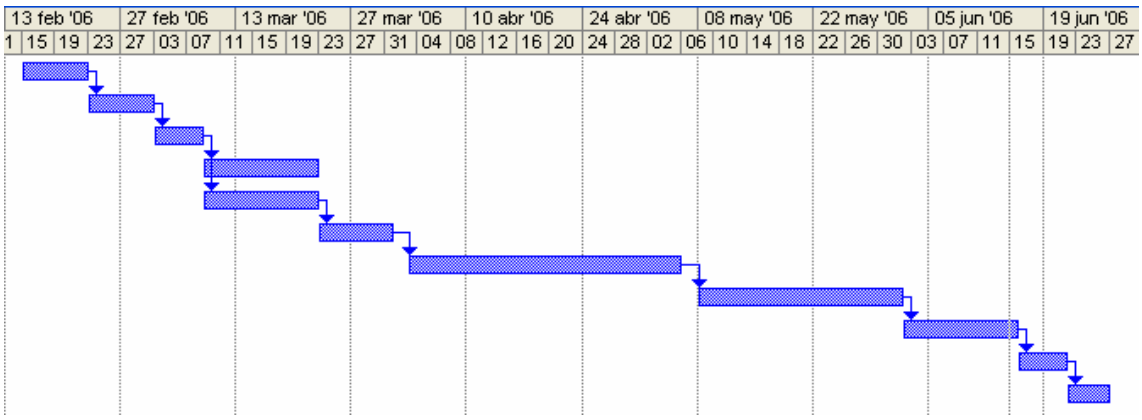

Fig. 1 Imagen del Diagrama Gantt de planificación del proyecto

CAPÍTULO 2. EL ENTORNO DEL PROYECTO

En este capítulo se describen las necesidades de creación de un sistema de calendario y las características y funcionalidades que este sistema ha de asumir. Así mismo se repasa la historia de su evolución y los diferentes avances acometidos en el campo. Finalmente repasamos las características técnicas de los protocolos utilizados por estas aplicaciones.

2.1. Necesidad de calendarios Interoperables

En la actualidad supone un gran esfuerzo programar una reunión que implica movilizar a trabajadores de diferentes empresas. Esto conlleva múltiples rondas de llamadas telefónicas o un buen número de envíos de correos electrónicos a fin de intentar poner de acuerdo a todas las personas que están implicadas. Estos contactos siempre suelen ser coordinados por una persona que intenta consensuar las diferentes opiniones y solventar los problemas que tengan los asistentes, intentando planificar un encuentro a la medida de todos.

A fin de solucionar este problema, durante los últimos años ha ido surgiendo diferentes aplicaciones software, que permitían a los usuarios mantener múltiples calendarios almacenados en sus máquinas. En el mundo de los negocios, los calendarios personales no resultan útiles si no permiten confirmar la disponibilidad de las personas, planificar reuniones, reservar lugares o acceder a la información incluso cuando el usuario se encuentra de viaje.

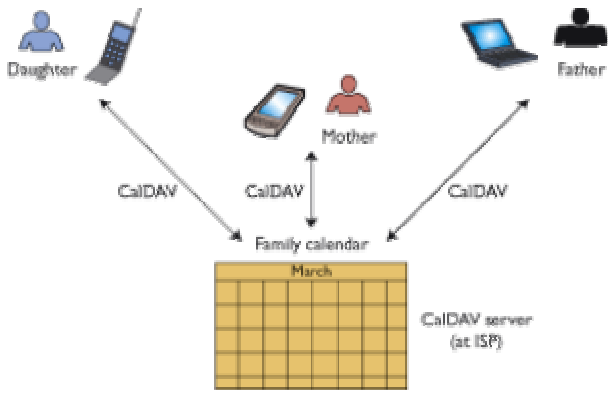

Fig. 2 Escenario que refleja un trabajo colaborativo entre diferentes entes. Cada elemento accede desde un dispositivo diferente a los datos de calendario almacenados en un servidor.

En el mercado, existen diferentes programas de calendario que pueden solventar los citados problemas de planificación. Así pues, Oracle Calendar o Microsoft Exchange/Outlook permiten que mediante un servidor de calendario, y conociendo el tiempo libre del que posee cada persona del grupo, se pueda proponer un horario de reunión a los asistentes, que posteriormente podrá ser aceptado o rechazado por ellos. La utilización de estas tecnologías puede

aportar beneficios, rebajando los tiempos de coordinación de decenas a escasos minutos.

No obstante, estos programas de planificación funcionan correctamente cuando las reuniones se fijan entre personas pertenecientes a una misma empresa, pero el verdadero problema surge al intentar organizar a personas de diferentes empresas, dado que se debe volver al método de rondas de llamadas y de envíos de correos electrónicos. Todo ello se solucionaría si se pudiera trabajar con un sistema de calendario independiente de las diferentes organizaciones y que fuera capaz de comunicarlas y almacenar los datos de forma estándar.

Para conseguir desarrollar un sistema de este tipo, el usuario necesita dos elementos principalmente. Primero, un lenguaje estándar que permita a todo el mundo trabajar con un mismo formato de datos. Es decir, necesitamos todos hablar con un mismo idioma para entendernos. Y en segundo lugar, se requiere un protocolo que ofrezca interoperabilidad, es decir, que pueda funcionar independientemente de la aplicación que sea elegida, permitiendo la existencia de diferentes clientes de calendario así como de diferentes servidores que se puedan comunicar por Internet intercambiando la información que poseen.

En definitiva, se requiere de un sistema que idealmente funcionará de forma tan eficiente como si se trabajara dentro de una misma empresa y debería proporcionar acceso a los calendarios de forma abierta, consistente y estándar.

2.2. Historia del desarrollo de Calendarios Interoperables

Durante los últimos años se ha hecho patente la necesidad de desarrollar una plataforma de calendario interoperable. Ante esta necesidad han sido diferentes los movimientos que se han realizado y diferentes las líneas de investigación encabezadas, desde la definición de un formato estándar de datos, pasando por la definición de los protocolos y su creación.

En Julio de 1996 la empresa americana Netscape Communications encabezó la formación de un grupo de trabajo dedicado al desarrollo de estándares para planificación de datos de calendario sobre la red Internet. Este grupo se convirtió en el denominado IETF's Calendaring and Scheduling (CalSch), que comenzó a trabajar desde Octubre de 1996 y cuyo trabajo se dio por finalizado en Septiembre de 2004. CalSch dividió su trabajo en tres líneas fundamentales:

- Desarrollar un modelo de datos y una representación textual para los eventos de tipo calendario (creación de la especificación iCalendar).
- Desarrollar el transporte de información de calendario(que se convirtió en el protocolo iTIP [iCalendar Transport-Independent Interoperability Protocol])
- La creación de una especificación de carácter general para acceder a los datos de calendario y planificación (que se convirtió en Calendar Access Protocol [CAP]).

Un problema clave que se encontraron a fin de desarrollar aplicaciones de tipo calendario interoperables, fue el de determinar una vía estándar para representar los datos de tipo calendario, datos de eventos, alarmas o datos que podían repetirse en el tiempo tal y como una reunión diaria. Así pues, decidieron partir de una especificación realizada años antes por el Internet Mail Consortium1, denominada vCalendar y que había obtenido un éxito más bien escaso. Después de dos años de refinamiento de aquel trabajo, CalSch publicó el RFC 2445, que es el que hoy en día se utiliza de forma mayoritaria para representar datos. Así se creó un nuevo formato denominado iCalendar, que comparte con vCalendar ciertos aspectos en la descripción de los datos. El formato iCalendar fue rápidamente aceptado y sustituyó al anterior formato vCalendar.

En otra de las vías de investigación del grupo de trabajo CalSch, se trabajó sobre un protocolo para poder gestionar los datos de tipo calendario denominado iTIP (iCalendar Transport-Independent Interoperability Protocol). Dicho protocolo permite funciones como crear, modificar o eliminar componentes de calendario. No obstante, el grupo de trabajo se limitó a redactar un documento que describía conceptualmente como realizar operaciones con calendarios, pero no proporcionó su implementación. Esto dejó lugar a que se creasen diferentes implementaciones para realizar las tareas de transporte. Así pues, se creó la especificación iMIP, que describía como llevar a cabo las operaciones propuestas en iTIP vía correo electrónico. Esta propuesta ha tenido cierto éxito, incluyendo diversas implementaciones por parte de algunos vendedores y proporcionando cierta interoperabilidad entre organizaciones. No obstante, iMIP no es el medio de transporte utilizado hoy en día por las principales aplicaciones de calendario.

La tercera línea de investigación se centró en la creación de un nuevo protocolo de transporte propio de este tipo de datos, que se conocería como CAP. CAP fue diseñado para proporcionar funcionalidades de acceso a los elementos calendario ubicados en los diferentes servidores, así mismo se encarga de especificar mecanismos de búsqueda en calendarios de otros usuarios. En el desarrollo de CAP, el grupo de trabajo CalSch desarrolló completamente un nuevo protocolo distinto a todos los ya existentes, aunque para ello se basó en diferentes protocolos ya conocidos como POP, IMAP o BEEP. No obstante CalSch no progresó más en el protocolo CAP, y el IETF cerró el grupo de trabajo en Septiembre de 2004, después de cuatro años de desarrollo, CAP no llegó a desarrollarse nunca.

Dado el lento progreso que experimentaba el grupo CalSch en sus desarrollos, la compañía Apple Computer trabajó de forma paralela, y en 2002 lanzó su propia aplicación de calendario personal denominada iCal. La aplicación iCal cumplía con el estándar de representación de datos iCalendar definido por el grupo de trabajo de la IETF, no obstante introducía interesantes novedades en el ámbito de los protocolos. Con iCal, un usuario podía publicar un calendario en un servidor que cumpliera con el protocolo WebDAV, desde el cual los

[1] El Internet Mail Consortium (IMC) es el nombre de un grupo creado por casi una veintena de empresas informaticas y que promueven actividades para implementar nuevos estándares, lograr la interoperatividad entre programas o solventar problemas de seguridad.

usuarios podían ver y descargar los eventos disponibles. Apple diseñó la aplicación para integrarse con su servicio propietario WebDAV Mac, pero iCal era capaz de interoperar con cualquier servidor WebDAV. Algunos clientes de código abierto adoptaron iCal sobre WebDAV como el primer estándar de facto de datos calendario.

Apple hizo una interesante elección al escoger WebDAV como medio de transporte. El protocolo extiende de HTTP añadiéndole nuevas características como: prevención de sobrescritura, operaciones con elementos (listar colecciones, mover, copiar, crear una nueva colección) y descripción de los datos. Estas nuevas características se combinan con las propiedades de lectura, escritura y eliminación de recursos que ya proporcionaba HTTP. WebDAV proporciona las características necesarias para la publicación de forma remota y poder compartir calendarios. Esta elección, resultó ser más atractiva que implementar una nueva infraestructura de servidores para un nuevo protocolo como proponía el grupo de trabajo con CAP. Apple demostró que los calendarios podrían ser tratados como cualquier otro recurso WEB accesible vía HTTP.

No obstante iCal poseía diferentes carencias en su funcionamiento tal y como dificultades para la búsqueda de información de los usuarios, información como los denominados tiempos libres/ocupados de los usuarios cuando se debía de hacer la búsqueda en un numeroso conjunto de personas o dificultades para encontrar los calendarios de otros usuarios. Dichos problemas residían a nivel de protocolo. WebDAV no proporciona soporte para acciones como localización de calendarios o búsquedas. Para proporcionar estas características se concluyó que se debería desarrollar un nuevo módulo al protocolo que añadiera dichas funcionalidades, y que pasaría a denominarse CalDAV (Calendaring and Scheduling Extensions to WebDAV).

CalDAV, que actualmente está en desarrollo, recoge las opciones desarrolladas por Apple en su iCal, e intenta englobar nuevas funcionalidades únicas y especificas para las tareas de calendario.

El protocolo CalDAV prevé proporcionar tres características principales añadidas a las que nos ofrecen las capas inferiores y que más adelante repasaremos con profundidad:

- Creación de calendarios. Los usuarios podrán crear fácilmente múltiples calendarios personales (un calendario para trabajo, otro para sus conferencias, uno para casa...) gracias a un nuevo método MKCALENDAR.
- Búsquedas en calendarios: Los usuarios podrán realizar diferentes consultas como los tiempos libres/ocupados en calendarios de otros usuarios, o consultar quien esta participando en una reunión.
- Seguridad en calendarios: El protocolo permitirá controlar si sus calendarios son visibles a otros usuarios, así como quien tiene permiso para cambiarlos.

A fin de poder conocer mejor las propuestas realizadas y la evolución experimentada por estas, en los siguientes apartados estudiaremos los protocolos que hemos visto y que fueron elegidos para trabajar con aplicaciones calendarios: HTTP, WebDAV y CalDAV.

2.3. El protocolo HTTP

2.3.1. Definición de HTTP

El Protocolo de Transferencia de Hipertexto[2] (Hypertext Transfer Protocol) cuya abreviación se corresponde con las siglas HTTP, es un sencillo protocolo que sigue una arquitectura cliente-servidor que hace posible los intercambios de información entre los clientes y los servidores Web. HTTP es un protocolo rápido y sencillo que permite la transferencia de múltiples tipos de información de forma eficiente y rápida. La especificación completa del protocolo HTTP 1/0 está recogida en el RFC 1945. Como una de sus principales características encontramos, que a diferencia de protocolos como FTP, HTTP es un protocolo de solo lectura que permite a los clientes conectarse y obtener la información que desea.

HTTP se basa en sencillas operaciones de solicitud/respuesta. Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud. El servidor responde con un mensaje similar, que contiene el estado de la operación y su posible resultado. Todas las operaciones pueden adjuntar un objeto o recurso sobre el que actúan; cada objeto Web (documento HTML, fichero multimedia o aplicación) es conocido por su URL. Los recursos se identifican con una URL o "Universal Resource Locator" que es un identificador único en el mundo que se asocia con el recurso dado.

Los recursos u objetos que actúan como entrada o salida de un comando HTTP están clasificados por su descripción MIME (Multi-Purpose Internet Mail Extensions, Extensiones de correo Internet multipropósito). Estas extensiones son una serie de convenciones o especificaciones dirigidas a que se puedan intercambiar a través de Internet todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario. De esta forma, el protocolo puede intercambiar cualquier tipo de dato, sin preocuparse de su contenido. La identificación MIME permitirá que el receptor trate adecuadamente los datos, identificándolos.

Existen cuatro métodos básicos que contempla HTTP y que un cliente puede utilizar para dialogar con el servidor:

- El método GET utilizado para recoger un objeto del servidor remoto.
- El método POST, para enviar información al servidor Web.
- El método PUT permite almacenar el contenido de la petición en el servidor bajo la URL designada en la petición. La principal diferencia entre POST y PUT se encuentra en el significado de la URL. En el caso

[2] , El hipertexto es el lenguaje e etiquetas contenido de las páginas web

del método POST, la URL identifica el recurso que va a manejar en contenido, mientras que en el PUT identifica el contenido.
- El método DELETE se utiliza para que el servidor borre el recurso indicado por la URL de la petición.

Ante estos métodos un servidor Web se encarga de devolver el resultado de la operación mediante unos determinados y tabulados códigos de respuesta que encabezan la respuesta que ejerce el servidor al cliente y el contenido si procede. Se pueden consultar estos códigos en los Anexos.

2.3.2. Modelado de objetos calendario como recursos HTTP

Con el modelo propuesto por Apple con su iCal, los objetos calendario pasaban a ser representados como simples recursos HTTP, con las ventajas que esto ofrece. Los calendarios, como recursos HTTP, son identificados por URLs y pueden ser devueltos como respuestas a peticiones realizadas con el método GET. Las aplicaciones de calendario pueden realizar fácilmente operaciones como eliminar o escribir recursos utilizando métodos propios de HTTP como DELETE y PUT respectivamente.

Dado que los eventos de calendario son representados como recursos HTTP, el protocolo debe decidir que valor va en el cuerpo y que tipo MIME se asigna a estos recursos. Esta elección determina el resultado que se devolverá ante una petición GET. El grupo de trabajo optó por usar el estándar iCalendar para representar todos los eventos de datos dentro del cuerpo del mensaje HTTP. Los servidores Web pueden almacenar ficheros con el tipo MIME “text/calendar,” y es corriente que los navegadores puedan mostrar datos iCalendar con aplicaciones de calendario.

Dentro del formato de datos iCalendar, un calendario está compuesto por eventos a los que alguna persona o grupo puede asistir tal y como reuniones, citas, representaciones, viajes o alguna acción que posea un tiempo de inicio y un tiempo de fin.

Cada evento es representado como un recurso y los contenidos de los recursos son representados en formato de texto. Debido a las posibles acciones que pueden realizar los usuarios es necesario proporcionar una URL como identificador para cada uno de los eventos, para poder crear, eliminar o sobrescribirlos.

De esta misma forma, se consideró que podría ser útil identificar mediante URLs las propiedades de un evento (propiedades como su localización o su tiempo de inicio y fin) para poder ser modificadas fácilmente, accediendo directamente al recurso. No obstante, y tras diferentes pruebas se consideró que sería más eficiente modificarlas sobrescribiendo el evento completo, dado que conllevaba un tiempo razonable de transmisión y procesado. Por tanto, el grupo de trabajo definió las diferentes propiedades que debían describir los eventos, pero no planeó identificar dichas propiedades de los eventos con URLs.

A continuación se muestra una petición GET de HTTP para un evento iCalendar almacenado en un servidor con su respuesta asociada.

Fig. 3 En la imagen se puede observar como la respuesta HTTP incluye un objeto iCalendar con un MIME TYPE definido como “text/calendar”.

2.3.3. Características HTTP utilizadas en calendarios

Con eventos iCalendar modelados como recursos Web, podemos utilizar la metodología básica de HTTP para solventar diferentes casos de uso.

- Para descargar un evento, una aplicación calendario utiliza el método GET e interpreta el cuerpo de la respuesta como un fichero iCalendar.
- Para crear un nuevo evento, la aplicación utiliza el método PUT (conjuntamente con una URL no existente) enviando un fichero iCalendar en el cuerpo de la petición.

- Para sobrescribir un evento existente o cambiar alguna de sus propiedades como su localización, los tiempos asignados o los asistentes, las aplicaciones de calendario utilizan el método PUT conjuntamente con la URL que se corresponde con el evento ya existente y que se desea cambiar. El cuerpo de esta petición estará conformado por un fichero iCalendar, con los nuevos valores para el evento en formato iCalendar.
- Para eliminar un evento existente, las aplicaciones utilizan el método DELETE con la URL perteneciente al evento.
- Para chequear si un evento ha cambiado desde la última descarga, la aplicación utiliza la cabecera Etag de HTTP.

Con HTTP obtenemos un gran número de funcionalidades que se contemplan para datos calendario, pero también observamos que no soporta diferentes operaciones útiles para datos calendario. Como se ha explicado anteriormente, el protocolo HTTP no tiene capacidad para listar todos los eventos de calendario de una persona o acciones como evitar la sobrescritura de calendarios. El protocolo WebDAV si soporta estas funciones, y por tanto un servidor de calendario debería trabajar conjuntamente con los protocolos HTTP y WebDAV.

Existen diferentes servidores HTTP en el mercado, pero existen dos que destacan por ser los más utilizados. Por un lado se encuentra Apache de código abierto desarrollado por la Apache Software Foundation e Internet Information Server (IIS) propietario de Microsoft. Por otro lado, y también de la asociación Apache Software Foundation encontramos un servidor y contenedor de páginas dinamicas, o también denominadas servlets, el conocido como Tomcat (también llamado Jakarta Tomcat o Apache Tomcat).

2.4. El protocolo WebDAV

2.4.1. Definición de WebDAV

En el nacimiento de la WWW (World Wide Web), y con la aparición de HTTP, la Web creció dentro de un medio de solo lectura. Uno de los principales objetivos desde entonces ha sido solventar esta limitación, intentándola transformar en un medio legible y editable.

WebDAV cuyas siglas representan “Web-based Distributed Authoring and Versioning” es un conjunto de extensiones del protocolo HTTP/1.1, que añade nuevos métodos y cabeceras. Se definen básicamente formatos y algoritmos con el fin de que el cliente y el servidor puedan interoperar.

Fig. 4 En la imagen se puede observar la filosofía de WebDAV, mantener las cabeceras y métodos de HTTP y extender sus funcionalidades sobre él.

Al definirse minuciosamente el protocolo, se facilita la creación de diferentes clientes en diversos lenguajes para implementar sus funcionalidades. WebDAV ha sido hecho público por el IETF bajo el RFC 2518. Se permite a los usuarios trabajar conjuntamente permitiendo editar y gestionar ficheros que se encuentran almacenados en servidores Web remotos.

El estándar ha sido desarrollado por un grupo de trabajo específico (IETF WebDAV Working Group) que ha trabajado para definir las extensiones necesarias para poder distribuir herramientas de autoridad en Web permitiendo interoperar a los diferentes usuarios, mientras se le ofrecen diferentes necesidades.

Las principales características de WebDAV:

- Bloqueo (control de concurrencia) Se proporciona bloqueo de escritura encargado de prevenir el problema de sobreescritura. Para conseguir un escalado robusto de Internet, donde la red podría ser desconectada de forma arbitraria y conseguir escalabilidad, dado que cada conexión abierta consume recursos del servidor, la duración del bloqueo es independiente de las conexiones de red de los usuarios.
- Propiedades: Las propiedades XML proporcionan almacenamiento para metadatos, tal y como una lista de autores pertenecientes a los recursos Web. Estas propiedades pueden ser añadidas, eliminadas y recuperadas de forma eficiente.
- Manipulación de Directorios: WebDAV soporta operaciones de copiado y movimiento de ficheros, operaciones similares a las que se pueden realizar con sistemas de ficheros.
- Utiliza XML: WebDAV tiene la distinción de ser el primer protocolo que utiliza el formato XML.

Se puede ver WebDAV como un sistema de ficheros en red, disponibles desde Internet, que trabaja con diferentes ficheros al mismo tiempo y que posee un buen comportamiento en ambientes de grandes latencias.

El siguiente esquema nos ayuda

WebDAV añade los siguientes métodos a HTTP:

- PROPFIND - Utilizado para recuperar propiedades almacenadas como XML de un recurso.
- PROPPATCH - Utilizado para cambiar y borrar múltiples propiedades en un recurso.
- MKCOL - Utilizado para crear colecciones.
- COPY - Utilizado para copiar un recurso desde una dirección a otra.
- MOVE - Utilizado para mover un recurso desde una dirección a otra.
- LOCK - Utilizado para bloquear un recurso, WebDAV soporta bloqueos compartidos y exclusivos.
- UNLOCK - Para borrar un bloqueo de un recurso.

2.4.2. Características WebDAV utilizadas en calendarios

La tecnología WebDAV proporciona funcionalidades en una capa superior a HTTP siendo estas imprescindibles para el funcionamiento de un servidor de calendario. Permite recopilar eventos que residan en el servidor Web de forma separada en calendarios o permite acceder a un simple evento. Así mismo permite consultar que eventos han sido añadidos o eliminados desde la última sincronización de calendario realizada. Para realizar estas acciones se utilizan las colecciones propias de WebDAV como modelado de calendarios. Una colección WebDAV permite listar recursos de forma clara, flexible y a la vez parseable dado que su lenguaje es XML. Al definir eventos como recursos, las operaciones WebDAV funcionan de la siguiente forma:

- Para listar todos los eventos calendario, se utiliza el método PROPFIND con la URL del calendario.
- Para listar los calendarios en un determinado lugar de un repositorio, se utiliza PROPFIND con la URL apropiada.
- Para copiar un evento en otro calendario o crear un evento duplicado dentro de un día del calendario se utiliza COPY.
- Para renombrar un evento o moverlo a otro calendario se utiliza MOVE.
- Para bloquear un evento o bloquear un calendario completo mientras se hacen cambios se utiliza lock/unlock.

WebDAV también define el método MKCOL, que crea colecciones de recursos, pero en calendarios no se utiliza el método, porqué un calendario es más que una colección corriente.

Cabe aclarar que aún siendo necesarios nuevos mecanismos, bajo un servidor WebDAV es posible sostener una aplicación calendario que funcione a pleno rendimiento, tal y como ya sucedió con Apple y su aplicación iCal.

2.5. Nuevos mecanismos Caldav

Una vez asentado un servicio de calendario sobre WebDAV y HTTP se listaron las funcionalidades extras necesarias para ofrecer nuevas funcionalidades y nuevas metodologías. Estas se recogieron en la extensión al protocolo WebDAV que propone CalDAV.

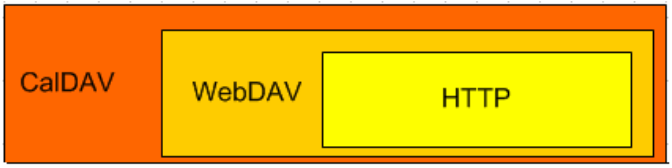

Fig. 5 En la imagen se puede observar la filosofía de CalDAV, añadiendo nuevas extensiones a WebDAV y HTTP.

2.5.1. Consulta de calendarios

El principal caso de uso que HTTP y WebDAV no pueden ofrecer es la búsqueda de un calendario. Debido a las características de los datos de tipo iCalendar se requiere un método de búsqueda que pueda trabajar con diferentes intervalos de tiempo y donde los eventos puedan ser repetidos.

Cuando se planifica una reunión, el caso de uso más común es la petición de todos los eventos que posee el usuario en un rango de tiempo dado. Para planificar una reunión con un socio, se desearía poder ver su planificación para la semana, incluyendo sus reuniones y su duración. Esto no es una petición complicada, dado que existe una sintaxis descrita que es asociada a los rangos de tiempo.

CalDAV utiliza unos informes que se encargan de especificar los intervalos de tiempos para datos de calendario. Utilizar este informe proporciona una solución para recopilar información mediante una única petición y su correspondiente respuesta que resultaría muy costosa de pedir con el método GET y peticiones del tipo PROPFIND. Así mismo, este tipo de informe solventa el problema de eventos repetidos e identificación de eventos ocurridos en un mismo rango de tiempos.

2.5.2. Identificación y creación de calendarios

Para tratar un calendario como una colección, el cliente tiene que conocer que el objeto se trata de un calendario. El mecanismo que WebDAV define para ello es la utilización de la propiedad *resourcetype* y CalDAV define un nuevo valor para ella: *CALDAV:calendar*.

Para pedir a servidores la creación de calendarios, los clientes necesitan un método que indique que tipo de recurso crear y que funcionalidad proporcionar. En CalDAV el cliente simplemente envía una petición MKCALENDAR al servidor, que se encargará de crear un calendario dentro de un servidor que cumpla con CalDAV. El servidor asigna automáticamente el valor correcto de *resourcetype* que permite a otros clientes detectar que el nuevo recurso es un calendario que contiene eventos.

2.5.3. Planificación reuniones con CalDAV e iMIP

Una de las soluciones propuestas es la de utilizar CalDAV conjuntamente con el protocolo iMIP. Un cliente puede utilizar los informes para consultar los tiempos libres/ocupados de otro usuario y entonces utilizar el estándar iMIP para enviar una invitación sobre correo electrónico con el fin de establecer una reunión. De esta forma CalDAV aumenta las posibilidades de que la invitación a una reunión no tenga problemas con los horarios.

2.5.4. Estado de CalDAV

El desarrollo de CalDAV no se ha concluido. Su especificación continúa en progreso bajo el desarrollo de un pequeño grupo de personas, que incluye representaciones de la denominada Open Source Applications Foundation y la compañía Oracle.

El trabajo actual en el desarrollo de CalDAV se encuentra en las manos de:

- Cyrus Daboo, de la empresa Isamet
- Bernard Desruisseaux, que trabaja en Oracle Calendar Server,
- Lisa Dusseault, de la fundación Open Source Application Foundation

A fin de aumentar la eficiencia en el proceso de estandarización, CalDAV se desarrolla en un grupo pequeño pero abierto. Se aceptan ideas y contribuciones a través de la lista abierta[3]. Una vez que se establezca un prototipo funcionando correctamente e interoperando, se comenzará el proceso de estandarización.

Diferentes aplicaciones han comenzado ya a desarrollar implementaciones CalDAV. El primer test de interoperabilidad tuvo lugar en Enero de 2005, y permitió comprobar la interoperabilidad entre dos servidores y tres clientes diferentes de calendario.

Una vez CalDAV demuestre un correcto funcionamiento se espera que aumente el número de aplicaciones que soporta CalDAV, y las aplicaciones que trabajan con WebDAV migren a este nuevo protocolo.

2.5.5. Software CalDAV

Como hemos repasado en el punto anterior, se está desarrollando diferentes aplicaciones que trabajan con CalDAV. No obstante el número de estas aún es muy pequeño. Así mismo su funcionamiento se considera aún en estado experimental y los resultados de su utilización está bajo la expectación de los desarrolladores.

La mayoría de aplicaciones que utilizan CalDAV se basa en aplicaciones de código abierto, el principal exponente es Mozilla Calendar Project[4]. Dada la popularidad de la distribución de los proyectos de Mozilla, se trata del cliente más utilizado y experimentado del momento. Las extensiones de calendario para Firefox y Thunderbird, la aplicación de calendario Sundbird, y el cliente aún en desarrollo Mozilla Lightning son ejemplos de aplicaciones interoperables que soportan CalDAV.

[3] http://lists.osafoundation.org/mailman/listinfo/ietf-caldav
[4] http://www.mozilla.org/projects/calendar/

Por otra parte, OSAF una fundación de desarrollo de aplicaciones de código abierto, ha desarrollado la aplicación cliente denominada Chandler[5] que proporciona funcionalidades CalDAV. Chandler esta disponible para Linux, Mac OS X y Windows. Paralelamente, OSAF está desarrollando una aplicación Web de calendario denominada Scooby.

Para el cliente Novell's Evolution[6] existe un módulo disponible que permite proporcionar la funcionalidad de CalDAV. La solución de Novell es una solución propietaria. En la misma línea existen proyectos no oficiales que añaden la funcionalidad de calendario CalDAV a Microsoft Outlook en Windows, tal y como caldavxp[7] y Open Connector [8].

Por otra parte están los servidores de Calendario que están más experimentados que los clientes. Así pues, OSAF ha desarrollado un servidor denominado Cosmo [9], de código abierto. Cosmo está basado en Java y viene empaquetado para Apache Tomcat. Aunque su objetivo principal es proporcionar las funcionalidades CalDAV, Cosmo implementa un servicio WebDAV completo, y posee una interfaz donde los usuarios pueden gestionar ambos servicios en su propia cuenta.

Otro servidor de código abierto es Bedework[10] una rama separada del servidor UW Calendar desarrollado por la universidad de Washington. Bedework es un servidor de código abierto y como Cosmo, esta escrito en JAVA. Bedework es utilizado en la universidad de Washington y por diferentes instituciones académicas y esta diseñado para trabajar con diferentes clientes.

Una tercera opción es el servidor de correo y calendario de Novel Hula11, que combina POP, IMAP, LDAP y funcionalidad CalDAV en un mismo paquete. Hula se ejecuta como una aplicación servidora sin necesidad de otros módulos. Los paquetes oficiales son ofrecidos por Novel para una variedad de distribuciones Linux y FreeBSD.

Oracle ha publicado su intención de proporcionar funcionalidades CalDAV en próximas versiones del servidor Oracle Calendar. Aunque en la actualidad no se proporciona soporte para CalDAV. Oracle es uno de los principales desarrolladores de especificaciones CalDAV y ha participado en la interoperabilidad de CalDAV utilizando diferentes prototipos.

2.6. Resumen y conclusiones

En el siguiente cuadro se expresa a modo resumen las funcionalidades que aportan los diferentes protocolos expuestos y que son utilizados en la evolución de los sistemas calendario.

---

[5] http://chandler.osafoundation.org/
[6] http://www.novell.com/products/desktop/features/evolution.html
[7] (http://dforge.cse.ucsc.edu/projects/caldavxp/)
[8] http://openconnector.org/).
[9] http://wiki.osafoundation.org/bin/view/Projects/CosmoHome
[10] http://www.bedework.org/bedework/),
[11] (http://hula-project.org/Hula_Project

Cabe destacar como los protocolos superiores acogen las propiedades dadas por los protocolos de menor nivel.

Tabla 1. Comparación de funcionalidades

	HTTP	WebDAV	CalDAV
Lectura y acceso a eventos	X		
Creación de nuevos eventos	X		
Eliminación de eventos	X		
Sobrescritura de eventos	X		
Creación de colecciones de eventos		X	
Mover eventos de localización		X	
Control de sobreescritura y bloqueo		X	
Búsqueda y consulta de planificación de eventos del usuario			X
Creación de calendarios			X

CAPÍTULO 3. Análisis de mercado. Benchmarking

Una vez repasados los diferentes protocolos y formatos que se utilizan en los sistemas de calendario, procederemos a estudiar las aplicaciones que existen en el mercado. En este capítulo se analizan las diferentes aplicaciones de calendario de código libre que se encuentran actualmente en el mercado. De este análisis hemos extraídos los puntos positivos y negativos de cada una de las aplicaciones. A partir de este trabajo se ha generado una lista de ideas para incorporar a la nueva plataforma.

3.1. Software

Como primer paso hemos procedido a la prueba de las diferentes plataformas que sobre software libre existen en el mercado, para ello hemos seleccionado las plataformas PC existentes de libre distribución. En este analisis, hemos intentado trabajar con las herramientas que implementa CalDAV ya existentes y expuestas anteriormente y tambien con un servidor WebDAV.

Podemos dividir el software existente en tres grandes grupos. En primer lugar, los servidores de calendario, que son piezas de software que únicamente proporcionan la opción de la creación de calendarios, gestión de usuarios y requieren su utilización conjuntamente con otra pieza de software que idealmente debe de ser totalmente independiente del servidor y forman el segundo grupo, los clientes de Calendario. Como tercer grupo podemos destacar los Clientes/Servidor de calendario que poseen la funcionalidad de la creación de calendarios y que a su vez permite el acceso a la información con un simple navegador Web, aunque también pueden llegar a ofrecer opciones de interoperabilidad con otros clientes de calendario. Hemos estudiado:

- Servidores Calendario:
  - Cosmo
  - Servidor WebDAV (Sin soporte CalDAV)

- Clientes Calendario:
  - Mozilla Calendar / SundBird
  - Chandler
  - Cliente para Microsoft Outlook

- Servidores/Clientes de calendario:
  - Kiko
  - UW Benkelor
  - Google Calendar

En las pruebas realizadas hemos intentado comprobar sus funcionalidades, haciendo hincapié en las posibilidades de autenticación que se ofrecen, las posibilidades de sincronización y la interoperabilidad que presenta.

3.2. Servidores de Calendario

Los servidores de Calendario son piezas que únicamente se utilizan para la creación de una cuenta de calendario y el aprovisionamiento de un usuario. La gestión de usuarios se debe realizar con mecanismos externos. Cuando hablamos de la gestión de usuarios nos referimos a la forma en que la aplicación permite que un usuario comparta sus calendarios, le de permisos de lectura a algunos usuarios o permisos de modificación de la información.

3.2.1 Cosmo

La fundación OSAF (Open Source Applications Foundation) ha creado un servidor CalDAV de código abierto denominado Cosmo. Cosmo que va por la versión 0.3 en desarrollo y que está basado en Java ha sido empaquetado para el servidor Web Apache Tomcat. El servidor tiene como principal propósito el soporte CalDAV, pero implementa un servidor WebDAV en su totalidad, y posee una interfaz donde los usuarios pueden gestionar sus servicios en una cuenta propia.

Fig. 6 Logotipo de Cosmo

Cosmo funciona en los sistemas Unix, pero ha sido probado su funcionamiento en sistemas Linux y OS X. El soporte para Windows no ha sido desarrollado por el momento, pero se espera en próximas versiones. OSAF ha publicado un servidor de libre acceso para pruebas en cosmo-demo.osafoundation.org, éste se corresponde con la versión 0.26.

Para acceder al servidor se ha de crear primeramente una cuenta en él, con el nombre del repositorio de calendarios a crear, un correo electrónico y una contraseña para poder acceder al calendario. De esta forma se produce una autenticación, donde solo un usuario gestor que conozca esta contraseña podrá crear y distribuir el calendario.

Tabla 2- Ejemplo de uno de los calendarios creados

Server:	cosmo-demo.osafoundation.org
Path:	/home/ProyectoPFC/
Username:	ProyectoPFC
Password:	*hidden*
Port number:	443
Use SSL:	Yes

La plataforma pone como restricción que el nombre del repositorio, usuario y correo electrónico sea único en la plataforma. Bajo esta interfaz, y como se puede observar en la tabla anterior, se ha producido la creación de un único repositorio denominado ProyectoPFC y donde más adelante e interactuando con sus clientes procederemos a crear los diferentes calendarios.

Teóricamente Cosmo posee múltiples posibilidades de interoperabilidad y puede funcionar con diferentes clientes de calendario como:

- Chandler 0.6
- Apple iCal para Mac
- Mozilla Calendar / Sunbird
- Otros clientes CalDAV como Mulberry para leer y escribir calendarios compartidos.

Cosmo funciona únicamente como servidor, y la información almacenada únicamente puede ser accedida bajo el empleo de un cliente externo.

Tras la creación de nuestro repositorio, hemos comprobado como con diferentes clientes se puede compartir eventos publicándose estos en el servidor y pudiendo ser descargados. No obstante, el cliente Chandler creado también por la fundación OSAF y que más adelante se estudia, es el que proporciona unos mejores resultados de interoperabilidad para el servidor Cosmo.

3.2.2 Servidor WebDAV

Otra de las opciones disponibles para arrancar un servidor de calendario es la puesta en marcha de un servidor WebDAV, que aun sin proporcionar las opciones que se esperan de CalDAV, puede ejercer la funcionalidad necesaria para un servidor de calendario. Para poder comprobar estas opciones hemos elegido poner en marcha un servidor WebDAV en nuestras máquinas y poder comprobar su funcionamiento.

Mediante este se podrían disponer de las opciones anteriormente comentadas, que nos ofrece poseer un servidor Web HTTP con las funcinalidades WebDAV añadidas. Para ello, y revisando las diferentes opciones que se ofrecían, hemos decidido recurrir al proyecto Jakarta, que ofrece un conjunto de herramientas de software libre escritas en Java y forma parte del Apache Software Foundation. El proyecto que ofrece Jakarta con finalidades WebDAV es conocido como Slide.

Fig. 7 Logotipo de Slide

Slide se define como un contenedor Web que ofrece soporte para WebDAV con las ventajas y funcionalidades que esto significa, y en una de sus versiones viene empaquetado conjuntamente con una distribución del proyecto Tomcat.

Tomcat es un contenedor de servlets, páginas dinámicas escritas en JAVA que se ejecutan en un servidor Web y que responden a una petición realizada por un cliente.

Después de instalar el software sobre una máquina y realizar las diferentes pruebas para utilizarlo como repositorio Web, hemos conseguido utilizarlo como repositorio para el almacenamiento de datos iCalendar. No obstante si quisiéramos poder aprovechar más las opciones que nos ofrece este tipo de servidor deberíamos de desarrollar una capa de software intermedia entre cliente y servidor, aprovechando la potencia que nos da el hecho de tener un servidor Tomcat, y que se encargara de ejercer una gestión de la información y ofrecer diferentes opciones añadidas.

Hemos conseguido interoperar con el servidor mediante diferente software de calendario como Chandler y Mozilla Calendar. Hemos tenido problemas de estabilidad con Chandler que achacamos al elemento cliente, y con Mozilla Calendar hemos tenido un comportamiento bastante correcto aunque las opciones que se ejercían se limitaban a las de utilizar el servidor Web como un mero repositorio de ficheros en formato iCalendar.

3.3. Clientes de Calendario

Los clientes de Calendario son piezas de software que se encargan de interactuar con los servidores o repositorios de calendario La gestión de usuarios, es decir, que usuarios tienen acceso al calendario y cuales no, se debe realizar con mecanismos externos propios de los servidores.

3.3.1. Chandler

Chandler es una aplicación cliente de calendario creada por OSAF, y cuyo funcionamiento está destinado principalmente a interactuar con el servidor de calendario, anteriormente estudiado, Cosmo, pero pensando en la interoperabilidad con otros servidores.

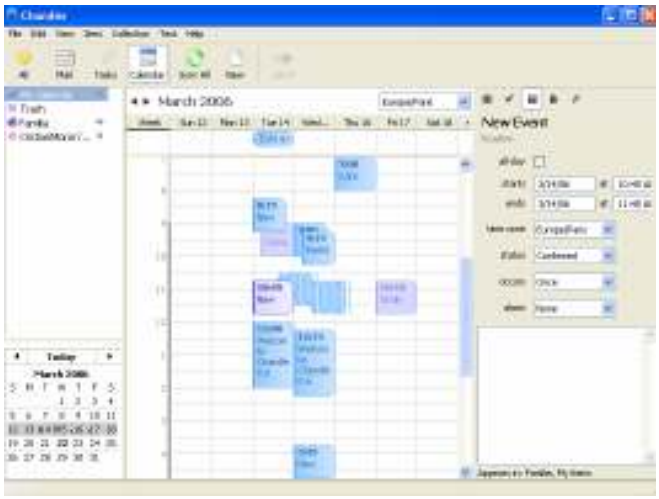

Fig. 8 Imagen de la interfaz principal de Chandler

Chandler, que va por su versión 0.6.1, posee diferentes características que la hacen ser una de las aplicaciones más completas del mercado en cuanto a funcionalidad.

Chandler funciona a partir de la creación de una cuenta en un servidor de calendario, preferiblemente Cosmo. El cliente permite crear nuevos eventos, tareas o notas y diferentes calendarios sobre los repositorios de calendario creados.

La filosofía de la aplicación se basa en la existencia de un único espacio de tiempo de usuario pero un conjunto de calendarios posibles a gestionar sobre este tiempo. Así pues se pueden compartir diferentes calendarios mediante la adquisición de sus diferentes eventos sobre el tiempo del usuario, mostrándose los conflictos de tiempo cuando se solapan diferentes actividades.

Uno de los puntos complicados para una aplicación cliente de calendario es la forma de tratar la gestión de usuarios. La aplicación permite que un usuario que posea un nombre de usuario y contraseña pueda acceder al repositorio, y tenga acceso a todos los datos y opciones posibles. Así permite que este usuario, que podría ser catalogado como usuario administrador, obtenga un acceso a los recursos mediante la generación de unas URLs que apuntan al calendario para más tarde distribuirlos dependiendo de los permisos que le quiera ofrecer a otro usuario con el que quiera compartirlo. La URL que se genera puede ser de dos tipos. Puede ser únicamente lectura, permitiendo a los usuarios que accedan a estos recursos consultar los eventos con permisos únicamente de lectura para conocer los tiempos libres/ocupados de los usuarios. O bien se puede extraer una dirección de lectura y escritura, de forma que a los usuarios que les entreguemos esta dirección pueden acceder y modificar la información existente en los calendarios.

Para los calendarios, Chandler permite una sincronización automática de los eventos, basándose en depositar la última información en el servidor, así como en descargar la información que algún otro usuario haya depositado con anterioridad. Para ello el cliente puede elegir que elementos de los existentes desea sincronizar y realizarla seleccionando la acción pertinente en la interfaz.

Así pues antes de publicar los datos la aplicación se descarga la última versión en el servidor, y posteriormente se depositan los nuevos datos.

El principal problema de la aplicación de calendario Chandler radica en los diferentes problemas y fallos que la hacen ser una aplicación inestable. La aplicación tiene múltiples problemas y puede llegar a no volver a ponerse en marcha sin ningún motivo aparente, debiéndola reinstalar para poderla utilizar.

3.3.2. Mozilla Calendar

Mozilla Calendard es el cliente de libre distribución que viene empaquetado para ser utilizado con el software de las distribuciones de Mozilla. En su última versión, la 1.5 puede ser utilizado conjuntamente con:

- Mozilla Firefox
- Mozilla Thunderbird
- Seamonkey
- Mozilla Application Suite

No obstante Mozilla también ha empaquetado su cliente de calendario en una aplicación individual que no necesita de las anteriores para funcionar denominada Mozilla SundBird. Utilizar Mozilla Calendar en vez de SundBird ofrece alguna ventaja como la de disponer alarmas vía correo electrónico en caso de utilizarse conjuntamente con Mozilla ThunderBird, pero su funcionalidad es básicamente la misma.

Mozilla ofrece una correcta e intuitiva interfaz de usuario que permite una fácil navegación por sus contenidos. Así como la fácil interacción con la aplicación permitiendo múltiples opciones.

Fig. 9 Interfaz principal de Mozilla Calendar

Mozilla Calendar permite la creación de calendarios, así como la creación en ellos de nuevos eventos y tareas. Por otra parte, se permite, la suscripción a

calendarios remotos ya existentes y depositados en un servidor de calendario. En este punto hemos comprobado su interoperabilidad.

Para suscribirse a un calendario remoto necesitas aprovisionar el repositorio donde se encuentra, conociendo su URL. En el caso de Cosmo, esta debe haber sido obtenida por medio de su cliente Chandler, o en el caso de un servidor WebDAV deberíamos aprovisionar la URL del repositorio que se haya aprovisionado.

Mozilla a diferencia de Cosmo, no ofrece la posibilidad de sincronización de eventos y utiliza el servidor de Calendario como un repositorio donde depositar sus datos de calendario. De hecho el cliente de calendario no posee ninguna opción de gestión, esta tarea se deja completamente en manos de la aplicación servidora.

Uno de los puntos destacables del servidor es que ofrece un amplio abanico de interoperabilidad. Mozilla Calendar trabaja perfectamente con un servidor WebDAV, o un servidor CalDAV. Así mismo hemos intentado trabajar con la cuenta creada en el servidor Cosmo, y hemos podido descargar los eventos en él introducidos sin ninguna complicación. No obstante hemos tenido problemas al intentar publicar eventos en él que achacamos a fallos del servidor. Como otra característica añadida, el cliente permite exportar e importar objetos de tipo calendario iCalendar desde y hacia ficheros locales, siendo almacenados en su formato estándar.

3.4. Cliente/Servidor integrado de Calendario

Otra de las piezas de software que existen en el mundo del software de calendario son los Clientes/Servidor integrados de calendario. Estos elementos intentan aglutinar las ventajas de un servidor de calendario conjuntamente con la funcionalidad de un cliente, intentando aprovechar las opciones de configuración que les permite ser un software integrado. Estos elementos permiten la creación por parte del usuario de un calendario o un repositorio de ellos en un servidor, y permiten la consulta de sus datos mediante un cliente Web. Su interoperabilidad suele ser muy limitada, y no se preocupan porqué un usuario pueda acceder desde un programa externo, pero ofrecen múltiples elementos de gestión de usuarios que lo hacen ser una alternativa interesante.

3.4.1. Bedework

Bedework es una aplicación Cliente/Servidor integrada de calendario de código abierto creado por la universidad de Washington y destinado en su desarrollo para ser utilizado por organizaciones educativas para facilitar la gestión de sus calendarios. La aplicación Bedework ha sido escrita en Java y viene empaquetada conjuntamente con un servidor Web Tomcat. El desarrollo de Bedework va por su versión 3.0.

La interoperabilidad que ofrece Bedework es más limitada que la que nos ofrece Cosmo, dado que el acceso al servidor se basa principalmente en el acceso Web, siendo su principal cliente cualquier navegador Web mediante el cual se pueda tener acceso a los eventos públicos o bien personales mediante una autenticación previa. Por tanto, la interoperabilidad existente en la actualidad, se basa en la posibilidad de exportar e importar los eventos en formato estándar iCalendar y compartirlos con otros usuarios que utilicen aplicaciones que trabajen con este formato, aunque se está trabajando de forma prioritaria para poder ampliar estas opciones en próximas versiones.

Bedework, en su interfaz principal permite el acceso a eventos públicos, fruto de un calendario publico, a eventos privados que serán propios de los usuarios registrados, y a una interfaz de gestión únicamente destinada a usuarios administradores.

Fig. 10 Interfaz principal de Bedework

Por tanto se pueden distinguir tres perfiles perfectamente definidos en la aplicación, que se corresponde con las tres acciones disponibles en la interfaz principal. El usuario público accederá a la información de carácter público, y lo podrá hacer sin realizar ninguna autenticación en el sistema. Sus posibilidades en la interfaz están limitadas a únicamente consultar eventos de carácter público de la organización. El usuario privado debe autenticarse en el sistema mediante un nombre y contraseña, y posee información de diferentes directorios dependiendo de los calendarios a los que él haya sido suscrito por un usuario administrador. El usuario puede suscribirse a diferentes calendarios para los que quiere recibir información, por tanto poder consultar los eventos y poder introducir nuevos eventos dentro del calendario. Un usuario administrador se encarga de, entre otras tareas, gestionar los recursos de los diferentes directorios gestionando los calendarios así como los usuarios.

Como punto fuerte, Bedework permite una alta personalización de la interfaz de calendario mediante la existencia de diferentes plantillas que le dan un aspecto muy diferente dependiendo de la elección realizada por el usuario.

La plataforma permite una gestión de los permisos que poseen los usuarios sobre los datos, siempre por parte de un usuario administrador, a los que

pueden acceder mediante la creación de grupos y calendarios destinados para estos grupos, así mismo, los calendarios personales requieren de una autenticación para poder ser modificados y el usuario accederá a los recursos que su perfil le permita.

En cuanto a sincronización de datos, no existe un mecanismo de sincronización de datos pero se están desarrollando diferentes métodos para ello. Los métodos de autenticación presentes se basan en mecanismos externos como la autenticación propia de Tomcat, aunque Bedework permite la interacción con bases de datos.

Como principal punto débil cabe destacar una interfaz poco lograda, con faltas de información y poco intuitiva que hace de su comprensión una ardua tarea para el usuario.

3.4.2. Kiko

Kiko es una aplicación Cliente/servidor integrada de calendario basada en Web que ofrece amplias opciones y una interfaz intuitiva y funcional, así como una eficiente gestión de los recursos.

Para acceder al repositorio, como primer paso se debe crear una cuenta en el servidor Kiko mediante el aprovisionamiento de un nombre de usuario su correspondiente contraseña y su correo electrónico. Estos datos deben de ser únicos en la plataforma. Una vez el usuario ha creado su cuenta puede autenticarse en la plataforma y acceder al repositorio de calendario y comenzar a trabajar con Kiko. Por tanto como posible usuario únicamente distinguimos a un usuario autenticado en el sistema. La cuenta de correo electrónico aprovisionada, permitirá recibir alertas ante nuevos eventos introducidos en los calendarios.

Un usuario autenticado dispone de diferentes acciones a realizar, puede introducir nuevos eventos como reuniones, tareas o alarmas de notificación para estos eventos, y para cada uno de estos eventos introducidos posee la posibilidad de elegir con que grupo de usuarios quiere compartirlos, dándoles visibilidad sobre los eventos. Puedes mostrarlo a un grupo determinado de usuarios, puedes mantenerlos privados y solo conocerlos tú o bien puedes hacer públicos y que la gente sepa que dicho tiempo lo tienes ocupado por la actividad. Así mismo en estos calendarios que se comparten los usuarios pueden interaccionar, añadiendo o posponiendo nuevos eventos.

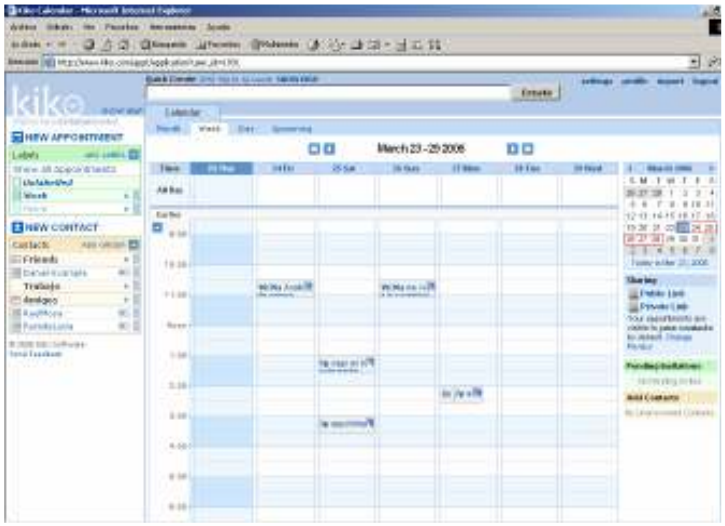

Fig. 11 Interfaz principal de Kiko

Desde la interfaz principal de la aplicación el usuario puede conseguir una URL pública perteneciente a un calendario conjuntamente con los eventos publicados en los calendarios que el usuario posee en su cuenta. Gracias a esta URL, que muestra todos los eventos propios del usuario y que tiene consideración de públicos, los demás usuarios pueden consultar visualmente el conjunto de tiempos libres/ocupados que posee el usuario. Un calendario público contiene los eventos que un usuario permite que todo el mundo conozca, es meramente informativo y ningún usuario podrá introducir nuevos eventos en él.

Kiko utiliza la gestión de grupos y usuarios, como mecanismo para compartir calendarios. Un usuario de Kiko puede crear diferentes calendarios asociándolos con otros usuarios mediante la creación de un grupo específico para ellos. De esta forma un usuario puede poseer diferentes calendarios donde interactuaran los usuarios a los que se les ha dado permiso. Kiko se basa en la solicitud por parte de cada usuario, del calendario al que desea pertenecer. Una vez solicitado, la petición llega a manos de un usuario administrador, mediante la notificación del evento en su interfaz, y donde se encargará de aceptarla o denegarla. Un ejemplo de uso, sería la creación de diferentes grupos como Trabajo, Familia, Amigos, que supondría la creación de estos tres tipos de calendario, y se colocarían los contactos pertinentes en cada uno de estos grupos asignándoles los calendarios.

Uno de los aspectos negativos de la aplicación reside en la sincronización, la aplicación no posee un método de sincronización directo sino que sincroniza con los calendarios del resto de usuarios al iniciar una sesión actualizándose los datos que hayan podido ser introducidos por estos usuarios hasta el momento. Por tanto, y como hemos comprobado, se pueden llegar a perder eventos si dos usuarios trabajan en el mismo momento.

El principal punto débil de Kiko es la interoperabilidad. Hemos intentado acceder a los recursos publicados con los diferentes clientes de calendario que poseíamos pero hemos visto que no se podía ni leer ni escribir en los repositorios. La única forma de acceder a estos es mediante un navegador

Web. No obstante permite importar datos en formato calendar y su predecesor vCalendar para poder ser incorporados al calendario, y poder compartir información pero no permite exportar los datos propios de la aplicación en ninguno de estos formatos.

3.4.3. Google Calendar

Google Calendar es el nuevo servicio servidor/cliente de calendario lanzado por la empresa americana Google, en su lucha por la creación e integración de nuevos servicios. Para su acceso un usuario únicamente necesita de una cuenta Google, aprovisionada por una cuenta de correo de Gmail.

La aplicación posee una interfaz muy intuitiva y funcional que se asemeja mucho a la proporcionada por la aplicación Kiko.

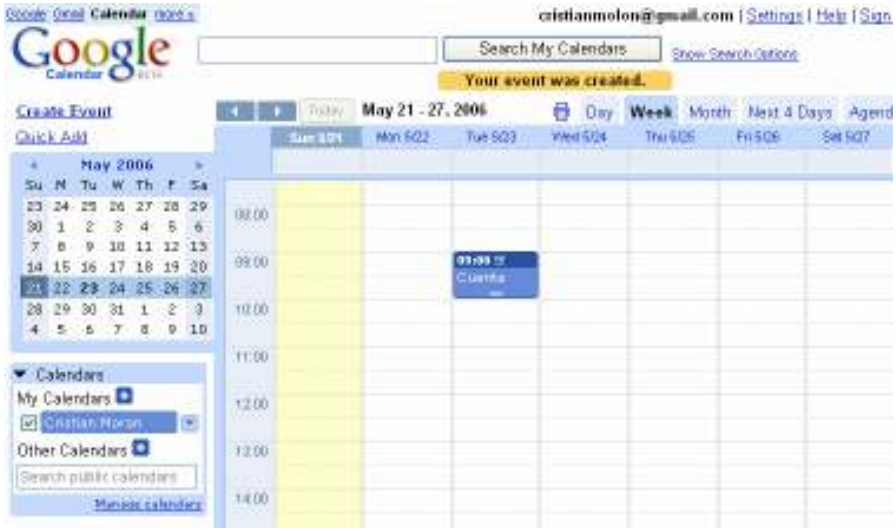

Fig. 12 Interfaz principal de Google Calendar

La interfaz permite múltiples opciones como la creación de eventos, la creación de diferentes calendarios, así como la creación de diferentes grupos de usuarios. Estos grupos de usuarios son utilizados para la gestión de los usuarios y engloban a los usuarios que posee cada calendario.

En cuanto a la gestión de usuarios, Google Calendar se comporta diferente a Kiko. En la aplicación es el usuario administrador quien debe invitar a otros usuarios a formar parte de su calendario, siempre que posean una cuenta en Google. Así mismo si un usuario desea tener acceso a un calendario, puede solicitarlo mediante una petición vía correo electrónico al administrador del calendario, que será depositado en el buzón de entrada de su cuenta de Gmail.

Google Calendar también presenta la opción de obtener calendarios públicos, donde conociendo el administrador de un calendario o la URL de este, se pueda tener acceso únicamente de lectura a la información publicada de tiempos libres/ocupados de este.

El punto más débil de Google Calendar se basa en la interoperabilidad. Google Calendar únicamente puede ser accedido mediante el navegador Web, y no mediante otras aplicaciones clientes de calendario. No obstante ofrece posibilidades de importación de datos en formato iCal o CSV (formato propio de los datos de Microsoft Outlook), pero no permite exportar los datos en ninguno de estos formatos.

En cuanto a sincronización Google Calendar, proporciona un interesante mecanismo de sincronización. Las conexiones se van actualizando de forma periódica y se van actualizando los eventos que en ellas residen, descargándose y publicándose periódicamente y de forma automática los datos.

Por otra parte, uno de los puntos más fuertes y donde Google no tiene competencia es en la integración que piensa ofrecer con otros productos de su familia como Gmail, Google Desktop, Google Reader e incluso Google Map pudiendo incluso consultar el lugar exacto de la reunión en un mapa.

Google Calendar viene avalado por el liderazgo de los productos que Google saca al mercado, con una interfaz vistosa y amigable y un completo motor de búsqueda, promete convertirse en el más famoso de los clientes de calendario actuales.

3.5. Conclusiones

Después de haber realizado diferentes pruebas nos hemos encontrado con que las aplicaciones de calendario poseen múltiples problemas, que parecen producirse por no estar muy bien depuradas y presentan diversos errores que provocan fallos en su utilización. También cabe destacar la escasa información que acompaña a los programas que muchas veces resultan poco intuitivos y difíciles de gestionar, lejos de los requisitos esperados para una aplicación calendario y el público al que va destinado.

Todo y esto cabe suponer que mejoraran, dado que actualmente se encuentran en una fase de mejora y pruebas. Así mismo cabe destacar como se van haciendo un hueco cada vez mayor las aplicaciones cliente/servidor, dado que al integrar ambas partes y ser fácilmente accesible mediante un navegador Web, suponen la alternativa más eficaz y sencilla para el usuario. Suponemos que estas aplicaciones dominaran en un mercado domestico, pero que no se impondrán en el mercado empresarial, donde la integración con otros sistemas y funcionalidades internas primaran sobre otros factores.

En la siguiente tabla se repasan los principales puntos fuertes y débiles de cada aplicación que más adelante estudiaremos incorporar a nuestra aplicación.

Tabla 3- Comparación del software Calendario.

	Puntos fuertes	Puntos débiles
Cosmo	-Ofrece interoperabilidad con	-Poco estable.

	diferentes sistemas de calendario. -Por cada usuario se crea un repositorio donde almacenar la información. -Gestión de acceso. A la información, solo se podrá acceder por medio de usuario y contraseña.	-No ofrece sincronización. -La forma de compartir recursos con el resto de usuarios se basa en la confianza (el usuario comparte su URL y usuario y contraseña con los usuarios que desea).
Servidor WebDAV	-Ofrece las características propias del protocolo: control de versiones, control de bloqueo, edición... -Ofrece funciones de repositorio. -Permite desarrollar una capa de software para proporcionar más funcionalidades al gusto del desarrollador.	-Es un simple servidor que no ofrece ninguna opción añadida que no sea la de repositorio, y que si se desea ofrecer, el desarrollador deberá trabajar en ella.
Chandler	-Permite una actualización de la información, así se evita que el usuario posea información desfasada y trabaje con ella.	-Poco estable. -La actualización se basa en un método propio con lo que no resulta interoperable. Así mismo no es automática.
Mozilla Calendar	-Muy interoperable. -Muy estándar. -La mayor parte de la lógica se deja al lado servidor.	-Implementa la funcionalidad básica del cliente, la lógica debe de estar en el servidor. Por tanto, no ofrece opciones de sincronización de datos ni gestión de usuarios ni autenticación.
Bedework	-La forma de compartir calendarios se basa en usuario y contraseña y la gestión de permisos gracias a la existencia de diferentes roles. -Nivel de personalización alto.	-No interoperable, funciona únicamente con navegador Web. -No ofrece sincronización de datos, (posibles solapamientos). -Interfaz poco intuítiva.
Kiko	-La forma de compartir calendarios se basa en usuario y contraseña y la gestión de permisos gracias a la existencia de diferentes roles. -Alertas por medio de correo electrónico.	-No interoperable, funciona únicamente con navegador Web. -No ofrece sincronización de datos, (posibles solapamientos).
Google Calendar	-La forma de compartir calendarios se basa en usuario y contraseña y la gestión de permisos gracias a la existencia de diferentes roles. -Buen modelo de sincronización automática de datos, que evita solapamientos. -Permite interacción con diferentes productos de Google.	-No interoperable, funciona únicamente con navegador Web.

CAPÍTULO 4. PROPUESTA DE CREACION DE UN SISTEMA CALENDARIO

En el siguiente capítulo se repasa la propuesta y los elementos escogidos para la creación de un sistema de calendario que nos permita probar nuevas funcionalidades y poner en marcha un sistema que cumpla con las características que nosotros deseamos.

4.1. Requisitos funcionales

El objetivo principal de la plataforma es la creación de un sistema de calendario, intentando cubrir las necesidades que creemos más interesantes en una plataforma de calendario, después de haber analizado los diferentes elementos software del mercado. Así pues, a partir de este análisis y como primer paso hemos definido los requisitos funcionales de nuestra plataforma:

- Gestión de usuarios: Creación de una cuenta de usuario en la plataforma para poder acceder al servicio, siendo el usuario y el calendario datos únicos en la plataforma. Para cada usuario se creará un repositorio de datos donde almacenar cada uno de sus calendarios en el interior. Un usuario podrá poseer más de un calendario con una misma cuenta.
- Control de acceso: La plataforma proporcionará un control de acceso para los usuarios, que deberán identificarse para poder acceder a los recursos.
- Posibilidad de autenticación por grupo sin tener que dar de alta a cada uno de los usuarios por separado, sino a un perfil de usuario. Útil para organizaciones de múltiples usuarios como por ejemplo universidades, donde los alumnos podrían entrar a consultar eventos únicamente conociendo cual es su rol.
- Existencia de una jerarquía de roles en la plataforma para cada uno de los calendarios existentes. Así se proporcionará un método para compartir información basado en peticiones de acceso. Existirán: usuario administrador, usuario pendiente, usuario.
- Plataforma interoperable que permita el acceso con diferente software cliente y estándar de calendario. Para ello utilizaremos un estándar en el formato de los datos de gran utilización y aceptación como iCalendar y un protocolo ya conocido como WebDAV.
- Sincronización en los datos y protección de sobrescritura de la información. Gracias a la sincronización se evitará que se introduzcan datos desfasados en la plataforma. Mediante la protección de sobrescritura de la información se consigue evitar que unos usuarios se solapen con otros, además añadiremos un control en los cambios de los recursos.
- Interacción entre usuarios y plataforma mediante el envío de correos electrónicos configurables, que permitan informar al usuario de los eventos acontecidos.
- Configuración de la plataforma mediante un fichero externo.

4.2. Escenario propuesto

Para poder cumplir con las características expuestas en el apartado anterior, el elemento servidor se dividirá en dos grandes bloques, que funcionan por separado, pero son necesarios para poder ofrecer un servicio completo de calendario. Por un lado, tenemos la parte de gestión de la aplicación, compuesta por una interfaz para la interacción con el usuario y la administración de la información de calendarios. Esta interfaz debe permitir la autenticación de un usuario en la plataforma, la posible creación de sus calendarios y la gestión de los usuarios que poseen acceso a sus calendarios.

Por otro lado, se debe crear una capa de software en el servidor Web, para permitir que los clientes puedan interactuar con el servidor de forma transparente, proporcionando diferentes funcionalidades añadidas cuando se intenta acceder a la información de calendario almacenada desde un cliente de correo. Es importante subrayar la idea de la transparencia de nuestra capa, dado que interactuará con el cliente de calendario Mozilla sin que el usuario tenga noticia de su actuación.

Para ello hemos decidido utilizar una arquitectura cliente-servidor, donde los elementos intenten trabajar por separado y sean lo más interoperables posibles, pudiendo acceder a los datos con diferentes clientes estándar ya existentes. Para nuestro desarrollo y pruebas, hemos considerado que la plataforma debe estar formada en su parte cliente por la aplicación Mozilla Calendar y el servidor debe permitir el acceso y un completo uso de sus posibilidades con dicho cliente. La elección de Mozilla Calendar se sustenta en que la aplicación creada por Mozilla cumple con los requisitos de tratarse de software libre, funcionar correctamente y de forma estable, y ser el cliente más estándar e independiente de los analizados.

Por tanto con la elección de un cliente fijo y existente en el mercado, nuestro desarrollo de funcionalidades se centra en la parte servidora de la plataforma. Para ello partimos de la idea de modelar los objetos calendario como recursos HTTP y ampliar las funcionalidades posibles sobre ellos mediante el protocolo WebDAV. Por tanto escogeremos, como base, un servidor WebDAV que cumpla con dos condiciones importantes. La primera tratarse de software libre, y la segunda permitir la creación de una capa de software en el elemento servidor que nos permita extender las funcionalidades existentes.

Para realizar las tareas de servidor WebDAV escogemos el servidor Slide del proyecto Jakarta que hemos repasado anteriormente. En una de sus distribuciones Slide viene empaquetado conjuntamente con el servidor Tomcat, cosa que nos permitirá la creación de una lógica en la parte servidor, basada en el lenguaje de programación Java.

Slide es fácilmente instalable y configurable, de forma que hemos configurado la plataforma para permitir accesos de lectura y escritura a los datos almacenados, así como realizar una gestión de usuarios y una autenticación a los recursos. La autenticación es un elemento clave en una aplicación de este

tipo, dado que se debe constatar constantemente los derechos que poseen los usuarios sobre los recursos que solicitan. Hemos configurado el servidor para que realice la autenticación contra los datos que son aprovisionados y almacenados en la base de datos de la plataforma.

4.3. Roles en la plataforma

Como hemos definido anteriormente, la plataforma estará conformada por dos partes. Una parte de gestión y una interfaz entre cliente y servidor, por tanto deberemos definir los actores o tipos de usuarios que deben de actuar en el sistema.

4.3.1. Roles en la plataforma de gestión

En la plataforma de gestión se han definido cuatro tipos de usuarios diferentes, que poseen diferentes permisos que tiene cada uno de ellos para realizar ciertas acciones.

-Usuario sin autenticar: Es un usuario que navega por Internet y que no ha dado de alta ningún calendario en el sistema, o no se ha autenticado aún. Tiene acceso a la información que se muestra en la pagina principal, pero necesitará autenticarse si quiere tener acceso a su información. Puede darse de alta o pedir una solicitud para formar parte de un calendario.

-Usuario autenticado: Es un usuario que se ha autenticado en la plataforma mediante la introducción de su nombre de usuario y contraseña y su posterior verificación. Posee las mismas opciones que un usuario sin autenticar y además puede consultar la información de sus calendarios, tal y como los usuarios que posee o los eventos, contactar con los usuarios, personalizar sus opciones y desconectarse para volver a ser un usuario no autenticado.

-Usuario autenticado pendiente: Es un usuario que únicamente existe en la plataforma de gestión. Se trata de un usuario que ha sido autenticado en el sistema y que esta pendiente del resultado de su solicitud de acceso a un calendario. El usuario posee las mismas opciones que un usuario sin autenticar y además puede consultar el estado de la solicitud del calendario pendiente.

-Usuario autenticado administrador: Es un usuario que únicamente existe en la plataforma de gestión. Se trata de un usuario que ha sido autenticado en el sistema y que posee un calendario cuyos permisos son de administrador. Este usuario posee las mismas funcionalidades que un usuario autenticado, y además puede aceptar o denegar peticiones de acceso, por parte de otros usuarios, a formar parte de un calendario. Así mismo puede conceder privilegios de usuario administrador.

4.3.2. Roles en la interfaz entre cliente y servidor

En la plataforma de interfaz entre cliente y servidor actúan los actores que intentan acceder a los datos mediante el cliente de calendario. Existen tres actores posibles en el sistema:

-Usuario sin autenticar: Es un usuario cuya única opción que posee es solicitar un calendario y aprovisionar su nombre y contraseña. Si lo hace correctamente se transforma en un usuario autenticado.

-Usuario autenticado: Es un usuario que ha sido autenticado y puede descargar el calendario que haya solicitado siempre que tenga permisos para él. Así mismo puede publicar eventos e información en él.

-Usuario autenticado administrador: Es un usuario que posee la misma funcionalidad que un usuario autenticado, teniendo acceso a sus calendarios y pudiendo publicar eventos, pero además puede eliminar un calendario, del que sea administrador, de la plataforma.

CAPÍTULO 5. Análisis y diseño

Una vez instalado y configurado nuestro servidor y adentrados en el entorno de programación, procedemos a la etapa de diseño de nuestra aplicación. La etapa de diseño en el desarrollo de un sistema consiste en presentar el diseño lógico del sistema. De esta forma se especifica cada uno de los componentes que lo forman y las relaciones que mantienen entre ellos. Para ello nos basaremos en el lenguaje unificado de modelado UML, que nos proporciona la metodología y semántica para representar sistemas orientados a objetos.

Para tener un diseño adecuado a los requerimientos del sistema, es necesario hacer previamente una descripción de la aplicación. En el primer apartado se identifican las principales funcionalidades presentes en el sistema, apoyado por un diagrama de casos de uso, utilizando la citada metodología UML que nos permite visualizar las funcionalidades presentes en la plataforma para cada uno de los actores identificables. Seguidamente se exponen los diagramas de clases UML, que nos permite visualizar la estructura de la aplicación y su organización. Estos diagramas son completados con una descripción para cada una de las clases y por ultimo, se expone la relación que mantienen entre ellas mediante un diagrama de secuencia UML.

5.1. Diagrama de casos de uso

Para desarrollar el diagrama de casos de uso se han identificado previamente los actores del sistema, que equivalen a los roles explicados en el apartado *Roles en la plataforma*. Una vez se conocen los actores, se listan todas las acciones que éstos pueden realizar. Cada acción será un caso de uso, por ejemplo: “Introducir un nuevo evento” o “Crear un nuevo calendario”. Hemos realizado un diagrama de casos de uso para cada una de las partes que posee la plataforma, en un primer momento presentamos el perteneciente a la plataforma de gestión conjuntamente con sus acciones posibles y posteriormente realizamos lo propio con la parte de interfaz entre servidor y cliente.

5.1.1. Diagrama de casos de uso del elemento gestor

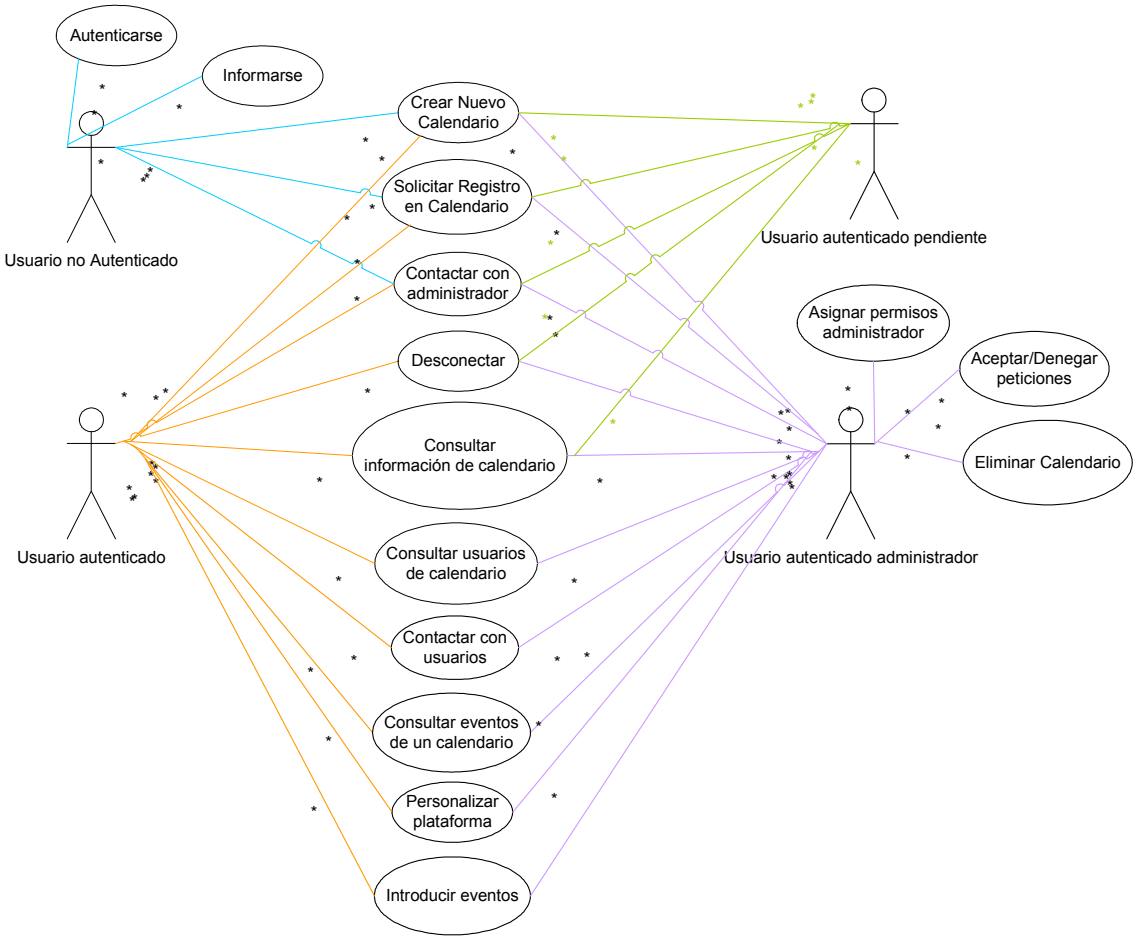

Fig. 13 Diagrama de casos de uso del elemento gestor

5.1.1.1. Autenticarse

El actor Usuario no conectado posee la funcionalidad de acceder a la plataforma mediante el aprovisionamiento de nombre de usuario y su correspondiente contraseña, y poder disfrutar de las posibilidades que se ofrecen. Una vez autenticado el usuario puede acceder a las opciones que se le ofrecen para gestionar su información.

El mecanismo de autenticación de un usuario, consiste obligatoriamente en la creación o la suscripción de un calendario. Un usuario que no posea calendarios será eliminado de la plataforma. A su vez, un usuario puede poseer más de un calendario con los que puede interactuar con la plataforma.

5.1.1.2. Informarse

El actor usuario no autenticado puede acceder a la información proporcionada por la plataforma. En ella se explica como acceder al sistema y las ventajas que ofrece la el sistema.

5.1.1.3. Crear nuevo calendario

Cualquier tipo de usuario puede crear un nuevo calendario que se le añadirá a sus calendarios existentes si ya existe el usuario o bien servirá para dar de alta al usuario en la plataforma. Si el usuario crea este nuevo calendario automáticamente será designado como administrador de éste y como administrador recibirá la consulta de acceso de otros usuarios y poseerá las funcionalidades de un usuario autenticado administrador.

5.1.1.4. Solicitar registro en calendario

Cualquier tipo de usuario puede solicitar su acceso a un calendario ya existente. Para ello deberá conocer el nombre de calendario y esperar la autorización del usuario administrador del calendario. Si el usuario ya existe en la plataforma, se le añadirá el nuevo calendario constando como pendiente. Si el usuario no existía en la plataforma, será dado de alta con el calendario como pendiente. En caso de ser denegado el permiso, será eliminado de esta.

5.1.1.5. Contactar con administrador

Cualquier tipo de usuario puede consultar con el administrador de la plataforma acerca de alguna duda de su funcionamiento. Para ello se le ofrece la posibilidad de enviarle un correo electrónico.

5.1.1.6. Desconectar

Un usuario que haya sido autenticado en la plataforma puede pasar a formar parte del grupo y funcionalidades de un usuario no autenticado mediante la desconexión de la plataforma, volviendo a la interfaz principal de presentación.

5.1.1.7. Consultar información de calendarios

Un usuario al autenticarse en la plataforma puede consultar los calendarios que el posee y el estado o rol que en ellos presenta.

5.1.1.8. Consultar usuarios de calendario

Un usuario autenticado, y cuya participación haya sido aceptada para el calendario, puede consultar los usuarios con los que comparte cada calendario, conociendo aspectos tal y como el nombre de estos usuarios, su rol en el calendario y su correo electrónico.

5.1.1.9. Contactar con usuarios

Mediante la consulta de los usuarios pertenecientes a los calendarios se le ofrece a los usuarios autenticados así como a los usuarios administradores la posibilidad de enviar un correo a cada uno de los usuarios con los que comparte calendario de forma individual, como un mensaje al grupo de usuarios.

5.1.1.10. Consultar eventos de un calendario

El actor usuario autenticado, así como el actor administrador pueden consultar los eventos pertenecientes a un calendario mediante la interfaz Web así como mediante su cliente de correo.

5.1.1.11. Personalizar plataforma

Un usuario autenticado puede configurar diferentes opciones en la plataforma como el conjunto de alarmas por correo electrónico que desea recibir.

5.1.1.12. Aceptar/Denegar peticiones de acceso a calendario

Un actor usuario autenticado administrador posee la funcionalidad de poder aceptar o denegar peticiones de acceso a calendario. Estas peticiones las recibirá en cuanto acceda a la plataforma, mostrándose el nombre de usuario y correo electrónico del individuo que solicita el acceso.

5.1.1.13. Asignar permisos de Administrador

Un actor usuario autenticado administrador posee la funcionalidad de asignar permisos de usuario administrador a otro usuario con el que comparta calendario. De esta forma ambos poseerán la posibilidad de aceptar o denegar peticiones de acceso a calendario repartiéndose las tarea de gestión de calendario.

5.1.2. Diagrama de casos de uso de la interfaz entre cliente y servidor

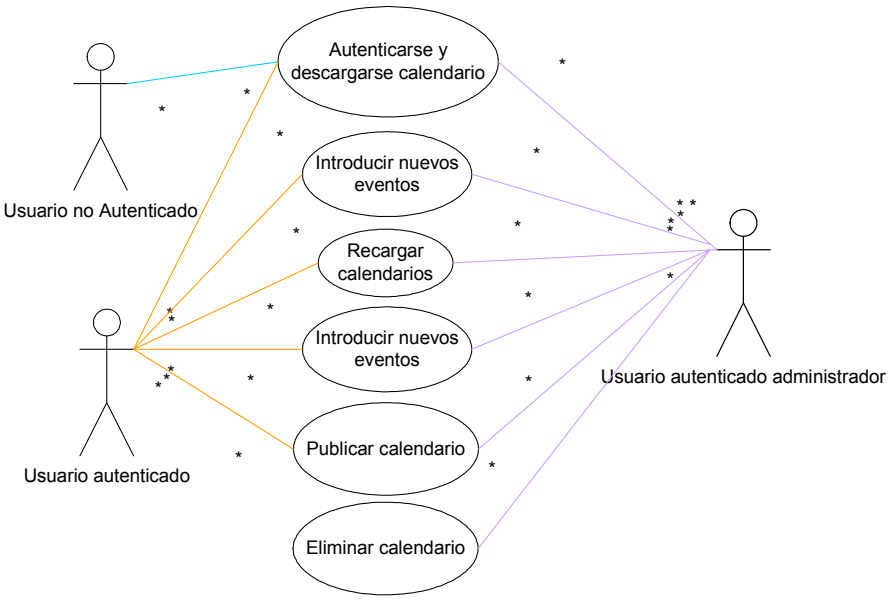

Fig. 14 Casos de uso de la interfaz entre cliente y servidor

5.1.2.1. Autenticarse y descargarse calendario

El actor Usuario no conectado posee la funcionalidad de acceder a la plataforma mediante el aprovisionamiento de nombre de usuario y su correspondiente contraseña, y poder disfrutar de las posibilidades que se ofrecen. Una vez autenticado el usuario puede acceder a las opciones que se le ofrecen para gestionar su información.

El mecanismo de autenticación de un usuario, consiste obligatoriamente en la creación o la suscripción de un calendario. Un usuario que no posea calendarios será eliminado de la plataforma. A su vez, un usuario puede poseer más de un calendario con los que puede interactuar con la plataforma.

5.1.2.2. Recargar calendario

Un usuario puede refrescar su calendario descargándose los últimos eventos que hayan sido publicados en el servidor después de la última descarga realizada, y anulando los eventos que hayan sido introducidos desde entonces.

5.1.2.3. Eliminar Calendario

Un usuario administrador posee la opción de eliminar su calendario. De esta forma todos los usuarios que posean este como propio lo perderán. Esta opción esta disponible mediante el cliente de calendario que utilice.

5.1.2.4. Introducir un nuevo evento en el calendario

Los actores usuario autenticado así como un usuario autenticado administrador pueden introducir nuevos eventos en la plataforma mediante su cliente de calendario. En la introducción de nuevos eventos se producirá un proceso de bloqueo y verificación de que la información introducida es la más reciente en el servidor y no hay sobreescritura.

5.1.2.5. Publicar calendario

Los actores usuario autenticado así como un usuario autenticado administrador pueden decidir subir al servidor toda la información que han introducido nueva con la opción de publicar calendario. De esta forma se sobrescribirá la información en el servidor.

5.2. Diagrama de clases

El diagrama de clases consiste en representar el diseño de la aplicación, mostrando las clases que conforman el sistema y sus interrelaciones. En el sistema, tendremos dos diagramas de clases diferentes según se trate de la plataforma gestión o bien de la interfaz entre cliente y servidor. De entre ellos es ilustrativo repasar el diagrama de clases de la interfaz entre cliente y servidor para hacernos una idea de la lógica que se esconde en la interoperabilidad de los clientes.

5.2.1. Diagrama de clases de la interfaz entre cliente y servidor

Como clase principal tenemos la clase Action que intenta ofrecer una interfaz para que los clientes puedan interoperar con ella. Así pues dependiendo del método HTTP con el que los clientes accedan a esta, el método invocado será uno u otro. Esta interfaz, pues, consiste en mapear los correspondiente métodos HTTP que nos interesan y realizar la acción correspondiente.

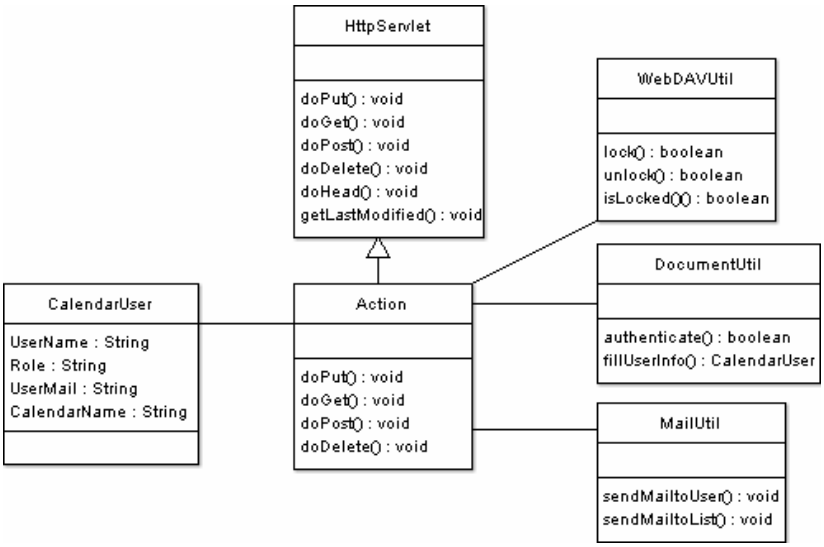

Fig. 15 Diagrama de Clases del cliente

La clase MailUtil proporciona los métodos para interoperar con el servidor de correo y proceder , por tanto, al envío de correos, a los usuarios que así lo hayan indicado, tanto de forma individual como a una lista de usuarios.

La clase DocumentUtil proporciona las funcionalidades para interactuar con la base de datos del fichero XML. En ella se encuentran funcionalidades como la autenticación, añadir información de calendario o eliminar un calendario ya existente.

La clase WebDAVUtil proporciona las funcionalidades propias del protocolo WebDAV. Encontramos la posibilidad de verificar si un fichero esta protegido o de protegerlo y desprotegerlo.

La clase HTTPServlet contiene los métodos básicos, para realizar las principales acciones HTTP por parte de un elemento servlet.

La clase Action sobreescribe la clase HTTPServlet en las acciones HTTP que a nosotros nos interesa y hereda o adquiere el resto de métodos que este presenta. Al sobrescribir estas acciones, permite interceder en las acciones deseadas correspondientes con los métodos Put, Get, Post y Delete de HTTP.

El método CalendarUser, se corresponde con un objeto del tipo usuario que engloba los datos que representan a un usuario como su nombre, su correo electrónico o el rol que posee en el calendario.

5.3. Diagrama de actividades

El Diagrama de Actividades permite representar flujos de trabajo como un conjunto de actividades que están conectadas a través de transiciones y que pueden verse sujetas a diferentes condiciones. Repasaremos tres de los principales flujos que puede comprender la actividad de un usuario y que clarificará tres de las opciones más importantes proporcionadas por la plataforma.

5.3.1. Diagrama de actividades para la creación de un calendario

Este diagrama de actividades posee como actor cualquiera de los diferentes tipos de usuarios presentes en la plataforma. Para suscribirse a un nuevo calendario, el usuario empieza por rellenar un formulario, del cual se verificará que sus datos son únicos en la plataforma, si estos datos son únicos se procederá a la creación de una carpeta de usuario y la posterior creación de un calendario vacío en formato iCalendar. Si el usuario no es nuevo en la plataforma pero ha aprovisionado su nombre y contraseña correspondiente a un usuario existente, se considerará que se trata de éste y se le añadirá un nuevo calendario a su repositorio. Para finalizar se le dará permisos de administrador, como creador de dicho calendario.

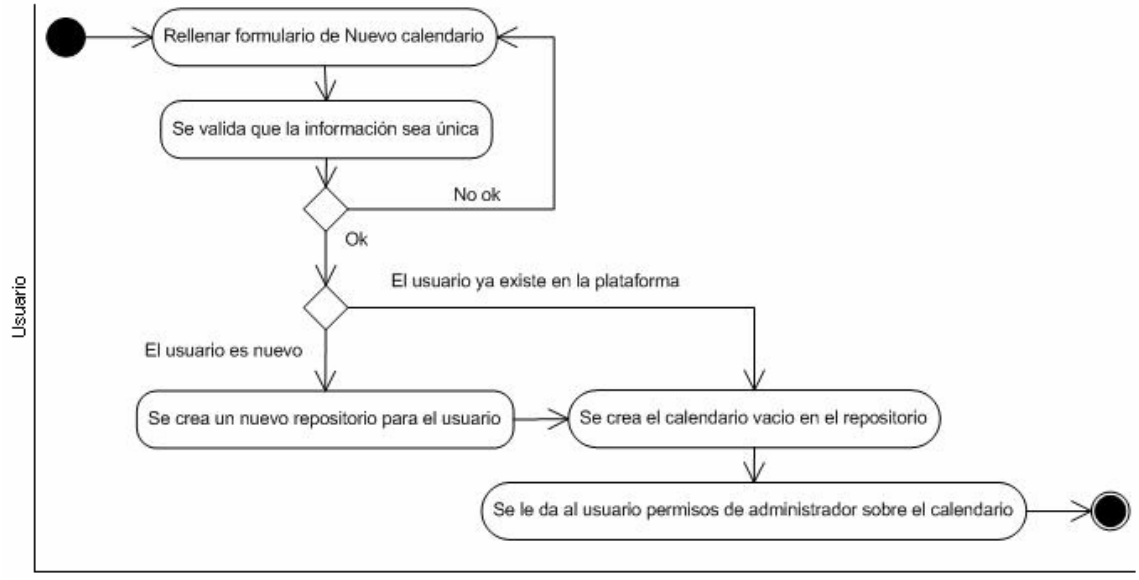

Fig. 16 Diagrama de actividades para la creación de calendario

5.3.2. Diagrama de actividades de un cliente de calendario

En la siguiente figura se representa la fase de registro de un usuario sobre un calendario ya existente en la plataforma.

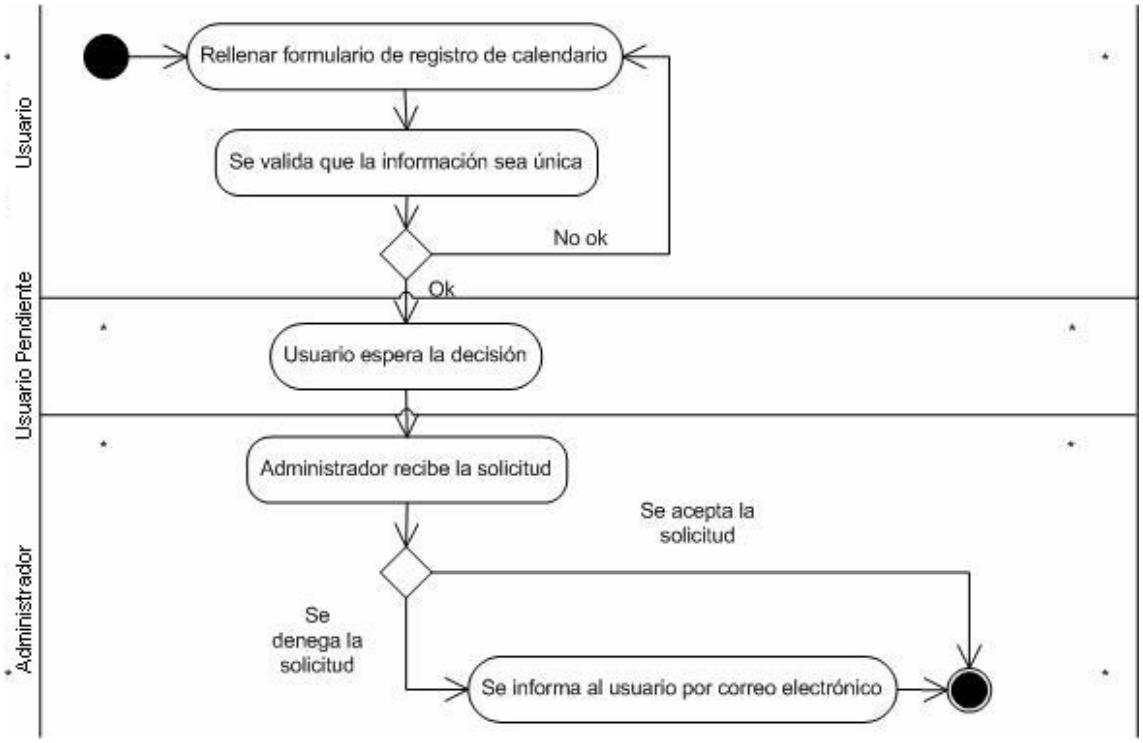

Fig. 17 Diagrama de actividades para un cliente de calendario

Un usuario ya sea registrado o no registrado, puede solicitar el acceso a un calendario. El siguiente paso es confirmar que este calendario se encuentra en la plataforma. Si no se encuentra en ella el usuario deberá de nuevo verificar la información introducida. Si se encuentra, el usuario pasará a ser un usuario pendiente y la solicitud irá a parar a manos del administrador que decidirá si acepta o deniega la solicitud. Si el administrador acepta la solicitud, el usuario podrá interactuar con el calendario pasando a ser un usuario registrado. Si por el contrario decide denegar la solicitud, el usuario será eliminado de la plataforma y se le enviará un correo electrónico indicándole el hecho.

5.3.3. Diagrama de actividades para la interacción con un cliente

En el siguiente diagrama se demuestran las principales funcionalidades que poseen los diferentes usuarios interactuando con la interfaz del servidor mediante un cliente de calendario.

Como primera acción un usuario debe autenticarse en la plataforma y descargarse la información de un calendario. Una vez descargado se distingue con una sesión al usuario y se almacenan sus datos para indicar que este usuario tiene permisos para interactuar con el servidor y el calendario descargado, y se bloquea el calendario para evitar que otros usuarios sobrescriban los datos, protegiéndolos.

Así pues, un usuario registrado o un usuario administrador, pueden realizar dos acciones comunes, introducir nuevos eventos en el calendario, o bien consultar únicamente los eventos. De introducir nuevos eventos, el usuario los creará en

local y una vez se den por bueno los cambios realizados publicará su calendario en el servidor. En publicar los datos, el servidor reconocerá los datos de sesión de usuario y permitirá sobrescribir la información que residía hasta el momento. Es importante anotar que como se ha explicado en apartados anteriores, y por motivos de eficiencia, se volverá a sobrescribir todo el fichero de calendario. Una vez sobrescrito se desbloqueará la información hasta entonces protegida, y se enviará un correo electrónico a los usuarios que así lo hayan configurado, indicando que el calendario ha cambiado.

De haber querido consultar los datos únicamente, se habrá puesto en marcha un temporizador que de haber pasado un tiempo determinado y configurable, desprotegerá los datos y permitirá que consumido este tiempo prudencial otros usuarios puedan tener acceso a ellos.

Si un usuario desea acceder a un calendario protegido, y para evitar que este piense que hay un error en la plataforma, o que los datos no son accesibles, se ha decidido devolver un código de estado HTTP del tipo 401 indicando que el usuario no está autorizado para tener acceso a estos recursos.

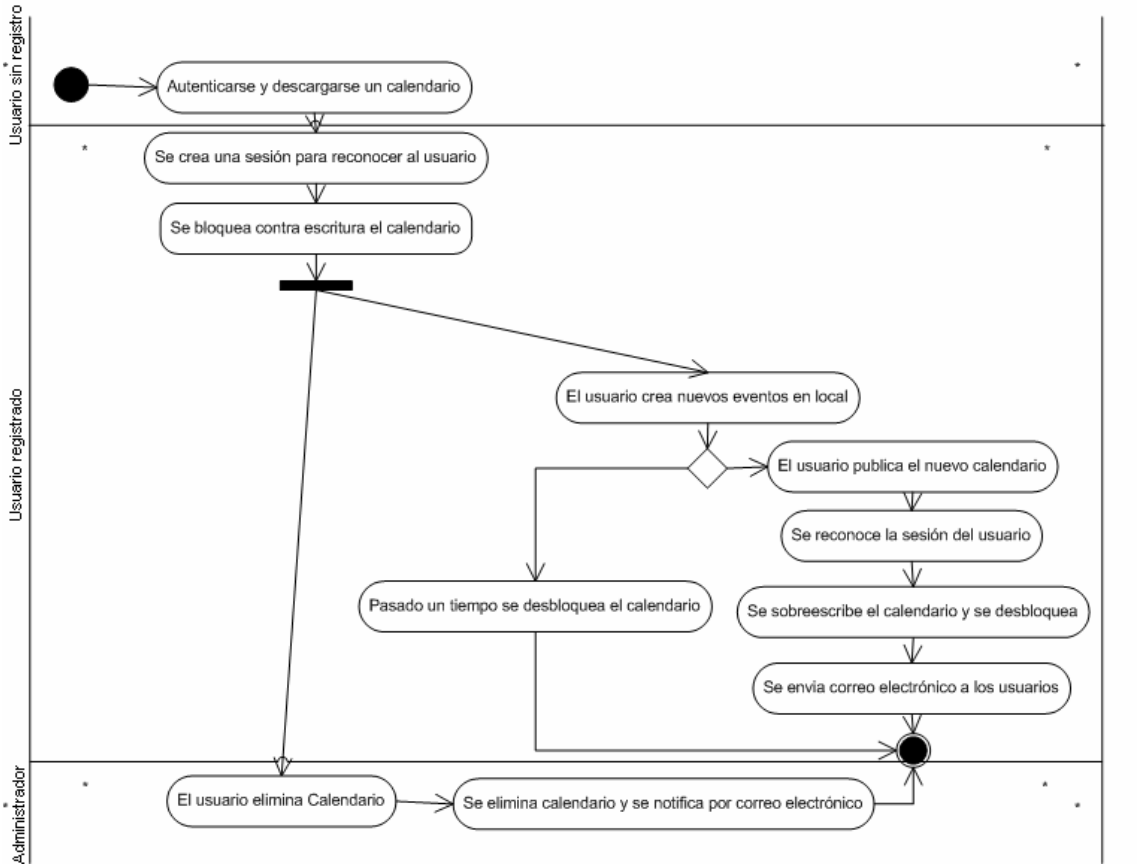

Fig. 18 Diagrama de actividades para la interacción con el cliente

CAPÍTULO 6. IMPLEMENTACIÓN Y PRUEBAS

En este capítulo se explican los aspectos principales relacionados con la implementación de la aplicación diseñada anteriormente, incluyendo las pruebas realizadas para asegurar el correcto funcionamiento del sistema.

6.1. *Tecnologías y herramientas utilizadas*

Las tecnologías y herramientas con las que se han implementado la aplicación son:

- **sdk 1.4.2:** versión de Java 2 Software Developer's Kit utilizada para el desarrollo de la parte servidora del sistema.
- **JavaScript:** Tecnología que nos permitirá comprobar en el lado cliente que los datos introducidos son correctos por los usuarios sin sobrecargar innecesariamente el servidor.
- **Eclipse SDK:** Plataforma de desarrollo escogida para desarrollar sobre la tecnología Java.
- **Librerías java**
  - Librería iCalj: Utilizada para la lectura de datos en formato iCalendar.
  - Librería WebDAV Slide Client: Utilizada para realizar las funcionalidades correspondientes al protocolo WebDAV.
  - Librería Java Mail: Utilizada para el envío de mensajes en formato correo electrónico mediante el aprovisionamiento de un servidor de correo SMTP.
- **Motor Servlets,** software que permite la ejecución de servlets o páginas dinámicas. El software elegido se corresponde con el producto de Jakarta Tomcat, que cumple con la especificación estándar definida por Sun Microsystems Servlets 2.2.
- **Microsoft Windows XP:** sistema operativo utilizado.

6.2. Estructura de la plataforma

La estructura de la plataforma implementada consta de tres capas:

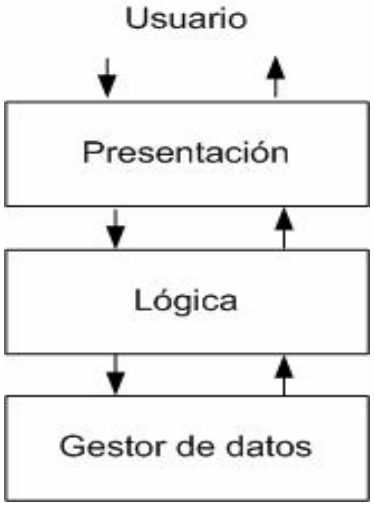

Fig. 18 Modelo de tres capas utilizado

- Capa de presentación del elemento gestor: Es la parte que ven los usuarios que para acciones de gestión acceden a la interfaz Web desde un navegador, estará conformada por código html y al igual que los clientes de calendario, interactuará con la capa lógica. En el elemento cliente de calendario esta capa la presentará el cliente escogido y no se deberá desarrollar, en nuestro caso Mozilla Calendar.
- Capa lógica: Corresponde al Servidor Web. Se denomina capa lógica ya que es donde se procesan los datos de los formularios y se realizan las acciones solicitadas llamando a los métodos de las clases de Java Desde las clases de Java se realizan las consultas a las bases de datos. El resultado final se le devuelve al usuario en su navegador mediante servlets o bien mediante una aplicación calendario.
- Capa de gestión de datos: La capa de datos para almacenar la información está compuesta por ficheros XML, y la lectura de estos ficheros será realizada por las librerías JAVA DOM incluidas en la versión del SDK de Java escogido.

6.3. *Capa de presentación*

Este apartado nos muestra las diferentes Interfaces gráficas creadas para nuestro cliente de calendario, destinadas a la interacción entre usuario y sistema. Así mismo se describen cuales son las cualidades de cada uno de los casos.

6.3.1. Interfaz gráfica inicial del elemento gestor

El primer contacto que se tiene con la aplicación por parte de un usuario que navega por Internet es el portal de bienvenida de nuestra aplicación calendario.

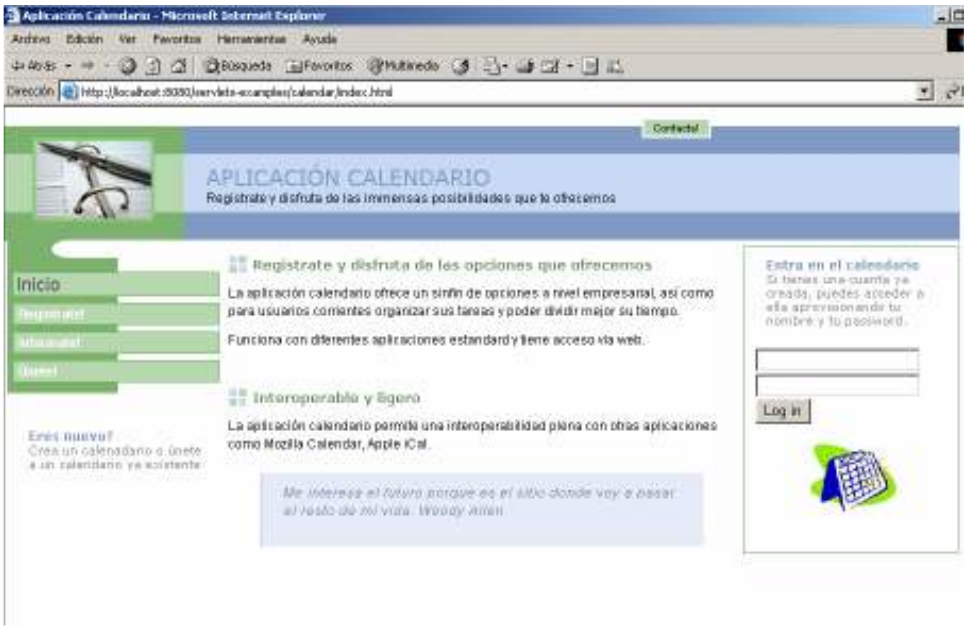

Fig. 19 Interfaz gráfica inicial del elemento gestor

Las opciones de las que se dispone en esta interfaz se basan en las opciones descritas anteriormente por un usuario no autenticado. Si el usuario aprovisiona sus datos correctamente, se autenticaría en el sistema y podría actuar sobre sus datos de usuario.

6.3.2. Interfaz gráfica de consulta de eventos en el elemento gestor

Un usuario puede consultar los datos de su calendario, seleccionándolo del conjunto de calendarios registrados mediante la interfaz Web, y podrá acceder a la información de éste. Así mismo se le mostrará un menú de opciones con nuevas funcionalidades para interactuar con el calendario escogido, así como los usuarios con los que se comparte la información.

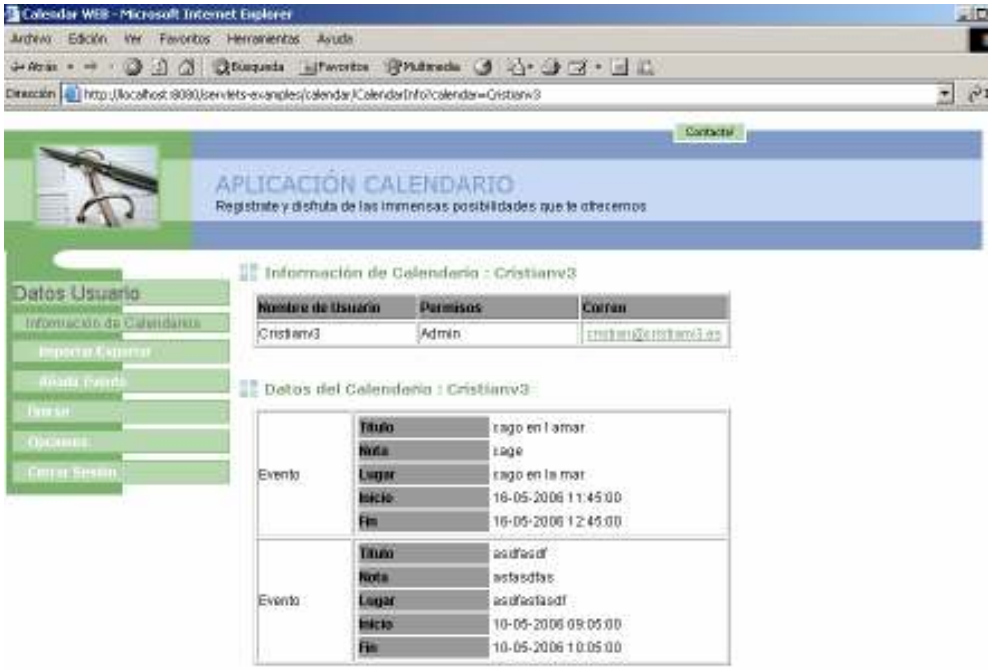

Fig. 20 Interfaz gráfica de usuario autenticado

6.3.3. Interfaz gráfica de administrador en el elemento gestor

Cuando un usuario administrador se conecte a su cuenta de calendario, recibirá las consultas realizadas por parte de otros usuarios acerca del acceso a su calendario.

Fig 21 Interfaz gráfica de administrador en el elemento gestor

Para ello se le ofrecerán las opciones de aceptar o denegar la solicitud del usuario que le demanda el acceso. En el caso de que alguno de los usuarios administradores acepten la petición, el usuario será dado de alta y se le enviará un correo a los demás usuarios administradores indicándoles el hecho.

6.3.4. Interfaz gráfica de solicitud de autenticación en el cliente

La siguiente interfaz pertenece a una aplicación cliente de calendario, donde se puede observar que al solicitar la información perteneciente a un calendario la aplicación solicita el nombre y contraseña del usuario para poderlo autenticar.

Fig. 22 Autenticación en un cliente calendario

De ser autenticado el usuario podrá trabajar con el calendario.

6.4. Capa lógica

La capa lógica se encarga de trabajar interactuando con la aplicación de gestión así como con los posibles clientes de calendario que quieran extraer información de la plataforma.

El servidor escogido, tal como se detalló anteriormente, es el servidor WebDAV empaquetado conjuntamente con el servidor Tomcat. Al trabajar conjuntamente con el servidor Tomcat nos permite desarrollar una lógica en éste en lenguaje JAVA.

El servidor recibe las solicitudes de los usuarios de Internet en cuanto al elemento gestor o bien de los usuarios de calendario mediante el cliente de calendario, las procesa y devuelve los datos.

Las páginas dinámicas recogen las solicitudes a través de formularios, realizan las acciones pertinentes y representan los resultados. El procesamiento de datos se realiza desde clases de Java.

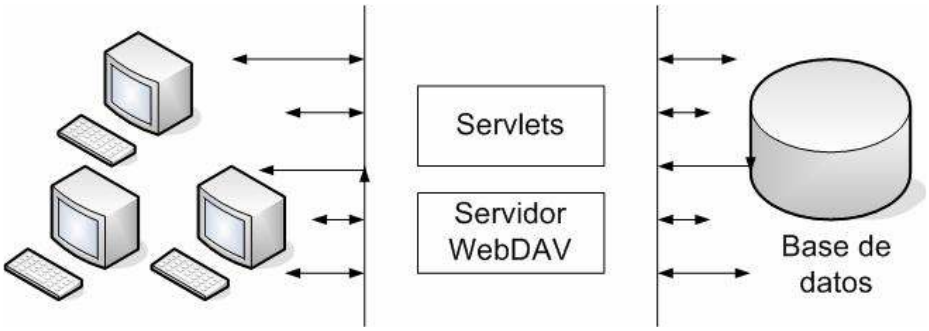

Fig. 23 Esquema de las comunicaciones del servidor Web

El servidor proporciona:

- Funcionamiento con sesiones: Cuando un usuario accede a unos datos de calendario, se le asignará una sesión. La sesión se destruye cuando el usuario cierra su navegador, cuando procede a Finalizar sesión del menú de opciones, o bien si está mucho tiempo sin utilizarlo y ésta caduca.
- Encriptación de datos: Se ha configurado el servidor Tomcat para que permita conexiones SSL. Esto quiere decir que utiliza el protocolo SSL para recibir y enviar los datos al cliente encriptados con una clave que sólo ellos conocen, y aumentando la seguridad que posee estos en los datos.
- Utilización de alarmas de correo electrónico: En una aplicación calendario resulta importante el poder informar a otros usuarios que ha habido cambios o han ocurrido hechos relevantes en un calendario. Para ello se necesita aprovisionar un servidor SMTP que realice la función de envíos de correos electrónicos. Se han definido diferentes niveles de alarmas que el usuario podrá configurar según sus preferencias.

- Característicasas WebDAV: Cuando un usuario se descarga el contenido de un calendario se procede a proteger el fichero de escritura durante un tiempo configurable y estimado en un inicio de 10 minutos. Así pues durante este tiempo nadie tendrá acceso a este recurso y si se intentara acceder se le devolvería al cliente de calendario un mensaje conforme el servidor no esta disponible. Transcurrido este tiempo el usuario perdería el control de la sesión.

6.5. Capa de gestión de datos

Para almacenar los datos introducidos en la plataforma, y dada su naturaleza, hemos decidido utilizar ficheros en lenguaje XML para conformar la capa de gestión de datos. XML, cuyas siglas representan Extensible Markup Language, o lenguaje extensible de etiquetas, es un lenguaje utilizado para estructurar información en un documento o en general en cualquier fichero que contenga texto. Ha ganado mucha popularidad en los últimos años debido a ser un estándar abierto y libre, creado por el Consorcio World Wide Web, W3C (los creadores de la WWW).

XML permite la creación de etiquetas por parte del desarrollador tal y como si fueran los campos de una base de datos. El lenguaje Java posee herramientas para una fácil lectura y escritura de datos en XML y se ha decidido desarrollar unas clases librerías que faciliten el trabajo con este tipo de ficheros. El modelo de fichero seguirá la siguiente estructura:

```
<?xml version="1.0" encoding="UTF-8"?>
<user-list>
        <user>
                <user-name> </user-name>
                <user-mail> </user-mail>
                <user-passwd> </user-passwd>
                <options>
                        <alert></alert>
                        <timezone></timezone>
                </options>
                <calendar>
                        <calendar-name> </calendar-name>
                        <role> </role>
                        <permission> </permission>
                </calendar>
        </user>
</user-list>
```

Repasando el fichero podemos constatar su modularidad, viendo como resultaría sencillo añadir nuevos usuarios dentro del bloque user. Así mismo también se puede ver como para estos usuarios se pueden añadir nuevos calendarios.

CAPÍTULO 7. CONCLUSIONES

Una vez diseñada, e implementada nuestra aplicación, en este capítulo se hace un estudio medioambiental del proyecto y se valora hasta que punto se han cumplido los objetivos planificados inicialmente. Posteriormente se comprueba que se han obtenido los resultados esperados realizando un balance de los objetivos fijados en un principio. Para acabar, se plantean posibles mejoras y ampliaciones futuras y se exponen las conclusiones personales que se han extraído después de finalizar este Proyecto de Final de Carrera.

7.1. Estudio de ambientalización

La utilización de un sistema de calendario supone grandes ventajas medioambientales. Como principal ventaja encontramos la forma de informatizar el soporte de la información de planificación, pudiéndonos ahorrar elementos como agendas, calendarios u hojas de papel para apuntar citas, consultas. De esta forma ahorramos principalmente en el gasto de papel y las consecuencias que esto implica.

Así mismo existe otra ventaja que nos ofrece, como el ahorro de rondas interminables de consultas tanto telefónicas como vía correo electrónico, y por tanto el descenso de gasto energético que esto supone. Gracias a la aplicación de calendario los usuarios se pueden planificar más rápida y eficientemente utilizando menos recursos.

7.2. Comprobación de la planificación inicial

El desarrollo del proyecto ha sido eficiente, dado que prácticamente se han cubierto todas las faenas planificadas dentro del tiempo previsto, obteniendo los resultados esperados. Todo y esto han surgido pequeñas desviaciones en la planificación inicial, pero que han sido lo suficientemente pequeñas para no afectar en el transcurso del proyecto.

7.3. Objetivos Cumplidos

El propósito principal del proyecto consistía en adentrarse en el entorno de un sistema de calendario, con el fin de ser capaces de construir una aplicación de este tipo y comprobar las dificultades que esto entraña y el funcionamiento que conlleva. Este objetivo ha sido cumplido de forma satisfactoria.

Si se analizan los objetivos previamente definidos a la realización del trabajo, se puede decir que han sido cumplidos en su totalidad. En el transcurso del proyecto hemos ido cumplimentando cada uno de los objetivos inicialmente propuestos, mostrando cual ha sido el resultado obtenido. Nos hemos introducido primero en los requisitos y necesidades del sistema, posteriormente, hemos comprobado el funcionamiento de diferentes

aplicaciones ya existentes en el mercado extrayendo os aspectos más interesantes de cada una de ellas. Por ultimo, hemos diseñado una aplicación ayudándonos de la metodología UML, la hemos implementado, la hemos probado y hemos extraído conclusiones de su implementación, para así dar por acabado el proyecto.

7.4. Mejoras y ampliaciones futuras

Una vez finalizada la aplicación, y cerciorado que están cumplimentados los requisitos básicos que nos habíamos planteado al comenzar nuestro proyecto, observamos también la existencia de posibles mejoras a realizar que habrían ayudado a perfeccionar considerablemente la aplicación final obtenida.

Una posible mejora a nuestra aplicación consistiría en la creación de una interfaz completa por parte de nuestro portal de gestión, proporcionando el conjunto de funcionalidades propias de un cliente de calendario completo con una arquitectura cliente/servidor. Este tipo de arquitecturas de sistema calendario es el que está mejor posicionado en la actualidad, y nos permitiría ampliar el abanico de interoperabilidad a los navegadores Web.

Para ello habría que ampliar las opciones que son dadas en la interfaz Web en nuestro portal, enlazándolas con las opciones ya implementadas para un cliente de calendario. Esto sería fácilmente desarrollado al haber conseguido separar las capas de presentación y de lógica.

Otra posible mejora, podría ser agregar soporte RSS a nuestra plataforma de calendario. RSS es un formato de datos XML desarrollado para adaptarse a diferentes entornos que tengan como característica una frecuente actualización, permitiendo compartir información y utilizarla desde otros sitios Web o programas. Por ejemplo RSS es ampliamente utilizado por portales de noticias, para informar de los acontecimientos a sus usuarios.

Las aplicaciones que permiten leer ficheros RSS, son conocidas como agregadores, y permiten a sus suscriptores la actualización de los contenidos así como informes de nuevos eventos acontecidos.

Mediante el soporte RSS obtendríamos un medio de intercambio y notificación de datos que es ampliamente reconocido y utilizado en la actualidad y que nos ofrecería ventajas y mayor interacción entre los usuarios. Para ello deberíamos desarrollar un nuevo módulo en nuestra plataforma gracias a librerías ya existentes y que ofrecen soporte RSS para Java como RSSLibJ.

7.5. Futuro de las aplicaciones calendario

El mundo de los calendarios interoperables representa un entorno aún en formación, donde las dificultades contraídas al crear una estandarización han provocado que varias empresas luchen por crear el estándar de facto, que se convierta en la aplicación más popular y utilizada por los usuarios.

No obstante el lanzamiento definitivo de CalDAV supondrá un punto de inflexión en el entorno. Si este estándar es ampliamente aceptado, conllevará, al fin, una estandarización que permita la interoperabilidad del diferente software de calendario en el mercado, siendo ya la elección del sistema únicamente cuestión de gustos por parte del usuario.

Hasta entonces, y en la actualidad, el mercado aún debe presentar una expansión debida a que los usuarios de Internet comenzaran a utilizar de forma más extensa este tipo de aplicaciones para planificar su tiempo. Así mismo, en este entorno domestico se utilizará de forma mayoritaria aplicaciones con arquitectura Cliente/Servidor integradas donde poder acceder a los datos mediante un simple navegador Web. Estas aplicaciones resultan sencillas y no requieren de software extra instalado en la máquina del usuario, así mismo el almacenamiento se realiza en servidores en red. De entre ellas y dado la apuesta realizada así como la integración que ofrece con productos ya conocidos destacamos Google Calendar, que se hará seguramente un hueco importante entre los internautas.

No obstante creemos que en el mundo empresarial la tendencia será diferente. Las aplicaciones reinantes resultaran aquellas con arquitectura distribuida, dado que podrán ofrecer un mayor número de opciones añadidas a cambio de la simplicidad del modelo anterior. Esta adopción, además, significará un abandono de las aplicaciones de planificación actuales, restringidas al ámbito de la propia empresa, y que si quieren continuar dominando el mercado, deberán lanzar nuevas versiones que ofrezcan la deseada interoperabilidad.

7.6. Conclusiones personales

Una vez finalizada la realización del proyecto y sopesando todos los momentos puedo valorar ésta como una experiencia satisfactoria gracias a que he podido crear por mi mismo un sistema de calendario en su totalidad, utilizando entornos nuevos y mejorando y expandiendo mis conocimientos en un lenguaje de objetos como es JAVA.

Una de las cosas más importantes a destacar han sido los conocimientos que he adquirido a lo largo de la elaboración del trabajo. La etapa de implementación me ha permitido profundizar en la problemática y los requisitos que encierran este tipo de aplicaciones. La etapa de diseño me ha ayudado a comprender como realizar los esquemas previos y descriptivos de la aplicación gracias al modulado UML.

Todo y esto, hay que destacar las dificultades encontradas en el proyecto. Ha sido complicado introducirse en protocolos y estándares recientes y no conocidos por mi parte así como la utilización de nuevas librerías para el lenguaje JAVA a fin de realizar funciones propias de éstos.

Por otra parte gracias al trabajo realizado he aprendido a llevar a cabo un proyecto en su totalidad, empezando por una planificación inicial por medio del

diagrama de Gantt, teniendo que llevar un control del tiempo y de las tareas realizadas.

Finalmente, espero que el proyecto me sirva cara a un futuro a fin de saber como planificar las tareas que me sean asignadas, a afrontar problemas nuevos o incluso a trabajar en un entorno que ya conozco como el entorno de calendario.

BIBLIOGRAFIA

[1] Lisa Dusseault, Jim Whitehead. *Open Calendar Sharing and Scheduling with CalDAV*. [En línia]. Página web, URL http://dsonline.computer.org/portal/site/dsonline/menuitem.9ed3d9924aeb0dcd82ccc6716bbe36ec/index.jsp?&pName=dso_level1&path=dsonline/0504&file=w2sta.xml

[2] D. Stenerson, F. Dawson. Internet Calendaring and Scheduling Core Object Specification (iCalendar). [En línia]. Página web, URL http://www1.ietf.org/rfc/rfc2445.txt

[3] E. James Whitehead,Jr., Meredith Wiggins. *WebDAV: IETF Standard for Collaborative Authoring on the Web.* [En línia]. Página web, URL http://ftp.ics.uci.edu/pub/ietf/webdav/intro/webdav_intro.pdf

[4] *Nathan Willis. Introduction to CalDAV* . . [En línia]. Página web, URL http://software.newsforge.com/article.pl?sid=06/02/09/2140226&tid=74

[5] http://ftp.ics.uci.edu/pub/ietf/webdav/

[6] http://www.mozilla.org/projects/calendar/

[7] http://www.calsch.org/ietf/drafts.html

ANNEXOS

TÍTULO: Serveis col·laboratius d'agenda, contactes i tasques

TITULACIÓN: Ingeniería en Telecomunicación

AUTOR: Cristian Morón Espadas

DIRECTOR: Roc Meseguer Pallarès

FECHA:

ANEXO 1. FORMATO DE DATOS ICALENDAR

8.1. Formato iCalendar

El formato iCalendar es el estándar, definido por el RFC 2445, para el intercambio de datos de tipo calendario. Dicho formato es también conocido como iCal y ha sido diseñado para describir datos basados en calendario (tal y como alarmas, eventos...).

La especificación iCalendar es resultado del trabajo del IETF Calendaring and Scheduling Working Group en su tarea de la creación de un sistema de calendario y no se encarga de describir que se debe hacer con estos datos, dado que el formato es completamente independiente del protocolo de transporte, y por tanto se requieren de otros métodos.

ICalendar está basado principalmente en su predecesor como protocolo de calendario, el denominado vCalendar, que fue especificado por el Internet Mail Consortium (IMC). Después de que iCalendar viera la luz, la misma IMC aconsejó a los desarrolladores de software aprovecharse de las ventajas de este nuevo estándar y hacer su software compatible con ambos formatos: vCalendar 1.0 e iCalendar.

Los datos iCalendar especifican la cabecera HTTP de tipo de contenido (“MIME Content-type”) con el valor: “text/calendar”. De esta forma un cliente al recibir los datos y leer la cabeza HTTP puede reconocer que los datos que encapsula el mensaje se corresponden con datos de tipo calendario. La extensión de los ficheros de calendario es “ics”, aunque existen otras extensiones reconocidas. La extensión de fichero “ifb” es utilizada para nombrar a un fichero con conteniendo de información de tiempo libre y ocupado designado con el mismo MIME Content-type.

El tipo de fichero con extensión “iCal” es utilizado por sistemas operativos Apple Macintosh para designar un fichero con información de calendario y planificación. En la misma línea, el tipo de fichero “iFBf” es utilizado por sistemas operativos Apple Macintosh para designar un fichero que almacena información de tiempo libre o ocupado con el mismo MIME Content-type.

8.2. Objetos iCalendar

8.2.1. Core object

El principal objeto en iCalendar es el conocido como Calendaring and Scheduling Core Object. Este elemento se encarga de encapsular una colección de información de datos de calendario y planificación. Típicamente, esta información consistirá en un único objeto iCalendar. No obstante, se pueden agrupar un conjunto de objetos iCalendar de forma secuencial.

Según el estándar, la primera línea del fichero debe coincidir con: "BEGIN: VCALENDAR", y la ultima con: "END: VCALENDAR". Los contenidos

encapsulados entre estas dos líneas son denominados “icalbody” o el cuerpo del fichero iCal. Podemos distinguir dos tipos de datos en el cuerpo del objeto iCalendar consistentes en las propiedades de calendario y en los componentes del calendario (pueden haber uno o más componentes en un calendario).

Las propiedades de calendario son atributos cuyo valor aplican al calendario por completo. Los componentes de calendario son colecciones de propiedades que expresan una particular semántica o una acción de calendario. Por ejemplo, el componente calendario puede especificar un evento, una acción a realizar, una reunión introducida en el calendario, información de la zona horaria, información del tiempo libre/ocupado del usuario, o una alarma. Los componentes de calendario comienzan todos por la letra “V”.

El cuerpo de calendario sin contener ninguna información se corresponde con:

```
BEGIN:VCALENDAR
VERSION
 :2.0
PRODID
 :-//Mozilla.org/NONSGML Mozilla Calendar V1.0//EN

END:VCALENDAR
```

8.2.2. Eventos (VEVENT)

Un componente VEVENT se trata de una agrupación de propiedades que describen un evento que representa un determinado tiempo planificado en el calendario. Un evento aceptado hará que el tiempo se considere como ocupado. Para evitar este hecho, un evento se puede definir como “transparente”. Un componente VEVENT por su parte, puede incluir una alarma de tipo VALARM. Los eventos tienen propiedades como DTSTART que definen su tiempo de inicio, y DTEND que define su tiempo de finalización. Si el evento es recurrente DTSTART se encargará de definir el inicio del primer evento.

Eventos repetidos como aniversarios o recordatorios diarios son representados también por el componente VEVENT. No obstante, estos eventos no incluirían la propiedad DTEND.

8.2.3. Tarea (VTODO)

El componente VTODO describe un elemento tarea introducido en el calendario. Un ejemplo de un elemento tarea lo podemos ver en el siguiente ejemplo, siendo planificada para el día 15 de Abril de 1998. En el ejemplo se puede observar como ha sido especificada una alarma recordatoria hacia la medianoche. Esta alarma se repetirá cuatro veces cada hora.

```
BEGIN:VCALENDAR
VERSION:2.0
```

```
PRODID:-//ABC Corporation//NONSGML My Product//EN
BEGIN:VTODO
DTSTAMP:19980130T134500Z
SEQUENCE:2
UID:uid4@host1.com
ORGANIZER:MAILTO:unclesam@us.gov
ATTENDEE;PARTSTAT=ACCEPTED:MAILTO:jqpublic@host.com
DUE:19980415T235959
STATUS:NEEDS-ACTION
SUMMARY:Submit Income Taxes
BEGIN:VALARM
ACTION:AUDIO
TRIGGER:19980403T120000
ATTACH;FMTTYPE=audio/basic:http://host.com/pub/audio-
 files/ssbanner.aud
REPEAT:4
DURATION:PT1H
END:VALARM
END:VTODO
END:VCALENDAR
```

8.2.4. Journal entry (VJOURNAL)

Un componente VJOURNAL describe una entrada de jornada realizada en el calendario. La acción no especifica un tiempo de inicio, ni tiene efectos en los cómputos de tiempo libre y tiempo ocupado (se trata como una entrada transparente). Aun estar especificado en el estandar, pocas implementaciones de iCalendar soportan entradas de tipo VJOURNAL.

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ABC Corporation//NONSGML My Product//EN
BEGIN:VJOURNAL
DTSTAMP:19970324T120000Z
UID:uid5@host1.com
ORGANIZER:MAILTO:jsmith@host.com
STATUS:DRAFT
CLASS:PUBLIC
CATEGORY:Project Report, XYZ, Weekly Meeting
DESCRIPTION:Project xyz Review Meeting Minutes\n
 Agenda\n1. Review of project version 1.0 requirements.\n2.
Definition
 of project processes.\n3. Review of project schedule.\n
 Participants: John Smith, Jane Doe, Jim Dandy\n-It was
  decided that the requirements need to be signed off by
  product marketing.\n-Project processes were accepted.\n
 -Project schedule needs to account for scheduled holidays
  and employee vacation time. Check with HR for specific
  dates.\n-New schedule will be distributed by Friday.\n-
```

```
 Next weeks meeting is cancelled. No meeting until 3/23.
END:VJOURNAL
END:VCALENDAR
```

8.2.5. Tiempo libre/ocupado (VFREEBUSY)

Un componente VFREEBUSY se utiliza para describir tres acciones:

- Una petición para consultar los tiempos asignados como libre/ocupado de un usuario.
- Una respuesta a la petición anterior
- La publicación de un elemento de tiempo ocupado

Cuando se utiliza para realizar peticiones la propiedad ATTENDEE especifica los usuarios de calendario cuya información va a ser solicitada. La propiedad ORGANIZER especifica el usuario que esta pidiendo la información. Las propiedades DTSTART y DTEND especifican el tiempo de ventana para el que el tiempo está siendo pedido. Las propiedades UID y DTSTAMP ayudan a indiciar un correcto secuenciado de las peticiones de la información.

Por otra parte, cuando se utiliza como respuesta la propiedad ATTENDEE especifica el usuario de calendario que debe responder a la petición, la propiedad ORGANIZER especifica el usuario de calendario que originalmente pidió la información, la propiedad FREEBUSY contiene la información solicitada y UID y DTSTAMP ayudan a indiciar un correcto secuenciado de las peticiones de la información.

Por ultimo, cuando se utiliza para publicar información de los tiempos ocupados, la propiedad ORGANIZER especifica el usuario de calendario asociado con el tiempo ocupado, el DTSTART y el DTEND especifican una ventana de tiempo con ambos incluidos. La propiedad DTSTAMP especifica la fecha/tiempo donde el objeto iCalendar fue creado.

```
BEGIN:VCALENDAR
 VERSION:2.0
 PRODID:-//RDU Software//NONSGML HandCal//EN
 BEGIN:VFREEBUSY
 ORGANIZER:MAILTO:jsmith@host.com
 DTSTART:19980313T141711Z
 DTEND:19980410T141711Z
 FREEBUSY:19980314T233000Z/19980315T003000Z
 FREEBUSY:19980316T153000Z/19980316T163000Z
 FREEBUSY:19980318T030000Z/19980318T040000Z
 URL:http://www.host.com/calendar/busytime/jsmith.ifb
 END:VFREEBUSY
 END:VCALENDAR
```

8.2.6. Otro tipo de componentes

Otros tipos de componentes definidos por el estándar incluyen VTIMEZONE para la definición de zonas horarias y VALARM para la definición de alarmas. Algunos componentes pueden incluir otros componentes por ejemplo VALARM suele ser incluida por otros componentes, y algunos componentes son a menudo definidos para soportar otros componentes definidos a posteriori (VTIMEZONE se usa de esta forma).

ANEXO 2.SOFTWARE WEBDAV

9.1. Software que soporta WebDAV

9.1.1. Clientes WebDAV

Existe diferente software que puede funcionar como cliente o bien como servidor WebDAV. En este apartado piezas de software que se pueden utilizar que soportan el protocolo WebDAV.

- Servidor HTTP Apache
- davfs2
- eZpublish
- GanttProject
- I(2) Drive WebDAV Server
- Jakarta Slide (también RFC 3253 y RFC 3744)
- Jakarta Tomcat
- Jigsaw
- KDE Desktop con Konqueror file manager
- Kiwi
- lighttpd
- Microsoft Exchange
- Microsoft IIS
- Gnome Desktop con Nautilus file manager
- OpenACS
- OSAF Chandler
- Plone
- SAP NetWeaver (Knowledge Management)
- Subversion (incluyendo versionado (checkout-merge-checkin)!)
- Virtuoso Universal Server
- WebDrive: Virtual Drive Client maps a drive to a WebDAV Server
- WebCT
- Xythos WebFile Server y WebFile Client (también RFC 3253 y RFC 3744)
- Zope
-

ANEXO 3.CÓDIGOS DE RESPUESTA HTTP

En el siguiente anexo se van a repasar los tipos de código HTTP que devuelven como respuesta los servidores Web. Conocer estos códigos nos ayudara a saber cual es la respuestas más adecuada a peticiones realizadas or el cliente.

10. Códigos de respuesta HTTP

10.1. 1xx Información

La petición ha sido recibida y se continua el proceso.

- 100: Continue
- 101: Switching Protocols

10.2. 2xx Éxitoso

La acción fue recibida correcatamente, entendida y aceptada.

- 200: OK
- 201: Created
- 202: Accepted
- 203: Non-Authoritative Information
- 204: No Content
- 205: Reset Content
- 206: Partial Content

10.3. 3xx Redirección

Ante este tipo de respuesta, el cliente debe completar información adicional para completar la petición.

- 300: Multiple Choices
- 301: Moved Permanently
- 302: Moved Temporarily (HTTP/1.0)
- 302: Found (HTTP/1.1)
- 303: See Other (HTTP/1.1)
- 304: Not Modified
- 305: Use Proxy
- 306: (no longer used, but reserved)
- 307: Temporary Redirect
-

10.4. 4xx Error de cliente

La petición posee mala sintaxis o no puede ser completada

- 400: Bad Request
- 401: Unauthorized
- 402: Payment Required
- 403: Forbidden
- 404: Not Found
- 405: Method Not Allowed
- 406: Not Acceptable
- 407: Proxy Authentication Required
- 408: Request Timeout
- 409: Conflict
- 410: Gone
- 411: Length Required
- 412: Precondition Failed
- 413: Request Entity Too Large
- 414: Request-URI Too Long
- 415: Unsupported Media Type
- 416: Requested Range Not Satisfiable
- 417: Expectation Failed

10.5. 5xx Server Error

El servidor falló al intentar completar una respuesta valida.

- 500: Internal Server Error
- 501: Not Implemented
- 502: Bad Gateway
- 503: Service Unavailable
- 504: Gateway Timeout
- 505: HTTP Version Not Supported
- 509: Bandwidth Limit Exceeded (No oficial)