

Universidad Autónoma Metropolitana Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Proyecto Terminal:
CALENDARIOS COMPARTIDOS Y EVENTOS GRUPALES
(CCEG): APLICACIÓN DE ESCRITORIO PARA LA GESTIÓN DE EVENTOS
INDIVIDUALES Y GRUPALES EN CALENDARIOS DE GOOGLE

Alumna:
Lucía Echenique Álvarez
Matricula: 207305364

Asesoras:
Dra. Beatriz Adriana González Beltrán
Departamento de Sistemas

Dra. María Lizbeth Gallardo López
Departamento de Sistemas

Resumen

Hoy en día las personas tienen muchas actividades que realizar, entre los que se encuentran: eventos culturales, deportivos o académicos, reuniones de trabajo o entre amigos, citas médicas; por eso es necesario tener todos estos eventos registrados para no olvidar llevarlos a cabo. Las aplicaciones de calendario permiten mantener eventos almacenados de manera organizada.

El objetivo de esta aplicación es diseñar e implementar una aplicación de escritorio que permita gestionar eventos individuales y grupales en calendarios de Google. Para cumplir este objetivo fue necesario diseñar los módulos de usuario, grupos, eventos y calendarios, y la implementación de los mismos.

La aplicación de escritorio “CALENDARIOS COMPARTIDOS Y EVENTOS GRUPALES (CCEG)” ha sido programada utilizando el lenguaje Java, utiliza como base el API de Google Calendar y tiene un nuevo módulo para la gestión de grupos y grupos de grupos que le dan al usuario una forma sencilla de agregar invitados a un evento.

En esta aplicación de escritorio el usuario puede gestionar calendarios, eventos y grupos desde el escritorio de su PC e incluso puede hacer uso de algunos de los módulos mencionados anteriormente sin necesidad de una conexión a internet.

La interfaz gráfica está diseñada de tal forma que los usuarios de Google Calendar se sientan familiarizados y puedan hacer un buen uso de la aplicación.

Tabla de contenidos

Resumen.....	2
Tabla de contenidos	3
Introducción	4
Objetivos	4
Objetivo general.....	4
Objetivos particulares	4
Antecedentes	5
Referencias externas.....	5
Referencias internas.....	5
Justificación	6
Metodología empleada en el proyecto.....	6
Desarrollo del proyecto.....	7
Diseño del sistema	7
Diagrama de casos de uso.....	7
Diagrama de clases.....	9
Diagrama de dominio.....	11
Estructura de la base de datos	12
Uso del sistema	13
Crear un evento.....	14
Crear un calendario.....	15
Crear un grupo simple.....	16
Crear grupo de grupos	17
Hardware.....	18
Software	18
Conclusiones y perspectivas del proyecto	19
Bibliografía	20
Apéndice A. Listado del código fuente desarrollado	21

Introducción

El presente documento presenta la descripción de la aplicación de escritorio “CALENDARIOS COMPARTIDOS Y EVENTOS GRUPALES (CEEG)” que se realizó como proyecto terminal de la Licenciatura en Ingeniería en Computación.

La aplicación CCEG busca gestionar eventos individuales y grupales en calendarios de Google, de tal forma que los usuarios de Gmail puedan administrar sus eventos desde su computadora y trabajar en modo desconectado.

Objetivos

Objetivo general

Diseñar e implementar una aplicación de escritorio que permita gestionar eventos individuales y grupales en calendarios de Google.

Objetivos particulares

- Diseñar el módulo de usuarios.
- Diseñar el módulo de eventos del usuario.
- Diseñar el módulo de grupos del usuario.
- Diseñar el módulo de calendarios del usuario.
- Implementar el módulo de usuarios.
- Implementar el módulo de los eventos del usuario.
- Implementar el módulo de grupos del usuario.
- Implementar el módulo de calendarios del usuario.
- Integrar los módulos en la aplicación calendario.
- Hacer pruebas a la aplicación para verificar su funcionalidad.

Antecedentes

Podemos encontrar varias aplicaciones de calendario que permiten al usuario crear, eliminar, modificar y consultar eventos, tales como: Google Calendar, Lightning, iCalendar y Evolution.

Referencias externas

Google calendar: Es un calendario en línea (aplicación Web) en donde se pueden añadir eventos e invitaciones, compartir calendarios y hacer búsquedas de eventos. Para usar esta aplicación es necesario tener una cuenta en *gmail* [1].

Lightning: Es un complemento de Mozilla Thunderbird que permite al usuario ver y crear eventos, al igual que suscribirse a calendarios de red [2].

iCalendar: Permite crear calendarios, compartir calendarios por Internet y recibir notificaciones de eventos próximos vía correo electrónico o SMS. Esta aplicación está desarrollada para la plataforma Mac [3].

Evolution: Es un gestor de calendario de Linux que permite crear eventos, mandar notificaciones e incluso puede importar contenidos de otras aplicaciones como Google Calendar [4].

Referencias internas

El proyecto terminal titulado “*Aplicación web de administración de horarios para estudiantes*” permite a un estudiante administrar sus horarios de cursos y de actividades extraescolares dentro de un calendario. De manera particular, esta aplicación permite a un usuario definir diferentes períodos o ciclos escolares en los que se encuentra inscrito, las materias que corresponden a cada uno de los períodos y las actividades asignadas a cada materia [5]. Aunque este trabajo gestiona un calendario, todos los eventos son individuales y no existen eventos grupales, como se tiene planeado dentro de esta propuesta de proyecto terminal.

Justificación

Hoy en día las personas tienen muchas actividades que realizar, entre las que se encuentran: eventos culturales, deportivos o académicos, reuniones de trabajo o entre amigos, citas médicas; por eso es necesario tener todos estos eventos registrados para no olvidar llevarlos a cabo. Las aplicaciones de calendario permiten mantener eventos almacenados de manera organizada.

Si bien existen diferentes aplicaciones de calendario (ver sección Antecedentes) que permiten gestionar eventos, compartir calendarios, recibir notificaciones de eventos próximos y proporcionar diferentes vistas del calendario, no existen aplicaciones que incluyan el concepto de grupos al momento de manejar los eventos a compartir. Es por ello que el objetivo de esta propuesta de proyecto terminal es diseñar e implementar una aplicación de escritorio que gestione eventos individuales y grupales en calendarios de Google. Esta aplicación permitirá al usuario gestionar los calendarios desde su computadora, permitiéndole trabajar aún en momentos que no haya conexión.

Metodología empleada en el proyecto

La metodología empleada en la elaboración del proyecto fue el proceso unificado.

Los aspectos distintivos del Proceso Unificado están capturados en tres conceptos clave: dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Esto es lo que hace único al Proceso Unificado.

El Proceso Unificado usa el Lenguaje de Modelado Unificado (UML) en la preparación de todos los diagramas del sistema.

Desarrollo del proyecto

Diseño del sistema

Diagrama de casos de uso

La aplicación toma como base el API de Google Calendar. La funcionalidad de la aplicación se puede ver en la figura siguiente (ver Figura 1).



Figura 1. Diagrama de casos de uso

Registro del usuario

Este módulo permitirá que el usuario se registre para poder establecer el contacto con el servidor Google Calendar. La información que deberá ingresar el usuario es: nombre, contraseña y correo electrónico. Esta información deberá coincidir con la de Google Calendar.

Gestión de calendarios

El módulo de gestión de calendarios está constituido por los casos de uso siguientes:

- **Alta de un calendario:** el usuario podrá crear un nuevo calendario e ingresará la información siguiente: nombre del calendario, descripción, ubicación, zona horaria del calendario. Además, el usuario elegirá si quiere que el calendario sea público o si quiere compartirlo con determinadas personas, para la segunda opción será necesario introducir el correo electrónico de la persona y especificar sus permisos.
- **Eliminación de un calendario:** el usuario podrá borrar el calendario.

- **Modificación de un calendario:** el usuario podrá cambiar la información del calendario, como las personas con quien quiere compartirlo, la descripción, etc.
- **Consulta de un calendario:** el usuario podrá visualizar los eventos que se encuentren en el calendario.

Gestión de eventos

El módulo de gestión de eventos está constituido por los casos de uso siguientes:

- **Alta de un evento:** el usuario podrá crear un evento e ingresará al sistema los datos siguiente: nombre del evento, fecha de inicio, fecha de término, hora de inicio, hora término, lugar, calendario, descripción. Además, el usuario podrá definir si el evento es grupal y, en su caso, el sistema solicitará el ingreso de los contactos o de los grupos que desea invitar.
- **Modificación de un evento:** el usuario podrá cambiar la información de algún evento.
- **Eliminación de un evento:** el usuario podrá borrar un evento.
- **Consulta de un evento:** el usuario podrá visualizar la información de un evento.
- **Notificación de eventos:** el usuario podrá recibir un correo electrónico informándole de un evento próximo, también se tendrá la opción de ventanas emergentes para el mismo fin.

Gestor de grupos

El módulo de gestión de grupos está constituido por los casos de uso siguientes:

- **Crear grupo:** el usuario podrá crear un grupo a partir de una lista de contactos, e incluso, podrá crear un grupo a partir de los grupos ya definidos y de la lista de contactos.
- **Eliminar grupo:** el usuario podrá borrar un grupo de contactos.
- **Modificar grupo:** el usuario podrá cambiar información de sus grupos.
- **Consultar grupo:** el usuario podrá visualizar los integrantes de un grupo.

Diagrama de clases

En el siguiente diagrama se visualizan las clases que contiene la aplicación (Figura 2).

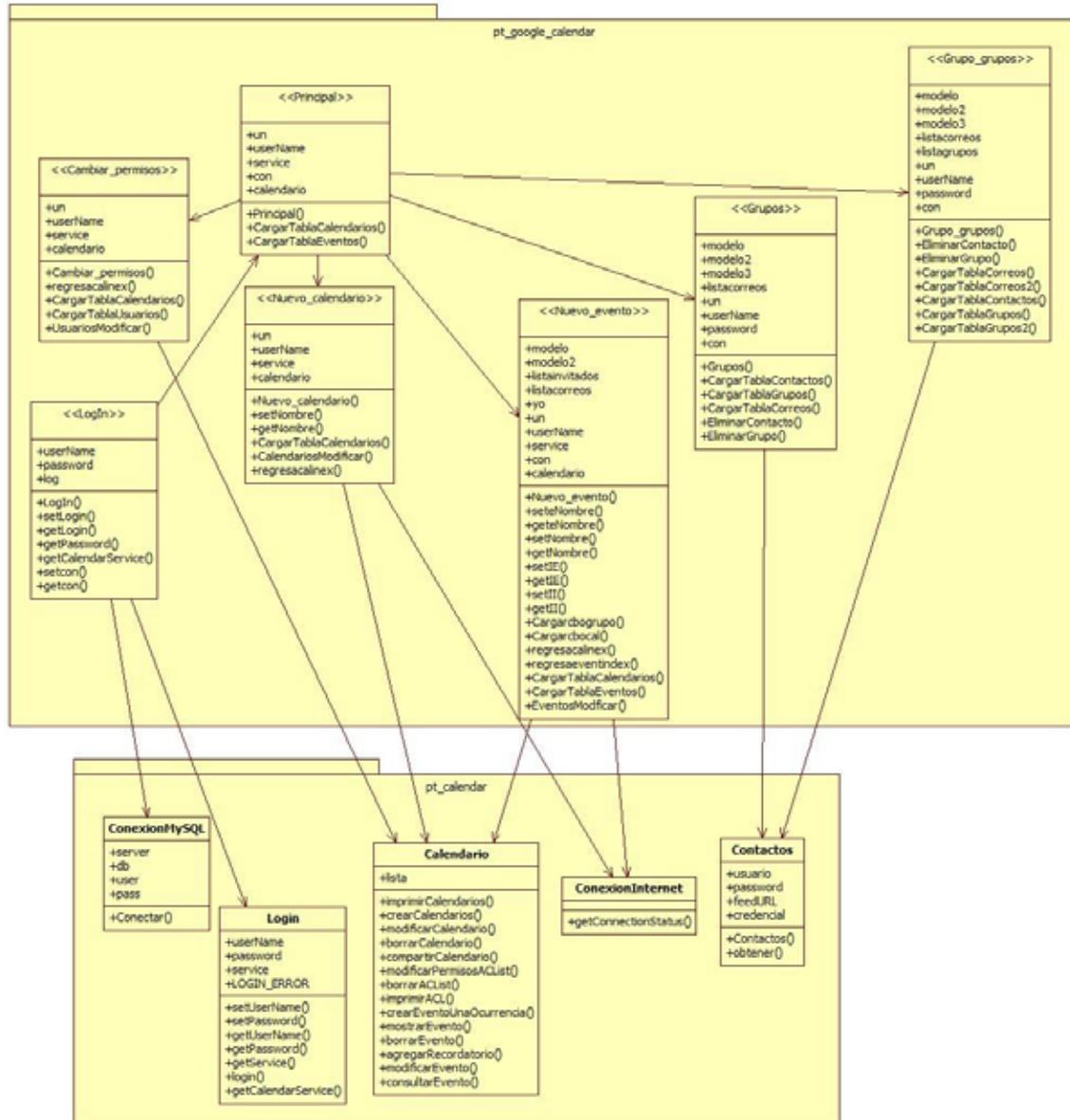


Figura 2. Diagrama de clases.

Dentro del proyecto hay dos paquetes que separan las clases de la interfaz de las clases que contienen la parte funcional de la aplicación, su contenido se describe a continuación:

1. pt_calendar: En este paquete se encuentran las clases de java que contienen las acciones que se llevan a lo largo de la aplicación.
 - a) Calendario.java: esta clase contiene los métodos de la API de Google Calendar.
 - b) ConexionInternet.java: esta clase nos dice si tenemos conexión a Internet.
 - c) ConexionMYSQL.java: esta clase hace la conexión con la base de datos.
 - d) Contactos.java: esta clase regresa los contactos de Gmail.
 - e) Login.java: esta clase hace la conexión con Google.

2. pt_google_calendar: En este paquete se encuentran los jFrames que construyen la interfaz de la aplicación.
 - a) Cambiar_permisos.java: esta clase se encarga de la gestión de los usuarios secundarios de los calendarios.
 - b) Grupo_grupos.java: esta clase se encarga de la gestión de grupos de grupos.
 - c) Grupos.java: esta clase se encarga de la gestión de grupos simples.
 - d) LogIn.java: esta clase es la que verifica la información del usuario y permite el ingreso a la aplicación.
 - e) Nuevo_calendario.java: esta clase se encarga de la gestión de calendarios.
 - f) Nuevo_evento.java: esta clase se encarga de la gestión de eventos.
 - g) Principal.java: esta es la ventana principal.
 - h) Menu_grupos.java: esta es la ventana del menú de tipo de eventos.

Diagrama de dominio

El diagrama siguiente especifica las funcionalidades de cada módulo de la aplicación (ver Figura 3).

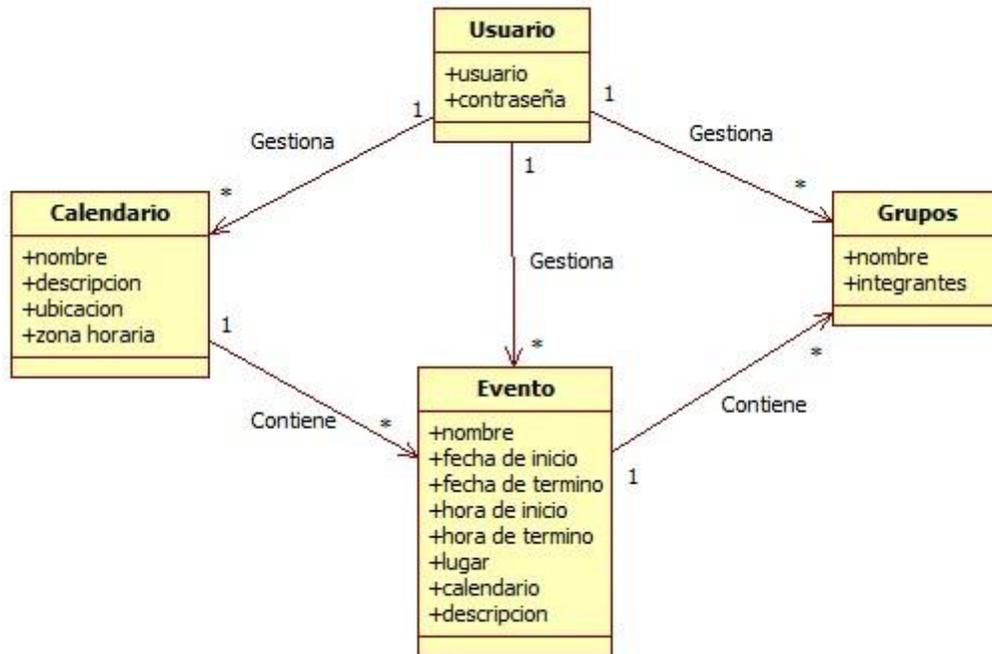


Figura 3. Diagrama de dominio

Estructura de la base de datos

El diagrama siguiente visualiza el diseño de la base de datos de la aplicación (ver Figura 4).

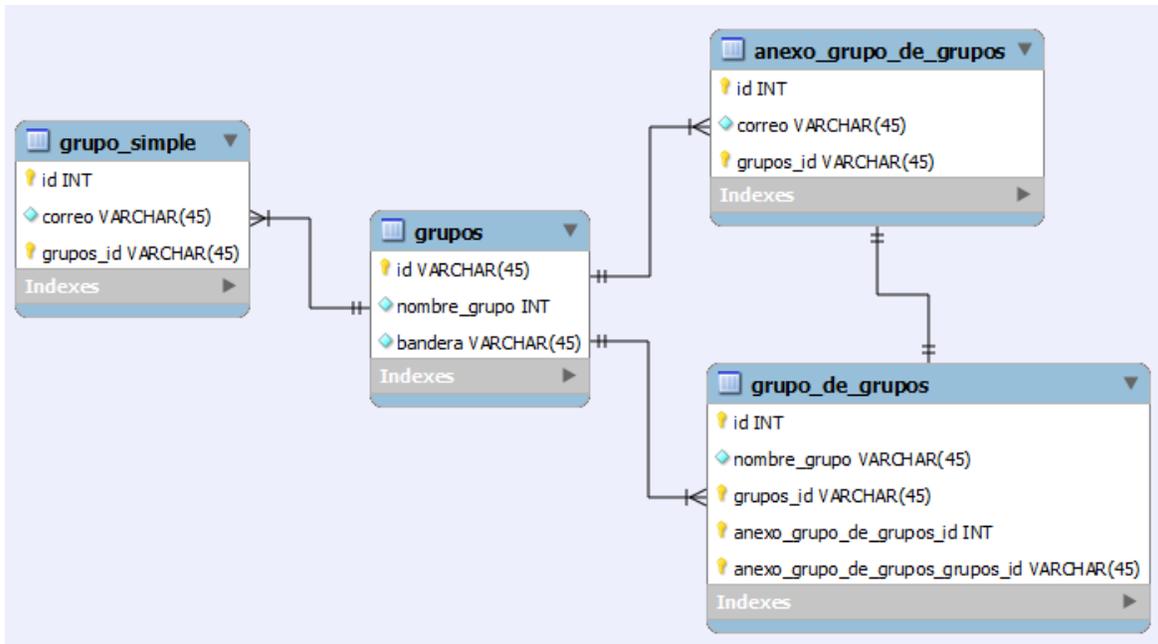


Figura 4. Estructura de la base de datos

Tabla grupos

Esta tabla contiene los nombres de los grupos simples, de los grupos de grupos y de las tablas anexas, sus atributos son los siguientes:

- Nombre_grupo: nombre del grupo (grupo simple, grupo de grupos, anexo de un grupo).
- Bandera: identificador del tipo de grupo (0-grupo simple, 1-grupo de grupos, 2-grupo anexo)

Tabla grupo simple

Contiene el identificador y los integrantes de un grupo simple, sus atributos son los siguientes:

- Id: contador
- Correo: correo electrónico de los integrantes.

Tabla grupo de grupos

Contiene los nombres de los grupos simples que forman al grupo de grupos, sus atributos son los siguientes:

- Id: contador
- Correo: correo electrónico de los integrantes.

Tabla anexo_grupo_de_grupos

Contiene los contactos de los grupos de grupos que no pertenecen a ningún grupo simple, sus atributos son los siguientes.

- Correo: correo electrónico de los integrantes.

Uso del sistema

Los casos de uso son los siguientes:

- Crear calendario
- Borrar calendario
- Modificar calendario
- Consultar calendario
- Crear evento
- Borrar evento
- Modificar evento
- Consultar evento
- Crear grupo
- Borrar grupo
- Modificar grupo
- Consultar grupo

Los casos más importantes se describen a continuación:

Crear un evento

Para crear un evento, ingrese el título, fecha de inicio y de término, y ubicación. De manera opcional, ingrese la descripción y los invitados al evento. Una vez ingresados todos los datos, oprima el botón “Crear evento”. Si se desea agregar un recordatorio, después de crear el evento ingrese un número de minutos del recordatorio y oprima el botón “Aceptar”.

The screenshot shows a software window titled "Gestión de eventos". The main area is divided into several sections:

- Datos del evento:** Contains fields for "Titulo:", "Tipos de eventos:" (a dropdown menu), "Lugar:", "Calendario:" (a dropdown menu showing "Calendario Lucia"), and "Descripción:" (a large text area).
- Fecha:** A date field set to "2013-04-24".
- Selección de calendario:** A table with the following content:

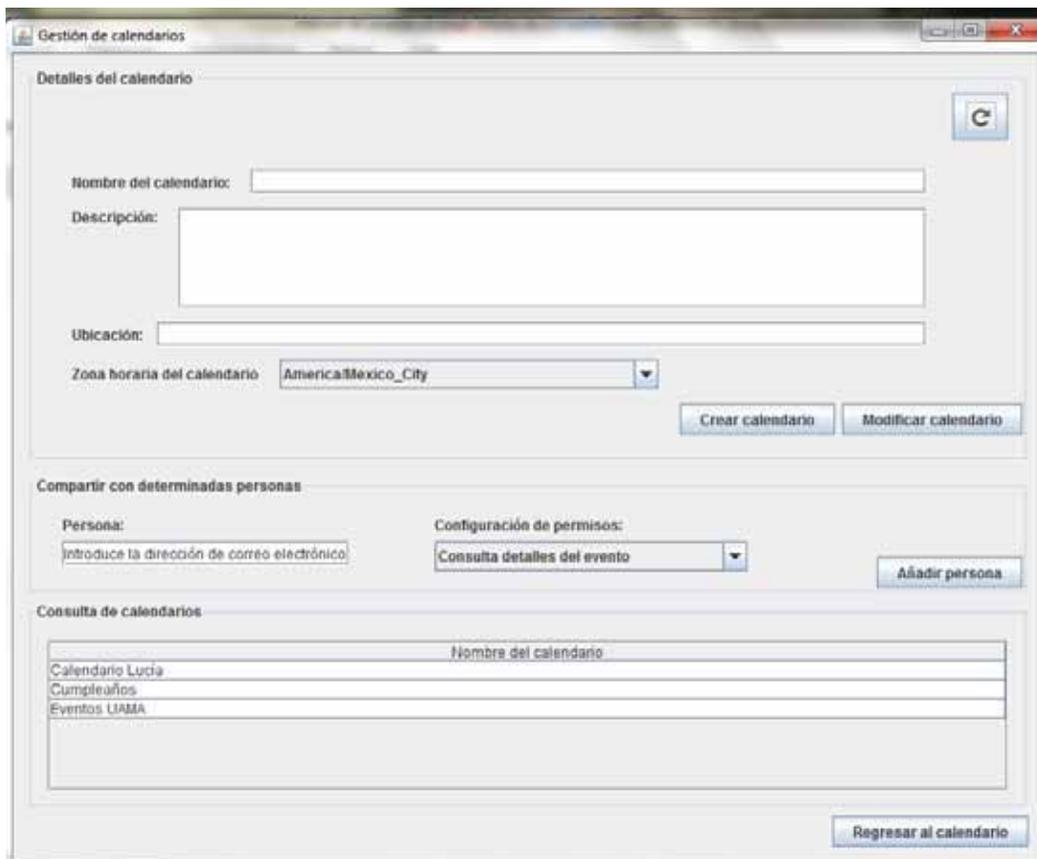
Nombre del calendario
Calendario Lucia
Cumpiaños
Eventos UAMA
- Eventos del calendario:** A large empty rectangular area.
- Buttons:** "Crear evento" and "Modificar evento" are located below the main form fields.
- Añadir recordatorio:** A section with a "Recordatorios:" field, the unit "minutos", and an "Aceptar" button.
- Añadir tipo de evento:** A section with a "Nuevo tipo de evento:" field and an "Agregar" button.
- Other:** An "Añadir invitados" field with an "Añadir" button, a "Seleccionar grupo:" dropdown menu (set to "(ninguno)"), and a "Regresar al calendario" button at the bottom right.

Figura 5. Ventana Gestión de eventos

Crear un calendario

Para crear un calendario ingrese el nombre, ubicación, descripción y zona horaria del calendario y oprima el botón “Crear calendario”. Los calendarios creados se encuentran en la tabla que está en la parte inferior de la ventana (ver Figura 6).

Si desea compartir el calendario, ingrese el correo electrónico en donde se encuentra la etiqueta “Persona:” y en la etiqueta “Configuración de permisos:” seleccione el permiso que quiere darle y por último oprima el botón “Añadir persona”.



The screenshot shows a web application window titled "Gestión de calendarios". It is divided into three main sections:

- Detalles del calendario:** Contains input fields for "Nombre del calendario:", "Descripción:", and "Ubicación:". There is a dropdown menu for "Zona horaria del calendario" currently set to "America/Mexico_City". Buttons for "Crear calendario" and "Modificar calendario" are located at the bottom right of this section.
- Compartir con determinadas personas:** Includes a "Persona:" field with the placeholder "Introduce la dirección de correo electrónico:", a "Configuración de permisos:" dropdown menu set to "Consulta detalles del evento", and an "Añadir persona" button.
- Consulta de calendarios:** A table listing existing calendars. The table has a header "Nombre del calendario" and the following entries:

Nombre del calendario
Calendario Lucía
Cumpleaños
Eventos LIAMA

A "Regresar al calendario" button is located at the bottom right of the window.

Figura 6. Ventana Gestión de calendarios.

Crear un grupo simple

Para crear un grupo ingrese el nombre y oprima el botón derecho del ratón sobre los contactos que se quieran agregar al grupo, agregue cuantos contactos quiera y cuando termine oprima el botón "Crear" (ver Figura 7).

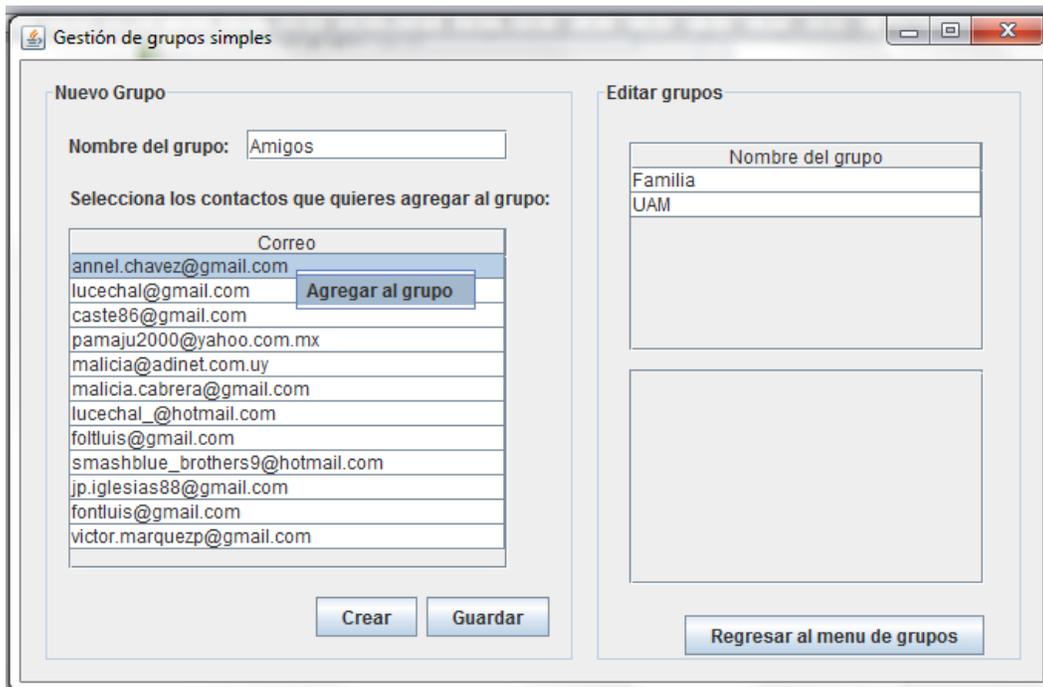


Figura 7. Crear un grupo

Crear grupo de grupos

Para crear un grupo de grupos, ingrese el nombre del grupo, de la tabla que tiene como etiqueta “Grupos simples” seleccione un grupo que quiera agregar y oprima el botón derecho del ratón sobre el nombre del grupo, y seleccione la opción “Agregar grupo al grupo”, este paso puede repetirlo si quiere agregar más grupos de la tabla.

Además de agregar grupos puede agregar contactos de la tabla que se encuentra en la parte izquierda de la ventana, para poder hacerlo, seleccione el contacto que quiera agregar y oprima el botón derecho del ratón sobre el contacto seleccionado, y seleccione la opción “Agregar”, este paso puede repetirlo si quiere agregar más contactos al grupo. Como último paso, oprima el botón “Crear” (ver Figura 8).

Nota: Un contacto de la lista de contactos puede ya estar dentro de un grupo simple. Es responsabilidad del usuario si éste añade un grupo y un contacto de la lista de contactos que ya se encuentra en este grupo. En este caso, el contacto recibirá dos notificaciones para el mismo evento.

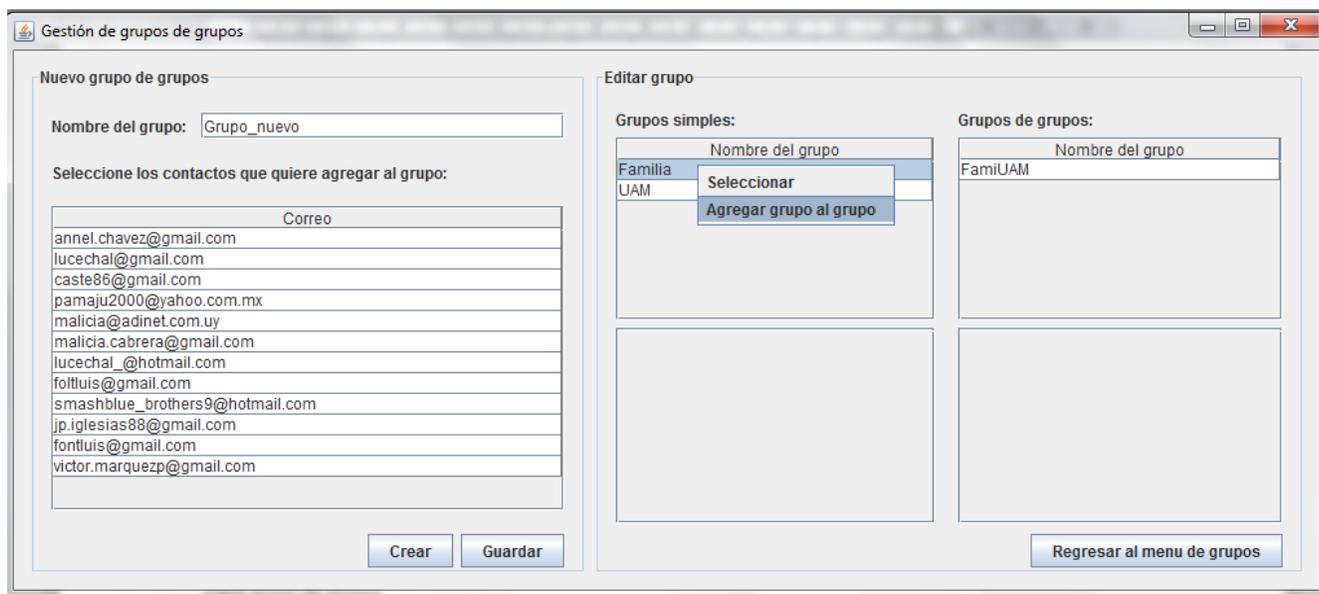


Figura 8. Crear un grupo de grupos

Hardware

La aplicación se desarrolló en una computadora con sistema operativo Windows 7 con las siguientes características:

1. Procesador Intel U4100 a 1.30GHz
2. Sistema operativo Windows 7 de 64 bits.
3. Disco duro de 288 GB.
4. Memoria RAM de 3GB.

Software

La aplicación se desarrolló con el siguiente software:

1. El manejador de base de datos utilizado fue MySQL, versión 5.2. Se puede descargar un instalador del siguiente URL: <http://dev.mysql.com/downloads/installer/>
2. La aplicación está desarrollada en el lenguaje de programación Java. La plataforma Java utilizada fue Java Enterprise Edition.
3. El ambiente de desarrollo utilizado fue NetBeans, versión 7.2.1.
4. La interfaz gráfica utiliza Java Swing.
5. La aplicación necesita el API de Google calendar. La guía de desarrollo "Google Calendar API v2 Developer's Guide: Java" se encuentra en el siguiente URL: https://developers.google.com/google-apps/calendar/v2/developers_guide_java

A continuación se listan los URL desde donde se pueden descargar las bibliotecas necesarias.

- a) Biblioteca Google Calendar. Las bibliotecas cliente de las APIs Google Data están disponibles para ayudarle a escribir aplicaciones cliente que acceden a la su API favorito.

<https://code.google.com/p/gdata-java-client/downloads/list>

- b) Bibliotecas guava. Contiene varias de las bibliotecas centrales de Google que nos sirven en nuestros proyectos basados en Java: colecciones, almacenamiento en caché, primitivas de apoyo, bibliotecas de concurrencia, las anotaciones comunes, procesamiento de cadenas, I / O, y así sucesivamente.

<https://code.google.com/p/guava-libraries/>

- c) Biblioteca jcalendar. Biblioteca que sirve para la visualización de calendarios en la interfaz gráfica.

<http://www.toedter.com/en/jcalendar/>

Conclusiones y perspectivas del proyecto

La aplicación “CALENDARIOS COMPARTIDOS Y EVENTOS GRUPALES (CEEG)”, que se desarrolló como proyecto terminal, está habilitada para poder gestionar calendarios y eventos en Google Calendar. Como se especifica en la propuesta, se creó el módulo de gestión de grupos para facilitar la creación de eventos grupales.

La siguiente tabla muestra los objetivos específicos, así como el alcance que se obtuvo en el desarrollo del este proyecto terminal.

	Objetivos	Descripción
✓	Diseñar el módulo de usuarios.	Se desarrollaron los diagramas correspondientes tomando en cuenta la estructura de Google Calendar.
✓	Diseñar el módulo de eventos del usuario.	Se desarrollaron los diagramas correspondientes tomando en cuenta la estructura de Google Calendar.
✓	Diseñar el módulo de grupos del usuario.	Se hicieron los diagramas y escenarios correspondientes para este módulo.
✓	Diseñar el módulo de calendarios del usuario.	Se desarrollaron los diagramas correspondientes tomando en cuenta la estructura de Google Calendar.
✓	Implementar el módulo de usuarios.	Se implementó correctamente mediante el uso de java Swing.
✓	Implementar el módulo de los eventos del usuario.	Se implementó correctamente mediante el uso de java Swing.
✓	Implementar el módulo de grupos del usuario.	El modulo se implementó con éxito mediante el uso de java Swing.
✓	Implementar el módulo de calendarios del usuario.	El modulo se implementó con éxito mediante el uso de java Swing.
✓	Integrar los módulos en la aplicación calendario.	Los módulos se integraron con éxito mediante el uso de java Swing.
✓	Hacer pruebas a la aplicación para verificar su funcionalidad.	Se realizaron pruebas sobre las operaciones de inserción, consulta, modificación y borrado, correspondientes a cada uno de los módulos.

La realización de este proyecto me permitió reafirmar mis conocimientos del lenguaje Java, y mis conocimientos sobre la implementación basado en jframes de java Swing; además del manejo de base de datos en MySQL y JDBC.

Considero que el avance sobre la aplicación fue sustancial; sin embargo aún hay funcionalidades que se podrían desarrollar como el calendario en la página principal, de tal forma que el usuario pueda visualizar sus eventos como en la página de Google Calendar. Cabe mencionar que no se pudo desarrollar lo anterior mencionado porque los recursos que se encontraron estaban destinados a proyectos web.

Bibliografía

[1] Google Calendar. Fecha de consulta: 01 de junio de 2010. Disponible en:
<https://support.google.com/calendar/?hl=es>

[2] Lightning. Fecha de consulta: 22 de junio de 2013. Disponible en:
<http://www.mozillamessaging.com/es-ES/thunderbird>

[3]iCalendar. Fecha de consulta: 23 de junio de 2013. Disponible en:
<http://www.mackenty.org/images/uploads/how%20to%20use%20iCal.pdf>

[4] Evolution. Fecha de consulta: 23 de junio de 2013. Disponible en:
<http://projects.gnome.org/evolution/doc/evolution24>.

[5] A. M. Velázquez-Canales, "Aplicación web de administración de horarios para estudiantes", proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana, Azcapotzalco, México, 2008.

Apéndice A. Listado del API del código fuente desarrollado

Class Calendario

```
java.lang.Object  
└─pt_calendar.Calendario
```

```
public class Calendario  
extends java.lang.Object
```

Esta clase contiene los métodos de la API de Google Calendar

Constructor Summary

[Calendario](#)()

Method Summary

void	agregarRecordatorio (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL, int index, int eventIndex, int numMinutes) Método para crear recordatorio de los eventos
void	borrarACLlist (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL, java.lang.String userEmail, int index) Borramos a un usuario de la ACL del calendario
void	borrarCalendario (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedUrl, int index) Método que borra un calendario de acuerdo a un índice
void	borrarEvento (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL,

	<pre>int index, int eventoIndex)</pre> <p>Método que borra los eventos de un calendario</p>
void	<pre>compartirCalendario (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL, java.lang.String userEmail, com.google.gdata.data.acl.AclRole role, int index)</pre> <p>Compartimos el calendario con un usuario</p>
void	<pre>consultarEvento (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL, int index, int eventIndex)</pre> <p>Método que consulta los datos de un evento</p>
void	<pre>crearCalendario (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedUrl, java.lang.String[] dato)</pre> <p>Método que crea un calendario</p>
java.util.ArrayList<java.lang.String>	<pre>imprimirACL (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL, int index)</pre> <p>Imprime el access control list de un calendario (ACL)</p>
java.lang.String[]	<pre>imprimirCalendarios (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedUrl)</pre> <p>Método que imprime los títulos de cada calendario y los guarda en un arreglo</p>
void	<pre>modificarCalendario (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL, int index, java.lang.String[] dato)</pre> <p>Método que modifica un calendario existente</p>
void	<pre>modificarEvento (com.google.gdata.client.calendar.CalendarService service, java.net.URL feedURL,</pre>

	<pre>int index, int eventIndex, java.lang.String titulo, java.lang.String contenido, java.lang.String fechaInicio, java.lang.String fechaFin, java.lang.String lugar)</pre>
void	<pre>modificarPermisosACLlist (com.google.gdata.client.c alendar.CalendarService service, java.net.URL feedURL, java.lang.String userEmail, com.google.gdata.data.acl.AclRole newRole, int index) Modificar los permisos de un usuario</pre>
java.lang.String[]	<pre>mostrarEvento (com.google.gdata.client.calendar.Ca alendarService service, java.net.URL feedURL, int index) Método que muestra los eventos de un calendario</pre>
void	<pre>singleOcurrenceEvent (com.google.gdata.client.cale ndar.CalendarService service, java.net.URL feedURL, int index, java.lang.String titulo, java.lang.String contenido, java.lang.String fechaInicio, java.lang.String fechaFin, java.lang.String lugar, java.util.ArrayList participante) Método de servicio para crear eventos de una sola ocurrencia</pre>

Constructor Detail

Calendario

```
public Calendario()
```

Method Detail

imprimirCalendarios

```
public java.lang.String[]
imprimirCalendarios (com.google.gdata.client.calendar.CalendarService service,
                    java.net.URL feedUrl)
                    throws java.io.IOException,
                    com.google.gdata.util.ServiceException
```

Método que imprime los títulos de cada calendario y los guarda en un arreglo

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

crearCalendario

```
public void
crearCalendario (com.google.gdata.client.calendar.CalendarService service,
                java.net.URL feedUrl,
                java.lang.String[] dato)
                throws java.io.IOException,
                com.google.gdata.util.ServiceException
```

Método que crea un calendario

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

modificarCalendario

```
public void
modificarCalendario(com.google.gdata.client.calendar.CalendarService service,
                    java.net.URL feedURL,
                    int index,
                    java.lang.String[] dato)
throws java.io.IOException,
        com.google.gdata.util.ServiceException
```

Método que modifica un calendario existente

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

borrarCalendario

```
public void
borrarCalendario(com.google.gdata.client.calendar.CalendarService service,
                    java.net.URL feedUrl,
                    int index)
throws java.io.IOException,
        com.google.gdata.util.ServiceException
```

Método que borra un calendario de acuerdo a un índice

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

compartirCalendario

```
public void
compartirCalendario(com.google.gdata.client.calendar.CalendarService service,
                    java.net.URL feedURL,
                    java.lang.String userEmail,
                    com.google.gdata.data.acl.AclRole role,
                    int index)
    throws java.io.IOException,
           com.google.gdata.util.ServiceException
```

Compartimos el calendario con un usuario

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

modificarPermisosACLList

```
public void
modificarPermisosACLList(com.google.gdata.client.calendar.CalendarService
service,
                    java.net.URL feedURL,
                    java.lang.String userEmail,
                    com.google.gdata.data.acl.AclRole newRole,
                    int index)
    throws java.io.IOException,
           com.google.gdata.util.ServiceException
```

Modificar los permisos de un usuario

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

borrarACLlist

```
public void
borrarACLlist(com.google.gdata.client.calendar.CalendarService service,
                java.net.URL feedURL,
                java.lang.String userEmail,
                int index)
                throws java.io.IOException,
                com.google.gdata.util.ServiceException
```

Borramos a un usuario de la ACL del calendario

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

imprimirACL

```
public java.util.ArrayList<java.lang.String>
imprimirACL(com.google.gdata.client.calendar.CalendarService service,
             java.net.URL feedURL,
             int index)
             throws
             java.io.IOException,
             com.google.gdata.util.ServiceException
```

Imprime el access control list de un calendario (ACL)

Throws:

```
java.io.IOException
com.google.gdata.util.ServiceException
```

singleOcurrencyEvent

```
public void
singleOcurrencyEvent (com.google.gdata.client.calendar.CalendarService ser
vice,
                        java.net.URL feedURL,
                        int index,
                        java.lang.String titulo,
                        java.lang.String contenido,
                        java.lang.String fechaInicio,
                        java.lang.String fechaFin,
                        java.lang.String lugar,
                        java.util.ArrayList participante)
throws com.google.gdata.util.ServiceException,
      java.io.IOException
```

Método de servicio para crear eventos de una sola ocurrencia

Throws:

com.google.gdata.util.ServiceException

java.io.IOException

mostrarEvento

```
public java.lang.String[]
mostrarEvento (com.google.gdata.client.calendar.CalendarService service,
                java.net.URL feedURL,
                int index)
                throws
com.google.gdata.util.ServiceException,
                java.io.IOException
```

Método que muestra los eventos de un calendario

Throws:

com.google.gdata.util.ServiceException

java.io.IOException

borrarEvento

```
public void
borrarEvento(com.google.gdata.client.calendar.CalendarService service,
              java.net.URL feedURL,
              int index,
              int eventoIndex)
              throws com.google.gdata.util.ServiceException,
              java.io.IOException
```

Método que borra los eventos de un calendario

Throws:

com.google.gdata.util.ServiceException

java.io.IOException

agregarRecordatorio

```
public void
agregarRecordatorio(com.google.gdata.client.calendar.CalendarService serv
ice,
                    java.net.URL feedURL,
                    int index,
                    int eventIndex,
                    int numMinutes)
                    throws com.google.gdata.util.ServiceException,
                    java.io.IOException
```

Método para crear recordatorio de los eventos

Throws:

com.google.gdata.util.ServiceException

java.io.IOException

modificarEvento

```
public void
modificarEvento(com.google.gdata.client.calendar.CalendarService service,
                 java.net.URL feedURL,
                 int index,
                 int eventIndex,
                 java.lang.String titulo,
                 java.lang.String contenido,
                 java.lang.String fechaInicio,
                 java.lang.String fechaFin,
                 java.lang.String lugar)
    throws com.google.gdata.util.ServiceException,
           java.io.IOException
```

Throws:

com.google.gdata.util.ServiceException

java.io.IOException

consultarEvento

```
public void
consultarEvento(com.google.gdata.client.calendar.CalendarService service,
                 java.net.URL feedURL,
                 int index,
                 int eventIndex)
    throws com.google.gdata.util.ServiceException,
           java.io.IOException
```

Método que consulta los datos de un evento

Throws:

com.google.gdata.util.ServiceException

java.io.IOException

pt_google_calendar

Class Cambiar_permisos

```
java.lang.Object
├── java.awt.Component
│   ├── java.awt.Container
│   │   ├── java.awt.Window
│   │   │   ├── java.awt.Frame
│   │   │   │   └── javax.swing.JFrame
│   │   │   │       └── pt_google_calendar.Cambiar_permisos
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class Cambiar_permisos
extends javax.swing.JFrame
```

Esta clase se encarga de la gestión de los usuarios secundarios de los calendarios

Constructor Summary

Cambiar_permisos ()	
-------------------------------------	--

Method Summary

static void	main (java.lang.String[] args)
-------------	--

Constructor Detail

Cambiar_permisos

```
public Cambiar_permisos()
    throws java.io.IOException,
           com.google.gdata.util.ServiceException
```

Throws:

java.io.IOException

com.google.gdata.util.ServiceException

Method Detail

main

```
public static void main(java.lang.String[] args)
```

pt_calendar

Class ConexionInternet

java.lang.Object

└─ pt_calendar.ConexionInternet

```
public class ConexionInternet  
extends java.lang.Object
```

Esta clase nos dice si tenemos conexión a Internet

Constructor Summary

[ConexionInternet](#) ()

Method Summary

java.lang.String [getConnectionStatus](#) ()

El método regresa si hay conexión a internet o no

Constructor Detail

ConexionInternet

```
public ConexionInternet()
```

Method Detail

getConnectionStatus

```
public java.lang.String getConnectionStatus()
```

El método regresa si hay conexión a internet o no

pt_calendar

Class ConexionMySQL

```
java.lang.Object
```

```
└─ pt_calendar.ConexionMySQL
```

```
public class ConexionMySQL
```

```
extends java.lang.Object
```

Esta clase hace la conexión a la base de datos

Constructor Summary

ConexionMySQL()

Method Summary

java.sql.Connection	Conectar()
---------------------	----------------------------

Hace la conexión con la base de datos

Constructor Detail

ConexionMySQL

```
public ConexionMySQL()
```

Method Detail

Conectar

```
public java.sql.Connection Conectar()
```

Hace la conexión con la base de datos

pt_calendar

Class Contactos

```
java.lang.Object
```

```
└─ pt_calendar.Contactos
```

```
public class Contactos  
extends java.lang.Object
```

Esta clase regresa los contactos de Gmail

Constructor Summary

<pre>Contactos(java.lang.String user, java.lang.String pass) Constructor de Clase</pre>

Method Summary

<pre>java.util.ArrayList<java.lang.String></pre>	<pre>obtener() Devuelve una lista con todos los contactos</pre>
--	---

Constructor Detail

Contactos

```
public Contactos(java.lang.String user,  
java.lang.String pass)
```

```
throws java.net.MalformedURLException,  
        com.google.gdata.util.AuthenticationException
```

Constructor de la clase

Throws:

```
java.net.MalformedURLException
```

```
com.google.gdata.util.AuthenticationException
```

Method Detail

obtener

```
public java.util.ArrayList<java.lang.String> obtener()
```

throws

```
com.google.gdata.util.ServiceException,
```

```
java.io.IOException
```

Devuelve una lista con todos los contactos

Throws:

```
com.google.gdata.util.ServiceException
```

```
java.io.IOException
```

pt_google_calendar

Class Grupo_grupos

```
java.lang.Object
├── java.awt.Component
│   ├── java.awt.Container
│   │   ├── java.awt.Window
│   │   │   ├── java.awt.Frame
│   │   │   │   └── javax.swing.JFrame
│   │   │   │       └── pt_google_calendar.Grupo_grupos
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class Grupo_grupos
extends javax.swing.JFrame
```

Esta clase se encarga de la gestión de grupos de grupos

Constructor Summary

[Grupo_grupos](#) ()

Method Summary

static void [main](#)(java.lang.String[] args)

Constructor Detail

Grupo_grupos

```
public Grupo_grupos()
    throws java.net.MalformedURLException,
           com.google.gdata.util.AuthenticationException,
           com.google.gdata.util.ServiceException,
           java.io.IOException,
           java.sql.SQLException
```

Throws:

```
java.net.MalformedURLException  
com.google.gdata.util.AuthenticationException  
com.google.gdata.util.ServiceException  
java.io.IOException  
java.sql.SQLException
```

Method Detail

main

```
public static void main(java.lang.String[] args)
```

pt_google_calendar

Class Grupos

```
java.lang.Object  
└─ java.awt.Component  
    └─ java.awt.Container  
        └─ java.awt.Window  
            └─ java.awt.Frame  
                └─ javax.swing.JFrame  
                    └─ pt_google_calendar.Grupos
```

All Implemented Interfaces:

```
java.awt.image.ImageObserver, java.awt.MenuContainer,  
java.io.Serializable, javax.accessibility.Accessible,  
javax.swing.RootPaneContainer, javax.swing.WindowConstants
```

```
public class Grupos  
extends javax.swing.JFrame
```

Esta clase se encarga de la gestión de grupos simples

Constructor Summary

[Grupos](#) ()

Method Summary

static void [main](#)(java.lang.String[] args)

Constructor Detail

Grupos

```
public Grupos ()
    throws java.net.MalformedURLException,
           com.google.gdata.util.AuthenticationException,
           com.google.gdata.util.ServiceException,
           java.io.IOException,
           java.sql.SQLException
```

Throws:

- java.net.MalformedURLException
- com.google.gdata.util.AuthenticationException
- com.google.gdata.util.ServiceException
- java.io.IOException
- java.sql.SQLException

Method Detail

main

```
public static void main(java.lang.String[] args)
```

pt_calendar

Class Login

java.lang.Object

└ pt_calendar.Login

```
public class Login
extends java.lang.Object
```

Clase que accesa al servicio de GoogleCalendar de un usuario.

Constructor Summary

[Login](#) ()

Method Summary

com.google.gdata.client.calendar.Calendar Service	getCalenderService () Devuelve la información necesaria para el login
java.lang.String	getPassword () Obtiene el password de la cuenta de correo gmail
com.google.gdata.client.calendar.Calendar Service	getService () Obtiene en nombre del servicio
java.lang.String	getUserName () Obtiene el nombre de usuario de la cuenta de correo gmail
boolean	login () Se insertan las credenciales

		para acceder al servicio GoogleCalendar
	void	<u>setPassword</u> (java.lang.String pass) Establece el password de la cuenta de correo gmail.
	void	<u>setUserName</u> (java.lang.String user) Establece el nombre de usuario de la cuenta correo gmail

Constructor Detail

Login

```
public Login()
```

Method Detail

setUserName

```
public void setUserName(java.lang.String user)
```

Establece el nombre de usuario de la cuenta correo gmail

setPassword

```
public void setPassword(java.lang.String pass)
```

Establece el password de la cuenta de correo gmail.

getUserName

```
public java.lang.String getUserName()
```

Obtiene el nombre de usuario de la cuenta de correo gmail

getPassword

```
public java.lang.String getPassword()
```

Obtiene el password de la cuenta de correo gmail

getService

```
public com.google.gdata.client.calendar.CalendarService getService()
```

Obtiene el nombre del servicio

login

```
public boolean login()
```

Se insertan las credenciales para acceder al servicio GoogleCalendar

getCalenderService

```
public com.google.gdata.client.calendar.CalendarService  
getCalenderService()
```

Devuelve la información necesaria para el login

pt_google_calendar

Class LogIn

```
java.lang.Object
├─ java.awt.Component
│   └─ java.awt.Container
│       └─ java.awt.Window
│           └─ java.awt.Frame
│               └─ javax.swing.JFrame
│                   └─ pt_google_calendar.LogIn
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class LogIn
extends javax.swing.JFrame
```

Esta clase es la que verifica la información del usuario y permite el ingreso a la aplicación

Constructor Summary	
LogIn()	
Method Summary	
com.google.gdata.client.calendar.CalendarService	getCalendarService() Obtiene el servicio
java.sql.Connection	getcon() Obtiene la conexión a la base de datos
java.lang.String	getLogin()

	Obtiene el login
java.lang.String	getPassword() Obtiene la contraseña
static void	main (java.lang.String[] args)
void	setcon (java.sql.Connection c on) Establece la conexión a la base de datos

Constructor Detail

LogIn

```
public LogIn()
```

Method Detail

getLogin

```
public java.lang.String getLogin()
```

Obtiene el login

getPassword

```
public java.lang.String getPassword()
```

Obtiene la contraseña

getCalendarService

```
public com.google.gdata.client.calendar.CalendarService
```

```
getCalendarService()
```

Obtiene el servicio

setcon

```
public void setcon(java.sql.Connection con)
```

Establece la conexión a la base de datos

getcon

```
public java.sql.Connection getcon()
```

Obtiene la conexión a la base de datos

main

```
public static void main(java.lang.String[] args)
```

pt_google_calendar

Class main

java.lang.Object

└ pt_google_calendar.main

```
public class main
extends java.lang.Object
```

Constructor Summary

main()	
------------------------	--

Method Summary

static void	main (java.lang.String[] args)
-------------	--

Constructor Detail

main

```
public main()
```

Method Detail

main

```
public static void main(java.lang.String[] args)
```

pt_google_calendar

Class menu_grupos

```
java.lang.Object
├─ java.awt.Component
│   └─ java.awt.Container
│       └─ java.awt.Window
│           └─ java.awt.Frame
│               └─ javax.swing.JFrame
│                   └─ pt_google_calendar.menu_grupos
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class menu_grupos
extends javax.swing.JFrame
```

Constructor Summary

menu_grupos ()	
--------------------------------	--

Method Summary

static void	main (java.lang.String[] args)
-------------	--

Constructor Detail

```
menu_grupos
public menu_grupos ()
```

Method Detail

```
main
public static void main (java.lang.String[] args)
```

pt_google_calendar

Class Nuevo_calendario

```
java.lang.Object
├─ java.awt.Component
│   └─ java.awt.Container
│       └─ java.awt.Window
│           └─ java.awt.Frame
│               └─ javax.swing.JFrame
│                   └─ pt_google_calendar.Nuevo_calendario
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class Nuevo_calendario
extends javax.swing.JFrame
```

Esta clase se encarga de la gestión de calendarios

Constructor Summary	
Nuevo_calendario()	

Method Summary	
static void	main (java.lang.String[] args)

Constructor Detail	
--------------------	--

Nuevo_calendario

```
public Nuevo_calendario()
    throws java.io.IOException,
           com.google.gdata.util.ServiceException
```

Throws:

java.io.IOException

com.google.gdata.util.ServiceException

Method Detail

main

```
public static void main(java.lang.String[] args)
    throws java.io.IOException,
           com.google.gdata.util.ServiceException
```

Throws:

java.io.IOException

com.google.gdata.util.ServiceException

pt_google_calendar

Class Nuevo_evento

```
java.lang.Object
├─ java.awt.Component
│   └─ java.awt.Container
│       └─ java.awt.Window
│           └─ java.awt.Frame
│               └─ javax.swing.JFrame
│                   └─ pt_google_calendar.Nuevo_evento
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class Nuevo_evento
    extends javax.swing.JFrame
```

Esta clase se encarga de la gestión de eventos

Constructor Summary

[Nuevo_evento](#) ()

Method Summary

static void [main](#)(java.lang.String[] args)

Constructor Detail

Nuevo_evento

```
public Nuevo_evento()  
    throws java.io.IOException,  
           com.google.gdata.util.ServiceException,  
           java.sql.SQLException
```

Throws:

java.io.IOException

com.google.gdata.util.ServiceException

java.sql.SQLException

Method Detail

main

```
public static void main(java.lang.String[] args)
```

pt_google_calendar

Class Principal

```
java.lang.Object
├── java.awt.Component
│   ├── java.awt.Container
│   │   ├── java.awt.Window
│   │   │   ├── java.awt.Frame
│   │   │   │   └── javax.swing.JFrame
│   │   │   │       └── pt_google_calendar.Principal
```

All Implemented Interfaces:

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class Principal
extends javax.swing.JFrame
```

Esta es la ventana principal

Constructor Summary

[Principal](#)()

Method Summary

static void [main](#)(java.lang.String[] args)

Constructor Detail

Principal

```
public Principal()
    throws java.io.IOException,
           com.google.gdata.util.ServiceException
```

Throws:

java.io.IOException

com.google.gdata.util.ServiceException

Method Detail

main

```
public static void main(java.lang.String[] args)
    throws java.io.IOException,
           com.google.gdata.util.ServiceException
```

Throws:

java.io.IOException

com.google.gdata.util.ServiceException