

**UNIVERSIDAD CARLOS III DE MADRID**

**ESCUELA POLITÉCNICA SUPERIOR**

**PROYECTO  
CATEBus:  
CONTROL DE  
ASISTENCIA EN  
TRANSPORTE  
ESCOLAR**

**PROYECTO FIN DE CARRERA  
INGENIERÍA INFORMÁTICA**

**AUTOR: D. JAVIER SANABRIA FERNÁNDEZ  
TUTOR: D. VICENTE LUQUE CENTENO  
OCTUBRE 2011**

# AGRADECIMIENTOS

---

En primer lugar querría agradecer a mi familia su incondicional apoyo durante toda mi etapa universitaria y durante la realización de este proyecto.

Agradecer también a mis compañeros y amigos, que tantos buenos momentos hemos vivido en este tiempo, y por su paciencia y apoyo durante la realización tanto de la aplicación como de este documento.

Desearía agradecer especialmente a Pablo Pérez Rabadán, cuya idea de realizar un programa para que su hermana no perdiera el autobús, ha derivado en la realización de este proyecto fin de carrera. Gracias por tu ayuda, comprensión y apoyo siempre que lo he necesitado, y también por tu preocupación por el desarrollo del mismo. Me satisface que una idea nuestra se haya convertido en realidad.

# ÍNDICE DE CONTENIDOS

1.	INTRODUCCIÓN .....	11
1.1.	MOTIVACIÓN.....	11
1.2.	ALCANCE Y OBJETIVOS GENERALES .....	12
1.3.	DESARROLLO DE LA MEMORIA.....	13
2.	ESTADO DE LA CUESTIÓN .....	15
2.1.	SITUACIÓN ACTUAL DEL PROBLEMA.....	15
2.2.	ANÁLISIS DE APLICACIONES EXISTENTES .....	16
2.2.1.	Programa Papás .....	17
2.2.2.	iCare .....	18
2.2.3.	Attendance Plus.....	19
2.3.	COMPARATIVA DE HERRAMIENTAS EXISTENTES .....	19
2.4.	ANÁLISIS DE TECNOLOGÍAS EXISTENTES .....	21
2.5.	CONCLUSIONES .....	21
3.	TECNOLOGÍAS UTILIZADAS .....	23
3.1.	Java Enterprise Edition .....	23
3.1.1.	Servlets.....	23
3.1.2.	Páginas JSP .....	24
3.2.	JDBC.....	25
3.3.	SQL .....	25
3.4.	MySQL .....	26
3.5.	APACHE TOMCAT.....	26
3.6.	JAVASCRIPT .....	26
3.7.	AJAX.....	27
3.8.	JSON .....	28
3.9.	GOOGLE MAPS.....	29

3.10.	JAVA MAIL .....	30
3.11.	ITEXT .....	30
3.12.	JAVA CSV LIBRARY .....	31
3.13.	LIBRERÍAS COMMONS FILEUPLOAD Y COMMONS IO .....	32
4.	DESCRIPCIÓN DEL SISTEMA .....	33
4.1.	CARACTERÍSTICAS GENERALES .....	33
4.2.	FUNCIONES PRINCIPALES PARA PADRES .....	34
4.3.	FUNCIONES PRINCIPALES PARA EL PERSONAL DEL CENTRO .....	35
4.4.	FUNCIONES PRINCIPALES PARA LOS RESPONSABLES DE RUTA .....	36
5.	DISEÑO E IMPLEMENTACIÓN DE LA BASE DE DATOS .....	37
5.1.	DISEÑO DE LA BASE DE DATOS .....	37
5.1.1.	Diagrama Entidad-Relación .....	37
5.1.2.	Diagrama Relacional.....	39
5.1.3.	Descripción De Los Atributos De Las Tablas .....	43
5.2.	CREACIÓN DE USUARIOS Y PERMISOS A LA BASE DE DATOS .....	47
5.3.	DISEÑO DE LAS PRINCIPALES CONSULTAS SOBRE LA BASE DE DATOS .....	47
6.	COMPONENTES DE LA APLICACIÓN .....	49
6.1.	ARQUITECTURA DEL SISTEMA .....	49
6.2.	IMPLEMENTACIÓN DEL SISTEMA.....	51
6.2.1.	Vista .....	51
6.2.2.	Controlador .....	59
6.2.3.	Modelo .....	76
6.2.4.	Descripción de la implementación de las funcionalidades.....	76
7.	CONCLUSIONES Y TRABAJOS FUTUROS .....	97
7.1.	CONCLUSIONES .....	97
7.1.1.	Comprobación de cumplimiento de los requisitos.....	99

7.1.2.	Ventajas e inconvenientes de uso .....	101
7.2.	TRABAJOS FUTUROS .....	103
7.2.1.	Creación de consultas, listados y tratamiento de datos avanzados .....	103
7.2.2.	Creación de una aplicación basada en Android o para iPhone .....	103
7.2.3.	Comunicación vía SMS con los padres .....	104
7.2.4.	Inclusión de un servicio de localización del autobús .....	104
7.2.5.	Cobertura a centros de trabajo .....	104
7.2.6.	Informes personalizados .....	105
8.	ESTIMACIÓN Y PLANIFICACIÓN DEL PROYECTO .....	106
8.1.	CICLO DE VIDA .....	106
8.2.	ESTIMACIÓN DEL PROYECTO .....	107
8.2.1.	Resultados de la planificación .....	109
8.3.	PLANIFICACIÓN .....	115
8.4.	PRESUPUESTO .....	117
8.4.1.	Recursos Materiales Utilizados .....	117
8.4.2.	Salarios del personal .....	119
8.4.3.	Gastos indirectos .....	120
8.4.4.	Resumen .....	120
9.	INSTALACIÓN DE LA APLICACIÓN Y MANUAL DE USUARIO .....	122
9.1.	INSTALACIÓN .....	122
9.1.1.	Instalación de Linux .....	122
9.1.2.	Instalación de Java .....	123
9.1.3.	Instalación de MySQL .....	124
9.1.4.	Instalación de Apache Tomcat .....	124
9.1.5.	Creación de la base de datos .....	125
9.1.6.	Despliegue de la aplicación .....	125

9.2.	MANUAL DE USUARIO .....	126
9.2.1.	Personal Docente .....	126
9.2.2.	Padres.....	139
9.2.3.	Responsables de ruta.....	146
9.2.4.	Accesibilidad vía móvil.....	147
10.	REFERENCIAS .....	158

# ÍNDICE DE FIGURAS

Ilustración 1. Diagrama Entidad-Relación.....	39
Ilustración 2. Diagrama relacional .....	41
Ilustración 3: Ciclo del MVC .....	50
Ilustración 4. Diagrama de clases general.....	60
Ilustración 5. Diagrama de clases del paquete 'helper'.....	62
Ilustración 6. Diagrama de clases de operaciones.....	64
Ilustración 7. Diagrama de secuencia de inicio de sesión. ....	77
Ilustración 8. Diagrama de secuencia de nueva ruta. ....	79
Ilustración 9. Diagrama de secuencia de eliminar ruta.....	80
Ilustración 10. Diagrama de secuencia de insertar parada. ....	80
Ilustración 11. Diagrama de secuencia de modificación de parada.....	81
Ilustración 12. Diagrama de secuencia de eliminar parada.....	82
Ilustración 13. Diagrama de secuencia de alta responsable.....	83
Ilustración 14. Diagrama de secuencia de baja responsable.....	83
Ilustración 15. Diagrama de secuencia de asignar responsable a ruta. ....	84
Ilustración 16. Diagrama de secuencia de alta Padre. ....	85
Ilustración 17. Diagrama de secuencia de añadir alumno.....	85
Ilustración 18. Diagrama de secuencia de lista de alumnos por ruta. ....	86
Ilustración 19. Diagrama de secuencia de lista de alumnos con faltas.....	87
Ilustración 20. Diagrama de secuencia de entregar justificante. ....	87
Ilustración 21. Diagrama de secuencia de recorrer ruta .....	90
Ilustración 22. Diagrama de secuencia de información de ruta.....	91
Ilustración 23. Diagrama de secuencia de información parada. ....	92
Ilustración 24. Diagrama de secuencia de parada del curso.....	92
Ilustración 25. Diagrama de secuencia de parada ocasional. ....	93
Ilustración 26. Diagrama de secuencia de solicitar retraso. ....	93
Ilustración 27. Diagrama de secuencia de programar falta. ....	94
Ilustración 28. Diagrama de secuencia de historial de faltas.....	95
Ilustración 29. Diagrama de Secuencia de justificantes. ....	95
Ilustración 30. Diagrama de secuencia de cambio de contraseña .....	96
Ilustración 31. Ciclo de vida .....	107
Ilustración 32. Calendario del Proyecto por Fases.....	110

Ilustración 33. Coste Acumulado por Fase .....	111
Ilustración 34. Esfuerzo por Fases.....	112
Ilustración 35. Esfuerzo por módulo .....	112
Ilustración 36. Personal por fase del proyecto .....	113
Ilustración 37. Actividades por fase.....	114
Ilustración 38. Diagrama Gant .....	116
Ilustración 39. Diagrama de Gantt – 2.....	117
Ilustración 41. index.jsp .....	127
Ilustración 42. Menú desplegable .....	127
Ilustración 43. Añadir nueva ruta .....	128
Ilustración 44. Añadir paradas a ruta .....	129
Ilustración 45. Eliminar ruta.....	130
Ilustración 46. Añadir nueva parada .....	130
Ilustración 47. Modificar Parada .....	131
Ilustración 48. Borrado de parada.....	132
Ilustración 49. Añadir nuevo usuario .....	133
Ilustración 50. Correo electrónico enviado. ....	133
Ilustración 51. Eliminación de responsables.....	134
Ilustración 52. Asignación de Rutas .....	135
Ilustración 53. Nuevo Alumno.....	135
Ilustración 54. Importar alumnos desde fichero.....	136
Ilustración 55. Ejemplo de archivo de datos.....	137
Ilustración 56. Listados.....	137
Ilustración 57. Entrega de justificantes. ....	138
Ilustración 58. Personalizar opciones.....	139
Ilustración 59. Información de Rutas .....	140
Ilustración 60. Información de la parada.....	141
Ilustración 61. Asignación de parada del curso. ....	142
Ilustración 62. Parada ocasional. ....	142
Ilustración 63. Pedir Retraso.....	143
Ilustración 64. Programar falta de asistencia.....	144
Ilustración 65. Generar Justificante .....	144
Ilustración 66. Historial de faltas. ....	145
Ilustración 67. Cambio de Contraseña .....	146
Ilustración 68. Marcación de alumnos .....	147

Ilustración 69. Opciones móviles padres .....	148
Ilustración 70. Elección de alumno.....	148
Ilustración 71. Datos de la parada en móviles .....	149
Ilustración 72. Introducción de fecha .....	150
Ilustración 73. Menú principal personal docente .....	151
Ilustración 74. Menú gestionar rutas.....	151
Ilustración 75. Nueva ruta móvil .....	152
Ilustración 76. Menú gestionar paradas .....	153
Ilustración 77. Nueva parada móvil .....	153
Ilustración 78. Opciones de gestionar parada .....	154
Ilustración 79. Menú gestionar responsables de ruta .....	154
Ilustración 80. Nuevo responsable móvil .....	155
Ilustración 81. Opciones de gestionar responsables de ruta .....	155
Ilustración 82. Menú gestionar padres y alumnos.....	156
Ilustración 83. Opciones gestionar padres y alumnos .....	156
Ilustración 84. Añadir nuevo alumno móvil.....	157

# ÍNDICE DE TABLAS

---

Tabla 1 - Comparativa de herramientas.....	19
Tabla 2. Multiplicadores de Esfuerzo.....	108
Tabla 3. Factores de Escala COCOMOII.....	108
Tabla 4. Estimación global del Proyecto.....	109
Tabla 5. Estimación del calendario por fases.....	109
Tabla 6. Coste del proyecto.....	110
Tabla 7. Esfuerzo por fases.....	111
Tabla 8. Recursos humanos.....	113
Tabla 9. Esfuerzo por actividades.....	114
Tabla 10. Fechas estimadas del proyecto.....	115
Tabla 11. Fechas estimadas del proyecto por fases.....	115
Tabla 12. Ampliación de la estimación total.....	116
Tabla 13. Estimación de fechas de las nuevas funcionalidades por fases.....	116
Tabla 14. Gastos de recursos materiales.....	119
Tabla 15. Sueldos por categoría.....	119
Tabla 16. Salarios reales del proyecto.....	120
Tabla 17. Gastos indirectos asociados al proyecto.....	120
Tabla 18. Resumen del presupuesto.....	120

# 1. INTRODUCCIÓN

---

## 1.1. MOTIVACIÓN

Debido a la proliferación de centros educativos concertados/privados a las afueras de las ciudades, se ha hecho necesaria la utilización del transporte escolar como parte fundamental de la vida diaria de los alumnos. Este sistema de transporte, costado en gran medida por los padres de los alumnos, permite que, parando en ciertos puntos de las ciudades (realizando trazados específicos denominados rutas), los alumnos puedan esperar a su llegada para poder ser transportados al centro educativo. De esta forma, se concretan diversas horas en las que deben pasar a recoger a los alumnos, teniendo en cuenta factores tales como la distancia al centro, el flujo de tráfico, etc.

Gracias a este sistema, los padres de los alumnos pueden delegar en el centro el transporte de sus hijos, ya que debido tanto al nivel de vida, como a las necesidades de las familias, muchos padres trabajadores no tienen a quien delegar el transporte de sus hijos a los centros. Con la aparición de las rutas, los padres pueden acompañar a sus hijos a las paradas de las mismas antes de dirigirse a sus puestos de trabajos, de forma que mantienen controlados a los mismos, con la seguridad de es transporte directo al centro, en el que estarán controlados.

Pero al tratarse del transporte de escolares, pueden surgir numerosos casos en los que el alumnado no pueda acudir a coger la ruta de transporte a la hora acordada. Dichos casos se deben a que los alumnos se retrasan, bien porque se duermen, se entretengan en sus actividades antes de acudir al centro (desayunar, vestirse, etc.). En estos casos, los padres son conscientes de que los alumnos no asistirán a clase en dichas ocasiones, o que se pueden retrasar en acudir al punto de parada de la ruta (dentro de un margen de tiempo antes de que empiecen las clases). En estos casos, los padres acuden a la parada de la ruta y comprueban que no hay ningún alumno más esperando, lo que significa que la ruta ya ha pasado, y que deben ser ellos quienes transporten a sus hijos al centro.

Otras situaciones en las que se pueden dar casos parecidos, es en aquéllos donde la ruta pasa por las paradas antes de la hora prevista (por condiciones de buen tráfico, por ejemplo). Si se pudiese realizar un control en el que los padres puedan avisar de estos hechos y que las rutas pudiesen esperar a estos casos, tanto los padres (que harían pleno uso del servicio de transporte) y el centro (que no tendría que controlar los alumnos que acuden al

mismo por transporte privado) se verían beneficiados. Además. Los padres podrían indicar otra parada en la que sus hijos podrán subirse a la ruta, de esta forma, si la pierden en una parada, poder acudir a otra distinta para la utilización del servicio.

También puede no asistir a clase en días concretos, bien por enfermedad o causa médica (revisiones, vacunaciones, etc.). En dichos días, a las rutas acudirán un menor número de alumnos, incluso alguna de las paradas podría no ser necesaria, pudiendo optimizar la ruta de viaje, lo que supondría un ahorro tanto de tiempo, como de dinero (menor consumo del transporte). Además, en estos casos se puede agilizar la generación de justificantes de falta de asistencia, tanto por parte de los padres, como por del personal del centro.

Debido a estos hechos, se comprueba la necesidad, tanto de padres como de los responsables de los centros educativos, tanto de llevar un control específico del alumnado desde el primer contacto con el centro, como sucede con la asistencia de los mismos a las clases. De esta forma, los responsables de los centros pueden tener, desde antes incluso de que los alumnos utilicen las rutas de transporte, el número de ellos que asistirán a clase. Además, podrán agilizar y optimizar el sistema de rutas de transporte.

## 1.2. ALCANCE Y OBJETIVOS GENERALES

Con la realización de este proyecto, se propone controlar el acceso de los alumnos a los centros desde el momento en que se realiza el uso del sistema de transporte proporcionado por el mismo. De esta forma, se realizará una aplicación web, por la cual se podrán acceder a las rutas de transporte correspondiente, pudiendo ser administradas por parte de los centros, y que posibilite la interacción por parte de los padres de los alumnos.

De esta manera, los padres podrán avisar al centro, y, por consiguiente al sistema de transporte, de las posibles incidencias que puedan ocurrir y que afecten a la recogida de los alumnos en los puntos de ruta, tales como retrasos o no asistencia de los mismos; pudiendo dar la opción de elegir paradas alternativas de recogida, pedir que la ruta espere un poco de tiempo (dentro de un determinado margen), o que no espere a un determinado alumno en la parada.

Así, evitamos que los alumnos no utilicen el sistema de autobús de ruta, lo pierdan, o que el propio autobús espere de manera innecesaria cuando un alumno no va a asistir a las clases. Este hecho es poco beneficioso, tanto para los padres como para el colegio y los autobuses, ya que pueden realizar trayectos innecesarios.

Por parte de los centros y personal educativo, se podrá controlar los niños asociados a cada ruta, así como visualizar aquéllos que no realizarán el uso del servicio de transporte en determinadas fechas, o que han solicitado un retraso en el momento de llegada al punto de ruta.

Además, al realizar este control, podrán controlar la asistencia de los alumnos al centro o a determinadas clases, haciéndole llegar a los profesores qué alumnos no han cogido la ruta, con lo que no irán a clase, y poder generar justificantes de asistencia automáticamente.

Se facilitará también el acceso a los responsables de las rutas. Así, cada mañana los responsables sabrán a qué alumnos recoger y si deben esperar un tiempo específico a recoger a algún niño, o no pasar por diversos puntos; automatizando los servicios de rutas, y realizando nuevos recorridos (en caso de ser necesario) según estos hechos.

### **1.3. DESARROLLO DE LA MEMORIA**

En este apartado se va a explicar el contenido de las secciones que componen este documento.

En el capítulo 2, denominado estado de la cuestión, se tratará de dar una visión general del transporte escolar, indicando las principales deficiencias de control, así como referencias al aspecto económico y social del mismo. También se analizarán las principales herramientas existentes para el control de la asistencia de los alumnos a los centros escolares, así como una comparativa de cada una de ellas, y de lo que la aplicación a construir tiene en común con ellas, además de las novedades que aporta.

En el capítulo 3, se describirán las principales herramientas, APIs y tecnologías que se han utilizado en el desarrollo de este proyecto, así como en la implementación del mismo.

En el capítulo siguiente, se realiza una descripción del sistema que se implementará, detallándose las principales funcionalidades y requisitos que contendrá. De esta forma se obtiene una visión específica de lo que se va a construir, y de la amplitud a abarcar en la aplicación.

Seguidamente, en el capítulo 5, se detallará el diseño y la implementación de la base de datos que mantendrá los datos que la aplicación maneje. Además, se especificarán algunas de las consultas que se han implementado.

En el capítulo 6, se detalla el diseño y la implementación de la aplicación. Para ello, se especifica el modelo arquitectónico en el que se basará el sistema, explicando los componentes que se han implementado en el mismo. Además, se explicarán y detallarán las funciones y operaciones que la aplicación realiza, además de argumentar cómo se utilizan las tecnologías detalladas en el capítulo 3 para lograr un correcto funcionamiento de la aplicación.

En el capítulo siguiente se explican las conclusiones de la aplicación, en las cuales se detalla el proceso de realización de este proyecto, comentando los problemas encontrados. Además, se realiza una crítica al mismo, destacando los puntos a favor y en contra de su utilización. Por último, se detallan las futuras líneas de investigación que se podrían seguir para la mejora o ampliación de este proyecto.

En el capítulo 8 se detalla la estimación del proyecto, en la que se especifican el ciclo de vida seguido, la estimación de los recursos, esfuerzo y calendario necesarios para la realización del mismo, así como un calendario detallado de la realización de cada una de las fases del proyecto. Además se detalla el presupuesto del mismo, indicando gastos, recursos y salarios.

En el capítulo 9, se adjunta el manual de usuario, que será entregado a los consumidores de la aplicación. En él se detallan, de forma textual y gráfica, las operaciones a realizar con la aplicación, así como los datos a introducir y los resultados que producirá en el sistema.

Por último, se incluyen las referencias a los libros y páginas web en las que se ha buscado información, o de las que se ha hecho uso y referencia directa para la implementación del proyecto o la redacción de esta memoria.

## 2. ESTADO DE LA CUESTIÓN

---

### 2.1. SITUACIÓN ACTUAL DEL PROBLEMA

Debido a la situación y condiciones de crecimiento demográfico de las ciudades, muchos de los centros educativos existentes han sido construidos según las necesidades de este crecimiento; dando lugar a zonas en las que no existen estos centros y, por el contrario, otras zonas donde se encuentran concentrados, obligando a los estudiantes a desplazarse hacia ellos.

En el caso de los centros públicos de enseñanza, la mayoría de las ciudades, poseen líneas de autobuses urbanos o interurbanos, u otros medios de transporte (metro, tranvía), que hace posible el transporte de alumnos hacia los centros de enseñanza; aunque en muchos casos son los propios padres de los alumnos los que transportan a sus hijos en transporte privado.

En algunas comunidades autónomas, como la Comunidad de Madrid, poseen decretos [1] que regulan un servicio de transporte escolar gratuito para aquellas poblaciones de las que no dispongan centros escolares, o cuya situación diste considerablemente del núcleo de población; como por ejemplo en el IES Camilo José Cela de Pozuelo de Alarcón, si bien, no oferta este servicio para aquéllos centros públicos que no cumpla dichas condiciones, o centros de otra índole.

En el caso de los centros de enseñanza concentrada/privada, la situación es distinta. En primer lugar, al no ser de propiedad pública, las ciudades no reservan suelo público para la construcción de los mismos. Debido a esto, muchos de dichos centros se encuentran alejados de los núcleos urbanos o en las afueras de las ciudades.

Para el transporte de los alumnos a dichos centros, se emplean servicios de rutas de autobuses, que recogen a éstos en ciertos puntos o paradas de las ciudades. Dichas rutas están controladas por el personal de transporte que únicamente controla el trayecto realizado y que se detiene a la hora indicada en cada parada, para llegar a su destino a la hora acordada.

El problema al que se enfrentan muchos de los padres que utilizan este servicio es la pérdida de la ruta, es decir, que el autobús no haya esperado el tiempo suficiente al alumno, o que, al tardar menos en otras paradas, haya pasado con antelación. Lo que supone

desaprovechar el sistema contratado, y la necesidad de los padres de utilizar el transporte privado para el desplazamiento hacia el centro escolar.

Por otro lado, pueden existir días en que parte del alumnado no asista a clase (por enfermedad, por ejemplo) o que se tenga que desplazar a otra parada extraordinariamente, de forma que las rutas estándares no sean las más adecuadas para realizar el servicio, ya que existirían paradas con una afluencia mínima o nula de alumnos, resultando un coste innecesario, el cual se podría evitar si los padres comunicasen con antelación la no asistencia de su hijo a clase, o la utilización de nuevas paradas para este servicio; pudiéndose realizar un inspección de las rutas de transporte.

De este modo, se puede realizar un control exhaustivo del alumnado al centro, conociendo desde un primer momento los alumnos que asistirán a clase, de forma que se puedan optimizar los procesos de registros de faltas de asistencia y generación de justificantes, además de proporcionar a los padres la información en tiempo real.

Debido a que los beneficios de estos centros provienen de los padres de los alumnos, además de alguna sociedad colaboradora, es importante optimizar el servicio de transporte escolar, de forma que suponga el menor coste posible (en términos de tiempo y dinero) al colegio, y en el fondo a los padres, y que ofrezca un servicio actualizado y de gran utilidad a las partes implicadas: padres, personal docente y servicio de transporte.

## **2.2. ANÁLISIS DE APLICACIONES EXISTENTES**

Existen hoy en día diversas aplicaciones para el control, tanto de la asistencia a clase de los alumnos, como del seguimiento escolar y rendimiento del alumno en clase; si bien este último punto se aleja del propósito del presente proyecto. De estas aplicaciones se pueden distinguir aquellas orientadas únicamente al profesorado y personal docente, y a otras en las que los padres y madres de los alumnos pueden participar con el centro educativo en el control y orientación de los mismos.

Cabe destacar que entre estas aplicaciones, existen tanto soluciones públicas (financiadas por ministerios y consejerías de educación), como herramientas privadas, que suministran a los clientes tanto los programas a utilizar, como las herramientas sobre las que se realizará el uso diario de las mismas.

Vamos a analizar las principales aplicaciones existentes que realizan estas operaciones, como son el Programa Papás [1], ofertado por la consejería de Educación y

Ciencia de la Junta de Comunidades de Castilla-La Mancha, el conjunto de herramientas iCare [2], perteneciente a la empresa Orgamation, o Attendance Plus [3], desarrollado por Rediker Software.

### 2.2.1. Programa Papás

Papás (<http://educacion.jccm.es/delphos-papas/>), es una iniciativa de la Consejería de Educación y Ciencia, accesible a través de Internet, por la que los centros educativos podrán ofrecer servicios por Internet a padres y alumnos. De esta manera, pretenden abrir un nuevo canal de comunicación entre el centro y las familias, mediante el cual mejorar la atención a los padres y madres del alumnado del centro.

Papás ofrece numerosos servicios para todas las partes interesadas, las cuales están diferenciadas en diversos niveles de acceso según su procedencia, así como distintos identificadores de usuarios. Además ofrece una interfaz personalizada para cada uno de ellos.

Para padres y tutores de los alumnos, ofrece un canal de comunicación personalizado por el cual podrán realizar un seguimiento del progreso escolar mediante datos actualizados acerca de deberes, controles, notas, faltas de asistencia, etc. Además, oferta una comunicación con los profesores de forma fácil y directa. Por último, permite el acceso a los servicios de información del centro y sus actividades (como el tablón de anuncios, o el servicio de mensajes a móviles o correo electrónico).

Para el alumnado, este servicio le posibilita el acceso a la información, estado y resultado de las tareas, deberes y controles asignados; además de facilitar una vía contacto con profesores, y consultar información relacionada con el centro.

Para el profesorado, esta aplicación les permite compartir los datos personalizados del seguimiento del progreso escolar con los alumnos y sus padres; además de facilitar una vía de comunicación fácil y directa con cada uno de ellos.

Este sistema, se ofrece en dos versiones, una versión adaptada para navegadores web estándares, para ser accesible desde cualquier ordenadores personal o portátil; y una versión para dispositivos móviles, adaptada para su visualización en PDAs, de forma que el personal docente puede utilizar los servicios directamente durante el horario escolar.

Existen numerosos centros educativos en Castilla-La Mancha, como el IES Virgen de Gracia en Puertollano (<http://www.iesvirgen.es/pagina/index.php>) que realizan un uso de este servicio.

### 2.2.2. iCare

iCare es un completo sistema software de gestión para el control de alumnos, que ofrece unos servicios personalizados a tres niveles: Para los directores y propietarios de los centros (que proporciona herramientas para mejorar la gestión ejecutiva y toma de decisiones), para el personal administrativo (proporcionando servicios que facilitan la automatización de las operaciones de facturación, programación, comida, etc.) y para profesores (suministrando medios para la comunicación con los padres y el seguimiento de los progresos de los alumnos).

De esta forma, iCare no solo ofrece un mantenimiento de registros y el seguimiento en tiempo real de todos los estudiantes, profesores, aulas y horarios; sino que también permite la automatización de los libros de cuentas de usuario, permitiendo recoger los datos completos de los derechos de matrícula; la organización de los registros médicos de los alumnos, recogiendo todos los datos médicos de éstos así como los registros de vacunación; y de los programas y menús de los comedores, de forma que permite gestionar la planificación de comidas, vigilar el cumplimiento de los estándares de nutrición y los controles de la producción de alimentos, compras e inventario.

Además, oferta un servicio web complementario, denominado iCareWeb, consistente en una colección de formularios web que funcionan directamente desde iCare. De esta forma se puede acceder a la aplicación iCare desde cualquier dispositivo conectado a Internet, de manera que pueda recoger y almacenar nuevos datos o permitir visualizar los datos a los padres y el personal autorizado. Los formularios de iCareWeb se ejecutan desde el propio sistema informático en la centro y están disponibles sólo para las personas que sean autorizadas para ello, mediante su usuario y contraseña. Según sea necesario, se puede activar, desactivar o cambiar la contraseña en cualquier momento para controlar el acceso a los datos.

Por último, esta empresa ofrece también una serie de hardware, desde el cual se pueda acceder de forma precisa a estos servicios, tales como ordenadores con pantallas táctiles, PDAs, ordenadores portátiles; así como herramientas de seguridad y acceso, tales como lectores de tarjetas, de huellas dactilares y sistemas de control de acceso.

Orgamation ofrece iCare para centros de cuidado infantil, centros de día, escuelas preescolares, guarderías, escuelas, etc.

### 2.2.3. Attendance Plus

Esta aplicación, integrada en el sistema de administración de centros educativos Administrator's Plus, es un módulo para la gestión de asistencia, diseñado para manejar los requisitos de asistencia de las escuelas públicas y privadas de todos los tamaños.

Este software permite introducir los datos de asistencia de manera rápida y fácil, usando las listas de los alumnos a través de sus fotos y su disposición en clase. También permite introducir estos datos manualmente o a través de un escáner.

Además, entre sus principales características destaca la posibilidad de la publicación en internet o en aplicaciones para PDAs de los datos de asistencia; realizar informes y justificantes de asistencia, y boletines vía correo electrónico con dichos datos, así como de definir periodos de asistencia para grupos de alumnos.

## 2.3. COMPARATIVA DE HERRAMIENTAS EXSISTENTES

A continuación se muestra una tabla comparativa, en la cual podemos observar las principales características de las aplicaciones anteriores, pudiendo compararlas con las propiedades y aportaciones de la herramienta a desarrollar en el mercado. Los síes referencian a características que presentan las aplicaciones, siendo los noes la representación de la ausencia de las mismas.

Tabla 1 - Comparativa de herramientas

	Control den el transporte	Control de asistencia en el centro	Control de servicios del centro	Herramientas de gestión de recursos	Comunicación con padres	Generación de justificantes	Generación de informes de asistencia	Suministro de material móvil	Aplicación en remoto (web)	Adaptado a dispositivos móviles	Idioma Español	Personalización según usuarios
<b>Programa Papás</b>	No	Sí	No	No	Sí	No	No	No	Sí	Sí	Sí	Sí
<b>iCare</b>	No	Sí	Sí	Sí	No	Sí	Sí	Sí	No	Sí	No	No
<b>Attendace Pus</b>	No	Sí	No	No	No	Sí	Sí	No	No	No	No	No
<b>Proyecto CATEBus</b>	Sí	Sí	No	No	Sí	Sí	Sí	No	Sí	Sí	Sí	No

Como podemos comprobar en la tabla resumen anterior, las arquitecturas estudiadas ofertan un control de la asistencia de los alumnos al centro integrada en cada una de ellas. En las siguientes características, podemos observar múltiples diferencias entre ellas. La más notable es que Attendace Plus no ofrece una aplicación web para acceder a sus servicios, ni está adaptado para dispositivos móviles. Por otro lado, iCare necesita de una aplicación paralela para ello.

Salvando estas diferencias, actualmente iCare es la herramienta software más completa para la gestión de centros escolares, ofreciendo al centro gran cantidad de herramientas administrativas para ello. Programa Papás es una gran herramienta gratuita, que fomenta la interacción de los padres, y de los propios alumnos, en las relaciones con el centro escolar, para el beneficio de ambas partes.

CATEBus se centrará en un aspecto no soportado por ninguna de las anteriores herramientas, que es el control del sistema de transporte, así como incorporar las principales características de las anteriores, que complementan al sistema implementado, y con las que se podrá automatizar gran parte del control de asistencia a los centros escolares.

Respecto a las características que soportan otras herramientas y CATEBus no, cabe destacar que es la más completa de las estudiadas (sólo cuatro; respecto a las cinco de iCare, seis de Programa Papás, y hasta ocho de Attendace Plus). La aplicación a desarrollar no soporta el control de servicios del centro ni la gestión de sus recursos, ya que está pensada para un uso del control de asistencia exclusivamente, servicio utilizado por padres y responsables de los centros educativos para la regulación de los alumnos; desechando cualquier otro cometido o función que se salga éste ámbito. Es más, los centros posiblemente tendrán sus propias herramientas para ello, por lo que CATEBus sería un complemento a todas ellas, y no una sustitución (con la migración de datos que acarrearía) de las ya existentes en ellos.

En cambio, con la característica del suministro del material móvil, no se posee un sistema de distribución de dicho material, si bien se puede aconsejar a los centros de las tecnologías existentes para orientar a la adquisición de los equipos pertinentes.

Por último, en referencia a la personalización según usuarios, al abarcar CATEBus una serie de funcionalidades tan definidas, concisas y claras, las funciones que se presentan a los usuarios de la aplicación no necesitan ser añadidas u ocultadas en función de los distintos tipos de usuario.

## 2.4. ANÁLISIS DE TECNOLOGÍAS EXISTENTES

Para realizar estos servicios, existen a día de hoy numerosos dispositivos tecnológicos que permiten la ejecución y el acceso a aplicaciones web.

En primer lugar, la tecnología actual de los sistemas de telecomunicaciones hace posible que el acceso a internet esté disponible en la mayoría de los hogares, disponiendo de múltiples velocidades de conexión según las necesidades de los usuarios. Además, las características de los exploradores web hacen que sea más rápido acceder a aplicaciones web por parte del usuario, así como de ofrecer unas mayores prestaciones por parte de los desarrolladores.

De entre los dispositivos portátiles, el más extendido para este propósito son las PDAs (*Personal Digital Assistant*) o agendas electrónicas. Gracias a estos dispositivos, se puede acceder a aplicaciones, tanto locales como a través de internet, a través de una pantalla táctil y de un tamaño considerable, pudiendo acceder a los mismos servicios como si de un ordenador de sobremesa se tratase.

Pero más recientemente, gracias al desarrollo de la tecnología móvil y a los avances en las conexiones a internet inalámbricas, muchos de los móviles de nueva generación del mercado, soportan navegación web a través de servicios como GPRS (*General Packet Radio Service*) y Wi-Fi, lo que permite a los usuarios acceder a la red y a las aplicaciones y servicios que ofrece de forma fácil en cualquier lugar en el que se encuentre. Además, la proliferación de los dispositivos denominados *Smartphone* (teléfonos móviles con procesadores dedicados y sistemas operativos complejos (tantos libres como propietarios), permiten el acceso a aplicaciones, tanto web como específicas para cada sistema, de forma sencilla e intuitiva para el usuario.

## 2.5. CONCLUSIONES

Como se ha podido comprobar, existen desarrolladas aplicaciones para realizar un control automatizado del alumnado una vez haya entrado en el recinto escolar, para registro de su asistencia y su seguimiento académico. Sin embargo, este seguimiento se realiza una vez que el alumno se encuentra dentro del recinto académico.

Por este motivo es necesario un control, por parte de los centros educativos, del transporte al mismo, controlando el servicio de rutas de autobuses que estos organismos proporcionan.

De este modo, los padres estarán más implicados en los procesos educativos de los alumnos, y podrán justificar de manera más precisa y automatizada, la ausencia a clase de sus hijos, así como de los posibles retrasos que se puedan ocasionar. También, los padres podrán utilizar un servicio más adaptado a sus necesidades y más automatizado, de forma que pueda motivar a otros a utilizar dichos servicios (ya que la gran mayoría pagan un gasto específico por la utilización del servicio de transporte).

Además, a la vista de las aplicaciones anteriores, se podrá automatizar y realizar un control, previo de la asistencia a clase, por el que se pueda anticipar gran parte de los trámites necesarios para su justificación.

## 3. TECNOLOGÍAS UTILIZADAS

---

A continuación se va a proceder a explicar las aplicaciones, librerías y APIs utilizadas para la elaboración de este proyecto. Se aportarán, además, las propiedades y características principales de las mismas.

### 3.1. Java Enterprise Edition

Java Enterprise Edition (JEE) [4] define un estándar para el desarrollo de aplicaciones empresariales de varios niveles. La plataforma JEE simplifica las aplicaciones empresariales basándolas en pruebas estandarizadas, componentes modulares; ofreciendo un conjunto completo de servicios a esos componentes, y manejando muchos detalles del comportamiento de la aplicación automáticamente, sin programación compleja.

Dentro de las distintas APIs y tecnologías incluidas dentro de la especificación de JEE, en este proyecto se utilizarán las siguientes:

#### 3.1.1. Servlets

Los servlets son clases Java que se ejecutan en un servidor de aplicaciones, para procesar y contestar a las peticiones de los clientes que solicitan interactuar con una aplicación. Los servlets no se encuentran limitados a un protocolo de comunicaciones específico entre clientes y servidores, pero en la práctica podemos decir que se utilizan únicamente con el protocolo HTTP.

Un servlet es un programa que se ejecuta en un servidor Web, actuando como una capa intermedia entre una petición procedente de un navegador Web y aplicaciones, bases de datos o recursos del servidor Web.

Un servlet también se define por los trabajos o tareas típicas que realiza, estas tareas se comentan a continuación, en el orden lógico en el que se realizan:

- Leer los datos enviados por el usuario: normalmente estos datos se indican a través de formularios HTML que se encuentran en páginas Web. Aunque esta información también puede provenir de otras herramientas HTTP o bien desde applets u otras aplicaciones web.

- Buscar otra información sobre la petición que se encuentra incluida en la petición HTTP: esta información incluye detalles tales como las capacidades y características del navegador, cookies, el nombre de la máquina del cliente, etc.
- Generar los resultados: este proceso puede requerir acceder a una base de datos utilizando JDBC, utilizar un componente JavaBean, o generar la respuesta de manera directa.
- Formatear los resultados en un documento: en la mayoría de los casos implica incluir los resultados en una página HTML.
- Asignar los parámetros apropiados de la respuesta HTTP: esto implica indicar al navegador el tipo de documento que se le envía (por ejemplo HTML), asignar valores a las cookies, y otras tareas.
- Enviar el documento al cliente: el documento se puede enviar en formato de texto (HTML), formato binario (imágenes GIF), o incluso en formato comprimido como un fichero ZIP.

### 3.1.2. Páginas JSP

Una página JSP es un fichero de texto con la extensión JSP, que combina etiquetas HTML con nuevas etiquetas de script que permiten ejecutar código Java. Una página JSP tiene un aspecto muy similar al de una página HTML, pero se transformarán en clases de Java, que realmente actúan como servlets, para compilarse y generar los ficheros de clase correspondientes. Esta operación se dará cuando se ejecute una página JSP por primera vez, o cuando se modifique una página JSP existente.

El servlet que resulta de la transformación de la página JSP es una combinación del código HTML contenido en la página JSP, y del contenido dinámico indicado por las etiquetas especiales pertenecientes a la especificación JSP.

Cuando un cliente realiza una petición de una página JSP, se ejecutará dicha página devolviendo como resultado código HTML que se mostrará en el navegador. Este código HTML es el resultado, a su vez, de la ejecución de la página JSP y comprende el código HTML contenido en la página y el resultado del contenido dinámico que ha sido ejecutado por el contenedor de páginas JSP. El cliente (navegador Web), nunca va a poder observar el código fuente de la página JSP, lo que recibe es el resultado de la ejecución de la misma.

### 3.2. JDBC

El conector para bases de datos de Java, Java Database Connectivity (JDBC) [5], proporciona una serie de APIs que establecen un estándar para la conectividad y acceso a bases de datos independiente, entre el lenguaje de programación Java y una amplia gama de bases de datos, ya sean bases de datos basadas en el lenguaje SQL u otras fuentes de datos tabulares, tales como hojas de cálculo o ficheros planos. La API de JDBC proporciona una llamada API de nivel de acceso basado en SQL base de datos.

Esta tecnología, permite utilizar el lenguaje de programación Java para explotar capacidades para aplicaciones que requieren acceso a los datos de la aplicación.

Básicamente el API JDBC hace posible la realización de las siguientes tareas:

- Establecer una conexión con una base de datos.
- Enviar sentencias SQL (Structured Query Language).
- Manipular los datos.
- Procesar los resultados de la ejecución de las sentencias.

### 3.3. SQL

SQL, *Structured Query Language* (Lenguaje de consulta estructurado) es el lenguaje declarativo estándar para el acceso y la gestión de sistemas de bases de datos relacionales. Está formado por un conjunto de sentencias que permite controlar y administrar una base de datos de manera directa. Dichas sentencias están clasificadas en tres categorías o lenguajes:

- Lenguaje de Definición de Datos (DDL): contiene el conjunto de sentencias que permiten la creación y mantenimiento de la base de datos, tales como la creación, modificación y eliminación de objetos (tablas, usuarios) y privilegios sobre los mismos.
- Lenguaje de Definición de Vistas (VDL): con el que se podrán crear y manipular vistas de los datos almacenados.
- Lenguaje de Manipulación de Datos (MDL): contiene las sentencias necesarias para la manipulación de los objetos creados y el acceso a los datos de los mismos. Incluye operaciones tales como la inserción, actualización o eliminación de datos de una tabla, así como la búsqueda de datos específicos (consultas).

### 3.4. MySQL

El sistema gestor de bases de datos MySQL [6], nos permite la creación y administración de bases de datos relacionales, de forma potente y rápida, ofreciendo numerosas prestaciones de rendimiento y acceso a datos que hacen que sea uno de los sistemas de bases de datos más utilizados para desarrollar aplicaciones web. Entre estas características destaca su gran facilidad de instalación, la optimización para diferentes tipos de aplicaciones, así como su carácter multiusuario (capacidad por la cual permite tener a varios usuarios accediendo a los datos almacenados) y multihilo (soporte de varias peticiones sobre datos simultáneas).

Además, dado su carácter *open source*, código abierto, esta herramienta es desarrollada y distribuida libremente, contrariamente a otros gestores de bases de datos de gran utilización, tales como Oracle o Microsoft SQL Server.

Para facilitar la labor de los desarrolladores y administradores de la base de datos, existe una herramienta gráfica, denominada MySQLWorkbench, con la que nos permite desde diseñar la base de datos, a manejar las conexiones a las distintas aplicaciones y servidores con las que se comunica, o administrar los usuarios de la misma y sus privilegios.

### 3.5. APACHE TOMCAT

Apache Tomcat [7] es un servidor web desarrollado por la Apache Software Foundation (ASF) que implementa y ejecuta las especificaciones de Java para los servlets, así como páginas JSP; es decir, convierte páginas JSP en servlets. De este modo, Tomcat se convierte en uno de los contenedores de servlets *open source* más utilizados del mercado.

Tomcat puede utilizarse independientemente como servidor Web interno o junto con otros servidores de aplicaciones.

### 3.6. JAVASCRIPT

[8]Es un lenguaje de programación ligero basado en scripts, el cual ha sido diseñado para añadir interactividad al diseño de páginas web. Es un lenguaje interpretado, es decir, los scripts son ejecutados sin necesidad de una compilación previa. Su denominación oficial es *ECMAScript* y es desarrollado y mantenido la *ECMA organization*. Fue desarrollado por Brendan Eich.

Entre las funciones principales de JavaScript está: introducir texto y etiquetas dinamizantes en las páginas web, reaccionar ante eventos web (tales como seleccionar un elemento, hacer clic en un botón o realizar un gesto con el ratón), detectar el navegador del usuario, acceso a etiquetas y elementos HTML presentes en las páginas, así como su modificación o eliminación, etc.

Además, uno de los principales usos de este lenguaje es la validación de datos antes de ser enviados al servidor, ya que el código JavaScript se ejecuta inmediatamente mientras la página se carga en el navegador del usuario, sin necesidad de que se produzca una interacción entre las páginas web y el servidor.

Estos scripts pueden incluirse de dos formas, o bien dentro de la propia página web, en las etiquetas '`<head>`' o '`<body>`', o como referencia a un archivo externo con extensión '.js', el cual será referenciado dentro de la página web. De esta forma se pueden reutilizar las funciones implementadas en los archivos en distintas páginas web.

### 3.7. AJAX

El término AJAX [9] es un acrónimo de *Asynchronous JavaScript + XML*. No es una tecnología en sí misma, sino que está formado por varias tecnologías, las cuales son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

En las aplicaciones web, todas las acciones que realiza el usuario en las páginas desencadenan llamadas al servidor para el tratamiento de las mismas. El servidor procesa la petición del usuario, realiza las operaciones necesarias y devuelve los resultados al usuario, en forma de una nueva página web.

Esta técnica para crear aplicaciones web implica realizar peticiones continuas al servidor, en las que el usuario debe esperar a que se procese la petición y se cargue una nueva página con la nueva información solicitada. En el caso de que se deban realizar continuamente

peticiones al servidor, el usuario recibe la impresión de una carga lenta y de realizar muchas operaciones para llevar a cabo una funcionalidad.

AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página. Esto es debido a que el intercambio de la información entre el usuario y el servidor se produce en segundo plano, sin que el usuario sea consciente de que se realizan dichas peticiones. De esta forma, se elimina la recarga constante de las páginas. Esto se consigue mediante la creación de un elemento intermedio entre el usuario y el servidor que mejora la respuesta de la aplicación.

El funcionamiento de las aplicaciones que utilizan esta tecnología en la siguiente: las peticiones HTTP dirigidas al servidor se sustituyen por peticiones JavaScript que se realizan en dicha capa intermedia. Si las peticiones más simples no requieren intervención del servidor, la respuesta es inmediata. Si la interacción requiere una respuesta del servidor, la petición se realiza de forma asíncrona mediante AJAX. En este caso, la interacción del usuario tampoco se ve interrumpida por recargas de página o largas esperas por la respuesta del servidor.

Gracias a esta técnica, AJAX puede sustituir completamente a otras técnicas de programación web como Flash. También pueden llegar a sustituir a las aplicaciones de escritorio web.

### 3.8. JSON

JSON [9] (*JavaScript Object Notation*) es un formato sencillo para el intercambio de información. El formato JSON permite representar estructuras de datos y objetos en forma de texto. La especificación completa de JSON se puede consultar en RFC 4627 (<http://tools.ietf.org/html/rfc4627>).

JSON se ha convertido en una alternativa al formato XML para el intercambio de datos, ya que es más fácil de leer y escribir, además de ser mucho más conciso.

En la notación de los objetos, el contenido del mismo se encierra entre llaves ('{'y '}'). Los elementos de cada uno de los valores de los objetos se separan mediante coma (','). Por último, las claves y el valor de la misma se separan con dos puntos (':').

Un ejemplo de objeto representado en esta notación es:

```
var objeto = {clave1: valor1, clave2: valor2, clave3:
valor3,..., claveN: valorN};
```

### 3.9. GOOGLE MAPS

Google Maps [10], englobado dentro de las aplicaciones y servicios web que ofrece Google (desde el 6 de octubre del 2005, Google Maps es parte de Google Local), es un servidor de mapas, los cuales pueden incluirse dentro de aplicaciones web. Este servicio no sólo ofrece imágenes de mapas, sino que permite el desplazamiento por el mapa a gusto del usuario, así como ofrecer características de buscar emplazamientos, marcar varias localizaciones a la vez, cambiar el tipo de mapa, incluir información acerca de recursos o servicios cercanos a la localización, así como la inclusión de mensajes o indicaciones en cada emplazamiento.

Google ofrece una API gratuita (para uso no comercial). Su última versión es la 3.0, en la cual no se requieren de unas claves para poder acceder a la aplicación (como pasaba en las anteriores versiones). Esta API permite a los desarrolladores insertar las funciones más completas y útiles de Google Maps en sus aplicaciones web, así como superponer sus propios datos sobre ellas.

Existen versiones la API que funcionan sobre JavaScript, Flash, Google Earth, incluso sin el uso de JavaScript, pero perdiendo la mayor parte de sus funcionalidades, ya que devolvería una imagen estática de la localización solicitada.

En la realización de este proyecto se ha elegido la Maps JavaScript API en su versión 3 (ya que la versión 2 se ha sido descartada oficialmente por Google), la cual nos permite insertar un mapa de Google en las páginas web, así como manipular el mapa y añadir contenido a través de diferentes servicios.

Además, para la parte móvil de proyecto, se han elaborado mapas, mediante la versión estática de la API de Google Maps, la cual no hace uso de JavaScript ni AJAX para su elaboración. De este modo, se abre la compatibilidad de los mapas a aquellos dispositivos móviles que no soportan el uso de JavaScript o que sus recursos son limitados, abriendo el uso de la aplicación a un gran abanico de usuarios.

### 3.10. JAVA MAIL

La API Java Mail [11] nos ofrece un conjunto de herramientas, independientes de cualquier protocolo y plataforma, que nos permite construir aplicaciones que utilicen servicios de intercambio de mensajes vía email. Gracias a estas herramientas, podremos establecer una comunicación y sesión con un servidor SMTP y comunicarnos con el mismo para enviar mensajes, o consultar si han llegado nuevos emails a una determinada cuenta. Soporta además otros protocolos como TLS, SSL, MIME, POP3 o IMAP.

Esta API puede ser descargada desde la página de descargas de Java, y debe ser incluida en los proyectos que hagan uso de estas características.

### 3.11. ITEXT

iText [16] es una API que nos permite la creación, manipulación y manejo de documentos PDF, así como de los datos que en ellos se encuentra; centrándose en la automatización de las tareas. Debido a esto, no es una herramienta de usuario final, sino que el desarrollador debe incluir el código de la biblioteca de iText en las aplicaciones para poder automatizar el proceso de creación de PDF y su manipulación.

Las características a destacar de esta API son las siguientes:

- Permite generar documentos e informes basados en datos de un archivo XML o de una base de datos.
- Posibilita la creación mapas, libros y la explotación de numerosas funcionalidades disponibles en formato PDF.
- Agregar marcadores en los documentos (*bookmarks*), números de página, marcas de agua, y otras características para documentos PDF existentes.
- Dividir o concatenar las páginas de los archivos PDF.
- Servir documentos PDF generados dinámicamente o manipulados en un navegador web.

Por lo general, iText se utiliza en proyectos que tienen uno de los siguientes requisitos:

- El contenido no está disponible con antelación: es generado en base a la entrada del usuario o información en tiempo real procedente de una base de datos.
- Los archivos PDF no pueden ser generados de forma manual debido a la cantidad masiva de contenidos (un gran número de páginas o documentos).
- Documentos que deben ser creados en un proceso por lotes.
- El contenido debe ser personalizado, por ejemplo, el nombre del usuario final tiene que ser grabado en un número de páginas.

La mayor parte de estos requisitos se encuentran en las aplicaciones web, donde el contenido necesita ser generado de forma dinámica en un explorador web. Normalmente, esta información está representada en formato HTML, pero en muchos casos, es preferible el formato PDF para una mejor calidad de impresión, para la presentación idéntica a una variedad de plataformas, por razones de seguridad, o para reducir el tamaño del archivo.

Por estos motivos, se ha decidido utilizar esta herramienta (en su versión gratuita) para la generación de justificantes de faltas de asistencia en formato PDF. De esta manera, se recogerán los datos de dicha falta de la base de datos y será presentado al usuario en este formato.

### 3.12. JAVA CSV LIBRARY

Java CSV [17] es una librería pequeña, de carga rápida, implementada en código abierto para Java que nos permite leer y escribir archivos CSV y archivos de texto plano delimitado (varios datos separados entre sí por un carácter o secuencia de símbolos que actúan como delimitador entre ellos). Soporta una gran variedad de archivos, tales como CSV, texto, formato EXCEL, etc.

Se ha utilizado esta librería en la aplicación para realizar la inserción de varios datos de forma automática en la aplicación, de forma que no sea necesario insertar cada dato de uno en uno. Se ha de destacar, además, su sencillez y eficiencia al manejar los ficheros.

### 3.13. LIBRERÍAS COMMONS FILEUPLOAD Y COMMONS IO

Estas librerías nos ofrecen una serie de herramientas y clases en las que podemos realizar operaciones de entrada y salida de datos de gran tamaño, tales como la subida de ficheros a las aplicaciones.

El paquete Commons FileUpload[18] permite agregar una capacidad de carga de archivos a servlets y aplicaciones web de forma sencilla, robusta y con un alto rendimiento.

FileUpload analiza las peticiones HTTP que se ajusten a la norma RFC 1867[20], es decir, si una petición HTTP se envía mediante el método POST, y con un tipo de contenido "*multipart / form-data*", entonces FileUpload puede analizar esa petición, y que los resultados estén disponibles de una forma sencilla de utilizar por las aplicaciones.

Para ayudar a sus funciones, se dispone del paquete Commons IO [19]. Se trata de una biblioteca de utilidades para ayudar a desarrollar la funcionalidad de entrada/salida de datos en las aplicaciones web. Para ello, aporta clases de entrada y salida, filtros, utilidades, comparadores y monitores de archivos.

Estas dos librerías son utilizadas para poder realizar subidas de ficheros a la aplicación web, en concreto archivos CSV comentados en el epígrafe anterior. De esta forma, los usuarios podrán subir archivos a la aplicación para poder ser procesados automáticamente por la misma.

## 4. DESCRIPCIÓN DEL SISTEMA

---

En este punto del documento, se describirán las funcionalidades y objetivos que debe cumplir el sistema, y se establecerán los requisitos que éste debe satisfacer, indicando las características principales de cada uno de ellos.

El objetivo de este proyecto es la construcción de un sistema, a través de una aplicación web, mediante la cual los padres y alumnos de los centros escolares, y los responsables del centro, puedan intercambiar información acerca del sistema de transporte hacia el centro, y del control de la asistencia a clase según el uso de dicho sistema de transporte.

### 4.1. CARACTERÍSTICAS GENERALES

Para ello, distinguimos tres tipos de usuarios según las necesidades de cada uno de ellos. Los usuarios son los siguientes:

- **Padres:** los cuales accederán a la aplicación para consultar las rutas e información de los alumnos.
- **Responsables del centro:** profesores o personal del centro que tramita el control de faltas de asistencia y controla el servicio de transportes contratado.
- **Responsables de las rutas:** personal docente encargados del control de los alumnos en las rutas.

Cada uno de estos usuarios podrá acceder a la aplicación a través de un formulario de entrada, mediante el cual serán redirigidos a sus propios contenidos. Dicho formulario validará si son usuarios del sistema o si sus datos han sido introducidos correctamente.

Para almacenar dichos datos, además de toda la información que maneje la aplicación, será necesaria la utilización de un sistema gestor de bases de datos, que contendrá toda la información que necesite la aplicación.

A continuación se procede a describir las distintas funcionalidades que ofrecerá la aplicación a través de cada uno de los usuarios que la utilizarán.

## 4.2. FUNCIONES PRINCIPALES PARA PADRES

Una vez accedida a la aplicación, los padres de los alumnos podrán acceder a la información y operaciones a través de un menú desplegable en el que seleccionarán la opción deseada.

Referente al tema de las rutas y las paradas, los padres podrán elegir la parada en la que se recogerá a los alumnos para un determinado curso académico. Conjuntamente, tendrán acceso a todas las rutas y paradas disponibles para tener toda la información para elegir su parada.

Además, dispondrán de una opción para notificar un retraso para acceder a la parada a entregar al alumno. Dicho retraso será únicamente de 5 minutos de horario previsto, marchándose la ruta en caso de que el tiempo haya pasado y el alumno no haya acudido a la parada

Otra opción a elegir consiste en seleccionar, excepcionalmente, otra parada en la que se recogerá al alumno. De esta forma podrán elegir de entre las paradas de su ruta asignada, una distinta en la que se recogerá al alumno el día determinado.

Dentro de las operaciones a realizar en relación a la asistencia a clase, se encuentra la opción de notificar falta de asistencia, de forma que pueda notificar al centro que el alumno en cuestión no podrá asistir a las clases un día específico. Para ello, podrá elegir tanto el día de la falta, como qué alumno dejará de asistir (en el caso de que tenga a varios matriculados en el centro).

Además, podrá confirmar las faltas de asistencia de los alumnos, de forma que, una vez producida la falta, podrá recoger e imprimir a través de la aplicación un justificante de asistencia del mismo por el periodo lectivo que notificó.

Es importante recalcar que, para realizar la programación de una falta, deberá tener lugar antes de la fecha en la que se va a producir; ya que en caso contrario, dicha falta se almacenará en la aplicación pero será imposible su procesamiento, ya que el día de paso de la ruta ya se produjo y, por lo tanto, no quedará registrada dicha falta.

Además, se podrá consultar un historial de todas las faltas de asistencia de un determinado alumno, y visualizar todos sus detalles, tales como día, si se ha justificado la falta o no, etc.

Además, los padres podrán tener algunas de las características de la aplicación en una versión para teléfonos móviles, también basada en web. Desde estos dispositivos se podrán acceder a las funcionalidades más urgentes de la aplicación, tales como consultar los datos de la parada de un alumno, o establecer un retaso para ese mismo día, antes de que la ruta de transporte llegue a la parada del alumno.

### **4.3. FUNCIONES PRINCIPALES PARA EL PERSONAL DEL CENTRO**

El personal docente poseerá herramientas para controlar tanto las rutas, como a los responsables de las mismas y los alumnos que las utilizan.

Referente a las rutas, podrán eliminar o modificar cualquier ruta o parada existente, así como añadir nuevas rutas y paradas dependiendo de la demanda y la utilidad de las mismas.

Con respecto a los responsables de las rutas, podrán dar de alta o eliminar a este personal docente, para llevar un control de los mismos. Además, podrán asignar una ruta a cada responsable y, de este modo, los alumnos asociados a la misma.

Cabe destacar que un responsable de ruta sólo puede estar asignado a una única ruta, es decir, no se podrá asignar un responsable a dos rutas diferentes.

También podrán recoger, una vez hayan llegado todas las rutas al centro escolar, las listas de los alumnos que no han ido en la ruta y, por lo tanto, han faltado a clase; así como aquellos que hayan notificado previamente, a través de la aplicación, dicha falta de asistencia. De este modo, podrán generar listados de asistencia para los profesores, de forma que sabrán qué alumnos no van a asistir y comprobarán si estos alumnos han asistido a clase mediante otras vías (transporte privado).

Además, una vez que haya sido entregado por el alumno el justificante de la falta, podrá introducir este hecho en la aplicación, de forma que se establezca un control de los alumnos que han justificado sus faltas y los que no. Para ello podrá listar los alumnos que poseen las faltas de asistencia sin justificar, para poder marcar que se han justificado y la actividad o causa por la cual han faltado a clase, teniendo un control informatizado de los alumnos.

También, para agregar nuevos alumnos a la aplicación, se ofrece la posibilidad de subir un fichero en formato CSV mediante el cual se podrá automatizar la carga de los datos almacenados en la base de datos.

Como en el caso de los padres, el personal docente del centro podrá acceder a la gran mayoría de estas funcionalidades a través de una aplicación web adaptada para la visualización y funcionamiento de dispositivos móviles. A través de ésta, podrán gestionar las rutas, a los responsables, así como a los alumnos y padres de los mismos.

#### **4.4. FUNCIONES PRINCIPALES PARA LOS RESPONSABLES DE RUTA**

Los responsables de las rutas, accederán a la lista de los alumnos asignados en dicha ruta para el día en cuestión, es decir, de aquéllos que no han solicitado a través de la aplicación la falta de asistencia.

En dicha lista, los responsables podrán marcar a los alumnos que se han subido a la ruta, llevando un control de ellos. Además, si algún alumno ha solicitado un poco más de tiempo para llegar a la misma, podrá visualizarlo para tener constancia del mismo y poder indicar al conductor de la misma que espere un poco más.

Una vez llegado a centro, podrán finalizar este control de alumnos, de forma que se graben los datos en la aplicación y la información asociada a los datos de ruta recogidos.

## 5. DISEÑO E IMPLEMENTACIÓN DE LA BASE DE DATOS

---

En este apartado se diseñará y se presentarán los scripts de implementación de la base de datos que se utilizará en la aplicación desarrollada. Para ello, se diseñará previamente la estructura de la base de datos, presentado el modelo entidad-relación de la misma, así como toda la explicación de las tablas que conformarán la base de datos. También, se definirán las consultas que nuestra aplicación realizará sobre los datos almacenados, incluyendo las principales sentencias de inserción y modificación de los mismos.

Además, se presentarán los scripts necesarios para su creación, implementados en el lenguaje SQL, así de cómo cualquier otra sentencia u operación necesaria para la puesta en funcionamiento de la base de datos.

### 5.1. DISEÑO DE LA BASE DE DATOS

#### 5.1.1. Diagrama Entidad-Relación

En primer lugar, y una vez comprendidas las especificaciones y requisitos que requiere la aplicación, procedemos a identificar las entidades, de las que almacenaremos y manejaremos información, así como las relaciones que se establecen en cada una de ellas.

En primer lugar en esta aplicación distinguimos claramente tres tipos de usuarios (Padres, personal docente y responsable de ruta) que, si bien se almacena la misma información de cada uno de ellos, la forma de acceder a los datos no es la misma, ya que cada tipo de usuario tiene acceso a un determinado tipo de información, así como a unas determinadas operaciones a realizar con los datos. De este modo, encontraremos cuatro entidades, la genérica Usuario que almacena los datos, y una entidad por cada tipo de usuario, de forma que se puedan controlar las relaciones con el resto de datos.

Encontramos además una entidad en la que se almacenan todos los datos personales de los alumnos. Como un padre puede tener varios alumnos a su cargo, pero un alumno sólo puede tener a un usuario en la aplicación, existe una relación de uno a varios entre ellos. Estos alumnos utilizan el transporte escolar en una de sus paradas, con lo que existe otra entidad, Parada, donde se almacena la información relacionada con la misma, dando lugar a otra relación uno a varios entre ellos.

Cabe destacar que ésta no es la única relación que existe entre estas dos entidades, ya que también se desea almacenar cuando un alumno cambia de parada excepcionalmente o solicita un retraso para acceder a la misma. Estos hechos hacen que sea necesario el almacenamiento de la fecha en que se realizan dichos trámites, ya que pueden ser realizados en múltiples ocasiones. Por ello, son necesarias dos relaciones más, de multiplicidad varios a varios.

Otro elemento importante son las rutas de autobuses, representado por la entidad Ruta, que contendrá toda la información relevante de ésta. Esta entidad está relacionada con Parada (ya que una ruta está formada por varias paradas) y con Personal Docente, que es el que se encarga de añadirlas, modificarlas o borrarlas. Estas relaciones son también de uno a varios.

Se desean almacenar los datos relacionados con las faltas de asistencia. Para ello, existirá una entidad Faltas que recogerá la información relativa a las mismas. Esta entidad está relacionada con los responsables de ruta, que introducen los registros de alumnos que no acuden a clase, y con el personal docente, que tramita las mismas y controlan su estado. Estas relaciones tienen multiplicidad uno a varios. También, se relaciona con la entidad Alumnos, ya que un alumno falta a clase en una fecha y con un motivo determinado, con una multiplicidad de la relación de uno a varios.

Por último, el personal docente decide los profesores o empleados que serán responsables de las rutas de transporte. Debido a ello, existe una relación entre estos dos tipos de usuarios, con una multiplicidad de uno a varios.

Con esta información procedemos a realizar el diagrama entidad-relación, junto con los principales atributos de cada entidad y relación, que serán explicados ampliamente posteriormente.

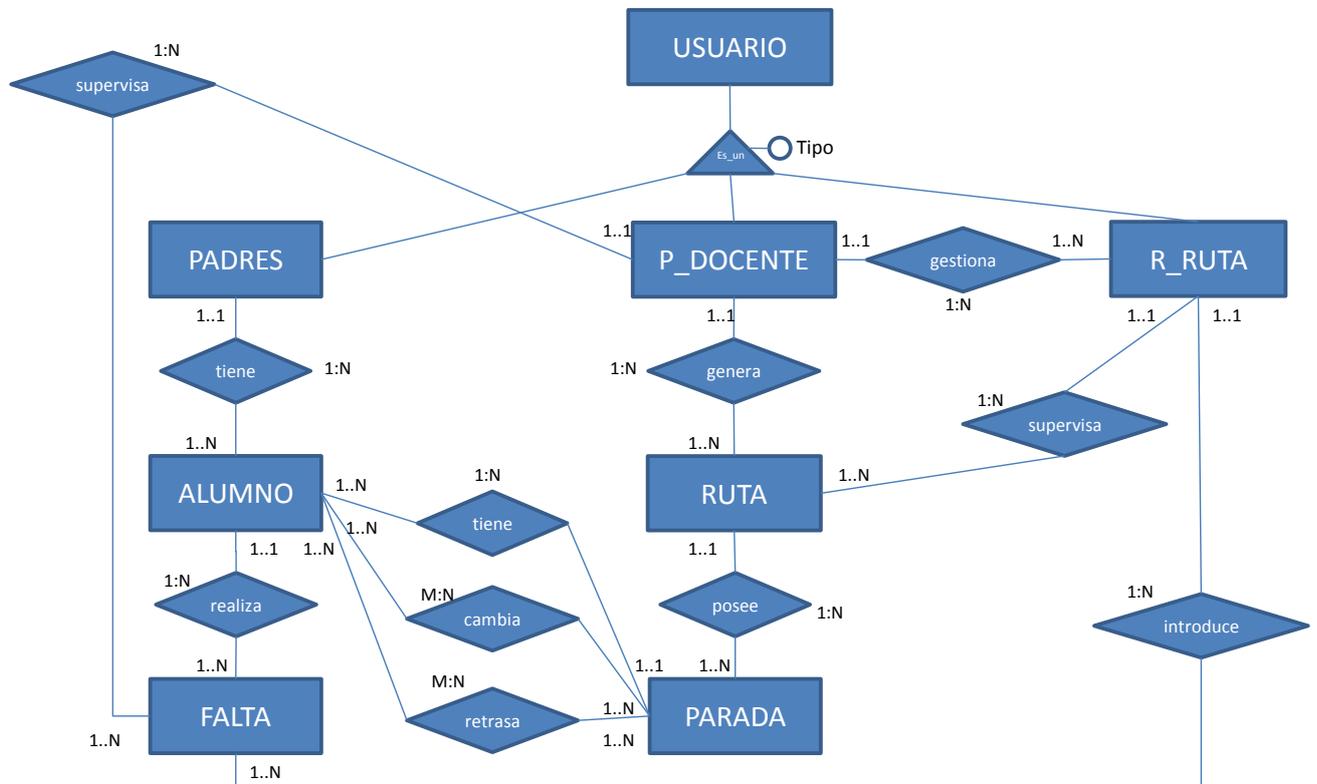


Ilustración 1. Diagrama Entidad-Relación.

### 5.1.2. Diagrama Relacional

Una vez elaborado este diagrama, el siguiente paso en el diseño de la base de datos es elaborar el modelo relacional de la misma. Para ello, se transformarán las entidades anteriores y sus relaciones en tablas y atributos de la base de datos. Para ello, se seguirán las siguientes reglas de transformación:

- Cada una de las entidades se transformará en una tabla (también denominada relación), en la que el atributo principal de la misma constituirá la clave principal de la tabla, que servirá para identificar unívocamente a cada elemento almacenado (tupla).
- Por cada relación uno a varios, se pasará la clave principal de la entidad con mayor multiplicidad a la de menor, convirtiéndose en una clave ajena, que identifica con qué tupla específica se encuentra relacionado.

- Por cada relación varios a varios, se creará una tabla intermedia, en la que la clave principal estará formada, como mínimo, por las claves principales de las entidades que participan en la relación. Además, se incorporarán los atributos propios de la relación.
- Cada relación padre-hijo, se transformará en una tabla para cada una de las entidades, en la que la clave de la entidad padre se propagará a las entidades hijo.

Aplicando estas reglas de transformación al modelo anterior, obtendremos el siguiente diagrama relacional. Este diagrama está realizado con la herramienta MySQLWorkbench, y se trata de una ampliación del modelo anterior, con la información de las tablas y todos sus atributos, en donde las claves principales aparecen representadas mediante llaves amarillas, y las claves ajenas mediante rombos morados.

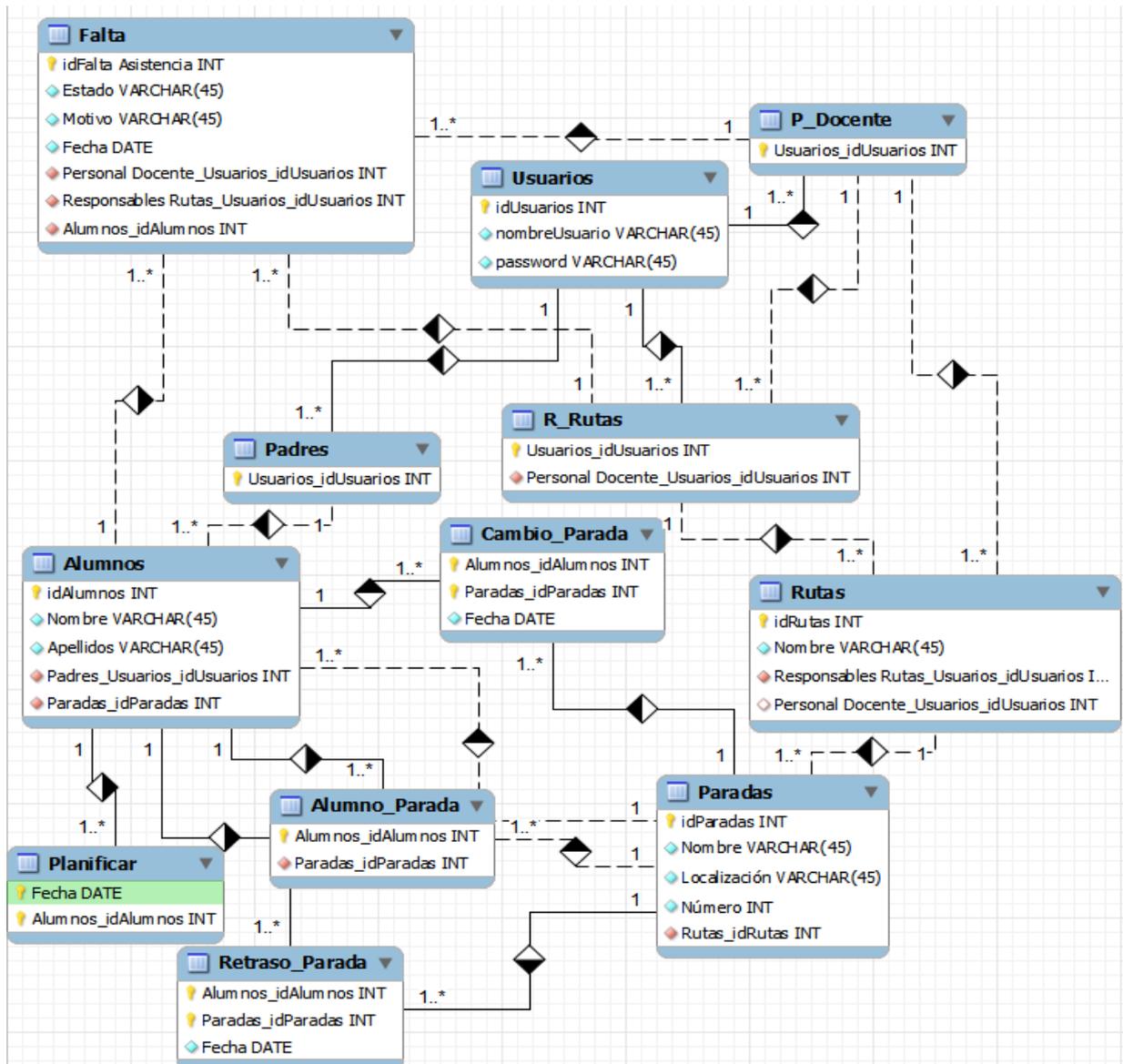


Ilustración 2. Diagrama relacional

Por último, para completar la información del diagrama, es importante identificar y establecer las opciones de modificación y borrado de las tuplas que contienen atributos ajenos. Para ello, de entre las múltiples opciones existentes, vamos a diferenciar en esta aplicación tres de ellas:

- En cascada: si una tupla se elimina/modifica, también lo hacen aquellas que estén referenciadas, es decir, que tengan como clave ajena la principal de ésta.
- Restrictivo: No se puede modificar/eliminar una tupla, mientras sigan existiendo tuplas referenciadas en otras tablas.

- No Action: al modificar/eliminar una tupla, no se realiza ninguna acción en las tuplas que estén referenciadas, es decir, es equivalente al borrado restrictivo, ya que no permite realizar ninguna acción.

A continuación, se especifican las opciones de borrado y modificación para cada una de las relaciones existentes entre tablas:

- Relación Padre- Alumno: Modificación en Cascada, Borrado en Cascada. Al eliminar la tupla de un padre, se elimina la información de los hijos referente a este. Esto se da en los casos de que los alumnos se den de baja del centro escolar y sea tenga que eliminarse la información relacionada con sus hijos.
- Relación P\_Docente – R\_Rutas: Modificación en Cascada, Borrado Restringido. No se podrán eliminar al personal docente que tenga a su cargo el control de algún responsable de ruta.
- Relación Rutas – Paradas: Modificación y borrado en Cascada. Si se decide eliminar una ruta de transporte, por los motivos que sea, será eliminada también cualquier información acerca de las paradas en la base de datos.
- Relación P\_Docente – Falta: Modificación en Cascada, Borrado Restringido. No se podrá eliminar al personal que haya controlado una falta de asistencia, mientras tenga referenciadas dichas faltas.
- Relación Alumno – Alumno\_Parada: Modificación en Cascada, Borrado Restringido. No se podrá eliminar una parada mientras haya alumnos que siga utilizando dicha parada.
- Relación Alumno – Retraso\_Parada: Modificación y Borrado en Cascada. Si se elimina un alumno, se eliminan todos sus registros de retraso.
- Relación Alumno – Cambio\_parada: Modificación y Borrado en Cascada. Si se elimina un alumno, se eliminan todos sus registros de cambios de parada excepcionales.
- Relación Alumno –Falta: Modificación y Borrado en Cascada. Si se elimina un alumno, se eliminan todas las faltas de asistencia que se hayan producido.
- Relación Paradas – Retraso\_Parada: Modificación en Cascada, Borrado Restringido. No se podrá eliminar una parada mientras haya alumnos que tengan programados retrasos en la misma.
- Relación Paradas – Cambio\_Parada: Modificación en Cascada, Borrado Restringido. No se podrá eliminar una parada mientras haya alumnos que hayan establecido dicha parada como cambio provisional.

- Relación Alumno\_Parada – Cambio\_Parada: Modificación en Cascada, Borrado Restringido. No se podrá eliminar una parada mientras haya alumnos que hayan establecido dicha parada como parada del curso.
- Relación Alumno – Planificar: Modificación y Borrado en Cascada. Si se elimina un alumno, se eliminan todas sus faltas planificadas.

### 5.1.3. Descripción De Los Atributos De Las Tablas

A continuación, procedemos a describir cada uno de los atributos de cada una de las tablas que componen la base de datos. Además, se detallará el tipo de dato que se almacenará en el mismo, así como si se trata de un atributo clave o no.

#### Tabla Usuarios

- idUsuarios. Clave principal que identifica a un usuario unívocamente. Su tipo de dato es entero.
- nombreUsuario. Nombre por el que se identifica a los usuarios de la aplicación, y será al que se le pida al usuario para acceder a la misma. Es de tipo alfanumérico (varchar), con una longitud máxima de 45 caracteres.
- password. Contraseña de acceso a la aplicación, de tipo alfanumérica, y con una extensión máxima de 128 caracteres y mínima de 8 (controlado por otra tecnología diferente a la base de datos).
- Email: dirección de correo electrónico del usuario. Es de tipo alfanumérico (varchar), con una longitud máxima de 40 caracteres

#### Tabla Padres

- Usuarios\_idUsuario. Clave principal que identifica a un usuario unívocamente. Su tipo de dato es entero.

#### Tabla Personal Docente

- Usuarios\_idUsuario. Clave principal que identifica a un usuario unívocamente. Su tipo de dato es entero.

### Tabla Responsable Ruta

- Usuarios\_idUsuario. Clave principal que identifica a un usuario unívocamente. Su tipo de dato es entero.
- Personal Docente. Clave ajena, que identifica al personal docente que le dio de alta en la aplicación. Es de tipo entero.

### Tabla Alumnos

- idAlumnos. Clave principal que identifica a un alumno unívocamente. Su tipo de dato es entero.
- Nombre. Nombre del alumno almacenado. Su tipo de dato es alfanumérico, con una extensión de 45 caracteres.
- Apellidos. Apellidos del alumno almacenado, de tipo alfanumérico y de 45 caracteres máximo.
- Padres\_Usuarios\_idUsuarios. Clave ajena que identifica cuál es su padre o madre. Su tipo de datos es entero.
- Paradas\_idParadas. Clave ajena que asocia al alumno a una determinada parada de una ruta. Su tipo de dato es entero.

### Tabla Rutas

- idRutas. Clave principal que identifica a una ruta unívocamente. Su tipo de dato es entero.
- Nombre. Nombre que recibe la ruta, más familiar y característico que el identificador. Su tipo de datos es alfanumérico, con una longitud máxima de 45 caracteres.
- R\_Rutas\_Usuarios\_idUsuarios. Clave ajena que identifica al responsable de dicha ruta, quién controlará el acceso a la misma a los alumnos. Si tipo es entero.
- P\_Docente\_Usuarios\_idUsuarios. Clave ajena que identifica al personal docente que ha creado dicha ruta, y que sirve para el control de su creación. Es de tipo entero.

### Tabla Paradas

- idParadas. Clave principal que identifica a una parada unívocamente. Su tipo de dato es entero.
- Nombre. Nombre que se le proporciona a la parada que la identifique de una forma más descriptiva que el anterior. Su tipo es alfanumérico, con una longitud máxima de 45 caracteres.
- Localización: Dirección en la cual se encuentra la parada. Su tipo es alfanumérico, con una longitud máxima de 80 caracteres.
- Numero: Numero de parada respecto al total de la ruta (Parada 3 de 10 por ejemplo). Su tipo de datos es entero.
- Rutas\_idRutas: Clave ajena que asocia dicha parada a una de las rutas definidas. Su tipo de dato es entero.

### Tabla Falta

- idFalta Asistencia. Clave principal que identifica a una parada unívocamente. Su tipo de dato es entero.
- Estado. Posible estado de la falta. Su tipo de datos es alfanumérico, y sujeto a uno de los siguientes casos: Prevista, No justificada y Justificada.
- Motivo. Explicación proporcionada por los padres o el alumno de la causa de la falta de asistencia. Su tipo es alfanumérico, con una longitud máxima de 45 caracteres.
- Fecha. Fecha en la que el alumno faltó a clase. Su tipo de dato es fecha (date).
- P\_Docente\_Usuarios\_idUsuario. Clave ajena que identifica al personal que se encarga de tramitar la falta de asistencia. De tipo entero
- R\_Rutas\_Usuarios\_idUsuarios. Clave ajena que identifica al responsable que anotó la falta. Su tipo de datos es entero.
- Alumnos\_idAlumnos. Clave ajena que identifica el alumno que ha causado falta. De tipo entero.

**Tabla Planificar**

- Alumnos\_idAlumnos. Componente de la clave principal, que identifica al alumno que planifica una falta de asistencia. De tipo entero.
- Fecha. Componente de la clave principal que detalla la fecha en la que se realizará la falta. De tipo date.

**Tabla Cambio\_Parada**

- Alumnos\_idAlumno. Componente de la clave principal, que identifica al alumno que pide el cambio de parada. De tipo entero.
- Paradas\_idParada. Componente de la clave principal, que identifica la nueva parada a la que acudirá el alumno. De tipo entero.
- Fecha. Componente de la clave principal que detalla la fecha en la que se realizará el cambio. De tipo date.

**Tabla Retraso\_Parada**

- Alumnos\_idAlumno. Componente de la clave principal, que identifica al alumno que pide el retraso en una parada. De tipo entero.
- Paradas\_idParada. Componente de la clave principal, que identifica la parada en la que esperará un tiempo a que llegue el alumno. De tipo entero.
- Fecha. Componente de la clave principal que detalla la fecha en la que se ha pedido la prórroga de tiempo. De tipo date.

**Tabla Alumno\_Parada**

- Alumnos\_idAlumno. Componente de la clave principal, que identifica al alumno que establece una parada por defecto para ser llevado al centro escolar. De tipo entero.
- Paradas\_idParada. Componente de la clave principal, que identifica la parada a la que acudirá el alumno. De tipo entero.

## 5.2. CREACIÓN DE USUARIOS Y PERMISOS A LA BASE DE DATOS

Una vez diseñada la base de datos se procede a su implementación para ser utilizada por la aplicación web. Para ello se crea un script en el cual se especifica la creación de las tablas y relaciones anteriormente expuestas, englobadas en la base de datos denominada CateBus. Dicha base de datos es creada por el administrador del sistema gestor de bases de datos.

Para que la base de datos pueda ser utilizada por la aplicación, y los responsables del centro escolar que así lo necesiten, se ha creado un usuario específico para su uso y explotación. Dicho usuario tiene las siguientes características:

- Nombre de usuario: CateBus
- Contraseña: root

La contraseña utilizada puede ser cambiada según las necesidades de los responsables del centro.

A continuación, se deben crear los permisos necesarios para su correcta utilización. Para ello, se ha otorgado a este usuario únicamente los permisos de inserción, modificación, borrado y consulta de datos, impidiendo que pueda eliminar tablas o relaciones esenciales del sistema, así como añadir nuevas entidades que puedan crear conflictos tanto con los datos ya almacenados, como con la interacción con la aplicación web.

## 5.3. DISEÑO DE LAS PRINCIPALES CONSULTAS SOBRE LA BASE DE DATOS

Para que Proyecto CATEBus pueda obtener la información necesaria para presentar y realizar operaciones con los datos almacenados, necesita realizar una serie de consultas a la base de datos para obtener dicha información. Las consultas más importantes, obviando aquellas de las que únicamente se obtiene información de una tabla, se muestran a continuación:

### **Listado de alumnos que utilizan una misma ruta**

```
select Alumnos.Nombre,Alumnos.Apellidos,Paradas.Nombre from
CateBus.Alumnos,CateBus.Paradas where Paradas.Rutas_idRutas =
IDRUTA and Paradas.idParadas = Alumnos.Paradas_idParadas;
```

**Listado de alumnos que han faltado a clase en una determinada fecha**

```
select Alumnos.Nombre,Alumnos.Apellidos,Falta.estado from
CateBus.Alumnos,CateBus.Falta where Falta.Fecha = FECHA and
Falta.Alumnos_idAlumnos = Alumnos.idAlumnos;
```

**Listado de alumnos que hayan programado una falta de asistencia a una ruta en una determinada fecha**

```
select Planificar.Alumnos_idAlumnos from
CateBus.Planificar, CateBus.Alumno_Parada, CateBus.Paradas where
Planificar.Fecha = FECHA and Planificar.Alumnos_idAlumnos =
Alumno_Parada.Alumnos_idAlumnos and
Alumno_Parada.Paradas_idParadas = Paradas.idParadas and
Paradas.Rutas_idRutas = RUTA;
```

**Listado de alumnos de que utilizan una parada de una ruta, determinado por su número de parada**

```
select Alumnos.idAlumnos,Alumnos.Nombre,Alumnos.Apellidos
from CateBus.Alumnos, CateBus.Alumno_Parada, CateBus.Paradas
where Paradas.Rutas_idRutas = RUTA and Paradas.Numero = NUMERO
and Alumno_Parada.Paradas_idParadas = Paradas.idParadas and
Alumno_Parada.Alumnos_idAlumnos = Alumnos.idAlumnos;
```

**Listado de alumnos que hayan solicitado de forma ocasional una parada de una ruta, identificada por su número de parada, en una fecha determinada**

```
select Alumnos.idAlumnos,Alumnos.Nombre,Alumnos.Apellidos
from CateBus.Alumnos,CateBus.Cambio_Parada,CateBus.Paradas where
Cambio_Parada.Alumnos_idAlumnos = Alumnos.idAlumnos and
Cambio_Parada.Fecha = FECHA and Cambio_Parada.Paradas_idParadas
= Paradas.idParadas and Paradas.Numero= NUMERO;
```

## 6. COMPONENTES DE LA APLICACIÓN

---

### 6.1. ARQUITECTURA DEL SISTEMA

La arquitectura elegida para el desarrollo de la aplicación es la comúnmente conocida como modelo-vista-controlador (MVC). También es conocida como la arquitectura de tres capas, y es una de las más utilizadas en la actualidad para la realización de proyectos software de la índole de la que se trata en este proyecto, es decir, aplicaciones web.

Esta arquitectura tiene como principal objetivo aislar los datos y los procedimientos o mecanismos principales para tratar esos datos, de la vista, así como modelar la transición entre los distintos estados del modelo.

- **Modelo:** es la representación de la información que maneja la aplicación. El Modelo en sí son los datos puros que puestos en contexto del sistema que proveen de información al usuario o a la aplicación en sí misma.
- **Vista:** es la representación del modelo en forma gráfica, disponible para la interacción con el usuario. En el caso de una aplicación Web, la Vista es una página HTM o JSP, con contenido dinámico sobre el cual el usuario puede realizar operaciones.
- **Controlador:** es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria, y modificando el Modelo en caso de ser necesario.

En la siguiente imagen se puede observar el ciclo normal de ejecución en el sistema.



Ilustración 3: Ciclo del MVC

El primer paso en el ciclo de vida, comienza cuando el usuario hace una solicitud al Controlador con información sobre lo que éste desea realizar. En ese momento, el Controlador decide a quién debe delegar la tarea, y es en este punto donde el Modelo empieza su trabajo. En esta etapa, el Modelo se encarga de realizar operaciones sobre la información que maneja para cumplir con lo que le solicita el Controlador.

Una vez que termina su labor, le devuelve al Controlador la información resultante de sus operaciones, el cual a su vez, dirige a la Vista. Ésta se encarga de transformar los datos en información visualmente entendible por parte del usuario. Finalmente, la representación gráfica es transmitida de regreso al Controlador y éste se encarga de transmitírsela al usuario.

Realizar esta descomposición en capas presenta las siguientes ventajas:

- Facilita agregar nuevos tipos de datos según sea requerido por la aplicación, ya que éstos son independientes del funcionamiento de las otras capas.
- Crea independencia de funcionamiento.
- Facilita el mantenimiento.
- Ofrece formas más sencillas para probar el correcto funcionamiento del sistema, al realizarse un sistema modular.
- Proporciona escalabilidad a la aplicación.

## 6.2. IMPLEMENTACIÓN DEL SISTEMA

Una vez diseñado el sistema y conociendo las principales funcionalidades y métodos que se deben realizar, se procede a la implementación de Proyecto CATEBus.

La implementación del sistema se ha dividido en la implementación de cada una de las capas arquitectónicas de las que se compone: vista, controlador y modelo.

### 6.2.1. Vista

La capa vista de Proyecto CATEBus la componen los siguientes elementos:

- Páginas JSP.
- Archivos CSS.
- Archivo JavaScript de comprobación de errores.
- Scripts AJAX.
- Llamadas a TagLibraries.
- Scripts Google Maps.

#### 6.2.1.1. Páginas JSP

Este tipo de páginas son las que representaran la información que será mostrada a los usuarios, y serán las encargadas de recoger sus acciones para comunicarse con la capa controlador y realizar las operaciones solicitadas.

La página principal de la aplicación será 'index.jsp', en la que se mostrará un formulario para el acceso del usuario al sistema. Desde ese mismo formulario ingresarán en la aplicación tanto los padres como representantes de rutas o personal docente.

El resto de páginas presentan una estructura común. En primer lugar presentan la cabecera del sistema web, consistente en el logotipo del proyecto, nombre del mismo y el botón que permite a los usuarios la salida del sistema, así como la eliminación de la información temporal almacenada por la aplicación.

A continuación se presenta el menú de acciones para los padres o el personal docente, que será distinto según se trate de uno u otro.

Por último, se presenta la información que será mostrada al usuario, ya sean datos para que pueda realizar acciones (representados en forma de formularios HTML) o listados de la información solicitada (presentados mediante tablas HTML).

Para el caso de los responsables de ruta, la presentación será ligeramente distinta. No tendrán ningún menú de acciones, sino que se presentarán una serie de formularios web en los que recogerán la información de aquellos alumnos que hayan faltado a la ruta el día en cuestión. Una vez finalizado dicho intercambio de información con el sistema, se procederá a finalizar la aplicación, cerrando la sesión del usuario.

Además existen dos páginas con las que se indican si las operaciones realizadas han tenido éxito o se ha producido un error en la realización de las mismas.

#### 6.2.1.2. Archivos CSS

Este tipo de archivos nos permite separar la representación de los datos de la página web, con los estilos o información gráfica referente a los mismos. De este modo, tenemos englobada la información de la presentación de todas las páginas que componen la aplicación.

Esta información está separada en dos archivos CSS. En el primero de ellos, denominado 'menus.css', se recogen todas las características visuales para la correcta presentación de los menús desplegados que presenta la aplicación.

El segundo de los ficheros, denominado 'estilos.css' contiene la información acerca de la presentación de los elementos más importantes que componen las páginas web, tales como enlaces, párrafos, títulos, tablas o el propio cuerpo de la página. Para cada uno de ellos se especifican los parámetros y datos para una correcta representación visual.

Además, se ha realizado un archivo específico para la presentación de los elementos visuales de la parte con accesibilidad móvil de la aplicación, denominado 'estilosM.css'. Gracias a esta distinción, es posible modificar aspectos visuales o de presentación de la aplicación, sin que ello afecte a la visualización en dispositivos móviles.

Dichos archivos pueden ser modificados por los responsables de los centros escolares según se adapten a sus necesidades (por ejemplo, si se incluye esta aplicación dentro de un sistema web propio y las propiedades de presentación debe ser acorde a las ya existentes).

### 6.2.1.3. Archivo JavaScript de comprobación de errores

Al realizar peticiones de información al usuario, éste puede introducir cualquier tipo de dato en los campos de los formularios, datos con formatos incorrectos, o incluso no introducir ninguno; con lo que es necesaria una comprobación de errores e informar al usuario de los datos que ha introducido incorrectamente, para que pueda corregirlos y comunicarse correctamente con el sistema.

Para que el servidor de la aplicación no realice esta tarea, y así se libere de carga de procesamiento, se ha elaborado un fichero JavaScript que contiene estas comprobaciones, denominado 'cErr.js', el cual contiene un método para cada uno de los formularios existentes en la aplicación. Para ello, al presionar el botón de envío de datos, se procederá a la llamada a estas funciones, pasándole como parámetros el formulario en cuestión.

Estas funciones comprueban que cada uno de los elementos del formulario posea datos, y que estos se ajusten a los valores y formatos adecuados, haciendo especial hincapié en los datos numéricos y en las fechas. Si los datos son correctos, se envía la información al servidor para su procesamiento. En caso contrario, se muestra al usuario una ventana con la información de los errores cometidos, y se corta el envío de información hasta que no sea pulsado nuevamente el botón que realiza dicha operación con los datos correctos.

### 6.2.1.4. Scripts AJAX

Algunas de las páginas realizadas, necesitan la intervención del servidor para mostrar cierta información de manera dinámica al usuario antes de que se produzca ningún envío de información al servidor. Por ejemplo, cuando un personal docente desea modificar una parada, en primer lugar se muestran las rutas disponibles y, en el momento en que el usuario selecciona una de ellas, se muestran automáticamente las paradas relativas a dicha ruta.

Este intercambio de información con el servidor se realiza mediante la utilización de scripts AJAX, que se encuentran dentro de las páginas JSP que hacen uso de esta funcionalidad.

Dichos scripts están formados por tres funciones principales:

### **Función inicializa\_xhr ()**

Esta función se emplea para encapsular la creación del objeto *XMLHttpRequest*. Este objeto es clave en las comunicaciones entre AJAX y el servidor, ya que permite realizar las comunicaciones con el servidor en segundo plano, sin necesidad de recargar las páginas. La implementación del objeto *XMLHttpRequest* depende de cada navegador, por lo que es necesario emplear una discriminación sencilla en función del navegador en el que se está ejecutando el código. Los navegadores que siguen los estándares (Firefox, Safari, Opera, Internet Explorer 7 y 8) implementan el objeto *XMLHttpRequest* de forma nativa, por lo que se puede obtener a través del objeto *window*. Los navegadores obsoletos (Internet Explorer 6 y anteriores) implementan el objeto *XMLHttpRequest* como un objeto de tipo ActiveX.

```
function inicializa_xhr() {  
    if (window.XMLHttpRequest) {  
        return new XMLHttpRequest();  
    }  
    else if (window.ActiveXObject) {  
        return new ActiveXObject("Microsoft.XMLSTTP");  
    }  
}
```

### **Función de carga de contenidos**

Las peticiones asíncronas que se producen mediante AJAX se realizarán una vez se haya cargado la página web. Por ello, se realiza esta función una vez se haya cargado la ventana del explorador. Cuando se haya producido esta carga, se realiza una llamada a la función *inicializa\_xhr ()* para preparar el objeto *XMLHttpRequest* denominado 'peticion'.

Una vez inicializada la petición, se prepara la función de que se encarga de procesar la respuesta del servidor. Esta acción se realiza mediante la propiedad 'onreadystatechange' del objeto 'peticion' e indicando la función que procesará.

A continuación, se realiza la petición HTTP al servidor web. El tipo de petición que se realizará será de tipo POST, y se indicará el nombre del recurso a solicitar en el servidor. Además, se establece la cabecera de petición 'Content-Type', para poder comunicarnos correctamente con la información intercambiada entre AJAX y el servidor. Para ello se establece el valor de esta cabecera a 'application/x-www-form-urlencoded'.

Para que el servidor pueda diferenciar desde que página web le llega la petición, se establecen cabeceras adicionales ésta para poder realizar una distinción de las mismas por parte de la aplicación, y gestionar su procesamiento de manera más eficiente y ordenada. En el caso de que sea necesario aportar más datos al servidor, se realizan de este mismo procedimiento.

Por último se envía la petición al servidor.

```
window.onload = function() {
    petición = inicializa_xhr();
    if (petición) {
        petición.onreadystatechange = muestraRutas;
        petición.open("POST",
"/servletAdmin?nocache="+Math.random(), true);
        petición.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded");

petición.setRequestHeader("pagina", "AjaxRutas");
        petición.send(null);
    } }
```

### **Función de muestra de contenidos**

Cuando se recibe la respuesta del servidor, la aplicación ejecuta de forma automática la función establecida anteriormente en la propiedad 'onreadystatechange'.

Esta función comprueba en primer lugar que se haya recibido la respuesta del servidor, mediante el valor de la propiedad 'readyState'. Si se ha recibido alguna respuesta, se comprueba que sea válida y correcta (comprobando si el código de estado HTTP devuelto es igual a 200).

Una vez comprobado que la respuesta es la esperada, se obtiene el elemento HTML que se actualizará con los valores recibidos de la petición. Estos valores deben ser previamente evaluados para comprobar que los datos que ha proporciona el servidor están bien contruidos. En caso de que sean los esperados, se actualiza dicho elemento.

Estos datos recibidos del servidor se encuentran en formato JSON.

```
function muestraRutas() {
    if (peticion.readyState == 4) {
        if (peticion.status == 200) {
            var lista = document.getElementById
("elegirRuta");

            var rutas = eval('(' +
peticion.responseText + ')');
            lista.options[0] = new Option ("-
selecciona -");

            var i=1;
            for (var x in rutas) {
                lista.options[i] = new Option
(rutas[x],x);
                i ++;
            } } } }
```

En el caso en el que no se obtenga respuesta del servidor o de que no haya elementos disponibles para mostrar, se indicará, en el elemento HTML en cuestión, un mensaje informando de que no se han encontrado resultados para la operación solicitada.

#### 6.2.1.5. Llamadas a Tag Libraries

Cuando el servidor envía información al usuario, las páginas web se encargan de visualizar los datos aportados por éste. En el caso de que la información no sea estática, sino que dependiendo de los resultados de las operaciones se muestre un tipo de información o unos datos distintos, se emplean el uso de las Tag Libraries para dichas acciones.

[12][13]En concreto, utilizamos una de las librerías estándar ya definidas: la Core Tag Library, que contiene un resumen de las etiquetas principales, entre las que se incluyen las relacionadas con las variables y control de flujo, así como una forma genérica para acceder a los recursos basados en URL, cuyo contenido puede ser incluido o procesado dentro de la página JSP.

De entre las etiquetas que nos ofrece esta librería se han utilizado las siguientes:

- `<c:choose>`: que permite realizar un control condicional de opciones.
- `<c:when>`: permite realizar una comparación o evaluación de parámetros o variables. Si resulta verdadera se accede al contenido encerrado entre en su etiqueta, hasta que se encuentra la etiqueta de cierre `</c:when>`.
- `<c:out>`: que permite el acceso al contenido de variables o parámetros para su impresión en la página web.

#### 6.2.1.6. Scripts Google Maps

[11]Para mostrar de forma más realista y, de esa forma, ayudar a la identificación de cada una de las paradas almacenadas en la aplicación, se ha utilizado la API de Google Maps, de forma que se visualicen mapas en los que se localizan las direcciones de las paradas.

Para ello se ha insertado código de la API antes mencionada, en forma de script JavaScript, en la página que recoge la información de cada una de las paradas: 'infoMapas.jsp'.

Las principales funciones para la visualización de mapas se especifican a continuación.

##### **Función initialize ()**

Esta función se encarga de inicializar las variables necesarias para la visualización el mapa (variable 'map') y la obtención de las coordenadas que mostrará el mapa (*geocoder*). Además, se especifican las opciones que establecen como se muestra el mapa, tales como el zoom, y el tipo de mapa.

```
function initialize() {
    geocoder = new google.maps.Geocoder();
    var myOptions = {zoom: 16, mapTypeId:
google.maps.MapTypeId.HYBRID }
    codeAddress();
    map = new
google.maps.Map(document.getElementById("map_canvas"),
myOptions);
}
```

### Función codeAddress ()

Para que una dirección sea visualizada en un mapa, se necesitan las coordenadas geográficas de la misma. Debido a que nuestra aplicación maneja direcciones físicas (calle, número, población y ciudad), se necesita realizar la traducción de dicha dirección a coordenadas (latitud y longitud).

Esto se consigue mediante el método 'geocode' del traductor a coordenadas. Para ello, necesitamos obtener la dirección que queremos traducir. Si la traducción se ha producido correctamente, incluimos esa dirección en el mapa.

Además, para mejorar la visualización de la localización en el mapa, se crea un icono que se situará en esa dirección ('marker'). Cuando se pulse con el ratón sobre el icono, aparecerá una ventana en que se visualizará el nombre completo de la dirección ('infowindow'). Esto último se consigue mediante la inclusión de un *listener*, el cual está esperando a la captación del clic del ratón y, en cuanto éste se produce, se abre la ventana con la información.

```
function codeAddress() {
    var mapas = document.getElementById("mapa");
    var address = algo.getAttribute("value");
    if (geocoder) {
        geocoder.geocode( { 'address': address},
function(results, status) {
            if (status == google.maps.GeocoderStatus.OK) {
                map.setCenter(results[0].geometry.location);
                var image = "/Images/icon.png";
                var marker = new google.maps.Marker({map: map,
position: results[0].geometry.location, title: address, icon:
image});

                var contentString = '<div id="content"><div
id="siteNotice"></div><div id="bodyContent"><p><b>' + address +
'</b></p></div></div>';

                var infowindow = new google.maps.InfoWindow({ content:
contentString, maxWidth: 200});

                google.maps.event.addListener(marker, 'click',
function() {infowindow.open(map,marker);});
```

```
        } else { alert("Error al obtener la información de  
GoogleMaps: " + status); }  
    });    } }
```

El mapa resultante aparecerá en la página web, en la etiqueta '<div id="map\_canvas"></div>', en la que se puede especificar atributos talos como la altura o anchura del mapa.

### 6.2.2. Controlador

El controlador de la aplicación está formado por dos servlets, así como por una serie de clases con una estructura común, denominadas 'helpers',

El primero de ellos, denominado 'servletAdmin', será el encargado de recoger cada una de las peticiones que realicen los usuarios que sean responsables de ruta y del personal docente del centro. Además, se encargará de la función de 'login' al sistema, es decir, será el encargado de comprobar que el usuario pertenece a la aplicación, de que la contraseña es correcta y le redirigirá a la página correspondiente. En el caso de los responsables de ruta y personal docente, les redirigirá a la página principal de sus respectivas funciones (indexAdmin.jsp o indexResponsable.jsp), y se creará la sesión del usuario en este servlet.

En el caso de que sea un padre el que acceda a la aplicación, se realizará una petición (por medio de una página jsp) al servlet correspondiente, creándose una sesión en el mismo y devolviendo la página de las acciones principales del mismo (indexPadres.jsp).

Como ya se ha comentado anteriormente, existe un segundo servlet (denominado 'servletPadres') en el que se recogerán y procesarán todas las peticiones que realicen los usuarios de tipo padre.

Gracias a esta distinción podemos repartir la carga de procesamiento de la aplicación en los dos servlets, optimizando la aplicación y el acceso a los recursos de la misma.

La implementación de los dos servlets es similar: reciben peticiones de la vista, y delegan su procesamiento a otras clases. Una vez procesada la información y obtenida una respuesta, ésta es enviada al servlet que procede a su devolución al usuario.

Para ayudar al servlet a procesar las peticiones, se han creado dos tipos de clases, mediante las cuales podemos organizar el acceso a todas las operaciones necesarias para dar una respuesta a los usuarios. Estas clases están englobadas en dos grupos (en la práctica son

dos paquetes java) denominados 'helper' y 'operaciones', que se explicarán en sucesivos epígrafes. Estas clases serán las que se comunicarán con el modelo para acceder a los datos o modificar la información de los mismos, así como de realizar las conexiones con la base de datos y realizar las operaciones requeridas sobre la misma.

A continuación presentamos un modelo de clases general del sistema implementado, indicando las dependencias entre los distintos paquetes y elementos.

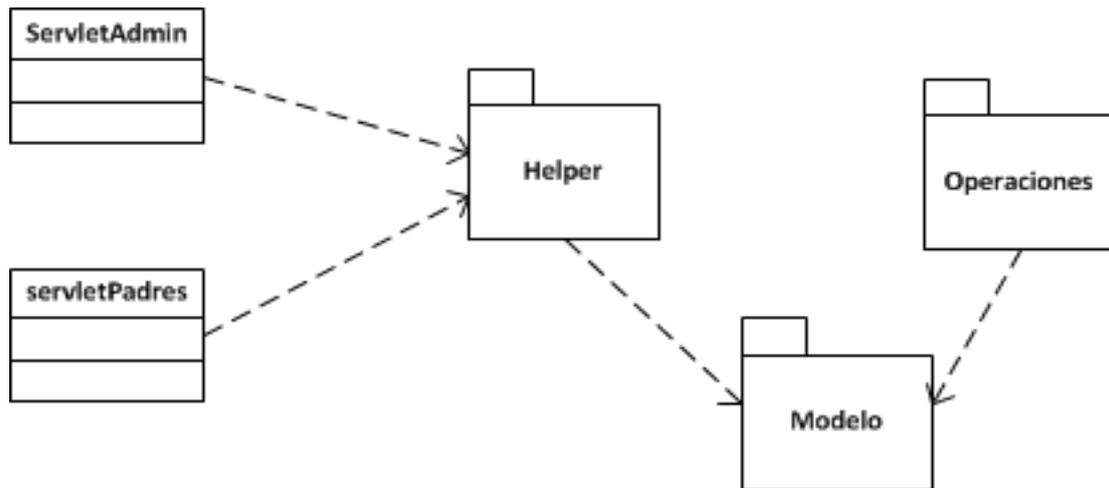


Ilustración 4. Diagrama de clases general

#### 6.2.2.1. Sesiones de servlet

Una vez que los usuarios han sido admitidos por la aplicación, se crea una sesión para cada uno de ellos. Estas sesiones se inicializan cuando el usuario entra en el sistema y se eliminan cuando el mismo sale, mediante la opción de cerrar sesión.

En las sesiones se almacena información relevante acerca del usuario, y que estará vigente y accesible hasta que la sesión acabe. Los datos que se almacenan son los siguientes:

- Identificador de usuario, que permitirá reconocer sobre que usuario estamos procesando información
- Nombre de usuario, el cual mostraremos en las páginas JSP para dar a conocer al manejador de la aplicación que ha accedido correctamente a la aplicación, y que es él el que está realizando las operaciones.

### 6.2.2.2. Helper

Este grupo de clases son las encargadas de recoger los parámetros y atributos de la petición del usuario, y transformarlos en los datos con los que la aplicación realizará las operaciones oportunas.

La estructura de las clases es similar para todas. En primer lugar existe un método que es llamado por el servlet correspondiente, y que identifica a cada petición de información. Este método realiza una llamada a la función que se encarga de procesar y recopilar la información para realizar dicha tarea.

Para las peticiones AJAX existe un método específico que accede a las operaciones de tratamiento de la petición.

Estas funciones se encargan de realizar las siguientes operaciones:

- Acceder a los parámetros y atributos de la petición, así como a la sesión del usuario si se necesita algún valor almacenado en ella.
- Comunicarse con el modelo para la creación de los objetos necesarios para procesar la información obtenida en los parámetros anteriores.
- Realizar llamadas a las funciones de las clases de operaciones, cuyos parámetros serán los objetos anteriormente creados, así como otro tipo de datos necesarios.
- Recoger los objetos devueltos por estas funciones y devolver la página web de destino, así como la respuesta a las operaciones (si es el caso) al servlet.

La relación de clases que pertenecen a este paquete son las siguientes:

- helpRuta: que se encarga de recoger y tratar los datos relativos a las rutas.
- helpParada: encargado de tratar la información relativa a las paradas.
- helpUsuario: encargado de recoger la información para la admisión de los usuarios en el sistema.
- helpAlumno: que se encarga de recoger información de los alumnos.
- helpPadre: encargado de la información relativa a los padres.
- helpFalta: encargado de tratar los datos de las faltas de asistencia.
- helpResponsable: recoge la información de las acciones de los responsables de ruta y se encarga de controlar la recogida de alumnos en las mismas.

A continuación se muestra el diagrama de clases simplificado (no aparecen los métodos de las clases), en los que se refleja la dependencia de los servlet con estas clases, ya que delegan sus funciones a las mismas.

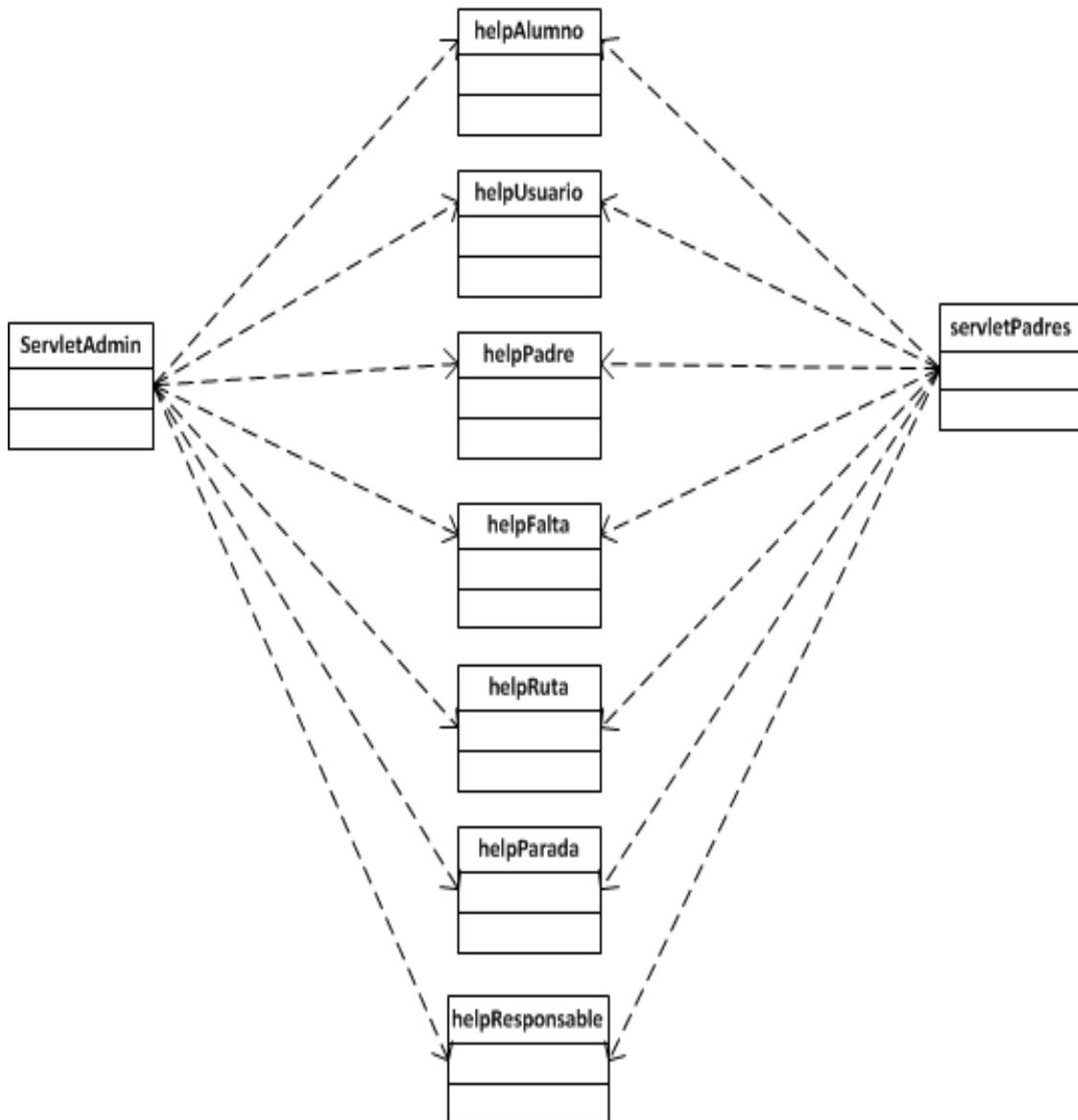


Ilustración 5. Diagrama de clases del paquete 'helper'.

### 6.2.2.3. Operaciones

Este grupo de clases son las encargadas de realizar las comunicaciones de la aplicación web con la base de datos, donde se encuentra almacenada la información.

Los métodos de estas clases son similares entre sí. En primer lugar realizan la conexión con la base de datos, una vez obtenida ejecutan una consulta o sentencia SQL para

la obtención o actualización de información con la misma. En caso de que estas operaciones hayan tenido éxito, se devuelven a las clases 'helper' el resultado obtenido o la nueva página web de destino. En caso contrario se devuelve la dirección de la página de error, que informará al usuario de que su operación no ha podido ser realizada.

Con respecto a las operaciones AJAX, el resultado devuelto serán los datos solicitados en notación JSON.

Para el acceso a la base de datos, es necesario otorgar a la aplicación la información de la localización de la base a consultar, así como del usuario y contraseña necesarios para acceder a la misma. Estos datos se almacenan en las siguientes variables:

```
String sURL = "jdbc:mysql://localhost:3306/CateBus";  
String sUsuario = "CateBus";  
String sPwd = "root";
```

A continuación, se utiliza el conector JDBC, que nos permite comunicar nuestra aplicación Java con MYSQL. Para ello, realizamos una nueva instancia de dicho conector, indicando el sistema gestor de bases de datos utilizado. Una vez instanciado, se procede a realizar la conexión con la éste.

```
Class.forName("com.mysql.jdbc.Driver").newInstance();  
con = DriverManager.getConnection(sURL,sUsuario,sPwd);
```

Existe una clase operaciones por cada una de las clases 'helper' anteriores. La nomenclatura de las clases es la siguiente: *nombreOp*, siendo *nombre* una de las entidades tratadas (alumno, padre, responsable, usuario, falta, parada, ruta).

A continuación se presenta el modelo de clases de operaciones, en el cual se refleja la dependencia entre estas clases y el paquete 'helper'.

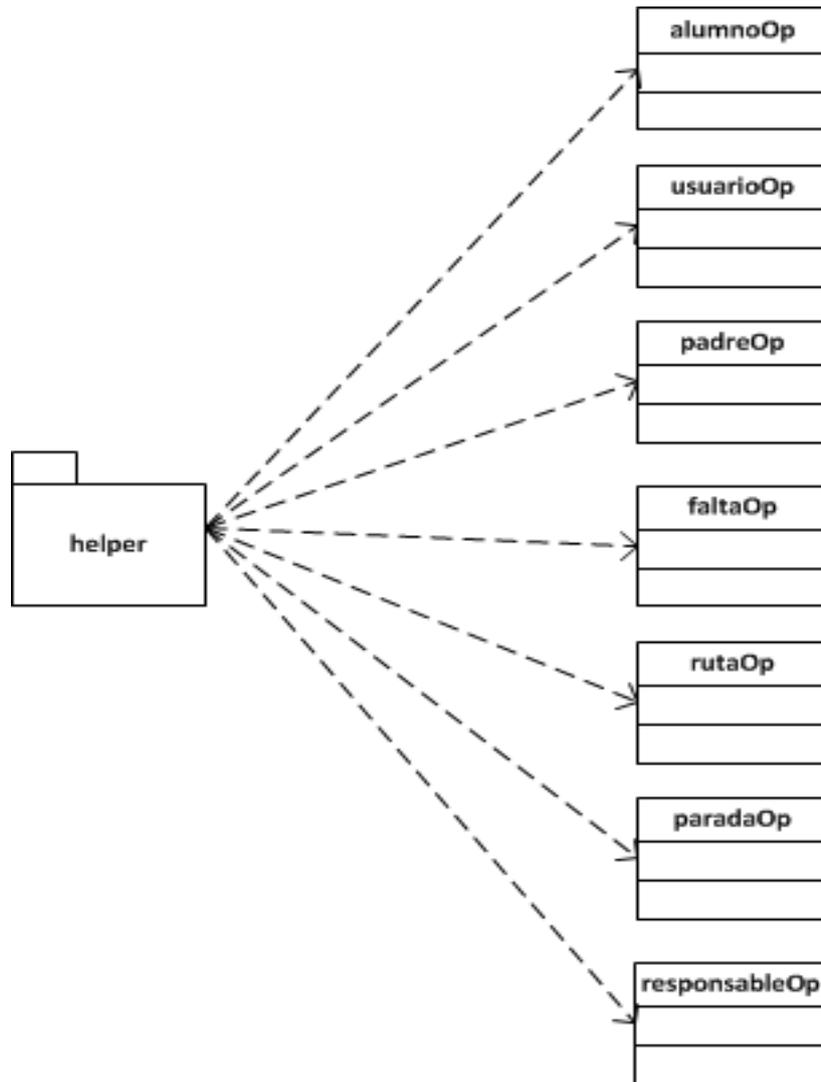


Ilustración 6. Diagrama de clases de operaciones.

#### 6.2.2.4. Adaptación a dispositivos móviles

Además de la aplicación principal, se ha generado una parte de la misma para poder ser visualizada en dispositivos móviles. Para ello, se ha utilizado el lenguaje XHTML, que proporciona mayor robustez y restricciones que HTML, lo que permite una mejor definición y estructura de las etiquetas de los documentos, de forma que sean entendibles por los navegadores de manera más eficiente. Conjuntamente, restringe el uso de CSS a archivos separados del documento, permitiendo un mínimo número de etiquetas en el mismo, de forma que permite separar la estructura y contenido de la página, con la presentación de la misma.

Para la realización de estas páginas, se han utilizado formatos de presentación adaptados a estos dispositivos, tales como tamaño de letra, tamaño de la información a mostrar, así como evitar el uso de tablas HTML, las cuales algunos dispositivos no muestran ni tabulan correctamente.

Estas páginas adaptadas a móviles, serán accesibles por dos tipos de usuarios, padres de los alumnos y el personal docente del centro. En el caso de los responsables de ruta no es necesaria, ya que está pensada e implementada para visualizarse en PDAs u otros dispositivos similares, los cuales procesan correctamente la información mostrada, y no es necesaria su adaptación.

En el caso de los primeros, esta versión se realiza debido a que podrían necesitar las funciones de la aplicación en diversas situaciones en las que no puedan acceder desde su hogar (no poder acceder a un ordenador, por ejemplo). Las funciones implementadas son:

- Pedir retraso: para aquellos padres que no puedan acceder a un ordenador un día concreto, o aquéllos que simplemente prefieran solicitar un retraso de última hora a través del móvil, se podrá solicitar el retraso en la parada el mismo día en el que se solicita.
- Información de la parada: si por algún motivo se necesita consultar las señas o dirección de la parada de un alumno, se podrá realizar a través de estos dispositivos, dando como resultado una página que genera un mapa (a través de Google Maps), mostrando, además, la dirección de la misma de manera textual.
- Solicitar parada ocasional: para aquéllos usuarios que necesiten cambiar de parada un día específico y no dispongan de un ordenador, podrán acceder a esta funcionalidad desde un móvil, indicando el alumno, la nueva parada y la fecha en la que se solicitará.
- Planificar falta de asistencia: se podrán planificar faltas desde estos dispositivos, seleccionando el alumno que faltará y la fecha en que se producirá dicha situación.

Para generar los mapas anteriormente mencionados, se utilizan los mapas estáticos de Google Maps [14]. Estos mapas se generan mediante una dirección URL, que será asociada a un elemento imagen del documento XHTML. La dirección está compuesta por la expresión `http://maps.google.com/maps/api/staticmap?`, seguida de una serie de parámetros, que son los siguientes:

- Center: Lugar donde estará centrado el mapa a generar, normalmente la dirección que se desea localizar. Lo estableceremos como una cadena de dirección, aunque también admitiría coordenadas geográficas.
- Zoom: Acercamiento del mapa sobre la posición elegida, en este caso un nivel de zoom de 15.
- Size: Tamaño del mapa generado, en este caso 250x250 pixeles.
- Mapttype: Tipo de mapa que se mostrará, es este caso un mapa normal.
- Mobile: Si la imagen generada debe estar adaptada dispositivos móviles, que se establecerá a verdadero (true).
- Sensor: Si el dispositivo en el que se mostrará, posee la tecnología para poder informar en qué posición se encuentra. Por defecto estará establecido a falso (false).
- Markers: marcadores que aparecerán en el mapa. En este caso aparecerá uno sobre la localización deseada. Para ello se especifican los siguientes valores, separados mediante el carácter "|".
  - Color: en este caso azul.
  - Etiqueta: en este caso se mostrará una P, reflejando que se trata de una Parada.
  - Dirección, en la cual estará localizada la etiqueta.

Así, la cadena de dirección que generará los mapas tendrá este formato:

```
http://maps.google.com/maps/api/staticmap?centerDIRECCION_D
E_LA_PARADA&zoom=15&size=250x250&mapttype=roadmap&mobile=true&sen
sor=false&markers=color:blue|label:P|DIRECCION_DE_LA_PARADA
```

En el caso del personal docente del centro, esta versión está implementada para que puedan acceder a ella aquéllos que no dispongan de un ordenador (equipo averiado, actualización de sistemas) o bien aquéllos que no puedan acudir al centro escolar (baja laboral, reuniones en otros centros) y deseen acceder a algunas funciones de la aplicación. Las funcionalidades implementadas para ellos son las siguientes:

- Gestionar rutas: Podrán añadir y eliminar rutas de forma sencilla desde sus teléfonos móviles, así como asignar directamente un responsable a las nuevas rutas, de entre los que están disponibles (no asignados a ninguna).

- Gestionar paradas: Los usuarios podrán añadir, modificar o eliminar paradas de la misma forma que en la versión normal de la aplicación.
- Gestionar responsables de ruta: Se habilitará el registro de nuevos responsables, la eliminación de éstos y su asignación a las rutas disponibles.
- Gestionar padres y alumnos: Los usuarios serán capaces de dar de alta a nuevos padres, así como también a sus respectivos hijos en la aplicación.

#### 6.2.2.5. Uso de Java Mail

En este proyecto se emplea el uso de esta API [12] para el envío de correos electrónicos que informen a los usuarios (padres y responsables de ruta) de su nombre de usuario y su contraseña de acceso al sistema. Para ello, se han creado dos clases nuevas, localizadas en el paquete 'Mail' que permiten el envío de correos.

La clase mandarMail.java es la encargada de la elaboración del mensaje, la configuración de los parámetros para el envío del correo, y del propio envío de mismo. Así, en primer lugar establecemos las propiedades de la conexión con el servidor SMTP que se utilizará, así como de las opciones que se usarán en el mismo.

En la realización de esta aplicación, se ha utilizado el servidor SMTP que ofrece Gmail (el servicio de correo electrónico de Google) para el envío de mensajes. Este servidor puede ser reemplazado por el servidor de correo que utilicen los centros escolares en los cuales se instalará la aplicación.

Las propiedades que estableceremos serán las siguientes:

- Host: dirección del servidor SMTP, en este caso el de Google.
- Puerto: puerto de escucha del servidor, en ese caso el 587.
- Disponibilidad de TLS: se trata de un protocolo criptográfico que permite establecer comunicaciones seguras en Internet.
- Nombre de usuario: nombre de la cuenta de correo desde la que se enviará el mismo.
- Autenticación del usuario: establece si se necesita dicha autenticación en el servidor SMTP.

Una vez establecidas dichas propiedades, creamos una nueva sesión en el servidor de correo, que nos permitirá el envío de los mensajes. A continuación procedemos a la

construcción del mensaje. El mensaje enviado seguirá las especificaciones del protocolo de intercambio de información MIME. Para ello, creamos un nuevo mensaje MIME, que necesita como argumento la sesión creada anteriormente.

A continuación establecemos las cabeceras del mensaje. Estas son: el tipo de contenido, que será texto plano y con el juego o grupo de caracteres que se utiliza Europa occidental; y que los caracteres sean imprimibles, es decir, que su representación sea imprimible en pantalla, sin caracteres que representan datos o variables de control.

Seguidamente, establecemos el emisor del mensaje, el destinatario del mismo, y el asunto del correo a enviar. Por último se proporciona el texto a enviar.

Para finalizar, se realiza el envío del mensaje. Para ello se utiliza la clase 'Transport' incluida en el API Java Mail, e implementada en la sesión que hemos creado anteriormente. Una vez obtenido el objeto anterior, se procede al envío del mensaje a los receptores establecidos en el cuerpo del mensaje.

A continuación se muestra el código que implementa las operaciones anteriormente mencionadas. En primer lugar la función que realiza la construcción de los parámetros, así como del envío del mensaje

```
public static void send() throws Exception{
    String host ="smtp.gmail.com";
    String puerto = "587";
    String from ="CATEBus@catebus.com";
    String to = user.getemail();

    Properties prop = new Properties();
    prop.setProperty("mail.smtp.host", host);
    prop.setProperty("mail.smtp.port", puerto);
    prop.setProperty("mail.smtp.auth", "true");
    prop.setProperty("mail.smtp.starttls.enable", "true");
    prop.setProperty("mail.smtp.user", USUARIO SMTP);

    try{
        SMTPAuthentication auth = new SMTPAuthentication();
        Session session = Session.getInstance(prop , auth );
        Message msg = getMessage(session, from, to, user);
```

```
Transport t = session.getTransport("smtp");
t.connect();
t.sendMessage(msg, msg.getAllRecipients()); }
catch(Exception e) {
    e.printStackTrace();
    return null;}}
```

Por último, se muestra el código que permite la creación de un nuevo mensaje en formato MIME.

```
private static MimeMessage getMessage() {
    try {
        MimeMessage msg = new MimeMessage(session);
        Msg.setHeader("Content-Type", "text/plain; charset=iso-
8859-1");
        msg.setHeader("Content-Transfer-Encoding", "quoted-
printable");
        msg.setText("Bienvenido al servicio CATEBUS \n\r. Usted
ha sido de alta con la siguiente información: \n\r + "Nombre de
usuario: NOMBRE_USUARIO \n\r" + "Contraseña: CONTRAÑA GENERADA
\n\r" + "\n\rAtentamente,\n\r El servicio de CATEBus.");
        msg.addRecipient(Message.RecipientType.TO, new
InternetAddress(to));
        msg.setFrom(new InternetAddress(from, "Servicio de
CATEBus"));
        msg.setSubject("Alta el CATEBUS");
        return msg;
    }
}
```

La segunda clase creada, 'SMTPAuthentication', extiende del autenticador de Java Mail, y en la cual detallamos el nombre de usuario y la contraseña necesarias para identificarse en el servidor SMTP en el caso de que sea necesario.

```
class SMTPAuthentication extends javax.mail.Authenticator
{
    public PasswordAuthentication
getPasswordAuthentication()
    {
        String username = NOMBRE DE USUARIO DEL SERVIDOR;
        String password = CONTRASEÑA
        return new PasswordAuthentication(username,
password);
    }
}
```

#### 6.2.2.6. Uso de iText

Para la generación de los justificantes de faltas de asistencia de los alumnos, se ha utilizado la herramienta iText [16]. Esta API nos permite la generación de documentos PDF a partir de la información almacenada en la aplicación.

Para ello, en primer lugar se ha de incluir dicha API en las bibliotecas del proyecto, para poder acceder a sus clases y métodos. Dichas librerías son 'itextpdf', que contiene las clases básicas y generales para la creación manejo de los archivos PDF, y 'itext-xtra', con funcionalidades añadidas.

Una vez incluidas las librerías, se especifican los tipos de letra que se utilizarán en el documento. Esta acción se realiza creando un nuevo objeto de la clase 'Font', indicando la familia de fuentes a utilizar, el tamaño de la misma, y, opcionalmente, otras posibilidades de presentación (negrita, cursiva, subrayado, etc.).

```
Font titulo = new Font(FontFamily.HELVETICA, 32, Font.BOLD);
Font subtitulo = new Font(FontFamily.HELVETICA, 12,
Font.UNDERLINE);
Font cuerpo = new Font(FontFamily.HELVETICA, 10);
```

A continuación se crea el documento PDF, indicándole el formato que tendrá (en este caso A4), así como el tamaño de los márgenes. Aunque esta última acción se puede realizar más adelante, es aconsejable definir los márgenes en el momento de creación del documento. Una vez creado, se realiza una instancia del mismo, indicándole la ruta en la que se

almacenará. Dicha ruta será la carpeta 'Justificantes', y será nombrado el archivo por su identificador de la falta a la que representa.

```
final String file = "/Justificantes/" + falta.getId()+
".pdf";
Document documento = new Document(PageSize.A4,50,50,50,50);
PdfWriter.getInstance(documento, new
FileOutputStream(file));
```

Una vez creado e instanciado, se comienza a insertar información en el mismo. En primer lugar, antes de introducir cualquier texto o imagen, se cumplimentan las siguientes propiedades del documento:

- Autor, en este caso el nombre del desarrollador de la aplicación. Este dato puede ser modificado por el nombre del centro escolar o del departamento del mismo.
- Aplicación o Método de creación, en este caso desde la aplicación CATEBus, utilizando iText.
- Título y asunto del documento, en este caso se trata del justificante de falta de asistencia de un alumno.

```
documento.open();
documento.addAuthor("Javier Sanabria Fernández");
documento.addCreator("CATEBus using iText");
documento.addTitle("Justificante de Falta - CATEBus");
documento.addSubject("Justificante de Falta de Asistencia
del alumno");
```

A continuación se insertará la imagen del logotipo de la aplicación. Para ello, se creará un objeto 'Image', al cual se le indicará la ruta de dicha imagen, la escala a la que se presentará y su alineación en el documento. Una vez creada, se añade en el documento PDF.

```
Image imgn;
final String logo = "/Justificantes/Recursos/Logo.gif";
imgn = Image.getInstance(logo);
imgn.scaleToFit(50,50);
imgn.setAlignment(Element.ALIGN_RIGHT);
```

```
documento.add(imgn);
```

Seguidamente, se crean los párrafos que contendrán la información en formato texto del documento. Para ello, se crea un objeto párrafo, al que se le indica el texto o información que contendrá, y el tipo de fuente que se le aplicará (de las creadas anteriormente). Posteriormente, se define la alineación que tendrá en el documento, así como el espacio que se dejará antes y después del párrafo. De esta forma se mejorará el espaciado y la presentación de la información. Una vez especificada esta información, se añade el párrafo al documento.

```
Paragraph p;  
p = new Paragraph ("CATEBus", titulo);  
p.setAlignment(Element.ALIGN_CENTER);  
p.setSpacingBefore(-50);  
p.setSpacingAfter(25);  
documento.add(p);
```

De esta forma se van añadiendo párrafos siguiendo el mismo procedimiento hasta que el documento queda completado. Para finalizarlo, se debe producir el cierre del mismo, indicando que el documento se ha finalizado y está disponible para su uso en la ruta especificada.

```
documento.close();
```

#### 6.2.2.7. Ficheros CSV

Para automatizar la subida de datos de los nuevos alumnos se ha realizado una funcionalidad por la que se podrá leer la información contenida en ellos y grabar estos datos en la base de datos. Para realizar esta funcionalidad se ha utilizado la biblioteca JavaCSV[17].

Para ello, se deberá tener un fichero CSV que reúna las siguientes características:

- Datos: Debe contener:
  - El identificador del padre en la aplicación.
  - Los apellidos del alumno.
  - El nombre de los mismos.

- Delimitador: Punto y coma ";".
- Cabecera: Que contenga una cabecera, da igual el texto que contenga.

Un ejemplo de contenido del fichero sería el siguiente:

```
Padre;Apellidos Alumno;Nombre Alumno
50712455;Jimenez Calvo;Pedro
50712455;Jimenez Calvo;Sandra
50712455;Jimenez Calvo;Leticia
43025789;Ramirez Sanchez;Roberto
48014300;Fernandez de la Torre;Alfonso
```

Una vez que hemos obtenido el fichero, se procede a abrir el mismo para su lectura. Para ello, especificamos la ruta donde se encuentra el archivo, y cuál es el delimitador que se ha utilizado. De esta forma creamos un nuevo lector de CSV.

```
String delimitador=";";
final String file = RUTA_DEL_ARCHIVO
FileReader r = new FileReader(file);
Leer = new CsvReader (r,delimitador.charAt(0));
```

A continuación procedemos a leer las cabeceras del fichero. Seguidamente procedemos a leer cada registro (línea) almacenada en el fichero, accediendo a su contenido para ser almacenado en un objeto alumno. El acceso a cada elemento se realiza por su posición en el fichero, siendo la primera posición el número cero.

```
String[] headers = null;
if(leer.readHeaders()) {
    headers = leer.getHeaders();
}
List lista = new ArrayList();
while(leer.readRecord()) {
    Alumno alumno = new Alumno();
    alumno.setPadre(Integer.parseInt(leer.get(0)));
    alumno.setApellidos(leer.get(1));
    alumno.setNombre(leer.get(2));
    lista.add(alumno);}
```

Para finalizar, se cierra el fichero y se concluye su lectura

```
leer.close();
```

#### 6.2.2.8. Subida de archivos

Se realizan subidas de ficheros en este proyecto para que los usuarios puedan utilizar ficheros CSV para la introducción de datos en la aplicación, de forma que el fichero pueda estar localizado en cualquier ordenador y no se le obliga a colocarlo en una ubicación específica y con un nombre específico en el servidor. Para ello, utilizamos las librerías Commons fileUpload[18] y Commons IO[19].

Antes de tratar con los datos en la aplicación, se necesita especificar que en el formulario HTML en el que se selecciona el archivo a subir, la codificación debe ser especial y especificar el siguiente atributo. De esta forma, las librerías lo detectarán y se podrá realizar la subida del archivo.

```
enctype="MULTIPART/FORM-DATA"
```

En primer lugar, se necesitan crear unas clases con las que manejaremos los datos que se envían al servlet. De esta forma recuperamos los elementos de la petición al servidor en una lista, de forma que podemos recorrerla y analizar los elementos uno a uno. Como se puede comprobar, se accede a la petición de distinta forma que en el resto de la aplicación.

```
FileItemFactory file_factory = new DiskFileItemFactory();
ServletFileUpload servlet_up = new
ServletFileUpload(file_factory);
List items = servlet_up.parseRequest(request);
```

Una vez obtenida la lista de los elementos, procedemos a recorrerla en busca del fichero que se desea subir. Así, consultamos si el elemento de la lista cumple la propiedad *isFormField*. En caso correcto, se trata del fichero solicitado, y se procede a copiarlo en el servidor. En caso contrario, se sigue buscando un elemento que lo cumple.

```
for(int i=0;i<items.size();i++){
    FileItem item = (FileItem) items.get(i);
    if (! item.isFormField()){
        File subido = new File(RUTA_FICHERO);
        item.write(subido); }}
```

#### 6.2.2.9. Fichero de Propiedades

Como se ha comentado en las especificaciones, los administradores podrán elegir si desean introducir los datos de los alumnos de forma manual (uno a uno) o de forma automatizada mediante un fichero de datos.

Para poder realizar esta distinción se ha utilizado un fichero de propiedades. Estos ficheros están compuestos por una serie de valores diferenciados mediante una clave, de esta forma: "clave = valor". Estos valores hacen referencia a variables o posibilidades de la aplicación web, que hace, entre otros usos, que ciertos datos o elementos puedan ser personalizados o sustituidos en la aplicación.

En este caso, se diferenciará si, al pulsar en el menú la opción "Añadir alumno", se redirigirá a la página en la que se añaden de uno en uno, o a un formulario para poder subir un fichero de datos. Para ello, el valor que se almacena en el fichero es "csv". Esta clave puede tomar el valor "TRUE" o "FALSE", según se habilite la subida del fichero o no.

Así, cuando se accede al enlace, el servlet realiza una lectura del fichero de propiedades y, según el valor leído, redirige al usuario a la página especificada.

Para almacenar el valor de la clave se realiza mediante la opción del menú "Personalizar". En ella, se puede seleccionar la opción deseada que se establezca por defecto. Una vez seleccionada, la aplicación accede al fichero de propiedades y carga el nuevo valor en la clave.

La utilización de este tipo de ficheros se realiza por si, posteriormente, los centros educativos quieren añadir más tipo de personalizaciones (tales como páginas distintas, cambio de colores de fondo, inclusión de logos, etc.), se puedan realizar estableciendo en este fichero los datos necesarios para ello (rutas de archivos, colores, valores de datos especiales...).

### 6.2.3. Modelo

El modelo de la aplicación está compuesto por las clases que contienen la información esencial y los datos que la aplicación irá creando, manejando o modificando; y cuyos valores serán los recogidos por las peticiones del usuario, o aquellos valores recuperados de la base de datos.

Estas clases son JavaBeans, los cuales contienen únicamente los atributos de los objetos a los que representan, así como los métodos 'get' y 'set' para poder acceder y modificar la información de los mismos.

Estas clases representan a los siguientes objetos del mundo real: usuarios, alumnos, paradas, rutas y faltas.

### 6.2.4. Descripción de la implementación de las funcionalidades

Es este apartado se detallarán las principales operaciones que se realizan en el sistema, así como se adjuntarán los diagramas de secuencia simplificados entre los distintos sistemas y clases.

#### 6.2.4.1. Inicio de sesión

Cuando un usuario desea iniciar sesión, introduce su nombre de usuario y contraseña en el formulario ofrecido. Al pulsar el botón entrar se pasan dichos valores, junto con el nombre de la página, y el resto de la petición a 'servletAdmin'. Éste, según el nombre de la página recibido, delega a otra clase el procesamiento de la información. Esta acción del servlet se repite para todas las operaciones que se realicen entre el navegador del cliente y el servidor de la aplicación, y no volverá a ser detallada.

Una vez delegado el procesamiento a la clase 'helpusuario', ésta se crea un nuevo objeto 'Usuario' para poder operar los datos introducidos con la base de datos. Además, se realiza la transformación de la clave recibida, a un hash codificado mediante el algoritmo SHA-1, que es uno de los que proporcionan mayor nivel de seguridad.

Una vez realizadas estas operaciones, se produce una llamada al método correspondiente de la clase 'usuarioOp', la cual realiza la comprobación del usuario.

En primer lugar se realiza una consulta a la base de datos para saber si el nombre de usuario existe en la aplicación. Una vez que la comprobación resulta exitosa, se comprueba que la contraseña almacenada en la base de datos es la misma que la que ha codificado. Si las contraseñas coinciden, se almacena en el sistema el identificador del usuario obtenido en la consulta, y con dicho dato se comprueba si el usuario pertenece a alguno de los tipos soportados por la aplicación (padres, responsables de ruta o personal docente).

En el caso de los responsables de ruta o personal docente, esta función devuelve la dirección de la página principal a sus operaciones a 'helpUsuario', y éste al servlet; el cual crea un objeto de sesión y almacena en él el identificador del usuario, así como su nombre. Si se trata de un usuario padre, la página devuelta será la de direccionamiento al 'servletPadre', creándose una sesión para el usuario en dicho servlet y no en el del resto de usuarios.

Si se produce algún error en la comprobación del usuario y contraseña, se devuelve a la página de entrada al sistema de nuevo, para que el usuario introduzca unos nuevos valores correctos.

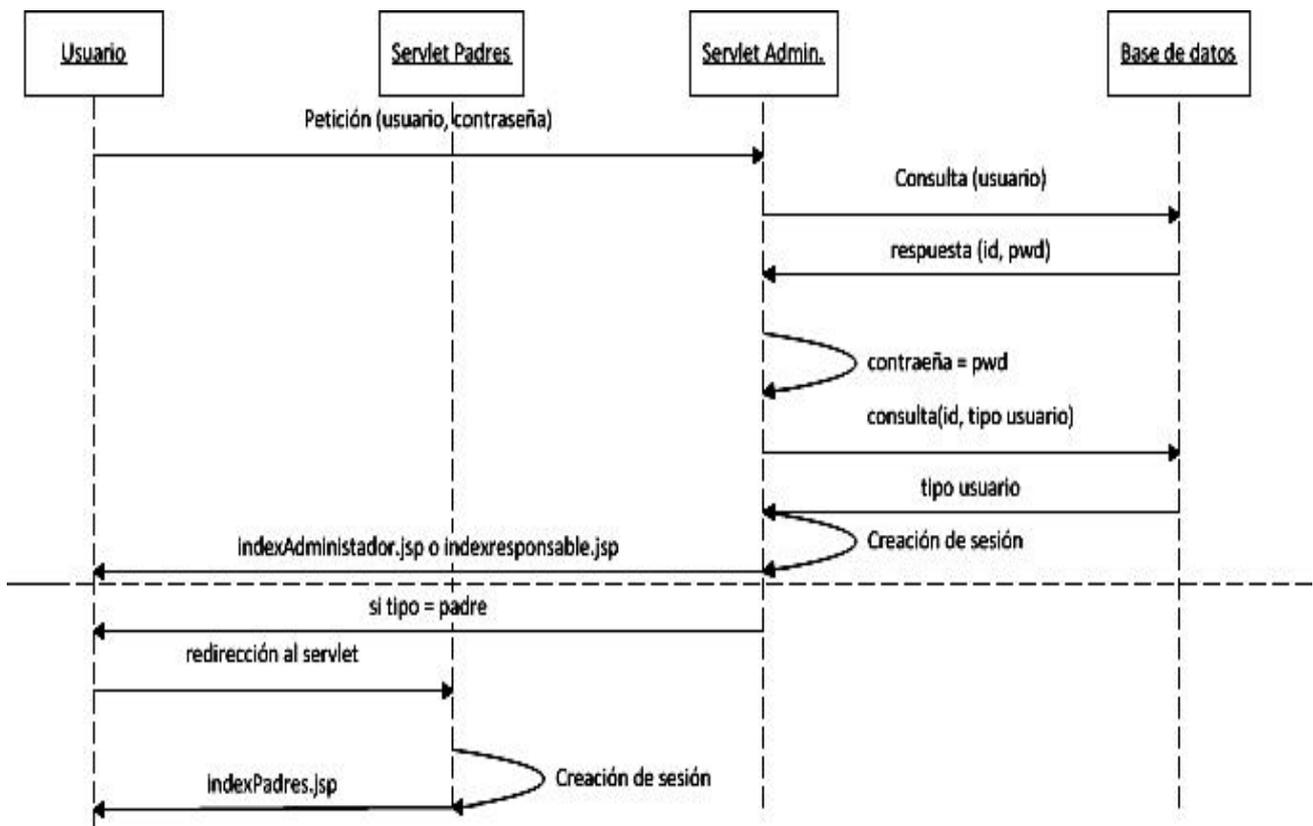


Ilustración 7. Diagrama de secuencia de inicio de sesión.

#### 6.2.4.2. Creación de una nueva ruta

Para crear una nueva ruta, el personal docente debe de seleccionar dicha opción en el menú, e introducir el nombre de la ruta y el usuario que será responsable de la misma. La lista de nombres de responsables se realiza mediante peticiones AJAX al servlet, quien reconoce la petición y delega la misma a la clase 'helper', y ésta a la clase de operación correspondiente. Dicha clase se conecta a la base de datos, obtiene los identificadores y nombres de los responsables y procede a formatear estos datos a notación JSON para ser devuelta a la página web. Ésta recibe la información y rellena la lista desplegable con los nombres de usuario de los responsables de ruta. Esta operación se realiza siempre que haya que seleccionar datos de listas desplegables, o que requieran mostrar información almacenada en la base de datos.

Una vez introducida la información, y asegurado que no contiene errores (si los tuviera, se mostraría una ventana con los errores encontrados durante la comprobación de los mismos mediante JavaScript), se pulsa el botón correspondiente y se envía la petición al servlet, que reconoce la misma y delega en la clase 'helper'. Ésta obtiene la sesión del usuario para acceder a su identificador y lo almacena junto con el resto de datos en una nueva instancia de la clase ruta del modelo.

A continuación se instancia la clase operaciones correspondiente y se realiza la inserción de los datos obtenidos, junto con un identificador de ruta creado en la operación, en la base de datos. Si ocurre algún error se devuelve la dirección de la página de error.

Si no se produce ningún error, la clase 'helper' devuelve la dirección de la página que permite añadir paradas a la ruta. El usuario deberá introducir los datos necesarios y pulsar el botón correspondiente. En caso de que los datos sean correctos, se procesa la petición en el servlet, que vuelve a delegar al 'helper' y a la clase de operaciones correspondiente. Para ello, se crea una nueva instancia de la clase 'parada' del modelo y se guardan sus valores para ser almacenados posteriormente en la base de datos.

Si no se produce ningún error, se devuelve nuevamente la dirección de la página encargada de los datos de las nuevas paradas, así hasta que se pulse el botón que representa la acción de no introducir más paradas, en cuyo caso se cargará la página de inicio del personal docente.

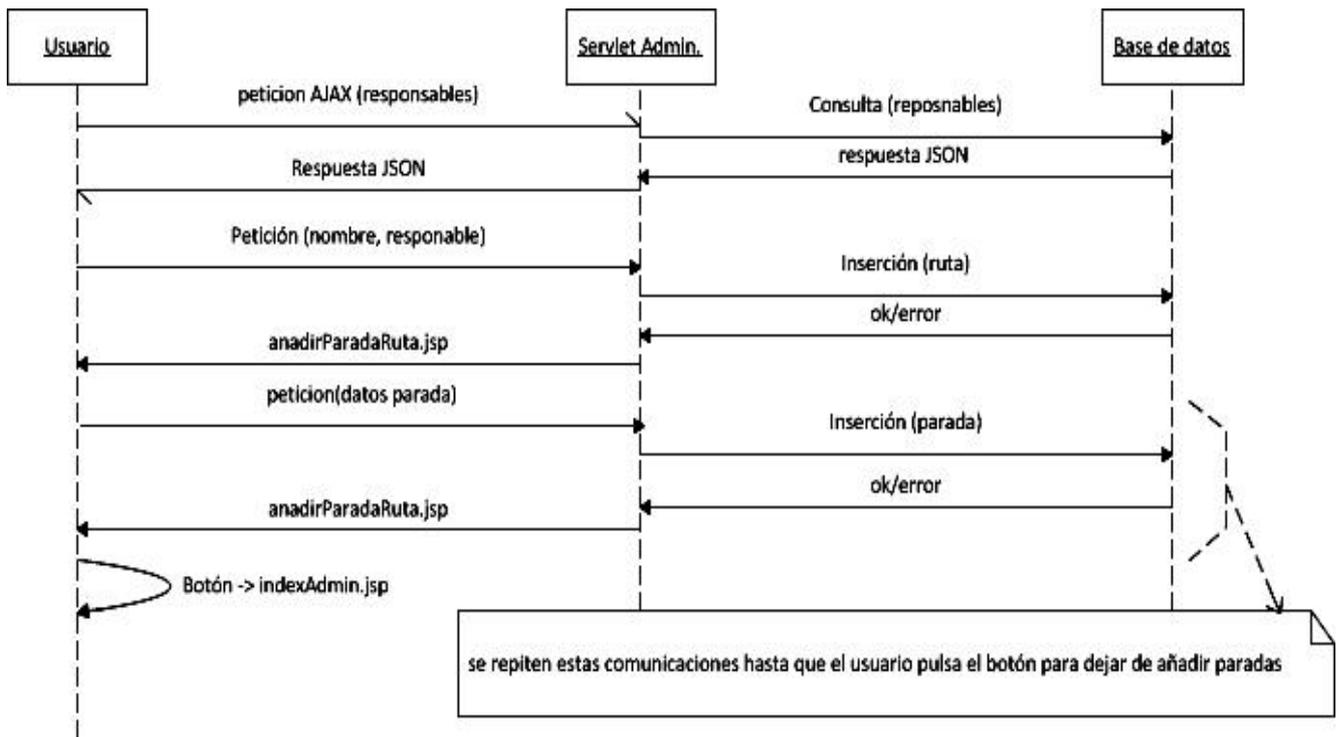


Ilustración 8. Diagrama de secuencia de nueva ruta.

#### 6.2.4.3. Eliminación de una ruta

El usuario selecciona de la lista desplegable (generada mediante AJAX) la ruta que desea eliminar. Esta petición llega al servlet que delega la petición a las clases correspondientes, las cuales instancian a la clase ruta del modelo y ejecutan la sentencia de borrado de dicha ruta en la base de datos.

Si no hay ningún error se devuelve a la página de éxito en la operación; en caso contrario, se le redirige a la página de error.

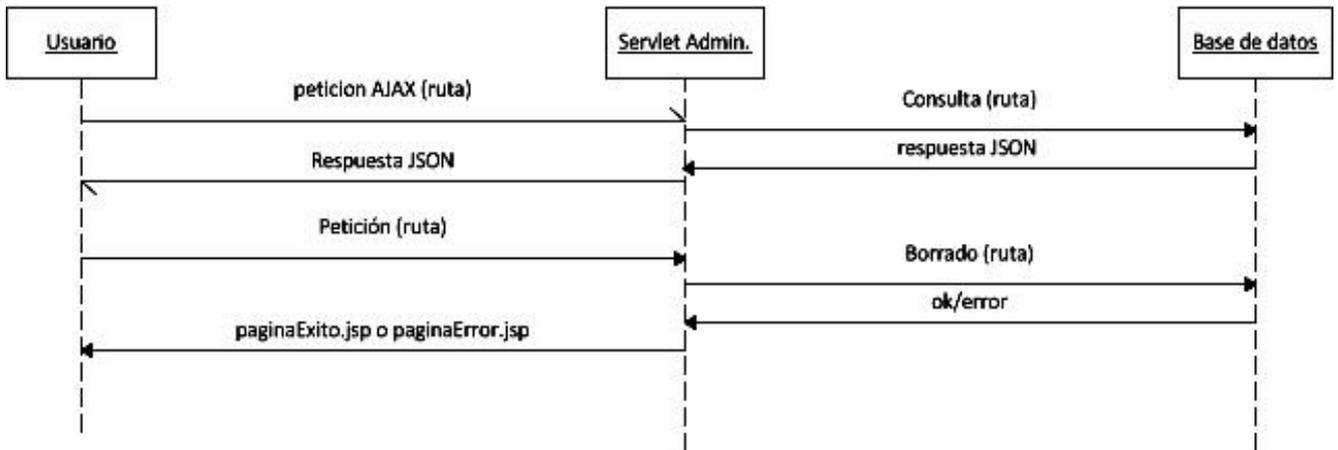


Ilustración 9. Diagrama de secuencia de eliminar ruta.

#### 6.2.4.4. Creación de una nueva parada

Si el usuario desea crear una nueva parada, debe introducir los datos de la misma, así como seleccionar la ruta que en la que desea que se introduzca. Si todo es correcto, la petición es enviada al servlet, que delega en las clases correspondientes las operaciones que realizan la creación de la instancia de parada y la almacenan en la base de datos.

Si no hay ningún error se devuelve a la página de éxito en la operación; en caso contrario, se le redirige a la página de error.

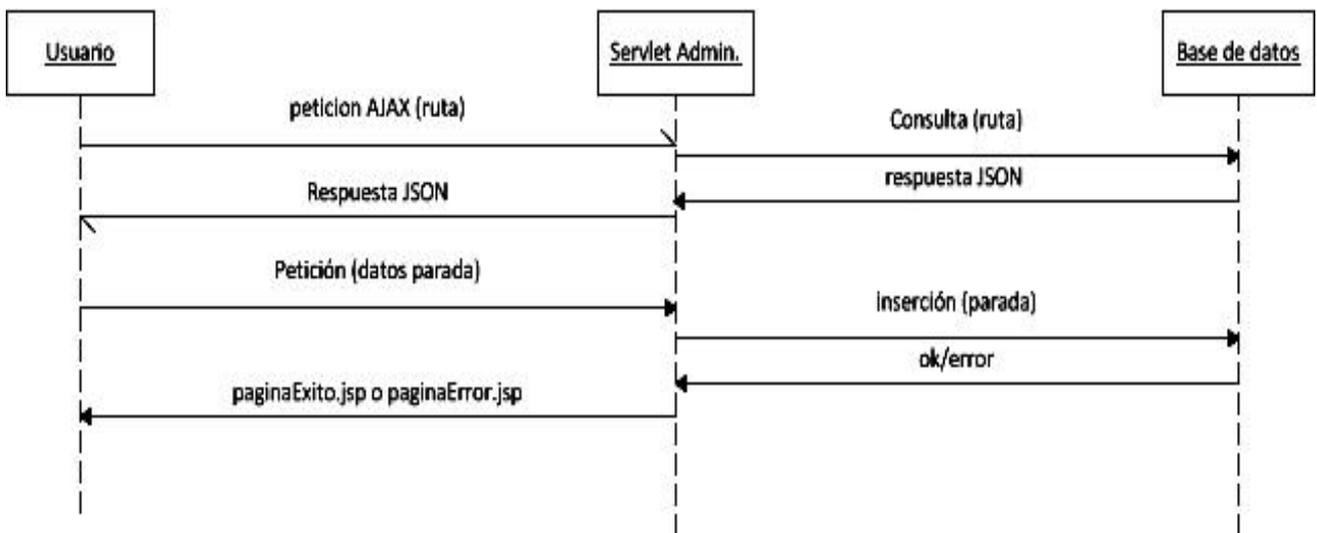


Ilustración 10. Diagrama de secuencia de insertar parada.

#### 6.2.4.5. Modificación de una parada

En esta ocasión se selecciona de la lista desplegable una ruta primero, y después una de sus paradas para que sea modificada (obtenidas por AJAX). Una vez es enviada la información al servidor, el servlet (mediante la delegación de las operaciones) obtiene de la base de datos los datos de la ruta seleccionada, los cuales encapsula en la petición, como atributos en el objeto 'request'; y devuelve una nueva página, con un formulario con los datos obtenidos para que sean modificados a gusto del usuario. Una vez realizada la modificación, se procede de la misma manera que en una creación de una nueva parada, ejecutando esta vez una sentencia de modificación en la base de datos.

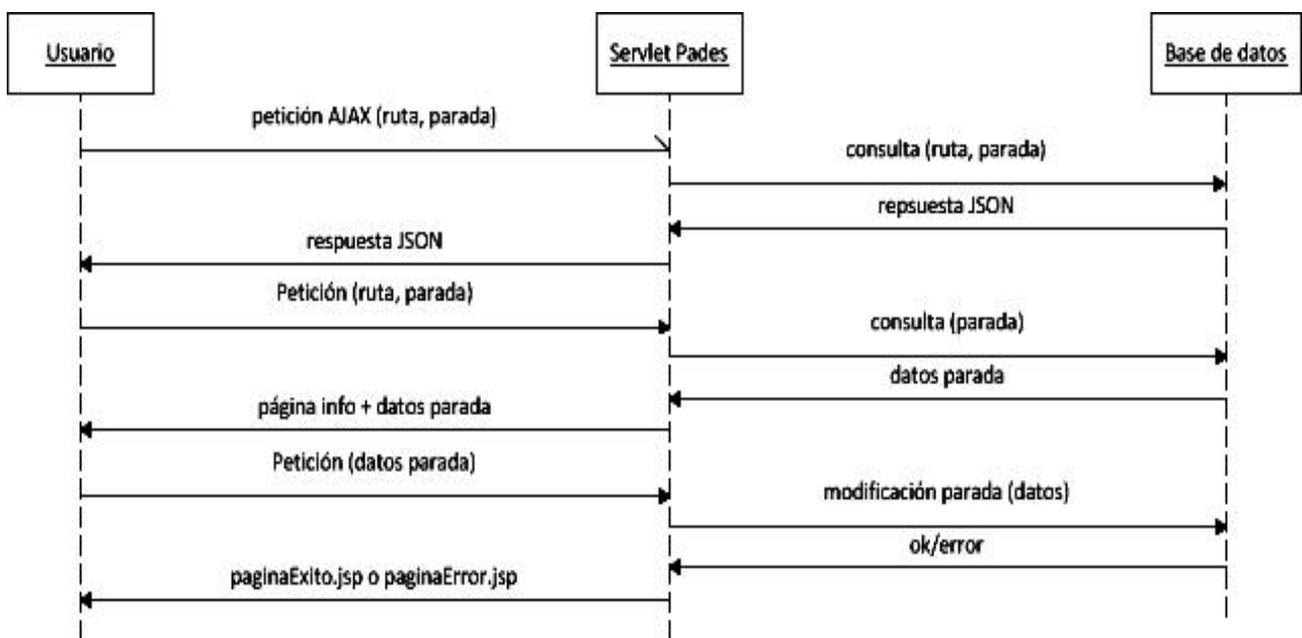


Ilustración 11. Diagrama de secuencia de modificación de parada.

#### 6.2.4.6. Eliminación de una parada

Esta operación se asemeja a la de eliminación de una ruta, seleccionando esta vez primero la ruta, y luego la parada a eliminar. Se procesa la petición en sus clases correspondientes y se devuelve la página de éxito o error según sea el caso.

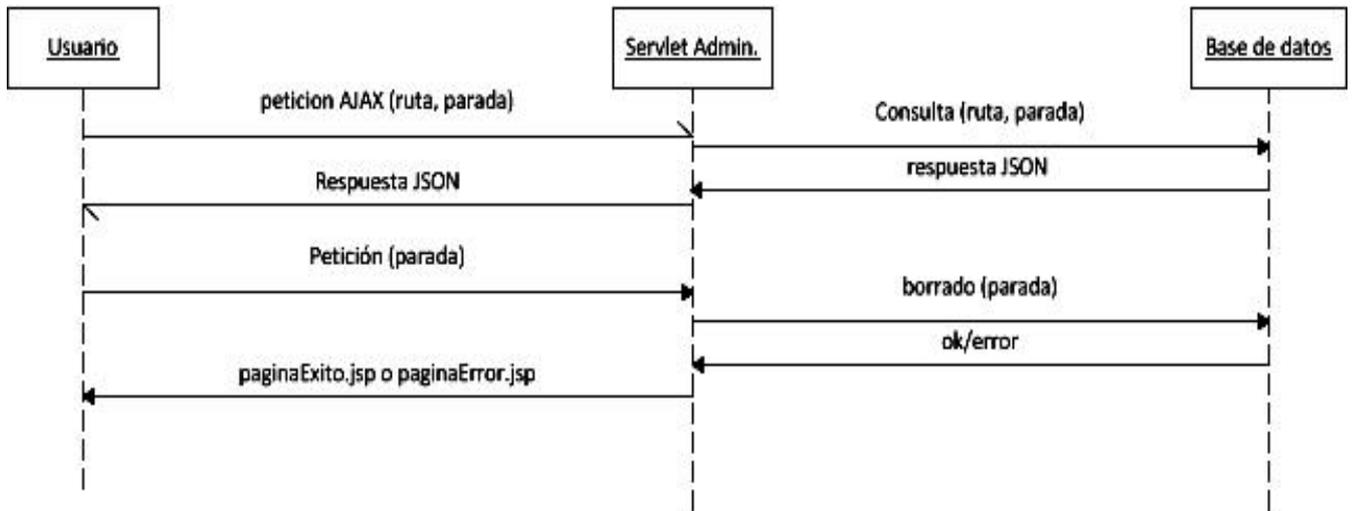


Ilustración 12. Diagrama de secuencia de eliminar parada.

#### 6.2.4.7. Alta de un nuevo responsable

Para dar de alta un nuevo usuario de que sea responsable de ruta, se debe introducir el nombre de usuario y la dirección de correo electrónico del mismo.

Dicha información se envía al servlet quien instancia a la clase 'helpResponsable' para que cree el nuevo objeto usuario con los nuevos datos, y se obtiene el identificador del usuario almacenado en la sesión. A continuación se genera una cadena aleatoria de ocho caracteres, la cual será la contraseña con la que se autenticará el responsable. Esta contraseña se almacenará cifrada, en forma de HASH codificado mediante el algoritmo SHA-1.

A continuación, se instancia la clase de operaciones y se realiza la inserción en la base de datos de la información obtenida, devolviendo la página de éxito.

Por último, se utilizan las clases del paquete 'Mail' para realizar el envío de un correo electrónico al responsable creado, comunicándole que está dado de alta en la aplicación, y se le facilita su nombre de usuario y contraseña.

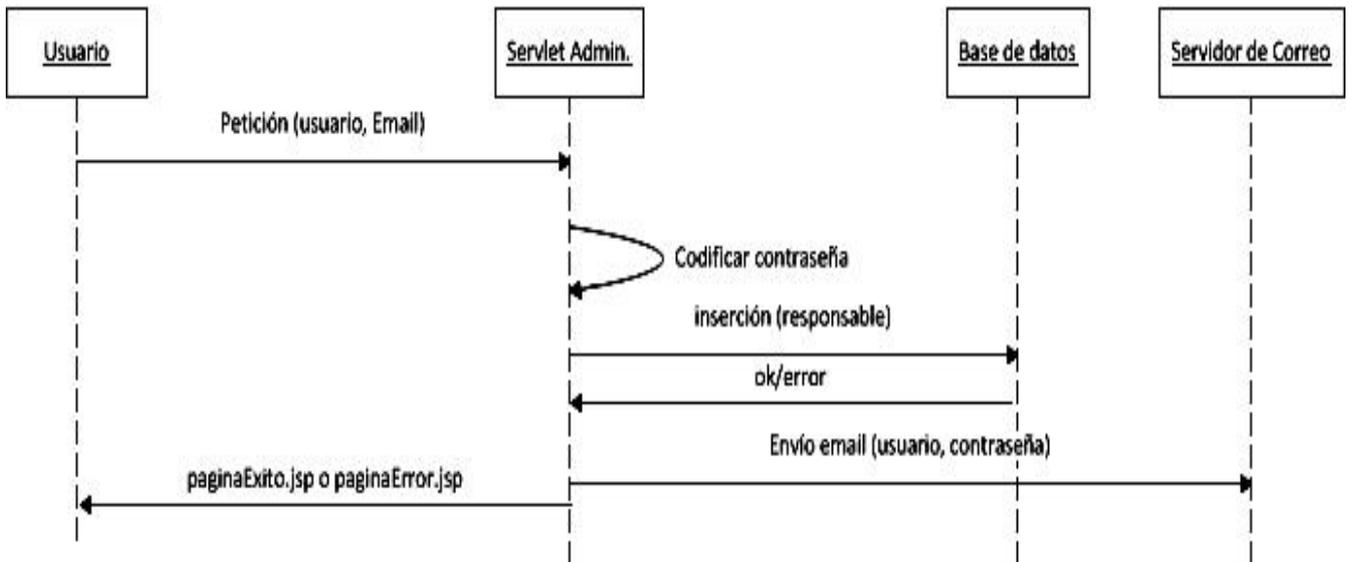


Ilustración 13. Diagrama de secuencia de alta responsable.

#### 6.2.4.8. Baja de un administrador

Las operaciones a realizar se asemejan a las del apartado 6.2.4.3, seleccionando el responsable a eliminar. Se procesa la petición en sus clases correspondientes y se devuelve la página de éxito o error según sea el caso.

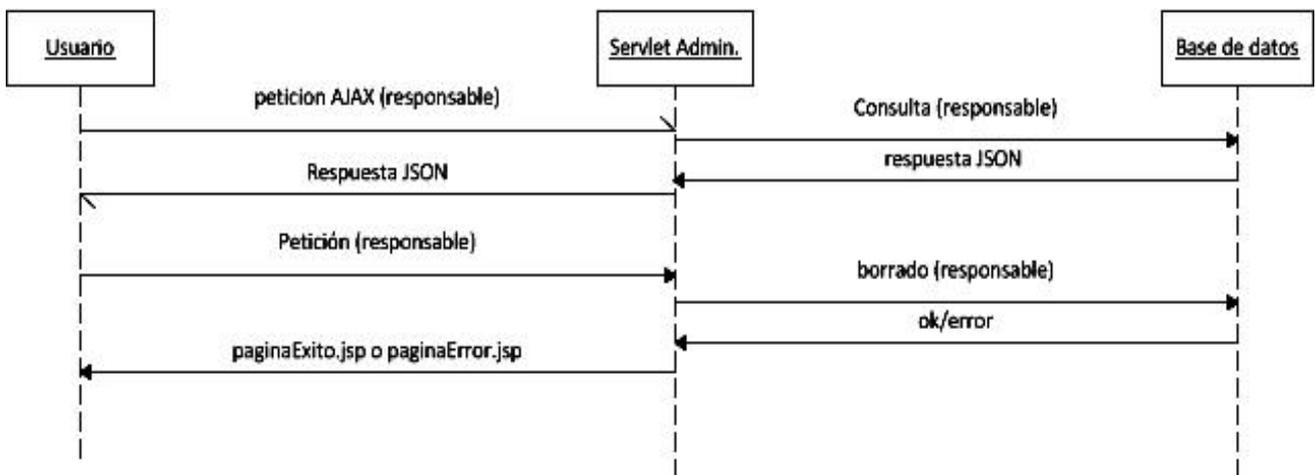


Ilustración 14. Diagrama de secuencia de baja responsable.

#### 6.2.4.9. Asignación de rutas

Si se desea asignar un responsable libre a una ruta ya existente, y desvinculando al anterior usuario de esa ruta, se debe seleccionar la ruta y el responsable.

En el caso de que el responsable a asignar, ya este asignado a otra ruta devolverá a la página de error, ya que no pueden existir dos rutas con el mismo responsable. Si no hay ningún otro error se modifica la ruta para actualizar al responsable en la base de datos y se devuelve la página de éxito

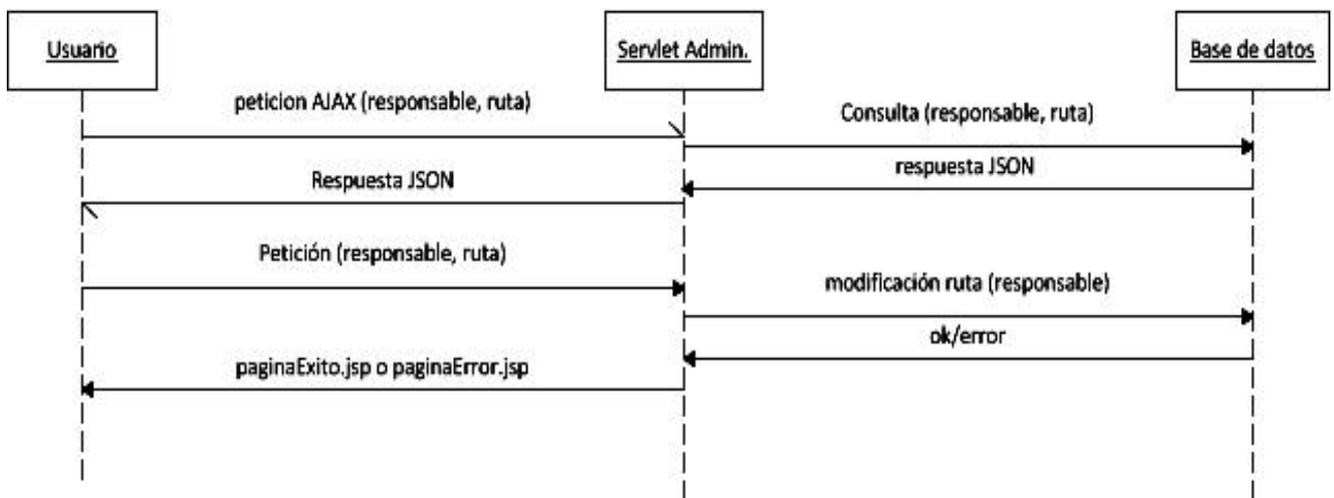


Ilustración 15. Diagrama de secuencia de asignar responsable a ruta.

#### 6.2.4.10. Alta de nuevo padre

Las operaciones a realizar son las mismas que las de creación de un nuevo responsable, diferenciando las clases en la que se delega el tratamiento de la petición.

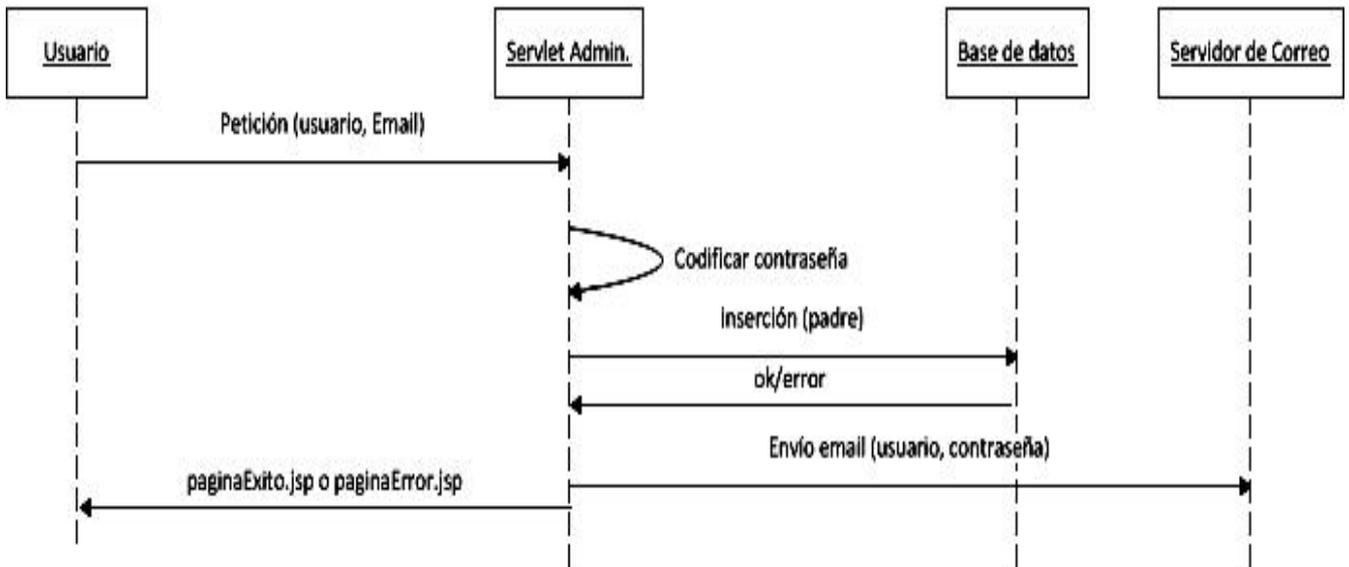


Ilustración 16. Diagrama de secuencia de alta Padre.

#### 6.2.4.11. Alta de nuevo alumno

Para dar de alta un nuevo alumno, se deben introducir sus datos personales así como indicar el usuario padre al que pertenecen. Una vez introducidos se procesa la petición, creándose el objeto alumno oportuno, y se almacena en la base de datos.

Si no hay ningún error se devuelve a la página de éxito en la operación; en caso contrario, se le redirige a la página de error.

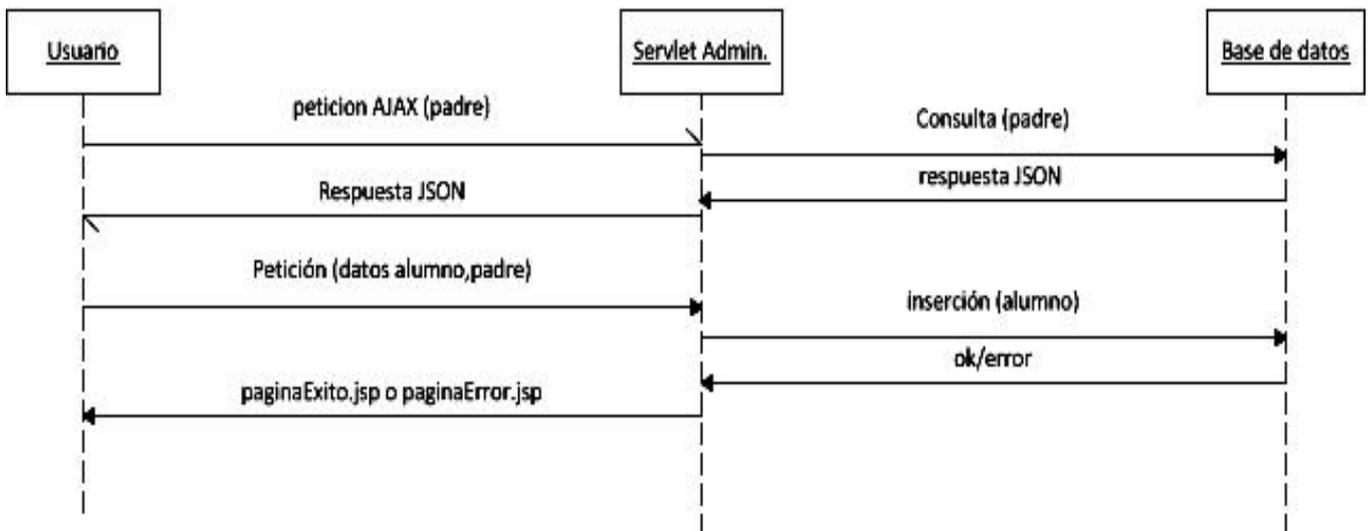


Ilustración 17. Diagrama de secuencia de añadir alumno.

En el caso de que se introduzcan los datos de los alumnos a través de un archivo de datos, la petición contendrá únicamente el archivo solicitado, y se realizarán las inserciones necesarias por cada tupla del fichero. El acceso a las clases de la aplicación es la misma y la respuesta (error o éxito) también.

#### 6.2.4.12. Lista de alumnos por ruta

Para que el usuario genere el listado de alumnos por cada ruta, se debe seleccionar la ruta en la que se desean consultar. Una vez enviada la petición, el servlet delega a 'helpAlumno' la preparación de los datos y objetos, y la clase 'alumnoOp' se encarga de realizar la consulta a la base de datos. Por cada alumno encontrado se instancia un nuevo 'Alumno' y lo almacena en una lista, que es la variable que se devuelve al servlet.

Para la visualización de esta lista, en la página JSP de retorno, se ha utilizado la *Core Tag Librarie* para poder comprobar que la lista devuelta contiene elementos, creando una tabla y, por cada elemento de la lista, crear una nueva fila con la información relativa a cada alumno. En el caso de que la lista estuviese vacía, se mostraría un mensaje indicando tal hecho.

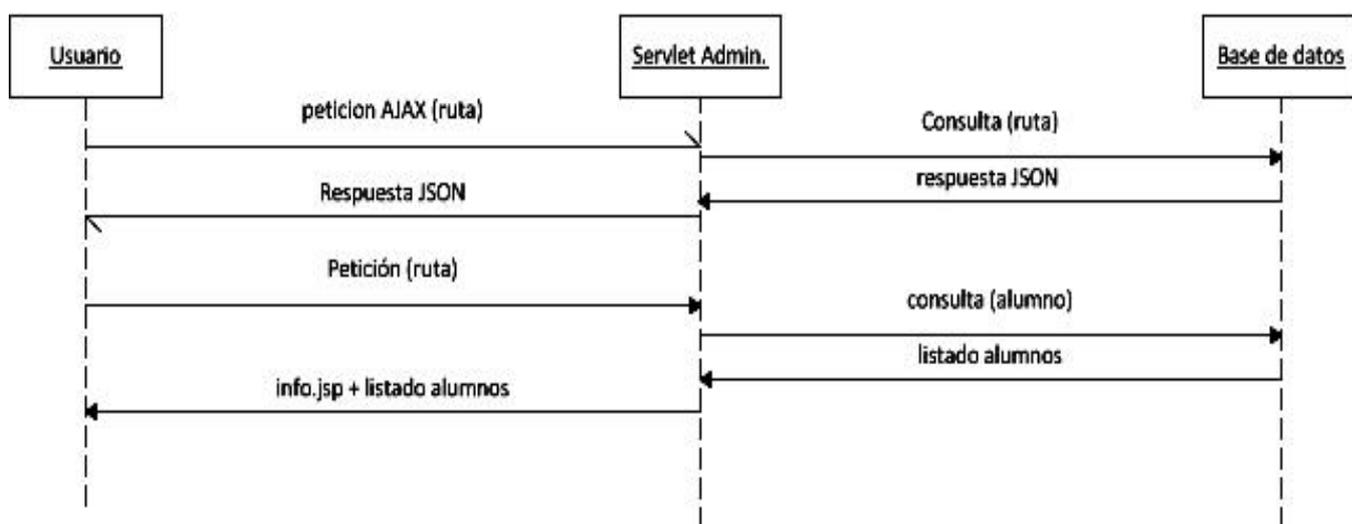


Ilustración 18. Diagrama de secuencia de lista de alumnos por ruta.

#### 6.2.4.13. Listado de alumnos por falta

Para obtener este listado, el procesamiento es similar al anterior, sólo que el parámetro solicitado al usuario es la fecha en la que desea consultar las faltas que se han

producido. Se procede de la misma forma que en el apartado anterior, y el tratamiento de la respuesta es el mismo

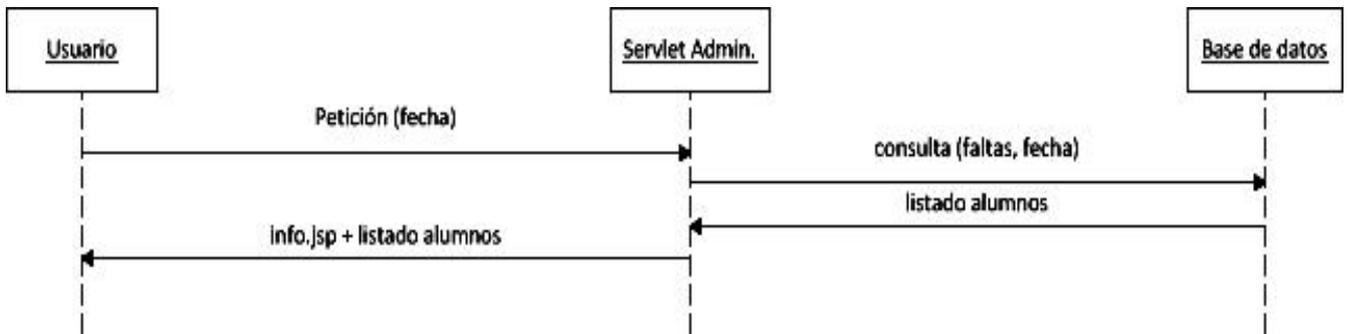


Ilustración 19. Diagrama de secuencia de lista de alumnos con faltas.

#### 6.2.4.14. Entrega de justificantes

Para dejar constancia que se ha recibido un justificante de una falta, se debe seleccionar el alumno en cuestión y la fecha de la falta a justificar, además del motivo por el que se ha producido la misma y que se ha escrito en el documento generado por la aplicación. Una vez obtenida la información, se procesa mediante el método seguido por todas las operaciones, y se realiza la modificación de la falta ya existente, añadiendo el motivo de la misma, y cambiando su estado a 'JUSTIFICADA', para que no aparezca más en los listados de faltas pendientes de justificar.

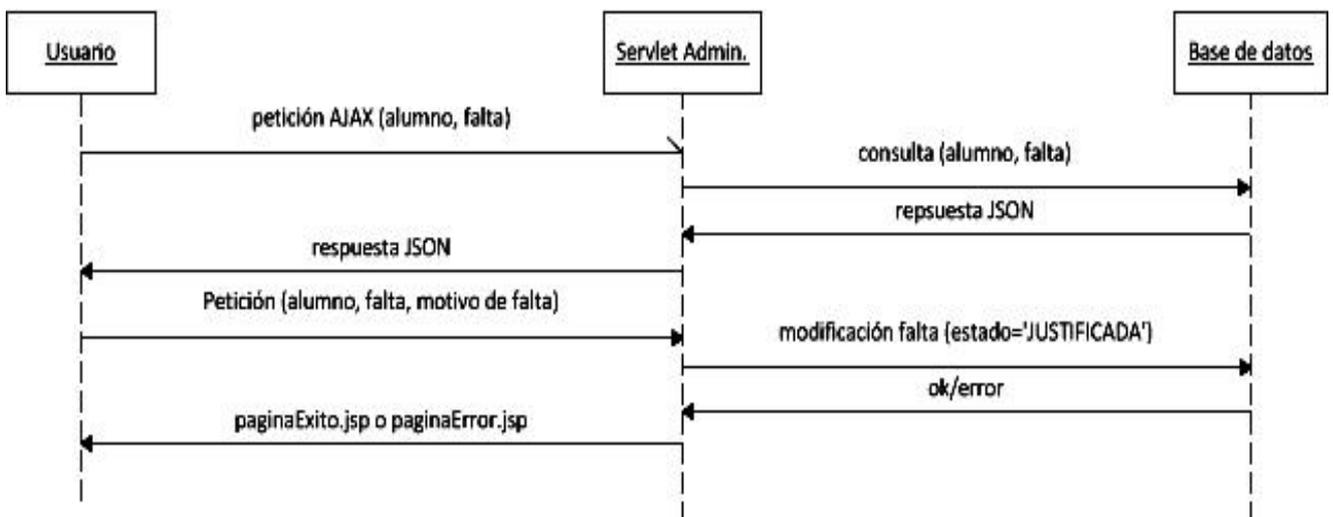


Ilustración 20. Diagrama de secuencia de entregar justificante.

#### 6.2.4.15. Recorrer ruta

Cuando los responsables de ruta han accedido al sistema, deben presionar un botón para empezar a realizar su ruta de paradas. Cuando la clase 'helpResponsable' recibe la petición, obtiene el identificador del responsable de su sesión y la fecha actual del sistema. Con estos datos y la ayuda de las clases de operaciones, se obtiene de la base de datos el identificador de la ruta y el número de paradas de la misma.

Además, comprobamos quien ha programado su falta de asistencia para este día, y actualizamos la base de datos con la información de dichos alumnos, incluyendo que el estado de la falta es 'INJSUSTIFICADA'.

Una vez realizadas estas operaciones, se prepara un listado con los alumnos que deben ser recogidos. Para ello debemos realizar una serie de consultas a la base de datos: consultar los alumnos que tienen dicha parada como parada del curso y, de cada alumno, se comprueba si han solicitado un retraso o un cambio de parada. En el caso de que se haya solicitado el retraso se indicará en el objeto 'alumno', y en caso del cambio de parada no se tendrá en cuenta a dicho alumno, ya que no será recogido en la parada inicial. Además, se consulta si ha programado una falta para ese día, de forma que tampoco se muestra en el listado dichos alumnos, debido a que no acudirán al centro escolar.

A continuación, se realiza ese mismo proceso consultando los alumnos que hayan solicitado un cambio a la parada que estamos tratando y, de la misma forma, se comprueba si se ha solicitado un retraso.

De esta serie de consultas a la base de datos obtenemos un listado con los alumnos que deben ser recogidos, y son devueltos al servlet para su tratamiento en la página JSP. Además, también devolvemos información acerca de la ruta, el responsable, el número de paradas y el número de la parada actual (estos datos se pasarían como campos ocultos para el control del recorrido de las paradas).

Con el uso de la *tag librarie*, mostramos un listado de cada alumno con dos *checkbox*, uno para indicar que el alumno ha asistido, y otro para indicar que ha faltado. En el caso de que no hubiera ningún alumno a recoger en la parada, se mostraría el mensaje de texto correspondiente.

Una vez recogidos a los alumnos en la parada, se pulsaría el botón para continuar en la siguiente parada. Cuando esta petición llega al 'servletPadre' y 'helpResponsable', comprobamos, en primer lugar, si se ha marcado que algún alumno ha faltado a la ruta; en ese caso se procede a añadir en la base de datos la información de las nuevas faltas, con su estado con valor 'INJUSTIFICADA'.

Seguidamente, se comprueba que existe una nueva parada en la ruta. En caso afirmativo, se conseguiría un nuevo listado de alumnos de la nueva parada de la misma forma que para la anterior.

Así, se procedería hasta que no haya más paradas en la ruta a mostrar. Cuando se produzca este hecho, no se sacarán más listados y se procedería a eliminar de la base de datos los alumnos que hayan programado su falta, ya que ya ha quedado reflejada como falta oficial. Por último, se devuelve al navegador del responsable el mensaje de fin de ruta y la opción de cerrar sesión, ya que no realiza más operaciones en la aplicación hasta el siguiente día lectivo.

Cuando se cierra la sesión del usuario, se invalidan todos los datos almacenados en la misma y se elimina dicha sesión del servlet.

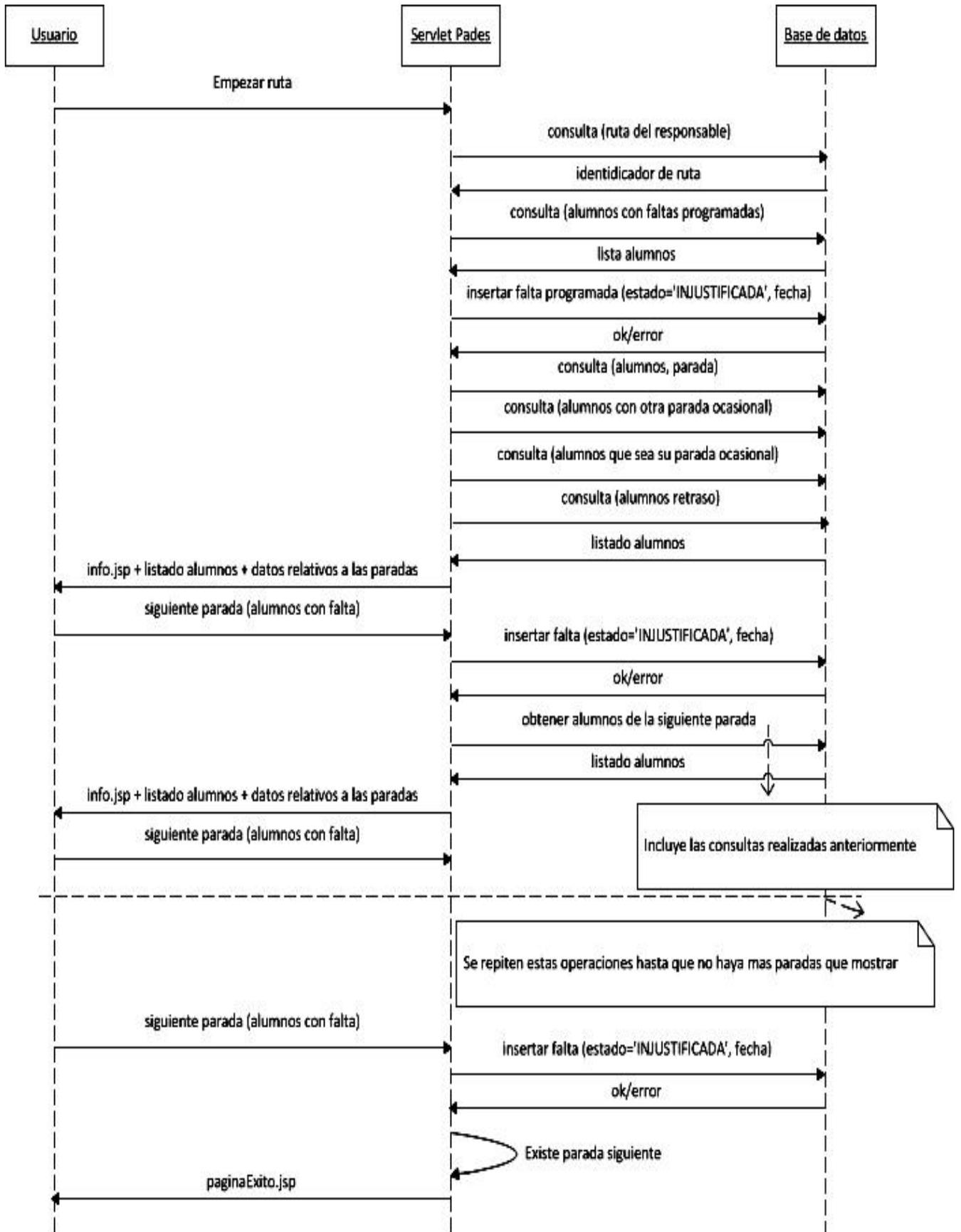


Ilustración 21. Diagrama de secuencia de recorrer ruta

6.2.4.16. Información de rutas

Para obtener el listado de rutas del sistema, se ha de elegir una ruta de la lista desplegable. A continuación, se procede de la misma forma que en el apartado 4.2.4.12, devolviendo, esta vez, un listado con la información de las paradas de la ruta. Este listado es presentado al usuario en forma de tabla con los detalles de cada parada, así como el identificador del responsable y un enlace a la información de cada parada individual.

Si el usuario selecciona el enlace a la información de la ruta, se realiza una petición al servlet con la información de la parada a mostrar, realizando las operaciones necesarias y mostrando la salida que se detalla en el siguiente apartado.

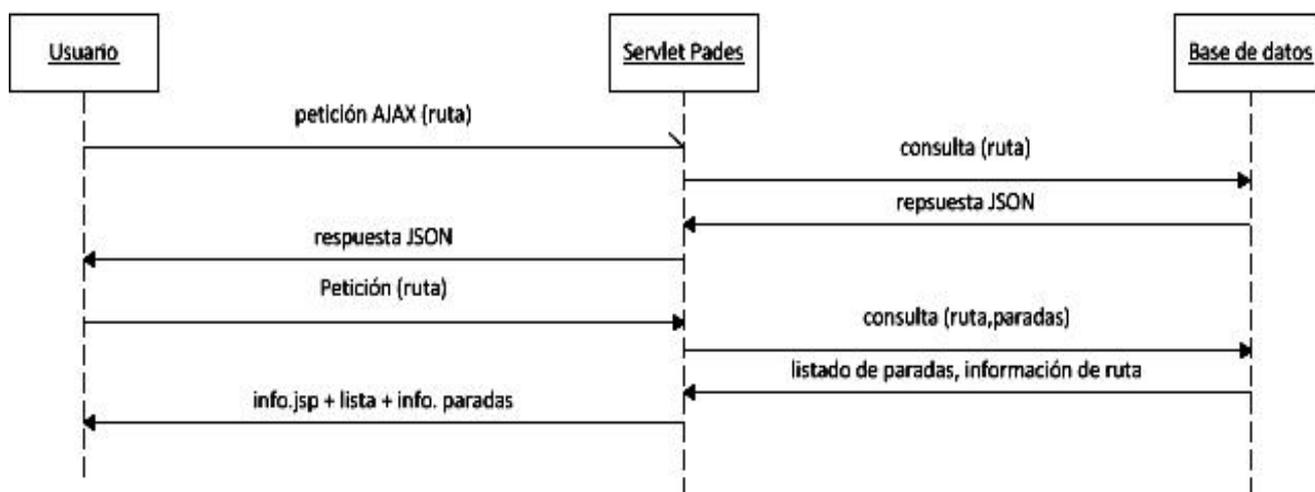


Ilustración 22. Diagrama de secuencia de información de ruta.

6.2.4.17. Información de paradas

Si el usuario desea conocer los detalles de una parada, debe elegir la ruta en la que está y el nombre de la parada en cuestión. Estos datos llegan al servidor, donde son tratados para consultar a la base de datos la información de dicha parada, que es devuelta por el servidor.

Con las mismas operaciones que en el resto de listados, se muestra al usuario una tabla con la información de la ruta y, con la utilización del script de Google Maps, se muestra un mapa con la localización real de la parada, así como un mensaje con la dirección escrita en él.

Si ha ocurrido algún error se mostraría un mensaje al usuario indicando que no es posible mostrar la información requerida.

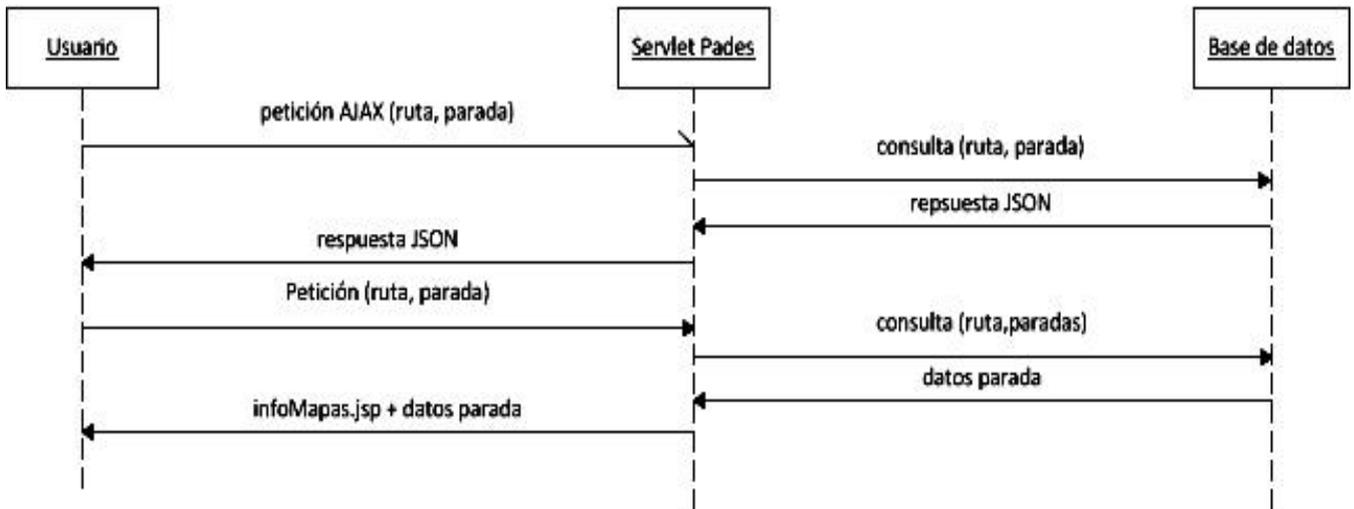


Ilustración 23. Diagrama de secuencia de información parada.

#### 6.2.4.18. Asignación de parada del curso

Para realizar dicha asignación, basta con seleccionar a un alumno, el nombre de la ruta y, a continuación, el nombre de la parada en la que el transporte escolar parará para recogerle. Una vez seleccionados los datos, se realiza la petición (mediante la pulsación de un botón), y se tramita en el servidor, añadiendo una nueva tupla en la base de datos que relacione al alumno con la para deseada.

Si no hay ningún error se devuelve a la página de éxito en la operación; en caso contrario, se le redirige a la página de error.

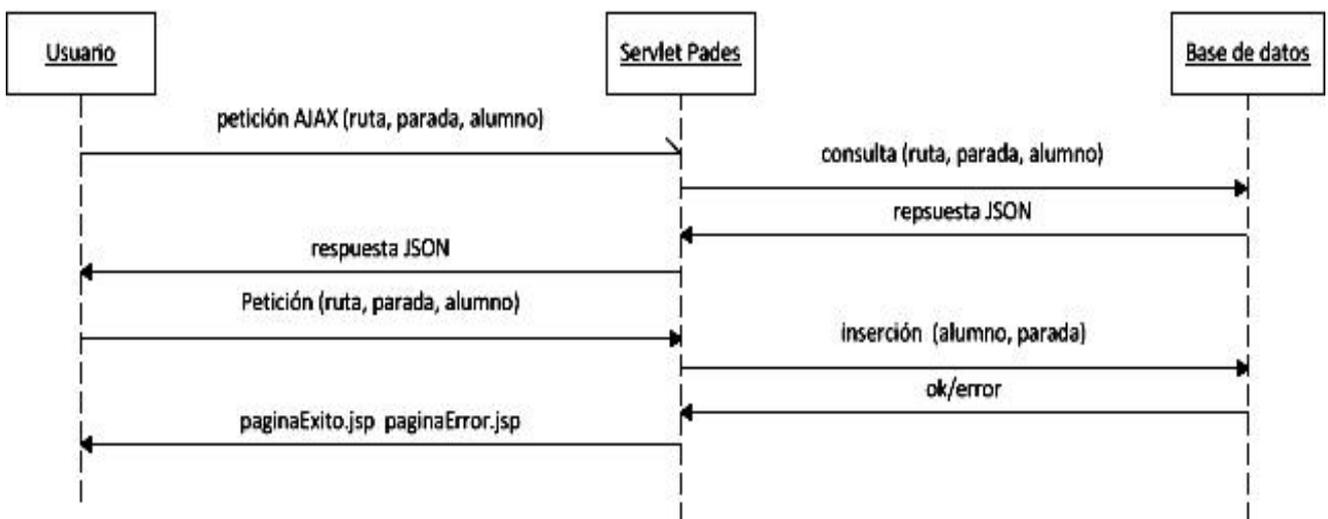


Ilustración 24. Diagrama de secuencia de parada del curso.

6.2.4.19. Asignación de parada ocasional

De la misma forma que se asigna una parada para el curso, se asigna una parada para un día determinado, recogiendo además de los datos anteriores, la fecha en la que tendrá lugar el cambio. Los datos del cambio se almacenan en la tabla correspondiente.

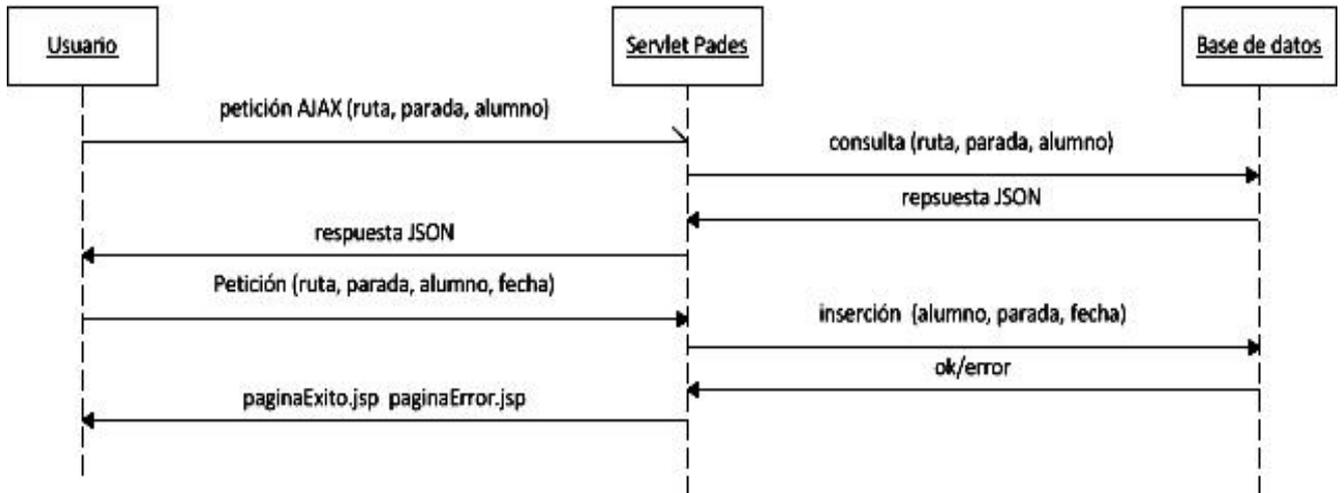


Ilustración 25. Diagrama de secuencia de parada ocasional.

6.2.4.20. Especificar retraso

Para especificar un retraso, únicamente se indica el alumno que se retrasará y la fecha en la que se producirá dicho retraso. Las peticiones se tratan de la misma manera que las anteriores, y se almacena el alumno y la fecha del retraso en la tabla correspondiente.

Si no hay ningún error se devuelve a la página de éxito en la operación; en caso contrario, se le redirige a la página de error.

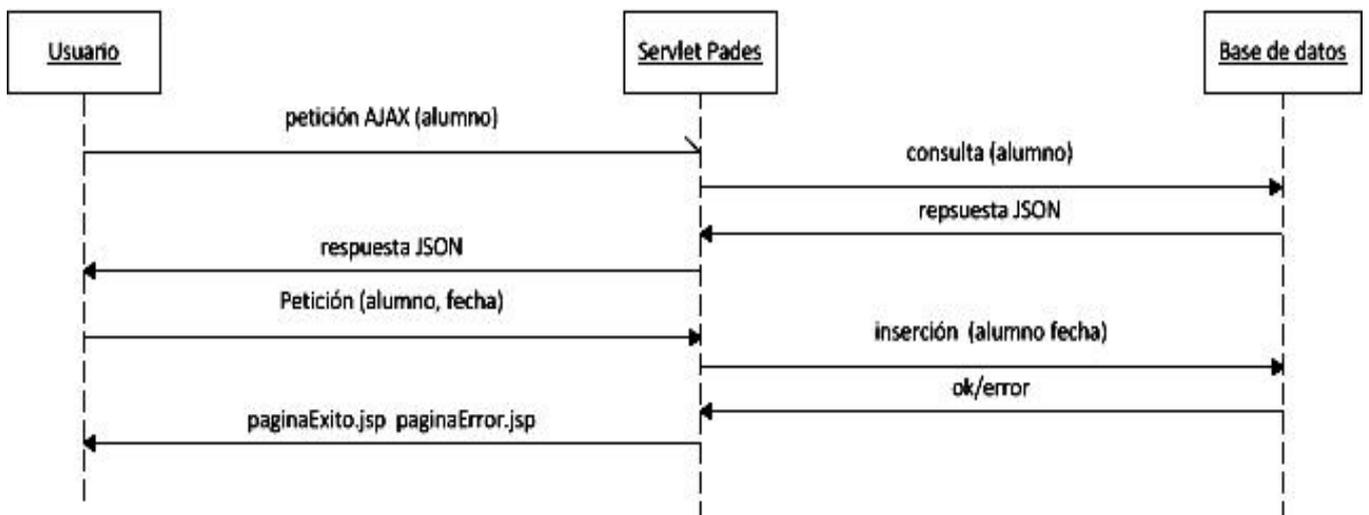


Ilustración 26. Diagrama de secuencia de solicitar retraso.

6.2.4.21. Programar una falta

Como en el apartado anterior, para programar una falta es necesario conocer el alumno que faltará y la fecha de la misma. Una vez recibida la información en el servidor, se almacena en la base de datos dicha información para ser tomada en cuenta cuando los responsables realicen las rutas.

Si no hay ningún error se devuelve a la página de éxito en la operación; en caso contrario, se le redirige a la página de error.

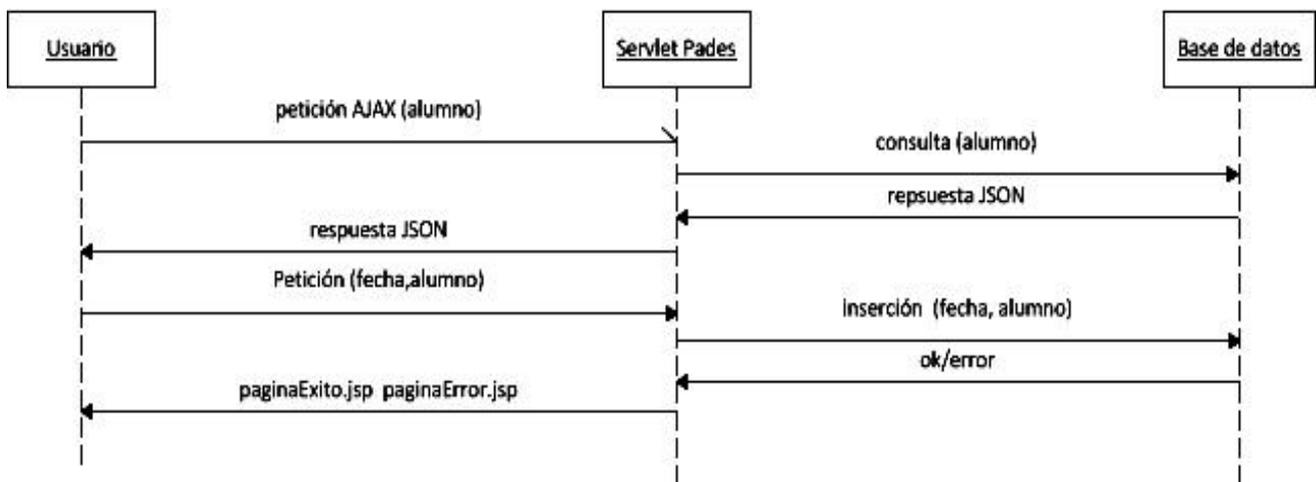


Ilustración 27. Diagrama de secuencia de programar falta.

6.2.4.22. Historial de faltas

Se procede de la misma forma que para la visualización de los listados anteriores, obteniendo como parámetro del usuario el nombre del alumno. Se devuelve al servlet un listado con todas las faltas que ha realizado el alumno, así como su estado y el motivo si esta ha sido justificada.

En el caso de todos los alumnos, se envía en la petición todos los alumnos asociados a al padre.

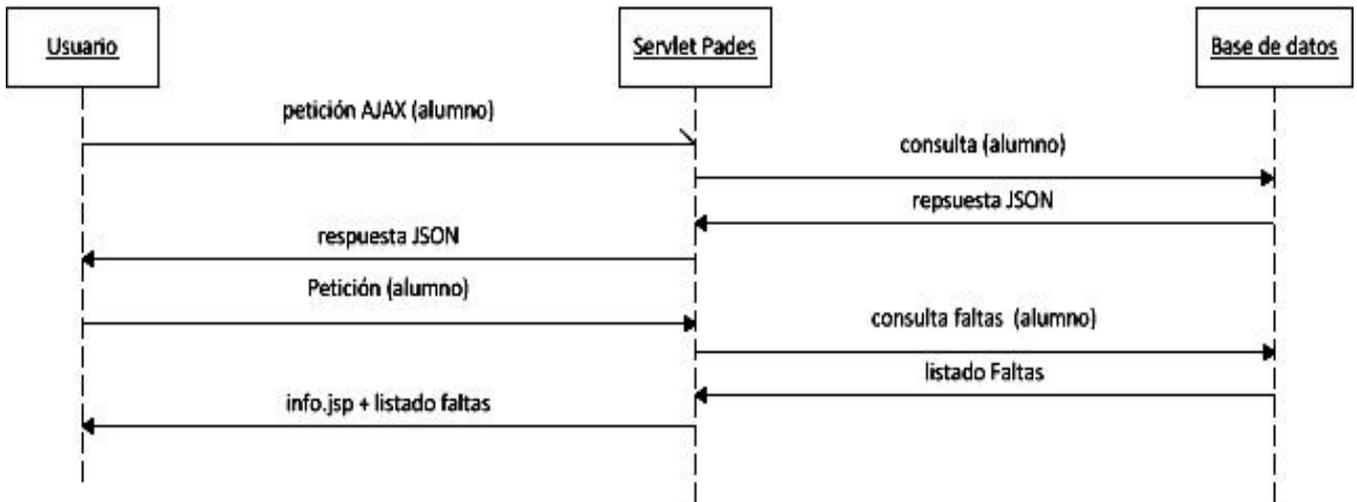


Ilustración 28. Diagrama de secuencia de historial de faltas.

#### 6.2.4.23. Impresión de justificantes

Si el usuario desea generar un justificante de una falta cometida en una determinada fecha para ser entregado en el centro, puede hacerlo eligiendo la opción del menú y rellenando dichos campos. Se le generará en el navegador un archivo, que podrá imprimir y rellenar para su entrega.

Se ha creado un formato de archivo estándar, en formato XHTML denominado 'CATEBus', que los centros escolares pueden modificar respetando la presentación que mantengan o incluyendo más información que deba aportar el alumno para justificar su falta

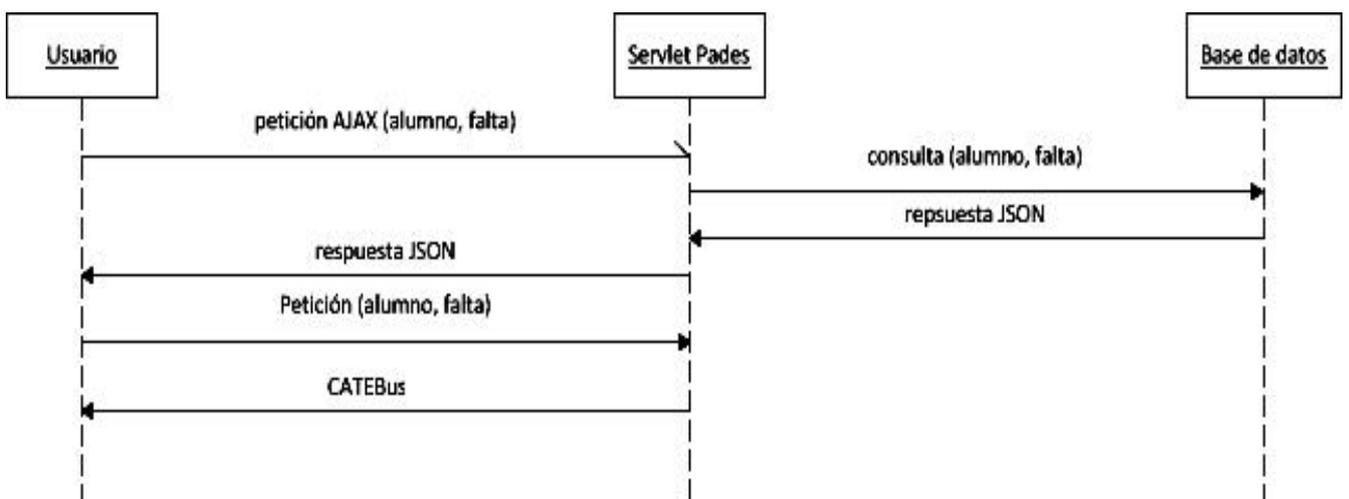


Ilustración 29. Diagrama de Secuencia de justificantes.

6.2.4.24. Cambio de contraseña

Para realizar este cambio, se necesita que el usuario introduzca su contraseña actual y la nueva contraseña almacenar. Esta petición llega al servlet, que delega su tratamiento a la clase 'padreHelp', la cual realiza las llamadas necesarias para la codificación de las dos contraseñas proporcionadas (mediante el protocolo SHA-1) y las consultas a realizar en la base de datos. En primer lugar se comprueba si la contraseña actual proporcionada coincide con la almacenada en la base de datos. En caso de que sean iguales, se procede al cambio de contraseña, mostrando la página de éxito en la operación. En caso contrario, se mostrará la página de error

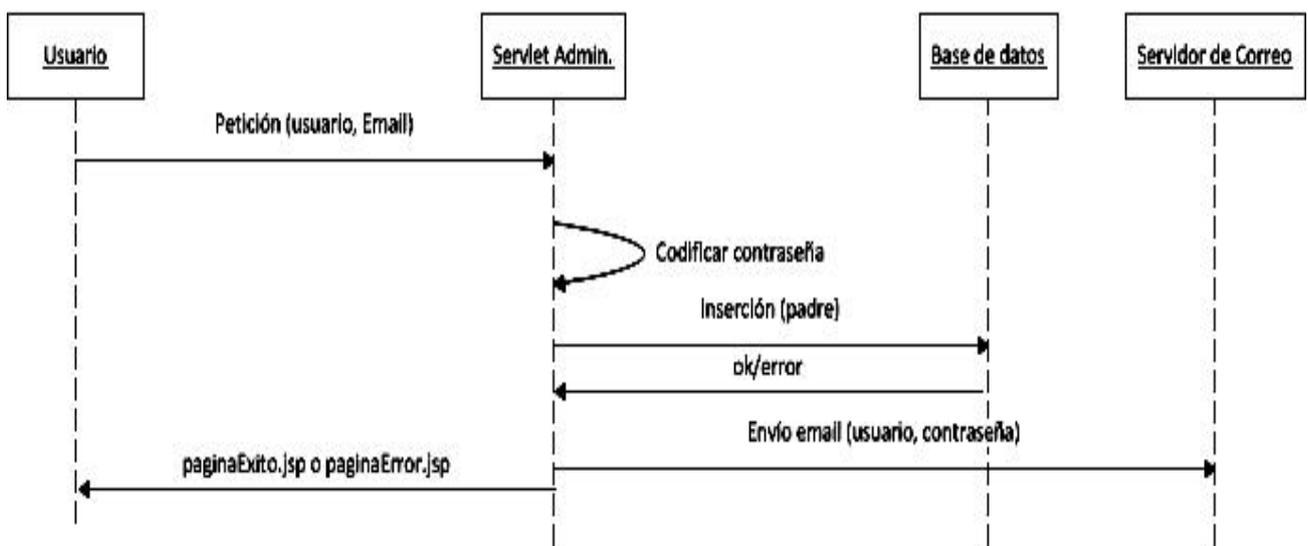


Ilustración 30. Diagrama de secuencia de cambio de contraseña

## 7. CONCLUSIONES Y TRABAJOS FUTUROS

---

En este epígrafe se realizará una evaluación del sistema construido y se comprobarán que los objetivos especificados en el apartado 4 se han cumplido. Además, se detallan las líneas de trabajo futuras que se pueden seguir con la realización de este proyecto, tanto en nuevas funcionalidades como adaptarlo a otras tecnologías o situaciones diferentes.

### 7.1. CONCLUSIONES

En primer lugar, se destaca que el desarrollo de la aplicación se ha realizado de una manera clara y progresiva. Comenzó siendo una idea en la que se tenía una serie de bases claras: una aplicación donde los padres puedan controlar o decidir en el sistema de transporte escolar contratado por el colegio o centro, pudiendo solicitar variaciones en el mismo como retrasos o paradas especiales.

Una vez obtenida la idea, se investigó acerca de las herramientas actuales que se encargan en el control de alumnos, así como en el control de asistencia de los mismos, descubriendo que existen aplicaciones o dispositivos que registran el acceso de los alumnos en el centro, pero ninguno automatizado desde el transporte escolar. En ese momento, el proyecto se amplió para abarcar esta situación y dar un primer control de asistencia y faltas de los alumnos.

A continuación, se establecieron las funcionalidades más importantes y los requisitos que la aplicación debería implementar, así como los tipos de usuarios que soportaría y dispositivos en los que se debería visualizar. Dichas especificaciones están explicadas en el punto 4 de esta memoria, y han sido ampliadas según se ido avanzando el desarrollo del proyecto, como se comentará más adelante.

Seguidamente se realizó el diseño de la base de datos que alberga los datos mantenidos por la aplicación. Se diseñaron varias opciones hasta elegir la solución más óptima para los accesos y consultas requeridas.

El siguiente paso fue el diseño de las páginas web y los ficheros de estilos CSS. Se tuvieron en consideración múltiples estilos generales de página, así como formatos de menús y visualización de datos, así como uso de imágenes u otros elementos multimedia, hasta

encontrar un diseño propio, distintivo y que sea fácilmente navegable. Una vez elegido dicho estilo, se empezó la implementación de las páginas, incluyendo cabeceras, menús, formularios, así como el espacio requerido para la visualización de los datos generados por la aplicación. Además, se realizó el archivo JavaScript de comprobación de errores, dando como resultado las páginas listas para la interacción con el servidor.

Posteriormente, se inició la construcción del controlador y el modelo de la aplicación, es decir, del servlet y las clases que comuniquen a éste con las páginas web y la base de datos, que realizan las operaciones indicadas por el usuario. En primer lugar, se implementaron las funcionalidades del personal del centro escolar, con su conjunto de pruebas y cambios asociados a éstas. A continuación se realizó la implementación de las funcionalidades a realizar por el tipo de usuario padre, y por último, las de los responsables de ruta.

Además, se ha realizado unas páginas específicas para ser visualizadas en dispositivos móviles, que presentan algunas de las funcionalidades accesibles para los padres de los alumnos y del personal docente del centro escolar.

Cabe destacar que, mientras se realizaban los requisitos de la aplicación, búsqueda de herramientas, así como el diseño e implementación de la base de datos, se ha ido redactando la documentación relacionada con dichos aspectos, de forma que se ha quedado reflejado en esta memoria cada uno de los pasos que se han seguido, ayudando posteriormente a la localización y resolución de problemas (por ejemplo, datos faltantes en la base de datos, o datos definidos pero no utilizados por la aplicación).

Sin embargo, se han encontrado numerosas dificultades durante la realización de este proyecto, respecto a la implementación de ciertas partes, las cuales se explican a continuación:

Una de las dificultades tratadas ha sido la realización de algunos mapas mediante la API de Google Maps, concretamente la visualización de polilíneas, de forma de que dos o más localizaciones se muestren unidas mediante una línea recta. El primer problema encontrado es que, al necesitar Google coordenadas geográficas, no se realizaba correctamente la transformación coordenada-dirección, de forma en que se mostraba el mapa, pero no las localizaciones ni la línea de las debería unir.

También, se encontraron problemas respecto a donde se debe centrar el mapa, dato necesario para su visualización, y que se debe especificar desde un inicio (en formato de coordenada geográfica). Al ser varias localizaciones, se encontraron problemas de

visualización del mapa, ya que dependiendo de dónde se centrara, podrían quedar paradas sin mostrar, obligando al usuario a desplazarse por el mapa.

Debido a estos problemas, no ha podido ser posible presentar un mapa de polilíneas en la aplicación. Sin embargo, se ha incluido un mapa en el que se representan todas las paradas de cada ruta. La visualización se encentra en la última parada de la ruta, pero se puede arrastrar el mapa, así como cambiar el nivel de zoom y el tipo de mapa, para que los usuarios puedan contemplar las paradas de la ruta de la forma más cómoda para ellos.

También se han encontrado dificultades a la hora de generar un justificante para el usuario. En un primer momento se consideró en presentar un archivo PDF al usuario para su descarga, con los datos del alumno, así como la fecha de la falta. Sin embargo, no pudo ser posible su realización, debido a que las herramientas y librerías consultadas necesitan licencia de pago, y no se ha encontrado ninguna de libre distribución que maneje los archivos PDF internamente de forma correcta.

A continuación se pensó en crear el archivo directamente la aplicación, pero la complejidad del formato, así como el uso de stream de datos en la mayoría de los documentos PDF para almacenar la información, hace casi imposible esta opción. También se pensó en otros formatos para la descarga del documento, como DOC, DOCX o RTF, pero no proporcionaban los resultados deseados, o su tratamiento era igual de complicado o peor. Por ello, se adoptó la solución de crear un documento XHTML, que los padres podrán imprimir y rellenarlo, adjuntando la documentación correspondiente. Finalmente, se ha conseguido que la aplicación genere un documento PDF, para que sea imprimido y cumplimentado.

Además, de esta forma no obligamos a los centros escolares el uso de este justificante, sino que los pueden sustituir por los suyos propios que estén utilizando.

### **7.1.1. Comprobación de cumplimiento de los requisitos**

En este epígrafe se comprobarán que se han cumplido los requisitos y especificaciones que se plantearon en su inicio, así como se justificarán las funcionalidades desechadas o aquellas que se han incorporado a posteriori.

En primer lugar cabe destacar que se ha realizado cada una de las funcionalidades descritas en los apartados iniciales, las cuales son fácilmente localizables en la aplicación, y ofrecen los resultados esperados.

A continuación se presentan los requisitos y funcionalidades que se definieron en un principio y que han sufrido cambios en su implementación en la aplicación. Estos cambios se han producido debido a que se ha determinado la ampliación del ámbito de la aplicación en algunos aspectos o de las peticiones del tutor del proyecto:

- **Acceso a la aplicación:** como se especificó, el acceso a la aplicación se realiza mediante un formulario que valida los datos almacenados en el sistema. Si bien, se ha realizado una aplicación que utiliza un sistema de cifrado de las claves almacenadas en el sistema, mediante el algoritmo SHA-1, de forma que esta información sea segura y muy difícilmente se vea comprometida.
- **Información de paradas y rutas:** se ha añadido al requisito la visualización gráfica de las paradas mediante Google Maps, tanto en la versión normal como en la versión móvil de la aplicación, para una mejor localización de las mismas.
- **Alta de responsables:** se han añadido funcionalidades en la implementación de esta opción, de forma de que la clave no sea introducida por el usuario, sino que será generada por la aplicación y enviada a su dirección de correo electrónico. Las claves generadas serán almacenadas de forma encriptada, siguiendo el procedimiento explicado anteriormente.
- **Alta de padres y alumnos:** el personal del centro será el encargado de dar el alta de los padres y sus respectivos alumnos. Para dar de alta a los padres, se seguirá el mismo procedimiento que en el caso anterior. En cambio, para dar de alta a un alumno, basta con introducir sus datos personales y asociarle con un padre.
- **Alta de alumnos personalizada:** el personal del centro podrá elegir si introducirá los alumnos de uno en uno o de forma automatizada desde un fichero CSV que se subirá a la aplicación.
- **Cambio de clave:** Se ha añadido esta funcionalidad para los padres de los alumnos, de forma que puedan cambiar la contraseña generada a una nueva elegida por ellos. Con este añadido ofrecemos posibilidades de personalización y accesibilidad.
- **Acceso móvil:** debido a su complejidad, en lugar de preparar toda la aplicación para su visualización en un dispositivo móvil, se ha implementado una serie de funcionalidades específicas para su visualización en estos dispositivos, que no soportan las tecnologías implementadas en la versión normal (tales como JavaScript, Ajax, mapas dinámicos, etc.). Además, en un primer momento, se decidió que únicamente los usuarios padres dispondrían de este acceso. Durante el transcurso de la realización del proyecto, se consideraron otras funcionalidades del personal docente que serían necesarias dotarlas de accesibilidad móvil.

- **Comprobación de datos:** se ha implementado una serie de funciones JavaScript que permiten validar el formato de los datos introducidos en los formularios antes de ser enviados al servidor. De esta forma se libera de carga al mismo, y la comprobación es más rápida y cómoda para el usuario.
- **Justificantes de Faltas:** en un primer lugar se planteó la posibilidad de realizarlos en un formato estándar de Office, pero su desarrollo planteaba muchos problemas. Seguidamente, se optó por realizar un documento imprimible en formato XHTML. Finalmente, se ha conseguido que la aplicación genere un documento PDF, para que sea impreso y cumplimentado.

### 7.1.2. Ventajas e inconvenientes de uso

La principal ventaja de uso de la aplicación, es poder tener un control informatizado de las rutas escolares y los alumnos que la utilizan, así como controlar directamente las faltas de asistencia de los alumnos que utilizan dichas rutas. De esta forma, el personal del centro tiene un mejor control del manejo de este sistema de transporte y le facilita, en parte, la contabilización de las faltas de asistencia.

Para los padres de los alumnos, supone una herramienta por la cual podrán contactar con el centro de forma directa para solicitar su parada, así como para recogidas excepcionales o retrasos a la hora de llevar a los alumnos a la parada. Ésta última opción puede resultar de gran utilidad y puede ahorrar a los padres el llevar a los hijos, primero a la parada, y segundo al centro escolar, perdiendo tiempo y llegando tarde a sus puestos de trabajo, por ejemplo.

Además, permitiendo programar las faltas de asistencia, se ahorra tiempo y esfuerzo al centro educativo en contabilizar dicha falta y en su recogida por parte de la ruta escolar.

Para los responsables de las rutas, permite la sustitución de las listas de papel, que no se modifican desde el inicio de curso, a lista personalizables a cada responsable, con los alumnos a recoger y los cambios actualizados cada día.

Además, pensando en los padres y en que sea utilizada por personas de todas las edades, la interfaz de presentación de las páginas es sencilla, fácil de entender, con un menú desplegable sencillo de utilizar en el que las opciones se ven claramente, y donde las acciones a realizar se ejecutan en uno o dos pasos, introduciendo los datos necesarios y a pocos clics de la respuesta requerida.

Por otro lado, esta aplicación está realizada siguiendo las premisas de la orientación a objetos, de forma que el código está modulado y estructurado, de forma en la que realizar modificaciones o incluir nuevos módulos o funcionalidades se realicen de forma sencilla; permitiendo, además, la reutilización de código. Este proyecto no sustituye a otras aplicaciones de control y gestión del alumnado, sino que está concebida como un extra y un apoyo a estas herramientas, incluyendo funcionalidades y necesidades no contempladas por estas.

Igualmente, el hecho de poder solicitar retrasos o consultar la dirección de la parada de un alumno mediante un móvil o un dispositivo de similares características, posibilitan una mejor accesibilidad de la aplicación, y una prestación añadida a los padres que la utilicen.

Pero el uso de esta aplicación también puede presentar alguna desventaja o problema a solucionar. Uno de estos problemas es el abuso en los retrasos de los alumnos en llegar a la parada de la ruta. Los padres pueden aprovechar esta funcionalidad para retrasar indiscriminadamente la llegada de los alumnos al centro, solicitando todos los días retrasos injustificados para que la ruta les espere. Se debería realizar políticas de buen uso de esta opción, penalizando a aquéllos usuarios que soliciten varios retrasos en una determinada cantidad de tiempo, o que no les permitan solicitarlos si lo han hecho ya una vez anteriormente y, aun así, no han acudido a la parada a tiempo.

También hay que tener en consideración acerca de este hecho, que puede que, aunque el alumno no haya viajado a la ruta en el transporte solicitado, pueda llegar al centro escolar por otros medios, de modo que la falta originada por el responsable de ruta no tenga validez. Esto se puede solucionar de diversas formas, como por ejemplo justificar la falta comunicando que ese día se usó otro medio de transporte para llevar al alumno, o adjuntar algún módulo a la aplicación que pueda contemplar este caso.

## 7.2. TRABAJOS FUTUROS

Con la realización de esta aplicación web, se abre la puerta a una serie de ampliaciones del mismo para añadir funcionalidades, así como ampliar a nuevos tipos de usuarios o de situaciones en las que este tipo de aplicaciones se pueden utilizar.

### 7.2.1. Creación de consultas, listados y tratamiento de datos avanzados

Si las consultas ofrecidas son insuficientes para los centros escolares donde puede ser implementada y desplegada esta aplicación, se pueden añadir nuevas búsquedas más exhaustivas, así como la elaboración de nuevos listados más complejos, tales como historiales de faltas por cursos académicos, organización por cursos y clases. Por otro lado, si añadimos la información de los profesores de cada aula, se podrían consultar los listados de los alumnos y, en el caso de que hayan perdido la ruta, pero hayan asistido a clase, poder eliminar la falta generada de forma automática.

Además, si se ampliasen los datos almacenados del alumno a otros como su dirección postal, se podrían realizar algoritmos que sugieran directamente las paradas más próximas a su dirección o su localidad. Asimismo, gracias al uso de Google Maps, se podrían desarrollar nuevos mapas que siguiesen las paradas de una ruta, o incluso que se generasen las indicaciones que permitiesen al usuario conocer cómo llegar a la parada más cercana.

### 7.2.2. Creación de una aplicación basada en Android o para iPhone

Con la aparición de los nuevos dispositivos móviles, cada vez se están desarrollando nuevos lenguajes de programación y entornos adaptados para ellos. Una línea de trabajo futura sería la adaptación de la aplicación web a una aplicación para móviles y dispositivos Android. De esta manera tendrían un acceso más rápido a la misma, sin necesidad de estar buscando la página web y accediendo desde el propio navegador, que puede no ser totalmente compatible.

Por otro lado, Con la liberación de su comercialización, cada vez es más extendido el uso de móviles Apple en el mercado. Éstos móviles cuentan con una gran capacidad de procesamiento y son compatibles con muchas de los paradigmas de programación existentes. Con la creación de una aplicación para ellos, optimizamos los recursos ofrecidos por el dispositivo y damos servicio a un grupo de usuarios en alza.

También se podría ampliar las adaptaciones a teléfonos con otros sistemas operativos Windows Phone o con Symbian.

### 7.2.3. Comunicación vía SMS con los padres

Debido al uso masivo de la telefonía móvil, se podría avisar a los padres de los alumnos de las incidencias que hayan ocurrido, tales como la falta de asistencia del alumno en la ruta, o la confirmación de la justificación de la falta. Además, se les podría indicar cualquier incidencia que haya ocurrido en el centro educativo, tales como la falta de profesores o incidencias producidas por los alumnos, incluso las últimas noticias del centro o de la aplicación.

Además, los padres podrían solicitar el retraso en llegar a la parada a través de mensajes de texto, de forma más rápida y cómoda para los padres, pero costosa, ya que acarrearía un coste con la compañía operadora del usuario. Además, para poder establecer este servicio, se tendría que contratar un número al que lleguen los mensajes y redirigir las respuestas a nuestra aplicación. Con el número del teléfono del solicitante (dato nuevo que se tendría que mantener en la base de datos), y la palabra clave o cuerpo de mensaje identificativo, que se tendría que establecer por defecto y dar a conocimiento de los usuarios de la aplicación, se podría asignar a los alumnos un retraso en la parada establecida.

### 7.2.4. Inclusión de un servicio de localización del autobús

Cada vez aumentan las aplicaciones que establecen un uso de los dispositivos que soportan la geolocalización, que permite conocer la situación de una persona u objeto en unas determinadas coordenadas terrestres. Empresas como Google (gracias a Google Earth por ejemplo), son capaces de prestar la funcionalidad de situar geográficamente entidades. Si esta tecnología fuera aplicada a nuestro sistema web, se podría indiciar en qué parada se encuentra el autobús en todo momento, o si se encuentra circulando entre paradas. Esta información sería del todo útil, no sólo a los padres (que podrían saber el tiempo que tienen para preparar al alumno para coger la ruta), sino también al personal docente, que tendría localizado al autobús en todo momento, y podrían adelantarse a posibles retrasos del mismo.

### 7.2.5. Cobertura a centros de trabajo

Debido a que no sólo se usa un sistema de transporte externo para acceder a los centros escolares, sino que muchas empresas poseen un sistema de transporte que permite el desplazamiento de los trabajadores desde sus hogares al puesto de trabajo y viceversa (en los últimos tiempos se está fomentando este sistema para, de esta manera, fomentar el ahorro energético y reducir la contaminación y atascos en las grandes ciudades). Por ello, esta

aplicación se podría actualizar o reinterpretar como una aplicación de control de transporte empresarial.

Para ello, cada trabajador obtendría una cuenta de usuario para poder localizar las paradas de la empresa y elegir en cual quiere que se le recoja, al igual que ocurre con los alumnos. La parte de administrador recogería esa información y mantendría informados a los trabajadores de las faltas a las rutas, así como de las modificaciones en las mismas o de avisos o noticias relacionadas con su actividad laboral. Por último, sería necesario un responsable de la ruta, que realice las operaciones de confirmación de asistencia a la cada una de las rutas de transporte, característica con la que no cuentan la mayoría de las empresas que emplean este servicio, que simplemente pone a disposición de los trabajadores un autobús especial regulado por el propio conductor, mientras que los centros educativos tienen a personal que se encarga del control de acceso de los alumnos a los autocares. Es más, en muchas ocasiones son los mismos trabajadores los que indican al conductor donde tienen que recogerles o dejarles.

#### **7.2.6. Informes personalizados**

En la generación del justificante, se ha incluido un justificante estándar para que los padres puedan rellenarlo y los alumnos lo entreguen en el centro escolar. Debido a que cada centro se rige por unas normas de presentación distintas, así como de unos estilos propios, la aplicación deja se preparada para poder afrontar dichos cambios, así como para proceder a la generación de nuevos informes descargables, tales como listados de alumnos, o listados finales de faltas, para el control de las mismas en el centro escolar. La simplicidad de la información de la aplicación permite que se impriman los resultados por pantalla de forma fácil y concisa, con lo que se podrían obtener los mismos en formatos de texto, para la posterior inclusión de datos, o en formato PDF para la impresión directa de los mismos.

## 8. ESTIMACIÓN Y PLANIFICACIÓN DEL PROYECTO

---

En este capítulo se procederá a explicar y detallar las estimaciones y planificaciones que se han realizado para la realización de este proyecto. Para ello, se van a aportar tanto datos numéricos como datos gráficos para un correcto análisis e interpretación de los mismos.

A partir de estos datos se van a estimar los valores esenciales para el correcto desarrollo del proyecto, tales como el coste del proyecto, el esfuerzo requerido (medido en meses/hombre), esfuerzo requerido en cada una de las fases, etc. Además se dará una planificación en el tiempo de la duración de cada una de estas fases y, por consiguiente, del proyecto total.

Asimismo, se ofrecerá el ciclo de vida utilizado para el desarrollo del proyecto.

### 8.1. CICLO DE VIDA

El ciclo de vida utilizado para la realización de este proyecto ha sido un ciclo de vida en Espiral Prototipado, ya que es el que más se ajusta a las necesidades de desarrollo. Esto se debe tanto a las distintas fases en las que se ha dividido los objetivos y le desarrollo del mismo, como las distintas entregas que se ha ido realizando para comprobar tanto su correcto funcionamiento como su aceptación.

Gracias a este modelo de ciclo de vida se puede presentar al cliente distintos prototipos en los que se refleja la evolución del proyecto, y de esta forma involucrar más al cliente en la creación de la aplicación, como por ejemplo la detección de errores de análisis o añadir y modificar funcionalidades. El último prototipo realizado consistirá la aplicación final.

A continuación se muestra una representación gráfica del mismo.

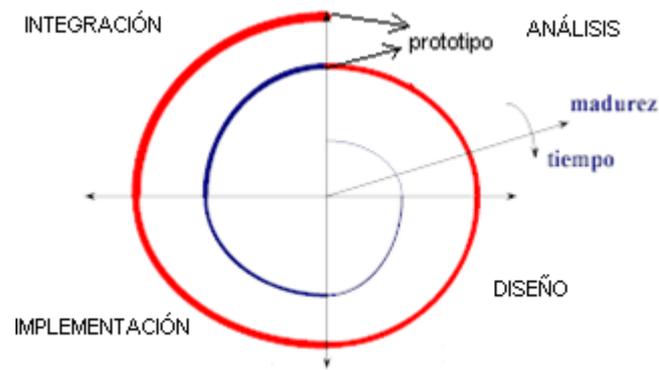


Ilustración 31. Ciclo de vida

En cada iteración, compuesta por cuatro fases: análisis, diseño, implementación e integración y pruebas, se va realizando una funcionalidad distinta que se añade a las que ya se encuentran implementadas. En este proyecto se han realizado tres iteraciones, desarrollando en cada una de ellas las funcionalidades referentes al personal del centro, a los padres y a los responsables de las rutas. Así mismo, se han realizado una nueva iteración más al término del proyecto, para para la visualización en móviles de las funcionalidades para el personal del centro.

## 8.2. ESTIMACIÓN DEL PROYECTO

Para la realización de la estimación del proyecto se ha utilizado el modelo COCOMO II [14], que es uno de los modelos más utilizados profesionalmente para la realización de estas operaciones.

Para la utilización de este modelo se han de utilizar una serie de parámetros de entrada para producir los resultados requeridos. Estos parámetros tratan acerca de la complejidad de la aplicación, la experiencia de los desarrolladores, así como otros factores relevantes para el proyecto. Estos parámetros se dividen en dos grupos: multiplicadores de esfuerzo y factores de escala.

### Multiplicadores de esfuerzo (EAF)

Tabla 2. Multiplicadores de Esfuerzo

Multiplicador	Valor	Descripción
RELY	Alto	Fiabilidad alta, pérdidas recuperables.
DATA	Alto	Alto volumen de datos que se manejan, así como múltiples accesos a bases de datos.
DOCU	Alto	Valor alto referente a la documentación del proyecto.
CPLX	Nominal	La complejidad nominal: paso de mensajes simples, operaciones básicas.
RUSE	Alto	El nivel de reutilización de la aplicación es alto, pudiéndose ampliar fácilmente a nuevas características.
TIME	Alto	Alta restricción del tiempo de ejecución, sesiones de usuario establecidas en el tiempo.
STOR	Nominal	Las necesidades de memoria son bajas, ya que no se almacenan grandes cantidades de datos en memoria.
PVOL	Bajo	No se especifica ningún cambio en la plataforma del sistema.
ACAP	Nominal	Capacidad normal del analista.
AEXP	Nominal	Experiencia del analista normal.
PCAP	Nominal	Alta capacidad del programador.
PEXP	Nominal	Experiencia normal del programador.
LTEX	Nominal	Experiencia normal en la plataforma de programación.
PCON	Muy Alto	El personal se mantendrá constante.
TOOL	Alto	Integridad de la aplicación moderada y uso de un ciclo de vida maduro.
SITE	Muy Alto	Desarrollo multi-lugar, comunicaciones mediante banda ancha, preparado para varias plataformas y dispositivos

### Factores de Escala (EF)

Tabla 3. Factores de Escala COCOMOII

PREC	Nominal	Se han desarrollado herramientas anteriores en el lenguaje de programación utilizado.
FLEX	Alto	Existe una gran flexibilidad en el desarrollo de la aplicación
RESL	Nominal	Se han resuelto gran parte de los riesgos.
TEAM	Muy Alto	Gran cohesión del equipo de trabajo.
PMAT	Nominal	Valor normal de madurez de los procesos.

### 8.2.1. Resultados de la planificación

A continuación mostramos los resultados generales, obtenidos de la estimación realizada, proporcionados por el método empleado. Para ello, introducimos los datos anteriores en la herramienta de estimación y generemos los siguientes resultados:

En primer lugar obtenemos la estimación global del proyecto:

**Tabla 4. Estimación global del Proyecto.**

Proyecto	CATEBus
Tamaño Total (líneas de código)	3961
Esfuerzo del desarrollo (meses hombre)	11,5
Tiempo de desarrollo (meses)	7,86
Productividad estimada (líneas de código/meses hombre)	343,644
Coste total estimado (euros)	17289,69

Seguidamente se desglosarán los resultados obtenidos por fases y características de las mismas, así como por la planificación del mismo.

#### 8.2.1.1. Calendario

A continuación mostramos los datos relativos a la estimación del calendario, es decir, el tiempo previsto que se empleará para el desarrollo y realización del proyecto.

**Tabla 5. Estimación del calendario por fases**

Fase	Calendario (%)	Calendario (Meses)	Esfuerzo (%)	Esfuerzo (Meses)	Recursos Humanos
Planificación y Análisis	16,65%	1,308991	7,00%	0,806852	0,616393
Diseño	24,33%	1,912108	17,00%	1,959499	1,024785
Programación	54,69%	4,298886	63,02%	7,263919	1,689721
Integración y Pruebas	20,98%	1,649084	19,98%	2,303045	1,39656

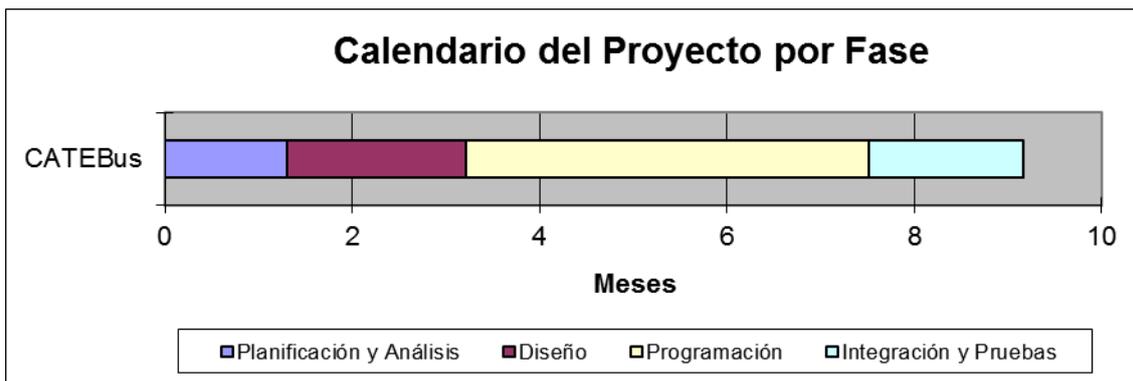


Ilustración 32. Calendario del Proyecto por Fases

8.2.1.2. Costes

En este epígrafe se resumen las estimaciones del coste total del proyecto, así como los costes en cada una de las fases del mismo.

Tabla 6. Coste del proyecto

Fase	Hitos	Coste por Persona Mes	Coste por Fase	Coste Acumulado
				0 €
<b>Planificación y Análisis</b>	Revisión de Planes y Requisitos (PRR)	1.500,00 €	1.210 €	1.210 €
<b>Diseño</b>	Revisión del Diseño (PDR)	1.500,00 €	2.939 €	4.150 €
<b>Programación</b>	Revisión de Programación (UTCR)	1.500,00 €	10.896 €	15.045 €
<b>Integración y Pruebas</b>	Revisión de Aceptación de Software(SAR)	1.500,00 €	3.455 €	18.500 €

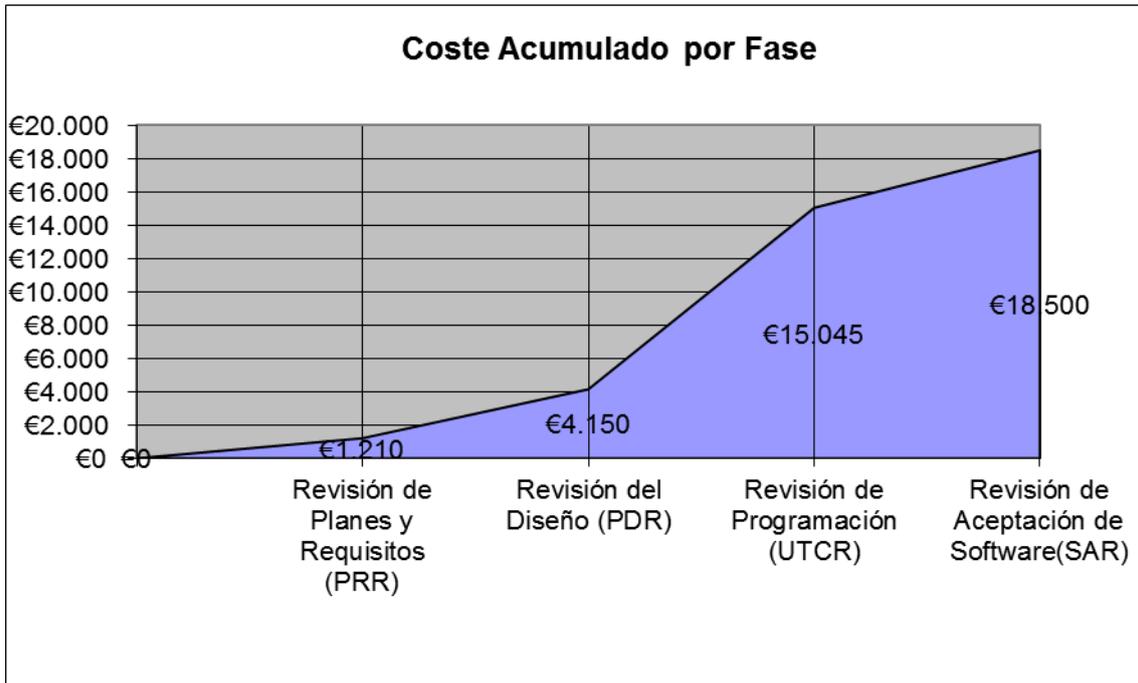


Ilustración 33. Coste Acumulado por Fase

### 8.2.1.3. Esfuerzo

En este apartado se especifica los datos del esfuerzo requerido, medido en Meses Hombre, para la realización del proyecto.

Tabla 7. Esfuerzo por fases.

Fase	Esfuerzo (Meses)	Porcentaje
Planificación y Análisis	0,806852	7,00%
Diseño	1,959499	17,00%
Programación	7,263919	63,02%
Integración y Pruebas	2,303045	19,98%

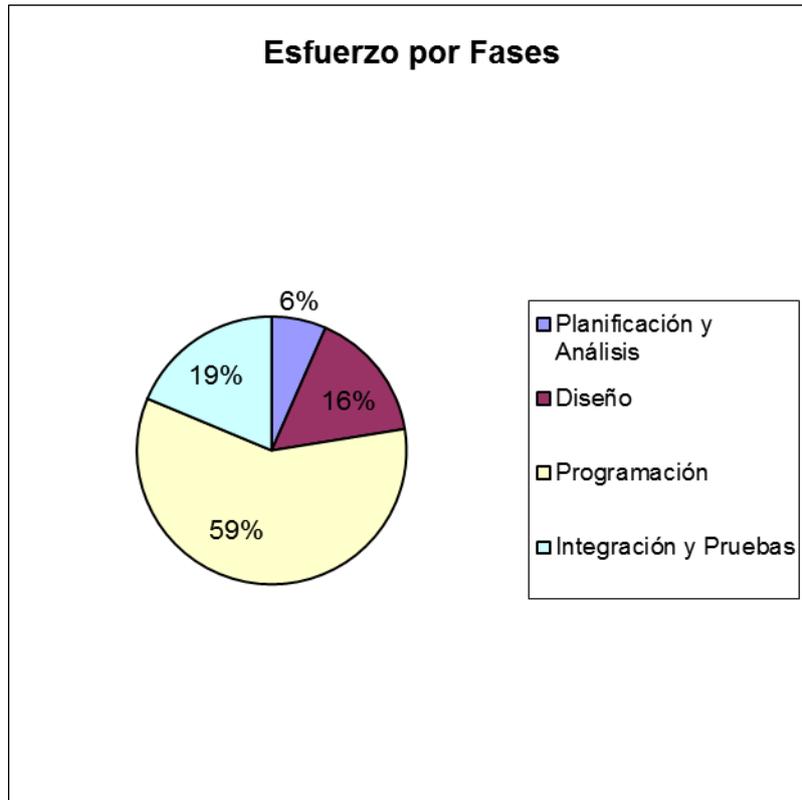


Ilustración 34. Esfuerzo por Fases

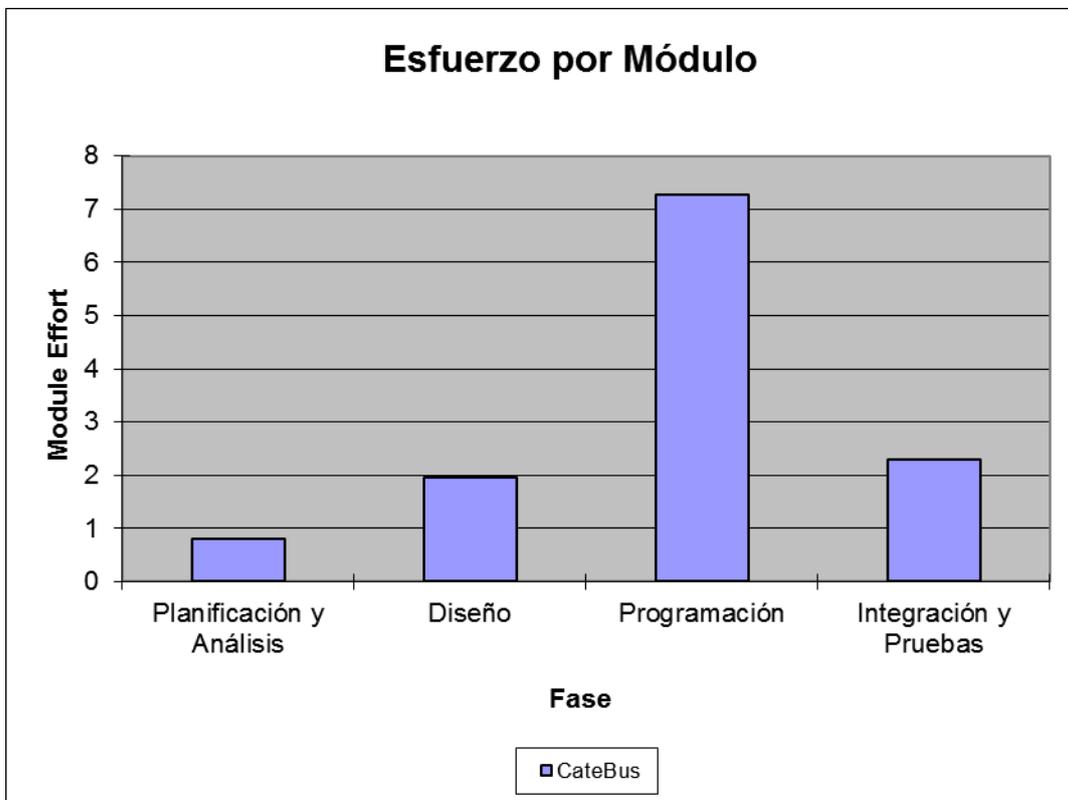


Ilustración 35. Esfuerzo por módulo

8.2.1.4. Recursos Humanos

A continuación se muestra la información relativa a la estimación de los recursos humanos necesarios para la realización de la aplicación.

Tabla 8. Recursos humanos

Fase	Personal a Jornada Completa
Planificación y Análisis	0,616393
Diseño	1,024785
Programación	1,689721
Integración y Pruebas	1,39656

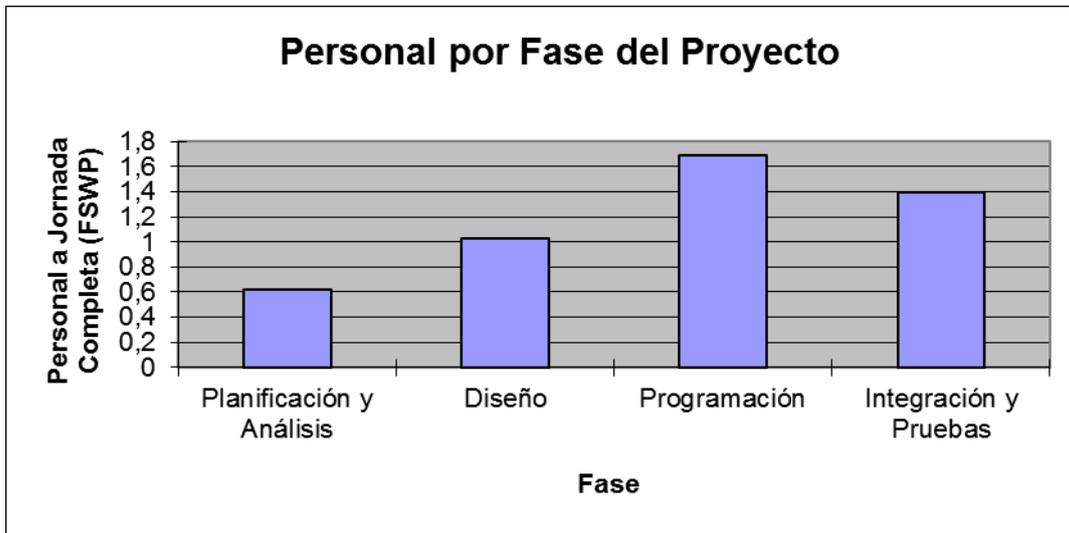


Ilustración 36. Personal por fase del proyecto

8.2.1.5. Actividades

En este apartado se especifica la estimación del esfuerzo requerido en cada una de las distintas actividades llevadas a cabo en el desarrollo de la aplicación.

Tabla 9. Esfuerzo por actividades

Fase	Planificación y Análisis	Diseño	Programación	Integración y Pruebas
Planificación y Análisis	0,384652	0,244937	0,290557	0,057576
Diseño	0,130415	0,803394	0,581114	0,115152
Programación	0,022808	0,238342	4,104114	0,775059
Pruebas	0,02149	0,09138	0,302427	0,057576
Verificación y Validación	0,04973	0,120772	0,520345	0,729447
Administración	0,122425	0,248331	0,532924	0,191995
CM/QA	0,026921	0,055583	0,496604	0,191995
Manuales	0,048411	0,15676	0,435835	0,184244

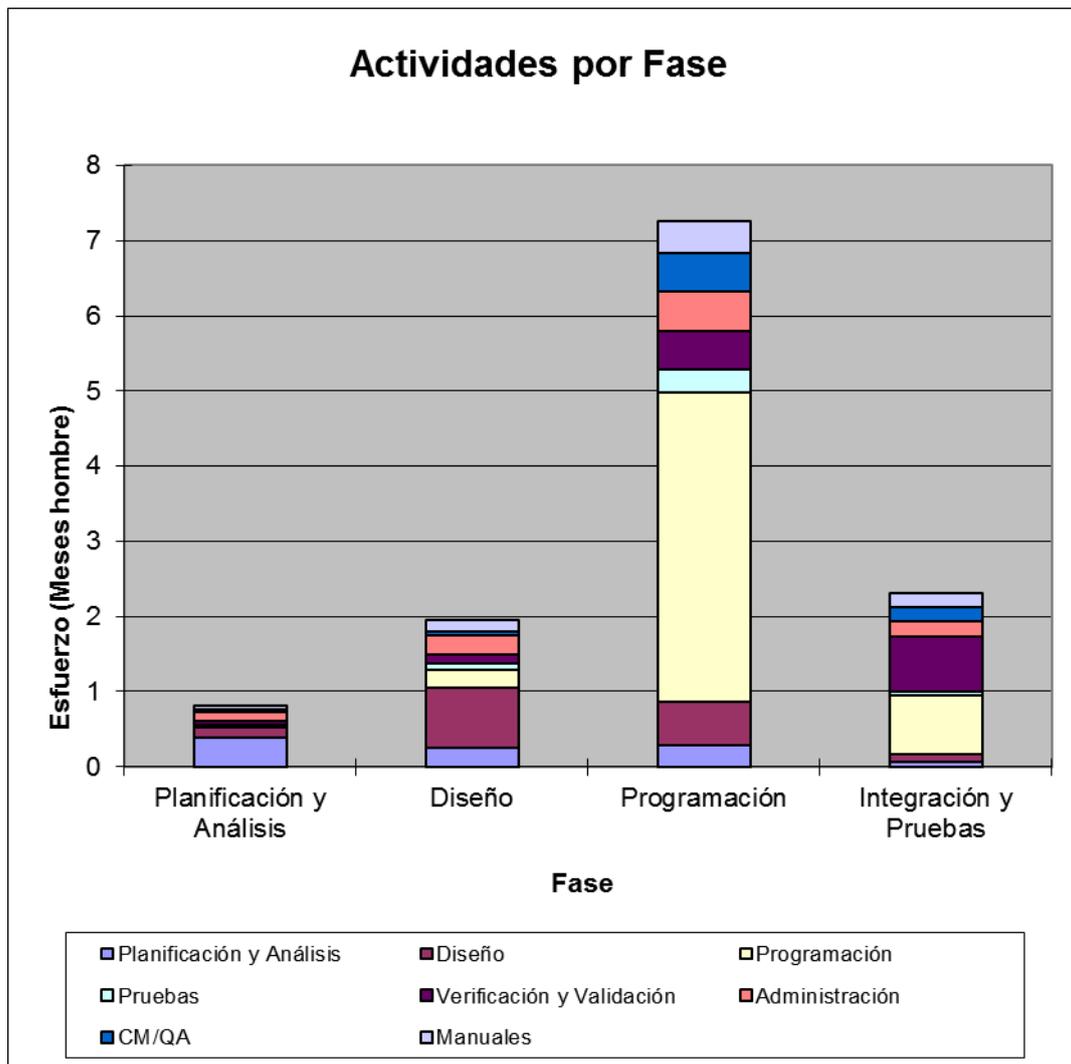


Ilustración 37. Actividades por fase

### 8.3. PLANIFICACIÓN

A continuación se muestra la planificación total realizada del proyecto por cada una de las distintas fases de las que consta.

Tabla 10. Fechas estimadas del proyecto

Fase	Duración
Planificación y Análisis	24 días
Diseño	40 días
Programación	100 días
Integración y Pruebas	20 días

Debido a que el ciclo de vida del proyecto se especifica un desarrollo en espiral con tres iteraciones, se desglosan estas planificaciones en cada una de ellas.

Para la realización de esta gráfica se ha utilizado la herramienta Microsoft Project, una de las aplicaciones más extendidas para realizar trabajos de planificación y seguimiento de proyectos.

Tabla 11. Fechas estimadas del proyecto por fases.

Fase	Duración	Fecha Comienzo	Fecha Fin
Planificación y Análisis 1	8 días	Miércoles 27/01/10	Viernes 05/02/10
Diseño 1	25 días	Lunes 08/02/10	Viernes 12/03/10
Programación 1	45 días	Lunes 15/03/10	Viernes 14/05/10
Integración y Pruebas 1	9 días	Lunes 17/05/10	Jueves 27/05/10
Planificación y Análisis 2	8 días	Viernes 28/05/10	Martes 08/06/10
Diseño 2	5 días	Miércoles 09/06/10	Martes 15/06/10
Programación 2	35 días	Miércoles 16/06/10	Martes 03/08/10
Integración y Pruebas 2	6 días	Miércoles 04/08/10	Miércoles 11/08/10
Planificación y Análisis 3	8 días	Jueves 12/08/10	Lunes 23/08/10
Diseño 3	10 días	Martes 24/08/10	Lunes 06/09/10
Programación 3	20 días	Martes 07/09/10	Lunes 04/10/10
Integración y Pruebas 3	5 días	Martes 05/10/10	Lunes 11/10/10

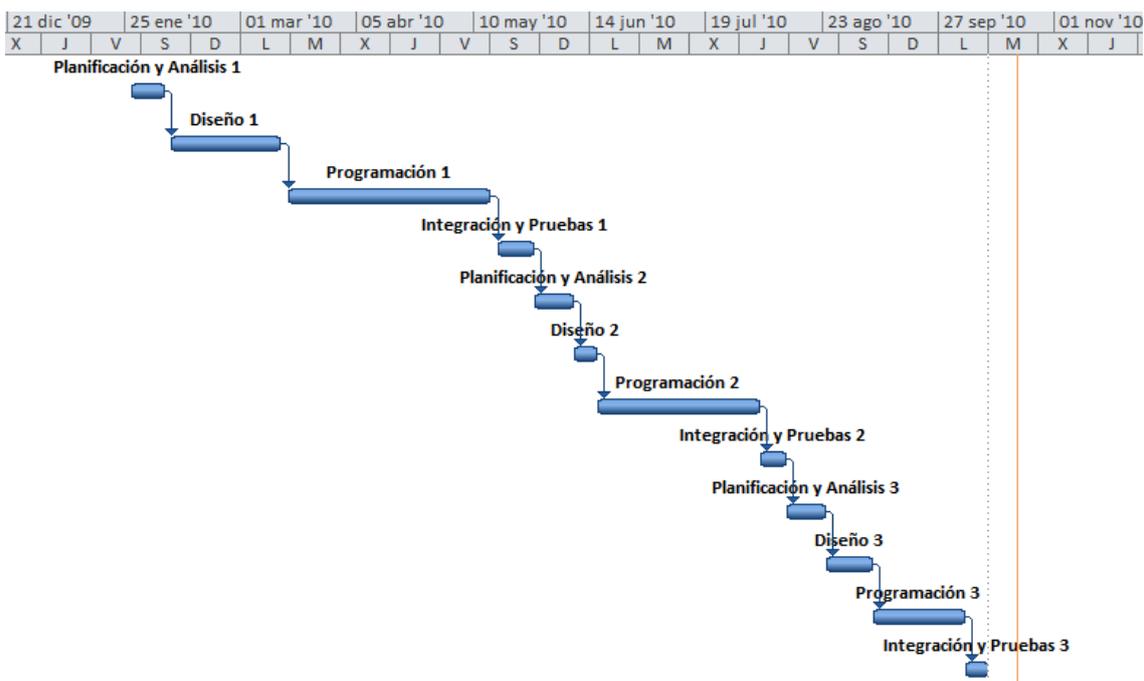


Ilustración 38. Diagrama Gant

Una vez terminado el proyecto, como se ha comentado anteriormente, se incluyeron la adaptación de las funcionalidades para dispositivos móviles del personal docente, de modo que se ha ampliado el tiempo de desarrollo del proyecto. Además, se refinaron los estilos CSS de la aplicación, proporcionando una mejor visualización de la información. Debido a esto, se incluyen nuevas estimaciones de dicho periodo de tiempo empleado en implementar las nuevas funciones.

Tabla 12. Ampliación de la estimación total

Fase	Duración
Planificación y Análisis	28 días
Diseño	43 días
Programación	107 días
Integración y Pruebas	25 días

Tabla 13. Estimación de fechas de las nuevas funcionalidades por fases

Fase	Duración	Fecha Comienzo	Fecha Fin
Planificación y Análisis 4	4 días	Lunes 04/04/11	Jueves 07/04/11
Diseño 4	3 días	Viernes 08/04/11	Martes 12/04/11
Programación 4	7 días	Miércoles 13/04/11	Viernes 29/04/11
Integración y Pruebas 4	5 días	Martes 03/05/11	Lunes 09/05/11

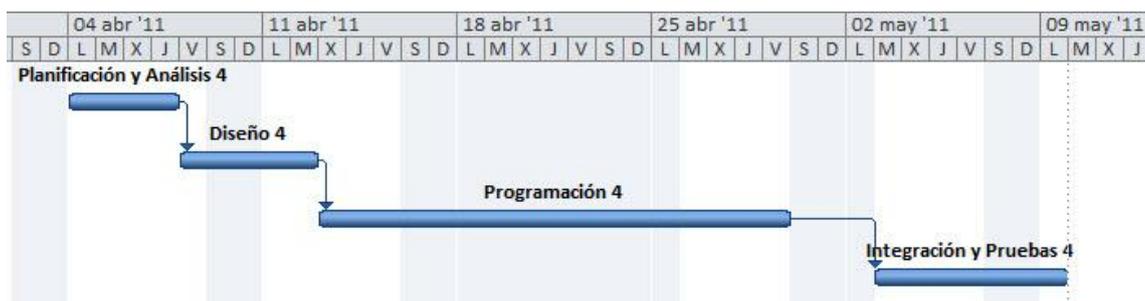


Ilustración 39. Diagrama de Gantt – 2

Los salarios totales del personal se desarrollarán en el presupuesto del proyecto.

## 8.4. PRESUPUESTO

En este apartado se muestran y analizan los costes del proyecto. Para ello, se detallan tanto las herramientas software como hardware, servicios utilizados, etc.; así como referencias a los datos de tiempo, esfuerzo y precio expuesto en epígrafes anteriores. Se aunarán todos los datos y se ofrecerá un coste final del proyecto.

Realizamos esto una vez estimado los costes y el esfuerzo estimado, para ajustar y contemplar los gastos reales del proyecto, es decir, incluir gastos hardware/software, riesgos, etc.; gastos no soportados por el modelo de estimación.

### 8.4.1. Recursos Materiales Utilizados

Es esta sección se enumerarán los recursos tanto software como hardware que se han utilizado para el desarrollo de este proyecto.

#### 8.4.1.1. Recursos software

- Sistema Operativo Ubuntu: Sistema operativo de libre distribución basado en Linux en el que se ha desarrollado el proyecto. Coste: 0€
- Java JDK (*Java Development Kit*) 6: Kit que engloba las herramientas de desarrollo necesarias para la creación de aplicaciones en el lenguaje Java. Coste: 0€
- NetBeans IDE versión 6: Entorno de desarrollo de la aplicación Coste: 0€
- Apache Tomcat versión 6: Servidor web en el que se despliega y ejecuta la aplicación web. Coste 0€

- MySQL: Sistema gestor de bases de datos de distribución libre. Coste: 0€
- Java Mail: Librerías que permiten la comunicación de la aplicación con servidores de correo electrónico, así como la creación y manejo de los mismos. Coste: 0€
- Java CSV Library: Librería que permite la manipulación de este tipo de ficheros. Coste: 0€
- iText: API que permite el manejo de ficheros PDF: Utilizada la versión de libre distribución. Coste: 0€
- Commons FileUpload y Commons IO: Librerías para controlar y manejar la entrada y salida de ficheros en la aplicación. Coste: 0€
- Adobe Reader 9: Programa que permite la visualización de archivos PDF. Coste: 0€
- Microsoft Office Professional 2010. Paquete de aplicaciones que contiene: Word, Excel, PowerPoint, Outlook, OneNote, Access y Publisher Coste: 699.00€
- Microsoft Project Professional 2010. Herramienta de planificación y seguimientos de proyectos Coste: 1.300,00 €

#### 8.4.1.2. Recursos hardware

- Ordenador personal Dell Studio XPS 7100: Características:
  - Procesador AMD Phenom(tm) II X6 1045T
  - 8GB RAM
  - 1TB de disco duro
  - Tarjeta gráfica ATI Radeon HD 5150
  - Grabadora CD+DVD
  - Coste: 799€
- Pantalla LG: Coste 150€
- Cable Modem Cisco EPC3825: Coste 80€

### 8.4.1.3. Resumen

Tabla 14. Gastos de recursos materiales

Recursos	Total
Herramientas Software	1999,00€
Herramientas Hardware	1029,00€
<b>TOTAL</b>	<b>3028,00€</b>

### 8.4.2. Salarios del personal

A continuación mostramos los gastos asociados al personal cualificado que ha trabajado durante la realización del proyecto. Este proyecto lo ha realizado una persona que ha adoptado distintos roles, que se especifican a continuación

Tabla 15. Sueldos por categoría

Descripción	Sueldo neto/mes	Sueldo bruto/mes	Sueldo bruto/año	Coste/hora
Analista	1170€	1500€	21000€	12,50€
Diseñador	1170€	1500€	21000€	12,50€
Programador	1170€	1500€	21000€	12,50€
Responsable de Pruebas	1170€	1500€	21000€	12,50€

Estos sueldos están calculados teniendo en cuenta las siguientes características:

- Coste/hora es el coste bruto del trabajador en una hora de trabajo.
- Bruto/año indica el sueldo bruto del trabajador en 14 pagas.
- Neto/mes indica el sueldo final al mes del trabajador descontando la parte del I.R.P.F (20%) y de la Seguridad Social (2.4%).
- La jornada laboral es de 6 horas/día y los días trabajados al mes es de 20.

Una vez obtenidos estos datos, se procede a calcular los gastos del personal que se imputan al desarrollo de este proyecto.

Tabla 16. Salarios reales del proyecto

Descripción	Horas	Coste/hora	Total
Analista	168	12,5€	2100,00€
Diseñador	258	12,5€	3225,00€
Programador	642	12,5€	8025,00€
Responsable de Pruebas	150	12,5€	1875,00€
<b>TOTAL</b>			<b>15225,00€</b>

### 8.4.3. Gastos indirectos

Ahora mostramos los gastos indirectos que se han producido en el desarrollo del proyecto.

Tabla 17. Gastos indirectos asociados al proyecto

Descripción	Coste
Electricidad	Incluido en los costes indirectos
Agua	Incluido en los costes indirectos
Alquiler del local	Incluido en los costes indirectos
Amortización inmobiliario	Incluido en los costes indirectos
Gastos de comunidad	Incluido en los costes indirectos
Costes de estructura	Incluido en los costes indirectos
Limpieza	Incluido en los costes indirectos
Gastos de mantenimiento	Incluido en los costes indirectos
Desplazamientos	Incluido en los costes indirectos
<b>Costes indirectos (10%)</b>	<b>15225,00 € * 0.10 = 1522.50 €</b>

### 8.4.4. Resumen

Finalmente, sumamos todos los gastos justificados anteriormente para la confección del presupuesto total del proyecto.

Tabla 18. Resumen del presupuesto

Descripción	Coste
Recursos materiales	3028,00 €
Personal	15225,00€
Gastos indirectos	1522,50€
<b>Total</b>	<b>19775,50€</b>

Al coste total del proyecto obtenido se le debe sumar un margen de gastos imprevistos al mismo. Se ha determinado que ese margen sea del 10% del coste del proyecto.

Además, se deben de calcular los beneficios que se obtienen con la realización del proyecto. Se ha determinado que el margen de beneficios es del 20% del coste.

Sumado estos dos últimos costes, el resultado del proyecto final es de:

Coste total + Margen de imprevistos + Beneficios = 19775,5€ + 1977,55€ + 3955,1€ = **25708,15 €, I.V.A incluido.**

## 9. INSTALACIÓN DE LA APLICACIÓN Y MANUAL DE USUARIO

---

En este apartado se especifican los pasos y aplicaciones necesarias para la puesta en funcionamiento de la aplicación desarrollada. Además, se incorpora un manual de usuario para que los empleados e interesados en utilizar este sistema puedan conocer la funcionalidad del mismo y cómo acceder a los datos manejados.

### 9.1. INSTALACIÓN

El proyecto CATEBus se ha desarrollado en Linux, bajo la distribución Kubuntu, con lo que el primer paso sería la instalación de dicho sistema operativo u otro Linux. La aplicación está realiza en el lenguaje orientado objetos Java, con lo que se deberán de descargar las librerías y distribuciones necesarias para su correcto funcionamiento. Como sistema gestor de bases de datos, utiliza MySQL, en su versión para Linux.

Durante este apartado se detalla la instalación de cada una de las aplicaciones y sistemas anteriores.

#### 9.1.1. Instalación de Linux

La distribución de Linux Ubuntu, así como el resto de distribuciones de su familia (como Kubuntu), están disponibles gratuitamente y se pueden descargar desde su página web (<http://www.ubuntu.com/desktop/get-ubuntu/download>).

Una vez descargada la imagen, basta con grabarla en un cd y arrancar la máquina donde se realizará la instalación con el cd introducido. De este modo nos saldrá una serie de opciones, desde la que podremos elegir la instalación del sistema operativo. Durante la instalación nos pedirán algunos datos, como dónde se realizará la instalación, si se necesita formatear el disco duro para su correcta instalación, el nombre del equipo y un nombre de usuario y contraseña. Además, es posible que se pregunten valores como la región en la que se encuentran, idioma del teclado o franja horaria, así como el tipo de conexión a internet.

Durante el proceso de instalación, es posible que el equipo se reinicie varias veces para una correcta configuración. Transcurridos unos minutos, tendremos listo el sistema operativo en el que se ejecutará y desplegará la aplicación web elaborada.

### 9.1.2. Instalación de Java

En primer lugar, como algunas distribuciones de Linux incluyen ya las librerías de java, comprobamos que están instaladas. Para ellos abrimos un nuevo terminal de usuario e introducimos la siguiente sentencia:

```
dpkg -get-selections | grep sun-java
```

El resultado será los paquetes de java instalados en la máquina. En el caso de que no aparezca nada tras la ejecución del comando, o se quiera estar seguro de tener instalada la versión válida; basta con ejecutar el siguiente comando:

```
sudo apt-get install sun-java6-jdk
```

A continuación se pedirá la contraseña del root, y comenzará la descarga de los paquetes adicionales necesarios y la instalación de los mismos.

En el caso de que no encuentre el paquete anterior, deberá comprobar que tiene aceptados todos los paquetes de software de Linux, ya que, si no acepta todos, es posible que no encuentre el JDK oficial de java.

Una vez realizada la instalación, se deben definir una serie de variables de entorno, que localicen la máquina virtual java (JAVA\_HOME), así como la ruta de los directorios donde el compilador de Java podrá encontrarlos ficheros (PATH). Para ello debemos de modificar el fichero .bashrc, almacenado oculto dentro de la carpeta home del usuario. Para editarlo introducimos el siguiente comando en la Shell de Linux:

```
sudo gedit /home/nombre_usuario/.bashrc
```

Y nos aparecerá el editor de textos con el archivo en modo escritura. Si nos pregunta previamente por la contraseña del administrador, la introducimos. Nos desplazamos al final del archivo y añadimos las siguientes líneas:

```
JAVA_HOME='ruta del directorio donde se ha instalado java'  
PATH=$JAVA_HOME/bin:$PATH  
export JAVA_HOME  
export PATH
```

Una vez cerrado el archivo tendríamos configurado Java en el sistema.

### 9.1.3. Instalación de MySQL

[14]Para la instalación del sistema gestor de bases de datos, se necesitan descargar dos paquetes *mysql-server* y *mysql-client*. Para ello, abrimos la consola de comando de Linux e introducimos la siguiente sentencia:

```
sudo apt -get install mysql-server
```

A continuación nos pedirá que introduzcamos la contraseña del usuario root. Una vez introducida, se producirá la búsqueda del paquete en los repositorios del sistema operativo en internet y procederá a instalar los nuevos paquetes, y los adicionales que necesite; así como la modificación de aquellos que sean necesarios. Si en algún momento necesita la confirmación del usuario, ésta será dada para continuar con la instalación. Cabe destacar que al instalarse el paquete *mysql-server*, éste instala a su vez el *mysql-client*.

Durante la instalación del sistema gestor, se pedirá al usuario que se defina la contraseña del administrador de la misma (root). Se deberá definir esta contraseña y memorizarla, para realizar las operaciones de creación de tablas y usuarios desde la cuenta del administrador de la base de datos.

### 9.1.4. Instalación de Apache Tomcat

Para la instalación de este servidor, accedemos a descargar los archivos del mismo desde su página web (<http://tomcat.apache.org/download-60.cgi>), una vez descargados, los descomprimos en el directorio */usr/local*.

Para que se instale correctamente en el sistema necesitamos exportar las siguientes variables de entorno en el sistema, de la misma forma que hicimos con las variables de entorno para java

```
export CATALINA_HOME='ruta del directorio donde está  
instalado tomcat';
```

```
export PATH=$PATH:$CATALINA_HOME/bin
```

### 9.1.5. Creación de la base de datos

Las sentencias relativas a la creación de la base de datos, así como las sentencias de creación de usuarios y permisos, se encuentran en el fichero CateBus.sql. Para poder transportar esas sentencias a MySQL se realizará lo siguiente:

Accederemos a MySQL como administradores a través del terminal de comandos introduciendo la siguiente sentencia

```
mysql -u root -p
```

Se pedirá la contraseña del usuario root de la base de datos, la cual facilitaremos. En ese momento entraremos en la Shell (terminal de comandos) de MySQL. En ese momento introduciremos el siguiente comando:

```
source CateBus.sql
```

Esto se introduce suponiendo que nos encontramos en el mismo directorio que el fichero .sql; en caso contrario, habría que indicarle la ruta completa del mismo. En ese momento se ejecutarán cada una de las sentencias almacenadas en el archivo y aparecerán los resultados por pantalla. Una vez asegurados de que no ha aparecido ningún error podremos abandonar MySQL mediante la orden *quit*

### 9.1.6. Despliegue de la aplicación

Para desplegar la aplicación en el servidor, basta con copiar el archivo de despliegue CateBus.war al directorio \$CATALINA\_HOME/webapps

Una vez realizada la acción se procede a iniciar Tomcat, para ello introducimos en el intérprete de comandos la sentencia

```
startup.sh
```

## 9.2. MANUAL DE USUARIO

A continuación se procede a detallar en profundidad cómo se utiliza la aplicación, las funcionalidades que ofrece, así como los valores a introducir, permisos y una guía de imágenes explicativas de las distintas operaciones. Para ello, se dividirá este manual en cada uno de los tres tipos de usuarios soportados por la aplicación: personal docente, padres y responsables de rutas.

Los destinatarios de este documento son estos tipos de usuario. Si se desea consultar detalles específicos de la programación o implementación del proyecto, deberá dirigirse a apartados anteriores.

### 9.2.1. Personal Docente

En el momento de adquisición de la aplicación, se generarán varios usuarios y contraseñas para que el personal docente del centro escolar pueda acceder a los recursos de la aplicación.

#### 9.2.1.1. Inicio de sesión

Una vez obtenidas las cuentas de usuario, el personal docente podrá ingresar en la aplicación desde la página principal de la misma. Para ello deberán rellenar los campos *usuario* y *contraseña* del formulario de inicio de sesión. Dichos campos deben cumplir las siguientes normas:

- El nombre de usuario no puede contener espacios en blanco.
- La contraseña debe tener como mínimo una longitud de 8 caracteres.

Si se produce algún error, o algún campo introducido no es correcto, se devolverá de nuevo la página de inicio para un nuevo intento de acceder a la aplicación.

Este inicio de sesión es el mismo para todos los participantes en la aplicación.



Ilustración 40. index.jsp

### 9.2.1.2. Menú principal

Una vez ingresado en la aplicación, el usuario se encontrará con la siguiente pantalla, a través de la cual podrá navegar por todas las opciones disponibles a través del menú desplegable.



Ilustración 41. Menú desplegable

Seguidamente de detallarán cada una de ellas.

### 9.2.1.3. Añadir nueva ruta

Una vez elegida la opción de añadir nueva ruta, el usuario deberá introducir el nombre de la nueva ruta en el campo *nombre*, así como elegir, de la lista desplegable mostrada, el responsable que se hará cargo de controlar dicha ruta.



The screenshot shows the 'Añadir nueva ruta' form in the Proyecto CATEBus admin interface. At the top, there is a school bus icon, the text 'Proyecto CATEBus', and a 'Bienvenido admin' message with a 'Cerrar Sesión' button. Below this is a navigation bar with four buttons: 'Rutas y Paradas', 'Responsables de ruta', 'Alumnos', and 'Justificantes'. The main form area is titled 'Datos de la nueva Ruta:' and contains two input fields: 'Nombre de la Ruta:' with the value 'Ruta Transporte Sur' and 'Responsable:' with a dropdown menu showing '- selecciona -'. At the bottom of the form are two buttons: 'Guardar' and 'Borrar'.

Ilustración 42. Añadir nueva ruta

Es importante recalcar que no se puede asignar dos rutas distintas a un mismo responsable, ya que no es posible que controle dos rutas al mismo tiempo.

Cabe destacar también que, en este formulario, así como en el resto de formularios existentes, no se podrá dejar los campos en blanco, o con valores no permitidos. Si se diese este caso, se mostraría una ventana que informará al usuario de los errores que ha cometido al escribir los datos.

En caso de que la operación haya resultado satisfactoria, se mostrará al usuario una página en la que podrá ir añadiendo las paradas que componen dicha ruta. Para ello, deberá añadir los siguientes datos:

- Nombre de la parada.
- Número de parada: se refiere a la posición que ocupa la parada en la ruta, es decir, establece el orden en el que el transporte escolar parará por las paradas.
- Dirección: se debe introducir la dirección donde el autobús parará. Para ello se debe introducir el tipo de vía, nombre de la vía, número, población y ciudad; para que dicha dirección sea única y no pueda ser confundida entre distintas ciudades.

Una vez validados los datos, se procede a la grabación de la parada introducida y se devuelve a la misma página, para poder seguir introduciendo nuevas paradas.

En caso de que se deseen borrar los valores introducidos en los formularios, se debe pulsar el botón borrar. De esa forma toda la información tecleada será eliminada. Ésta operación es útil para borrar los campos que estén erróneos antes de mandarlos.

En el caso de que no se deseen introducir más paradas, se deberá pulsar el botón correspondiente. De ese modo, la acción habrá acabado y se podrán elegir otras opciones del menú. Cabe recordar que si se introduce información en los formularios que se desee almacenar, se debe pulsar el botón *guardar* antes de querer dejar de añadir paradas.

The screenshot shows the 'Proyecto CATEBUS' web interface. At the top left is a school bus icon. To its right is the text 'Proyecto CATEBUS' and 'Bienvenido admin'. A 'Cerrar Sesión' button is located in the top right. Below this is a navigation menu with four buttons: 'Rutas y Paradas', 'Responsables de ruta', 'Alumnos', and 'Justificantes'. The 'Rutas y Paradas' button is selected. Below the menu is the 'Datos de la nueva Parada' form. It contains three input fields: 'Parada número:' with the value '1', 'Nombre:' with the value 'Av. Guerrero 1', and 'Localización:' with the value 'Avenida María Guerrero 20, Leganés, Madrid'. At the bottom of the form are three buttons: 'Guardar', 'Borrar', and 'No añadir más paradas'.

Ilustración 43. Añadir paradas a ruta

#### 9.2.1.4. Eliminar ruta

Para realizar esta operación, simplemente se ha de elegir la parada que se desea eliminar de la lista desplegable y pulsar el botón *eliminar*. Es importante conocer que, para que la parada sea eliminada, no deben existir alumnos que tengan paradas asignadas a la misma. En este caso, se producirá un error en su eliminación.

Si la eliminación se ha producido con éxito, se mostrará la siguiente pantalla de éxito, indicando que la acción ha sido realizada correctamente. En caso contrario, se mostrará la pantalla de error.



Ilustración 44. Eliminar ruta

#### 9.2.1.5. Añadir nueva parada

Al seleccionar esta opción, deberemos introducir los datos de la parada (especificados en el apartado 8.2.1.3), así como elegir la ruta en la que se incluirá.

Se debe tener en cuenta que en el campo *número de parada* se deben respetar los números ya establecidos, ya que si se repiten dos o más números, se pueden producir conflictos a la hora de recoger a los alumnos, solapándose las paradas.



Ilustración 45. Añadir nueva parada

### 9.2.1.6. Modificar Parada

Gracias a esta opción se puede modificar una parada ya establecida. Para ello se debe elegir en primer lugar la ruta en la que se encuentra y, a continuación, la parada en cuestión y pulsar el botón *modificar*.

A continuación, se mostrarán los datos de la parada almacenados, para que el usuario pueda mantenerlos si quiere, y eliminar o modificar los campos que considere oportunos. Pulsando el botón *guardar*, se almacenan los cambios introducidos en la aplicación.

The screenshot displays the user interface for modifying a stop. At the top, there is a school bus icon and the text 'Proyecto CATEBus'. To the right, it says 'Bienvenido admin' and 'Cerrar Sesión'. Below this is a navigation bar with four buttons: 'Rutas y Paradas', 'Responsables de ruta', 'Alumnos', and 'Justificantes'. The 'Rutas y Paradas' button is highlighted. Underneath, the section 'Datos de la Parada a Modificar' contains two dropdown menus. The first is labeled 'Ruta a la que pertenece' and is set to 'Ruta Transporte Sur'. The second is labeled 'Parada a modificar' and is set to '- selecciona -'. A 'Modificar' button is positioned at the bottom of the form.

Ilustración 46. Modificar Parada

### 9.2.1.7. Borrar Parada

Se deberá seleccionar la ruta que contiene esa parada, así como la parada a eliminar, y pulsar el botón correspondiente.

Como en el caso de las rutas, si la parada está asociada a algún alumno se producirá un error en su eliminación.



The screenshot shows the admin interface for 'Proyecto CATEBus'. At the top left is a school bus icon. To its right is the text 'Proyecto CATEBus' and 'Bienvenido admin'. A 'Cerrar Sesión' button is located in the top right. Below this is a navigation bar with four buttons: 'Rutas y Paradas', 'Responsables de ruta', 'Alumnos', and 'Justificantes'. The 'Rutas y Paradas' button is highlighted. Below the navigation bar, the section 'Datos de la Parada a Eliminar:' contains two dropdown menus: 'Ruta a la que pertenece:' with the value '- selecciona -' and 'Parada a eliminar:' with the value 'Seleccione una ruta'. An 'Eliminar' button is positioned below these dropdowns.

Ilustración 47. Borrado de parada

#### 9.2.1.8. Alta de Responsables y Padres

Ambas acciones son muy similares y serán explicadas como si de una se tratara.

Para que sean datos de alta, ambos usuarios deberán notificarlo por escrito al centro. En el caso de los padres, puede realizarse en el momento de solicitar la matriculación en cualquier curso. En este documento se deberá especificar un nombre de usuario y una dirección de correo electrónico válida. Cabe recordar que el nombre de usuario no deberá contener espacios en blanco.

Una vez que el personal administrativo dispone de esta información, podrá introducirla en los campos correspondientes y pulsar el botón *guardar*. Si la operación ha resultado exitosa, se mandará un correo electrónico a los interesados en la dirección proporcionada, confirmando su número de usuario y una contraseña segura creada. Una vez reciban este correo, podrán acceder a la aplicación con los datos incluidos en el mismo.



Ilustración 48. Añadir nuevo usuario



Ilustración 49. Correo electrónico enviado.

9.2.1.9. Eliminación de responsables

Eligiendo el responsable que se desea eliminar de la aplicación y pulsando el botón de acción, se eliminaría a un responsable del sistema. Éste deberá no estar asignado a ninguna ruta, y no estar asociado a ninguna falta de asistencia. Esto es debido a que se desea recoger un historial con las faltas y los responsables que han puesto dicha falta. Si al eliminar un responsable se eliminase toda su información, se perderían datos de faltas de asistencia.



Ilustración 50. Eliminación de responsables.

9.2.1.10. Asignación de rutas.

Si se desea realizar un cambio de responsable de ruta libre, es decir, sin ninguna ruta asignada, a una ruta ya existente; se podrá utilizar esta opción. Seleccionando la ruta y el responsable correspondiente, se realiza dicho cambio



Ilustración 51. Asignación de Rutas

#### 9.2.1.11. Nuevo alumno

El personal administrativo será el encargado de introducir la información de los alumnos en la aplicación. Para ello, se deben proporcionar los datos personales de los mismos, así como seleccionar a qué padre pertenecen. Una vez proporcionados estos datos, se pulsa el botón *guardar* y, si todo transcurre con normalidad, se muestra la página de éxito en la operación.



Ilustración 52. Nuevo Alumno.

En el caso que se quiera utilizar un archivo de datos CSV, se le mostrará al usuario un botón en el que podrá elegir el fichero a cargar de entre los existentes en los directorios del disco duro. Una vez elegido, al pulsar guardar comenzará la carga del fichero y el almacenamiento de los datos contenidos en él.



**Ilustración 53. Importar alumnos desde fichero**

Una vez concluida la carga de datos, y si todas las operaciones han sido correctas, se mostrará la página de éxito en la operación. En caso de producirse algún error, se mostrará la pantalla correspondiente.

El archivo CSV deberá mantener la siguiente estructura, y cada elemento debe tener como delimitador el carácter de punto y coma “;”.

CABECERA

CUERPO

Pudiendo contener la cabecera cualquier texto explicativo de los elementos a los que representa, y el cuerpo deben ser los siguientes valores:

- Identificador del padre de la aplicación.
- Apellidos del alumno.
- Nombre del alumno.

Un ejemplo de fichero válido sería el siguiente:

```

Padre;Apellidos Alumno;Nombre Alumno
46012369;Jimenez Calvo;Pedro
46012369;Jimenez Calvo;Sandra
46012369;Jimenez Calvo;Leticia
36874010;Ramirez Sanchez;Roberto
51236958;Fernández de la Torre;Alfonso
    
```

Ilustración 54. Ejemplo de archivo de datos

### 9.2.1.12. Listados

La aplicación podrá ofrecer distintos listados con la información almacenada. Concretamente se podrán visualizar los alumnos existentes en cada ruta, y las faltas cometidas en una jornada lectiva.

Para el primer caso, basta con seleccionar la ruta a consultar y pulsar el botón correspondiente. Si no se produce ningún error, se mostrará el listado de los alumnos que pertenecen a dicha ruta en una nueva página.



Ilustración 55. Listados

Para el segundo caso, se deberá especificar una fecha de consulta. Dicha fecha deberá estar escrita en el siguiente formato "DD/MM/AAAA". En caso de no ser así, se mostrará un mensaje indicando el error producido. Una vez introducida la fecha, se mostrará una página con los detalles de las faltas cometidas en esa jornada.

9.2.1.13. Entrega de justificantes.

Los padres, deberán entregar un justificante escrito al centro escolar en el que se especifiquen los motivos de la falta, así como entregar los documentos que acrediten dicho motivo. Para dejar constancia de este hecho, el personal podrá justificar las faltas que se hayan reflejado en la aplicación. Para ello, deberán seleccionar el alumno pertinente y la fecha de la falta a justificar. Además se introducirá un resumen del motivo detallado en el informe entregado. Una vez introducido los datos, pulsamos el botón y, si todo ocurre exitosamente, se habrá cambiado el estado de la falta, pasando a ser una falta justificada.

Ilustración 56. Entrega de justificantes.

9.2.1.14. Personalizar Añadir Alumno

Al seleccionar esta opción del menú Alumnos, los responsables del centro podrán seleccionar el modo estándar de introducción de alumnos en la aplicación. Para ello, deberán seleccionar la opción que prefieran, de forma que el botón se oscurezca. Una vez elegida la opción pulsamos el botón *Guardar*, donde se almacenará en la aplicación la opción seleccionada.



Ilustración 57. Personalizar opciones

## 9.2.2. Padres

Para que los padres puedan acceder a la aplicación, han de solicitar su intención al personal del centro, indicando nombre de usuario y dirección de correo electrónico. El personal les dará de alta en el sistema, remitiéndoles un correo electrónico en el que se especificarán el nombre de usuario y su contraseña de acceso. Para acceder al sistema se procede de la forma explicada en el epígrafe 8.2.1.1.

### 9.2.2.1. Visualización de rutas y paradas

Los padres podrán visualizar toda la información relativa a las rutas y sus respectivas paradas, para poder elegir una parada del curso o alguna ocasional de manera clara y precisa. Para ello, únicamente tendrán que seleccionar la ruta de la lista desplegable y, si se trata de una parada en concreto, también la parada. Pulsando el botón correspondiente, se generan nuevas páginas con la información de los elementos seleccionados.


Bienvenido padre1  
Proyecto CATEBus
Cerrar Sesión

Información
Paradas
Retraso
Faltas de asistencia
Cambiar Contraseña

Ruta Ruta Transporte Sur

Responsable: Responsablea



Numero	Nombre	Dirección
1	Av. Guerrero 1	Avenida Maria Guerrero 18, Leganes, Madrid <a href="#">Ver Parada</a>
2	Av. Guerrero 2	Avenida Maria Guerrero 60, Leganes, Madrid <a href="#">Ver Parada</a>
3	Reina Sofia	Calle Reina Sofia 17, Leganes, Madrid <a href="#">Ver Parada</a>
4	Gran Bretana	Avenida Gran Bretaña 12, Leganés, Madrid <a href="#">Ver Parada</a>
5	Priorato	Calle Priorato 23, Leganes, Madrid <a href="#">Ver Parada</a>

**Ilustración 58. Información de Rutas**

En el caso de las rutas, se podrán visualizar todas sus paradas, así como el responsable que controla la ruta. Además, se ofrecen enlaces en cada una de ellas, que llevan a una página en la que se visualizan sus datos, así como un mapa en el cual se puede localizar espacialmente su dirección. Este mapa es dinámico y puede ser manejado para visualizar localizaciones de alrededor, así como el cambio de zoom o de tipo de mapa mostrado.

En caso de producirse algún error, se mostraría la página de error, o se indicaría que no se tiene información para dicha ruta o parada.



Bienvenido *Roberto Alvarez*

**Proyecto CATEBus**

Cerrar Sesión

Información
Paradas
Retraso
Faltas de asistencia
Cambiar Contraseña

DATOS DE LA PARADA



Parada número: 1  
 Nombre: Av. Guerrero 1  
 Dirección: Avenida María Guerrero 20, Leganés, Madrid

Ilustración 59. Información de la parada

### 9.2.2.2. Asignación de parada del curso y parada ocasional

Los padres deberán asignar para cada uno de sus hijos, una parada en la que serán recogidos para su transporte al centro escolar. Para poder ser seleccionada, se deberá seleccionar el alumno a tratar, la ruta a la que pertenece la parada y la parada misma.



**Proyecto CATEBus** Bienvenido *Roberto Alvarez* [Cerrar Sesión](#)

[Información](#)
[Paradas](#)
[Retraso](#)
[Faltas de asistencia](#)
[Cambiar Contraseña](#)

**Asignar la Parada del curso**

Alumno:

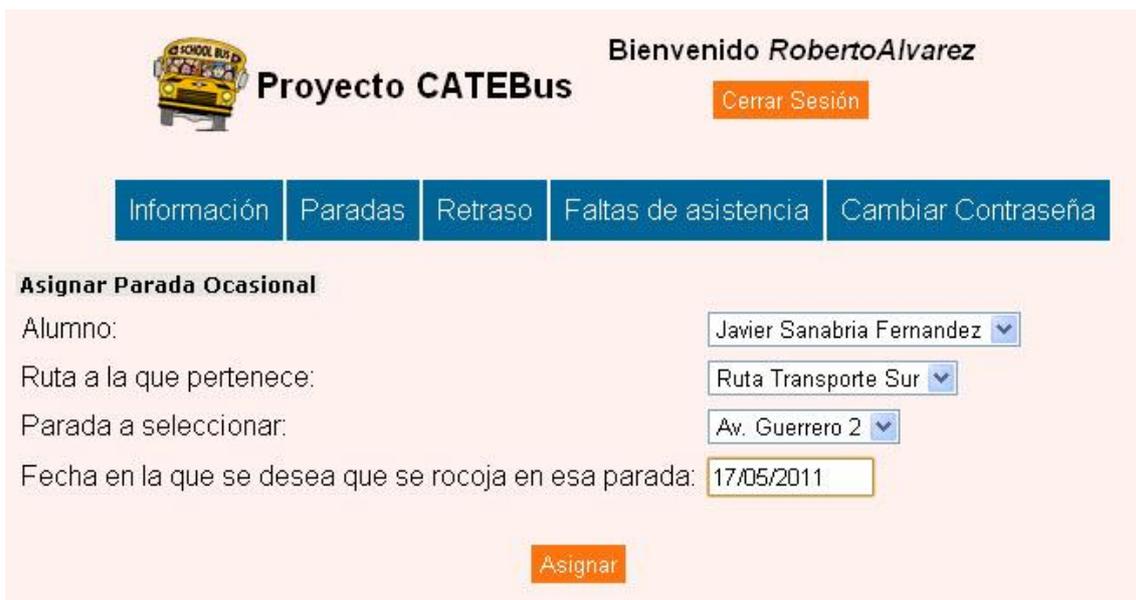
Ruta a la que pertenece:

Parada a seleccionar:

[Asignar](#)

Ilustración 60. Asignación de parada del curso.

Así mismo, se puede solicitar una parada distinta en un momento determinado, debido a problemas de desplazamiento, imposibilidad de llevar al alumno a la parada normal, etc. Para ello se deberá especificar el alumno, la parada ocasional y la fecha en la que se producirá dicho cambio. Una vez pulsado el botón, se almacenará en la aplicación este evento y será tratado en el momento de que el responsable de la ruta vaya a recoger a los alumnos.



**Proyecto CATEBus** Bienvenido *Roberto Alvarez* [Cerrar Sesión](#)

[Información](#)
[Paradas](#)
[Retraso](#)
[Faltas de asistencia](#)
[Cambiar Contraseña](#)

**Asignar Parada Ocasional**

Alumno:

Ruta a la que pertenece:

Parada a seleccionar:

Fecha en la que se desea que se rocoja en esa parada:

[Asignar](#)

Ilustración 61. Parada ocasional.

### 9.2.2.3. Solicitar retraso

Si un alumno se va a retrasar, por un periodo máximo de 5 minutos, en llegar a la parada del transporte, puede indicar este hecho a la aplicación, para que el responsable lo sepa y espere ese tiempo a la llegada del alumno. Si transcurridos 5 minutos el alumno no llega, el transporte seguirá su ruta hacia la siguiente parada, con la consecuente falta de asistencia a la ruta.

Para solicitar un retraso, se debe seleccionar el alumno en cuestión. Así como la fecha en la que se producirá.

The screenshot shows the user interface of the Proyecto CATEBus application. At the top left is a yellow school bus icon. To its right, the text 'Proyecto CATEBus' is displayed. Further right, the user is logged in as 'Bienvenido Roberto Alvarez', with a 'Cerrar Sesión' button next to it. Below the header is a navigation menu with five buttons: 'Información', 'Paradas', 'Retraso', 'Faltas de asistencia', and 'Cambiar Contraseña'. The 'Retraso' button is highlighted. Underneath, the 'Asignar Retraso' section contains a form with two fields: 'Alumno:' with a dropdown menu showing 'Javier Sanabria Fernandez' and 'Fecha en la que se producirá el retraso:' with a date input field containing '18/05/2011'. A 'Retraso' button is positioned below the date field.

Ilustración 62. Pedir Retraso

### 9.2.2.4. Programar falta de asistencia

Si se conoce que en una determinada fecha un alumno va a faltar a clase y, por consiguiente, a la ruta de transporte; se puede programar dicha falta en la aplicación, para que los responsables de ruta no esperen por ellos. Para ello se debe seleccionar el alumno que faltará y en qué fecha (se recuerda que el formato para una fecha es "DD/MM/AAAA").

The screenshot shows the 'Proyecto CATEBus' interface. At the top left is a school bus icon. To its right, the text 'Bienvenido Roberto Alvarez' is displayed above a 'Cerrar Sesión' button. Below this is a navigation bar with five buttons: 'Información', 'Paradas', 'Retraso', 'Faltas de asistencia', and 'Cambiar Contraseña'. The 'Faltas de asistencia' button is highlighted. Underneath, the section 'Nueva Falta de Asistencia' contains a form with 'Alumno:' set to 'Javier Sanabria Fernandez' and 'Fecha en la que el alumno faltará:' set to '19/05/2011'. An 'Asignar' button is located at the bottom of the form.

Ilustración 63. Programar falta de asistencia.

#### 9.2.2.5. Generar justificante

Los padres podrán descargar desde la aplicación el documento justificante de la falta de un alumno. Para ello, deberán elegir el alumno y la fecha en que se produjo la falta.

The screenshot shows the 'Proyecto CATEBus' interface. At the top left is a school bus icon. To its right, the text 'Bienvenido Roberto Alvarez' is displayed above a 'Cerrar Sesión' button. Below this is a navigation bar with five buttons: 'Información', 'Paradas', 'Retraso', 'Faltas de asistencia', and 'Cambiar Contraseña'. The 'Faltas de asistencia' button is highlighted. Underneath, the section 'Generar Justificante' contains a form with 'Alumno:' set to 'Javier Sanabria Fernandez' and 'Falta:' set to '- selecciona -'. A 'Generar Justificante' button is located at the bottom of the form.

Ilustración 64. Generar Justificante

### 9.2.2.6. Historial de faltas

Se pone a disposición de los padres un historial de las faltas cometidas por un determinado alumno. De este modo, podrán conocer los días que el alumno no ha cogido la ruta de transporte, así como el estado de dicha falta, es decir, si la falta está justificada o no.

Para ello, deben seleccionar el alumno del que se quiere consultar su historial.



Ilustración 65. Historial de faltas.

### 9.2.2.7. Cambio de contraseña

Debido a que las contraseñas generadas por la aplicación son poco intuitivas y con dificultad para ser recordadas, se pone a disposición de los padres la opción de cambiar la contraseña a una clave definida por el usuario, de forma que le sea más fácilmente memorizable.

Para realizar esta acción deberá introducir la contraseña actual y la nueva contraseña, que sustituirá a la anterior. Si la contraseña actual coincide con la almacenada en el sistema se producirá el cambio, y desde ese momento podrá acceder a la aplicación con la nueva contraseña.



The screenshot shows the user interface for the Proyecto CATEBus application. At the top left is a yellow school bus icon with the text 'ESCOLO BUS' on it. To its right, the text 'Proyecto CATEBus' is displayed. Further right, the user is greeted with 'Bienvenido Roberto Alvarez' and a 'Cerrar Sesión' button. Below this, there is a horizontal menu with five blue buttons: 'Información', 'Paradas', 'Retraso', 'Faltas de asistencia', and 'Cambiar Contraseña'. The 'Cambiar Contraseña' button is highlighted. Underneath the menu, the section 'Datos del Nuevo Responsable:' is shown. It contains two input fields: 'Contraseña actual:' and 'Nueva contraseña:'. At the bottom of this section, there are two orange buttons: 'Guardar' and 'Borrar'.

Ilustración 66. Cambio de Contraseña

### 9.2.3. Responsables de ruta

Los responsables de las rutas deberán ser dados de alta por el personal docente que maneje la aplicación. Para ello, éstos proporcionarán un nombre de usuario y una dirección de correo electrónico. En el momento en que sean dados de alta, recibirán en su correo electrónico el nombre de usuario facilitado y su contraseña de acceso a la aplicación.

Una vez hayan accedido a la aplicación, podrán comenzar a introducir la información de los alumnos que se suben en la ruta de transporte. Para ello, y por cada parada, se les mostrará el nombre y apellidos de los alumnos a los que se espera que asistan, y dos opciones disponibles, asistencia o falta, que se seleccionará según sea el caso. Una vez hayan terminado con esa parada, pulsarán el botón siguiente parada, a través del cual accederán a la información de la próxima parada.

Si alguno de los alumnos ha solicitado un retraso, se comunicará este hecho al lado del nombre del alumno, para que el responsable sepa que tiene que esperar como máximo cinco minutos a que el alumno llegue.



Ilustración 67. Marcación de alumnos

En el momento en el que no queden más paradas que mostrar, se mostrará la página de éxito en la operación, y permitirá al usuario desconectarse de la aplicación.

#### 9.2.4. Accesibilidad vía móvil

Para acceder a los servicios que ofrece la aplicación para dispositivos móviles, se debe introducir la dirección que permite mostrar la página de inicio de sesión adaptada para estos dispositivos (/Movil/index.jsp para padres y /Movil/indexAd.jsp para el personal docente).

Dicha página ofrece las mismas características que el inicio de sesión normal, con lo que el usuario deberá ingresar su nombre en la aplicación, su contraseña y pulsar el botón entrar.

##### 9.2.4.1. Menú Principal Padres

Una vez en la aplicación podrá ver un pequeño menú con las dos opciones disponibles, y la opción de cerrar la sesión del usuario y desconectarse de la aplicación.



Ilustración 68. Opciones móviles padres

#### 9.2.4.2. Establecer retraso.

Una vez pulsada la opción de establecer nuevo retraso, se carga la página en la cual se debe elegir al alumno al que se desea que se espere en la parada. Para ello, bastará con pinchar en su nombre.



Ilustración 69. Elección de alumno

Si la operación ha resultado exitosa, se redirige al usuario a una página en la que se indica tal hecho, y le muestra la opción de volver al menú principal.

9.2.4.3. Ver parada.

Como en el caso anterior, al pulsar esta opción se le muestra al usuario la opción de elegir el alumno del cual desea consultar los datos de su parada. Una vez seleccionado, se mostrará una nueva página en la que podrá observar, el nombre de la parada, su número según la ruta y la dirección de la misma, junto con una imagen que representa el mapa de la localización de la misma, para ayudar al posicionamiento de la parada al usuario.



Ilustración 70. Datos de la parada en móviles

9.2.4.4. Solicitar parada ocasional

Como en casos anteriores, al pulsar en esta opción se muestra al usuario el listado de los alumnos de los que quiere solicitar la parada ocasional. A continuación se le muestra dos

páginas más, en las cuales podrá elegir la ruta y la parada en las que se desea recoger al alumno.

A continuación, deberá introducir la fecha en la que se producirá dicho cambio ocasional. Se recuerda que el formato con el que se deberá incorporar la fecha a la aplicación es "DD/MM/AAAA", siendo *DD* el día de la semana representado por dos dígitos; *MM*, los dos dígitos que representan el mes; y *AAAA* cuatro dígitos que representan el año.

Ilustración 71. Introducción de fecha

Una vez recopilado todos los datos necesarios, al pulsar botón *guardar* se recopilarán todos ellos y se establecerá el cambio de parada para la fecha especificada. Si la operación ha resultado exitosa, se redirige al usuario a una página en la que se indica tal hecho, y le muestra la opción de volver al menú principal.

#### 9.2.4.5. Planificar falta de asistencia

Para planificar una falta de asistencia, el usuario deberá especificar en un primer momento el alumno que faltará a clase dicho día, y por lo tanto a la ruta. Una vez seleccionado, el usuario deberá introducir la fecha en la que se producirá la falta, del mismo modo que en la operación anterior. Para ello deberá realizarlo en el formato adecuado recordado anteriormente.

Si las operaciones han resultado exitosas, se redirige al usuario la página en la que se le indica tal hecho, y le muestra la opción de volver al menú principal.

#### 9.2.4.6. Menú principal del personal docente

Una vez en la aplicación podrá ver un pequeño menú con las cuatro opciones disponibles, y la opción de cerrar la sesión del usuario y desconectarse de la aplicación.



Ilustración 72. Menú principal personal docente

#### 9.2.4.7. Gestionar rutas

En esta opción accederemos a todas las operaciones que se pueden realizar con las rutas. Una vez accedida, aparece un menú en el que podemos seleccionar una de las rutas del sistema para realizar operaciones con ella, o bien añadir una nueva ruta.



Ilustración 73. Menú gestionar rutas

Si seleccionamos la opción de añadir ruta, accederemos a una página en la cual especificaremos el nombre de la misma. A continuación, elegiremos uno de entre los responsables de ruta libres registrados. Si la operación ha resultado exitosa, se redirige al usuario a una página en la que se indica tal hecho, y le muestra la opción de volver al menú principal.



The screenshot displays the mobile application interface for 'Proyecto CATEBus: Administración Móvil'. At the top, there is a header with a bus icon and the title. Below this is a section titled 'Datos Nueva Ruta'. Inside this section, there is a text input field labeled 'Nombre'. At the bottom of the form, there are two orange buttons: 'Continuar' and 'Cancelar'.

Ilustración 74. Nueva ruta móvil

#### 9.2.4.8. Gestionar paradas

Si se selecciona esta opción, se accederá a la sección en la que se gestionan los datos de las paradas de las rutas. En primer lugar se debe seleccionar la ruta sobre la que trabajar. A continuación, deberemos seleccionar en el menú la ruta sobre la que queremos operar, o bien la opción añadir nueva parada.



Ilustración 75. Menú gestionar paradas

Al seleccionar añadir parada, accederemos al formulario en el que deberemos introducir los datos de la misma y pulsar el botón *continuar*.



Ilustración 76. Nueva parada móvil

Si seleccionamos una de las paradas, accederemos a un nuevo menú en el que se muestran las opciones a realizar, mostrándose en cada caso los mismos formularios que para la versión normal, pero siguiendo el estilo y visualización adaptada a móviles.



Ilustración 77. Opciones de gestionar parada

Si las operaciones han resultado exitosas, se redirige al usuario la página en la que se le indica tal hecho, y le muestra la opción de volver al menú principal.

#### 9.2.4.9. Gestionar responsables de ruta

En esta opción accederemos a todas las operaciones que se pueden realizar con los responsables de rutas. Una vez accedida a la misma, aparece un menú en el que podemos seleccionar a uno de los responsables almacenados en el sistema para realizar operaciones con él, o bien añadir un nuevo responsable.



Ilustración 78. Menú gestionar responsables de ruta

Al seleccionar la opción de añadir nuevo responsable, accederemos al formulario en el que se solicita el nombre de usuario y la dirección del correo electrónico del mismo, a la cual se enviará la contraseña del sistema.

Ilustración 79. Nuevo responsable móvil

Si seleccionamos una de las paradas, accederemos a un nuevo menú en el que se nos muestran las opciones a realizar. En el caso de asignar el responsable a una ruta, accederemos a un nuevo menú para elegir la ruta de la cual se encargará dicho usuario. Si se selecciona eliminar, se elimina al responsable, siempre y cuando no tenga asignada ninguna ruta de transporte.

Ilustración 80. Opciones de gestionar responsables de ruta

Si las operaciones han resultado exitosas, se redirige al usuario la página en la que se le indica tal hecho, y le muestra la opción de volver al menú principal.

9.2.4.10. Gestionar padres y alumnos

Como en los apartados anteriores, se accede a un menú para seleccionar alguno de los padres del sistema, o la posibilidad de almacenar uno nuevo en el sistema.



Ilustración 81. Menú gestionar padres y alumnos

Si se quiere almacenar uno nuevo, se procede de la misma forma que para agregar un nuevo responsable que se ha explicado en el epígrafe anterior. En el caso de que se seleccione un padre existente, se le direccionará a una página en la que podrá elegir las opciones a realizar con dicho padre.



Ilustración 82. Opciones gestionar padres y alumnos

Al seleccionar la opción añadir alumno, accedemos a un formulario en el que se introducirán los datos del mismo, y se almacenarán pulsando el botón correspondiente. Si las

operaciones han resultado exitosas, se redirige al usuario la página en la que se le indica tal hecho, y le muestra la opción de volver al menú principal.



The screenshot shows the mobile application interface for 'Proyecto CATEBus: Administración Móvil'. At the top left, there is a small bus icon. The title 'Proyecto CATEBus: Administración Móvil' is displayed in a large, bold, black font. Below the title, there is a light blue header bar with the text 'Datos Nuevo Alumno:' in a bold, black font. Underneath this header, there are two input fields: the first is labeled 'Nombre:' and the second is labeled 'Apellidos:'. Both fields are empty and have a light blue border. Below the input fields, there is a red button with the text 'Guardar' and a grey button with the text 'Cancelar'.

Ilustración 83. Añadir nuevo alumno móvil

## 10. REFERENCIAS

---

[1] Real decreto de la CAM:

[http://www.madrid.org/dat\\_capital/servicio/s\\_publicostransportes.htm](http://www.madrid.org/dat_capital/servicio/s_publicostransportes.htm), (visitado en enero 2010.)

[2] Programa Papás: <http://educacion.jccm.es/delphos-papas/>, (visitado en enero

2010.)

[3] iCare: <http://www.orgamation.com/products/icare-management-software.html>,

(visitado en enero 2010.)

[4] Attendance Plus: [http://www.rediker.com/attendance\\_plus.html](http://www.rediker.com/attendance_plus.html), (visitado en

enero 2010.)

[5] JEE: <http://www.oracle.com/technetwork/java/javaee/overview/index.html>,

(visitado en enero 2010.)

[6] JDBC: [http://www.oracle.com/technetwork/java/javase/tech/index-jsp-](http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136101.html)

136101.html, (visitado en enero 2010.)

[7] MYSQL: <http://www.sun.com/software/products/mysql/features.jsp>, (visitado en

enero 2010.)

[8] Apache Tomcat: <http://tomcat.apache.org/>, (visitado en enero 2010.)

[9] JavaScript: [http://www.w3schools.com/js/js\\_intro.asp](http://www.w3schools.com/js/js_intro.asp), (visitado en octubre 2010.)

[10] Introducción a AJAX, Javier Eguíluz Pérez, junio 2008. (visitado en abril 2010.)

[11] Google Maps: <http://code.google.com/intl/es-ES/apis/maps/index.html>, (visitado

en septiembre 2010.)

[12] Java Mail: <http://www.oracle.com/technetwork/java/index-jsp-139225.html>,

(visitado en octubre de 2010)

[12] <http://www.oracle.com/technetwork/java/index-jsp-135995.html> (visitado en

octubre de 2010)

[13] <http://download.oracle.com/javaee/5/tutorial/doc/bnakh.html> (visitado en

octubre de 2010)

[14] COCOMO II: [http://sunset.usc.edu/csse/research/COCOMOII/cocomo\\_main.html](http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html)  
(visitado en octubre de 2010)

[14] <http://code.google.com/intl/es-ES/apis/maps/documentation/staticmaps/>  
(visitado en septiembre de 2010)

[15] <http://www.guia-ubuntu.org/index.php?title=MySQL> (visitado en octubre de 2010)

[16] iText: <http://itextpdf.com/itext.php> (visitado en abril de 2011)

[17] Java CSV Library: <http://sourceforge.net/projects/javacsv/> (visitado en abril de 2011)

[18] Commons fileUpload: <http://commons.apache.org/fileupload/> (visitado en abril de 2011)

[19] Commons IO: <http://commons.apache.org/io/> (visitado en abril de 2011)

[20] RFC 1867: Form-based File Upload in HTML <http://www.ietf.org/rfc/rfc1867.txt>  
(visitado en abril de 2011)