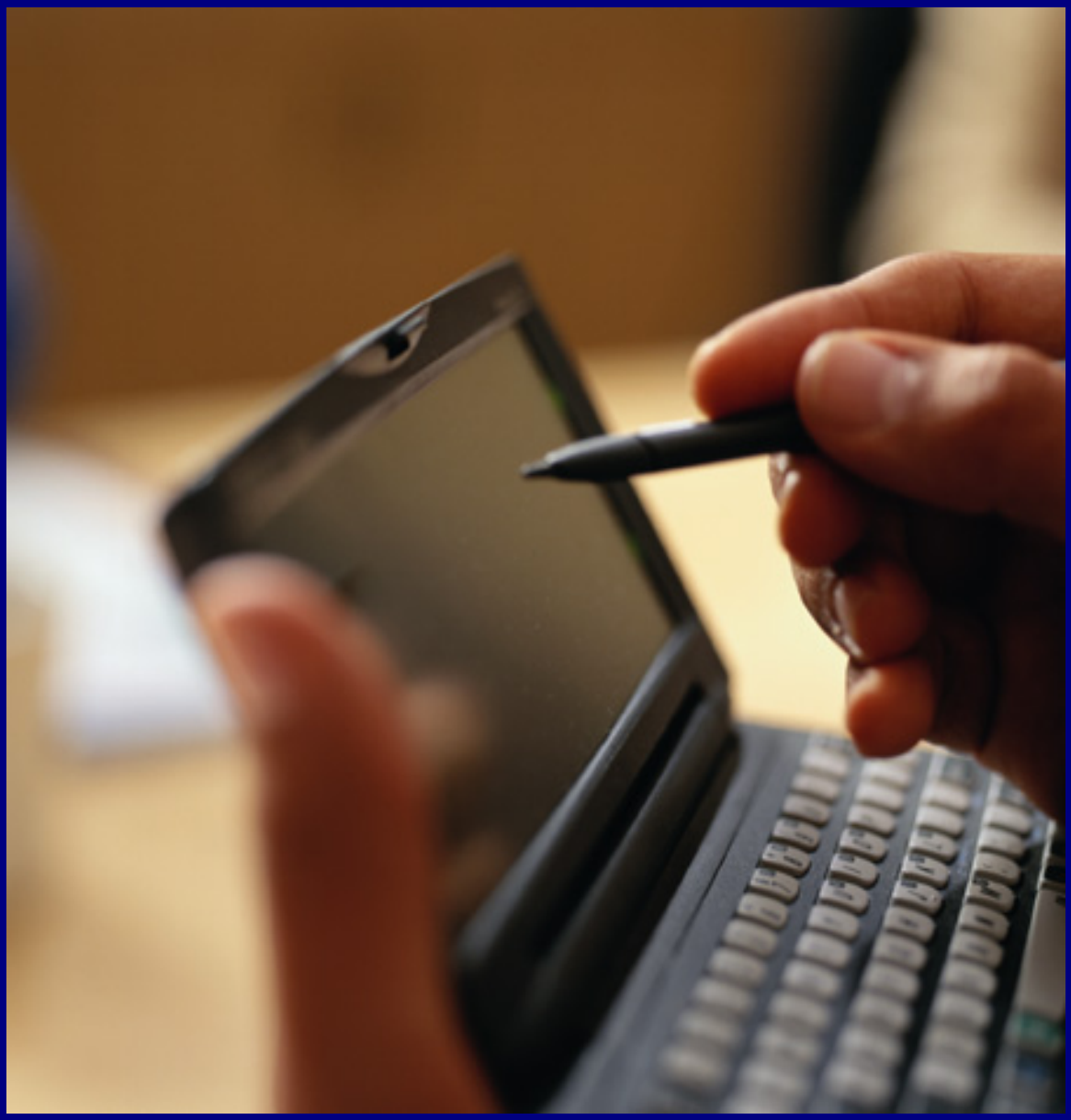
# Diseño e Implementación de un Sistema de Control de Proyectos

**César Manuel Viteri Ramírez**
ITIG

**Gladys Utrera Iglesias**

10 de Enero de 2004

*Trabajo Final de Carrera - Bases de Datos Relacionales*



# Diseño e Implementación de un Sistema de Control de Proyectos

*César Manuel Viteri Ramírez*

*A mi mujer Lidia, sin*
*cuyo apoyo y cariño constantes*
*no hubiera llegado hasta aquí.*

# 1 Resumen

El presente Trabajo Final de Carrera tiene como objetivo la realización del diseño e implementación de un sistema de control de proyectos para una organización tal que su estructura interna y funcionamiento sigan los parámetros especificados en el enunciado proporcionado por la consultora de la asignatura.

El sistema ha de recoger información sobre cuanto tiempo está trabajando cada trabajador y en qué, y cuales han sido los gastos mientras ha estado asignado a cada proyecto o tarea. De ésta forma, además de tener un control horario del trabajador, la organización sabrá perfectamente si la realización de un proyecto ha sido rentable y dispondrá durante su desarrollo de datos sobre la evolución de costes y horas para realizar las pertinentes actuaciones correctivas.

Para alcanzar este objetivo, se nos ha pedido lo siguiente:

- Diseñar la base de datos (Diagrama E/R), hacer el script de creación de las tablas, índices, etc. necesarios e implementar los procedimientos almacenados que se requieren para que las aplicaciones los utilicen
- Implementar un mecanismo de inicialización de la base de datos tal que simule un año de funcionamiento normal del sistema de control de proyectos, teniendo en cuenta los supuestos considerados en el enunciado. Este mecanismo funcionará haciendo llamadas a los procedimientos desarrollados, será aleatorio y se ajustará a lo que la realidad y al sentido común marcan, de modo que no será una inicialización cíclica de fácil detección.

De acuerdo con los requisitos, para desarrollar la base de datos hemos utilizado Oracle 8i y PL/SQL. Para el desarrollo de la Aplicación de Generación de Datos no se especificaba ninguna tecnología, y dada mi experiencia y conocimientos he seleccionado Java y la programación orientada a objetos para realizarla.

# 2 Índice

## 2.1 Índice de Contenidos

1 Resumen ....4
2 Índice ....5
2.1 Índice de Contenidos ....5
2.2 Índice de Figuras ....5
2.3 Índice de Tablas ....6
3 Cuerpo de la Memoria ....7
3.1 Introducción ....7
3.1.1 Justificación del TFC y contexto en el que se desarrolla ....7
3.1.2 Objetivos del TFC ....7
3.1.3 Enfoque y método seguido ....8
3.1.4 Planificación del proyecto ....8
3.1.5 Productos obtenidos ....12
3.2 Desarrollo de la Base de Datos ....14
3.2.1 Enfoque del Desarrollo de la Base de Datos ....14
3.2.2 Revisión de los Requerimientos ....14
3.2.3 Diseño Conceptual ....17
3.2.4 Diseño Lógico ....21
3.2.5 Diseño de Procedimientos Almacenados ....25
3.2.6 Instalación y Configuración del SGBD Oracle ....31
3.2.7 Construcción y Pruebas de la Base de Datos ....31
3.2.8 Seguridad, Roles y Privilegios ....31
3.2.9 Base de Datos Desarrollada ....31
3.3 Desarrollo de la Aplicación de Generación de Datos ....32
3.3.1 Enfoque del Desarrollo ....32
3.3.2 Revisión de los Requerimientos ....32
3.3.3 Componentes de la Aplicación de Generación de Datos ....33
3.3.4 Diagramas de Clases ....34
3.3.5 Especificación de la Lógica de Generación de Datos ....36
3.3.6 Construcción y Pruebas de la Aplicación de Generación de Datos ....37
3.3.7 Aplicación Desarrollada ....38
3.4 Conclusiones ....39
4 Glosario ....40
5 Bibliografía ....41
6 Anexos ....42
6.1 Instrucciones de Instalación de la Base de Datos ....42
6.2 Instrucciones de Instalación de la Aplicación de Generación de Datos ....42
6.3 Recursos e Infraestructura Empleados ....43
6.3.1 Hardware ....43
6.3.2 Software ....43

## 2.2 Índice de Figuras

Ilustración A: Planificación TFC ....10
Ilustración B: Listado de Tareas ....11
Ilustración C: Diagrama E/R de la Base de Datos ....17
Ilustración D: Modelo Lógico de la Base de Datos ....21
Ilustración E: Diagrama de Componentes de la Aplicación de Generación de Datos ....33
Ilustración F: Diagrama de Clases de las Entidades de Negocio ....34
Ilustración G: Diagrama de Clases de los Data Access Objects ....35
Ilustración H: Diagrama de Clases para el resto de Clases ....36

## 2.3 Índice de Tablas

Tabla 1: Hoja de Resumen de Fechas del Proyecto ..... 12
Tabla 2: Reglas de Negocio para la Base de Datos ..... 14
Tabla 3: Puntos Abiertos de los Requerimientos ..... 15
Tabla 4: Requerimientos para los Procedimientos Almacenados ..... 25
Tabla 5: Listado de Procedimientos Almacenados ..... 27
Tabla 6: Reglas de Negocio para la Aplicación de Generación de Datos ..... 32

# 3 Cuerpo de la Memoria

## *3.1 Introducción*

### 3.1.1 Justificación del TFC y contexto en el que se desarrolla

El trabajo final de carrera está pensado como un ejercicio de síntesis de los conocimientos adquiridos en otras asignaturas de la carrera y que requiere ponerlos en práctica conjuntamente en un proyecto concreto. Normalmente, y como ocurre en este caso, el TFC es un trabajo con aplicación práctica y vinculado al ejercicio profesional de la informática.

Los TFC del área de bases de datos relacionales pretenden dar la oportunidad de consolidar los conocimientos adquiridos en dicha área durante los estudios y ponerlos en práctica utilizando algún sistema de gestión de bases de datos con el cuál no haya trabajado durante los mismos.

Este trabajo cumple todos estos objetivos, ya que en él se ponen en práctica los conocimientos adquiridos en las asignaturas de Bases de datos I y Bases de datos II, empleándolos en el diseño y construcción de una base de datos relacional sobre el SGBD Oracle, de los procedimientos almacenados que usarán las aplicaciones para interactuar con ella, y de una aplicación de generación de datos que simula un año de funcionamiento del sistema y que hará uso de los procedimientos implementados.

Adicionalmente y de forma complementaria, en la planificación se utilizan los conocimientos adquiridos en Gestión de Organizaciones y Proyectos Informáticos, y en la construcción de la aplicación se usan los métodos y técnicas aprendidas en Ingeniería del Software I y Técnicas de Desarrollo de Software, y los conocimientos adquiridos en las asignaturas de programación orientada a objetos.

La realización exitosa de este trabajo final de carrera demostrará que he asimilado y aplicado con éxito de forma integrada todos estos conocimientos y habilidades, capacitándome para mi desempeño futuro como profesional de la informática.

### 3.1.2 Objetivos del TFC

El objetivo del presente Trabajo Final de Carrera consiste en la realización del diseño e implementación de un sistema de control de proyectos para una organización tal que su estructura interna y funcionamiento sigan los parámetros especificados en el enunciado proporcionado por el consultor de la asignatura.

El sistema ha de recoger información sobre cuanto tiempo está trabajando cada trabajador y en qué, y cuales han sido los gastos mientras ha estado asignado a cada proyecto o tarea. De ésta forma, además de tener un control horario del trabajador, la organización sabrá perfectamente si la realización de un proyecto ha sido rentable y dispondrá durante el desarrollo de datos sobre la evolución de costes y horas para realizar las pertinentes actuaciones correctivas.

Para alcanzar este objetivo, se nos ha pedido que realicemos lo siguiente:

- Diseñar la base de datos (Diagrama E/R), hacer el script de creación de las tablas, índices, etc. necesarios e implementar los procedimientos almacenados que se requieren para que las aplicaciones los utilicen
- Implementar un mecanismo de inicialización de la base de datos tal que simule un año de funcionamiento normal del sistema de control de proyectos, teniendo en cuenta los supuestos considerados en el enunciado. Este mecanismo funcionará haciendo llamadas a los procedimientos implementados, será aleatorio y se ajustará a lo que la realidad y al sentido común marcan, de modo que no será una inicialización cíclica de fácil detección.

En cuanto a la tecnología a emplear, para la base de datos se nos requiere que utilicemos Oracle 8i o superior. Para el desarrollo de la Aplicación de Generación de Datos no se especifica en el enunciado ninguna tecnología específica; dada mi experiencia y conocimientos, he seleccionado Java y la programación orientada a objetos para llevar a cabo su desarrollo.

### 3.1.3 Enfoque y método seguido

El primer paso para afrontar este proyecto ha sido realizar un análisis inicial de los requerimientos recogidos en el enunciado y elaborar un plan realista y ajustado que me permitiera alcanzar en plazo, alcance y calidad los objetivos establecidos en el apartado anterior.

Para conseguir una estimación ajustada del volumen de trabajo y una planificación lo más realista posible, se ha estimado conveniente utilizar el ciclo de vida clásico (o en cascada) con objeto de desarrollar el nuevo sistema. Considero que la utilización de este ciclo de vida, mi capacidad de análisis y mi experiencia en las herramientas y técnicas utilizadas han permitido que la estimación de los tiempos necesarios para las actividades fuera objetiva y que haya logrado su cumplimiento de forma adecuada.

En los procesos de desarrollo de los productos previstos en el trabajo se han utilizado los métodos, técnicas y herramientas aprendidos en el curso de los estudios y que más adecuados resultaban para obtener cada uno de los productos a desarrollar:

- La planificación se ha realizado de acuerdo con los métodos aprendidos en la asignatura de Gestión de Organizaciones y Proyectos Informáticos.
- Se han aplicado los métodos y técnicas de diseño de bases de datos relacionales aprendidos en la asignatura Bases de Datos I (diagramas Entidad/Relación, diseño lógico, normalización de tablas, etc.) para el desarrollo de la base de datos y la elaboración de sus scripts correspondientes.
- Los métodos de desarrollo aprendidos en Bases de Datos I y II se han utilizado para desarrollar los procedimientos almacenados requeridos para la realización de operaciones sobre la base de datos.
- Finalmente, se han empleado los métodos adquiridos en Ingeniería de Software I, Técnicas de Desarrollo de Software y las técnicas aprendidas en las diferentes asignaturas de programación y Base de Datos II para el desarrollo de aplicaciones orientadas a objetos que acceden a bases de datos relacionales (notación y diagramas UML, uso de patrones, técnicas de programación en java) para la aplicación de generación de datos.

El seguimiento de este enfoque y la aplicación de los métodos previstos ha permitido la consecución exitosa de los objetivos del trabajo final de carrera.

### 3.1.4 Planificación del proyecto

El plazo establecido para la entrega de los elementos resultantes del Trabajo Final de Carrera (Plan de Trabajo, Producto, Memoria y Presentación) fue originalmente el 5-01-2005, posteriormente aplazada por la consultora hasta el 10-01-2005.

La planificación elaborada para este proyecto, así como la estimación del esfuerzo, presentan un plan de trabajo adecuado para permitir alcanzar estos objetivos. Se adoptó como fecha inicial del proyecto el 23-09-2004, es decir, la fecha de entrega del enunciado, y se previó la realización de tres entregas parciales en las siguientes fechas con el fin de cumplir con las entregas de la evaluación continua:

- Entrega del Plan de Trabajo (PEC 1): 1-10-2004
- Entrega BD finalizada (PEC 2): 2-11-2004 (aplazado por el consultor previa solicitud, fecha real 3-11-2004)
- Entrega Aplicación de Generación de Datos finalizada (PEC 3): 9-12-2004 (aplazado por el consultor previa solicitud, fecha real 15-12-2004)

**Calendario para el desarrollo del proyecto**

La fecha inicial del proyecto es el 23-09-2004, mientras que la fecha límite era el 5-01-2005 (posteriormente modificada por el consultor al 10-01-2004). Esto supone un plazo de 105 días naturales (tras el ajuste 110) para la realización del trabajo; sin embargo, debido a mi disponibilidad de tiempo no me resultó posible fijar una jornada laboral de duración fija, ni designar a priori los días laborables de la semana.

Para la estimación de esfuerzos se consideró en general una dedicación prevista de entre 4 y 6 horas por cada semana natural del plan (7 días). No obstante, se realizaron ajustes en las duraciones de algunas tareas para tener en cuenta que su periodo de realización incluye fines de semana o festivos, ya que en esos días se pueden concentrar un mayor número de horas de trabajo.

Mi trabajo a tiempo completo me supone una dedicación semanal media de 50 horas, distribuida normalmente de lunes a viernes, y con frecuencia implica trabajo en fin de semana o viajes al extranjero, como ha sucedido en varias ocasiones a lo largo del semestre. Estos problemas han hecho que dispusiera de menos días para trabajar de los previstos inicialmente. En concreto, algunos fines de semana de noviembre o el puente de diciembre tuve que trabajar y no dispuse de ellos para la elaboración del TFC como inicialmente había previsto.

Aunque se ha cumplido el plazo global previsto para el proyecto, durante la ejecución del trabajo ha sido necesario pedir aplazamientos para las entregas de las PECs 2 y 3. Estos aplazamientos se han debido a problemas para equilibrar la carga de trabajo prevista con la dedicación a mi trabajo o a otras asignaturas, ya que la estimación de esfuerzo que realicé inicialmente ha resultado bastante ajustada.

**Elaboración de la planificación**

El proceso de elaboración de la planificación se presenta de forma detallada en el Plan de Trabajo que se entrega junto con esta memoria. Se realizaron las siguientes actividades para llevarlo acabo, siguiendo los métodos aprendidos en la asignatura de Gestión de Organizaciones y Proyectos Informáticos:

- Se identificaron los objetivos previstos, el alcance de los entregables y los hitos principales del proyecto.
- Se eligió una estructura de proyecto basada en el ciclo de vida en cascada, como se ha detallado en el apartado anterior para alcanzar estos objetivos.
- Se realizó una descomposición estructural en actividades (WBS o Work Breakdown Structure) adecuada, y se detalló el contenido y alcance de las actividades identificadas.
- Una vez identificadas las tareas y actividades requeridas, se realizó su planificación temporal, elaborando una tabla de precedencias entre actividades y estimando su duración de acuerdo con mi experiencia y conocimientos.

En el siguiente apartado se presenta el plan de trabajo propuesto inicialmente para la realización del trabajo.

## Planificación propuesta

Con la ayuda del MS-Project y considerando la fecha de inicio del proyecto 23-09-2004, la planificación que se propuso para el trabajo es la siguiente:

| Id | Nombre de tarea | Duración | Comienzo |
|---|---|---|---|
| 1 | **TFC BD** | **105 días** | **jue 23/09/04** |
| 2 | **01 - Planificación TFC (PEC 1)** | **9 días** | **jue 23/09/04** |
| 3 | 01.01 - Elaboración Plan de Trabajo TFC | 9 días | jue 23/09/04 |
| 4 | 01.02 - Entrega Plan de Trabajo TFC (PEC 1) | 0 días | vie 01/10/04 |
| 5 | **02 - Desarrollo BD (PEC 2)** | **32 días** | **sáb 02/10/04** |
| 6 | 02.01 - Revisión Requerimientos BD | 2 días | sáb 02/10/04 |
| 7 | **02.02 - Diseño BD** | **10 días** | **lun 04/10/04** |
| 8 | 02.02.01 - Diseño conceptual BD | 5 días | lun 04/10/04 |
| 9 | 02.02.02 - Diseño lógico BD | 5 días | sáb 09/10/04 |
| 10 | 02.03 - Instalación y Configuración SGBD ORACLE | 12 días | sáb 02/10/04 |
| 11 | **02.04 - Construcción y Pruebas BD** | **18 días** | **jue 14/10/04** |
| 12 | 02.04.01 - Construcción scripts creación BD | 3 días | jue 14/10/04 |
| 13 | 02.04.02 - Pruebas unitarias scripts creación BD | 3 días | dom 17/10/04 |
| 14 | 02.04.03 - Construcción disparadores BD | 3 días | mié 20/10/04 |
| 15 | 02.04.04 - Pruebas unitarias disparadores BD | 3 días | sáb 23/10/04 |
| 16 | 02.04.05 - Construcción procedimientos almacenados BD | 3 días | mar 26/10/04 |
| 17 | 02.04.06 - Pruebas unitarias procedimientos almacenados | 3 días | vie 29/10/04 |
| 18 | 02.05 - Revisión Final Documentación y Entregables BD | 2 días | lun 01/11/04 |
| 19 | 02.06 - Entrega BD Finalizada (PEC 2) | 0 días | mar 02/11/04 |
| 20 | **03 - Desarrollo Aplicación Generación de Datos (PEC 3)** | **37 días** | **mié 03/11/04** |
| 21 | 03.01 - Revisión Requerimientos Aplicación Generación de Dat | 2 días | mié 03/11/04 |
| 22 | 03.02 - Diseño Aplicación Generación de Datos | 5 días | vie 05/11/04 |
| 23 | 03.03 - Construcción Aplicación Generación de Datos | 9 días | mié 10/11/04 |
| 24 | 03.04 - Pruebas Unitarias Aplicación de Generación de Datos | 9 días | vie 19/11/04 |
| 25 | 03.05 - Pruebas Integradas Aplicación de Generación de Datos | 10 días | dom 28/11/04 |
| 26 | 03.06 - Revisión Final Documentación y Entregables Aplicación | 2 días | mié 08/12/04 |
| 27 | 03.07 - Entrega Aplicación Generación de Datos Finalizada (PE | 0 días | jue 09/12/04 |
| 28 | **04 - Elaboración Memoria y Presentación TFC (ENTREGA FIN/** | **27 días** | **vie 10/12/04** |
| 29 | 04.01 - Elaboración Memoria | 20 días | vie 10/12/04 |
| 30 | 04.02 - Elaboración Presentación TFC | 7 días | jue 30/12/04 |
| 31 | 04.03 - Entrega Memoria y Presentación TFC (ENTREGA FINAL | 0 días | mié 05/01/05 |

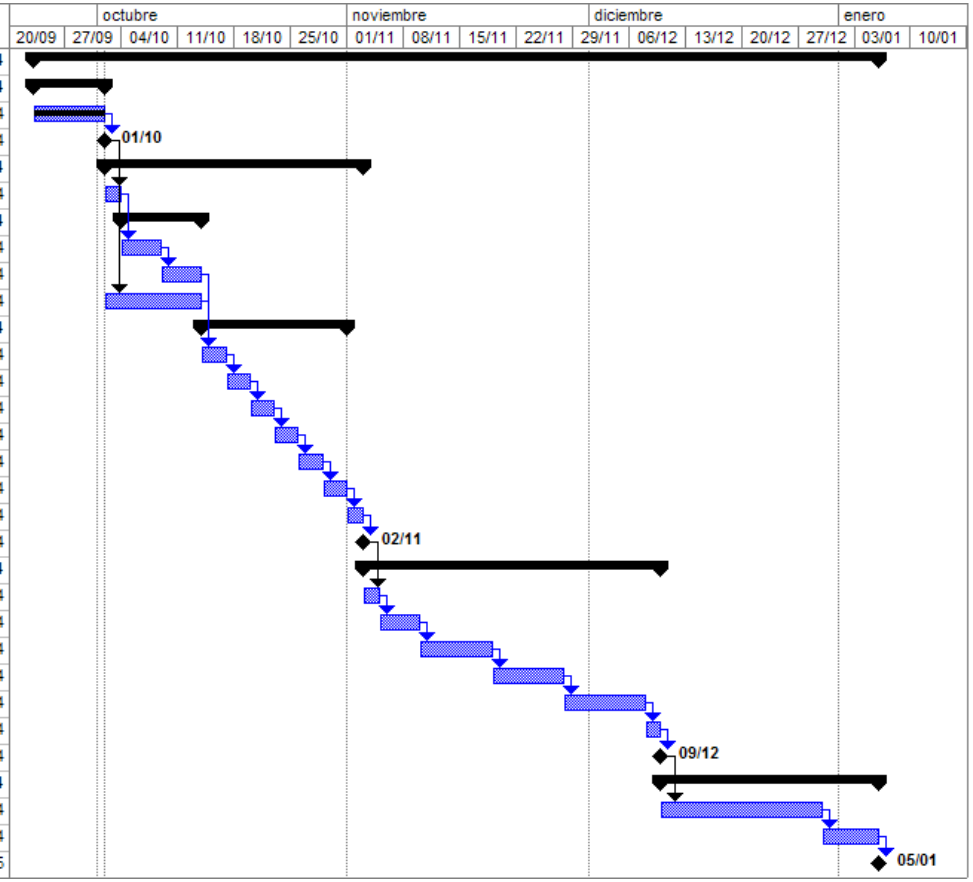



**Ilustración A: Planificación TFC**

Con esta planificación, el TFC finalizaría el 5-01-2005, con la entrega de los productos y documentos previstos. La modificación posterior por parte de la consultora de la fecha final de entrega hasta el 10-01-2005 no se ha recogido, y se ha utilizado para proporcionar 5 días de trabajo adicional a las tareas 04.01 y 04.02.

La planificación temporal de cada actividad y subactividad es la siguiente:

| Id | | Nombre de tarea | Duración | Comienzo | Fin | Predecesoras | Nombres de los rec |
|---|---|---|---|---|---|---|---|
| 1 | | **TFC BD** | **105 días** | **jue 23/09/04** | **mié 05/01/05** | | |
| 2 | | **01 - Planificación TFC (PEC 1)** | **9 días** | **jue 23/09/04** | **vie 01/10/04** | | |
| 3 | ✓ | 01.01 - Elaboración Plan de Trabajo TFC | 9 días | jue 23/09/04 | vie 01/10/04 | | |
| 4 | | 01.02 - Entrega Plan de Trabajo TFC (PEC 1) | 0 días | vie 01/10/04 | vie 01/10/04 | 3 | |
| 5 | | **02 - Desarrollo BD (PEC 2)** | **32 días** | **sáb 02/10/04** | **mar 02/11/04** | | |
| 6 | | 02.01 - Revisión Requerimientos BD | 2 días | sáb 02/10/04 | dom 03/10/04 | 4 | |
| 7 | | **02.02 - Diseño BD** | **10 días** | **lun 04/10/04** | **mié 13/10/04** | | |
| 8 | | 02.02.01 - Diseño conceptual BD | 5 días | lun 04/10/04 | vie 08/10/04 | 6 | |
| 9 | | 02.02.02 - Diseño lógico BD | 5 días | sáb 09/10/04 | mié 13/10/04 | 8 | |
| 10 | | 02.03 - Instalación y Configuración SGBD ORACLE | 12 días | sáb 02/10/04 | mié 13/10/04 | 4 | |
| 11 | | **02.04 - Construcción y Pruebas BD** | **18 días** | **jue 14/10/04** | **dom 31/10/04** | | |
| 12 | | 02.04.01 - Construcción scripts creación BD | 3 días | jue 14/10/04 | sáb 16/10/04 | 9;10 | |
| 13 | | 02.04.02 - Pruebas unitarias scripts creación BD | 3 días | dom 17/10/04 | mar 19/10/04 | 12 | |
| 14 | | 02.04.03 - Construcción disparadores BD | 3 días | mié 20/10/04 | vie 22/10/04 | 13 | |
| 15 | | 02.04.04 - Pruebas unitarias disparadores BD | 3 días | sáb 23/10/04 | lun 25/10/04 | 14 | |
| 16 | | 02.04.05 - Construcción procedimientos almacenados BD | 3 días | mar 26/10/04 | jue 28/10/04 | 15 | |
| 17 | | 02.04.06 - Pruebas unitarias procedimientos almacenados BD | 3 días | vie 29/10/04 | dom 31/10/04 | 16 | |
| 18 | | 02.05 - Revisión Final Documentación y Entregables BD | 2 días | lun 01/11/04 | mar 02/11/04 | 17 | |
| 19 | | 02.06 - Entrega BD Finalizada (PEC 2) | 0 días | mar 02/11/04 | mar 02/11/04 | 18 | |
| 20 | | **03 - Desarrollo Aplicación Generación de Datos (PEC 3)** | **37 días** | **mié 03/11/04** | **jue 09/12/04** | | |
| 21 | | 03.01 - Revisión Requerimientos Aplicación Generación de Datos | 2 días | mié 03/11/04 | jue 04/11/04 | 19 | |
| 22 | | 03.02 - Diseño Aplicación Generación de Datos | 5 días | vie 05/11/04 | mar 09/11/04 | 21 | |
| 23 | | 03.03 - Construcción Aplicación Generación de Datos | 9 días | mié 10/11/04 | jue 18/11/04 | 22 | |
| 24 | | 03.04 - Pruebas Unitarias Aplicación de Generación de Datos | 9 días | vie 19/11/04 | sáb 27/11/04 | 23 | |
| 25 | | 03.05 - Pruebas Integradas Aplicación de Generación de Datos | 10 días | dom 28/11/04 | mar 07/12/04 | 24 | |
| 26 | | 03.06 - Revisión Final Documentación y Entregables Aplicación | 2 días | mié 08/12/04 | jue 09/12/04 | 25 | |
| 27 | | 03.07 - Entrega Aplicación Generación de Datos Finalizada (PEC 3) | 0 días | jue 09/12/04 | jue 09/12/04 | 26 | |
| 28 | | **04 - Elaboración Memoria y Presentación TFC (ENTREGA FINAL)** | **27 días** | **vie 10/12/04** | **mié 05/01/05** | | |
| 29 | | 04.01 - Elaboración Memoria | 20 días | vie 10/12/04 | mié 29/12/04 | 27 | |
| 30 | | 04.02 - Elaboración Presentación TFC | 7 días | jue 30/12/04 | mié 05/01/05 | 29 | |
| 31 | | 04.03 - Entrega Memoria y Presentación TFC (ENTREGA FINAL) | 0 días | mié 05/01/05 | mié 05/01/05 | 30 | |

**Ilustración B: Listado de Tareas**

**Hoja de resumen de fechas del proyecto**

Los hitos fundamentales del proyecto quedan recogidos en la siguiente tabla. Las fechas de entrega reales que son posteriores a la inicialmente prevista fueron solicitadas y aprobadas previamente por la consultora de la asignatura:

**Tabla 1: Hoja de Resumen de Fechas del Proyecto**

| TFC de BDR de César Manuel Viteri Ramírez<br>Resumen | | |
|---|---|---|
| **Hito** | **Fecha Planificada** | **Fecha Entrega Real** |
| Fecha inicio | 23/09/2004 | 23/09/2004 |
| Fecha entrega plan de trabajo TFC (PEC 1) | 01/10/2004 | 01/10/2004 |
| Fecha entrega BD Finalizada (PEC 2) | 02/11/2004 | 03/11/2004 |
| Fecha entrega Aplicación Generación de Datos Finalizada (PEC 3) | 09/12/2004 | 15/12/2004 |
| Entrega Memoria y Presentación TFC (ENTREGA FINAL) | 05/01/2005 | 10/01/2005 |
| Fecha finalización | 05/01/2005 | 10/01/2005 |

### 3.1.5 Productos obtenidos

Los entregables que componen en este Trabajo Final de Carrera son los siguientes:

- El Plan de Trabajo, que recoge la planificación y estimación de las tareas necesarias para llevar a cabo los objetivos previstos.
- El Producto, que consiste en los desarrollos correspondientes a la Base de Datos (en adelante BD) y a la Aplicación de Generación de Datos, y su correspondiente documentación técnica asociada.
- La presente Memoria, que es el documento que sintetiza el trabajo realizado y muestra que se han alcanzado los objetivos propuestos. Incorpora toda la información relevante para comprender el problema planteado, la metodología utilizada para su resolución y detalla la solución elaborada.
- La Presentación, que resume de forma clara y concisa el trabajo realizado y los resultados obtenidos.

Las entregas parciales que se han realizado en el curso de la evaluación continua y que se contemplan en la planificación realizada han sido las siguientes:

**Planificación TFC (PEC 1)**

Siguiendo las metodologías de planificación aprendidas, se determinan las tareas necesarias para alcanzar los objetivos del proyecto y se estima el esfuerzo y los recursos necesarios para llevarlas a cabo.

**Desarrollo BD (PEC 2)**

Esta tarea comprende las siguientes actividades:

- Una revisión de los requerimientos proporcionados en el enunciado para la BD, para evaluar si son suficientes para comenzar el diseño e identificar los puntos abiertos que requieran una clarificación.
- La elaboración de un diseño conceptual de la BD por medio de diagramas E/R y su traslación a un diseño lógico.
- Paralelamente a estas tareas de diseño, se llevará a cabo la instalación y configuración del SGBD Oracle en los equipos asignados al proyecto, de modo que esté disponible al inicio del desarrollo.
- La construcción y pruebas de los objetos de la BD, que incluyen el esquema de tablas, los disparadores que garantizarán el cumplimiento de las reglas de negocio y los procedimientos almacenados que proporcionarán las operaciones exigidas.
- Una vez finalizado el desarrollo, se realizará una revisión del código desarrollado y de la documentación para verificar que son completos, homogéneos y consistentes.
- Finalmente, la realización de la entrega prevista por la evaluación continua.

**Desarrollo Aplicación Generación de Datos (PEC 3)**

Esta tarea comprende las siguientes actividades:

- Una revisión de los requerimientos proporcionados en el enunciado, para evaluar si son suficientes para comenzar el diseño e identificar los puntos abiertos que requieran una clarificación.
- La elaboración de un diseño de la aplicación por medio de diagramas UML.
- La construcción de las clases identificadas para la aplicación.
- La realización de las pruebas unitarias de las clases construidas.
- La realización de las pruebas integradas de funcionamiento de la aplicación una vez probadas las clases individuales.
- Una vez finalizado el desarrollo, se realizará una revisión del código desarrollado y de la documentación para verificar que son completos, homogéneos y consistentes.
- Finalmente, la realización la entrega prevista por la evaluación continua

Los resultados de estas tareas se describen con detalle en los siguientes apartados de este documento.

## 3.2 Desarrollo de la Base de Datos

### 3.2.1 Enfoque del Desarrollo de la Base de Datos

El diseño y desarrollo de la base de datos se ha llevado a cabo de acuerdo con los métodos y técnicas aprendidos en las asignaturas de Bases de Datos I y II.

- Se revisaron los requerimientos para asegurar su comprensión e identificar las principales entidades de negocio y las reglas que la base de datos tiene que plasmar. Se resolvieron las dudas mediante una serie de consultas a la consultora, que actúa como nuestro cliente.
- En paralelo a estas actividades se procedió a instalar y configurar el sistema gestor de bases de datos, definiendo un esquema de usuarios adecuado para garantizar la seguridad del sistema.
- A partir de los requerimientos se procedió a elaborar un modelo conceptual de la base de datos. Se identificaron los atributos de las entidades, las claves primarias que permiten identificar las instancias únicas y las interrelaciones que las vinculan. Se documentaron todas las decisiones adoptadas para la elaboración del modelo.
- Este modelo conceptual se transformó en un modelo lógico que pudiera ser transformado posteriormente a su vez en un modelo físico. Se aplicaron los principios aprendidos en Bases de Datos I y técnicas de normalización para obtener un modelo que proporcionase una base sólida para la aplicación.
- Finalmente, se plasmó este modelo en sentencias de definición de la base de datos por medio de una herramienta de desarrollo y administración de base de datos.
- Los procedimientos almacenados que implementan las operaciones necesarias para las aplicaciones que van a utilizar esta base de datos fueron identificados mediante un análisis de las operaciones asociadas a cada entidad de negocio identificada. Se identificaron también las reglas que cada procedimiento debía garantizar.
- Una vez definidos, se construyeron y probaron de forma unitaria los procedimientos almacenados.

### 3.2.2 Revisión de los Requerimientos

Los requerimientos proporcionados en el enunciado fueron analizados para la realización de las tareas de diseño y asegurar el cumplimiento de todos sus puntos. En aquellos puntos o reglas que no quedaban claramente definidas o donde surgía una duda sobre su interpretación utilicé frecuentemente mi experiencia de 6 años utilizando dos aplicaciones similares de reporte de horas y gastos como criterio para comparar y validar mi comprensión de los requerimientos y del problema de negocio examinado.

Las reglas de negocio establecidas por el enunciado son las siguientes:

**Tabla 2: Reglas de Negocio para la Base de Datos**

| Reglas de Negocio para la Base de Datos |
|---|
| Se pide al estudiante que realice el diseño e implementación de un sistema de control de proyectos para una organización tal que su estructura interna y funcionamiento sigan los siguientes parámetros:<br>• La organización está implantada en diferentes ciudades.<br>• En cada centro de trabajo de la organización hay n departamentos.<br>• Cada departamento tiene una serie de empleados que no pueden estar en más de un departamento.<br>• Cada empleado tiene una función tipificada, jefe de departamento, jefe de proyecto, dibujante, ingeniero, economista, …<br>• Un empleado puede cambiar de departamento, de centro de trabajo y de función, cuando cambie alguna de éstas características, no se deberá modificar los datos del trabajador sino que se dará de alta otra vez y “el trabajador antiguo” se “desactivará”.<br>• Dentro de cada departamento hay jefes de proyectos, los cuales tendrán asignados uno o más proyectos (o tareas). |

**Reglas de Negocio para la Base de Datos**

- Cada proyecto sólo puede tener asignado un jefe de proyecto.
- Cada proyecto tendrá n trabajadores del departamento al cual pertenece el jefe de proyecto.
- Cada trabajador podrá estar asignado a n proyectos.
- Cada trabajador podrá entrar y salir del centro de trabajo n veces y las mismas quedarán registradas.
- A cada proyecto se pueden asignar gastos de cualquier tipo, y cada asignación de gastos que se haga estará relacionado con una persona.
- Cada trabajador tendrá asignado un costo por hora.
- Cada proyecto se ha presupuestado y está aceptado por el cliente, por lo tanto antes de comenzar ya se sabe que cobrará. Este dato es importante y el sistema deberá almacenarlo al abrir el proyecto.
- Para controlar los tipos de gastos se habrá de montar un sistema de tablas tal que sea sencillo tipificar los gastos.
- La dirección de la organización dará permiso a cada tipo de trabajador sobre cada tipo de gasto que exista, de manera que si no tiene permisos sobre un tipo no se podrán cargar los gastos.
- Una vez un proyecto ha finalizado no se podrá realizar operaciones sobre él, es decir asignarle un trabajador o gastos o el que sea.

El sistema ha de recoger información sobre cuanto tiempo el trabajador está trabajando y en qué, y cuales han sido los gastos mientras ha estado asignado a cada proyecto o tarea. En ésta forma, además de tener un control horario del trabajador, la organización sabrá perfectamente si la realización de un proyecto ha sido rentable y dispondrá de datos en el medio del desarrollo de costos y horas para realizar las actuaciones correctivas.

(En general una baja de cualquier elemento no implicará una eliminación de la tabla sino una marca de borrado).

Como consecuencia de esta revisión de los requerimientos se identificaron una serie de puntos abiertos que fueron remitidos a la consultora para su contestación, tal como hubiéramos hecho con un cliente. Detallamos su resolución en la siguiente tabla.

**Tabla 3: Puntos Abiertos de los Requerimientos**

| ID | Descripción Punto Abierto | Resolución |
|---|---|---|
| 1 | ¿Puede haber varios centros de trabajo en una misma ciudad? | No |
| 2 | Los departamentos quedan perfectamente identificados con su nombre, o es necesario hacer referencia al centro de trabajo al que pertenecen para identificarlos perfectamente? | Es necesario hacer referencia al centro de trabajo. |
| 3 | ¿Debe haber un jefe de departamento por cada departamento? Se menciona en las funciones, pero no se dice nada al respecto. | Existe siempre un jefe de departamento por cada departamento. |
| 4 | Se indica que los jefes de proyecto tendrán asignados uno o más proyectos o tareas. ¿No es posible que en un momento dado un jefe de proyecto no tenga proyectos asignados? | Tendrá uno o más proyectos, si no tuviera tiene tareas asignadas. |
| 5 | ¿Es necesario que los proyectos tengan algún empleado aparte del jefe del proyecto, o puede haber proyectos que no tengan asignado ningún otro empleado? | Puede no tener otro empleado asignado. |

| ID | Descripción Punto Abierto | Resolución |
|---|---|---|
| 6 | No se indica cómo se determina el costo por hora que un empleado tiene asignado. Puede ser un atributo del empleado, o estar determinado por la función que le corresponde, o puede ser un valor acordado caso por caso para cada proyecto. No es posible modelarlo sin saber qué política se sigue a este respecto. | El costo por hora de un empleado está asociado a la función del mismo. Si ésta cambia, cambiará el costo. |
| 7 | ¿Puede un empleado estar asignado a dos proyectos simultáneamente en el tiempo? Esto es, con dedicación parcial. Hay que tener en cuenta que esto complicaría bastante el llevar el control de horas, ya que habría que controlar no sólo el inicio y fin de cada asignación, sino también el porcentaje de dedicación. | Un empleado puede estar asignado a varios proyectos. Se llevará control de horas dedicadas a cada proyecto. |

Una vez validada nuestra comprensión de los requerimientos, procedimos a realizar el diseño de la BD.

### 3.2.3 Diseño Conceptual

**Diagrama E/R**

El punto de partida para el diseño conceptual de la base de datos es la realización con ayuda de Visio 2000 de un diagrama E/R:

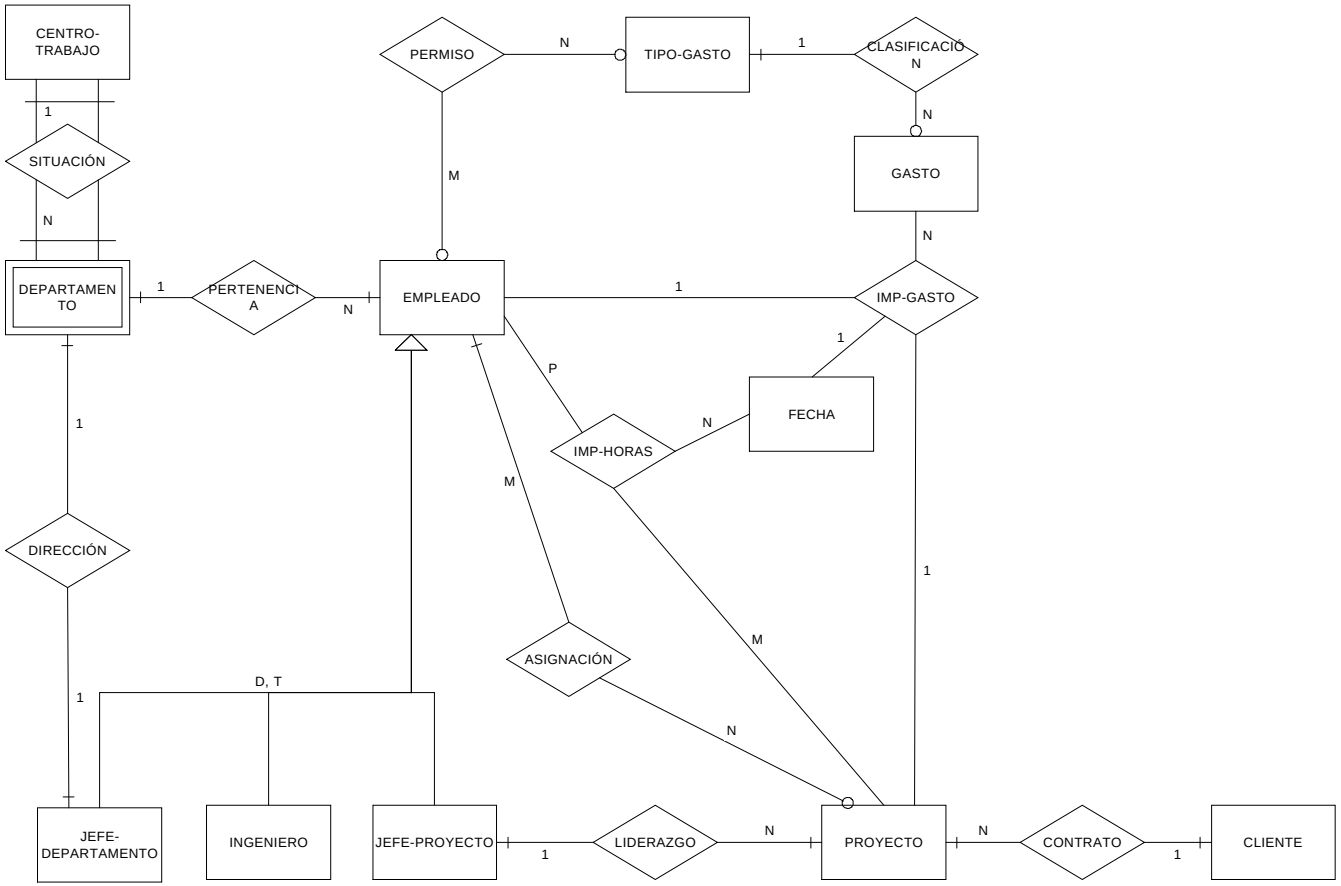



**Ilustración C: Diagrama E/R de la Base de Datos**

**Atributos de las Entidades**
A continuación se detallan los atributos que se han definido para las entidades identificadas en el diagrama E/R presentado en el apartado anterior. Los campos que componen la clave primaria identificada para cada entidad se han subrayado.

**CENTRO**
nombre_centro, fecha_alta_cen, fecha_baja_cen, cod_estado

**DEPARTAMENTO**
nombre_departamento, fecha_alta_dep, fecha_baja_dep, cod_estado

**EMPLEADO**
cod_empleado, fecha_alta_emp, fecha_baja_emp, cod_estado, NIF, nombre_emp, apellido, coste_hora, coste_hora_extra

**TIPO_GASTO**
cod_tipo_gasto, descripción

**GASTO**
cod_gasto, importe, descripción

**FECHA**
fecha, tipo_jornada

**PROYECTO**
cod_proyecto, nombre_proy, fecha_alta, fecha_baja, cod_estado, presupuesto

**CLIENTE**
cod_cliente, nombre_cli, documento, tipo_documento, dirección, fecha_alta, fecha_baja, cod_estado

**Atributos de las Interrelaciones**
A continuación se detallan los atributos que se han definido para las interrelaciones identificadas en el diagrama E/R presentado.

**IMP_HORAS**
horas, tipo, fecha_imp

**IMP_GASTO**
fecha_imp

**LIDERAZGO**
fecha_inicio_lid, fecha_fin_lid

**ASIGNACIÓN**
fecha_inicio_asig, fecha_fin_asig

**PERMISO**
fecha_inicio_per, fecha_fin_per

**Decisiones de Diseño y Observaciones sobre las Entidades**
En este apartado se describen las decisiones adoptadas para el modelado de cada una de las entidades.

**CENTRO**
Se ha confirmado que en una ciudad no puede haber más de un CENTRO de trabajo. En consecuencia, se ha optado por utilizar el nombre de la ciudad para identificar cada centro. La fecha de alta es parte de la clave primaria para tener en cuenta los históricos.

Puesto que el centro es referenciado por otras entidades, en la baja se barajan las siguientes opciones:

- Dar de baja en cascada todo lo relacionado
- No permitir mientras haya departamentos activos
- Permitir que se indique un centro al que se transfiere todo lo relacionado.

La opción seleccionada es no permitir mientras haya departamentos activos.

**DEPARTAMENTO**
DEPARTAMENTO no queda perfectamente identificado por su nombre o identificador. Hace falta hacer referencia al CENTRO al que pertenecen para identificarlos perfectamente, y en consecuencia se representa DEPARTAMENTO como una entidad débil. La fecha de alta es parte de la clave primaria para tener en cuenta los históricos.

Al dar de alta un departamento será necesario dar de alta o indicar un empleado que será su jefe. Puesto que el departamento es referenciado por otras entidades, en la baja se barajan las siguientes opciones:

- Dar de baja en cascada todo lo relacionado
- No permitir mientras haya empleados activos
- Permitir que se indique un departamento al que se transfiere todo lo relacionado.

La opción seleccionada es no permitir mientras haya empleados activos.

**EMPLEADO**
EMPLEADO tiene que tener una fecha de alta y fecha de baja para reflejar los históricos de cambio de departamento, centro de trabajo y función. Esto implica que la fecha de alta es parte de la clave primaria del empleado. Se ha incluido un indicador (cod_estado) que señala que registro del empleado es el más reciente (activo), para facilitar su localización.

Para la clave primaria del empleado se propone el uso de un número de empleado autogenerado secuencial que se asigna a cada persona al entrar en la empresa, evita los problemas con nombres comunes (José García Pérez, por ejemplo), y evita que se manejen datos propios del empleado como el NIF.

Un EMPLEADO puede cambiar de un departamento a otro, independientemente del centro de trabajo al que pertenezcan, y puede cambiar de una función a otra.

Hay que considerar que un cambio de función o la baja de un empleado pueden provocar que un proyecto o un departamento queden sin jefe o que haya dos jefes para un departamento en un momento dado, o que un empleado de un departamento distinto al del jefe de proyecto esté asignado a un proyecto. Es necesario tener esto en cuenta al realizar los cambios. La baja del último empleado de un departamento debería controlarse también.

No hay límite al número de veces que un empleado puede cambiar de centro de trabajo.

La lista de entidades subclase de EMPLEADO que se muestra en el diseño conceptual no es exhaustiva.

Las subclases de empleado no tienen campos específicos, pero el coste relacionado con cada empleado depende de la función que desempeña.

**TIPO_GASTO**
Ninguna.

**GASTO**
Ninguna.

**FECHA**
Esta entidad permite a la empresa definir un calendario laboral, y distinguir a través del atributo tipo_jornada las jornadas laborables, los fines de semana y los festivos designados.

**PROYECTO**
En el diseño conceptual no se refleja la restricción de qué los N trabajadores de un proyecto deben pertenecer al mismo departamento que el jefe de proyecto. Esto tendrá que controlarse mediante un disparador o regla de negocio.

Un proyecto puede no tener otros empleados aparte del jefe de proyecto asignado.

PROYECTO tiene un atributo presupuesto, y un CLIENTE al que corresponde.

No se refleja la restricción de que los gastos de un empleado en un proyecto tienen que ser de un tipo sobre el que tiene permiso. Esto tendrá que controlarse mediante un disparador o regla de negocio.

Tampoco se contempla que sólo podrán imputar horas a un proyecto los empleados que están asignados a él. También tendrán que contemplarse con un disparador o regla de negocio.

Una vez que un PROYECTO se cierra, no se pueden cargar gastos ni asignar trabajadores. Esto tendrá que controlarse mediante un disparador o regla de negocio.

**CLIENTE**
Se considera que un CLIENTE tiene por lo menos un proyecto relacionado, no se considera como tal hasta que no contrata uno.

**Decisiones de Diseño y Observaciones de las Interrelaciones**
En este apartado se describen las decisiones adoptadas para el modelado de cada una de las interrelaciones.

**IMP_HORAS**
El atributo horas tendrá que tener un valor entre 1 y 24. Las horas que imputa un empleado pueden ser normales o extraordinarias. El atributo fecha_imp registra la fecha en que se realiza la imputación.

**IMP_GASTO**
El atributo fecha_imp registra la fecha en que se realiza la imputación.

**LIDERAZGO**
La baja o cambio de función de un jefe de proyecto sólo se podrán hacer tras asignar a otro jefe de proyecto del mismo departamento los proyectos activos que lidere.

**ASIGNACIÓN**
Las asignaciones de un empleado deben haber sido terminadas antes de poder darlo de baja.

**PERMISO**
Los permisos se deben dar de baja al darse de baja un cliente o modificarse su centro de trabajo, departamento o función.

### 3.2.4 Diseño Lógico

**Modelo Lógico de Base de Datos**

El siguiente apartado recoge como transformamos en una base de datos relacional el modelo conceptual presentado en el apartado anterior, y las decisiones que adoptamos al hacerlo.

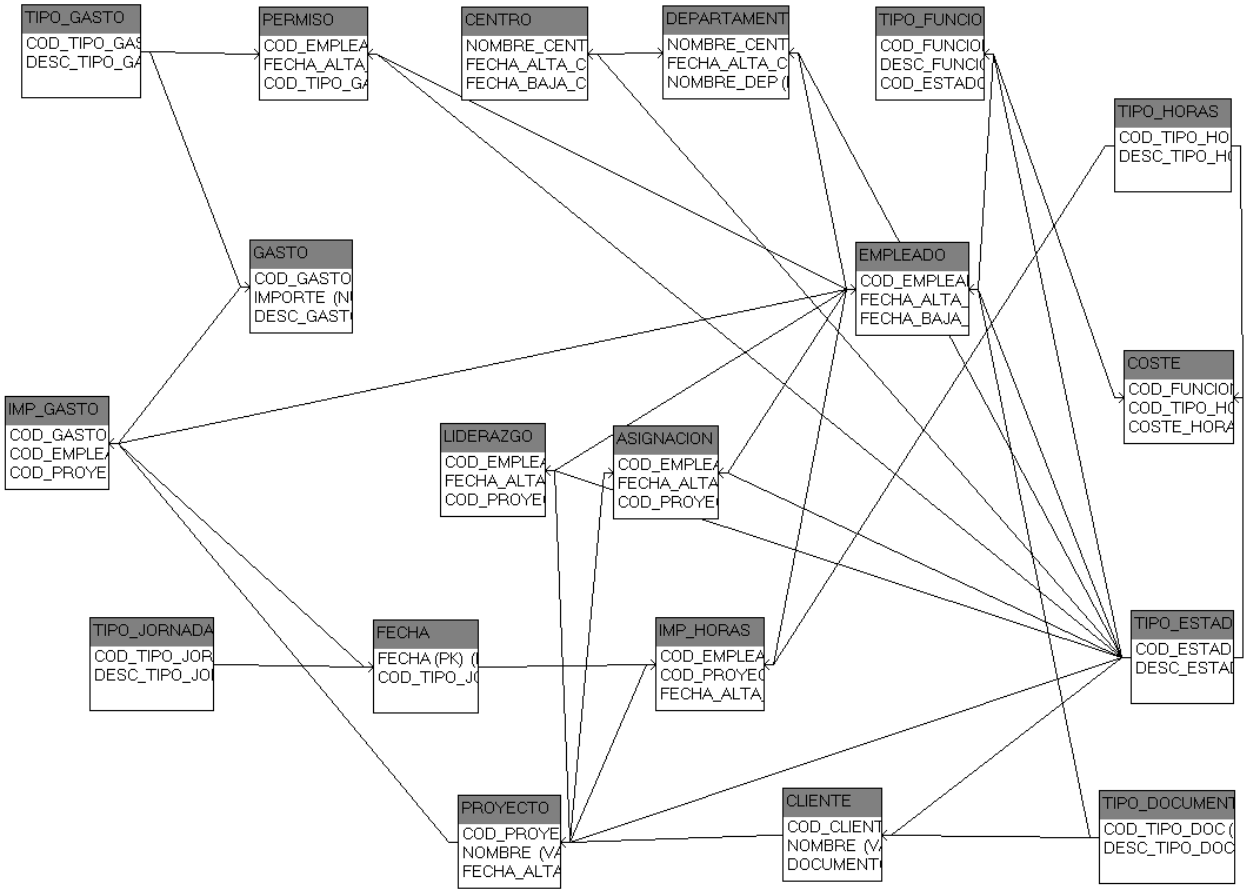



**Ilustración D: Modelo Lógico de la Base de Datos**

**Transformación de las Entidades e Interrelaciones**

A continuación detallamos los campos de las tablas a las que se traducen las entidades e interrelaciones identificadas en el diseño conceptual, y las referencias entre tablas por medios de claves foráneas. Los campos que forman parte de la clave primaria de las tablas se presentan subrayados.

**TIPO_ESTADO**
(cod_estado, desc_estado)

**TIPO_FUNCIÓN**
(cod_función, desc_funcion, cod_estado)
donde [cod_estado] referencia TIPO_ESTADO

**TIPO_GASTO**
(cod_tipo_gasto, desc_tipo_gasto)

**TIPO_DOCUMENTO**
(cod_tipo_doc, desc_tipo_documento)

**TIPO_JORNADA**
(cod_tipo_jornada, desc_tipo_jornada)

**TIPO_HORAS**
(cod_tipo_horas, desc_tipo_horas)

**CENTRO**
(nombre_centro, fecha_alta_cen, fecha_baja_cen, cod_estado)
donde [cod_estado] referencia TIPO_ESTADO

**DEPARTAMENTO**
(nombre_centro, fecha_alta_cen, nombre_dep, fecha_alta_dep, fecha_baja_dep, cod_estado, cod_empleado, fecha_alta_emp)
donde [nombre_centro, fecha_alta_cen] referencia CENTRO,
[cod_estado] referencia TIPO_ESTADO,
y [cod_empleado, fecha_alta_emp] referencia EMPLEADO

**EMPLEADO**
(cod_empleado, fecha_alta_emp, fecha_baja_emp, cod_estado, documento, cod_tipo_doc, nombre_emp, apellido, nombre_centro, fecha_alta_cen, nombre_dep, fecha_alta_dep, cod_función)
donde [nombre_centro, fecha_alta_cen, nombre_dep, fecha_alta_dep] referencia DEPARTAMENTO,
[cod_estado] referencia TIPO_ESTADO,
[cod_tipo_doc] referencia TIPO_DOCUMENTO
y [cod_función] referencia TIPO_FUNCIÓN

**PERMISO**
(cod_empleado, fecha_alta_emp, cod_tipo_gasto, fecha_inicio_per, fecha_fin_per, cod_estado)
donde [cod_empleado, fecha_alta_emp] referencia EMPLEADO,
[cod_estado] referencia TIPO_ESTADO
y [cod_tipo_gasto] referencia TIPO_GASTO

**GASTO**
(cod_gasto, importe, descripción, cod_tipo_gasto)
donde [cod_tipo_gasto] referencia TIPO_GASTO

**FECHA**
(fecha, cod_tipo_jornada)
donde [cod_tipo_jornada] referencia TIPO_JORNADA

**CLIENTE**
(cod_cliente, nombre_cli, documento, cod_tipo_doc, dirección_cli, fecha_alta_cli, fecha_baja_cli, cod_estado)
donde [cod_estado] referencia TIPO_ESTADO
y [cod_tipo_doc] referencia TIPO_DOCUMENTO

**PROYECTO**
(cod_proyecto, nombre_proy, fecha_alta, fecha_baja, cod_estado, presupuesto, cod_cliente)
donde [cod_cliente] referencia CLIENTE
y [cod_estado] referencia TIPO_ESTADO

**IMP_GASTO**
(cod_gasto, cod_empleado, fecha_alta_emp, cod_proyecto, fecha, fecha_imp)
donde [cod_gasto] referencia GASTO
y [cod_empleado, fecha_alta_emp] referencia EMPLEADO
y [cod_proyecto] referencia PROYECTO
y [fecha] referencia FECHA

**IMP_HORAS**
(cod_empleado, fecha_alta_emp, cod_proyecto, fecha, cod_tipo_horas, fecha_imp, horas)
donde [cod_empleado, fecha_alta_emp] referencia EMPLEADO
[cod_proyecto] referencia PROYECTO,
[fecha] referencia FECHA
[tipo_horas] referencia TIPO_HORAS

**LIDERAZGO**
(cod_empleado, fecha_alta_emp, cod_proyecto, fecha_inicio_lid, fecha_fin_lid, cod_estado)
donde [cod_empleado, fecha_alta_emp] referencia EMPLEADO,
[cod_estado] referencia TIPO_ESTADO
y [cod_proyecto] referencia PROYECTO

**ASIGNACIÓN**
(cod_empleado, fecha_alta_emp, cod_proyecto, fecha_inicio_asig, fecha_fin_asig, cod_estado)
donde [cod_empleado, fecha_alta_emp] referencia EMPLEADO,
[cod_estado] referencia TIPO_ESTADO
y [cod_proyecto] referencia PROYECTO

**COSTE**
(cod_funcion, cod_tipo_horas, fecha_alta_coste, fecha_fin_coste, cod_estado, coste_hora)
donde [cod_funcion] referencia TIPO_FUNCIÓN,
[cod_tipo_horas]referencia TIPO_HORAS
y [cod_estado] referencia TIPO_ESTADO

**Decisiones de Diseño Lógico y Comentarios**
En este apartado se describen las decisiones adoptadas la transformación a un modelo lógico de cada una de las entidades o interrelaciones.

**TIPO_ESTADO**
Se añade para facilitar la gestión de los estados existentes.

**TIPO_FUNCIÓN**
Se han separado de la tabla EMPLEADO las características propias de cada tipo de función para facilitar su mantenimiento y evitar redundancias. Además, se evita que se pierdan los valores si en un momento dado no hay empleados con una función dada. Las funciones se crean desactivadas, y deben activarse, pero no se podrán activar hasta que no tengan todos los costes para todos los tipos de horas.

**TIPO_GASTO**
Ninguna.

**TIPO_DOCUMENTO**
Se añade para facilitar la gestión de los distintos tipos de documentos identificativos existentes.

**TIPO_JORNADA**
Se añade para facilitar la gestión de los tipos de jornada existentes en el calendario laboral de la empresa.

**TIPO_HORAS**
Se añade para facilitar la gestión de los tipos de horas trabajadas contempladas por la empresa.

**CENTRO**
Ninguna.

**DEPARTAMENTO**
La referencia a EMPLEADO indica el jefe de departamento, por tanto el tipo de ese empleado debe ser. Obsérvese que hay un deadlock de definición entre esta tabla y empleado, ya que cada una requiere una referencia obligatoria a la otra. Solucionamos esto haciendo en la implementación que los campos que referencian al jefe de departamento no sean requeridos y garantizando en el proceso de alta de departamento y de su jefe que estos campos siempre quedan informados y posteriormente sólo se modifican de forma consistente.

**EMPLEADO**
Dado que las subclases no tienen atributos, se ha optado por representar las distintas funciones mediante un atributo cod_funcion. Sí que hay valores que dependen de la función, y para facilitar su mantenimiento se han trasladado a una tabla aparte.

**PERMISO**
Se ha incluido una fecha de alta y un estado en la clave del permiso, ya que es razonable que un permiso pueda ser concedido y revocado más de una vez y que es relevante mantener el histórico y verificar si un gasto era aplicable en el momento en que se produjo.

**GASTO**
Los gastos se identifican con un código autogenerado. Es importante ver que la relación con el empleado que lo origina se establece a través de IMP_GASTO.

**FECHA**
Ninguna.

**CLIENTE**
Ninguna.

**PROYECTO**
Ninguna.

**IMP_GASTO**
Se añade para facilitar la auditabilidad la fecha en que se realiza la imputación.

**IMP_HORAS**
Se añade para facilitar la auditabilidad la fecha en que se realiza la imputación.

**LIDERAZGO**
En este caso se han añadido las fechas en que un jefe determinado lidera un proyecto, ya que se contempla la eventualidad de que un jefe se dé de baja de la empresa o sea sustituido por otro.

**ASIGNACIÓN**
Se contempla también el caso de que un empleado esté asignado a un proyecto (particularmente en el caso de uno de larga duración), salga de él y luego regrese.

**COSTE**
Se ha definido esta tabla para poder gestionar las tarifas asociadas a cada tipo de función para cada tipo de hora trabajada (normal, extraordinaria, etc.) que la empresa contempla. La tabla permite gestionar un histórico para evitar que un cambio de tarifas provoque modificaciones en los costes históricos de un proyecto.

### 3.2.5 Diseño de Procedimientos Almacenados

Los requerimientos del enunciado en lo referente a procedimientos almacenados son los siguientes:

**Tabla 4: Requerimientos para los Procedimientos Almacenados**

| Requerimientos para los Procedimientos Almacenados |
|---|
| El estudiante deberá diseñar […] e implementar los procedimientos almacenados que se requieren para que las aplicaciones los utilicen. Estos procedimientos deberán ser como mínimo:<br>• Alta de proyecto<br>• Cierre de proyecto<br>• Alta, Baja, Modificación del Centro, Departamento, Gasto, Tipo de Gasto …<br>• Asignación / desasignación de trabajadores y gastos al proyecto.<br>• Trabajador entra a trabajar.<br>• Trabajador sale de trabajar.<br>• Alta de un trabajador.<br>• Baja de un trabajador.<br>• Modificación de los datos del trabajador.<br>• Cambio de centro/departamento/tipo de trabajador.<br>• Consulta de trabajador, departamento …<br><br>[…]<br><br>Los procedimientos tendrán las siguientes características que no podrán ser obviadas por el futuro ingeniero informático que desarrolle el sistema:<br>• Tendrá como mínimo un parámetro de salida denominado RST de tipo string el cual tomará el valor 'OK' si la ejecución ha sido exitosa y 'ERROR: Tipo de error' en caso de fracaso.<br>• Tendrá utilización de excepciones, por lo tanto el cuerpo del procedimiento tendrá el aspecto:<br>Procedure ( param1 (entrada o salida) tipo,<br>param2 (entrada o salida) tipo,<br>….<br>Rst out tipo) is<br>Begin<br>….<br>….<br>commit;<br>except<br>rollback;<br>cierre de las excepciones<br>end |

Para identificar los procedimientos almacenados necesarios para la base de datos, elaboramos una tabla que cruza las entidades definidas en la base de datos con los cuatro tipos generales de operaciones: Consulta, Alta, Modificación y Baja.

Esto nos ha permitido identificar una serie de procedimientos almacenados que consideramos imprescindibles para el correcto funcionamiento de la aplicación, mientras que otro conjunto de procedimientos bastante menor, se ha considerado como mejoras deseables que se desarrollarán si los plazos del proyecto lo permiten.

Esto no quiere decir que la lista de procedimientos actual sea exhaustiva y completa. Cuando se acometa el desarrollo de la aplicación de gestión de proyectos que utilizarán los usuarios finales probablemente se detectaran necesidades de nuevos procedimientos almacenados, probablemente consultas especializadas o que retornen listados completos de registros de algunas tablas.

En la tabla siguiente se recogen los nombres de los procedimientos almacenados, que suele ser claramente descriptivo de su función, y se acompañan de las principales reglas de negocio, decisiones y comentarios que se han tenido en cuenta para su diseño e implementación.

Una descripción ampliada del propósito de cada procedimiento almacenado y de su forma de utilización puede encontrarse en el encabezado de su código fuente, cuyo formato está estandarizado.

Respecto a la nomenclatura, los procesos que llevan el indicador (OK) han sido desarrollados y están completos. Los que indican (Mejora) no se han desarrollado al ser de menor importancia o prioridad, y podrán acometerse como mejoras posteriores. Algunos llevan también la indicación de “Proc. Privado”, que indica que es un procedimiento que es llamado internamente por otros y que no debe ser invocado directamente por los usuarios, lo que debe tenerse en cuenta al conceder permisos de ejecución.

**Tabla 5: Listado de Procedimientos Almacenados**

| Entidad | Alta | Modificación Datos | Baja | Consulta |
|---|---|---|---|---|
| **CENTRO** | **AltaCentro (OK)**<br>Un centro tiene que tener al menos un DEPARTAMENTO<br>Se crea en el mismo SP. | **ModDatosCentro (Mejora)** | **BajaCentro (OK)**<br>No permitir mientras haya departamentos activos | **ConsultaCentroActivo (OK)** |
| **DEPARTAMENTO** | **AltaDepartamento (OK)**<br>Un departamento tiene que pertenecer a un centro activo.<br>Un departamento tiene que tener como mínimo un empleado, con la función "JEFE DE DEPARTAMENTO". Se crea en el mismo SP o se transfiere desde otro departamento un empleado existente. | **ModDatosDep (Mejora)**<br><br>**ModJefeDep(OK)**<br>Proc privado. Se indica un empleado, se le asigna al departamento mediante ModDepEmp<br>y se le hace jefe mediante ModFuncEmpNoJefe. | **BajaDepartamento (OK)**<br>No permitir mientras haya empleados activos. Si es el último del centro, dar de baja el centro. | **ConsultaDepartamentoActivo (OK)** |
| **EMPLEADO** | **AltaEmpleado (OK)**<br>Proc. privado<br>No puede haber un empleado activo con el mismo código.<br>Un empleado tiene que ser asignado a un departamento y una función activas. Si el empleado que se crea es JEFE DE PROYECTO, se crea un proyecto interno para sus tareas no facturables a clientes.<br>Si su función es JEFE DE DEPARTAMENTO, debe verificarse que en su | **ModDatosEmp (OK)**<br>Modificar datos propios del empleado.<br><br>**ModFuncEmp (OK)**<br>Al modificarse el jefe de departamento debe haber sustituto: Cambiar automáticamente la función del nuevo a JEFE DE DEPARTAMENTO.<br><br>**ModFuncEmpNoJefe (OK)**<br>Proc. privado. Impedir si es jefe de proyecto y tiene proyectos abiertos. | **BajaEmpleado (OK)**<br>Si el empleado es JEFE DE DEPARTAMENTO, es necesario designar un sustituto. Si no hay y es el último empleado del departamento, éste se da de baja.<br>Si no es jefe llama a BajaEmpNoJefeDep.<br><br>**BajaEmpNoJefeDep (OK)**<br>Proc. privado. Definido para evitar dependencias circulares con ModJefeDep. | **ConsultaEmpleadoActivo (OK)** |

| Entidad | Alta | Modificación Datos | Baja | Consulta |
|---|---|---|---|---|
| | departamento no haya otra persona con esa función.<br><br>**AltaEmpleadoNuevo (OK)**<br>Llama a AltaEmpleado, pero generando primero el código de empleado. | **ModDepEmp (OK)**<br>Cambio de departamento y centro de un empleado. No se puede transferir al jefe de un departamento mientras tenga esa función. Impedir si es jefe de proyecto y tiene proyectos abiertos (esto se controla en BajaEmpNoJefeDep, que se llama). | Comprueba que la fecha de baja no es anterior a su última imputación de gastos y horas.<br>Si es JEFE DE PROYECTO, se cierra su proyecto de tareas no facturables. Si lidera algún otro proyecto no se permite su baja.<br>Se cierran todas sus asignaciones a proyecto. | |
| **TIPO_FUNCION** | **AltaFuncion (OK)**<br>Controlar que no exista ya. Se crean desactivadas. Es necesario crear luego todos sus costes. | **ActivarFuncion (OK)**<br>Al activar función, comprobar tiene coste para todos los tipos de horas. Si no, no se permite. | **BajaFuncion (OK)**<br>No se permite si hay empleados (activos o inactivos) que la tengan. | **ConsultaFunciónActiva (OK)** |
| **TIPO_GASTO** | **AltaTipoGasto (OK)**<br>Controlar que no exista ya. | **ModTipoGasto (Mejora)** | **BajaTipoGasto (OK)**<br>No permitir si hay gastos de ese tipo, o permisos sobre ese tipo de gasto. | |
| **PERMISO** | **AltaPermiso (OK)**<br>Controlar que no exista ya un permiso igual y activo. | N/A | **BajaPermiso (OK)**<br>Controlar que la fecha de baja es posterior a la del último gasto imputado de este tipo por el empleado. | |
| **GASTO** | **AltaGasto (OK)**<br>Proc. privado. Se debe llamar sólo desde AltaImpGasto, una vez realizadas las validaciones. | **ModGasto (Mejora)** | **BajaGasto (Mejora)** | |
| **FECHA** | **AltaFecha (OK)** | **ModFecha (Mejora)** | **BajaFecha (Mejora)** | |
| **PROYECTO** | **AltaProyecto (OK)**<br>En el alta de un proyecto se | N/A. | **BajaProyecto (OK)**<br>Al dar de baja un proyecto, | **ConsultaProyecto (OK)** |

| Entidad | Alta | Modificación Datos | Baja | Consulta |
|---|---|---|---|---|
| | debe comprobar que se asigne a un cliente activo y existente en la tabla CLIENTE.<br>También es necesario asignar un empleado determinado con función JEFE DE PROYECTO al proyecto como líder. | | es necesario dar de baja todas las asignaciones al mismo y su relación de liderazgo con el jefe de proyecto.<br>(Mejora) Impedir que se desactive el proyecto interno de los jefes de proyecto. | |
| **IMP-GASTO** | **AltaImpGasto (OK)**<br>Verificar que el trabajador está asignado a ese proyecto o lo lideran.<br>Verificar que los empleados que imputan gastos a un proyecto tiene permisos para ese tipo de gasto<br>Verificar que no se imputan gastos a un proyecto cerrado. | N/A | N/A | |
| **IMP-HORAS** | **AltaImpHoras (OK)**<br>Verificar que los empleado que imputan horas a un proyecto está asignado a él o lo lidera.<br>Verificar que no se imputan horas a un proyecto cerrado.<br>Nota: Haría falta un trigger para control del total de horas imputadas por fecha: no más de 24 en total, máximos de los distintos tipos, etc. No se conocen las reglas a aplicar. | N/A | N/A | |
| **LIDERAZGO** | N/A | **ModLid (OK)** | N/A | |

| Entidad | Alta | Modificación Datos | Baja | Consulta |
|---|---|---|---|---|
| | | Se sustituye un jefe de proyecto por otro.<br>Ambos deben ser del mismo departamento para garantizar que se sigue cumpliendo la regla de asignación de empleados. | | |
| **ASIGNACIÓN** | **AltaAsignacion (OK)**<br>Asegurar que sólo se asignan a un proyecto empleados del departamento de su jefe.<br>Asegurar que no se asignan empleados a proyectos cerrados. | N/A | **BajaAsignación (OK)**<br>La baja de la asignación no puede ser anterior a la ultima imputación de gastos u horas. | |
| **CLIENTE** | **AltaCliente (OK)**<br>Al dar de alta un cliente es necesario dar de alta también su primer proyecto. | **ModCliente (Mejora)** | **BajaCliente (Mejora)** | **ConsultaClienteActivo (Mejora)** |
| **COSTE** | **AltaCoste (OK)** | **ModCoste (Mejora)** | **BajaCoste (Mejora)** | **ConsultaCosteActivo (Mejora)** |
| **TIPO_HORAS** | N/A | N/A | N/A | N/A |
| **TIPO_DOCUMENTO** | N/A | N/A | N/A | N/A |
| **TIPO_JORNADA** | N/A | N/A | N/A | N/A |
| **TIPO_ESTADO** | N/A | N/A | N/A | N/A |

### 3.2.6 Instalación y Configuración del SGBD Oracle

En paralelo a la revisión de requerimientos y el diseño, se llevó a cabo la instalación del SGBD Oracle en su versión 8.1.7, que comportó algunas dificultades. El proceso de instalación no se ejecutaba correctamente en uno de los ordenadores utilizados, ya que los menús no funcionaban. Una investigación del problema consultando las bases de datos de conocimiento de Oracle y los foros especializados permitió identificar que la causa es un problema conocido en la instalación de esta versión en ordenadores con procesador Pentium. Se realizaron los pasos recomendados para resolver el problema y se concluyó la instalación y configuración de los usuarios y de la base de datos sin otros problemas.

### 3.2.7 Construcción y Pruebas de la Base de Datos

La construcción de la base de datos a partir del diseño lógico y la programación de los procedimientos almacenados se realizó con la aplicación TOAD (http://www.quest.com/toad/) en su versión 7.2. Esta aplicación ha permitido la creación y definición de las sentencias DDL que crean las tablas de la base de datos, y la programación y pruebas de los procedimientos almacenados.

Cada uno de los elementos construidos se ha probado; las tablas mediante inserciones de datos para comprobar el cumplimiento de restricciones definidas y los procedimientos almacenados mediante ejecuciones sistemáticas que han comprobado su comportamiento con datos correctos y con datos incorrectos. En caso de error se ha verificado que se generaran las excepciones adecuadas.

### 3.2.8 Seguridad, Roles y Privilegios

Se han creado dos usuarios distintos, TFC_OWN y TFC_USER con distintos privilegios y roles para gestionar adecuadamente la base de datos del proyecto TFC.

TFC_OWN es un usuario con el privilegio RESOURCE, lo que le permite crear tablas y procedimientos almacenados. Este usuario será el propietario de todos los objetos de base de datos que se generen en este proyecto.

TFC_USER en cambio es un usuario que tiene el privilegio CONNECT y al que sólo se concederán permisos para ejecutar procedimientos almacenados. Este usuario o uno con los mismos permisos que este es el que utilizarán todas aquellas futuras aplicaciones que accedan a nuestra base de datos. Esto implicará que el único modo que tendrán de realizar operaciones será a través de los procedimientos almacenados que les proporcionamos, lo que unido a un desarrollo riguroso de los mismos garantizará que se cumplan todas las reglas de negocio definidas. Los procedimientos almacenados actúan durante su ejecución con los privilegios de su propietario, de modo que a través de ellos podemos proporcionar una interfaz bien definida para que los usuarios y aplicaciones puedan realizar cualquier acción necesaria.

Sin embargo, durante el periodo de desarrollo y para facilitar el manejo y las pruebas se ha dotado a TFC_USER de algunos privilegios adicionales que lo convierten en un administrador de datos de la aplicación. Esto significa que tiene permisos para seleccionar, insertar, borrar y modificar registros en todas las tablas de TFC_OWN y se ha definido unos sinónimos privados para poder referenciarlas sin necesidad de especificar el propietario. Al hacer esto, lo que buscamos es un usuario que nos permita trabajar y manipular los datos durante las pruebas pero que no tenga capacidad para alterar por descuido o error los objetos de la base de datos. Estos privilegios se le deberán retirar al pasar a producción y como mucho se conservarán en algún usuario específico destinado al mantenimiento en producción de la base de datos.

### 3.2.9 Base de Datos Desarrollada

Los ficheros que contienen las sentencias SQL necesarias para crear la base de datos Oracle 8i diseñada y el código PL/SQL de los procedimientos almacenados desarrollados se entregan adjuntos a este documento, junto con sus instrucciones de instalación.

## 3.3 Desarrollo de la Aplicación de Generación de Datos

### 3.3.1 Enfoque del Desarrollo

Los requerimientos establecidos en el enunciado para la aplicación de generación de datos aleatorios no especifican ninguna tecnología para su implementación. Por tanto, he seleccionado la que he considerado más adecuada para ello de acuerdo con mi experiencia y conocimientos, que es un desarrollo orientado a objetos utilizando Java.

La programación orientada a objetos con Java se trabajó en la asignatura de bases de datos II. La API JDBC proporciona una forma sencilla de acceder a la base de datos utilizando el driver JDBC nativo que proporciona Oracle, lo que nos permite definir luego una lógica y un tratamiento de datos complejo como los que requiere esta aplicación.

Además, realizar este trabajo mediante programación orientada a objetos tiene la ventaja de que un diseño bien realizado permitirá que las clases que describen entidades de negocio y gestores de acceso a datos construidos para esta aplicación puedan ser reutilizadas posteriormente con pocas o ninguna modificación para la construcción de la aplicación de gestión de proyectos que utilizarán los usuarios finales del sistema.

En el curso de este desarrollo se realizaron las siguientes actividades:

- Se revisaron los requerimientos para asegurar su comprensión e identificar las principales entidades de negocio y las reglas que la aplicación tiene que plasmar. Se definieron las nuevas reglas y restricciones necesarias de acuerdo con mi experiencia y conocimiento del ámbito del problema.
- A partir de los requerimientos se procedió a elaborar un modelo de componentes de la aplicación. Se definió su arquitectura, se identificaron los patrones aplicables y se definieron los principales tipos de clases que la constituirían.
- Se realizaron diagramas de clases para identificar con precisión las clases necesarias, sus atributos y métodos.
- Se especificó la lógica de generación de datos aleatorios que implementaría las reglas identificadas.
- Finalmente, se construyeron y probaron, tanto de forma unitaria como integrada el código fuente correspondiente al diseño realizado.

### 3.3.2 Revisión de los Requerimientos

Los requerimientos proporcionados en el enunciado fueron analizados para la realización de las tareas de diseño y asegurar el cumplimiento de todos sus puntos. En aquellos puntos o reglas que no estaban explícitamente definidos y según se indica, utilicé mi experiencia usando este tipo de aplicaciones para adoptar supuestos razonables acerca de la generación de los datos. De acuerdo con el enunciado, las reglas son las siguientes:

**Tabla 6: Reglas de Negocio para la Aplicación de Generación de Datos**

| Reglas de Negocio para la Aplicación de Generación de Datos |
|---|
| El estudiante también deberá implementar un mecanismo de inicialización de la base de datos tal que simule un año de funcionamiento, teniendo en cuenta que el número de centros de la organización es de entre 5 y 10, que cada centro tiene un número de trabajadores de entre 25 y 150, un número de departamentos de entre 2 y 4, y cada departamento tiene un número de miembros entre 5 y 40. Este mecanismo funcionará haciendo llamadas a los procedimientos implementados, será aleatorio y se ajustará al que la realidad y sentido común marca, (es decir no será una inicialización cíclica de fácil detección). |

Las reglas de generación de datos aleatorios que se han elaborado y que cumplen con todos los puntos recogidos en estos requerimientos se recogen con detalle en el apartado 3.3.5 Especificación de la Lógica de Generación de Datos.

### 3.3.3 Componentes de la Aplicación de Generación de Datos

El siguiente diagrama muestra los componentes principales de la Aplicación de Generación de Datos y sus relaciones de dependencia y uso:

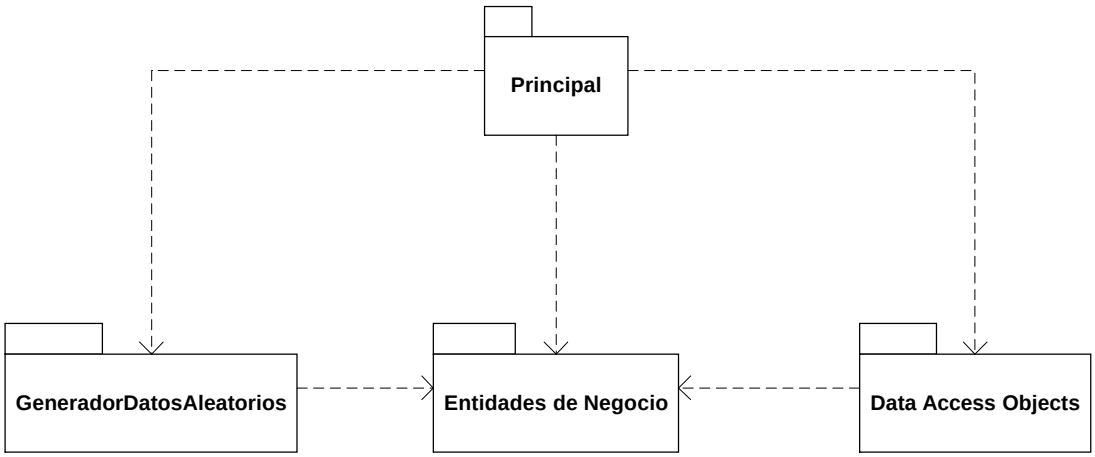



**Ilustración E: Diagrama de Componentes de la Aplicación de Generación de Datos**

El componente Principal es el que controla la ejecución de la aplicación y el que contiene la lógica que determina qué entidades de negocio hay que crear para simular los datos de actividad del año y el número en que se crearán.

El componente de Entidades de Negocio engloba las clases que representan dichas entidades en la aplicación para su gestión y manejo. Las entidades se corresponden en general con las identificadas durante el diseño de la base de datos, que forman el punto de partida para su identificación.

Para generar aleatoriamente los datos de cada una de las instancias de entidades de negocio individuales que la lógica del componente Principal determina que es necesario crear, se utiliza un componente de generación de datos aleatorios. Este componente se encarga de generar datos de aspecto realista para el nombre, dirección, documento de identidad, etc.

Una vez que se han generado todos los datos de la estructura y la actividad, se utilizan las clases del componente Data Access Objects (DAO) para realizar las operaciones necesarias para reflejar las entidades de negocio en la base de datos. Estas clases utilizan las facilidades que proporciona la API JDBC para ejecutar los procedimientos almacenados correspondientes a las operaciones, pasando los parámetros y recuperando los resultados.

En los siguientes apartados detallaremos las clases que forman parte de cada uno de estos componentes identificados.

### 3.3.4 Diagramas de Clases

**Diagrama de entidades de negocio**

El punto de partida para el diseño de la aplicación es la realización con ayuda de Visio 2000 de un diagrama de clases que traslada a un modelo de objetos las entidades identificadas en el diseño conceptual y lógico elaborado para la entrega anterior. Esto nos permite trabajar con las entidades en la aplicación manteniendo relaciones consistentes con las entidades almacenadas en la base de datos implementada:

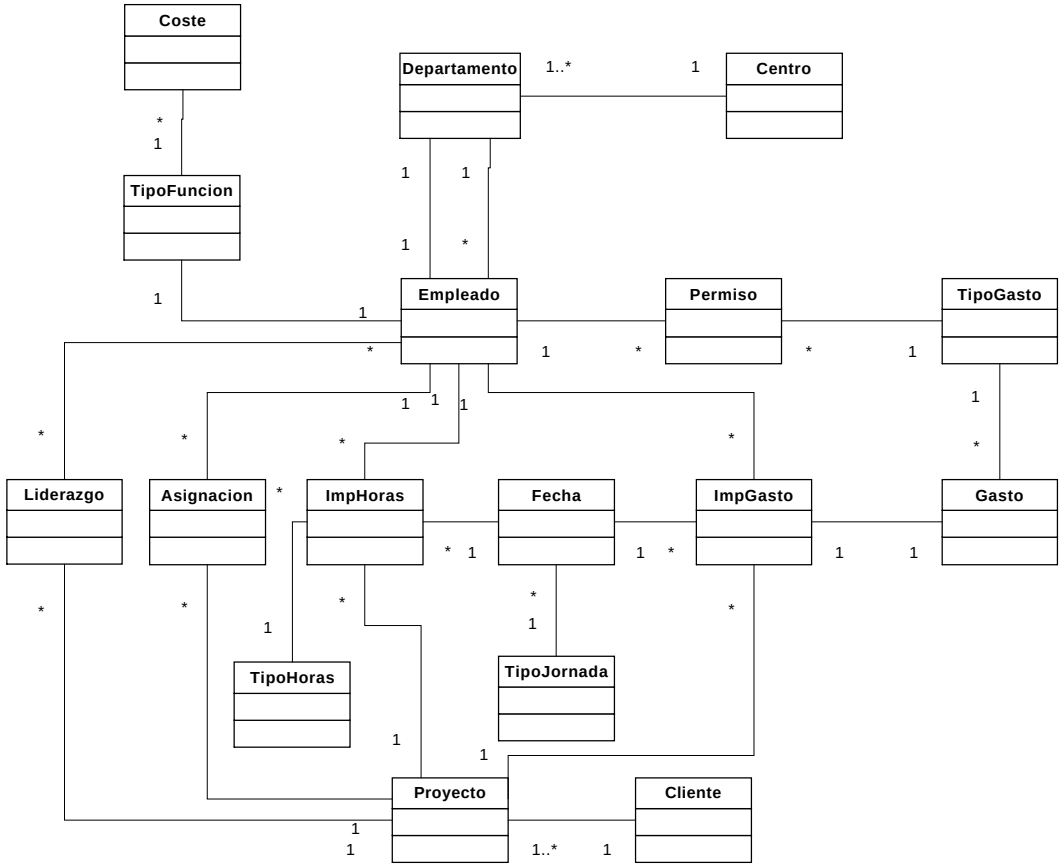



**Ilustración F: Diagrama de Clases de las Entidades de Negocio**

**Data Access Objects**

Para desacoplar en lo posible los objetos que representan las entidades de negocio de la aplicación de la implementación específica de la capa de persistencia (en este caso la base de datos que hemos implementado en el SGBD Oracle) hemos aplicado el patrón DAO (Data Access Objects) recomendado por Sun. De este modo limitamos y acotamos los puntos donde es necesario realizar cambios en caso de una posible modificación o sustitución del gestor de bases de datos relacionales que soporta la implementación. Los gestores de datos ofrecen unos servicios bien definidos a las entidades de negocio que tratan de hacer trasparentes todos los detalles de implementación de la persistencia.

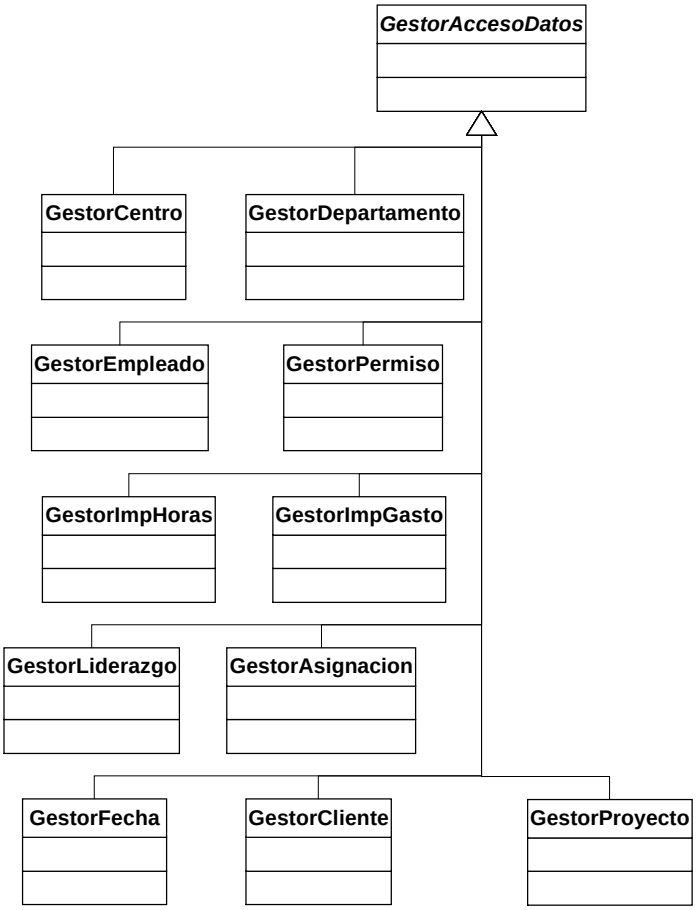



**Ilustración G: Diagrama de Clases de los Data Access Objects**

Sin embargo, el modo de funcionamiento de la aplicación de generación de datos es más parecido, por su naturaleza, al de una aplicación estructurada, ya que genera grandes lotes de datos que tienen que ser tratados secuencialmente. Por este motivo no hemos aplicado el patrón DAO como aparece en Core J2EE Patterns, permitiendo que los gestores obtengan conexiones a medida que son necesarias para cada operación.

Hemos optado por hacer que cada instancia de un gestor tome como parámetro una conexión a la base de datos que es única y obtenida por la aplicación que lo instancia (en este caso Principal) y que se mantiene a lo largo de la ejecución. Puesto que no hay múltiples clientes ni hebras de ejecución, no hay problemas de concurrencia al hacer esto sí garantizamos que los recursos se liberan correctamente una vez utilizados.

Este modo de funcionamiento hace que la aplicación de generación de datos, que trabaja con un volumen de datos alto en un periodo de tiempo breve, sea más eficiente, ya que se elimina el coste en tiempo de realizar la conexión y desconexión en cada una de las operaciones. No obstante, una consecuencia es que estos gestores no deben ser reutilizados directamente sin

analizar este punto para una aplicación on-line, que sí debe tener en cuenta los factores que hemos comentado como concurrencia, etc.

**Otras Clases Identificadas**
Las otras clases importantes que se han identificado para la construcción de esta aplicación son las siguientes:

**GeneradorDatosAleatorios**
Esta clase tiene como finalidad generar información aleatoria y de aspecto razonablemente realista y consistente para los atributos de las principales entidades de negocio: Centro, Departamento, Empleado, Cliente y Proyecto. Proporciona una variedad de datos adecuada para generar las cantidades de datos para simular un año de funcionamiento. Como futura evolución, podría evaluarse la posibilidad de hacerla parametrizable por medio de ficheros de configuración

**Principal**
La clase principal contiene la mayor parte de la lógica de la aplicación de generación de datos y es la clase ejecutable principal del proyecto. Es una especialización de la clase abstracta Aplicación, que proporciona los métodos comunes para una aplicación que se conecta a una base de datos. La clase principal utiliza para conectarse a Oracle los métodos de la clase BDAccessor, que encapsula la funcionalidad de creación y gestión de conexiones con la base de datos para separarlos de la aplicación. La lógica y las reglas de generación de datos recogidas su método principal se recogen con más detalle en el siguiente apartado.

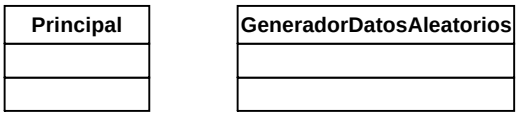



**Ilustración H: Diagrama de Clases para el resto de Clases**

**Clases auxiliares**
Se han utilizado o construido otras clases además de las dos anteriores. En particular cabe destacar la clase BDAccessor, que como ya se ha indicado proporciona los métodos para crear y gestionar una conexión con la base de datos, las clases TestGestorCliente y TestGestorProyecto que se utilizaron para realizar las pruebas de concepto inicial de los objetos de acceso a datos, y las clases Utilidades, FileProcessor y FileProcessorException, que fueron proporcionadas en la asignatura de BD II para simplificar algunas tareas de presentación de información y gestión de ficheros. Estas últimas tres finalmente no se utilizan en esta versión pero pueden resultar interesantes para futuras evoluciones de la aplicación.

## 3.3.5 Especificación de la Lógica de Generación de Datos

La lógica de generación de datos aleatorios se ciñe a los requerimientos especificados en el enunciado del TFC. Para aquellos comportamientos que no estaban explícitamente definidos hemos optado por realizar una serie de asunciones razonables. La descripción en pseudocódigo de la lógica aplicada por la aplicación para generar los datos con los que se simula un año de funcionamiento es la siguiente:

***** APLICACIÓN DE GENERACION DE DATOS *****

INICIALIZACION DEL CALENDARIO LABORAL
    Para cada día del año, construimos la fecha, indicando jornada laborable o festiva

CREACION DE LA ESTRUCTURA ORGANIZATIVA
    Creación de los centros de la organización [5-10 en total]
        Creación de los departamentos de cada centro [2-4 por centro]
            Creación de empleados de los departamentos [5-40 por departamento]
            Verificación de las restricciones de empleados por centro [25-150]
            Verificación de cupos funcionales: Un jefe para cada departamento.
            Resto: jefes de proyecto [25%], economistas [25%], ingenieros [50%]

GENERACIÓN DE PROYECTOS Y ASIGNACIONES
Creación de la lista de clientes [se ha elegido un volumen de 20-40 clientes]
Creación de proyectos de cada cliente [1-4 proyectos por Cliente al año]
El primer proyecto lleva la fecha de alta del cliente
Generación fecha inicio aleatoria para cada proyecto
Selección duración cada proyecto [60 - 240 días]
Calculo fecha de fin del proyecto. Si es del año siguiente queda abierto.
Elegimos jefe del proyecto aleatoriamente entre los existentes.
Asignamos personas del mismo departamento que el jefe al proyecto

GENERACIÓN DE ACTIVIDAD
Por cada departamento se recorren todos sus empleados
Para cada empleado se recorre todo el calendario
Para cada fecha se identifican los proyectos activos para esa persona
Pueden imputarse horas y gastos como líder o como asignado.
Tiene preferencia imputar horas en aquellos que se lidera
Sólo se contemplan días laborables
Los días se asignan completos a un proyecto para simplificar.
Si hay varios posibles se elige uno al azar para imputar en esa fecha.
Si hay un proyecto válido, se imputan horas siempre.
Las imputaciones de gasto son aleatorias.
Si es posible imputar horas, hay un 20% por empleado por día de imputar un gasto]
Las imputaciones siguen esta distribución [50% comida (9 €), 40% transporte (12 €), 10% otros (15 €)]

ROTACIÓN, MODIFICACIÓN DE FUNCIONES Y TRASLADOS
No se han incluido estas funciones dentro de esta versión.

INSERCION DE LOS DATOS EN LA BD
Se insertan todos los datos creados anteriormente siguiendo el orden lógico de creación para garantizar las dependencias.

***** FIN DE LA ESPECIFICACION ***"**

Esta lógica está implementada en el método “main” de la clase Principal. Se han definido como constantes todos los parámetros que se utilizan en la generación de datos aleatorios, lo que permite realizar cambios de forma fácil y centralizada en las reglas de generación de datos aleatorios. Como una futura mejora, podría plantearse la posibilidad de recoger esta parametrización en ficheros de configuración.

### 3.3.6 Construcción y Pruebas de la Aplicación de Generación de Datos

La construcción de la aplicación de generación de datos a partir del diseño se realizó con el entorno integrado de desarrollo (IDE) NetBeans en su versión 3.6. Este entorno he permitido realizar los desarrollos de forma eficiente y ha proporcionado las herramientas necesarias para realizar las pruebas unitarias e integradas de forma rápida y completa.

Cada uno de los elementos construidos se ha probado mediante ejecuciones sistemáticas que han comprobado su comportamiento con datos correctos y con datos incorrectos. En caso de error se ha verificado que se generaran las excepciones adecuadas y que estas se manejan correctamente.

En el curso de las pruebas se detectaron dos errores que obligaron a corregir procedimientos almacenados desarrollados en la entrega anterior: un error en el orden de los parámetros de una invocación de un procedimiento dentro de otro y un truncado inesperado de una variable que generaba una excepción.

Estos problemas, que no habían sido detectados por la batería de pruebas de la fase anterior, se hicieron evidentes al realizar pruebas integradas con la aplicación de generación de datos, dado que ésta generaba un número de casos posibles superior al empleado para las pruebas unitarias de los procedimientos. Su corrección fue rápida y mínima.

La solidez del diseño de la base de datos y de los procedimientos almacenados queda evidenciada por el hecho de que no fue necesario en ningún momento modificar la funcionalidad de los procedimientos almacenados para construir una aplicación que funcionara de forma correcta ni fue necesario añadir procedimientos que no estuvieran contemplados.

### 3.3.7 Aplicación Desarrollada

Los ficheros que contienen el código fuente de la aplicación desarrollada y su documentación técnica asociada se entregan adjuntos a este documento, junto con sus ficheros de configuración e instrucciones de instalación.

## 3.4 Conclusiones

Una vez finalizado todo el proceso de desarrollo de acuerdo con la planificación realizada, podemos verificar que los objetivos que identificamos al inicio y que se detallan en el apartado 3.1.2 se han cumplido. La base de datos, los procedimientos almacenados y la aplicación de generación de datos han sido diseñados y construidos de acuerdo con los requerimientos especificados en el enunciado proporcionado para este TFC, y se entregan acompañando este documento.

También se han cumplido los objetivos de más largo alcance que hemos detallado al comienzo de este documento y que justifican la realización de este trabajo. El proceso de desarrollo que detalla esta memoria, y la calidad de los productos obtenidos, reconocida a lo largo de la evaluación continua, evidencian que he aplicado de forma rigurosa y metódica los conocimientos, metodologías, técnicas y habilidades que he adquirido a lo largo de mis estudios de informática, en particular en el área de las bases de datos relacionales en la que este trabajo se centra.

Este trabajo ha servido también para demostrar mi capacidad de afrontar un proyecto de cierta complejidad, haciendo un uso integral de mis conocimientos y habilidades para llevar a cabo con éxito y de forma independiente actividades relacionadas con la planificación, el análisis, el diseño, la construcción y las pruebas a lo largo de todo el ciclo de desarrollo de un software.

Por último, sólo me queda señalar que la realización de este trabajo ha constituido una experiencia enriquecedora que me permitirá desempeñar mejor mi trabajo en los roles de desarrollo o gestión que asuma en mi futura actividad profesional dentro de la informática.

# 4 Glosario

**Atributo de una entidad**: Propiedad que interesa de una entidad.
**Base de Datos**: Es un conjunto estructurado de datos que representa, entre otros, entidades y su interrelaciones, con integración y compartimentación de datos.
**BD**: Ver Base de Datos.
**Diseño Conceptual**: Etapa del diseño de una base de datos que obtiene una estructura de la información de la futura base de datos independiente de las tecnología que se quiere utilizar.
**Diseño Lógico**: Etapa del diseño de una base de datos que parte del resultado del diseño conceptual y lo transforma de modo que se adapte al modelo del SGBD con el que se desea implementar la base de datos.
**Disparador**: Acción o procedimiento almacenado que se ejecuta automáticamente cuando se lleva a cabo una operación INSERT, DELETE o UPDATE sobre alguna tabla de la BD.
**Driver JDBC**: Implementación particular del JDBC para un SGBD concreto.
**Entidad Débil**: Entidad cuyos atributos no la identifican completamente, sino que sólo la identifican de forma parcial.
**Entidad**: Objeto del mundo real que podemos distinguir del resto de los objetos y del cual nos interesan algunas propiedades.
**Interrelación**: Asociación entre entidades.
**Java**: Java es un lenguaje de programación expresamente diseñado para utilizarse en el entorno distribuido de Internet. Fue diseñado para que fuera muy similar a C++, pero es más fácil de utilizar que este y sigue un modelo de programación orientada a objetos.
**JDBC**: Java Database Connectivity. Es un API estándar de Java para interactuar con las abstracciones y conceptos de las BD relacionales y, en concreto, para trabajar con SQL.
**Permiso**: Privilegio que se otorga a un usuario (o conjunto de usuarios) para realizar una operación determinada sobre un cierto objeto de la BD.
**PL/SQL**: En el SGBD Oracle, PL/SQL es un lenguaje procedimental que extiende SQL. Su propósito es combinar el lenguaje de la base de datos con un lenguaje de programación procedimental para la definición de acciones o funciones.
**Procedimiento Almacenado**: Acción o función definida por un usuario que proporciona un determinado servicio. Una vez ha sido creado, se guarda en la BD y pasa a ser tratado como un objeto más de ésta. La ejecución de un procedimiento puede devolver ninguno, uno o más valores.
**Rol**: Agrupación de privilegios sobre alguno de los componentes de una BD.
**SGBD**: Ver Sistema de Gestión de Bases de Datos.
**Sistema de Gestión de Bases de Datos**: Software que gestiona y controla bases de datos. Sus principales funciones son las de facilitar la utilización simultánea a muchos usuarios de tipos diferentes, independizar al usuario del mundo físico y mantener la integridad de los datos.
**SQL**: Lenguaje pensado para describir, crear, actualizar y consultar bases de datos. Fue concebido por IBM a finales de los años setenta y estandarizado por ANSI e ISO en el año 1986 (el último estándar del SQL es de 1999). Actualmente lo utilizan casi todos los SGBD del mercado (incluso algunos SGBD no relacionales y algunos sistemas de ficheros).
**TFC**: Trabajo Final de Carrera.

# 5 Bibliografía


- Material de Bases de Datos I
  Jaume Sistac Planas (coordinador)
  UOC, 1ª Edición (Febrero 2002)

- Material de Bases de Datos II
  Jaume Sistac Planas (coordinador)
  UOC, 2ª Edición (Febrero 2004)

- Material de Ingeniería de Software I
- Benet Campderrich Falgueras, Recerca Informatica S.L.
  UOC, 1ª Edición (Febrero 2002)

- Material de Técnicas de Desarrollo de Software
  Fatos Xhafa
  UOC, 1ª Edición (Febrero 2003)

- Material de Gestión de Organizaciones y Proyectos Informáticos
  Miquel Barceló García, Joan Antoni Pastor i Collado
  UOC, 1ª Edición (Febrero 2004)

- Oracle8i On-Line Generic Documentation Release 3 (8.1.7)
  http://www.oracle.com/technology/documentation/oracle8i.html

- Core J2EE Patterns: Best Practices and Design Strategies
  Deepak Alur, John Crupi and Dan Malks.
  Editorial Prentice Hall / Sun Microsystems Press. 1ª edición (Junio 2001)
  http://java.sun.com/blueprints/corej2eepatterns/index.html

- Thinking in Java 3rd Edition
  Bruce Eckel.
  Edición electrónica gratuita (Noviembre 2002)
  http://www.mindview.net/Books/TIJ/

# 6 Anexos

## 6.1 Instrucciones de Instalación de la Base de Datos

En una instancia de Oracle 8i ya existente o una nueva creada mediante DBA Studio, se deben ejecutar en orden los siguientes scripts que se encuentran el directorio Producto/sql:

Con usuario SYS o SYSTEM:

- CREATE_USERS.sql
  Crea los usuarios TFC_OWN y TFC_USER, que son respectivamente el propietario de los objetos de la base de datos del proyecto y un usuario como aquellos con el que las futuras aplicaciones se conectarán. Se crearán también sus correspondientes esquemas. Sus contraseñas son respectivamente OWN y USER.

Con usuario TFC_OWN:

- TFC_OWN.sql
  Crea todos los objetos de base de datos, secuencias, tablas y procedimientos almacenados. Concede permisos de ejecución de procedimientos almacenados a TFC_USER.
- DATA_TFC_OWN.sql
  Crea los datos estáticos necesarios para que la aplicación funcione correctamente.

Con usuario TFC_USER:

- TFC_USER.sql
  Crea unos sinónimos privados de los procedimientos almacenados de TFC_OWN para poder ejecutarlos sin hacer referencia a su propietario.

También se adjunta para facilitar la restauración de la situación inicial tras la realización de pruebas o tras una ejecución de la aplicación de generación de datos un script de borrado (DELETE_LOADED_DATA.sql) que respeta los datos estáticos necesarios.

## 6.2 Instrucciones de Instalación de la Aplicación de Generación de Datos

El código fuente de la aplicación está en la carpeta Producto/TFC a partir del directorio donde se ubique esta entrega, y está englobado en un paquete con ese nombre.

En el directorio /TFC hay 3 scripts ejecutables:

- compile.bat: compila todos los ficheros *.java (IMPORTANTE: la ruta a las librerias JDBC classes12.zip y nls_charset12.zip de Oracle 8.1.7 debe estar en el CLASSPATH)
- docs.bat: genera los javadocs correspondientes a los ficheros fuente
- run.bat: ejecuta la Aplicación de Generación de Datos

La clase ejecutable es Principal, aunque hay otras que también pueden ejecutarse para llevar a cabo pruebas unitarias de sus métodos.

En la carpeta Producto/doc que puede encontrarse al mismo nivel que Producto/TFC están los javadocs generados a partir de la versión más reciente del código fuente Java.

La configuración del acceso a la BBDD debe hacerse a en el fichero bd.properties, que debe encontrarse siempre en Producto/TFC. Los datos que contiene son los de host, port y dbname.

Nota sobre la ejecución: Principal muestra en la consola datos sobre la ejecución; sin embargo los códigos identificadores que se utilizan para empleados, proyectos, etc. son provisionales y no corresponderán con los definitivos debido a que estos son asignados al realizar la inserción en la base de datos.

## *6.3 Recursos e Infraestructura Empleados*

En este apartado se detallan los recursos utilizados para llevar a cabo el proyecto.

### 6.3.1 Hardware

Se han utilizado en este proyecto dos ordenadores PC compatibles:

- Equipo de sobremesa HP t580: Intel Pentium IV a 3.2 GHz, 1 Gb de memoria RAM con SO Windows XP Home Edition.
- Ordenador portátil IBM Thinkpad T40: Intel Pentium M a 1,5 GHz, 1 Gb de memoria RAM con SO Windows XP Professional Edition.

### 6.3.2 Software

Se han utilizado en este proyecto los siguientes sistemas y aplicaciones para el desarrollo de la Base de Datos:

- Sistema Gestor de Base de datos: De acuerdo con lo solicitado en el enunciado, la base de datos se ha construido con el SGBD Oracle en su versión 8.1.7.
- Diagrama E/R: Microsoft Visio 2000.
- Desarrollo SQL y PL/SQL: La aplicación de desarrollo para Oracle TOAD de Queso Software en su versión 7.2.

Para la Aplicación de Generación de Datos, se han utilizado las siguientes aplicaciones:

- Diagramas UML: Microsoft Visio 2000.
- Entorno de desarrollo integrado: NetBeans IDE versión 3.6.

Para la elaboración de todos los documentos de este proyecto se han utilizado las aplicaciones de la familia Microsoft Office (Word, Excel, Powerpoint) en sus versiones XP o 2002.

La planificación y seguimiento del proyecto se ha llevado a cabo utilizando Microsoft Project 2000.