UNIVERSIDAD POLITECNICA DE VALENCIA

UNIVERSIDAD POLITÉCNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INFORMÁTICA APLICADA

PERMISOS Y LICENCIAS DE PERSONAL

PROYECTO FIN DE CARRERA
Código P.F.C.: DISCA-93

Javier Crespo Garzón

Gaspar Quiles Gomis

Curso 2010-2011

Personal
Solicitud de permisos y licencias

ÍNDICE

1. SITUACIÓN ACTUAL DE LA EMPRESA Y NECESIDADES

1.1. Descripción de situación actual

Actualmente, en el ámbito de la Administración Pública valenciana, y más concretamente en la Consellería de Infraestructuras y Transporte, existe la necesidad de informatizar parte de la gestión del departamento de Personal, en concreto lo concerniente a la petición de permisos y licencias de sus empleados.

Existe una gran diversidad de permisos y licencias, tantas como se recogen en la Ley 30 del Personal de la Administración Pública y en la Ley Pública Valencia:

-DIAS ADICIONALES ASUNTOS PARTICULARES
-DÍAS ADICIONALES VACACIONES
-ADOPCIÓN INTERNACIONAL
-ACUMULACION JORNADA POR LACTANCIA
-PATERNIDAD
-ADOPCIÓN O ACOGIMIENTO MENORES
-FALLECIMIENTO
-LICENCIA. POR ESTUDIOS
-ENFERMEDAD GRAVE FAMILIAR
-ENFERMEDAD COMÚN
-REDUCC. ENFERMEDAD MUY GRAVE
-REDUCC. DIARIA GUARDA LEGAL
-REDUCC. 1H. DIARIA POR MINUSVALÍA
-VACACIONES
-LICENCIA. ASUNTOS PROPIOS
-LICENCIA. SIN RETRIB. POR ENFERMEDAD FAMILIAR.
-MATRIMONIO O UNIÓN DE HECHO
-ETC...

Actualmente, la solicitud por parte de los empleados de la Consellería de Infraestructuras se realiza en papel preimpreso por triplicado.

Una vez rellenada la solicitud por el empleado, es comprobada y firmada por su jefe de departamento y depositado en la Subsecretaria, Departamento de Personal. Este departamento, deberá de comprobar manualmente que lo que solicita el empleado es correcto. Por ejemplo, un empleado se pide unos días de vacaciones. Personal deberá de comprobar, cuántos días había solicitado ese empleado de vacaciones en el año en curso, y que no se pase de 22 días laborables, que los días que se pide, formen al menos 7 días consecutivos, y un sin fín más de comprobaciones marcadas por la Ley 30 de la Función Pública y Ley de la Función Pública Valenciana.

Existen también una serie de permisos y licencias que aminoran en parte o en su totalidad las retribuciones del empleado público, como por ejemplo **LICENCIA. SIN RETRIB. POR ENFERMEDAD FAMILIAR, LICENCIA. SIN RETRIB. POR ENFERMEDAD FAMILIAR.** Estos permisos se tienen que comprobar manualmente e introducirlos en la aplicación de nóminas.

Los casos descritos anteriormente, son sólo una representación de la complejidad que supone llevar todo este proceso de forma manual. La cantidad de comprobaciones y anotaciones que debe realizar este departamente supone una carga y un coste considerable.

Toda esta complejidad, llevó a la necesidad de plantearse una aplicación que resolviese estos problemas, surguiendo así **PERMISOS Y LICENCIAS DE PERSONAL.**

1.2. Alcance de la aplicación

Se pretende que cada usuario (funcionarios, laborales, eventuales e interinos) pueda, desde la INTRANET, ver e introducir sus permisos y licencias, es decir, permitirá dar de alta, anular, borrar y consultar solicitudes de permisos y licencias, entre otras cosas, según determinados criterios. De este modo, se eliminaría el papel impreso que se utiliza en la actualidad y tendríamos una información actualizada y fácil de consultar en cualquier momento.

Para el personal que no disponga de ordenador, tendrá disponible en la INTRANET un formulario de solicitud de permisos y licencias rellenable, en formato PDF, que entregará en el Servicio de personal, siendo los empleados de este departamento los encargados de introducir los permisos y licencias en la nueva aplicación.

2. REQUISITOS. HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO DE LA APLICACIÓN

2.1. Restricciones económicas, tecnológicas, legales y operativas

2.1.1. Desarrollo en entorno WEB

Aprovechar las fortalezas de PHP y del Framework (gvHIDRA) actualmente usado en la Consellería para la presentación Web.

2.1.2. Utilización de Software OpenSource.

Con la finalidad de ahorrar costes en licencias de software, la Consellería de Infraestructuras y Transporte se haya inmersa en una política encaminada hacia el mundo del OpenSource, es decir desarrollar software con herramientas libres y sin coste económico.

2.1.3. La petición de los permisos se debe hacer desde la INTRANET.

2.1.4. Elección del Sistema de Gestión de Bases de Datos(POSTGRESQL)

Para no desviarnos de la linea de trabajo actual de la Consellería y con la finalidad de ahorrar costes en licencias de software, el SGBD elegido para albergar nuestros datos, será POSTGRESQL. Como se ha comentado anteriormente se trata de utilizar herramientas libres y sin coste económico.

2.1.5. Entorno de desarrollo configurable (IDE) (ECLIPSE)

El entorno de desarrollo elegido ha sido la herramienta ECLIPSE, entorno de desarrollo creado por IBM que se distribuye como herramienta Open Source.

Se puede instalar en cualquier sistema operativo que tenga instalada una máquina virtual Java.

Se trata de una herramienta extensible mediante plugins, los cuales le dicen a Eclipse como trabajar con determinados recursos (distintos tipos de lenguajes, Java, PHP, C/C++, etc)

En eclipse instalaremos un repositorio CVS, que es donde se irá subiendo las partes de código realizado y comprobado.

2.1.6. Elección del Lenguaje / Herramienta de desarrollo.(PHP/gvHIDRA)

El lenguaje elegido para el desarrollo será PHP, por lo que deberemos cargar en eclipse los plugings correspondientes de PHP (perpestiva PHP).

¿Qué es gvHIDRA?

gvHidra son las iniciales de Generalitat Valenciana: Herramienta Integral de Desarrollo Rápido de Aplicaciones.

Si hay que dar una explicación rápida de qué es este proyecto, podemos decir que se trata de un entorno de trabajo (framework) para el desarrollo de aplicaciones de gestión en entornos web con PHP siguiendo una guía de estilo (una guía para unificar los criterios de aspecto y usabilidad en el proceso de desarrollo de aplicaciones).

Evidentemente, esto no responde a la pregunta que da título a este punto. Por ello, vamos a explicar las motivaciones que llevaron a su creación y las características que cubre.

Un poco de historia

El Servicio de Organización e Informática (SOI) de la Conselleria de Infraestructuras y Transporte de la Generalitat Valenciana (CIT) ha trabajado tradicionalmente con la premisa de aumentar la productividad en base a entornos de trabajo que facilitan el desarrollo de aplicaciones. Concretamente, una de sus líneas de desarrollo, utilizaba unas plantillas en PowerBuilder que, en una arquitectura cliente servidor, reducían los tiempos de desarrollos resolviendo la parte general del problema.

Dentro de este marco, la CIT emprendió el proyecto gvPONTIS cuyo objetivo principal era migrar todos los sistemas a sistemas de código abierto. Entre otras facetas afectadas se encontraban los lenguajes de programación (se seleccionaron PHP y Java), los SGBD (se decidió fomentar el uso de Postgresql), la arquitectura (pasar al desarrollo web/tres capas)...

Con todo ello, se decidió crear un proyecto en PHP que, basándose en la guía de estilo de las aplicaciones de la CIT, aportara las mismas ventajas que las anteriores plantillas de PowerBuilder: aumentar la productiva de nuestros desarrollos.

Pero claro, a todos esos requerimientos, teníamos que añadir las dificultades que generaba el nuevo ámbito de trabajo: el entorno web (HTML, Javacript,...), el enfoque OpenSource, ... Por tanto se decidió incorporar como requerimiento la simplificación del entorno de trabajo para un desarrollador.

Con todo ello se creó un proyecto (igep: Implementación de la Guía de Estilo en Php) que componía el core del framework cuya primera versión estable salió el 16-11-2004. Este proyecto, al ser liberado con licencia GPL tomó la denominación gvHIDRA. A partir de este hito, empezaron a colaborar con el proyecto numerosas entidades públicas y privadas que nos ayudan a mantener el proyecto vivo y actualizado.

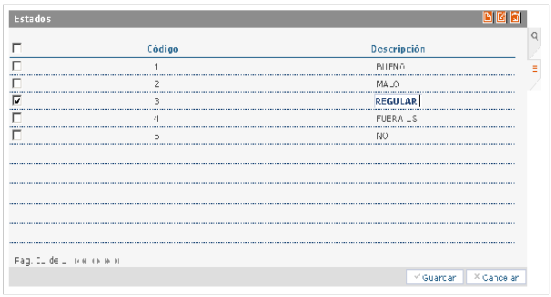
Pero ¿Qué es gvHIDRA?

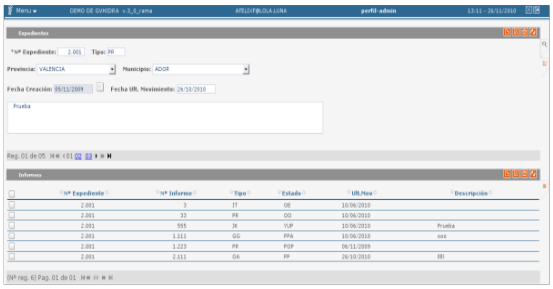
Con lo expuesto en el punto anterior, quedan claras las premisas que motivaron la creación de este proyecto pero, ¿Cómo se ha logrado unos objetivos tan ambiciosos? Bien, es difícil afirmar que se han alcanzado todos, pero seguramente sí, buena parte de ellos, quedan resueltos en el proyecto con las siguientes características del framework:

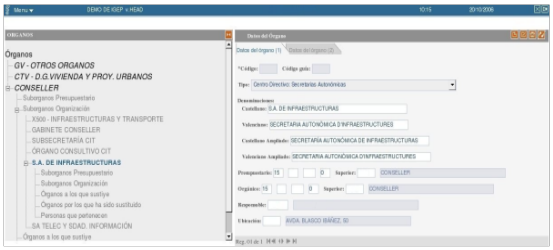
- Patrones de interfaz

Una de las tareas más costosas es en la definición de la interfaz con el usuario. Para facilitar el trabajo, se han definido una serie de patrones de básicos. Estos patrones definen la forma de representación de la información (formato tabular, registro,...) y la forma con la que interacciona el usuario con dicha información. Esto nos lleva a que, con la selección de un patrón, obtenemos el diseño global de la ventana.

Ejemplos de patrón simple tabular, maestro detalle y árbol.

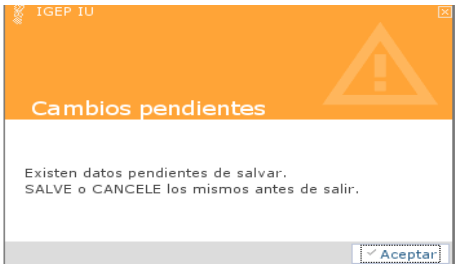






• Componentes complejos

La experiencia acumulada nos dice que todas las aplicaciones necesitan de una serie de componentes “complejos”. Ventanas de selección (en PB listas de valores), listas enlazadas, acciones de interfaz (en WEB tec. AJAX), mensajes de información,... El framework genera estos componentes simplificando su utilización en las aplicaciones.



• Operaciones preprogramadas y parametrizables

Al igual que con los componentes, se advertió que cierta problemática se repite en todas las aplicaciones. Por esa razón, se generalizó y resolvió en el framework, siendo incorporada a las aplicaciones de forma transparente. Algunos ejemplos son:

- Control de acceso concurrente: es importante que el garantizar la integridad de los cambios realizados por ello el framework incorpora de forma transparente un mecanismo de control.
- CRUD: el framework genera las sentencias SQL necesarias para Crear, Leer, Actualizar y Borrar un registro.
- Persistencia y validación de tipos: completando lo que nos ofrece el PHP, el framework incorpora objectos persistentes y validación de tipos de datos.

• Soporte a diferentes SGBD

A través del proyecto PEAR::MDB2, el framework permite trabajar con diversos SGBD. Además, incorpora un capa de intermedia propia del framework que nos permite independizarnos de las diferentes interpretaciones del SQL que hace cada gestor (definición de límites, transacciones, ...).

• Listados e informes

Uno de los mayores retos, en el ámbito del proyecto, fue generar informes de forma tan versatil como los datawindows de PowerBuilder. Se logró gracias al proyecto jasperreports. Apoyados por herramientas como el iReport, se consiguen listados e informes muy completos y en diferentes formatos.

• Arquitectura MVC

El framework garantiza la arquitectura MVC en todos nuestros desarrollos, forzando la separación de la lógica de negocio de la presentación mediante la distribución física de los ficheros fija.

• Control de la vista

Uno de los objetivos principales de la herramienta es simplificar la labor del programador. Con esta premisa, el proyecto gvHIDRA ha conseguido que un desarrollador de gvHIDRA realice una aplicación WEB sin necesidad de introducir ninguna línea de HTML o Javascript. La idea es que centre su trabajo únicamente en PHP, siendo así mucho más productivo.

• Custom y temas

La herramienta está pensada para su despliegue en diferentes organizaciones, por ello, se distribuye en una arquitectura App/Custom/Core que permite modificar tanto el aspectos (CSS, imágenes, ...) como definir comportamientos propios de la organización.

• Testing

Incorpora la herramienta PHPUnit para que se puedan realizar testeos sobre el código generado. También se incorporan consejos y reglas para poder realizar test automáticos son Selenium.

• Autentificación, auditoría y depuración

Para poder incorporarse en diferentes organismos incorpora un mecanismo de validación extensible siendo capaz de acloplarse a cualquier sistema de validación a través de PHP. Dispone de herramientas para reliazar auditorías y depuración.

Como buen proyecto opensource, el proyecto sigue en constante evolución incorporando nuevas funcionalidades que nos permitan, ante todo, ser más productivos.

Entendiendo el entorno

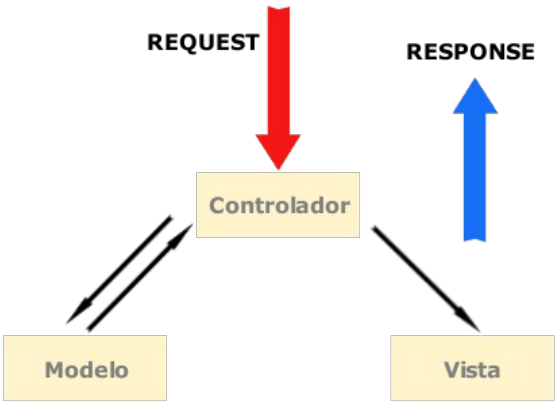
El framework gvHIDRA sigue una arquitectura MVC. Esta arquitectura tiene como objeto dividir en capas diferentes la lógica de negocio, la lógica de control y la presentación. Con ello conseguimos desarrollos más robustos y fuertes ante los cambios de requisitos. El siguiente diagrama muestra como funciona de forma esquemática la relación entre capas:

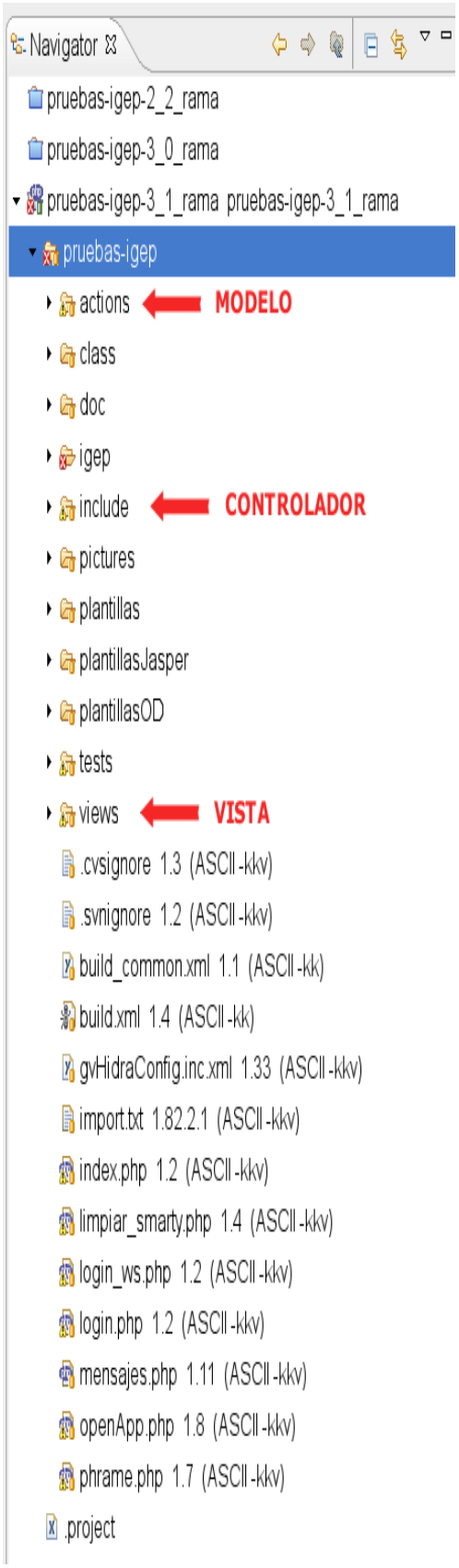
Esta división en capas se plasma físicamente esta división dentro de una aplicación gvHIDRA. Concretamente, para hacer un mantenimiento, el programador tendrá que trabajar en las siguientes partes de la estructura:

• actions: corresponde al modelo y en el se alojan las clases manejadoras. Estas clases nos permiten acceder a los datos y realizar los tratamientos sobre ellos. Muchos de los tratamientos y comportamientos vienen heredados de forma que nuestra clase simplemente debe añadir el comportamiento específico.

• include/mappings.php: corresponde con el controlador. Basado en el proyecto Phrame, se compone de un fichero que nos permite asociar una acción (solicitud del usuario) con la clase que lo va a resolver.
• views: corresponde con la vista. En este directorio encontraremos las vistas que designarán la pantalla a visulizarse.

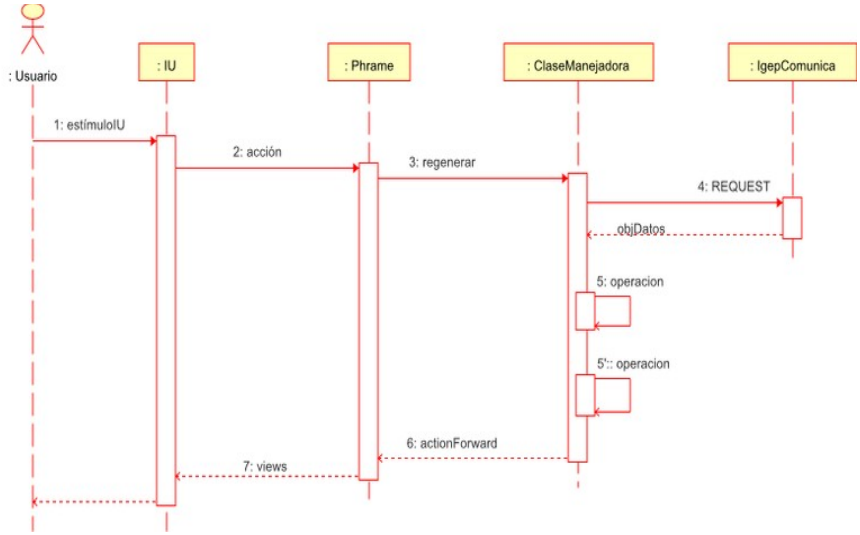
Aquí tenemos un esquema donde lo podemos ver un ejemplo de aplicación con las capas que componen la arquitectura MVC.




Navigator
pruebas-igep-2_2_rama
pruebas-igep-3_0_rama
pruebas-igep-3_1_rama pruebas-igep-3_1_rama
pruebas-igep
actions
MODELO
class
doc
igep
include
CONTROLADOR
pictures
plantillas
plantillasJasper
plantillasOD
tests
views
VISTA
.cvsignore 1.3 (ASCII-kkv)
.svnignore 1.2 (ASCII-kkv)
build_common.xml 1.1 (ASCII-kk)
build.xml 1.4 (ASCII-kk)
gvHidraConfig.inc.xml 1.33 (ASCII-kkv)
import.txt 1.82.2.1 (ASCII-kkv)
index.php 1.2 (ASCII-kkv)
limpiar_smarty.php 1.4 (ASCII-kkv)
login_ws.php 1.2 (ASCII-kkv)
login.php 1.2 (ASCII-kkv)
mensajes.php 1.11 (ASCII-kkv)
openApp.php 1.8 (ASCII-kkv)
phrame.php 1.7 (ASCII-kkv)
.project

A estas capas, falta añadir la interfaz con el usuario, las pantallas. Para la realización de las mismas, gvHIDRA hace uso de smarty y de unos plugins propios. Estos fichero, se ubican en el directorio templates y contienen la definición de una ventana con todos sus componentes.

Una vez vista la estructura física de una aplicación, es conveniente ver los componentes internos del framework a través del flujo interno que provoca una petición de pantalla. En el siguiente diagrama de secuencia, hemos colocado algunos de los actores que intervienen en el tratamiento de una petición así como sus operaciones.



1. El flujo se inicia con la aparición de un estímulo de pantalla lanzado por el usuario.

2. Este estímulo es transmitido al controler (en nuestro caso phrame) en forma de acción. Ahora es phrame quien, consultado con el fichero de mapeos (fichero mappings.php) es capaz de conocer que clase es la encargada de gestionar la acción. Una vez conocida la clase, la "levanta" y le cede el control pasándole todos los datos de la petición.

3. La clase (conocida como clase manejadora) una vez tiene el control realiza varios pasos.
   - Reconecta de forma automática a la base de datos (en el caso de que exista).
   - Parsea el contenido de la petición (REQUEST) encapsulando en un objeto iterador que agrupa el contenido en matrices por operación.
   - Lanza las distintas operaciones. Estas operaciones se extienden con el comportamiento extra que añade el programador.

Es decir, si lanzamos una acción de borrado, el framework realiza las operaciones necesarias para eliminar la tupla de la base de datos y el usuario tiene dos puntos de extensión opcional de dicho comportamiento: antes de borrar (métodos pre: para validaciones) y después de borrar (métodos post: para operaciones encadenadas).

4. Una vez finalizado todo el proceso, la clase manejadora devuelve un actionForward a phrame. Este lo descompone y se localiza la views seleccionada.

5. Finalmente, el views recoge la información y, con la plantilla (fichero tpl de smarty) muestra la información en pantalla.

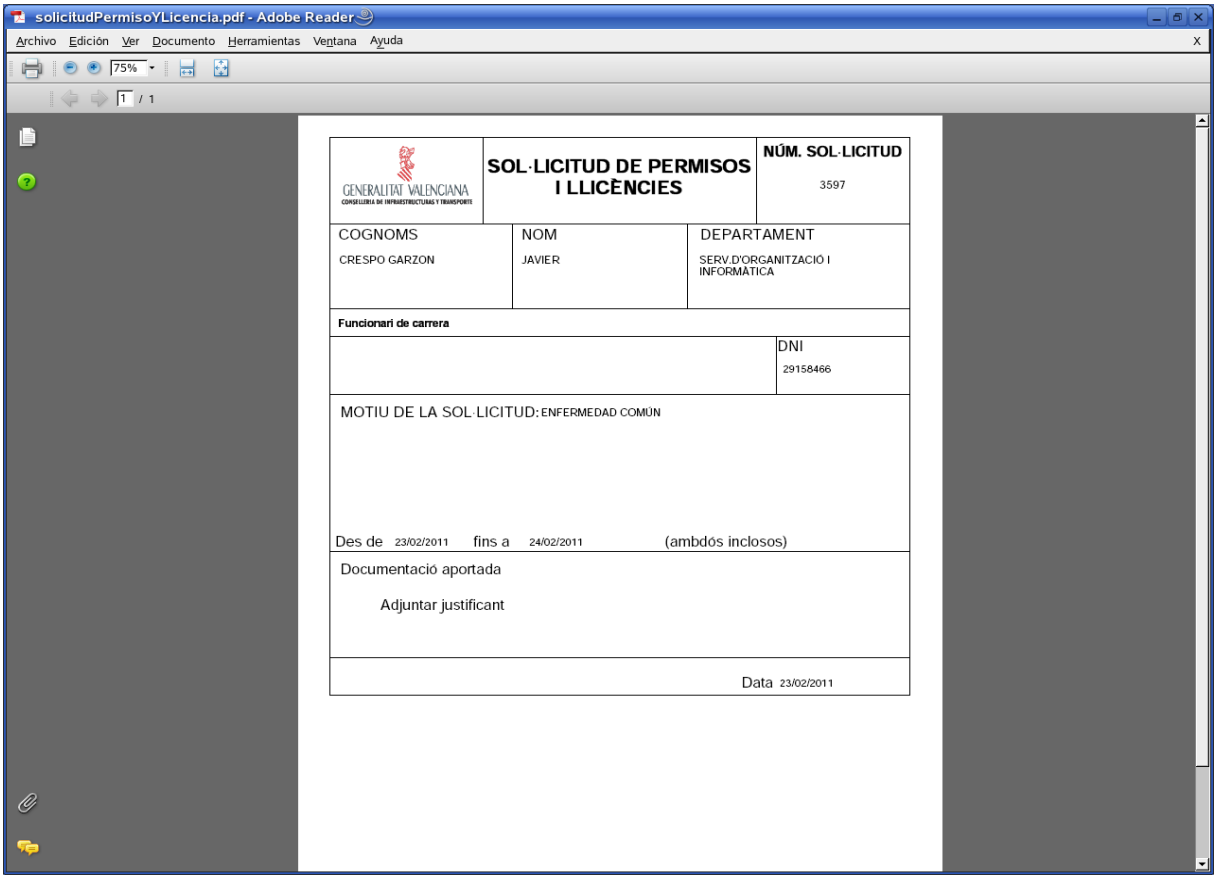
2.1.7. El motor de informes JasperReports y la herramienta iReport

Para la generación de informes, se ha optado por el motor JasperReports y la herramienta visual iReport que nos permitirán realizar los distintos informes y listados de la aplicación.

JasperReports es un motor de informes compuesto por una colección de librerías Java, y liberado bajo licencia abierta. Es un proyecto maduro que compite con los mejores generadores de informes propietarios,como pueda ser CrystalReports, etc...

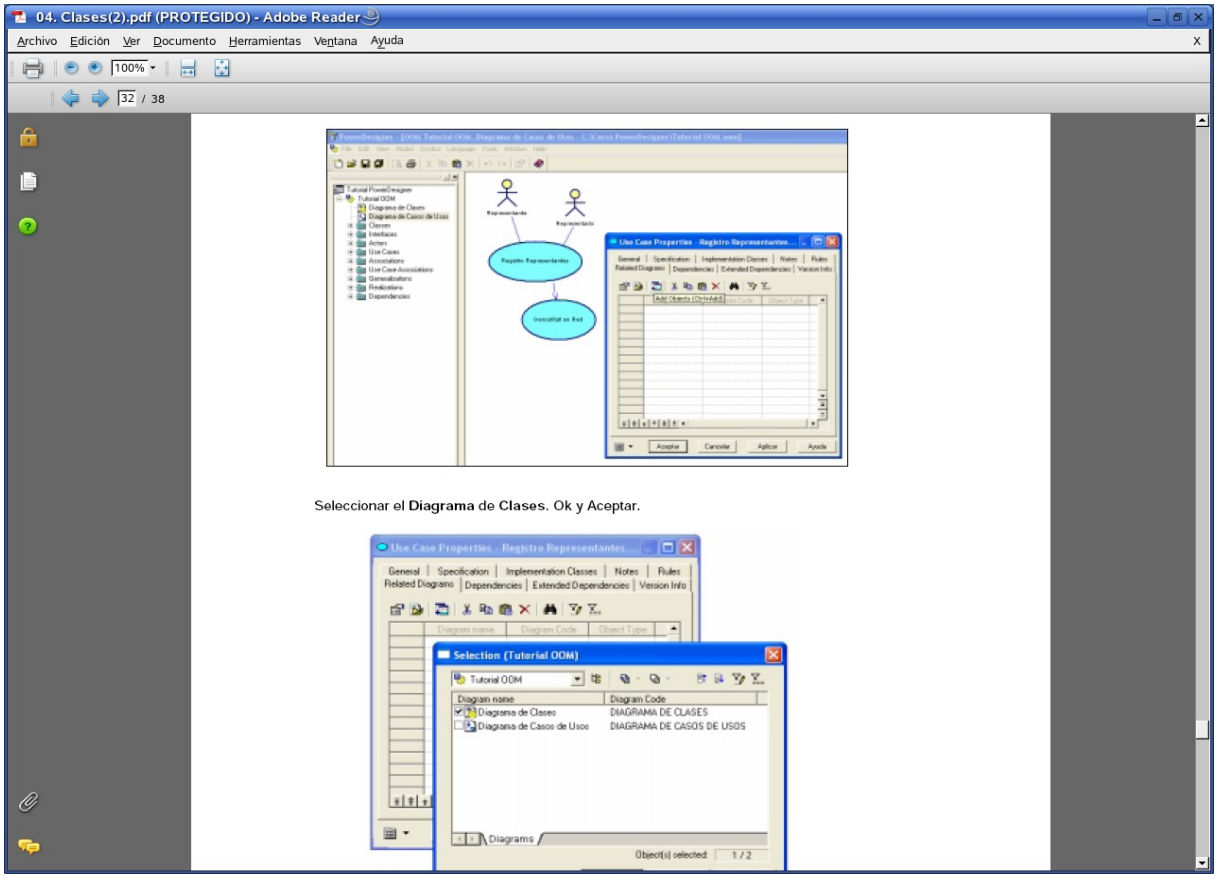
La herramienta iReport. El motor JasperReports, ha sido ampliado con otros proyectos que se encargan de facilitar el diseño de informes de forma visual. Entre los mismos, actualmente destaca de forma notable la herramienta elegida por nosotros para elaborar informes, iReport

Estas dos herramientas, nos permitirán generar los diversos informes requeridos por la aplicación de Personal de la Consellería, Por ejemplo, cuando un empleado solicita una solicitud, se generará una copia de su solicitud.



2.1.8. Power Designer

La herramienta elegida para realizar el análisis y disño del proyecto ha sido el PowerDesigner , ya que permite utilizar técnicas de modelamiento formales, pero sencillas, que captura reglas del negocio, procesos y requerimientos. Los requerimientos del negocio, documentados en la fase de análisis, disparan el desarrollo de la aplicación y el diseño de la base de datos.



3. REQUISITOS. DESCRIPCIÓN DE LA APLICACIÓN

El usuario dará de alta sus solicitudes de permisos y licencias y su responsable las validará. A continuación, pasarán al Servicio de personal todas las solicitudes para ser validadas por el tramitador, el revisor y el responsable de personal, antes de pasar a la firma del Subsecretario. Tan sólo se grabarán en la base de datos de personal aquellas solicitudes que hayan sido autorizadas por el Subsecretario.

El usuario podrá anular y borrar sus solicitudes, dependiendo del estado en que se encuentren, es decir, si su solicitud todavía no ha sido validada por su responsable, sólo podrá borrarla, lo que equivaldría a romper el papel que se rellena en la actualidad. En caso contrario, deberá anularla.

Para comprender mejor el alta y la anulación de una solicitud de permisos y licencia, así como ver los posibles estados por los que puede pasar una solicitud desde que se da de alta o anula, hasta que se autoriza, ver anexos A. y B.

3.1. Estados de una solicitud

Los estados por los que puede pasar una solicitud de permisos y licencias son los siguientes:

1. *Pendiente validar jefe servicio*: se encontrarán en este estado aquellas solicitudes de permisos y licencias que hayan sido dadas de alta por el interesado y no hayan sido validadas por su responsable.
2. *Afecta a las necesidades del servicio*: se encontrarán en este estado aquellas solicitudes de permisos y licencias que hayan sido dadas por no conforme por su responsable.
3. *Pendiente de autorización*: se encontrarán en este estado aquellas solicitudes de permisos y licencias :
    - cuyo interesado no está conforme con la decisión tomada por su responsable de no dar por conforme su solicitud, por lo que pasarán a ser validadas por el responsable de personal,
    - que hayan sido dadas por conforme por su responsable y no las haya validado aún el tramitador de personal.
    - que hayan sido validadas previamente por el tramitador de personal y no las haya validado aún el revisor de personal.
    - que hayan sido validadas por el revisor de personal y no las haya validado aún el responsable de personal,
    - que hayan sido dadas por conforme o no conforme por el responsable de personal y estén pendientes de la firma del Subsecretario o de la DGAA, según sea el caso.
1. *Autorizada/ No autorizada*: se encontrarán en este estado aquellas solicitudes de permisos y licencias que hayan sido firmadas por el Subsecretario o la DGAA, según sea el caso.

3.2. Tipos de mensajes

- *informativos*, que indican un resultado correcto,

- *avisos*, que indican situaciones anómalas pero no tienen porqué indicar que el permiso no es correcto, aunque tienen que ser revisados y validados por el Servicio de personal y aceptados o no, según el caso,

- *errores*, que impiden la introducción de un permiso.

3.3. Personas que intervienen en la tramitación de una solicitud

- *Usuario de la CIT*: realiza el alta de solicitudes de permisos y licencias. Una vez introducidos los datos requeridos por la pantalla de alta y devuelto los mensajes necesarios, la solicitud ya está lista para ser procesada por el responsable, quien la validará.

- *Responsable (jefe de servicio)*: revisa los permisos solicitados por las personas a su cargo que aún no hayan sido revisados por él, dándole conformidad o no según corresponda. Una vez validadas las solicitudes como conformes, pasarán a ser revisadas por el tramitador de personal.

- *Servicio de personal*: será el *tramitador de personal* la persona encargada de revisar todas las solicitudes, tanto si tienen mensajes de avisos como si no, requieran o no adjuntar un justificante o se encuentren en estado de no conformidad del interesado, que aún no hayan sido revisadas por él, dándolas por válidas o no según corresponda. A continuación, será el *revisor de personal* quien revisará las solicitudes que previamente hayan sido validadas por el tramitador de personal. Después, el *responsable de personal* se encargará de dar por válidas o no las solicitudes ya validadas por el revisor de personal.
    - En caso de ser resueltas por la CIT, si se dan por válidas, se emitirá una resolución positiva de todos los permisos seleccionados y, si se dan por no válidas, se emitirá una resolución negativa de cada permiso individualmente. Ambas se enviarán a la firma del Subsecretario que será la persona encargada de autorizar o no las solicitudes.
    - En caso que la solicitud sea tramitada por la DGAA (Dirección General de Administración Autonómica), no se emitirá resolución sino que será éste organismo quien autorice o no la solicitud.

  A continuación, todas las solicitudes autorizadas, se grabarán como permiso en la base de datos de personal.

3.4. Tipos de usuarios

- *Usuario de la CIT*: usuarios que consultan, dan de alta, anulan o borran sus solicitudes de permisos y licencias, y consultan las fichadas del mes y del mes anterior, y los días de licencia por asuntos propios y vacaciones.
  Los tipos de usuarios que se detallan a continuación, también dispondrán de estas funciones para la gestión de sus propias solicitudes.

- *Tramitador de personal*: usuarios del Servicio de personal que,
    - dan de alta las solicitudes de permisos y licencias de aquellas personas que no dispongan de ordenador, indicando en el alta, el nombre de la persona que realiza la solicitud,
    - dan de alta las solicitudes de tipo no solicitables (por ejemplo, IT, ...), o bien las solicitudes de tipo solicitables que se sigan rellenando a través de las hojas rosas, las cuales se grabarán directamente en la base de datos de personal, puesto que no tienen que pasar por ninguna firma, ya que es el propio Servicio de personal el que lo realiza.
    - validan todas las solicitudes, antes de pasar a ser validadas por el revisor de personal.

- *Revisor de personal*: usuarios del Servicio de personal que,
    - revisan las solicitudes que hayan sido validadas previamente por el tramitador de personal, antes de pasar a ser validadas por el responsable de personal.

- *Responsable*: usuarios responsables que dan conformidad o no a las solicitudes del personal a su cargo.

- *Responsable de personal*: usuarios del Servicio de personal que,
    - realizan la conformidad o no de las solicitudes del personal a su cargo y dan por válidas o no las solicitudes de todo el personal de la CIT, previa validación del revisor de personal,
    - realizan la impresión de las resoluciones, tanto positivas como negativas, para la firma del Subsecretario. Tras la firma, se darán de alta en la base de datos de personal aquellas solicitudes que se relacionen en la resolución positiva.

3.5. Menú de la aplicación

Según el tipo de usuario, se mostrarán más o menos opciones de menú.

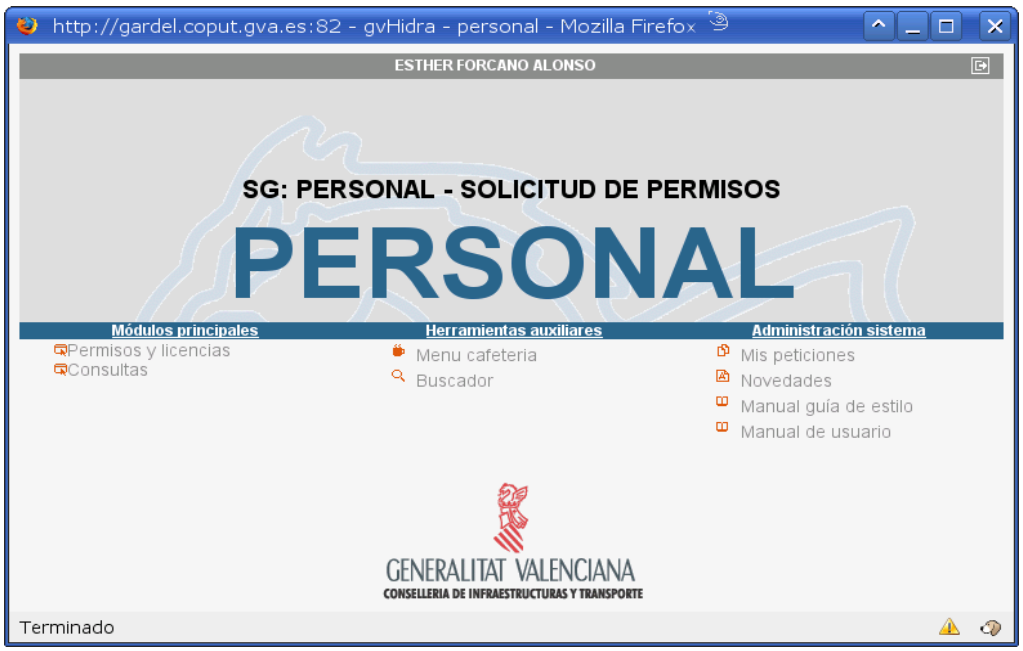


Figura 2.5.1: Menú para el usuario de la CIT.

3.6. Alta

Finalidad

En esta pantalla se realiza el alta de solicitudes de permisos y licencias.

Al acceder con el perfil usuario de la CIT, el comportamiento de la ventana obtiene la funcionalidad básica. Por tanto, podrá dar de alta solicitudes de permisos y licencias únicamente en su nombre, es decir, el usuario con el cual se conectó a la aplicación. Además, sólo podrán pedir solicitudes del periodo actual, excepto si son días de licencia por asuntos propios, en cuyo caso los podrán pedir para el periodo anterior sólo hasta el 15 de enero del periodo siguiente.

Tras dar de alta una solicitud, se enviará un correo a su responsable para notificárselo.

Funcionamiento

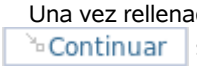
Perfil de acceso: Usuario de la CIT, tramitador, revisor, responsable y responsable de personal.

Forma de acceso: Desde el menú de aplicación *[Permisos y licencias][Alta]*.

La *pantalla de introducir nuevo registro* (Figura 3.1.2 y 3.1.3), al acceder con perfil usuario de la CIT, los flujos de navegación por la ventana serán los siguientes:

Al acceder a la ventana, veremos en un campo de texto no editable llamado “Usuario”, el nombre de quién está realizando la solicitud. En este caso, coincidirá con el usuario que accedió a la aplicación. Mediante una lista desplegable llamada “Motivo” podremos seleccionar la clase de solicitud que deseamos cursar. Ésta aparecerá ordenada según determinados criterios acordados por el Servicio de personal. Este desplegable tiene un comportamiento dinámico que mostrará/ocultará campos necesarios a rellenar según el tipo de solicitud elegida. (NOTA: No utilice los atajos de teclado para cambiar entre los elementos de la lista ya que éstos no ocultarán/mostrarán los campos necesarios para el motivo seleccionado).

Los campos a rellenar de carácter obligatorio se mostrarán marcados por el símbolo asterisco “ * ”. Éstos variarán según el motivo seleccionado, aunque existen algunos, como son las fechas, que siempre tendrá que rellenar. Para rellenar los campos de “Comienzo” y “Fin” puede usar el teclado numérico, o bien podrá hacer uso de la herramienta de calendario pulsando el botón adjunto al campo, el cual le permitirá elegir la fecha desde un calendario. Cuando se introduce una fecha en “Comienzo” de un mes diferente al actual, al pulsar en el calendario de la fecha de “Fin”, se muestra el mismo mes/año de la fecha de “Comienzo”. El campo “Observaciones” servirá para que añada cualquier información que desee que llegue al responsable de validación de su solicitud.

Una vez rellenados todos los campos obligatorios y los que haya considerado oportunos, podrá pulsar el botón 

situado en el lado inferior derecho de la ventana, el cual le llevará a una ventana donde aparecerá un resumen, cálculos de totales de días utilizados y posibles avisos añadidos a la solicitud en curso. En esta ventana podrá finalmente guardar su solicitud. En el caso de que la solicitud necesite adjuntar justificante, se deberá imprimir para adjuntarla a dicho justificante. Además, las solicitudes tramitadas por la Dirección General de Administraciones Autonómicas, como requieren la firma del interesado, del jefe del servicio y del Subsecretario / Director general, también deberán imprimirse para poderlas firmar.

La *pantalla de impreso* (Figura 3.1.4), nos muestra el impreso de la solicitud de permisos y licencias que el usuario acaba de dar de alta. Sólo accesible si la solicitud lo requiere, tanto porque se necesite adjuntar un justificante a la solicitud o al permiso como porque sea una solicitud tramitada por la Dirección General de Administraciones Autonómicas.

Pantallas del programa

Pantalla de introducir nuevo registro:

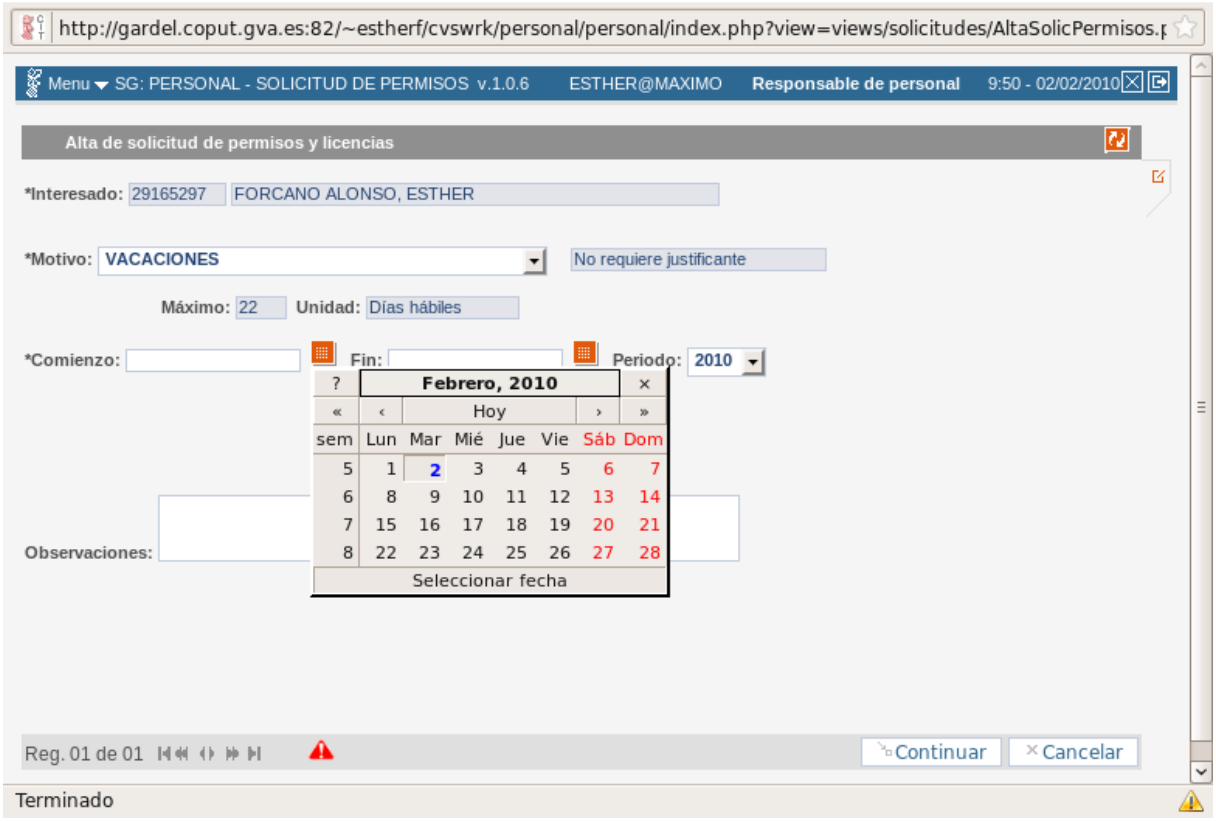


Figura 3.1.1: Uso del calendario

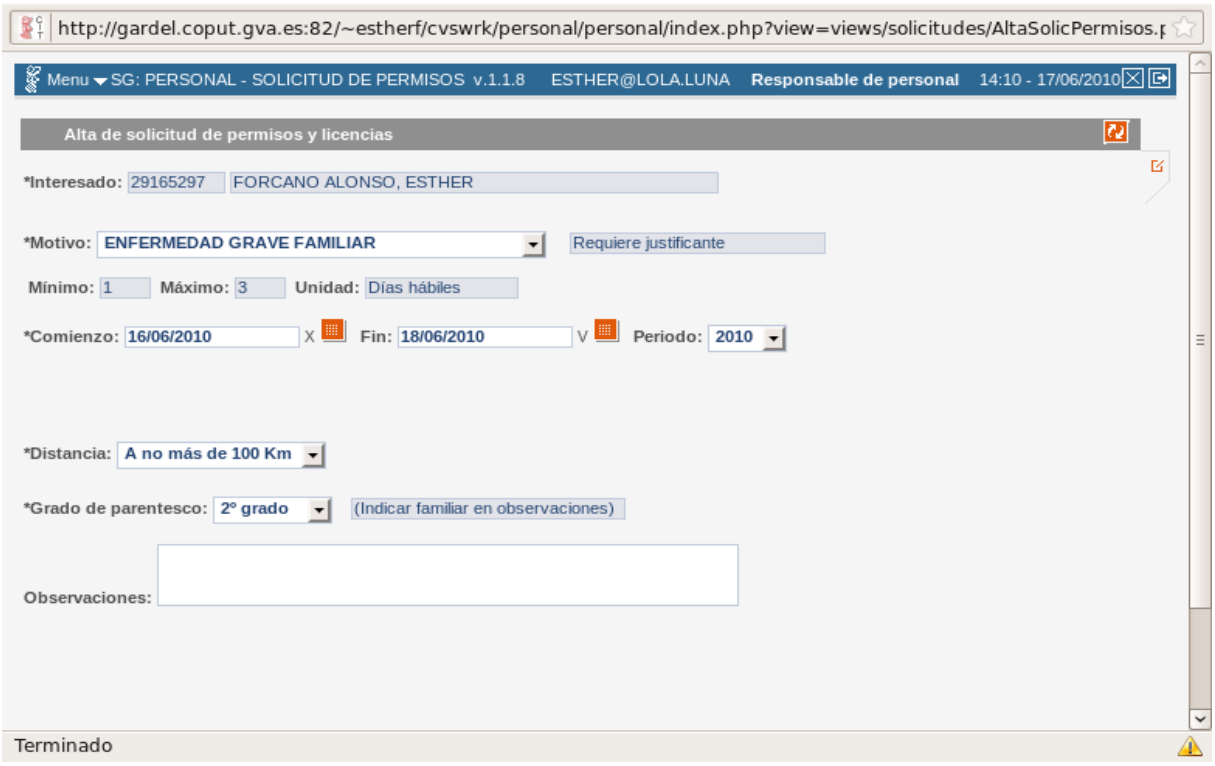


Figura 3.1.2: Pantalla de alta de una solicitud "Enfermedad grave familiar"

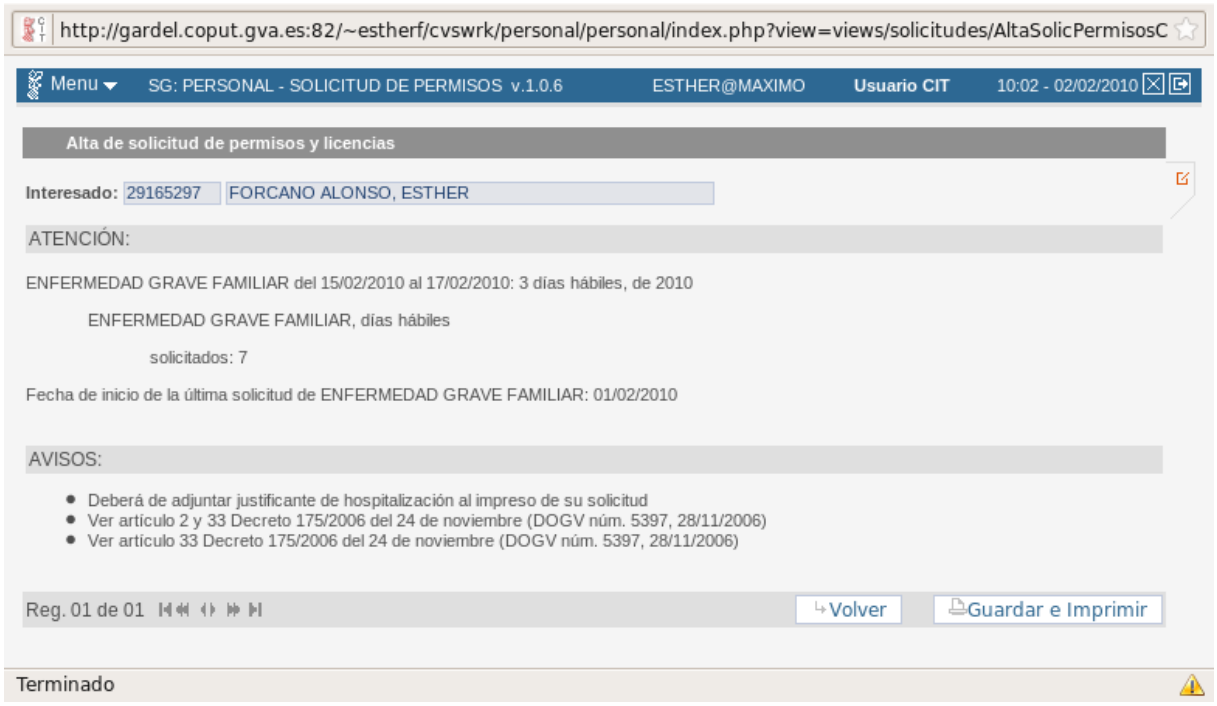


Figura 3.1.3: Pantalla de información de la solicitud

Pantalla de impreso:

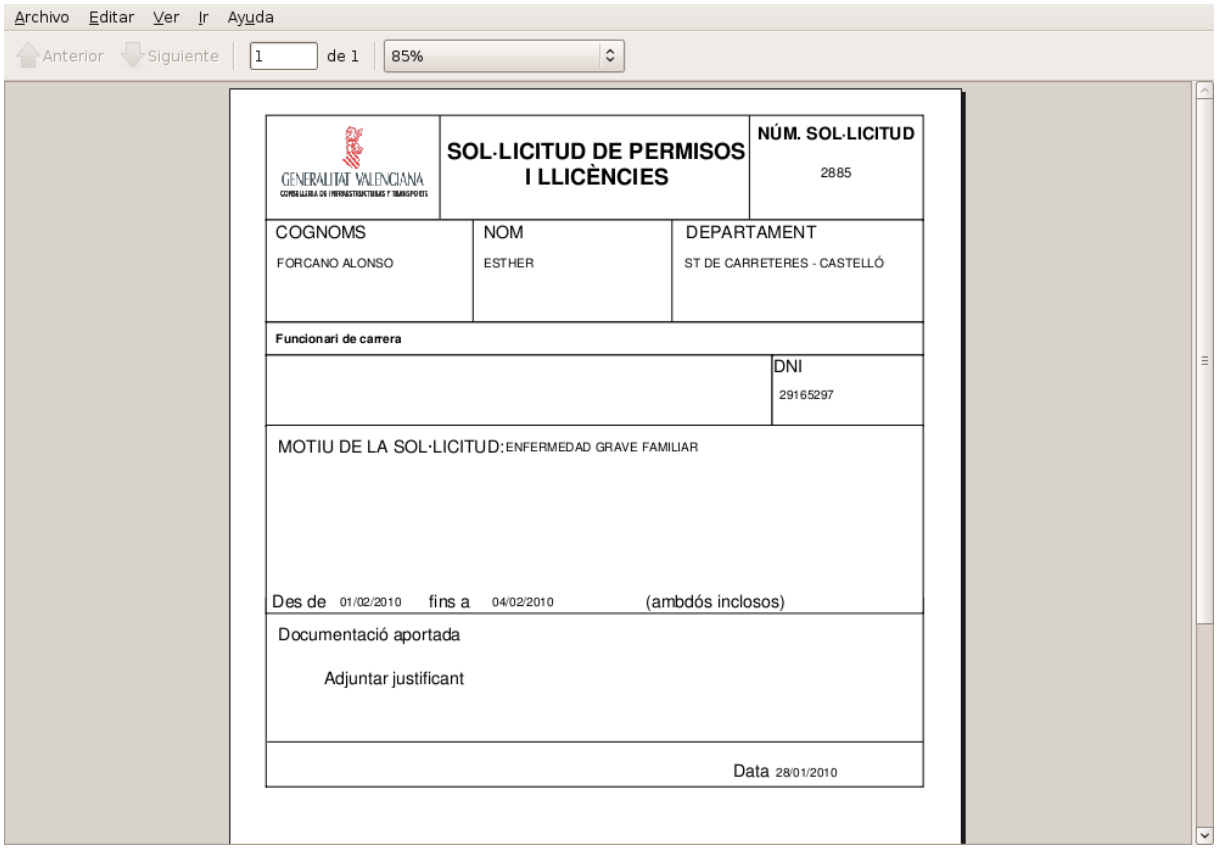


Figura 3.1.4: Pantalla de impreso de una solicitud de "Enfermedad grave familiar"

Campos de las pantallas

Pantalla de introducir nuevo registro:

Usuario: Nombre de la persona conectada a la aplicación. [Obligatorio][No editable]

Motivo: Lista desplegable de clases de permisos y licencias solicitables. [Obligatorio][Seleccionable]

Periodo: Lista desplegable de periodos anuales para el permiso o licencia que se solicita. [Obligatorio][Seleccionable]

Comienzo: Fecha de inicio del permiso o licencia que se solicita. [Obligatorio][Editable]

Fin: Fecha de fin del permiso o licencia que se solicita. [Obligatorio/Opcional según el motivo][Editable]

Día de la semana: Indicadores del día de la semana correspondientes a las fechas de comienzo y de fin. [No editable]

Observaciones: Área de texto para incluir comentarios relacionados con la clase de permiso o licencia que se solicita. [Opcional][Editable]

Fecha nacimiento del menor: Fecha de nacimiento del menor para los motivos de “Acumulación jornada por lactancia”, “Flexibilidad de permanencia obligatoria”, “Reducción 1h. diaria por lactancia” y “Reducción diaria guarda legal”. [Obligatorio según el motivo][Editable]

Distancia: Lista desplegable referente a la distancia en Km (más de 100 km o menos) para los motivos de “Enfermedad grave familiar”, “Fallecimiento” y “Traslado de domicilio habitual”, o bien, (más de 375 Km o menos) para los motivos “Matrimonio o unión de hecho (día celebr)” y “Matrimonio o unión de hecho familiar”. [Obligatorio][Seleccionable]

Grado de parentesco: Lista desplegable referente al grado de parentesco de un familiar respecto al interesado (1er grado, 2º grado), para los motivos de “Enfermedad grave familiar” y “Fallecimiento”. [Obligatorio][Seleccionable]

Adjunta justificante: Indica si el interesado aporta o no justificante para el motivo de “Enfermedad comun”. [Obligatorio][Seleccionable]

Pantalla de información de la solicitud:

Interesado: Nombre de la persona para la cual se tramita la solicitud. [No editable]

Información de la solicitud de permisos y licencias que el interesado está dando de alta. [No editable]

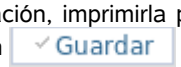
Avisos para el Servicio de personal que pudieran estar relacionados con dicha solicitud. [No editable]

Botones de las pantallas

Botón que permite pasar a la pantalla de información de la solicitud donde, si estamos de acuerdo con la información que se muestra en pantalla, podremos guardar 

la solicitud. En caso contrario, podremos volver 

a la pantalla de alta.

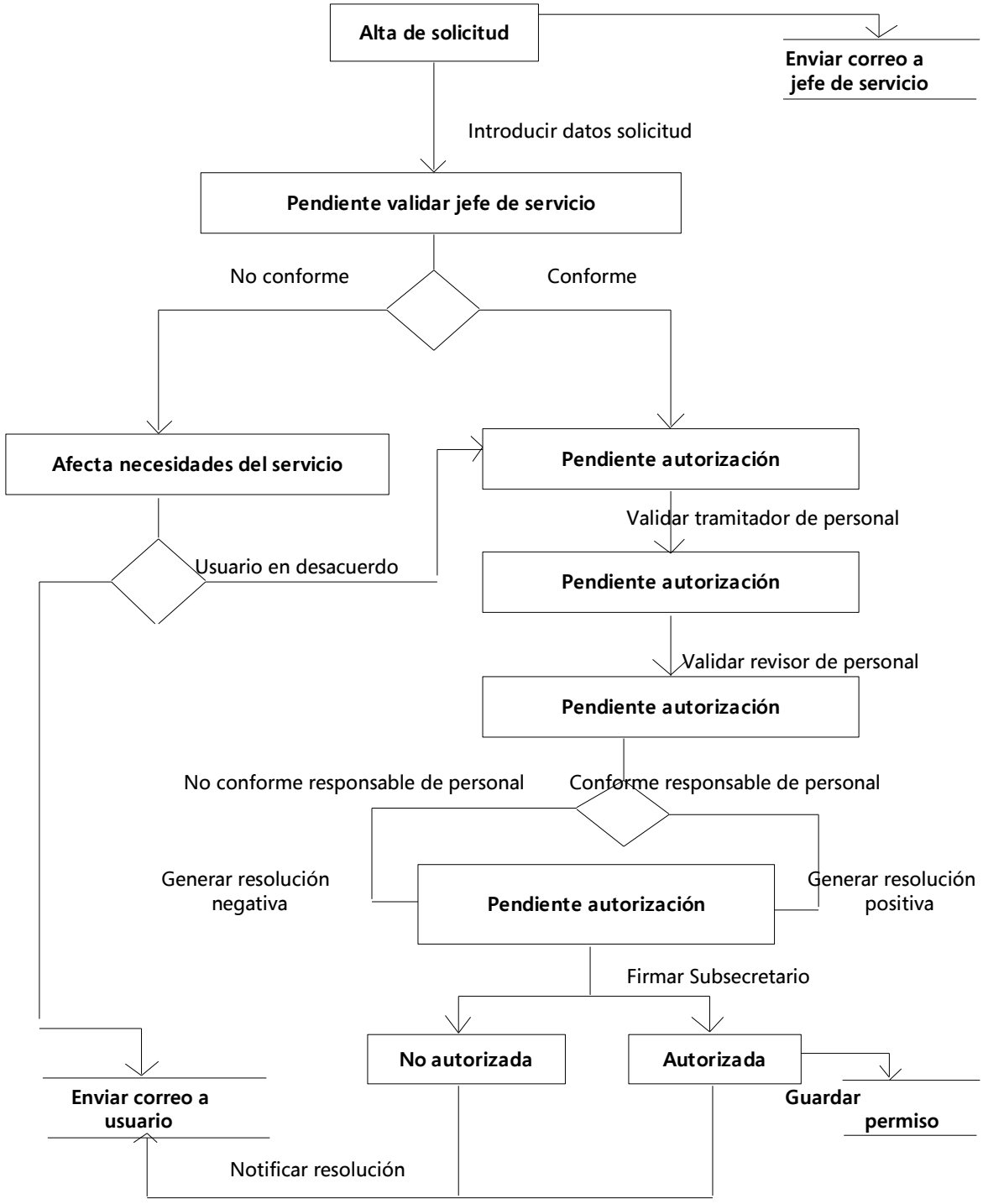
Botón visible en la pantalla de información para las solicitudes que requieran adjuntar justificante y las tramitadas por la Dirección General de Administración Autonómica. La acción que realiza es guardar la solicitud y, a continuación, imprimirla para poderla entregar en el Servicio de personal. Cuando es visible este botón, el botón 

no será visible.

 *Calendario*

 *Limpiar campos*

Diagrama de procesos del alta de solicitud de permisos y licencias



3.7. Anulación

Finalidad

En esta pantalla se realizará la anulación de todas aquellas altas de solicitudes de permisos y licencias que el interesado desee dejar sin efecto. Además, deben cumplir los siguientes requisitos:

- tengan firma de responsable, o de responsable y personal,
- no se hayan empezado a disfrutar, en cuyo caso, se actualiza el estado de la solicitud,
- se haya disfrutado algún día, en cuyo caso, habrá que modificar la parte de la solicitud que aún no se ha disfrutado, y crear tantas solicitudes nuevas como intervalos que ya se hayan disfrutado.

Tras la anulación de solicitudes de permisos y licencias, se enviará un correo a su responsable para notificárselo.

Funcionamiento

Perfil de acceso: Usuario de la CIT, tramitador, revisor, responsable y responsable de personal.

Forma de acceso: Desde el menú de aplicación *[Permisos y licencias][Anulación]*.

La *pantalla de listado* (Figura 3.2.1), nos muestra el listado de las solicitudes de permisos y licencias que el usuario puede anular. Podremos consultar el detalle de la solicitud o anular Anular la solicitud previamente seleccionada.

La *pantalla de detalle* (Figura 3.2.2), muestra el detalle de la solicitud seleccionada. Se accede pulsando sobre el botón de la *pantalla de listado*.

Pantallas del programa

Pantalla de listado:

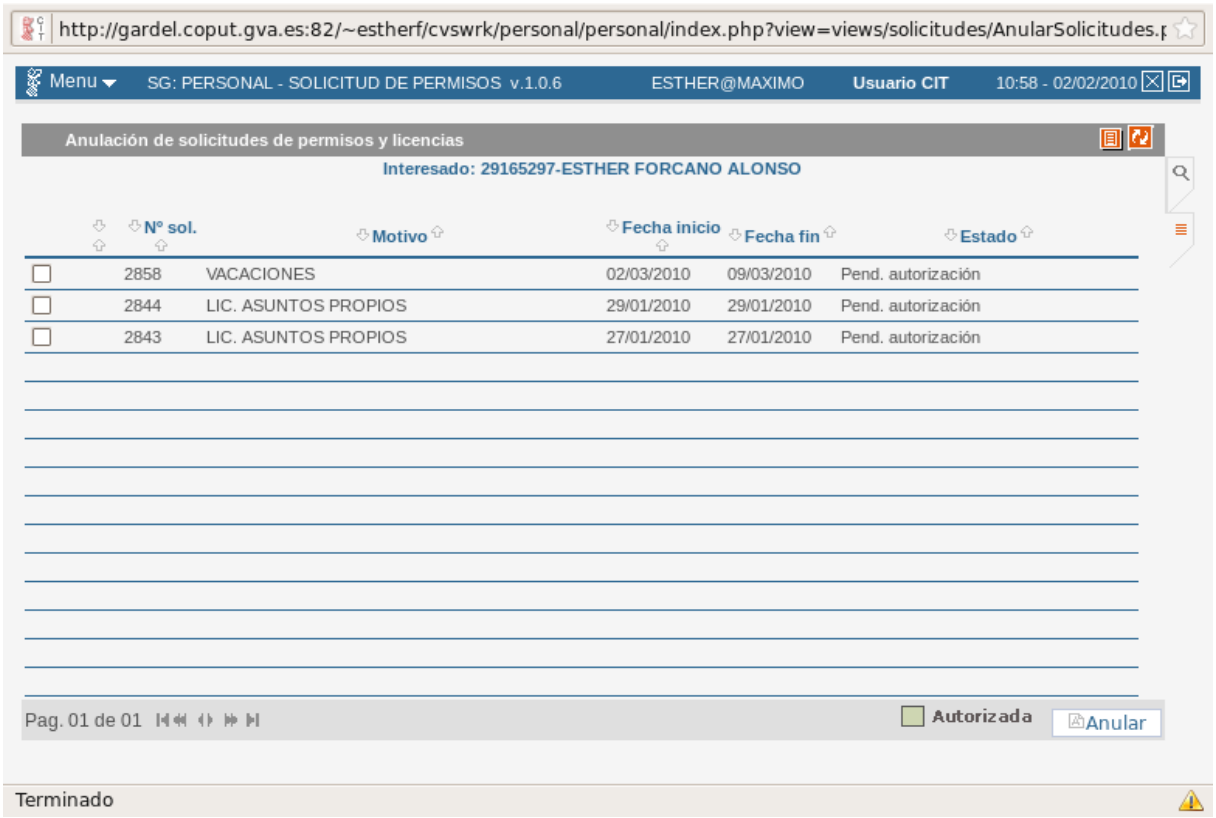


Figura 3.2.1 Pantalla de listado de solicitudes de permisos y licencias a anular

Pantalla de detalle:

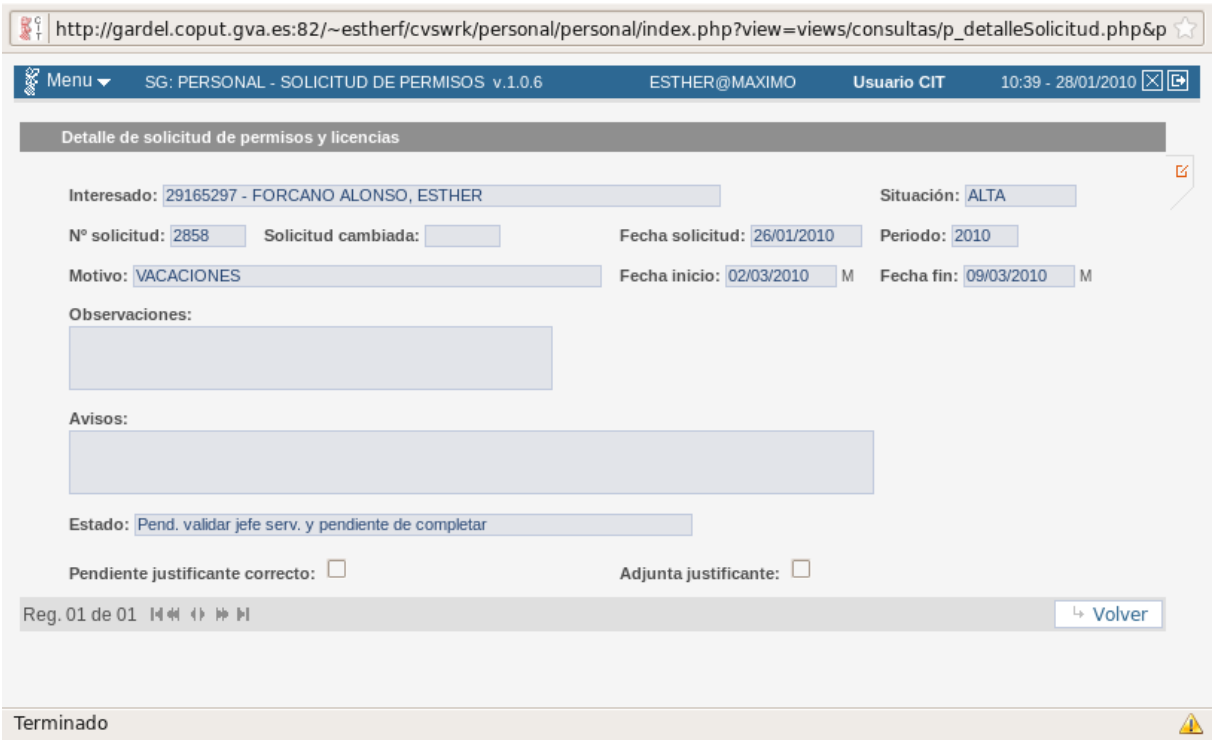


Figura 3.2.3 Pantalla de detalle de la solicitud de permisos o licencias

Campos y botones de las pantallas

Pantalla de listado:

Nº sol.: Nº de la solicitud de permisos y licencias. [No editable]
Motivo: Descripción de la clase de permiso. [No editable]
Fecha inicio: Fecha de inicio de la solicitud de permisos y licencias. [No editable]
Fecha fin: Fecha de fin de la solicitud de permisos y licencias. [No editable]
Estado: Estado en que se encuentra la solicitud de permisos y licencias: Pendiente validar jefe servicio / Afecta a las necesidades del servicio / Pendiente autorización / Autorizada / No autorizada. [No editable]

Checkbox

Indica que la solicitud tiene avisos para el Servicio de personal.

Sólo aparecerá cuando sea una solicitud de vacaciones pendiente de completar, es decir, no se hayan solicitado todas las vacaciones en su totalidad, para un determinado periodo.

Sólo aparecerá cuando sea una solicitud de vacaciones pendiente de completar y, además, tenga avisos para el Servicio de personal.

Autorizada El *fondo verde* en la solicitud de permisos y licencias indica que está autorizada.

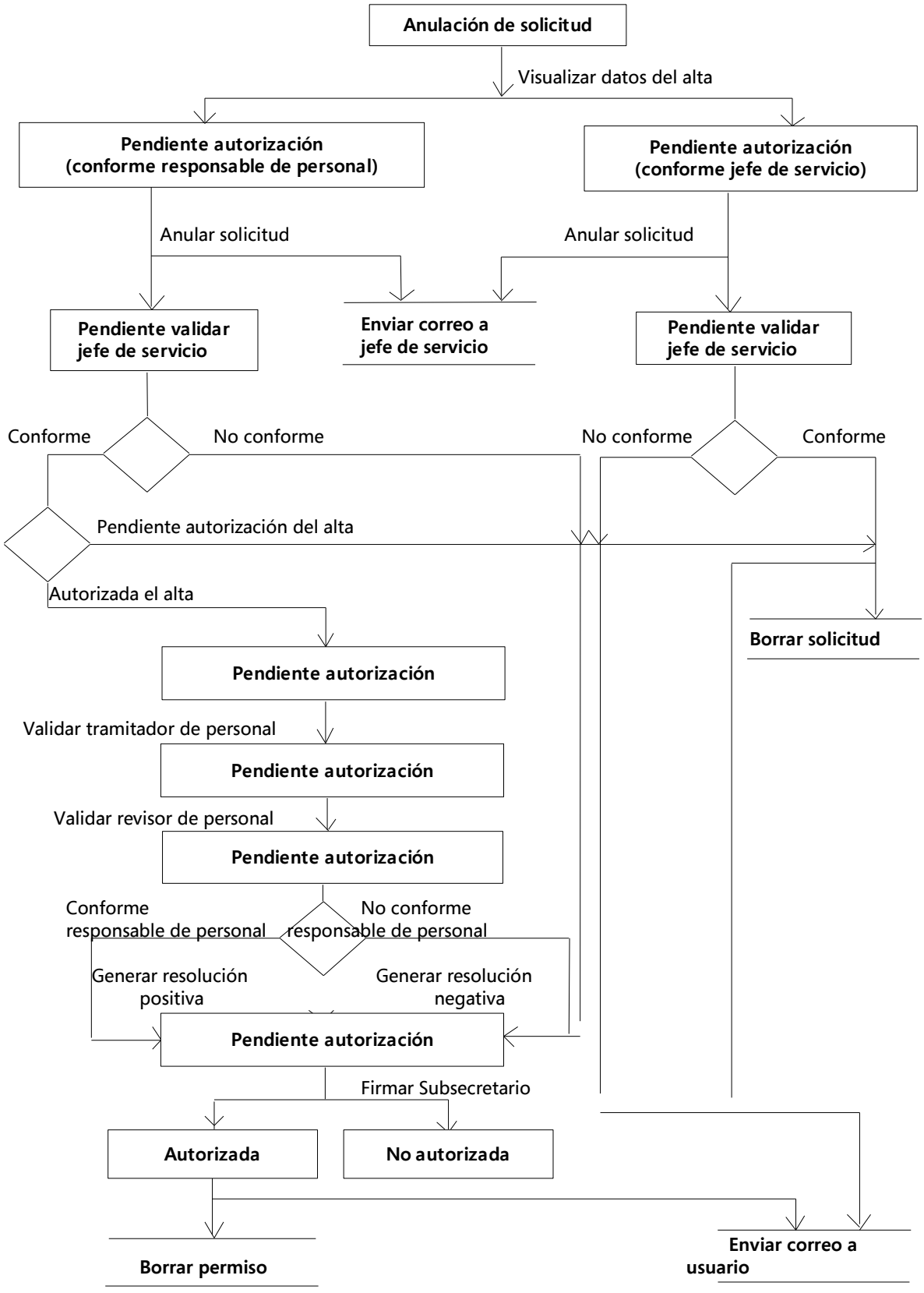
Anular Botón para *anular* las solicitudes de permisos y licencias seleccionadas previamente.

Ver Detalles (Figura 3.2.3)

Pantalla de detalle:

Interesado: Persona interesada de la solicitud de permisos o licencias. [No editable]
Situación: Indica la situación de alta de la solicitud de permisos o licencias. [No editable]
Nº solicitud: Nº identificador de la solicitud de permisos o licencias. [No editable]
Solicitud cambiada: Nº identificador de la solicitud de permisos o licencias a la que sustituye, para los motivos de "Vacaciones" y "Días adicionales vacaciones". [No editable]
Fecha solicitud: Fecha en que se realizó la solicitud de permisos o licencias. [No editable]
Periodo: Indica el año al que corresponde la solicitud de permisos o licencias.[No editable]
Motivo: Descripción de la clase de permisos o licencias. [No editable]
Fecha inicio: Fecha de inicio de la solicitud de permisos o licencias.[No editable]
Fecha fin: Fecha de fin de la solicitud de permisos o licencias.[No editable]
Día de la semana: Indicadores del día de la semana correspondientes a las fechas de inicio y de fin. [No editable]
Observaciones: Comentarios relacionados con la solicitud de permisos o licencias.[No editable]
Avisos: Expresamos los avisos que se mostraron al usuario cuando dio de alta su solicitud.[No editable]
Estado: Indica el estado en el que se encuentra la solicitud: Pendiente validar jefe servicio / Afecta a las necesidades del servicio / Pendiente autorización / Autorizada / No autorizada. [No editable]
Pendiente justificante correcto: Indica si el interesado ha presentado un nuevo justificante correcto requerido. [No editable]
Adjunta justificante: Indica si el interesado ha presentado el justificante requerido. [No editable]

Diagrama de procesos de la anulación de solicitud de permisos y licencias



3.8. Borrado

Finalidad

En esta pantalla, se realizará el borrado de todas aquellas solicitudes de permisos y licencias que se hayan dado de alta previamente, tanto altas como anulaciones, siempre y cuando sean:

- Altas de vacaciones pendientes de completar.
- Altas pendientes de validar por el responsable.
- Anulaciones pendientes de validar por el responsable.

El borrado de las altas eliminará la solicitud de la base de datos. En cambio, cuando se borre una anulación, la solicitud volverá al estado previo a la anulación, es decir, volverá a ser un alta.

Funcionamiento

Perfil de acceso: Usuario de la CIT, tramitador, revisor, responsable y responsable de personal.

Forma de acceso: Desde el menú de aplicación *[Permisos y licencias][Borrado]*.

La *pantalla de listado* (Figura 3.3.1), nos muestra el listado de las solicitudes de permisos y licencias que el usuario puede borrar. Podremos consultar el detalle de la solicitud o borrar Borrar la solicitud previamente seleccionada.

La *pantalla de detalle* (Figura 3.3.2), muestra el detalle de la solicitud seleccionada. Se accede pulsando sobre el botón de la *pantalla de listado*.

Pantallas del programa

Pantalla de listado:

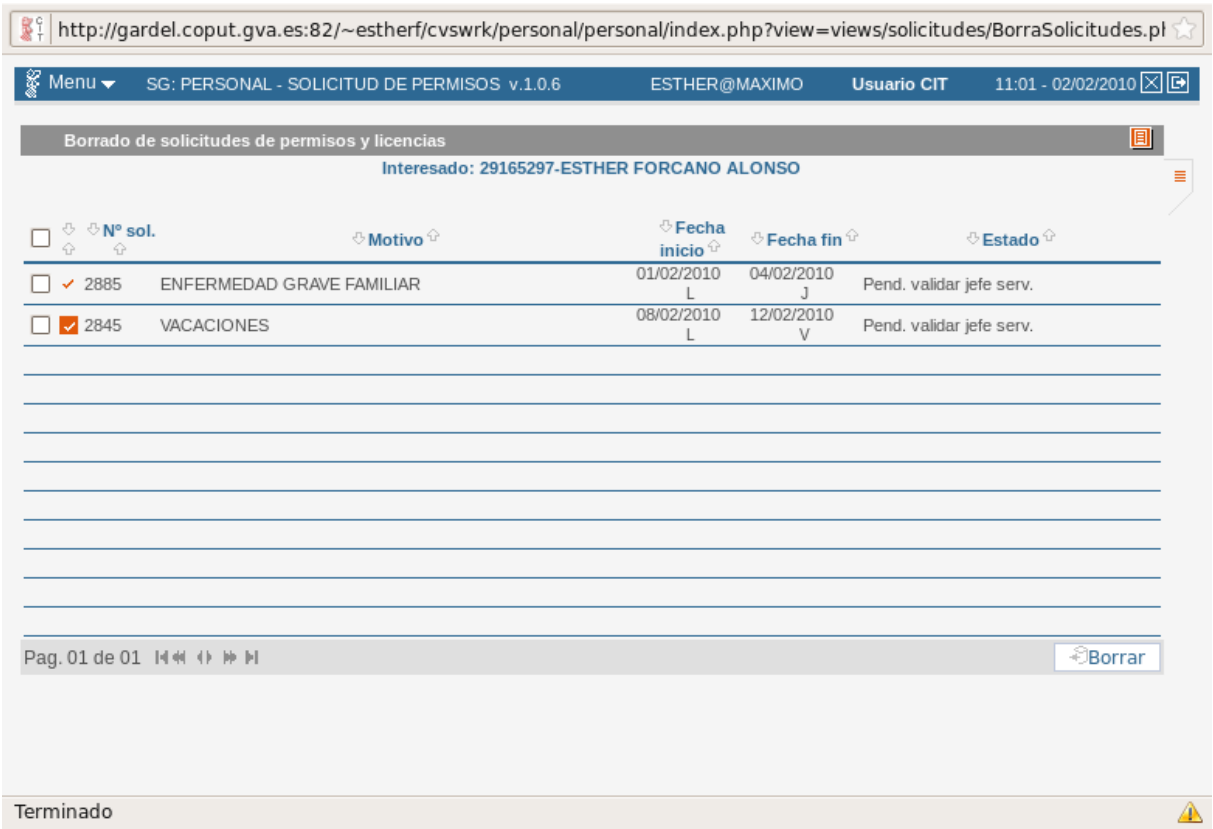


Figura 3.3.1 Pantalla de listado de solicitudes de permisos y licencias a borrar

Pantalla de detalle:

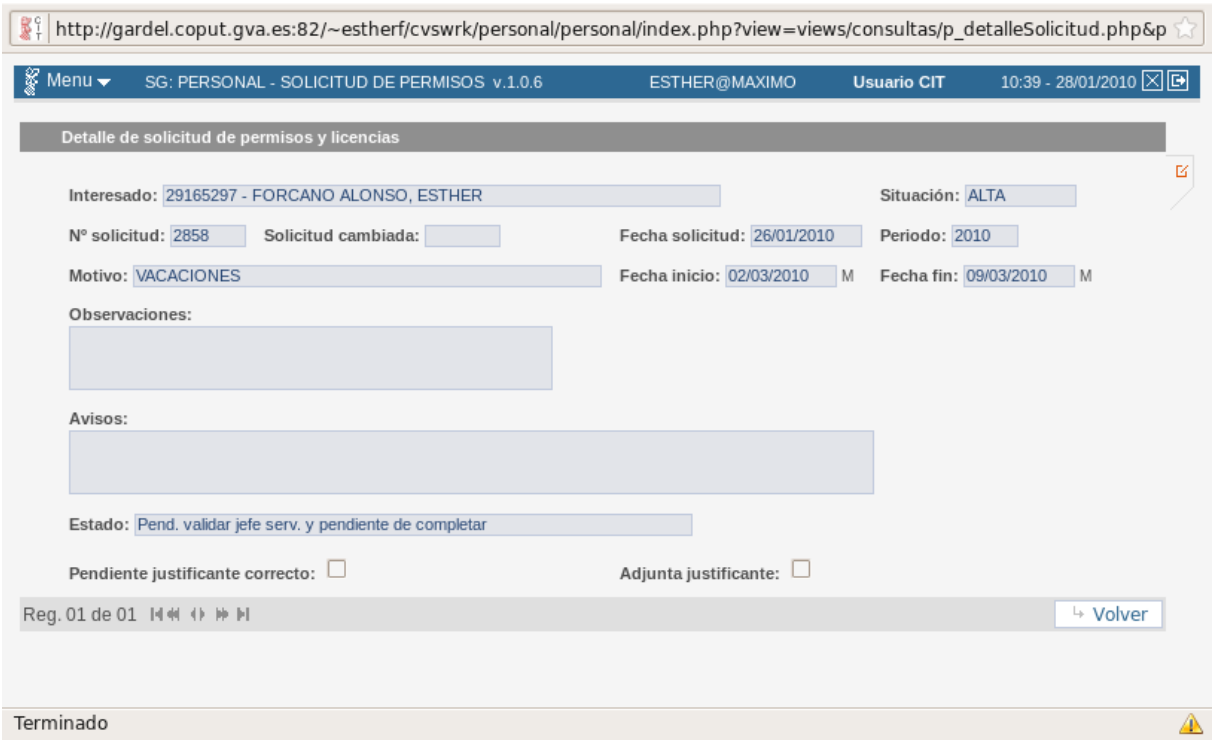


Figura 3.3.2 Pantalla de detalle de la solicitud de permisos o licencias

Campos y botones de las pantallas

Pantalla de listado:

Nº sol.: Nº de la solicitud de permisos y licencias. [No editable]
Motivo: Descripción de la clase de permiso. [No editable]
Fecha inicio: Fecha de inicio de la solicitud de permisos y licencias. [No editable]
Fecha fin: Fecha de fin de la solicitud de permisos y licencias. [No editable]
Estado: Estado en que se encuentra la solicitud de permisos y licencias: Pendiente validar jefe servicio / Afecta a las necesidades del servicio / Pendiente autorización / Autorizada / No autorizada. [No editable]

Checkbox

Indica que la solicitud tiene avisos para el Servicio de personal.

Sólo aparecerá cuando sea una solicitud de vacaciones pendiente de completar, es decir, no se hayan solicitado todas las vacaciones en su totalidad, para un determinado periodo.

Sólo aparecerá cuando sea una solicitud de vacaciones pendiente de completar y, además, tenga avisos para el Servicio de personal.

Borrar Botón para *borrar* las solicitudes de permisos y licencias seleccionadas previamente.

Ver Detalles (Figura 3.3.3)

Pantalla de detalle:

Interesado: Persona interesada de la solicitud de permisos o licencias. [No editable]
Situación: Indica la situación (Alta o anulación) de la solicitud de permisos o licencias. [No editable]
Nº solicitud: Nº identificador de la solicitud de permisos o licencias. [No editable]
Solicitud cambiada: Nº identificador de la solicitud de permisos o licencias a la que sustituye, para los motivos de "Vacaciones" y "Días adicionales vacaciones". [No editable]
Fecha solicitud: Fecha en que se realizó la solicitud de permisos o licencias. [No editable]
Periodo: Indica el año al que corresponde la solicitud de permisos o licencias.[No editable]
Motivo: Descripción de la clase de permisos o licencias. [No editable]
Fecha inicio: Fecha de inicio de la solicitud de permisos o licencias.[No editable]
Fecha fin: Fecha de fin de la solicitud de permisos o licencias.[No editable]
Día de la semana: Indicadores del día de la semana correspondientes a las fechas de inicio y de fin. [No editable]
Observaciones: Comentarios relacionados con la solicitud de permisos o licencias.[No editable]
Avisos: Expresamos los avisos que se mostraron al usuario cuando dio de alta su solicitud.[No editable]
Estado: Indica el estado en el que se encuentra la solicitud: Pendiente validar jefe servicio / Afecta a las necesidades del servicio / Pendiente autorización / Autorizada / No autorizada. [No editable]
Pendiente justificante correcto: Indica si el interesado ha presentado un nuevo justificante requerido. [No editable]
Adjunta justificante: Indica si el interesado ha presentado el justificante requerido. [No editable]

3.9. Control de presencia

Finalidad

En esta pantalla se mostrará la información de fichajes para el mes actual, incluyendo los fichajes del día, así como el saldo e incidencias, y también se podrá mostrar la información del mes anterior sin incluir el saldo.

Funcionamiento

Perfil de acceso: Usuario de la CIT, tramitador, revisor, responsable y responsable de personal.

Forma de acceso: Desde el menú de aplicación *[Consultas][Control de presencia][Fichadas del mes]* y *[Consultas][Control de presencia][Fichadas del mes anterior]*.

La pantalla listado (Figuras 4.1.1) nos muestra el listado de los fichajes del mes actual o anterior, según la opción de menú elegida. Si alguno de los fichajes de salida (Sal1, Sal2, Sal3, Sal4) va marcado con alguna señal (*, **, +, ++), indica que existe alguna incidencia en ese fichaje. Sólo en ese caso, podremos consultar el detalle del fichaje para ver la incidencia. Además, se podrá actualizar la información de los fichajes que se muestran.

La *pantalla listado en pdf* (Figura 4.1.2), nos muestra el listado de fichadas del mes actual o anterior, según la opción de menú elegida. Se accede pulsando sobre el botón 

de la *pantalla de listado*.

La *pantalla de detalle* (Figura 4.1.3), muestra el detalle del fichaje del día seleccionado siempre que haya alguna incidencia en alguno de los fichajes de ese día. Se accede pulsando sobre el botón  de la *pantalla de listado*.

Pantallas del programa

Pantalla listado del mes o del mes anterior:

Figura 4.1.1 Pantalla listado de control de presencia para las fichadas del mes o del mes anterior

Pantalla listado en pdf:

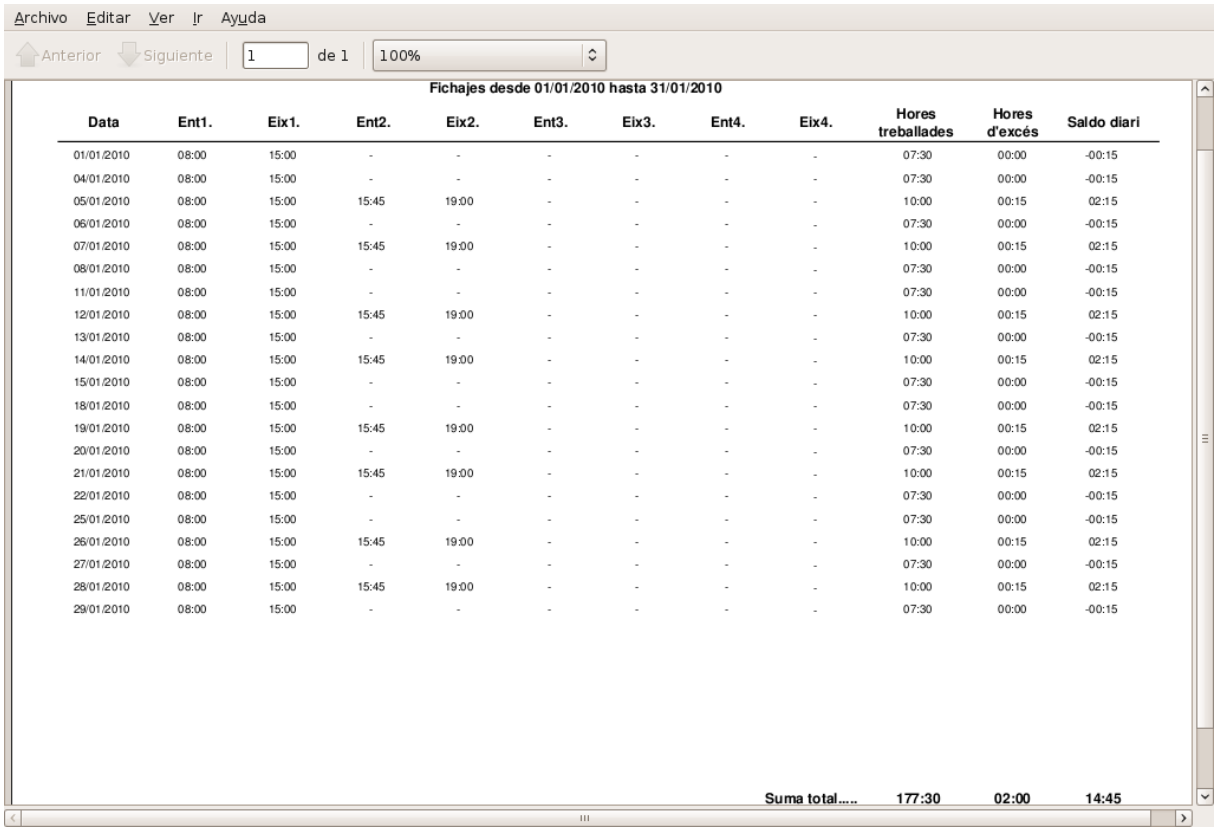


Figura 4.1.2 Pantalla listado de control de presencia en pdf para las fichadas del mes o del mes anterior

Pantalla de detalle:

Figura 4.1.3 Pantalla detalle de las fichadas del mes o del mes anterior

Campos y botones de las pantallas

Pantalla listado de fichadas del mes o del mes anterior:

En la *cabecera* se mostrará el total de horas trabajadas, el total de horas de exceso y el total de saldo diario disponible a fecha del día o a fin de mes, según se consulte el mes o el mes anterior, respectivamente..

Fecha: Fecha del día en el que hay fichajes o incidencias del interesado. [No editable]
Ent1: 1ª hora de entrada del interesado. [No editable]
Sal1: 1ª hora de salida del interesado. Puede llevar la señal * para indicar que existe alguna incidencia en ese fichaje. [No editable]
Ent2: 2ª hora de entrada del interesado. [No editable]
Sal2: 2ª hora de salida del interesado. Puede llevar la señal ** para indicar que existe alguna incidencia en ese fichaje. [No editable]
Ent3: 3ª hora de entrada del interesado. [No editable]
Sal3: 3ª hora de salida del interesado. Puede llevar la señal + para indicar que existe alguna incidencia en ese fichaje. [No editable]
Ent4: 4ª hora de entrada del interesado. [No editable]
Sal4: 4ª hora de salida del interesado. Puede llevar la señal ++ para indicar que existe alguna incidencia en ese fichaje. [No editable]
Horas trabajadas: Nº de horas trabajadas por el interesado en el día indicado en *Fecha*. [No editable]
Horas exceso: Nº de horas de exceso del interesado en el día indicado en *Fecha*. [No editable]
Saldo diario: Saldo del interesado una vez ha finalizado el día indicado en *Fecha*. Si es positivo indica saldo a su favor, y si es negativo indica el saldo que se debe. [No editable]
Incidencia: Incidencia de día completo. [No editable]

 *Checkbox*

 *Ver Detalles* (Figura 4.1.3)

 *Refrescar* la lista de fichajes

Indica que la incidencia de día completo está sin justificar

Botón para *ver el detalle de las fichadas del mes* en formato pdf.

Pantalla detalle de fichadas del mes o del mes anterior:

Fecha: Fecha del día en el que hay fichajes o incidencias del interesado. Viene indicado en el título del panel. [No editable]
Ent1: 1ª hora de entrada del interesado. [No editable]
Sal1: 1ª hora de salida del interesado. Puede llevar la señal * para indicar que existe alguna incidencia en ese fichaje. [No editable]
Ent2: 2ª hora de entrada del interesado. [No editable]
Sal2: 2ª hora de salida del interesado. Puede llevar la señal ** para indicar que existe alguna incidencia en ese fichaje. [No editable]
Ent3: 3ª hora de entrada del interesado. [No editable]
Sal3: 3ª hora de salida del interesado. Puede llevar la señal + para indicar que existe alguna incidencia en ese fichaje. [No editable]
Ent4: 4ª hora de entrada del interesado. [No editable]
Sal4: 4ª hora de salida del interesado. Puede llevar la señal ++ para indicar que existe alguna incidencia en ese fichaje. [No editable]
Horas trabajadas: Nº de horas trabajadas por el interesado en el día indicado en *Fecha*. [No editable]
Horas exceso: Nº de horas de exceso del interesado en el día indicado en *Fecha*. [No editable]
Saldo diario: Saldo del interesado una vez ha finalizado el día indicado en *Fecha*. Si es positivo indica saldo a su favor, y si es negativo indica el saldo que se debe. [No editable]
Incidencia: Incidencia del fichaje que lleve alguna señal (*, **, +, ++). [No editable]

3.10. Días de licencia por asuntos propios y vacaciones

Finalidad

En esta pantalla se mostrará la información de días de licencia de asuntos propios y de vacaciones solicitados por el interesado para el periodo actual y el anterior, visualizando el detalle y los totales solicitados. Los días se contarán como hábiles.

Además, cuando se consulta el periodo actual, se mostrará en la cabecera el total de días del periodo de asuntos propios, adicionales de asuntos particulares, vacaciones y adicionales de vacaciones que le corresponden al interesado en el periodo.

Funcionamiento

Perfil de acceso: Usuario de la CIT, tramitador, revisor, responsable y responsable de personal.

Forma de acceso: Desde el menú de aplicación *[Consultas][Días de licencia por asuntos propios y vacaciones][Periodo actual]* y *[Consultas][Días de licencia por asuntos propios y vacaciones][Periodo anterior]* .

La pantalla listado (Figuras 4.2.1 y 4.2.2) nos muestra el listado de los días de licencia por asuntos propios y vacaciones del periodo actual o anterior, según la opción de menú elegida.

Pantallas del programa

Pantalla listado:

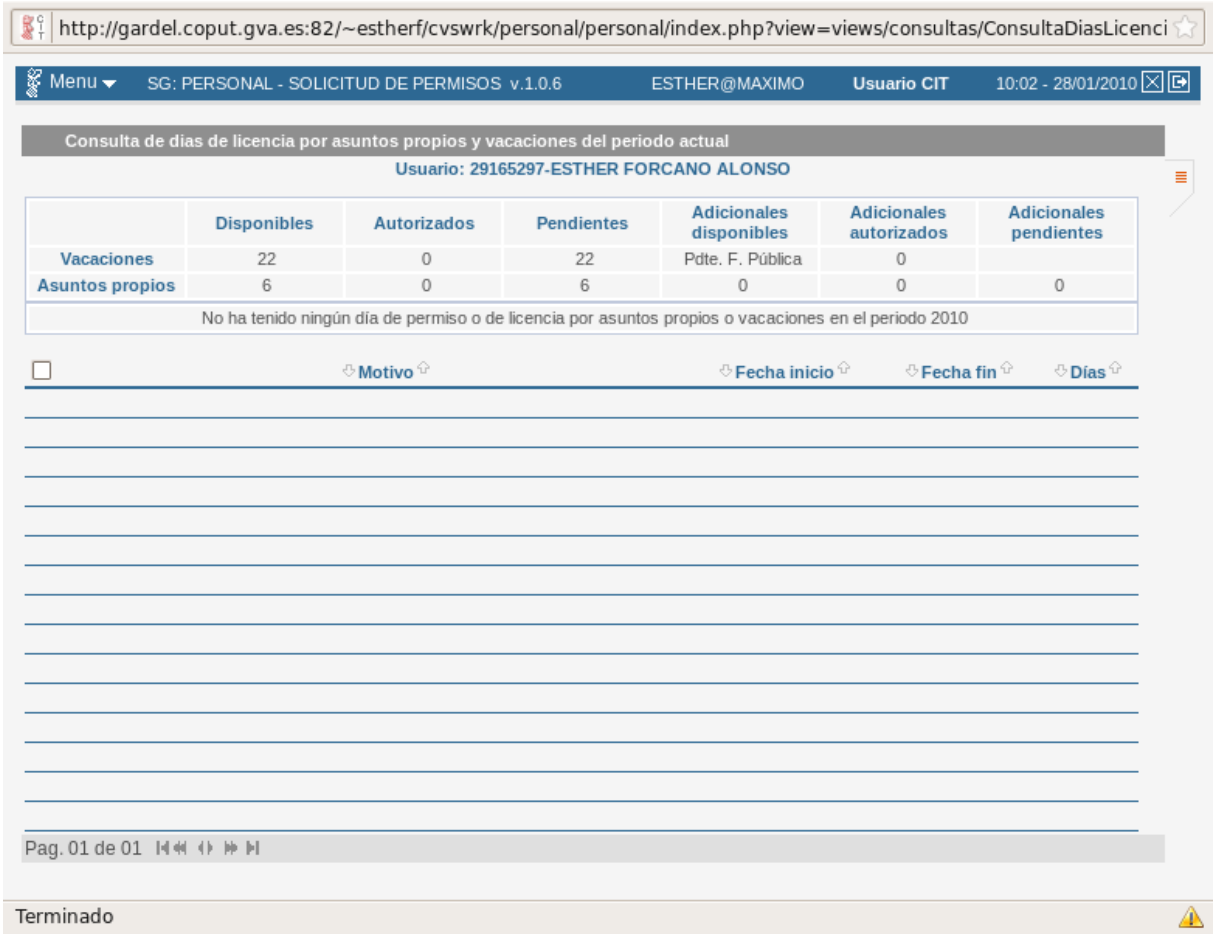


Figura 4.2.1 Pantalla consulta de días de licencia por asuntos propios y vacaciones del periodo actual

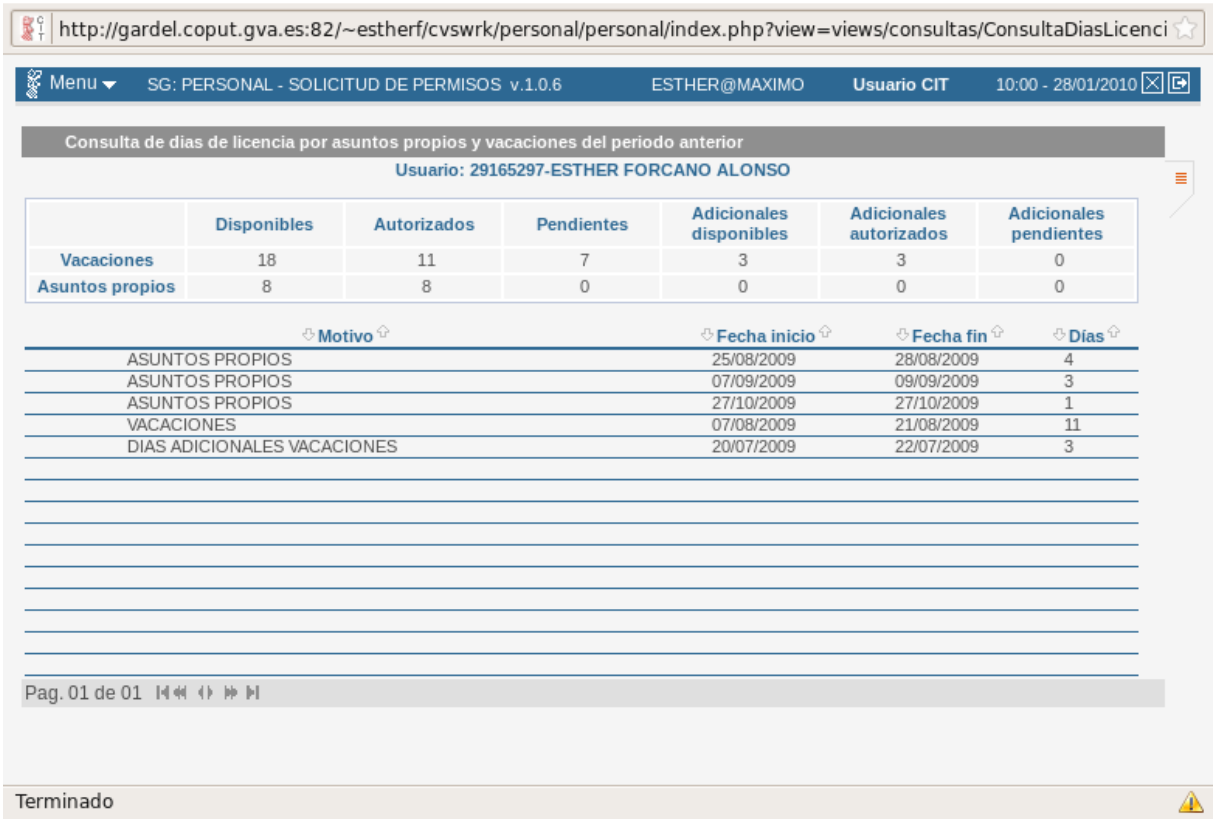


Figura 4.2.2 Pantalla consulta días de licencia por asuntos propios y vacaciones del periodo anterior

Campos de la pantalla

Pantalla de listado periodo actual:

En la *cabecera* se mostrará el total de días de los permisos o licencias de vacaciones y asuntos propios, y sus respectivos adicionales, disponibles, autorizados y pendientes en el periodo actual para el usuario interesado:

Vacaciones disponibles: Total días de vacaciones disponibles para el periodo actual que le corresponden al interesado.
Vacaciones autorizadas: Total días de vacaciones autorizadas al interesado para el periodo actual, que se detallan en el listado.
Vacaciones pendientes: Total días de vacaciones pendientes para el periodo actual que le corresponden al interesado.
Vacaciones adicionales disponibles: Total días de adicionales de vacaciones disponibles para el periodo actual que le corresponden al interesado. En caso de no disponer de dicha información, se indicará “Pdte. F. Pública”.
Vacaciones adicionales autorizadas: Total días de adicionales de vacaciones autorizadas al interesado para el periodo actual, que se detallan en el listado.
Vacaciones adicionales pendientes: Total días de adicionales de vacaciones pendientes para el periodo actual que le corresponden al interesado.

Asuntos propios disponibles: Total días de asuntos propios disponibles para el periodo actual que le corresponden al interesado.
Asuntos propios autorizados: Total días de asuntos propios autorizados al interesado para el periodo actual, que se detallan en el listado.
Asuntos propios pendientes: Total días de asuntos propios pendientes para el periodo actual que le corresponden al interesado.
Asuntos particulares adicionales disponibles: Total días de adicionales de asuntos particulares disponibles para el periodo actual que le corresponden al interesado.
Asuntos particulares adicionales autorizados: Total días de adicionales de asuntos particulares autorizados al interesado para el periodo actual, que se detallan en el listado.
Asuntos particulares adicionales pendientes: Total días de adicionales de asuntos particulares pendientes para el periodo actual que le corresponden al interesado.

Motivo: Descripción de la clase de permiso. [No editable]
Fecha inicio: Fecha de inicio del permiso o licencia solicitado. [No editable]
Fecha fin: Fecha de fin del permiso o licencia solicitado. [No editable]
Días: Nº de días del permiso o licencia solicitado. [No editable]

Pantalla de listado periodo anterior:

En la *cabecera* se mostrará el total de días de los permisos o licencias de vacaciones y asuntos propios, y sus respectivos adicionales, disponibles, autorizados y pendientes en el periodo anterior para el usuario interesado:

Vacaciones disponibles: Total días de vacaciones disponibles para el periodo anterior que le corresponden al interesado.
Vacaciones autorizadas: Total días de vacaciones autorizadas al interesado para el periodo anterior, que se detallan en el listado.
Vacaciones pendientes: Total días de vacaciones pendientes para el periodo anterior que le corresponden al interesado.
Vacaciones adicionales disponibles: Total días de adicionales de vacaciones disponibles para el periodo anterior que le corresponden al interesado.
Vacaciones adicionales autorizadas: Total días de adicionales de vacaciones autorizadas al interesado para el periodo anterior, que se detallan en el listado.
Vacaciones adicionales pendientes: Total días de adicionales de vacaciones pendientes para el periodo anterior que le corresponden al interesado.

Asuntos propios disponibles: Total días de asuntos propios disponibles para el periodo anterior que le corresponden al interesado.
Asuntos propios autorizados: Total días de asuntos propios autorizados al interesado para el periodo anterior, que se detallan en el listado.
Asuntos propios pendientes: Total días de asuntos propios pendientes para el periodo anterior que le corresponden al interesado.
Asuntos particulares adicionales disponibles: Total días de adicionales de asuntos particulares disponibles para el periodo anterior que le corresponden al interesado.
Asuntos particulares adicionales autorizados: Total días de adicionales de asuntos particulares autorizados al interesado para el periodo anterior, que se detallan en el listado.
Asuntos particulares adicionales pendientes: Total días de adicionales de asuntos particulares pendientes para el periodo anterior que le corresponden al interesado.

Motivo: Descripción de la clase de permiso. [No editable]
Fecha inicio: Fecha de inicio del permiso o licencia solicitado. [No editable]
Fecha fin: Fecha de fin del permiso o licencia solicitado. [No editable]
Días: Nº de días del permiso o licencia solicitado. [No editable]

3.11. Estado de mis solicitudes

Finalidad

En esta pantalla se mostrarán todas las solicitudes de permisos y licencias del usuario interesado para un determinado periodo. De esta forma, se podrá conocer en todo momento el estado en que se encuentran sus solicitudes.

Los posibles estados en los que se puede encontrar una solicitud son los siguientes: Pendiente validar jefe servicio / Afecta a las necesidades del servicio / Pendiente autorización / Autorizada / No autorizada.

En el caso de tener una solicitud en estado “Afecta a las necesidades del servicio”, y la persona interesada no estar de acuerdo con dicha decisión, se podrá acceder al detalle de dicha solicitud y mediante la utilización del botón “En desacuerdo”, podrá indicar que su solicitud sea revisada por el responsable de personal, quien tomará la decisión de autorizarla o no.

Funcionamiento

Perfil de acceso: Usuario de la CIT, tramitador, revisor, responsable y responsable de personal.

Forma de acceso: Desde el menú de aplicación *[Consultas][Estado de mis solicitudes]*.

La *pantalla de búsqueda* (*Figura 4.3.1*) sirve para obtener (botón 

) las solicitudes de permisos y licencias solicitadas previamente, según el el criterio de búsqueda seleccionado. Podremos buscar por *periodo,* por *fecha inicio,* por *fecha fin,* por *motivo,* por *solicitudes pendientes de firma, firmadas o todas,* o por cualquier combinación de éstas. Además, podremos limpiar campos de búsqueda.

La *pantalla listado* *(Figura 4.3.2)* nos muestra el listado del estado de mis solicitudes atendiendo a nuestro criterio de búsqueda seleccionado. En caso de haber seleccionado un motivo en la *pantalla de búsqueda*, se mostrará en la cabecera el total de días hábiles y naturales solicitados en el periodo. Las operaciones permitidas desde esta pantalla serán las siguientes: ver detalles de la solicitud seleccionada y refrescar la lista de solicitudes.

La *pantalla de detalle* (*Figura 4.3.3*), muestra el detalle de la solicitud seleccionada. Se accede pulsando sobre el botón  de la *pantalla de listado*.

Pantallas del programa

Pantalla de búsqueda:

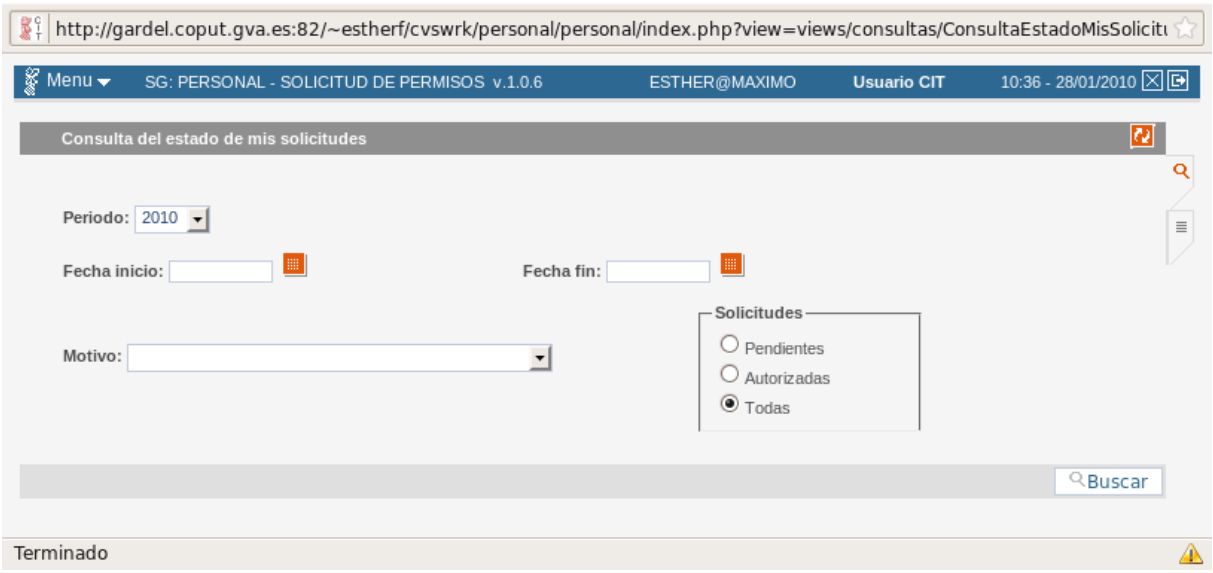


Figura 4.3.1: Pantalla de búsqueda del estado de mis solicitudes

Pantalla listado:

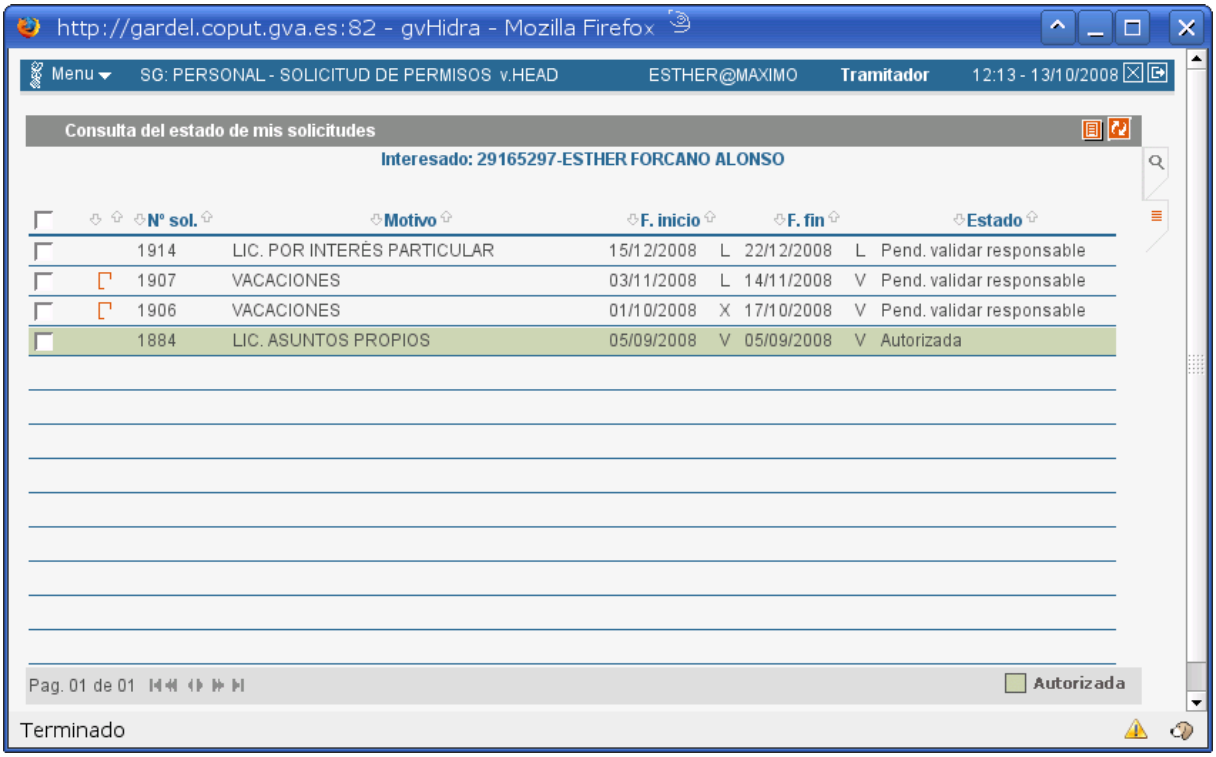


Figura 4.3.2: Pantalla listado tras consultar el estado de mis solicitudes

Pantalla detalle:

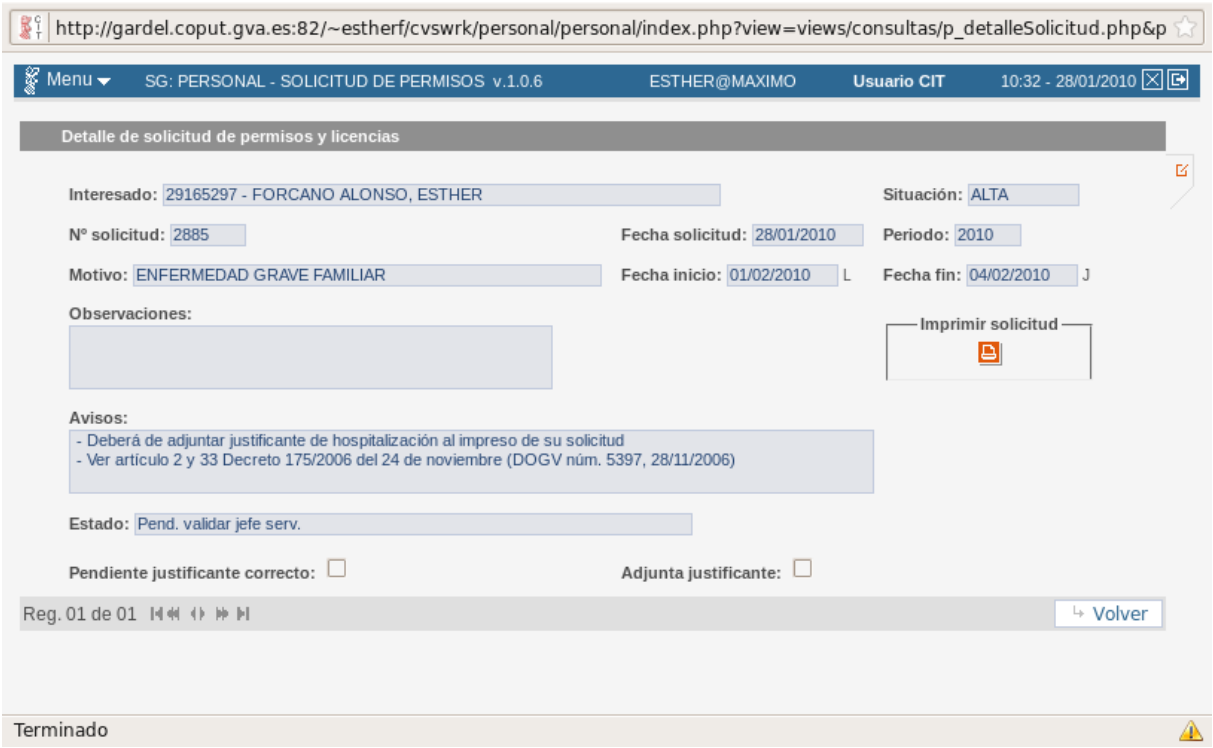


Figura 4.3.3: Pantalla detalle de solicitud de permisos y licencias

Campos y botones de las pantallas

Pantalla de búsqueda:

Periodo: Lista desplegable de periodos a buscar: el periodo anterior, el actual y el siguiente. Por defecto, el valor es el año actual. [Obligatorio][Seleccionable]
Fecha inicio: Fecha de inicio de las solicitudes de permisos y licencias. [Opcional][Editable]
Fecha fin: Fecha de fin de las solicitudes de permisos y licencias. [Opcional][Editable]
Día de la semana: Indicadores del día de la semana correspondientes a las fechas de inicio y de fin. [No editable]
Motivo: Lista desplegable de clases de permisos y licencias solicitables. [Opcional][Seleccionable]
Solicitudes:

- *Pendientes:* Busca las solicitudes que están pendientes de firma del Subsecretario. [Opcional][Seleccionable]
- *Autorizadas:* Busca las solicitudes que están firmadas por el Subsecretario. [Opcional][Seleccionable]
- *Todas:* Busca todas las solicitudes. Es el valor por defecto. [Opcional][Seleccionable]

 *Limpiar campos*

 *Calendario* (*Figura 4.3.4*)

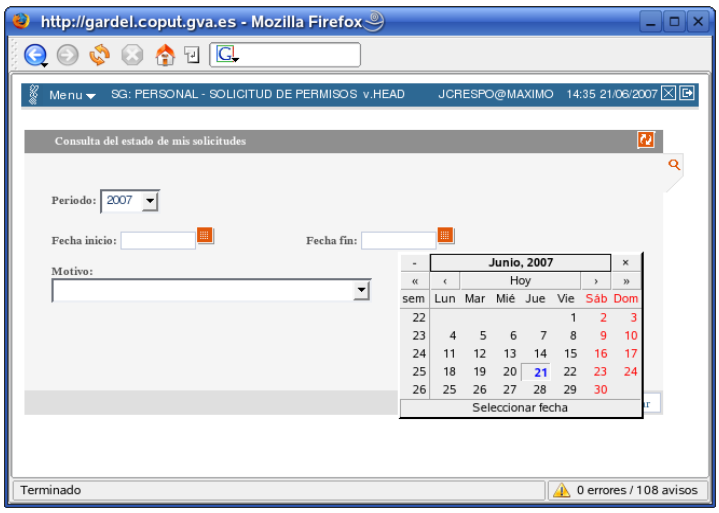


Figura 4.3.4: ToolTip calendario

Pantalla listado:

En caso de haber seleccionado un motivo en la pantalla de búsqueda, se mostrará en la *cabecera* el total de días hábiles y naturales solicitados en el periodo:

Total días hábiles solicitados [periodo]: Total días hábiles solicitados por el interesado en el periodo para un motivo determinado, que se detallan en el listado.
Total días naturales solicitados [periodo]: Total días hábiles solicitados por el interesado en el periodo para un motivo determinado, que se detallan en el listado.

Nº sol.: Nº de la solicitud de permisos y licencias. [No editable]
Motivo: Descripción de la clase de permiso. [No editable]
Fecha inicio: Fecha de inicio de la solicitud de permisos y licencias. [No editable]
Fecha fin: Fecha de fin de la solicitud de permisos y licencias. [No editable]
Estado: Estado en que se encuentra la solicitud de permisos y licencias: Pendiente validar jefe servicio / Afecta a las necesidades del servicio / Pendiente autorización / Autorizada / No autorizada. [No editable]

Checkbox

Indica que la solicitud tiene avisos para el Servicio de personal.

Sólo aparecerá cuando sea una solicitud de vacaciones pendiente de completar, es decir, no se hayan solicitado todas las vacaciones en su totalidad, para un determinado periodo.

Sólo aparecerá cuando sea una solicitud de vacaciones pendiente de completar y, además, tenga avisos para el Servicio de personal.

Autorizada El *fondo verde* en la solicitud de permisos y licencias indica que está autorizada.

Ver Detalles (Figura 4.3.3)

Refrescar la lista de solicitudes

Pantalla de detalle:

Interesado: Persona interesada de la solicitud de permisos o licencias. [No editable]
Situación: Indica la situación (Alta o anulación) de la solicitud de permisos o licencias. [No editable]
Nº solicitud: Nº identificador de la solicitud de permisos o licencias. [No editable]
Solicitud cambiada: Nº identificador de la solicitud de permisos o licencias a la que sustituye, para los motivos de “Vacaciones” y “Días adicionales vacaciones”. [No editable]
Fecha solicitud: Fecha en que se realizó la solicitud de permisos o licencias. [No editable]

Periodo: Indica el año al que corresponde la solicitud de permisos o licencias.[No editable]
Motivo: Descripción de la clase de permisos o licencias. [No editable]
Fecha inicio: Fecha de inicio de la solicitud de permisos o licencias.[No editable]
Fecha fin: Fecha de fin de la solicitud de permisos o licencias.[No editable]
Día de la semana: Indicadores del día de la semana correspondientes a las fechas de inicio y de fin. [No editable]
Observaciones: Comentarios relacionados con la solicitud de permisos o licencias.[No editable]
Avisos: Expresamos los avisos que se mostraron al usuario cuando dio de alta su solicitud.[No editable]
Estado: Indica el estado en el que se encuentra la solicitud: Pendiente validar jefe servicio / Afecta a las necesidades del servicio / Pendiente autorización / Autorizada / No autorizada. [No editable]
Pendiente justificante correcto: Indica si el interesado ha presentado un nuevo justificante requerido. [No editable]
Adjunta justificante: Indica si el interesado ha presentado el justificante requerido. [No editable]
Pendiente justificante correcto: Indica si el interesado ha presentado un nuevo justificante correcto requerido. [No editable]

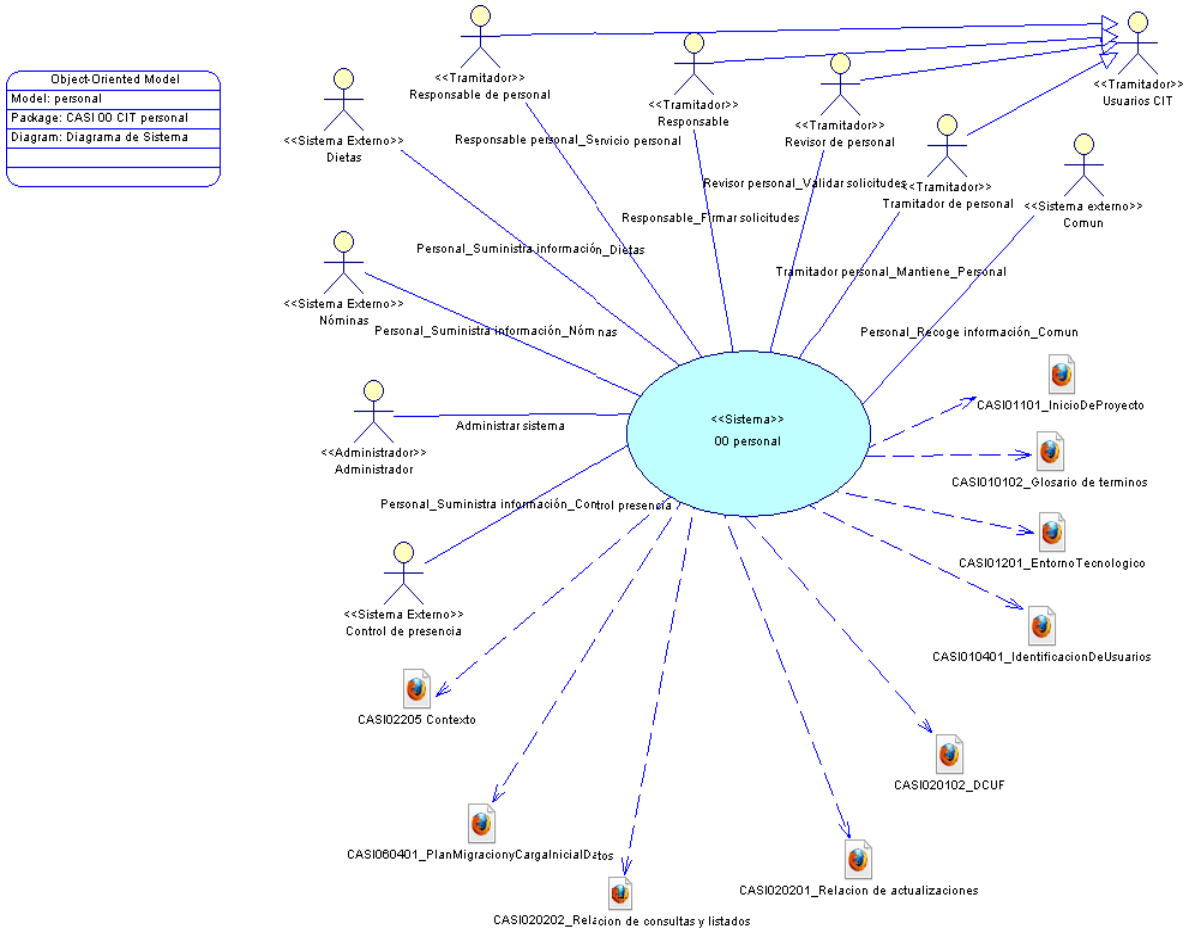
Sólo visible cuando el estado de la solicitud es “Afecta a las necesidades del servicio”, en cuyo caso, se podrá utilizar este botón para indicar que la solicitud debe ser revisada por el responsable de personal, quien tomará la decisión de autorizarla o no.


Visible cuando la solicitud de permisos y licencias requiere adjuntar un justificante, en cuyo caso, se podrá imprimir la solicitud para, posteriormente, graparla al justificante y entregarlo en el Servicio de personal. También, será visible cuando sea una solicitud tramitada por la Dirección General de Administraciones Autonómicas, en cuyo caso, se podrá imprimir la solicitud para, posteriormente, graparla a un impreso que previamente se habrá rellenado y entregarlo en el Servicio de personal.

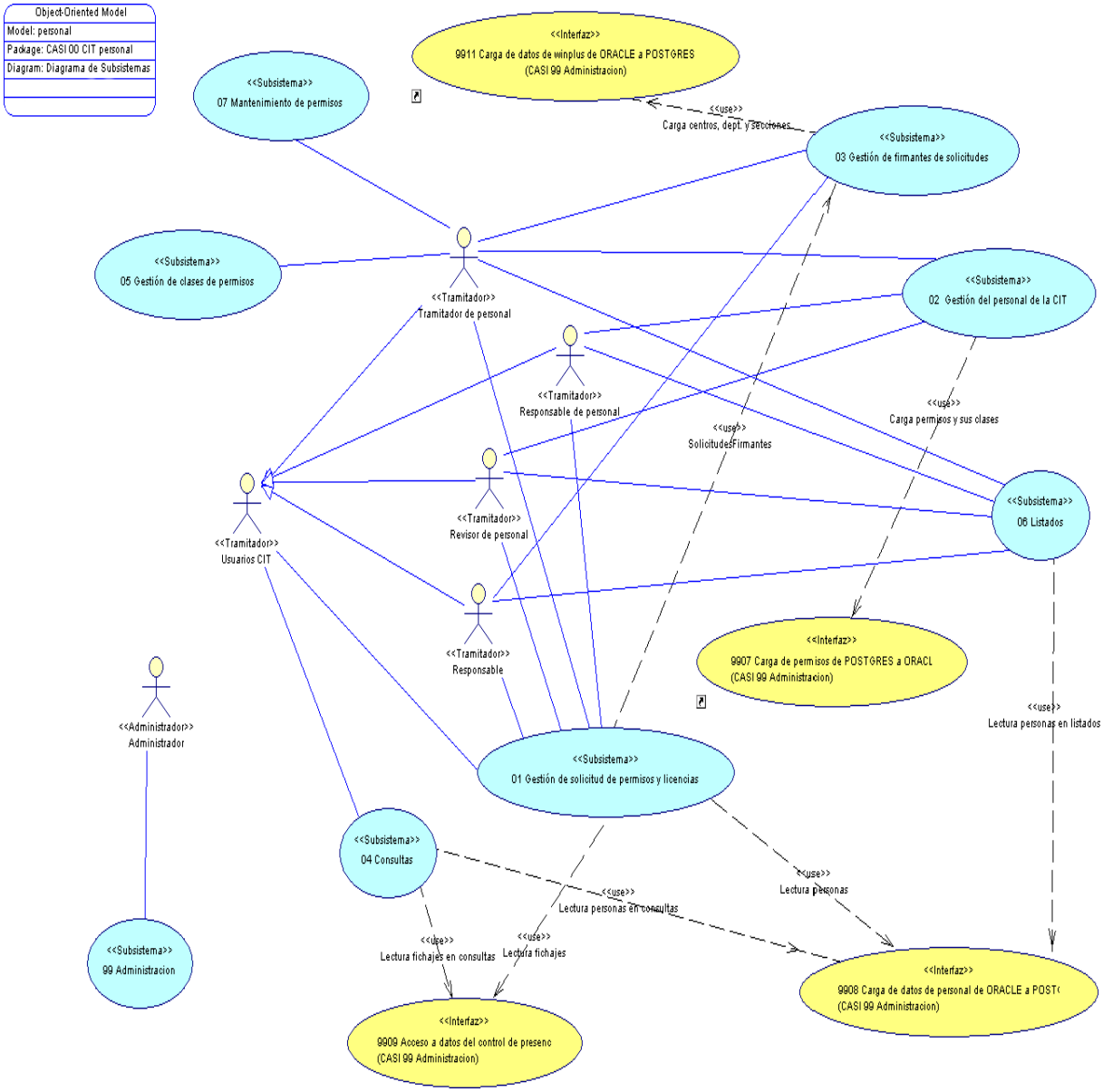
4. ANÁLISIS

4.1. Casos de uso

4.1.1. Nivel 0. Digrama de sistema.



4.1.2. Diagrama de Subsistemas.



4.1.3. Descripción casos de uso

Casos de uso – ALTA DE PERMISOS Y SOLICITUDES

Comentario:

El usuario de la CIT da de alta sus solicitudes de permisos y licencias, que pasan por diferentes estados hasta que son autorizadas o no. Ésta pasa a estado "pendiente validar responsable", y es su responsable quien la valida, autorizándola o no. En caso de autorización y teniendo la solicitud mensajes de avisos y/o necesidad de adjuntar justificante, o bien de no autorización y no conformidad del solicitante, pasan a la validación del tramitador de personal. En caso contrario, pasará directamente a la firma del responsable de personal. El tramitador de personal marcará la solicitude que le llega como válida o no, pasando entonces a la firma del responsable de personal. Si personal la autoriza, se graba en la base de datos.

Acciones:

-1.-Una vez el usuario introduce los datos en la pantalla de alta de solicitud, aparece información relacionada con la misma, es decir, días solicitados,mensajes de avisos y/o necesidad de adjuntar justificante...
-1.1. Si necesita adjuntar justificante, debe imprimir la solicitud para graparla al justificante y entregarla a personal.
-2. El usuario de la CIT acepta la solicitud pasando a estado "pendiente validar responsable", y será su responsable quien la valide, autorizándola o no.
-2.1. Se envía correo a responsable de la solicitud.
-3. En caso de autorización,
-3.1 si la solicitud tiene mensajes de avisos y/o necesidad de adjuntar justificante, pasa a estado " pendiente autorizar personal".
-3.1.1 Si personal la autoriza, se graba en la base de datos y pasa a estado "autorizada personal".
-3.1.2 Si no la autoriza, pasa a estado "no autorizada personal".
-3.2 si la solicitud no llega a personal por no tener avisos, y es autorizada por el responsable, se graba en la base de datos y pasa estado "autorizada responsable"
-4. En caso de no autorización, pasa a estado "no autorizada responsable".
-4.1 si el usuario está en desacuerdo, pasará a estado "pendiente autorizar personal", y será personal quien la valide, autorizándola o no.
-5. Se envía un correo al usuario para notificárselo.

Precondiciones:

-El usuario de la CIT que solicita el permiso debe estar dado de alta en la base de datos de personal, es decir, debe ser funcionario o laboral o eventual o interino, por lo tanto, debe existir en algún centro de trabajo. Además, debe tener un nº de registro personal, una provincia y un municipio. (f_validaUsuario)
-El usuario de la CIT que solicita el permiso debe estar ocupando una plaza.
-El usuario de la CIT debe tener permitido el acceso a la nueva aplicación de personal (esto se deberá de realizar por el administrador de base de datos para los nuevos usuarios, y con una carga inicial para los ya existentes).
-Los tramitadores de personal deben haber introducido correctamente todas las clases de permisos, indicando si deben o no adjuntar justificante, así como los el nº de días máximo de cada permiso y el de asuntos propios para los municipios que corresponda.
-Los tramitadores de personal deben haber dado de alta todos los responsables de firma (suplentes o no) correspondientes a cada grupo de firma.
-Las reglas de validación de permisos se deben haber dado de alta por los administradores informáticos.
-Los diferentes estados de solicitud se deben haber dado de alta por los administradores informáticos.

Postcondiciones:

-Si hay errores, no se grabará la solicitud en la base de datos.
- Si la solicitud lleva o no asociada "avisos", se grabará la solicitud en la base de datos.
- Si se requiere justificante, se imprimirá la solicitud para graparla al justificante.
- La solicitud pasará a estado "Pendiente validar responsable", salvo que el tipo sea de vacaciones, adicionales y no se hayan solicitado todos los días disponibles, en cuyo caso aparecerán como pendientes de completar
- Se enviarán por correo electrónico al responsable las solicitudes confirmadas por el usuario.
-Si se ha pedido un permiso, este se grabará también en la base de datos de la aplicación de personal actual.

Reglas de negocio:

_QuienSolicita

Sólo podrán solicitar permisos y licencias los funcionarios, interinos, eventuales y laborales, es decir, el personal que está dado de alta en personal.

_ErrorAvisos

Si al intentar dar de alta una solicitud aparecen mensajes de error, éstos impedirán la introducción del permiso. Sin embargo, si aparecen avisos, éstos indican situaciones anómalas pero no indican que el permiso no es correcto, aunque tienen que ser revisados y validados por el servicio de personal y aceptados o no, según el caso.

_ImprimirSolicJustif

Todas las solicitudes que requieran adjuntar justificante, deberán incluir un impreso de la solicitud para que se sepa a qué solicitud pertenece el justificante.
Para ello se imprimirá la solicitud, se grapará junto al justificante y se entregará a personal.

_ClasesPermisos

La lista de motivos de solicitud de permisos y licencias que pueden dar de alta todos los usuarios de la CIT, son los utilizados en la actuales hojas de “SOLICITUD DE PERMISOS Y LICENCIAS”, es decir, las que sean solicitables (Clases de permisos.solicitable='S'). Además, también dará de alta el tramitador del servicio de personal aquéllos permisos no solicitados desde las actuales hojas, es decir, los que sean NO solicitables (Clases de permisos.solicitable=N').

_MantenimientoClasesPermisos

La gestión de clases de permisos se dejará de realizar en la aplicación actual de personal (Oracle), y pasará a mantenerse en la nueva aplicación (Postgres), en la opción "Mantenimiento de clases de permisos"
En este nuevo mantenimiento de clases de permisos se podrán:
- Identificar los permisos que requieran justificante activando o no el campo “Justificante”
- Indicar el máximo y mínimo de días que se pueden tomar por clase de permiso, con los campos "Max. días" y "Min. días"
- Indicar si el máximo y mínimo de días que se pueden tomar son naturales o hábiles.
- Clasificar las clases de permisos en "Permiso", "Reducción", "Licencia" y "Horario" cambiando el campo correspondiente.
Además, sólo en el caso de "Licencia asuntos propios", se indicará el máximo de días por municipio.

_Periodo

En el alta: el período es el año actual. También será posible elegir el año anterior si la fecha actual es hasta el 1 de febrero, y el año siguiente durante el mes de diciembre.
En las consultas: el período es el año actual, por defecto. También será posible elegir el año anterior o el siguiente.

9B_0101_RestriccionesFechas

* Sólo para los motivos ‘IT’, 'MAD' y reducciones ('RJM', 'RJE', 'RJP') se permite no introducir la fecha final.
* No se podrá dar de alta una solicitud en la que coincida alguna fecha con otra solicitud, salvo que la otra solicitud sea de reducciones, y que además no sean las dos del mismo tipo de reducción.
* Se podrá solapar una 'IT' con vacaciones.

_CamposReducciones

* para una ‘IT’, se indicará también la causa,
* para 'RJM' se indicará el tipo de minusvalía ,
* para 'RJP' se indicará el porcentaje de reducción y
* para 'RJG', 'RJL', 'AJL' y 'FPO', se indicará la fecha de nacimiento del menor.

_PendValidarResp

Una vez introducidos los datos requeridos por la pantalla de alta de solicitudes y aceptados los datos validados, la solicitud ya está para ser procesada por el responsable quien la autorizará o no.

_CorreoResponsable

Se enviarán por correo electrónico al responsable las solicitudes de alta o anulación confirmadas por el usuario.

_SolicSinConAvisos

Todas aquellas altas de permisos SIN mensajes de avisos, una vez validadas por el responsable como autorizadas, pasarán directamente al responsable de personal, sin pasar por la validación del tramitador en personal.
Sólo pasarán por la validación del tramitador del servicio de personal aquellas solicitudes CON mensajes de avisos o que requieran justificante.

_ValidacionesVAC

Validaciones de vacaciones:
* El máximo de días de vacaciones viene determinado por los días trabajados en el año. Si se superan, se generará un aviso para personal.
* No se permitirá solicitar permisos de vacaciones de menos de 7 días naturales, en cuyo caso se mostrará una prohibición.
* Las solicitudes de permisos de vacaciones se pasarán a firma aunque no estén pedidos todos los días de vacaciones (y adicionales) disponibles para el periodo.
* Sólo se podrá coger un mes natural de vacaciones (del dia 'd' del mes al día 'd' del mes siguiente -1) si se piden en un único tramo y, además, la persona no tiene días adicionales.
* El nº de días de vacaciones y asuntos propios para aquéllos que no hayan trabajado todo el año será proporcional al nº de días trabajados, es decir, (nº días asuntos propios o vacaciones*días trabajados)/365.
* Si se ha solicitado licencia por interés particular igual o superior a 1 mes natural, deberá descontarse de la vacación anual el tiempo proporcional de la licencia por interés particular disfrutada (2 días y medio).
* El máximo de días adicionales viene determinado por la fecha de antigüedad de cada persona y por la información que envía Función Pública a primeros de año.

_ValidacionesLIPyVAC

Validaciones de "licencias por interés particular" y "vacaciones" cuando se supera o iguala el mes natural:
* Comprobar si procede devolución de retribuciones. Se calcula la parte proporcional de días de vacaciones (incluyendo adicionales) que no puede disfrutar porque corresponden a "licencias por interés particular".

_CorreoUsuario

Se enviarán por correo electrónico al usuario las confirmaciones de firma autorizada o no por su responsable de las solicitudes de alta o anulación, así como las anulaciones de firma, si las hubiera. Además, las notificaciones de las solicitudes tras la firma del Secretario.

_PendCompletarInteresado

Una vez introducidos los datos requeridos por la pantalla de alta de solicitudes y aceptados los datos validados, si la solicitud es de vacaciones o días adicionales, y no se solicitan todos los días disponibles, la solicitud si será procesada por el responsable aunque quede pendiente de solicitar el resto de días de vacaciones y/o adicionales restantes.

_ImprimirSolicLIP

Todas las solicitudes de Licencias por interés particular, deberán incluir un impreso de la solicitud, que será el mismo que se utiliza cuando se requiera justificante, para adjuntar al impreso que remiten a Función Pública.
Para ello se imprimirá la solicitud y se entregará a personal.

_ValidacionesADI

Validaciones de días adicionales:
* Si el interesado no dispone de días adicionales no podrá solicitar ningún día adicional

Casos de uso – ANULAR SOLICITUDES

Comentario:

El usuario de la CIT podrá anular las altas de sus solicitudes de permisos y licencias solicitadas previamente, que pasarán por diferentes estados hasta que sean autorizadas o no. Una vez el usuario solicita una anulación de un alta, ésta pasará a estado pendiente, y será su responsable quien la valide, autorizándola o no.

Si no pasa a personal, porque el alta de solicitud sólo tiene firma de responsable:
* si se autoriza la anulación, se borrará el alta de la solicitud de la base de datos,
* si no se autoriza la anulación, el alta de la solicitud seguirá su curso.

Si pasa a personal, porque el alta de solicitud tiene firma de responsable y de personal:
* si se autoriza la anulación, se borrará el alta de la solicitud de la base de datos,
* si no se autoriza la anulación, el alta de la solicitud seguirá su curso.

Acciones:

-1. Una vez el usuario decide qué solicitudes anular de entre las que se le muestran en pantalla, es decir, aquellas solicitudes que el usuario haya dado de alta previamente, siempre y cuando cumplan los requisitos de la regla "9A_0102_QueAnular":

-1.1 selecciona la/s solicitud/es y confirma la anulación pasando a estado "pendiente validar responsable", y será su responsable quien la valide, autorizándola o no,

-1.2 se envía correo a responsable de la anulación.

-2. Si la anulación no pasa a personal,

-2.1 si se autoriza la anulación, se borrará el alta de la solicitud de la base de datos, y la anulación pasa a estado "autorizada responsable",

-2.2 si no se autoriza la anulación, el alta de la solicitud seguirá su curso, y la anulación pasa a estado "no autorizada responsable".

-3. Si la anulación pasa por personal,

-3.1 si se autoriza la anulación, se borrará el alta de la solicitud de la base de datos, y la anulación pasa a estado "autorizada personal",

-3.2 si no se autoriza la anulación, el alta de la solicitud seguirá su curso, y la anulación pasa a estado "no autorizada personal"

-4. Se envía un correo al usuario para notificárselo.

Precondiciones:

-Tiene que haber altas de solicitudes firmadas por el responsable de personal.

Postcondiciones:

-Si hay errores, no se anulará la solicitud.
-La solicitud de anulación pasará a estado "Pendiente validar responsable".
-Si la anulación se autoriza, se borrará el alta de la solicitud de la base de datos.
-Se enviarán por correo electrónico al responsable las solicitudes de anulación confirmadas por el usuario, y a personal las autorizadas por el responsable sin haber pasado el alta por personal.

Reglas de negocio:

_CorreoResponsable

Se enviarán por correo electrónico al responsable las solicitudes de alta o anulación confirmadas por el usuario.

_QueAnular

Sólo se podrán anular las solicitdes de permisos y licencias que el usuario haya dado de alta previamente, siempre y cuando el alta tenga firma de responsable con estado PT, PN o PP, o de responsable y personal con estado AP.

_NoPersonal

Si el responsable no autoriza una anulación, no llegará a personal

_ComprobarFichadas

Cualquier anulación cuyo período sea parcialmente o totalmente anterior a la fecha del dia, se comprobará con las fichadas realizadas.

Casos de uso – BORRAR SOLICITUDES

Comentario:

El usuario de la CIT podrá borrar sus solicitudes de permisos y licencias que haya dado de alta previamente, tanto altas como anulaciones siempre y cuando sean:
* altas no autorizadas
* altas sin firma de responsable
* anulaciones no autorizadas
* anulaciones sin firma del responsable

Acciones:

1. Una vez el usuario decide qué solicitudes borrar de entre las que se le muestran en pantalla, es decir, aquellas solicitudes que el usuario haya dado de alta previamente, siempre y cuando cumplan los requisitos de la regla "_QueBorrar":
1.1 selecciona la/s solicitud/es y confirma el borrado de la solicitud de la base de datos.

Precondiciones:

-Tiene que haber solicitudes que poder borrar.

Postcondiciones:

- Si hay errores, no se borrará la solicitud.
- Si la solicitud a borrar es un alta, se borrará de la base de datos.
- Si la solicitud a borrar es una anulación, se cambiará su situación a la de alta de solicitud.

Reglas de negocio:

_QueBorrar

Sólo se podrán borrar las solicitudes de permisos y licencias que el usuario haya dado de alta previamente, tanto altas como anulaciones siempre y cuando sean:

* altas o anulaciones sin firma de responsable,
* altas pendientes de completar la solicitud de vacaciones

Para las altas o anulaciones no autorizadas, no permitimos el borrado porque se debería imprimir la resolución negativa correspondiente. Aunque se compruebe si la resolución ha sido impresa, no podemos borrar porque puede haberse impreso la resolución y por fallos de impresora, sistema, etc. sea necesario volver a imprimirla.

Casos de uso – CONSULTAR CONTROL DE PRESENCIA

Comentario:

Se mostrará la información de fichajes para el mes actual, incluyendo los fichajes del dia, y también se podrá mostrar la información del mes anterior.
Además, los tramitadores, revisores y responsables de personal podrán consultar los fichajes de cualquier usuario de la CIT.

Acciones:

1- El tramitador, revisor o responsable de personal eligen el usuario de la CITdel que quieren ver las fichadas.
2- El usuario elige si quiere ver las fichadas del mes actual o del anterior.
3- El usuario visualiza las fichadas del mes seleccionado previamente.

Reglas de negocio:

_AccesoControlPresencia

La consulta de control de presencia será accesible desde el menú de aplicaciones que se muestra tras la validación de usuario, para todo el personal. Además, esta misma consulta aparecerá como una opción de menú dentro de la aplicación de personal, a la que sólo accederá el personal funcionario, interino, eventual y laboral.

Casos de uso – CONSULTAR DÍAS DE LICENCIA POR ASUNTOS PROPIOS Y VACACIONES

Comentario:

Se mostrará la información de días de licencia de asuntos propios y de vacaciones autorizados para el periodo actual y el anterior.
Los días se contarán como hábiles.

Acciones:

1- El usuario visualiza en pantalla los días de licencia por asuntos propios, vacaciones y adicionales autorizados del periodo actual y anterior.

Casos de uso – CONSULTAR ESTADO DE MIS SOLICITUDES

Comentario:

Se podrán consultar las solicitudes de permisos de una persona.
Los parámetros de búsqueda serán los siguientes:
- Periodo
- Fecha inicio
- Fecha fin
- Motivo
- Solicitudes: Pendientes de Firma(PR,PP)/Firmadas/Todas

A esta consulta podrán acceder todos los usuarios que pueden hacer solicitudes de permisos y licencias

Acciones:

1- El usuario introduce los criterios de consulta, y tras pulsar en el botón Buscar, aparecerá una lista de sus solicitudes que cumplen el criterio de búsqueda.
2- El usuario podrá seleccionar una de las solicitudes para consultar el detalle de la solicitud

4.2. Diagramas de secuencias

Diagrama de secuencia – ALTA DE PERMISOS Y SOLICITUDES

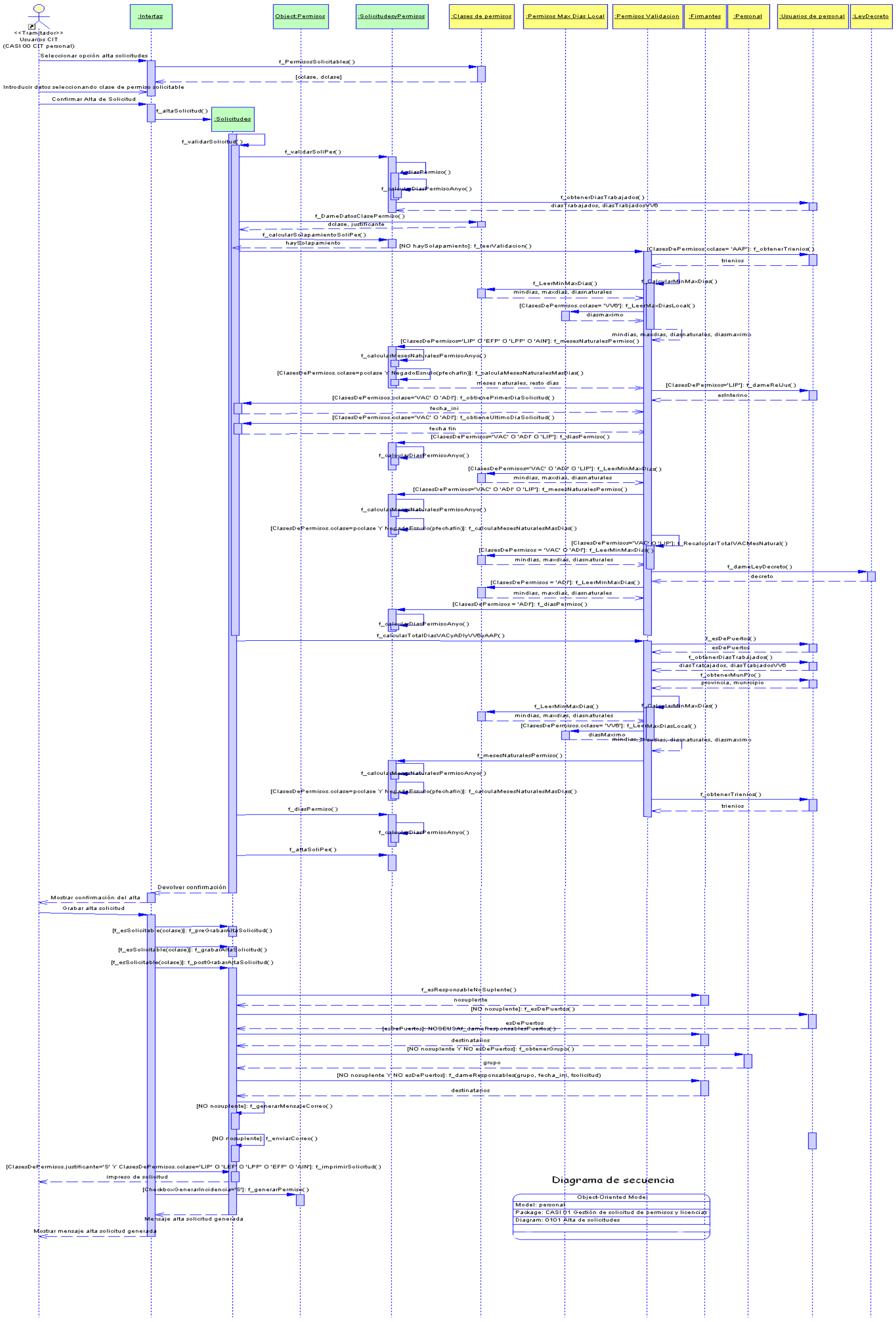


Diagrama de secuencia – ANULACIÓN DE SOLICITUDES

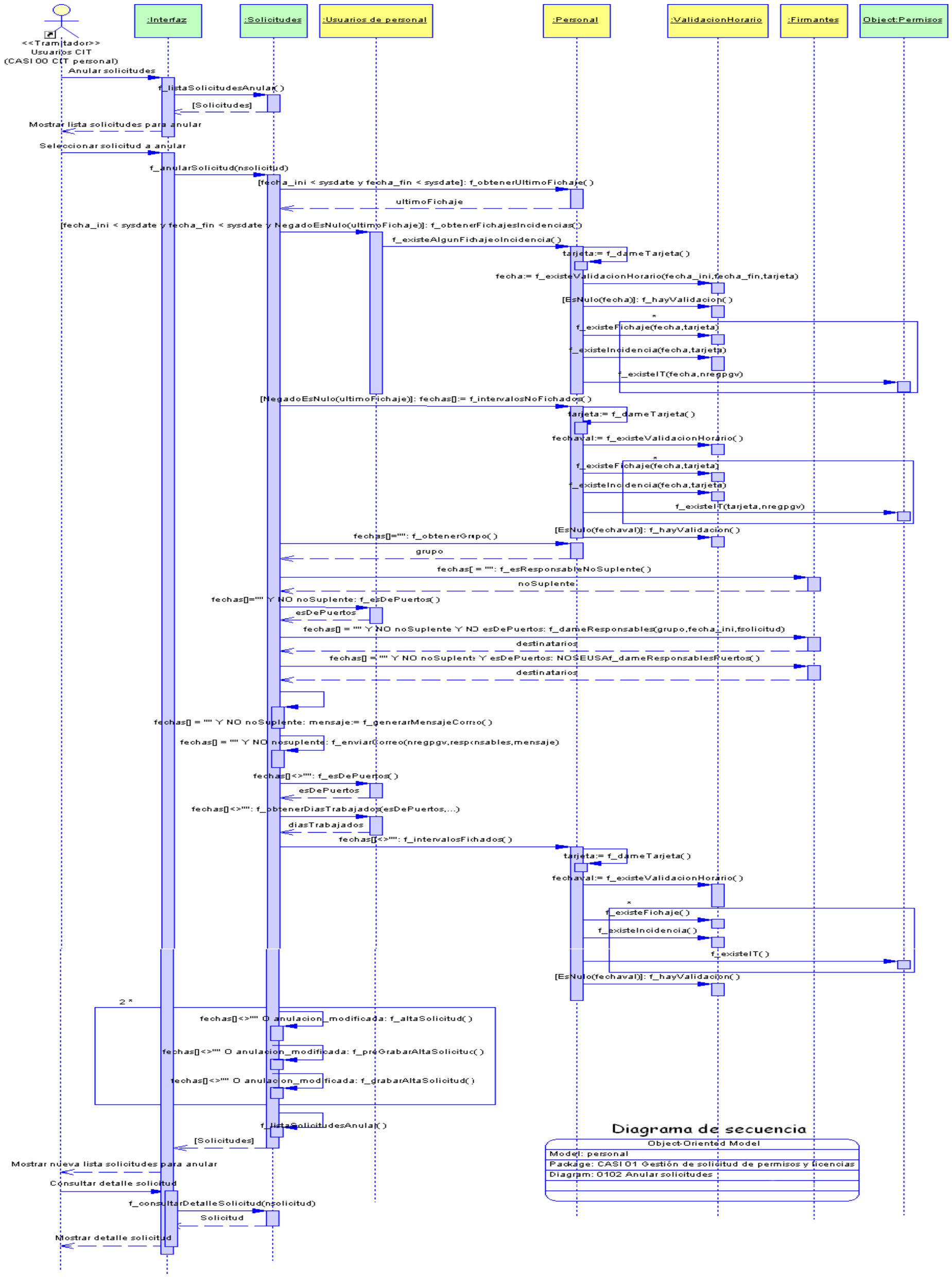


Diagrama de secuencia – BORRADO DE SOLICITUDES

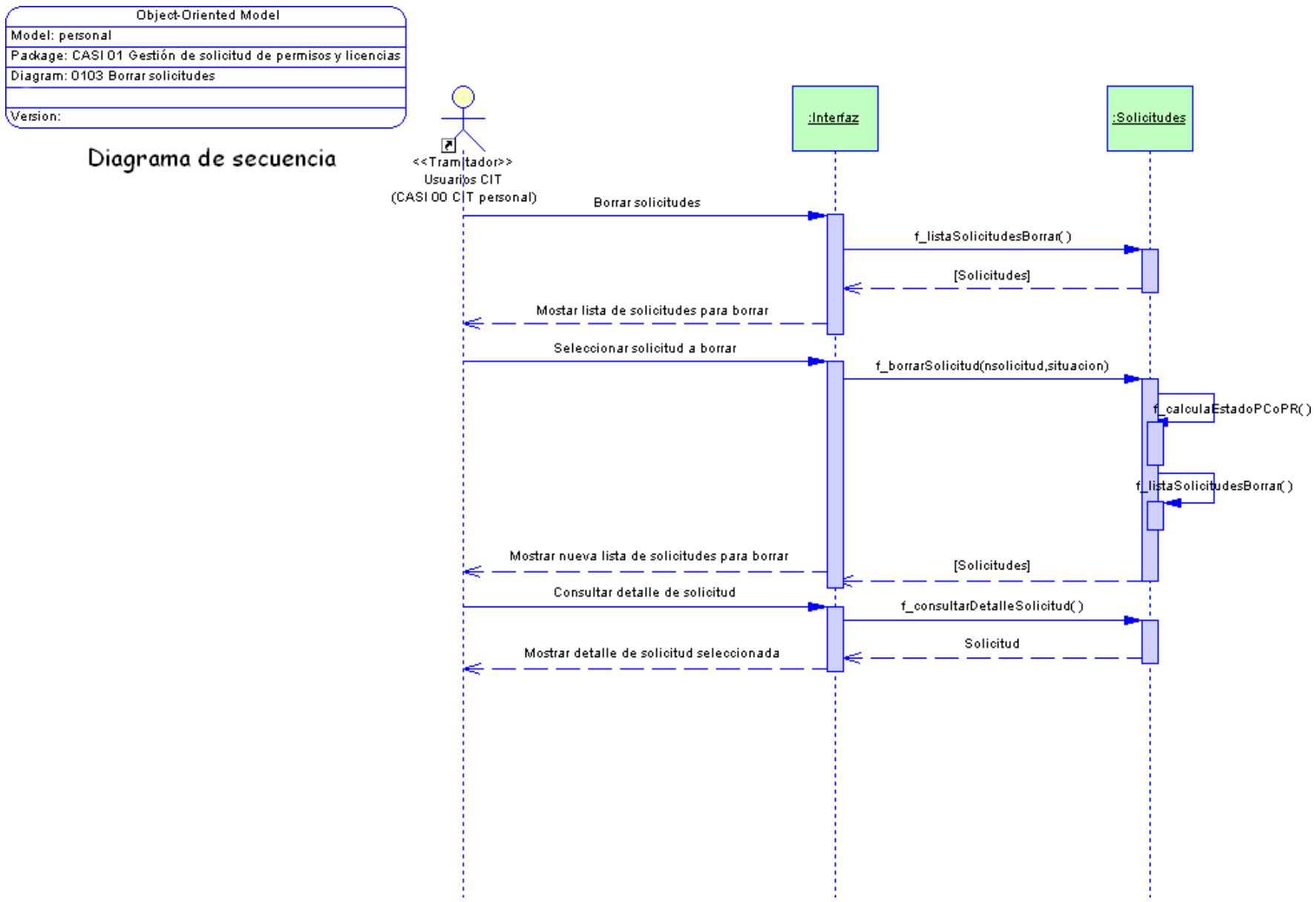


Diagrama de secuencia –CONSULTAR CONTROL DE PRESENCIA

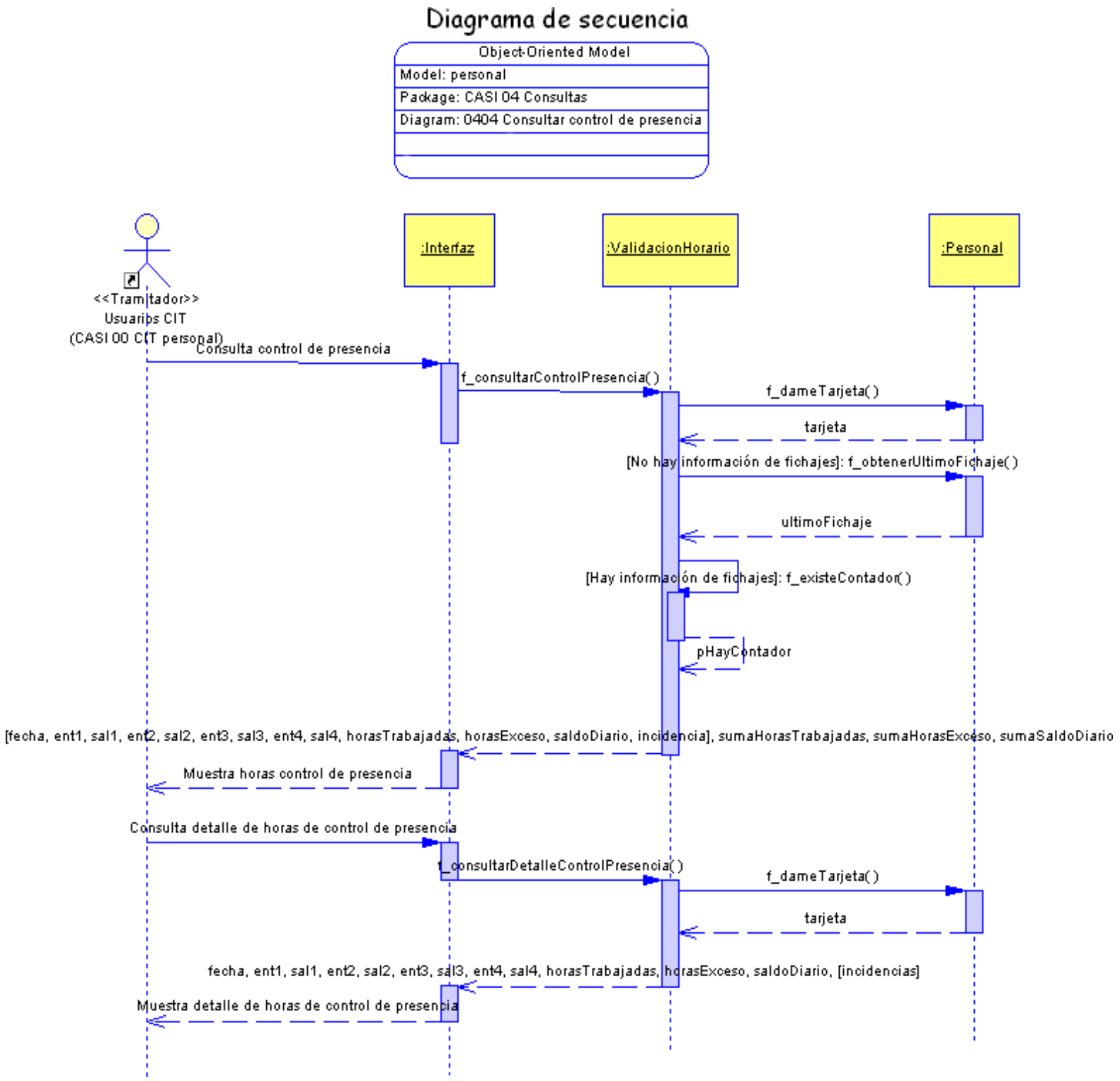


Diagrama de secuencia –CONSULTAR DÍAS DE LICENCIA POR ASUNTOS PROPIOS Y VACACIONES

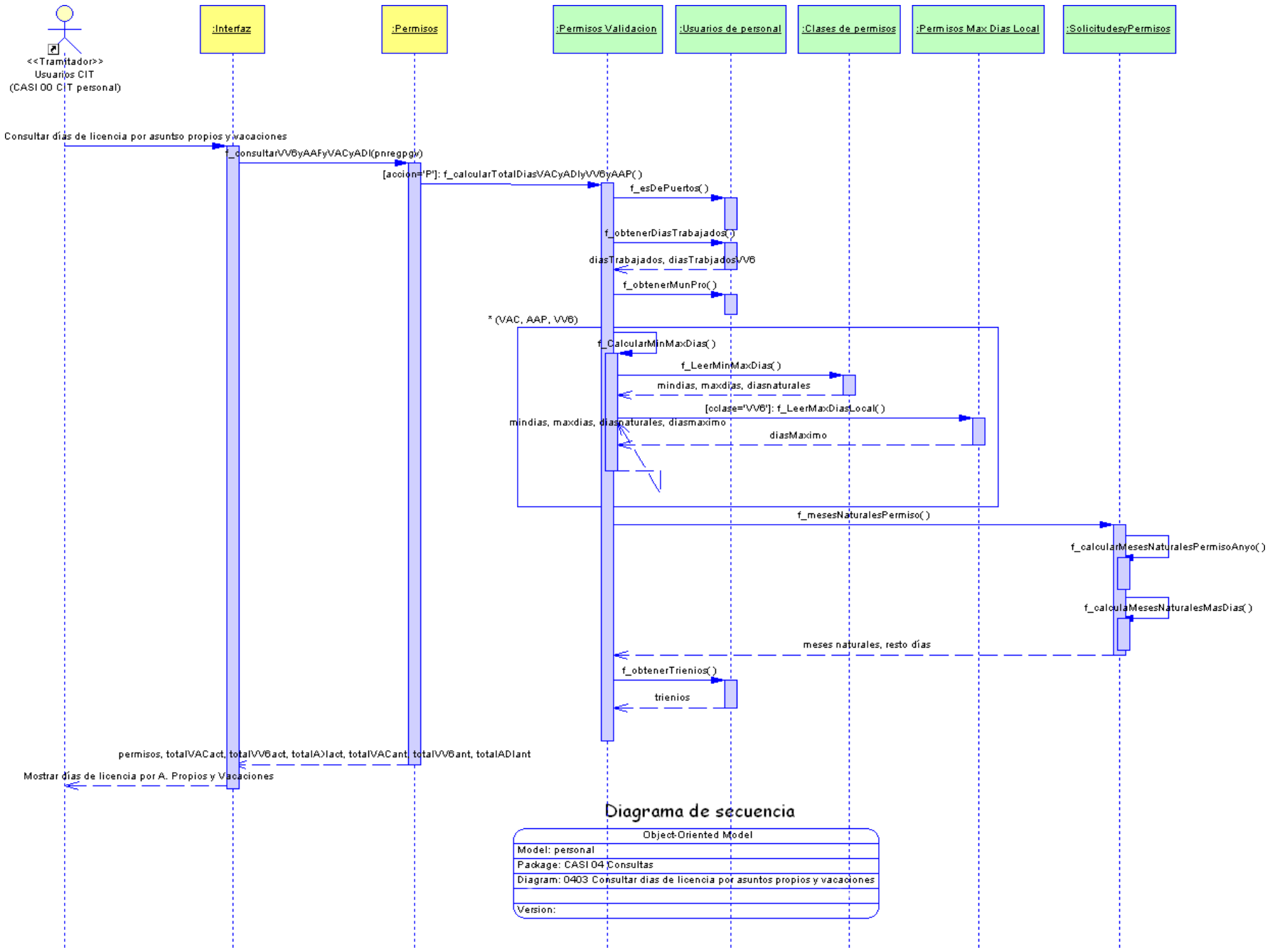
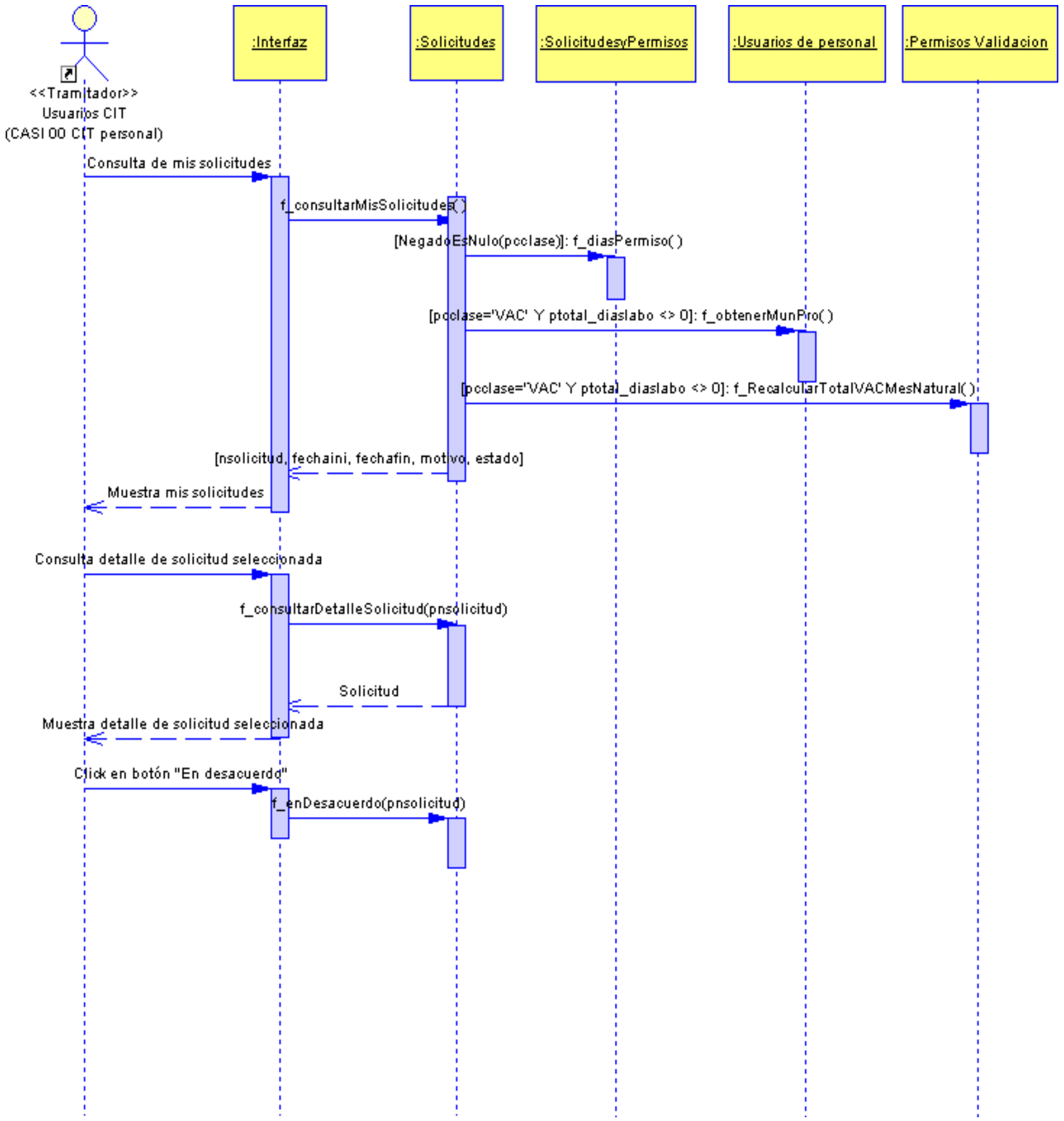


Diagrama de secuencia –CONSULTAR ESTADO DE MIS SOLICITUDES



4.3. Diagramas de clases

Diagrama de clase – ALTA DE PERMISOS Y SOLICITUDES

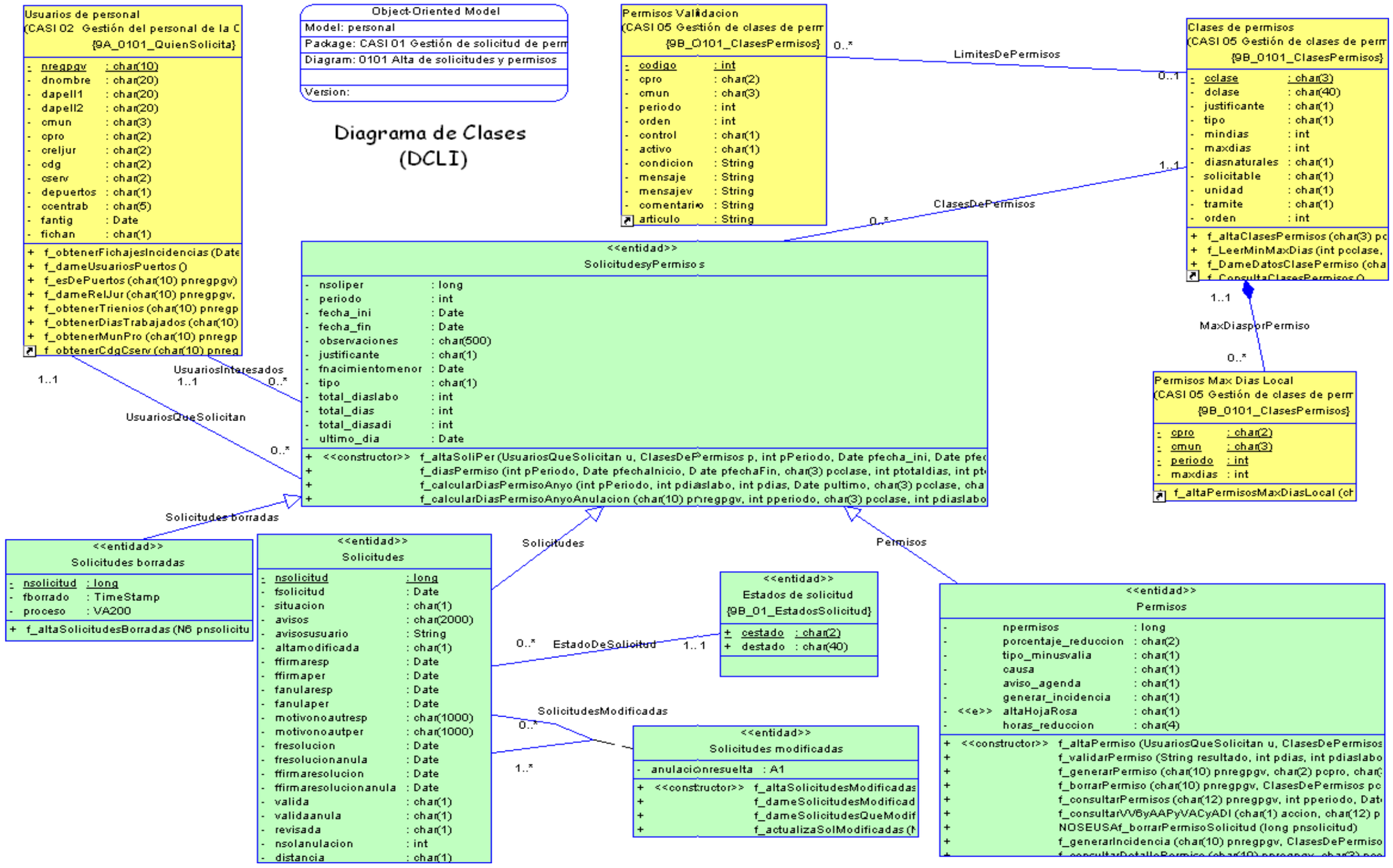


Diagrama de clase – ANULAR SOLICITUDES

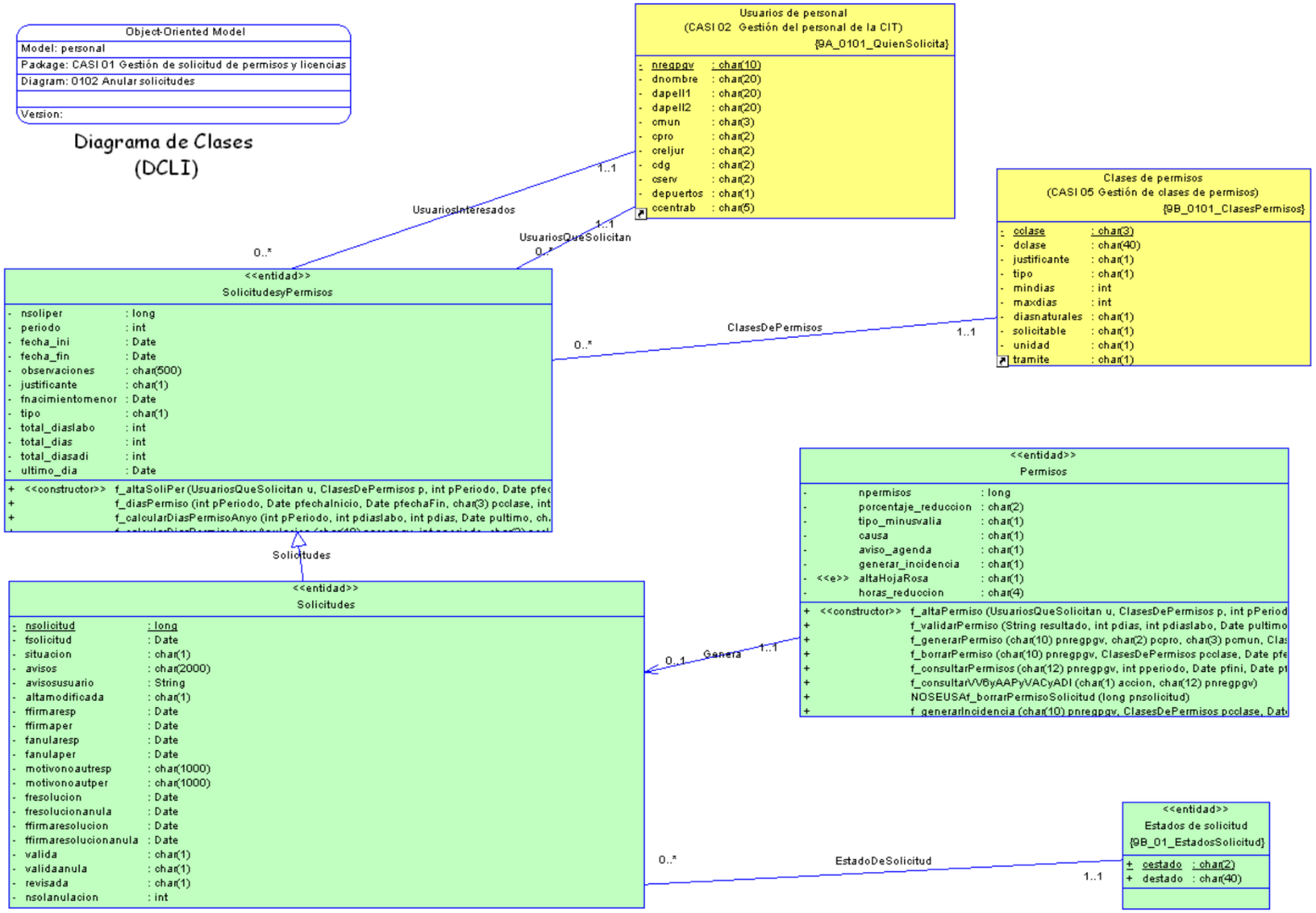


Diagrama de clase – CONSULTAR CONTROL DE PRESENCIA

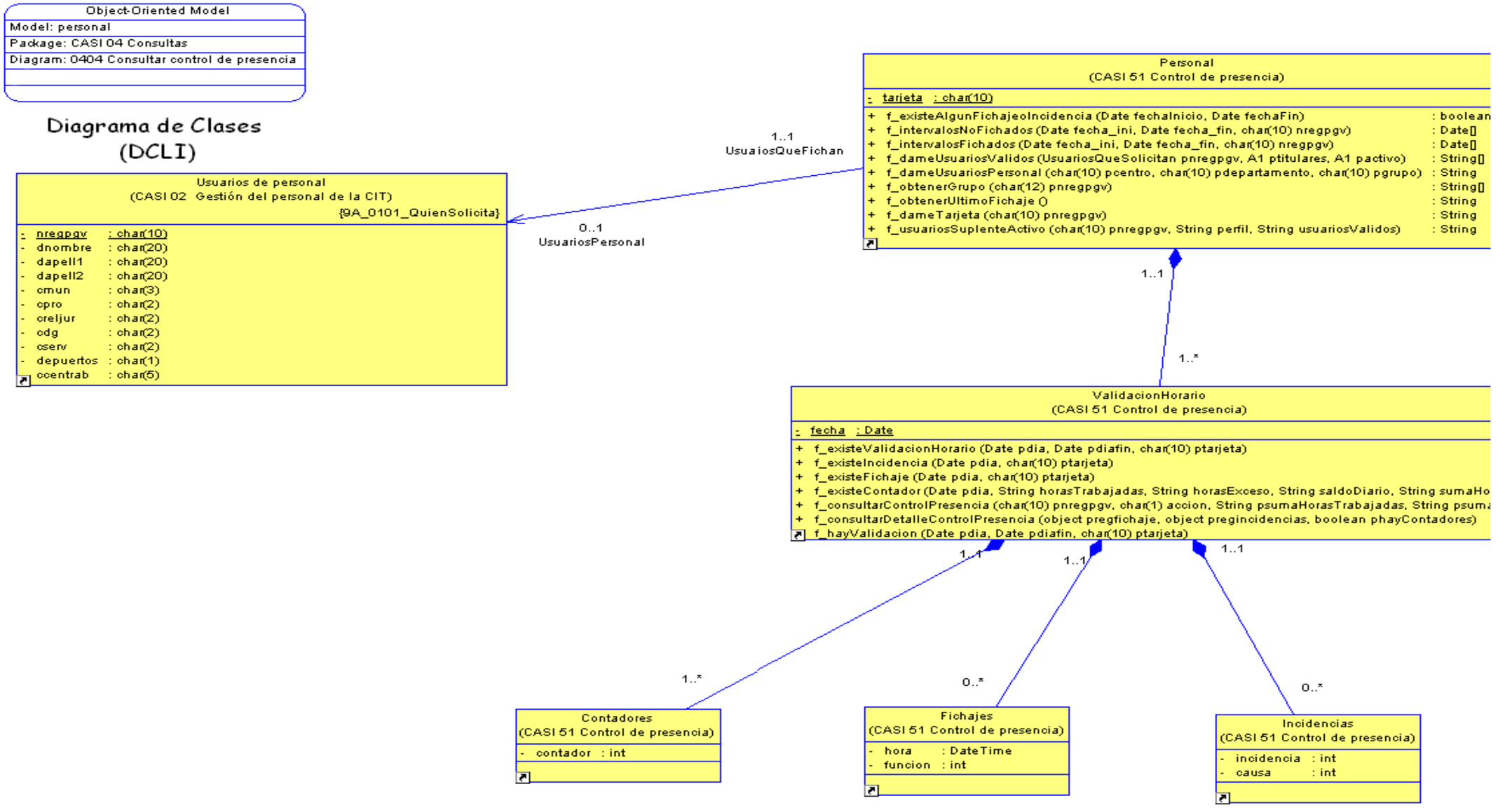
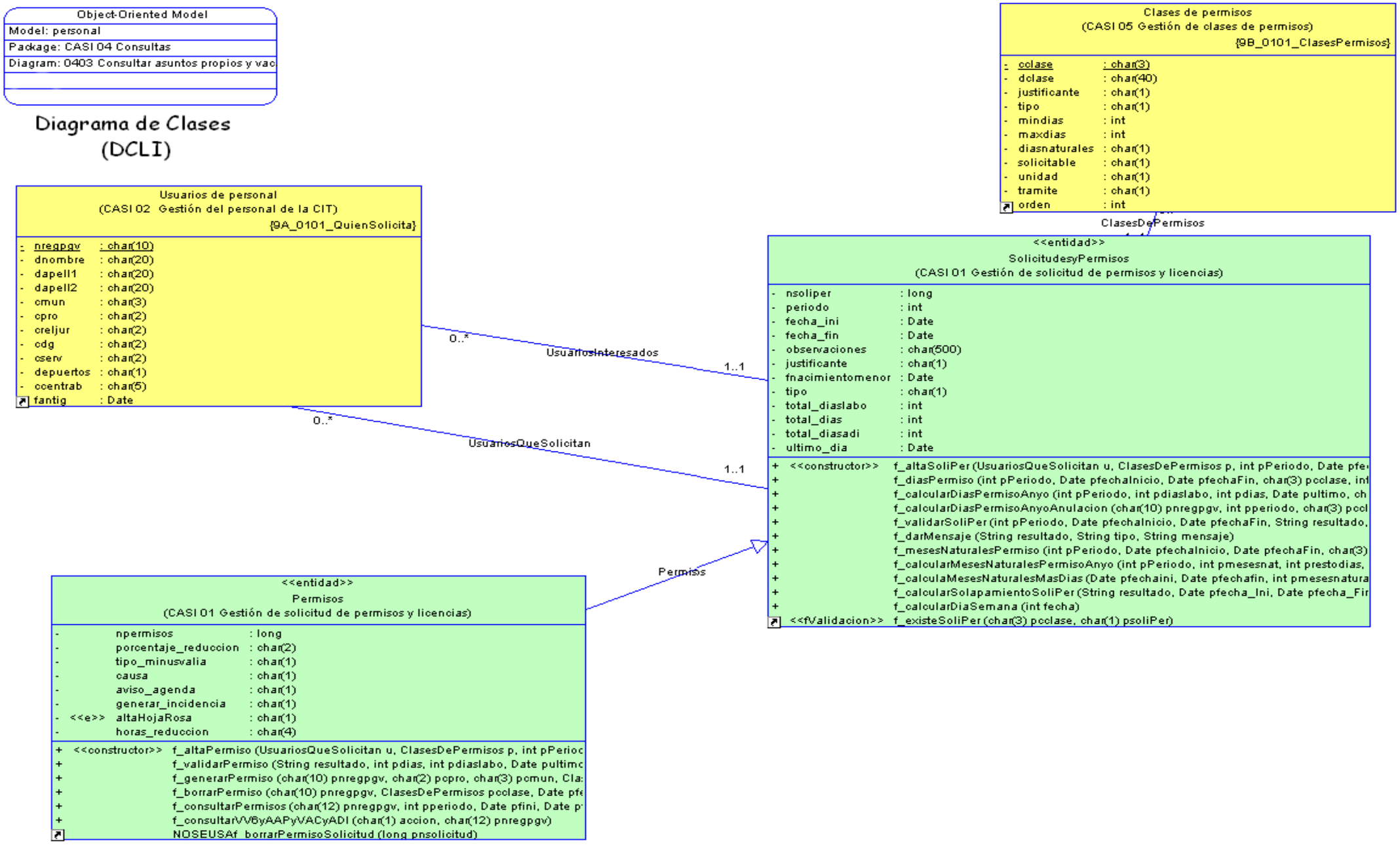


Diagrama de clase – CONSULTAR DÍAS DE LICENCIAS POR ASUNTOS PROPIOS Y VACACIONES



5. DISEÑO

5.1. Elección del Sistema de Gestión de Bases de Datos

La base de datos elegida para albergar el esquema de la aplicación ha sido PostGress, siguiendo la política actual de la Consellería hacia el sotfware libre, pero todavía se convive con Oracle, contenedora de algunos datos que por unos u otros motivos no se han migrado a Postgress. De hecho la aplicación de Permisos y Licencias accede a algunos datos en Oracle.

El Front-End elegido para el manejo de la B.D. ha sido pgAdmin III desde Linux.

5.2. Descripción de las principales tablas de nuestro esquema

Nomenclatura de las tablas

En nuestro esquema de B.D. distinguiremos entre *tablas* y *vistas* necesarias para nuestra aplicación

Todas nuestras tablas comenzarán por **tper_** que nos indicará que es una tabla de la base de datos Personal, y todas las vistas comenzarán por **vper_** que nos indicará que es una vista de la base de datos Personal, seguido del nombre descriptivo de la tabla.

Por ejemplo, **tper_solicitudes** es la tabla de Personal donde los usuarios irán grabando las solicitudes de sus permisos y licencias

Descripción de las principales tablas

tper_solicitudes

Tabla de Personal donde los usuarios irán grabando las solicitudes de sus permisos y licencias

Su esquema de creación será el siguiente:

```
CREATE TABLE tper_solicitudes(

nsolicitud numeric(6) NOT NULL DEFAULT nextval('sper_solicitud'::regclass), -- Nº de la solicitud de permisos y licencias

fsolicitud date NOT NULL, -- Fecha de la solicitud de permisos y licencias

periodo numeric(4) NOT NULL, -- Año al que corresponde el permiso

nregpgv char(10) NOT NULL, -- Nº de registro personal de la persona interesada

cclase varchar(3) NOT NULL, -- Código de la clase de permiso

fecha_ini date NOT NULL, -- Fecha de inicio del permiso que se solicita

fecha_fin date, -- Fecha de fin del permiso solicitado

observaciones varchar(500), -- Observaciones relacionadas con la clase de permiso que se solicita

cestado varchar(2) NOT NULL, -- Código del estado de la solicitud

situacion char(1) NOT NULL, -- Situación en la que se encuentra la solicitud: A - Solicitud de alta; X - Solicitud de anulación con firma de responsable en el alta, sin pasar por personal el alta; B - Solicitud de anulación con firma de personal en el alta

justificante char(1) NOT NULL, -- Indica si la persona aporta o no justificante (S/N)

avisos varchar(2000), -- Avisos ocurridos durante la introducción del permiso

altamodificada char(1) NOT NULL, -- Es un alta modificada: S - Nueva solicitud de alta tras una modificación parcial del alta; N - No es un alta modificada

nrpresp char(10), -- Nº de registro personal de la persona responsable

ffirmaresp date, -- Fecha de la firma del responsable

nrpper char(10), -- Nº de registro personal de la persona responsable de personal

ffirmaper date, -- Fecha de la firma de personal

nrpanularesp char(10), -- Nº de registro personal de la persona responsable que anula

fanularesp date, -- Fecha de la firma de anulación del responsable

motivonoautresp varchar(1000), -- Motivo de la no autorización del responsable

nrpanulaper char(10), -- Nº de registro personal de la persona responsable de personal que anula

fanulaper date, -- Fecha de la firma de anulación de personal
```

```
motivonoautper varchar(1000), -- Motivo de la no autorización de personal
fresolucion date, -- Fecha de la resolución de alta (positiva si estado = 'AP' y negativa si estado = 'NP')
fresolucionanula date, -- Fecha de la resolución de anulación (positiva si estado = 'AP' y negativa si estado = 'NP')
fnacimientomenor date, -- Indica la fecha de nacimiento del menor
valida char(1), -- Indica si la solicitud de alta ha sido validada como correcta por el tramitador o revisor de personal
validaanula char(1), -- Indica si la solicitud de anulación ha sido validada como correcta por el tramitador o revisor de personal
revisada char(1), -- Indica si la solicitud de alta ha sido revisada por el revisor de personal antes de pasar a la firma del responsable de personal
ffirmaresolucion date, -- Fecha de firma de la resolución de alta (positiva si estado = "AP" y negativa si estado = "NP")
ffirmaresolucionanula date, -- Fecha de firma de la resolución de anulación (positiva si estado = "AP" y negativa si estado = "NP")
nsolanulacion numeric(6), -- Nº de solicitud de una solicitud existente que se ha podido modificar por una anulación parcial
avisosusuario varchar(1000), -- Avisos ocurridos durante la introducción del permiso que hacen referencia a una ley o decreto
distancia char(1), -- Indica si EGP o DEF O TDO ocurre a más de 100 Km (L) o menos (C) o bien si MFA o MUH ocurre a más de 375 km (L) o menos (C)
pendientenuevojustificante char(1) NOT NULL DEFAULT 'N'::bpchar, -- Indica si la persona aporta o no un nuevo justificante correcto (S/N)
numresol numeric(2), -- Número de secuencia diaria de la resolución de alta (positiva si estado = 'AP' y negativa si estado = 'NP')
numresolanula numeric(2), -- Número de secuencia diaria de la resolución de anulación (positiva si estado = 'AP' y negativa si estado = 'NP')
nrptramitador char(10), -- Nº de registro personal de la persona que ha validado la solicitud de alta
fvalidatramitador date, -- Fecha de la validación de la solicitud de alta
requierejustificante char(1) NOT NULL DEFAULT 'N'::bpchar, -- Indica si el permiso de la solicitud requiere o no aportar justificante (S/N)
nrpanulatramitador char(10), -- Nº de registro personal de la persona que ha validado la solicitud de anulación
fvalidaanulatramitador date, -- Fecha de la validación de la solicitud de anulación
nrprevisor char(10), -- Nº de registro personal de la persona que ha revisado la solicitud de alta
fvalidarevisor date, -- Fecha de la revisión de la solicitud de alta
nrpanularevisor char(10), -- Nº de registro personal de la persona que ha revisado la solicitud de anulación
fvalidaanularevisor date, -- Fecha de la revisión de la solicitud de anulación
nrpgeneraresol char(10), -- Nº de registro personal de la persona que ha generado la resolución de la solicitud de alta
nrpgeneraresolanula char(10), -- Nº de registro personal de la persona que ha generado la resolución de la solicitud de anulación
nrpnotificaresol char(10), -- Nº de registro personal de la persona que ha notificado la resolución de la solicitud de alta
nrpnotificaresolanula char(10), -- Nº de registro personal de la persona que ha notificado la resolución de la solicitud de anulación
gradoparentesco char(1), -- Indica si el grado de parentesco de un familiar, para EGP o DEF, es de 1er grado (1) o 2º grado (2)
CONSTRAINT tper_solicitudes_pkey PRIMARY KEY (nsolicitud),
CONSTRAINT fk_estadodesolici_tper_estados FOREIGN KEY (cestado)
    REFERENCES tper_estadosol (cestado) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION,
CONSTRAINT fk_solclasesdepermis_tper_cod_ FOREIGN KEY (cclase)
    REFERENCES tper_cod_permisos (cclase) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION,
CONSTRAINT tper_solicitudes_altamodificada_check CHECK (altamodificada = 'S'::bpchar OR altamodificada = 'N'::bpchar),
CONSTRAINT tper_solicitudes_distancia_check CHECK (distancia = ANY (ARRAY['C'::bpchar, 'L'::bpchar])),
CONSTRAINT tper_solicitudes_gradoparentesco_check CHECK (gradoparentesco = ANY (ARRAY['1'::bpchar, '2'::bpchar])),
CONSTRAINT tper_solicitudes_justificante_check CHECK (justificante = 'N'::bpchar OR justificante = 'S'::bpchar),
CONSTRAINT tper_solicitudes_pendientenuevojustificante_check CHECK (pendientenuevojustificante = ANY (ARRAY['S'::bpchar, 'N'::bpchar])),
CONSTRAINT tper_solicitudes_requierejustificante_check CHECK (requierejustificante = ANY (ARRAY['N'::bpchar, 'S'::bpchar])),
CONSTRAINT tper_solicitudes_revisada_check CHECK (revisada IS NULL OR revisada = 'S'::bpchar OR revisada = 'N'::bpchar),
CONSTRAINT tper_solicitudes_situacion_check CHECK (situacion = 'A'::bpchar OR situacion = 'X'::bpchar OR situacion = 'Y'::bpchar OR situacion = 'B'::bpchar),
CONSTRAINT tper_solicitudes_valida_check CHECK (valida IS NULL OR valida = 'S'::bpchar OR valida = 'N'::bpchar),
CONSTRAINT tper_solicitudes_validaanula_check CHECK (validaanula IS NULL OR validaanula = 'S'::bpchar OR validaanula = 'N'::bpchar)
)
WITHOUT OIDS;
ALTER TABLE tper_solicitudes OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE tper_solicitudes TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE tper_solicitudes TO rpersonal;
GRANT SELECT ON TABLE tper_solicitudes TO rpersonal_c;
```

tper_permisos

Tabla de Personal donde los usuarios irán grabando los permisos y licencias que no sean solicitables

Su esquema de creación será el siguiente:

```
CREATE TABLE tper_permisos(
nregpgv char(10) NOT NULL, -- Nº de registro personal de la persona
cclase varchar(3) NOT NULL, -- Código de la clase de permiso
fecha_ini date NOT NULL, -- Fecha de inicio del permiso que se solicita
fecha_fin date, -- Fecha de fin del permiso solicitado
observaciones varchar(500), -- Observaciones relacionadas con la clase de permiso que se solicita
periodo numeric(4) NOT NULL, -- Año al que corresponde el permiso
nsolicitud numeric(6), -- Nº de la solicitud de permisos y licencias
porcentaje_reduccion varchar(2), -- Porcentaje de reducción de la clase de permiso 'RJP' o 'RJG'
tipo_minusvalia char(1), -- Tipo de minusvalía de la clase de permiso 'RJM': P: Padres; H: Hijos; R: Propios; O: Otros
causa char(1), -- Causa de la clase de permiso 'IT':
justificante char(1) NOT NULL, -- Indica si la persona aporta o no justificante (S/N)
aviso_agenda char(1) NOT NULL DEFAULT 'N'::bpchar, -- Se envia a la agenda un mensaje de aviso de cuándo finaliza el permiso o la reducción.
cpro char(2), -- Provincia
cmun char(3), -- Municipio
fnacimientomenor date, -- Indica la fecha de nacimiento del menor
horas_reduccion varchar(4), -- Horas de reducción del permiso
CONSTRAINT tper_permisos_pkey PRIMARY KEY (nregpgv, cclase, fecha_ini),
CONSTRAINT fk_genera_tper_solicitu FOREIGN KEY (nsolicitud)
    REFERENCES tper_solicitudes (nsolicitud) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION,
CONSTRAINT fk_pclasesdepermis_tper_cod_p FOREIGN KEY (cclase)
    REFERENCES tper_cod_permisos (cclase) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION,
CONSTRAINT tper_permisos_aviso_agenda_check CHECK (aviso_agenda::text = 'N'::text OR aviso_agenda::text = 'S'::text),
CONSTRAINT tper_permisos_causa_check CHECK (causa IS NULL OR (causa = ANY (ARRAY['C'::bpchar, '1'::bpchar, 'P'::bpchar, '2'::bpchar, 'L'::bpchar, '3'::bpchar]))),
CONSTRAINT tper_permisos_justificante_check CHECK (justificante = 'N'::bpchar OR justificante = 'S'::bpchar),
CONSTRAINT tper_permisos_tipo_minusvalia_check CHECK (tipo_minusvalia IS NULL OR tipo_minusvalia::text = 'P'::text OR tipo_minusvalia::text = 'H'::text OR tipo_minusvalia::text = 'R'::text OR tipo_minusvalia::text = 'O'::text)
)
WITHOUT OIDS;
ALTER TABLE tper_permisos OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE tper_permisos TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE tper_permisos TO rpersonal;
GRANT SELECT ON TABLE tper_permisos TO rpersonal_c;
GRANT SELECT ON TABLE tper_permisos TO rpersonal_repl;
```

tper_estadosol

Tabla de Personal que nos describe los posibles estados por los que puede pasar una solicitud.

Su esquema de creación será el siguiente:

```
CREATE TABLE tper_estadosol
(
cestado varchar(2) NOT NULL, -- Código del estado de la solicitud
destado varchar(40), -- Descripción del estado de la solicitud
CONSTRAINT tper_estadosol_pkey PRIMARY KEY (cestado)
)
WITHOUT OIDS;
```

```
ALTER TABLE tper_estadosol OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE tper_estadosol TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE tper_estadosol TO rpersonal;
GRANT SELECT ON TABLE tper_estadosol TO rpersonal_c;
```

tper_personas

Tabla de Personasl que trabajan en la Consellería.

Su esquema de creación será el siguiente:

```
CREATE TABLE tper_personas
(
nregpgv char(10) NOT NULL, -- Número Registro Personal G.V.
dapell1 varchar(20), -- Apellido 1
dapell2 varchar(20), -- Apellido 2
dnombre varchar(20), -- Nombre
fnacim date, -- Fecha de nacimiento
csexo char(1), -- Sexo (H;D)
ddirecc varchar(40), -- Dirección (Calle, número)
cpost char(5), -- Código postal
cpro char(2), -- Provincia residencia
cmun char(3), -- Municipio residencia
ntelef1 varchar(10), -- Teléfono 1
dap2con varchar(20), -- Apellido 2 cónyuge
ccausacese char(2), -- Código cese
fbajp date, -- Fecha de baja en personal
cgrupo char(1), -- Grupo clasificación
ngracon char(2), -- Grado consolidado
pjor char(2), -- Porcentaje de Jornada
dobserv varchar(70), -- Observaciones generales
factua date, -- Fecha de la última actualización
fantig date, -- Fecha de antigÜedad
ntrie numeric(2), -- Número de trienios
asin char(1), -- Liberado sindical
horas char(2), -- Horas sindicales
cursosantes94 char(3), -- Cursos realizados antes del 94
ctitulo char(5), -- Titulación Académica de la persona
faviso date, -- Fecha de revisión
tiporeduc char(1), -- Tipo de reducción: X - sin reducción; S - Con reducción de jornada de: <pjor>; N - Reducción de 1h diaria por minusvalía hasta: <faviso>
minusvalia_de char(1), -- Indica de quien es la minusvalía
agenda char(1), -- Indica si se ha avisado en la agenda del fin de la reducción
fconsol date,
cban numeric(4),
cagen numeric(4),
dg numeric(2),
ncta numeric(10),
freddesde date,
diasadi_act numeric(1), -- Dias adicionales de la persona en el año actual
diasadi_ant numeric(1), -- Dias adicionales del año anterior (sólo para el 2003)
tipoadmon varchar(1),
ccompe char(4),
fcomperesol date,
fgruporesol date,
prolongaservactivo char(1), -- Indica si la persona prolonga su servicio activo después de los 65 años hasta la edad máxima de jubilación que son los 70
```

```
ficha char(1) NOT NULL, -- Indica si la persona ficha (S) / no ficha (N) / ficha o no según el centro de trabajo (C)
fcesefuturo date, -- Fecha de cese de la persona en un futuro próximo
sindicato char(2), -- Sindicato al que está afiliado el liberado sindical
CONSTRAINT tper_personas_pkey PRIMARY KEY (nregpgv),
CONSTRAINT ckc_tper_persona_sindicato CHECK (sindicato IS NULL OR (sindicato = ANY (ARRAY['CO'::bpchar, 'UG'::bpchar, 'CC'::bpchar, 'IG'::bpchar, 'IS'::bpchar, 'OT'::bpchar]))),
CONSTRAINT tper_personas_ficha_check CHECK (ficha = ANY (ARRAY['S'::bpchar, 'N'::bpchar, 'C'::bpchar]))
)
WITHOUT OIDS;
ALTER TABLE tper_personas OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE tper_personas TO personal;
GRANT SELECT ON TABLE tper_personas TO rpersonal;
GRANT SELECT ON TABLE tper_personas TO rpersonal_c;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE tper_personas TO rpersonal_repl;
```

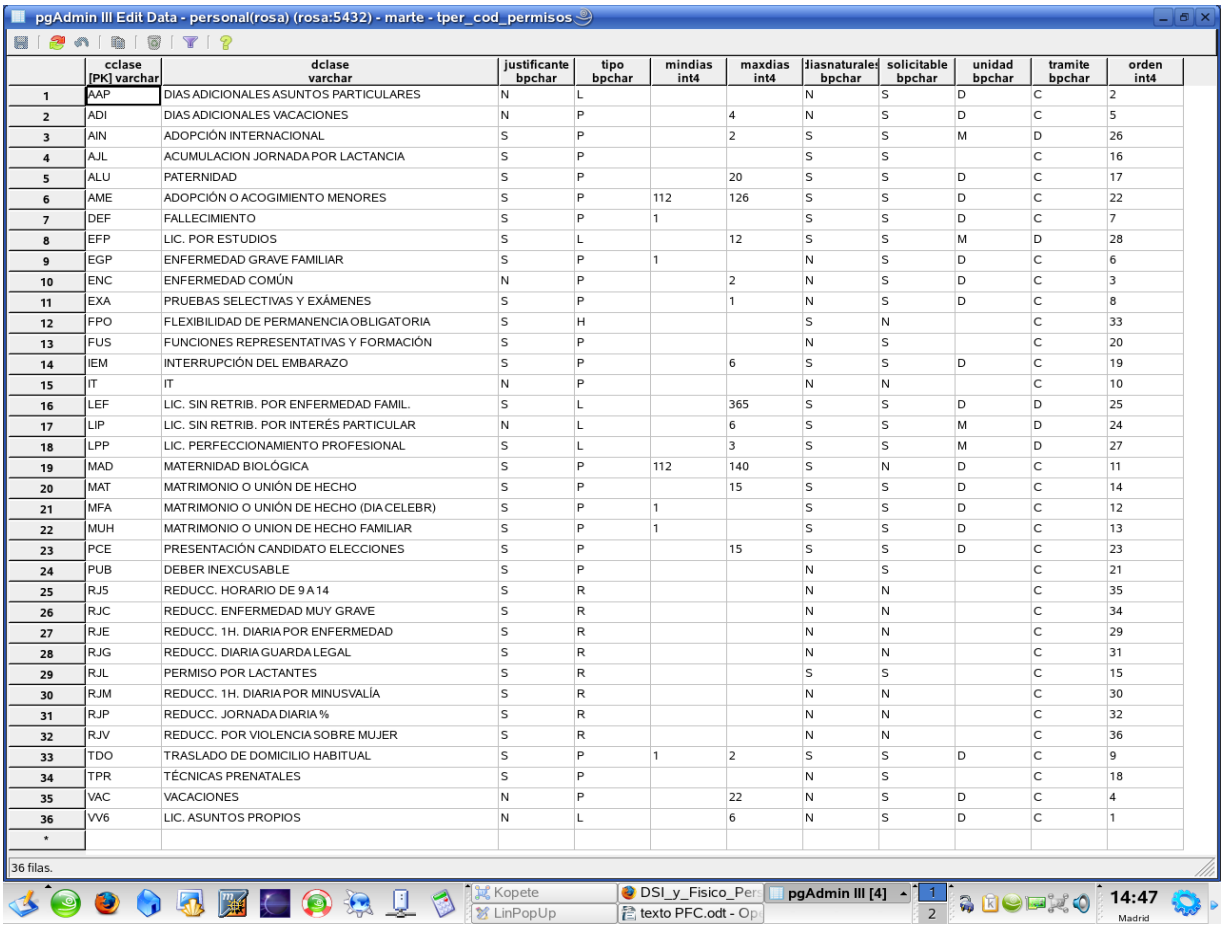
tper_cod_permisos

Tabla que nos describe los distintos tipos de permisos y licencias de que gozan los trabajadoresn en la Consellería.

Su esquema de creación será el siguiente:

```
CREATE TABLE tper_cod_permisos
(
cclase varchar(3) NOT NULL, -- Código de la clase de permiso
dclase varchar(40), -- Descripción de la clase de permiso
justificante char(1) NOT NULL, -- Indica si el permiso requiere o no justificante: S/N
tipo char(1) NOT NULL, -- Indica si es un permiso (P), licencia (L), reducción (R) u horario (H)
mindias int4, -- Mínimo número de lo indicado en unidad que se pueden disfrutar de la clase de permiso
maxdias int4, -- Máximo número de lo indicado en unidad que se pueden disfrutar de la clase de permiso
diasnaturales char(1) NOT NULL, -- Indica si mindias y maxdias cuentan días naturales (S) o hábiles (N)
solicitable char(1) NOT NULL, -- Indica si es una solicitud de un permiso solicitada por el usuario interesado (S) o un permiso introducido por el tramitador de personal (N)
unidad char(1), -- Indica si mindias y maxdias hacen referencia a días (D) o meses (M)
tramite char(1) NOT NULL, -- Indica si la clase de permiso es tramitada por la CIT (C) o por la Dirección General de Administración Autonómica (D)
orden int4 NOT NULL, -- Indica el orden en que se muestran las clases de permisos en la lista de motivos de una solicitud
CONSTRAINT tper_cod_permisos_pkey PRIMARY KEY (cclase),
CONSTRAINT tper_cod_permisos_diasnaturales_check CHECK (diasnaturales = 'S'::bpchar OR diasnaturales = 'N'::bpchar),
CONSTRAINT tper_cod_permisos_justificante_check CHECK (justificante = 'S'::bpchar OR justificante = 'N'::bpchar),
CONSTRAINT tper_cod_permisos_solicitable_check CHECK (solicitable::text = 'S'::text OR solicitable::text = 'N'::text),
CONSTRAINT tper_cod_permisos_tipo_check CHECK (tipo::text = 'P'::text OR tipo::text = 'L'::text OR tipo::text = 'R'::text OR tipo::text = 'H'::text),
CONSTRAINT tper_cod_permisos_tramite_check CHECK (tramite = ANY (ARRAY['C'::bpchar, 'D'::bpchar]))
)
WITHOUT OIDS;
ALTER TABLE tper_cod_permisos OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE tper_cod_permisos TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE tper_cod_permisos TO rpersonal;
GRANT SELECT ON TABLE tper_cod_permisos TO rpersonal_c;
GRANT SELECT ON TABLE tper_cod_permisos TO rpersonal_repl;
```

Ejemplo de los datos albergados en nuestra tabla donde se pueden ver los distintos permisos y licencias existentes para los empleados de la Consellería:



pgAdmin III Edit Data - personal(rosa) (rosa:5432) - marte - tper_cod_permisos

	cclase [PK] varchar	dclase varchar	justificante bpchar	tipo bpchar	mindias int4	maxdias int4	diasnaturales bpchar	solicitable bpchar	unidad bpchar	tramite bpchar	orden int4
1	AAP	DIAS ADICIONALES ASUNTOS PARTICULARES	N	L			N	S	D	C	2
2	ADI	DIAS ADICIONALES VACACIONES	N	P		4	N	S	D	C	5
3	AIN	ADOPCIÓN INTERNACIONAL	S	P		2	S	S	M	D	26
4	AJL	ACUMULACION JORNADA POR LACTANCIA	S	P			S	S		C	16
5	ALU	PATERNIDAD	S	P		20	S	S	D	C	17
6	AME	ADOPCIÓN O ACOGIMIENTO MENORES	S	P	112	126	S	S	D	C	22
7	DEF	FALLECIMIENTO	S	P	1		S	S	D	C	7
8	EFP	LIC. POR ESTUDIOS	S	L		12	S	S	M	D	28
9	EGP	ENFERMEDAD GRAVE FAMILIAR	S	P	1		N	S	D	C	6
10	ENC	ENFERMEDAD COMÚN	N	P		2	N	S	D	C	3
11	EXA	PRUEBAS SELECTIVAS Y EXÁMENES	S	P		1	N	S	D	C	8
12	FPO	FLEXIBILIDAD DE PERMANENCIA OBLIGATORIA	S	H			S	N		C	33
13	FUS	FUNCIONES REPRESENTATIVAS Y FORMACIÓN	S	P			N	S		C	20
14	IEM	INTERRUPCIÓN DEL EMBARAZO	S	P		6	S	S	D	C	19
15	IT	IT	N	P			N	N		C	10
16	LEF	LIC. SIN RETRIB. POR ENFERMEDAD FAMIL.	S	L		365	S	S	D	D	25
17	LIP	LIC. SIN RETRIB. POR INTERÉS PARTICULAR	N	L		6	S	S	M	D	24
18	LPP	LIC. PERFECCIONAMIENTO PROFESIONAL	S	L		3	S	S	M	D	27
19	MAD	MATERNIDAD BIOLÓGICA	S	P	112	140	S	N	D	C	11
20	MAT	MATRIMONIO O UNIÓN DE HECHO	S	P		15	S	S	D	C	14
21	MFA	MATRIMONIO O UNIÓN DE HECHO (DIA CELEBR)	S	P	1		S	S	D	C	12
22	MUH	MATRIMONIO O UNION DE HECHO FAMILIAR	S	P	1		S	S	D	C	13
23	PCE	PRESENTACIÓN CANDIDATO ELECCIONES	S	P		15	S	S	D	C	23
24	PUB	DEBER INEXCUSABLE	S	P			N	S		C	21
25	RJ5	REDUCC. HORARIO DE 9 A 14	S	R			N	N		C	35
26	RJC	REDUCC. ENFERMEDAD MUY GRAVE	S	R			N	N		C	34
27	RJE	REDUCC. 1H. DIARIA POR ENFERMEDAD	S	R			N	N		C	29
28	RJG	REDUCC. DIARIA GUARDA LEGAL	S	R			N	N		C	31
29	RJL	PERMISO POR LACTANTES	S	R			S	S		C	15
30	RJM	REDUCC. 1H. DIARIA POR MINUSVALÍA	S	R			N	N		C	30
31	RJP	REDUCC. JORNADA DIARIA %	S	R			N	N		C	32
32	RJV	REDUCC. POR VIOLENCIA SOBRE MUJER	S	R			N	N		C	36
33	TDO	TRASLADO DE DOMICILIO HABITUAL	S	P	1	2	S	S	D	C	9
34	TPR	TÉCNICAS PRENATALES	S	P			N	S		C	18
35	VAC	VACACIONES	N	P		22	N	S	D	C	4
36	VV6	LIC. ASUNTOS PROPIOS	N	L		6	N	S	D	C	1
*											

36 filas.

Kopete | DSI_y_Fisico_Per | pgAdmin III [4] | 1 | 14:47 Madrid
LinPopUp | texto PFC.odt - Op | 2

tper_cod_centros_trab

Tabla que nos describe los distintos centros de trabajos adcritos a la Consellería.

Su esquema de creación será el siguiente:

```
CREATE TABLE tper_cod_centros_trab

(

ccentrab char(5) NOT NULL, -- Código centro

dcentrab varchar(40), -- Descripción centro

direccion varchar(40), -- Dirección Centro de Trabajo

cmun char(3), -- Municipio

cpro char(2), -- Provincia

ntelefon char(10), -- Teléfono

codpost char(5), -- Código Postal

categoria char(2), -- Categoría del centro: ST, SC, OT

vigente char(1), -- Indica si el centro de trabajo esta vigente (S/N)

ficha char(1) NOT NULL, -- Indica si se ficha o no en el centro de trabajo (S/N)
```

```
nregpgv char(10), -- Indica el nº de registro personal de la persona firmante, en caso de ser un centro de trabajo donde no se fiche
ONSTRAINT tper_cod_centros_trab_pkey PRIMARY KEY (ccentrab),
ONSTRAINT tper_cod_centros_trab_ficha_check CHECK (ficha = ANY (ARRAY['S'::bpchar, 'N'::bpchar]))
)
WITHOUT OIDS;
ALTER TABLE tper_cod_centros_trab OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE tper_cod_centros_trab TO personal;
GRANT SELECT ON TABLE tper_cod_centros_trab TO rpersonal;
GRANT SELECT ON TABLE tper_cod_centros_trab TO rpersonal_c;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE tper_cod_centros_trab TO rpersonal_repl;
```

tper_firmantes

Tabla que nos describe los responsables que pueden firmar las solicitudes del personal a su cargo, durante un determinado período

Su esquema de creación será el siguiente:

```
CREATE TABLE tper_firmantes
(
nregpgv char(10) NOT NULL, -- Nº de registro personal de la persona
centro varchar(10) NOT NULL, -- Centro de la CIT
departamento varchar(10) NOT NULL, -- Departamento de la CIT
seccion varchar(10) NOT NULL, -- Sección de la CIT
f_inicio date NOT NULL, -- Fecha de inicio de validez de la autorización de firma
f_fin date, -- Fecha de fin de validez de la autorización de firma
suplente char(1) NOT NULL, -- El firmante es suplente o no
activo char(1) DEFAULT 'N'::bpchar, -- Indica si el firmante está activo cuando es suplente
f_fin_activo timestamp, -- Fecha y hora de fin de activación temporal del suplente activo
CONSTRAINT tper_firmantes_pkey PRIMARY KEY (nregpgv, centro, departamento, seccion),
CONSTRAINT tper_firmantes_activo_check CHECK (activo = 'S'::bpchar OR activo = 'N'::bpchar),
CONSTRAINT tper_firmantes_suplente_check CHECK (suplente = 'S'::bpchar OR suplente = 'N'::bpchar)
)
WITHOUT OIDS;
ALTER TABLE tper_firmantes OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE tper_firmantes TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE tper_firmantes TO rpersonal;
GRANT SELECT ON TABLE tper_firmantes TO rpersonal_c;
```

twps_centros

Tabla que nos describe los Centros de la CIT

Su esquema de creación será el siguiente:

```
CREATE TABLE twps_centros
(
codigo varchar(10) NOT NULL, -- Centro de la CIT
descripcion varchar(50), -- Descripción del centro de la CIT
responsable varchar(10),
CONSTRAINT twps_centros_pkey PRIMARY KEY (codigo)
)
WITHOUT OIDS;
ALTER TABLE twps_centros OWNER TO personal;
GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE twps_centros TO personal;
GRANT SELECT ON TABLE twps_centros TO rpersonal;
GRANT SELECT ON TABLE twps_centros TO rpersonal_c;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE twps_centros TO rpersonal_repl;
COMMENT ON TABLE twps_centros IS 'Centros de la CIT';
COMMENT ON COLUMN twps_centros.codigo IS 'Centro de la CIT';
COMMENT ON COLUMN twps_centros.descripcion IS 'Descripción del centro de la CIT';
```

twps_departamentos

Tabla que nos describe los Departamentos de la CIT

Su esquema de creación será el siguiente:

```
CREATE TABLE twps_departamentos
(
codigo varchar(10) NOT NULL, -- Departamento de la CIT
descripcion varchar(50), -- Descripción del departamento de la CIT
responsable varchar(10),
ivision varchar(1),
djunto varchar(10),
ecretaria varchar(10),
ONSTRAINT twps_departamentos_pkey PRIMARY KEY (codigo)

WITHOUT OIDS;
```

```
ALTER TABLE twps_departamentos OWNER TO personal;

GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE twps_departamentos TO personal;

GRANT SELECT ON TABLE twps_departamentos TO rpersonal;

GRANT SELECT ON TABLE twps_departamentos TO rpersonal_c;

GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE twps_departamentos TO rpersonal_repl;

COMMENT ON TABLE twps_departamentos IS 'Departamentos de la CIT';

COMMENT ON COLUMN twps_departamentos.codigo IS 'Departamento de la CIT';

COMMENT ON COLUMN twps_departamentos.descripcion IS 'Descripción del departamento de la CIT';
```

twps_secciones

Tabla que nos describe las Secciones de la CIT

Su esquema de creación será el siguiente:

```
CREATE TABLE twps_secciones

(

codigo varchar(10) NOT NULL, -- Sección de la CIT

descripcion varchar(50), -- Descripción de la sección de la CIT

responsable varchar(10),

CONSTRAINT twps_secciones_pkey PRIMARY KEY (codigo)

)

WITHOUT OIDS;

ALTER TABLE twps_secciones OWNER TO personal;

GRANT SELECT, UPDATE, INSERT, DELETE, REFERENCES, TRIGGER ON TABLE twps_secciones TO personal;

GRANT SELECT ON TABLE twps_secciones TO rpersonal;

GRANT SELECT ON TABLE twps_secciones TO rpersonal_c;

GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE twps_secciones TO rpersonal_repl;

COMMENT ON TABLE twps_secciones IS 'Secciones de la CIT';

COMMENT ON COLUMN twps_secciones.codigo IS 'Sección de la CIT';

COMMENT ON COLUMN twps_secciones.descripcion IS 'Descripción de la sección de la CIT';
```

6. IMPLEMENTACIÓN

6.1. Alta de solicitudes y permisos

Clase Manejadora – Alta de solicitudes y permisos

```php
<?php



class AltaSolicPermisos extends gvHidraForm_DB{

	var $solicitables='S';
	function AltaSolicPermisos(){



	$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');

	//Las tablas sobre las que trabaja
	$nombreTablas= array('vper_ocupacion');
	parent::__construct($g_dsn,$nombreTablas);

	//Sacamos nrp de la sesión
	$datos = IgepSession::dameDatosUsuario();
	//Hack para crear una lista dependiente que no depende de un valor q en principio no estaba en la tpl
	//Ya no usamos esta variable para manejar la lista pero la mantenemos para el resto de acciones
	if(IgepSession::dameRol()=='P_TRAMITA') $this->solicitables='N';
	else $this->solicitables='S';
	//La select que mostramos en la tabla
	$this->setSelectForSearchQuery( "SELECT " .
	                                substr(nregpgv, 1,8) as \"usuNif\"
	                               , nregpgv as \"usuNreg\"
	                               , dapell1 ||' '|| dapell2 ||', '|| dnombre as \"usuNombre\"
	                               , substr(nregpgv, 1,8) as \"DNI\"

	                               , dapell1 ||' '|| dapell2 ||', '|| dnombre as \"Nombre\"
	                               , 'N' as \"depuertos\"
	                               , '' as \"motivo\"" .
	                               , '".date('Y')."' as \"periodo\"
	                               , '' as \"causa\"
	                               , '' as \"tipo\"

	                               , 'N' as \"incidencia\"
	                               , 'N' as \"AltaHojaRosa\"
	                               , '' as \"horas\"
	                               , '' as \"porcentaje\"
	                               , '' as \"observaciones\"
	                               , '' as \"fechaIni\"
	                               , '' as \"fechaFin\"
	                               , '' as \"fechaNaci\"
	                               , '' as \"distancia\"
	                               , '' as \"distanciaMFAoMUH\"
	                               , '' as \"gradoParentesco\"
	                               , '' as \"indicarFamiliar\"
	                               , '' as \"aportaJustificante\"
	                               , '' as \"minimo\"
	                               , '' as \"maximo\"
	                               , '' as \"unidad\"
	                               , ' ' as \"reqJustificante\"
	                               , '".$this->solicitables."' as \"solicitables\"

	                                FROM vper_ocupacion where nregpgv ='".$datos[nrp]."'");

	/*Añadimos los Matching - Correspondecias campoTPL <-> campoBD*/
	$this->addMatching("usuNif","nregpgv","vper_ocupacion");
	$this->addMatching("usuNreg","nregpgv","vper_ocupacion");
	$this->addMatching("usuNombre","dnombre","vper_ocupacion");

	$this->addMatching("depuertos","depuertos","vper_ocupacion");
	$this->addMatching("DNI","nregpgv","vper_ocupacion");

	$this->addMatching("Nombre","dnombre","vper_ocupacion");

	$this->addMatching("motivo","motivo","vper_ocupacion");
	$this->addMatching("periodo","periodo","vper_ocupacion");
	$this->addMatching("causa","causa","vper_ocupacion");
	$this->addMatching("tipo","tipo","vper_ocupacion");

	$this->addMatching("incidencia","incidencia","vper_ocupacion");
	$this->addMatching("AltaHojaRosa","AltaHojaRosa","vper_ocupacion");
	$this->addMatching("porcentaje","porcentaje","vper_ocupacion");

	$this->addMatching("horas","horas","vper_ocupacion");

	$this->addMatching("observaciones","observaciones","vper_ocupacion");

	$this->addMatching("fechaIni","fechaIni","vper_ocupacion");
	$this->addMatching("fechaFin","fechaFin","vper_ocupacion");
```

```
$this->addMatching("fechaNaci","fechaNaci","vper_ocupacion");

$this->addMatching("distancia","distancia","vper_ocupacion");
$this->addMatching("distanciaMFAoMUH","distancia","vper_ocupacion");
$this->addMatching("gradoParentesco","gradoParentesco","vper_ocupacion");
$this->addMatching("aportaJustificante","aportaJustificante","vper_ocupacion");
$this->addMatching("minimo","minimo","vper_ocupacion");
$this->addMatching("maximo","maximo","vper_ocupacion");
$this->addMatching("unidad","unidad","vper_ocupacion");


$this->addMatching("reqJustificante","reqJustificante","vper_ocupacion");

$this->addMatching("solicitables","solicitables","vper_ocupacion");

//Definicion de los tipos de los campos

$motivo =  new gvHidraString(true,50);
$this->addFieldType('motivo',$motivo);


$fecha_ini =  new gvHidraDate(true);
$fecha_ini->setDayOfWeek('short');
$fecha_ini->setCalendar(true);
$fecha_ini->enableServerValidation(true);
$this->addFieldType('fechaIni',$fecha_ini);

$var_horas = new gvHidraString(false,4);
$var_horas ->setInputMask('#:##');

$this->addFieldType('horas',$var_horas);

$fecha_fin =  new gvHidraDate(false);
$fecha_fin->setDayOfWeek('short');
$fecha_fin->setCalendar(true);
$this->addFieldType('fechaFin',$fecha_fin);

$fecha_naci =  new gvHidraDate(false);
$fecha_naci->setCalendar(true);
$fecha_naci->setDayOfWeek('none');
$this->addFieldType('fechaNaci',$fecha_naci);


$cadena_8_obligatorio=new gvHidraString(true,8);
$cadena_8_obligatorio->setInputMask('########');
$this->addFieldType('DNI',$cadena_8_obligatorio);

$cadena_500_no_obligatorio=new gvHidraString(false,500);
$this->addFieldType('observaciones',$cadena_500_no_obligatorio);


//Ventana selección de usuarios interesados

$interesados = new IgepVentanaSeleccion('DNI','INTERESADOS',array("DNI","Nombre"));
$interesados->setDependencia(array('depuertos'),array('vper_ocupacion.depuertos'));
$this->addVentanaSeleccion($interesados);


/*Si el usuario conectado es tramitador, deberá poder dar de alta todas las clases
 de permisos. Si no es tramitador, solo podrá realizar solicitudes, es decir, aquellas
 clases de permisos que son solicitables */
if(IgepSession::dameRol()=='P_TRAMITA'){
          $listaMotivo = new IgepLista('motivo','MOTIVO_TODOS');
          $listaMotivo->addOpcion('',' ');
          $this->addLista($listaMotivo);
}else{
          $listaMotivo = new IgepLista('motivo','MOTIVO');
          $listaMotivo->addOpcion('',' ');
          $this->addLista($listaMotivo);
}




$listaCausa = new IgepLista('causa');//,'CAUSA');

$listaCausa->addOpcion("C","Enfermedad comÃºn - Seguridad Social");
$listaCausa->addOpcion("1","Enfermedad comÃºn - Muface");
$listaCausa->addOpcion("P","Accidente trabajo/Enf. profesional - Seguridad Social");
$listaCausa->addOpcion("2","Accidente trabajo/Enf. profesional - Muface");
$listaCausa->addOpcion("L","Accidente no laboral - Seguridad Social");
$listaCausa->addOpcion("3","Accidente no laboral - Muface");
$this->addLista($listaCausa);

$listaTipo = new IgepLista('tipo');//,'TIPO');

$listaTipo->addOpcion('','');
$listaTipo->addOpcion("P","Padres");
$listaTipo->addOpcion("H","Hijos");
$listaTipo->addOpcion("R","Propios");
$listaTipo->addOpcion("O","Otros");
$this->addLista($listaTipo);

// declaración del radio button
$listaAdjuntaJustificante = new IgepLista('aportaJustificante');
$listaAdjuntaJustificante->setRadio(true);
$listaAdjuntaJustificante->addOpcion('S',' Si');
$listaAdjuntaJustificante->addOpcion('N',' No<br/>');
$listaAdjuntaJustificante->setSelected('N');
$this->addLista($listaAdjuntaJustificante);
```

```
		$checkGenerarIncidencia = new IgepCheckBox('incidencia');
		$checkGenerarIncidencia->setValueChecked('S');
		$checkGenerarIncidencia->setValueUnchecked('N');
		$this->addCheckBox($checkGenerarIncidencia);

		// alta papel
		$checkAltaHojaRosa = new IgepCheckBox('AltaHojaRosa');
		$checkAltaHojaRosa->setValueChecked('S');
		$checkAltaHojaRosa->setValueUnchecked('N');
		$this->addCheckBox($checkAltaHojaRosa);


		$listaDistancia = new IgepLista('distancia');
		$listaDistancia->addOpcion('C','A no mÃ¡s de 100 Km');
		$listaDistancia->addOpcion('L','A mÃ¡s de 100 Km');
		$listaDistancia->setSelected('C');
		$this->addLista($listaDistancia);

		$listaDistanciaMFAoMUH = new IgepLista('distanciaMFAoMUH');
		$listaDistanciaMFAoMUH->addOpcion('C','A no mÃ¡s de 375 Km');
		$listaDistanciaMFAoMUH->addOpcion('L','A mÃ¡s de 375 Km');
		$listaDistanciaMFAoMUH->setSelected('C');
		$this->addLista($listaDistanciaMFAoMUH);

		$listaGradoParentesco = new IgepLista('gradoParentesco');
		$listaGradoParentesco->addOpcion('2','2Âº grado');
		$listaGradoParentesco->addOpcion('1','1er grado');
		$this->addLista($listaGradoParentesco);

		// SI el usuario tiene perfil de tramitador:
		//      La lista desplegable del periodo se cargará con el año actual, el anterior y el siguiente
		// SINO:
		//       La lista desplegable del periodo se cargará con el año actual (valor por defecto),
		//       con el año anterior si la fecha actual es hasta el 1 de febrero,
		//       y con el año siguiente durante el mes de diciembre.
		// FIN SI
		$listaPeriodo = new IgepLista('periodo');
		$periodo= date('Y');
		$listaPeriodo->addOpcion($periodo,$periodo);
		$listaPeriodo->setSelected($periodo);
		$fechactual = new gvHidraTimestamp();
		$fechactual->setTime(0,0,0);

		if(IgepSession::dameRol()=='P_TRAMITA'){
			// Para los tramitadores mostramos siempre el periodo anterior
			$listaPeriodo->addOpcion($periodo-1,$periodo-1);
			//el año actual

			//y el siguiente
			$listaPeriodo->addOpcion($periodo+1,$periodo+1);
		}else{

			//y el siguiente durante el mes de diciembre.
			$f2 = new gvHidraTimestamp("12/01/".date('Y'));
			$f2->setTime(0,0,0);
			$f3 = new gvHidraTimestamp("12/31/".date('Y'));
			$f3->setTime(0,0,0);
			if((gvHidraTimestamp::cmp($fechactual,$f2)<=0) and (gvHidraTimestamp::cmp($fechactual,$f3)>=0))
				$listaPeriodo->addOpcion($periodo+1,$periodo+1);
		}
		//La lista desplegable del periodo se cargará con el año actual (valor por defecto),
		$this->addLista($listaPeriodo);

		//Checkbox
		$checkDepuertos = new IgepCheckBox('depuertos');
		$checkDepuertos->setValueChecked('S');
		$checkDepuertos->setValueUnchecked('N');
		$this->addCheckBox($checkDepuertos);


		/*Acciones de interfaz*/
		$this->addAccionInterfaz('DNI','validaDNI');//Validacion de nombre
		$this->addAccionInterfaz('depuertos','limpiaPuertos');//Validacion de nombre
		$this->addAccionInterfaz('fechaIni','actualizaFechaFin');
		$this->addAccionInterfaz('fechaFin','validaFechas');//Validación de fechas
		$this->addAccionInterfaz('motivo','modificarClasePermiso');//Modificación del panel en funcion de la lista MOTIVO
		$this->addAccionInterfaz('periodo','modificarPeriodo');//Modificación del panel en funcion de la lista PERIODO
		$this->addAccionInterfaz('distancia','modificarDistancia');//Modificación del panel en funcion de la lista DISTANCIA
		$this->addAccionInterfaz('gradoParentesco','modificarGradoParentesco');//Modificación del panel en funcion de la
                                                                                lista DISTANCIA o GRADO PARENTESCO
		$this->addAccionInterfaz('distanciaMFAoMUH','modificarDistancia');//Modificación del panel en funcion de la lista
                                                                                DISTANCIA

		$this->addAccionInterfaz('horas','modificarHorasReduccion');// Cuando se modifica el valor del tiempo de reduccion
                                    se calcula el porcentaje equivalente y se graba en pantalla y en el campo reduccion.

		$this->addAccionInterfaz('fechaNaci','modificarFechaNacimientoMenor'); // Cuando se modifica el valor de la fecha de
                                    nacimiento del menor se actuliza el final del periodo de reduccion guarda legal.

		}

/* ------------------------- ACCIONES DE INTERFAZ -------------------------  */

function modificarHorasReduccion($objDatos){

		//Cuando se modifica el valor del tiempo de reducción se calcula el porcentaje equivalente y se graba en pantalla y
		en el campo reducción
		$motivo = $objDatos->getValue('motivo');
		$pcclase= substr($motivo,0,strpos($motivo,'|'));
		$pHorasReduccion=$objDatos->getValue('horas');
		$interesado = $objDatos->getValue('DNI');
		$pobservaciones = $objDatos->getValue('observaciones');
```

```
	$pHorasReduccion=trim($pHorasReduccion);
	if (strlen($pHorasReduccion)==0){
		$pHorasReduccion="0:00";
	} else {
		$pHorasReduccion=str_pad($pHorasReduccion,4,"0",STR_PAD_RIGHT);

	}
	$v_horas=substr($pHorasReduccion,0,1);
	$v_minutos=substr($pHorasReduccion,2,2);

	$objDatos->setValue('horas',$pHorasReduccion);

	if ($v_horas>8){
		$objDatos->setValue('horas','');
		$objDatos->setValue('porcentaje','');
		$objDatos->setValue('observaciones','');
		$this->showMensaje('APL-GENERICO',array("El tiempo de reducción no puede ser mayor de 8 horas"));
		return -1;
	 } else if (($v_horas==8)and $v_minutos>0){
		$objDatos->setValue('horas','');
		$objDatos->setValue('porcentaje','');
		$objDatos->setValue('observaciones','');
		$this->showMensaje('APL-GENERICO',array("El tiempo de reducción no puede ser mayor de 8 horas"));
		return -1;
	 }

	//Pasamos DNI a nregpgv
	$cons="SELECT nregpgv FROM vper_ocupacion where nregpgv like '".$interesado."__'";
	$res=$this->consultar($cons);
	$pnregpgv=$res['0']['nregpgv'];

	if (($pcclase=='RJP')or($pcclase=='RJG')or($pcclase=='RJC')or($pcclase=='RJ5')or($pcclase=='RJV')){

		$pporcentajeReducion= UsuariosDePersonal::f_damePorcentajeReduccion($pHorasReduccion, $pnregpgv,
                                                                                    $phorasSemana);

	}elseif (($pcclase=='RJM')or($pcclase=='RJE')){
		$pporcentajeReducion= UsuariosDePersonal::f_damePorcentajeReduccion('1:00', $pnregpgv,$phorasSemana);
	}

	if (($phorasSemana==null)or($phorasSemana==0)){
		UsuariosDePersonal::f_dameNombreApellidosInteresado($pnregpgv,$pnombre,$papellido1,$papellido2);
		$objDatos->setValue('porcentaje_reduccion','');
		$objDatos->setValue('observaciones','');
		$pmensaje="No se puede calcular el porcentaje de reducción por no disponer del nº de horas trabajadas en
                                    el puesto que ocupa ".$pnombre." ".$papellido1." ".$papellido2.".";
		$this->showMensaje('APL-GENERICO-AVISO',array($pmensaje));
		return -1;
	} else{
		$pporcentajeReducion=round((round($pporcentajeReducion * 100)) / 100);

	}

	$objDatos->setValue('porcentaje',$pporcentajeReducion);//campotexto

	if (substr($pobservaciones,0,9)=="Reducción"){
		$l_cadena=strlen($pobservaciones);
		$pobservaciones=substr($pobservaciones,35,$l_cadena);
		$pobservaciones="Reducción de ".$v_horas." horas y ".$v_minutos." minutos. ".$pobservaciones;
		$objDatos->setValue('observaciones',$pobservaciones);
	}else {
		$pobservaciones="Reducción de ".$v_horas." horas y ".$v_minutos." minutos. ".$pobservaciones;
		$objDatos->setValue('observaciones',$pobservaciones);
	}

	return 0;
}



function modificarFechaNacimientoMenor($objDatos){
	// Si cambia el valor fecha de nacimiento del menor para RJG, se actualiza la fechaFin
	$pfechaNacimientoMenor = $objDatos->getValue('fechaNaci');
	$var_FechaFin=Permisos::f_obtieneFechaFin($pfechaNacimientoMenor);
	$objDatos->setValue('fechaFin',$var_FechaFin);


}



function modificarGradoParentesco($objDatos){
	$motivo = $objDatos->getValue('motivo');
	$pcclase= substr($motivo,0,strpos($motivo,'|'));
	if ($pcclase=='EGP' or $pcclase=='DEF'){
		$pdistancia=$objDatos->getValue('distancia');
		$pgradoParentesco=$objDatos->getValue('gradoParentesco');
		Solicitudes::f_dameDiasMaximoEGPoDEF($pdistancia,$pgradoParentesco,$pdiasMaximo);

	}

	if(!is_null($pdiasMaximo)OR $pdiasMaximo!='' ){
		//ENTONCES Mostrar pdiasMaximo;
		$objDatos->setVisible('maximo',true);//campotexto
		$objDatos->setValue('maximo',$pdiasMaximo);//campotexto
	}

	return 0;
}
```

```
function modificarDistancia($objDatos){

          $motivo = $objDatos->getValue('motivo');
          $pcclase= substr($motivo,0,strpos($motivo,'|'));
          if ($pcclase=='EGP' or $pcclase=='DEF'){
                    $pdistancia=$objDatos->getValue('distancia');
                    $pgradoParentesco=$objDatos->getValue('gradoParentesco');
                    Solicitudes::f_dameDiasMaximoEGPoDEF($pdistancia,$pgradoParentesco,$pdiasMaximo);
          }
          elseif ($pcclase=='TDO'){
                    $pdistancia=$objDatos->getValue('distancia');
                    $pgradoParentesco=$objDatos->getValue('gradoParentesco');
                    Solicitudes::f_dameDiasMaximoMFAoMUHoTDO($pcclase,$pdistancia,$pdiasMaximo);
          }
          elseif ($pcclase=='MFA' or $pcclase=='MUH'){
                    $pdistancia=$objDatos->getValue('distanciaMFAoMUH');
                    Solicitudes::f_dameDiasMaximoMFAoMUHoTDO($pcclase,$pdistancia,$pdiasMaximo);
          }

          if(!is_null($pdiasMaximo)OR $pdiasMaximo!='' ){
                    //ENTONCES Mostrar pdiasMaximo;
                    $objDatos->setVisible('maximo',true);//campotexto
                    $objDatos->setValue('maximo',$pdiasMaximo);//campotexto
          }

          return 0;
}



function limpiaPuertos($objDatos){
          $objDatos->setValue('Nombre','');
          $objDatos->setValue('DNI','');
          return 0;
}



          ///Pone fecha fin al mismo valor que fecha inicio
function actualizaFechaFin($objDatos){
          $campoDisparador = $objDatos->getTriggerField();
          if($campoDisparador=='fechaIni'){
                    $objDatos->setValue('fechaFin',$objDatos->getValue('fechaIni'));
          }
          return 0;
}



//Validación DNI del interesado
function validaDNI($objDatos){


              $nif = $objDatos->getValue('DNI');
              if($nif!=''){
                   $depuertos = $objDatos->getValue('depuertos');
                   $res = $this->consultar( "SELECT nregpgv, dapell1 ||' '|| dapell2 ||', '|| dnombre as \"Nombre\" FROM " .
                            "vper_ocupacion WHERE nregpgv like '".$nif."__' and vper_ocupacion.depuertos='".$depuertos."'");
                   if(count($res)>0){
                       $objDatos->setValue('Nombre',$res[0]['Nombre']);
                       //$objDatos->setValue('nregpgv',$res[0]['nregpgv']);
                   }
                   else {
                       $objDatos->setValue('Nombre','');
                       //$objDatos->setValue('nregpgv','');
                       $objDatos->setValue('DNI','');
                       $this->showMensaje('APL-01',array($nif));
                       return -1;
                   }

                  // Si no estamos pidiendo un IT, llamamos a modificarClasePermiso. Cuando es una IT no hace falta, porque
                   //                                                  sino borrarémos el DNI y nombre de pantalla
                   $motivo = $objDatos->getValue('motivo');
                              $pcclase= substr($motivo,0,strpos($motivo,'|'));

                   if ($pcclase!='IT')

                             $this->modificarClasePermiso($objDatos);

               }
               else{
                   $objDatos->setValue('Nombre','');

                   $objDatos->setValue('DNI','');
               }
              return 0;
} //Fin de validaDNI


function modificarClasePermiso($objDatos){

          $nif = $objDatos->getValue('DNI');
          $periodo=$objDatos->getValue('periodo');
          $motivo = $objDatos->getValue('motivo');
          $pcclase= substr($motivo,0,strpos($motivo,'|'));

          //Fija la visibilidad de los campos según el motivo
          //Todos ocultos y limpios
          $objDatos->setVisible('fechaNaci',false);//campotexto
          $objDatos->setVisible('tipo',false);//lista
```

```
$objDatos->setVisible('causa',false);//lista
$objDatos->setVisible('horas',false);//campotexto  // #6342
$objDatos->setVisible('porcentaje',false);//campotexto
$objDatos->setVisible('incidencia',false);//check
$objDatos->setVisible('distancia',false);//lista
$objDatos->setVisible('distanciaMFAoMUH',false);//lista
$objDatos->setVisible('gradoParentesco',false);//lista
$objDatos->setVisible('minimo',false);//campotexto
$objDatos->setVisible('maximo',false);//campotexto
$objDatos->setVisible('unidad',false);//campotexto
$objDatos->setVisible('reqJustificante',true);//campotexto
$objDatos->setVisible('indicarFamiliar',false);//campotexto
$objDatos->setVisible('AltaHojaRosa',false);//check
$objDatos->setVisible('aportaJustificante',false);//lista radio button // 09/04/2010

switch ($pcclase){  // Switch replicado en el método "Volver"
                    case 'IT':
                              $objDatos->setVisible('causa',true);//lista
                              $objDatos->setVisible('incidencia',true);//lista
                              $objDatos->setChecked('incidencia',true);
                              $objDatos->setValue('DNI','');
                              $objDatos->setValue('Nombre','');

                              break;
                    case 'MAD':

                              $objDatos->setVisible('incidencia',true);//lista
                              $objDatos->setChecked('incidencia',true);
                              $objDatos->setVisible('reqJustificante',true);//campotexto
                              $objDatos->setValue('reqJustificante','Requiere justificante');
                              break;
                     case 'RJM':
                              $objDatos->setVisible('tipo',true);//lista tipo minusvalia

                              break;
                     case 'RJP':
                              $objDatos->setVisible('incidencia',true);//lista
                              $objDatos->setChecked('incidencia',true);
                              $objDatos->setVisible('horas',true);//campotexto
                              $objDatos->setVisible('porcentaje',true);//campotexto
                              break;

                     case 'RJC':
                              $objDatos->setVisible('horas',true);//campotexto
                              $objDatos->setVisible('porcentaje',true);//campotexto
                              break;
                     case 'RJ5':
                              $objDatos->setVisible('incidencia',true);//lista
                              $objDatos->setChecked('incidencia',true);
                              break;
                     case 'RJV':
                              $objDatos->setVisible('horas',true);//campotexto
                              $objDatos->setVisible('porcentaje',true);//campotexto
                              break;

                     case 'RJL':
                     case 'AJL':
                     case 'FPO':
                              $objDatos->setVisible('fechaNaci',true);//campotexto
                              break;
                     case 'RJG':
                              $objDatos->setVisible('horas',true);//campotexto
                              $objDatos->setVisible('porcentaje',true);//campotexto
                              $objDatos->setVisible('fechaNaci',true);//campotexto
                              break;
                     case 'EGP':
                              $objDatos->setVisible('distancia',true);//lista
                              $objDatos->setVisible('gradoParentesco',true);//lista
                              $objDatos->setVisible('indicarFamiliar',true);//campotexto
                              $objDatos->setValue('indicarFamiliar','(Indicar familiar en observaciones)');
                              break;
                     case 'MFA':
                              $objDatos->setVisible('distanciaMFAoMUH',true);//lista
                              break;
                     case 'MUH':
                              $objDatos->setVisible('distanciaMFAoMUH',true);//lista
                              break;
                     case 'TDO':
                              $objDatos->setVisible('distancia',true);//lista
                              break;
                     case 'DEF':
                              $objDatos->setVisible('distancia',true);//lista
                              $objDatos->setVisible('gradoParentesco',true);//lista
                              $objDatos->setVisible('indicarFamiliar',true);//campotexto
                              $objDatos->setValue('indicarFamiliar','(Indicar familiar en observaciones)');
                              break;
                     case 'ENC':
                              $objDatos->setVisible('aportaJustificante',true);//lista radio button
                              break;
                    default:
                              $objDatos->setVisible('observaciones',true);//textArea
                              break;
}

//Validacion del periodo:

if(IgepSession::dameRol()!='P_TRAMITA'){

          $periodo= date('Y');
          $fechactual = new gvHidraTimestamp();
          $fechactual->setTime(0,0,0);
          //Hasta el 15 de enero
```

```
		$f1 = new gvHidraTimestamp("01/16/".date('Y'));
		$f1->setTime(0,0,0);

		if ($fechactual<$f1) {

			$listaPeriodo = $objDatos->getLista('periodo');
			$listaPeriodo->clean();
			//La lista desplegable del periodo se cargará con el año actual (valor por defecto)
			$listaPeriodo->addOpcion($periodo,$periodo);
			//el anterior si la fecha actual es hasta el 15 de enero y la clase de permiso es 'VV6' o 'AAP'
			if ($pcclase == 'VV6' or $pcclase == 'AAP')
				$listaPeriodo->addOpcion($periodo-1,$periodo-1);
			//y el siguiente durante el mes de diciembre.
			$f2 = new gvHidraTimestamp("12/01/".date('Y'));
			$f2->setTime(0,0,0);
			$f3 = new gvHidraTimestamp("12/31/".date('Y'));
			$f3->setTime(0,0,0);
			if((gvHidraTimestamp::cmp($fechactual,$f2)<=0)and(gvHidraTimestamp::cmp($fechactual,$f3)>=0))
				$listaPeriodo->addOpcion($periodo+1,$periodo+1);
			$objDatos->setLista('periodo',$listaPeriodo);
		}
}

if(IgepSession::dameRol()=='P_TRAMITA'){

		$periodo= date('Y');
		$fechactual = new gvHidraTimestamp();
		$fechactual->setTime(0,0,0);
		//Hasta el 15 de enero
		$f1 = new gvHidraTimestamp("02/01/".date('Y'));
		$f1->setTime(0,0,0);

		if ($fechactual<$f1) {
			$listaPeriodo = $objDatos->getLista('periodo');
			$listaPeriodo->clean();
			//La lista desplegable del periodo se cargará con el año actual (valor por defecto)
			$listaPeriodo->addOpcion($periodo,$periodo);
			//el anterior si la fecha actual es hasta el 31 de enero
			$listaPeriodo->addOpcion($periodo-1,$periodo-1);
			//y el siguiente
			$listaPeriodo->addOpcion($periodo+1,$periodo+1);
			$objDatos->setLista('periodo',$listaPeriodo);

		}

		$retorno=ClasesDePermisos::f_esSolicitable($pcclase);

		// Si el motivo es SOLICITABLE y es TRAMITADOR, aparecerá el check "Alta papel"
		if($retorno['0']=='1'){
			$objDatos->setVisible('AltaHojaRosa',true);//lista
			$objDatos->setChecked('AltaHojaRosa',false);
		}
}



//Obtenemos datos
//Si volvemos de un salto estamos en otro tipo de operacion
if(is_null($objDatos->getValue('DNI',actualizar))){
		$interesado = $objDatos->getValue('DNI',seleccionar);
		if($interesado==''){$this->showMensaje('APL-48');return -1;}
		// 5/1/2010 INICIO : No tomamos el periodo anterior ya que se pone siempre el actual por defecto
		//	$periodo = $objDatos->getValue('periodo',seleccionar);

}else{
		$interesado = $objDatos->getValue('DNI',actualizar);
		if($interesado==''){$this->showMensaje('APL-48');return -1;}
		 //  No tomamos el periodo anterior ya que se pone siempre el actual por defecto
}

//Pasamos DNI a nregpgv
$cons="SELECT nregpgv FROM vper_ocupacion where nregpgv like '".$interesado."__'";
$res=$this->consultar($cons);
$interesado=$res['0']['nregpgv'];


$pnregpgv=$interesado;
$pperiodo=$periodo;


//Obtener la información del permiso
PermisosValidacion::f_LeerMinMaxDias($pcclase, $pdiasMinimo, $pdiasMaximo, $pdiasNaturales, $punidad);

if($pcclase == 'VAC' || $pcclase == 'ADI' || $pcclase == 'VV6' || $pcclase == 'AAP'){

		if ($pperiodo==date("Y")){
			$accion='P';
		}else if ($pperiodo==(date("Y")-1)){
			$accion='A';
		}
		//Obtenemos los totales de VAC, ADI, VV6 y AAP que le corresponden, y el año para el que se conocen los
                                                                                                                ADI

		$retorno= PermisosValidacion::f_calcularTotalDiasVACyADIyVV6yAAP($pnregpgv, $pperiodo,$totaldiasVAC,

                                        $totaldiasADI, $anyodiasADI, $totaldiasVV6, $totaldiasAAP,$accion);
		if($retorno<0){
			IgepDebug::setDebug(DEBUG_USER,'Error');
			return -1;
		}
		if ($totaldiasADI=='?'){
			$totaldiasADI=0;
```

```
                }

                if($pcclase == 'VAC')
                        $pdiasMaximo = $totaldiasVAC;
                else
                        if($pcclase == 'ADI')
                                if($anyodiasADI == $pperiodo)
                                        $pdiasMaximo = $totaldiasADI;
                                else
                                        $pdiasMaximo = 999999;
                        else
                                if($pcclase == 'VV6')
                                        $pdiasMaximo = $totaldiasVV6;
                                else
                                        if ($pcclase == 'AAP')
                                                $pdiasMaximo = $totaldiasAAP;
        }


        //Cuando se pide EGP o DEF, tenemos que calcular $pdiasMaximo mediante la función f_dameDiasMaximoEGPoDEF
        if ($pcclase=='EGP' || $pcclase=='DEF'){
                $pdistancia='C';
                $pgradoParentesco='2';
                $objDatos->setSelected('distancia',$pdistancia);
                Solicitudes::f_dameDiasMaximoEGPoDEF($pdistancia,$pgradoParentesco,$pdiasMaximo);

        }

        if ($pcclase=='MFA' || $pcclase=='MUH' || $pcclase=='TDO'){
                $pdistancia='C';
                if ($pcclase=='MFA' || $pcclase=='MUH'){
                        $objDatos->setSelected('distanciaMFAoMUH',$pdistancia);
                }
                else{
                        $objDatos->setSelected('distancia',$pdistancia);
                }
                Solicitudes::f_dameDiasMaximoMFAoMUHoTDO($pcclase,$pdistancia,$pdiasMaximo);
        }

        ClasesDePermisos::f_dameDatosClasePermiso($pcclase, $pdclase, $pjustificante);


        if($punidad == 'D')
                //ENTONCES mostrar "Días"
                $unidad='Días';
        else
                //SINO mostrar "Meses"
                $unidad='Meses';

        //Pongo el valor al campo unidad
        $objDatos->setValue('unidad','Días');

        if($pdiasNaturales == 'S')
                //ENTONCES mostrar " naturales"
                $objDatos->setValue('unidad',"$unidad naturales");
        else
                //SINO mostrar " hábiles"
                $objDatos->setValue('unidad',"$unidad hábiles");


        // Mostramos la información del permiso
        if($pjustificante == 'S' || $pcclase == 'ENC')
                //ENTONCES Mostrar "Requiere justificante"
                $objDatos->setValue('reqJustificante','Requiere justificante');
        else
                //SINO Mostrar "No requiere justificante"
                $objDatos->setValue('reqJustificante','No requiere justificante');



        //Reajuste y visibilidad de  max y min cuando son vacios
        if($pdiasMinimo!=0 and ($pdiasMaximo!=999999 AND $pdiasMaximo!=0) ){
                $objDatos->setVisible('minimo',true);//campotexto
                $objDatos->setVisible('maximo',true);//campotexto
                $objDatos->setVisible('unidad',true);//campotexto
        }else{
                if($pdiasMinimo==0 and ($pdiasMaximo!=999999 AND $pdiasMaximo!=0) ){
                        $objDatos->setVisible('maximo',true);//campotexto
                        $objDatos->setVisible('unidad',true);//campotexto
                }
                if($pdiasMinimo!=0 and ($pdiasMaximo==999999 OR $pdiasMaximo==0) ){
                        $objDatos->setVisible('minimo',true);//campotexto
                        $objDatos->setVisible('unidad',true);//campotexto
                }
        }

        //Pongo los valores de max y min
        $objDatos->setValue('minimo',$pdiasMinimo);//campotexto
        $objDatos->setValue('maximo',$pdiasMaximo);//campotexto

        return 0;
} //Fin de modificarClasePermiso



function modificarPeriodo($objDatos){

        $nif = $objDatos->getValue('DNI');
```

```
		$periodo=$objDatos->getValue('periodo');

		$motivo = $objDatos->getValue('motivo');
		$pcclase= substr($motivo,0,strpos($motivo,'|'));


		//Obtenemos datos
		//Si volvemos de un salto estamos en otro tipo de operacion
		if(is_null($objDatos->getValue('DNI',actualizar))){
				$interesado = $objDatos->getValue('DNI',seleccionar);
				if($interesado==''){$this->showMensaje('APL-48');return -1;}
				$periodo = $objDatos->getValue('periodo',seleccionar);
		}else{
				$interesado = $objDatos->getValue('DNI',actualizar);
				if($interesado==''){$this->showMensaje('APL-48');return -1;}
				$periodo = $objDatos->getValue('periodo',actualizar);
		}

		//Pasamos DNI a nregpgv
		$cons="SELECT nregpgv FROM vper_ocupacion where nregpgv like '".$interesado."__'";
		$res=$this->consultar($cons);
		$interesado=$res['0']['nregpgv'];



		$pnregpgv=$interesado;
		$pperiodo=$periodo;

		//Obtener la informaciÃ³n del permiso
		PermisosValidacion::f_LeerMinMaxDias($pcclase, $pdiasMinimo, $pdiasMaximo, $pdiasNaturales, $punidad);

		if($pcclase == 'VAC' || $pcclase == 'ADI' || $pcclase == 'VV6' || $pcclase == 'AAP'){

			if ($pperiodo==date("Y")){
				$accion='P';
			}else if ($pperiodo==(date("Y")-1)){
				$accion='A';
			}
			//Obtenemos los totales de VAC, ADI, VV6 y AAP que le corresponden,y el año para el que se conocen los ADI

			$retorno= PermisosValidacion::f_calcularTotalDiasVACyADIyVV6yAAP($pnregpgv, $pperiodo,$totaldiasVAC,

									$totaldiasADI, $anyodiasADI, $totaldiasVV6, $totaldiasAAP,$accion);
			if($retorno<0){
				IgepDebug::setDebug(DEBUG_USER,'Error');
				return -1;
			}
			if ($totaldiasADI=='?'){
				$totaldiasADI=0;
			}



			if($pcclase == 'VAC')
				$pdiasMaximo = $totaldiasVAC;
			else
				if($pcclase == 'ADI')
					if($anyodiasADI == $pperiodo)
						$pdiasMaximo = $totaldiasADI;
					else
						$pdiasMaximo = 999999;
				else
					if($pcclase == 'VV6')
						$pdiasMaximo = $totaldiasVV6;
					else
						if ($pcclase == 'AAP')
							$pdiasMaximo = $totaldiasAAP;
			}

			// IF replicado en el método "Volver"
			//Reajuste y visibilidad de  max y min cuando son vacios
			if($pdiasMinimo!=0 and ($pdiasMaximo!=999999 AND $pdiasMaximo!=0) ){
				$objDatos->setVisible('minimo',true);//campotexto
				$objDatos->setVisible('maximo',true);//campotexto
				$objDatos->setVisible('unidad',true);//campotexto
			}else{
				if($pdiasMinimo==0 and ($pdiasMaximo!=999999 AND $pdiasMaximo!=0) ){
					$objDatos->setVisible('maximo',true);//campotexto
					$objDatos->setVisible('unidad',true);//campotexto
				}
				if($pdiasMinimo!=0 and ($pdiasMaximo==999999 OR $pdiasMaximo==0) ){
					$objDatos->setVisible('minimo',true);//campotexto
					$objDatos->setVisible('unidad',true);//campotexto
				}
				if($pdiasMinimo==0 and ($pdiasMaximo==999999)){
					$objDatos->setVisible('minimo',false);//campotexto
					$objDatos->setVisible('maximo',false);//campotexto
					$objDatos->setVisible('unidad',false);//campotexto
				}
			}

			//Pongo los valores de max y min
			 $objDatos->setValue('minimo',$pdiasMinimo);//campotexto
			 $objDatos->setValue('maximo',$pdiasMaximo);//campotexto

		return 0;
} //Fin de modificarPeriodo


function validaFechas($objDatos){

		$fini = $objDatos->getValue('fechaIni');
```

```
		$ffin = $objDatos->getValue('fechaFin');
		if(empty($ffin))return 0;
		//Fecha fin no puede ser menor q fecha inicio
		if (gvHidraTimestamp::cmp($fini, $ffin) < 0){
			$this->showMensaje('APL-03');
			return -1;
		}
		return 0;
}//Fin validaFechas



		/* -------------------------FIN  ACCIONES DE INTERFAZ -------------------------  */


function preBuscar($objDatos){
		//Borramos panel de salto por si volvemos de CANCELAR
		IgepSession::borraPanel('saltoIgep');
		//Lanzamos impresiÃ³n si existe
		if (IgepSession::existeVariable("AltaSolicPermisosConfirmar","listadoSolicitudesPermisosYLicencias")){
			$actionForwardFork = $objDatos->getForward('mostrarSolicitud');
			$this->openWindow($actionForwardFork);
		}
		else
			IgepSession::borraPanel('AltaSolicPermisosConfirmar');
			return 0;
		}


function postBuscar($objDatos){

		$visibilidad="false";
		$objDatos->setValue('minimoVisible',$visibilidad);
		$objDatos->setValue('maximoVisible',$visibilidad);
		$objDatos->setValue('reqJustificanteVisible',$visibilidad);
		$objDatos->setValue('indicarFamiliarVisible',$visibilidad);
		$objDatos->setValue('unidadVisible',$visibilidad);
		$objDatos->setValue('fechaNaciVisible',$visibilidad);
		$objDatos->setValue('horasVisible',$visibilidad);
		$objDatos->setValue('porcentajeVisible',$visibilidad);


		// Listas
		$objDatos->setValue('distanciaVisible',$visibilidad);
		$objDatos->setValue('distanciaMFAoMUHVisible',$visibilidad);
		$objDatos->setValue('gradoParentescoVisible',$visibilidad);
		$objDatos->setValue('tipoVisible',$visibilidad);
		$objDatos->setValue('causaVisible',$visibilidad);
		$objDatos->setValue('incidenciaVisible',$visibilidad);
		$objDatos->setValue('AltaHojaRosaVisible',$visibilidad);
		$objDatos->setValue('aportaJustificanteVisible',$visibilidad);//09/04/2010
		return 0;
}

		/* -------------------------SALTO -------------------------  */

function accionesParticulares($str_accion, $objDatos) {
			switch($str_accion){
				case "volver":
					$actionForward = $this->volver($objDatos);
					break;
				case "limpiar":
					$actionForward = $this->limpiar($objDatos);
					break;

			}
			return $actionForward;
}



function volver($objDatos){

			//Cargamos los datos que teniamos antes
			$objSalto = IgepSession::damePanel('saltoIgep');
			$parametros = $objSalto->getParams();
			$datos = $parametros['arrayParams'];
			$datosAct= $datos;
			IgepDebug::setDebug(DEBUG_USER,'INI volver(): <pre>'.print_r($datos,true).'</pre>');
			IgepSession::borraPanel('AltaSolicPermisosConfirmar');
			//Añadimos las listas sacadas de objDatos
			$datosAct['0']['motivo'] = $objDatos->datosPreinsertados['motivo'];
			$datosAct['0']['motivo']['seleccionado'] = $datos['0']['motivo'];

			$datosAct['0']['periodo'] = $objDatos->datosPreinsertados['periodo'];
			$datosAct['0']['periodo']['seleccionado'] = $datos['0']['periodo'];

			$datosAct['0']['tipo'] = $objDatos->datosPreinsertados['tipo'];
			$datosAct['0']['tipo']['seleccionado'] = $datos['0']['tipo'];

			$datosAct['0']['causa'] = $objDatos->datosPreinsertados['causa'];
			$datosAct['0']['causa']['seleccionado'] = $datos['0']['causa'];

			$datosAct['0']['distancia'] = $objDatos->datosPreinsertados['distancia'];
			$datosAct['0']['distancia']['seleccionado'] = $datos['0']['distancia'];

			$datosAct['0']['distanciaMFAoMUH'] = $objDatos->datosPreinsertados['distanciaMFAoMUH'];
			$datosAct['0']['distanciaMFAoMUH']['seleccionado'] = $datos['0']['distanciaMFAoMUH'];

			$datosAct['0']['gradoParentesco'] = $objDatos->datosPreinsertados['gradoParentesco'];
```

```
$datosAct['0']['gradoParentesco']['seleccionado'] = $datos['0']['gradoParentesco'];

$datosAct['0']['aportaJustificante'] = $objDatos->datosPreinsertados['aportaJustificante'];//09/04/2010
$datosAct['0']['aportaJustificante']['seleccionado'] = $datos['0']['aportaJustificante'];//09/04/2010


//Ponemos los campos visibles

$datosAct['0']['reqJustificanteVisible']='false';//campotexto
$datosAct['0']['indicarFamiliarVisible']='false';//campotexto
$datosAct['0']['minimoVisible']='false';//campotexto
$datosAct['0']['maximoVisible']='false';//campotexto
$datosAct['0']['unidadVisible']='false';//campotexto
$datosAct['0']['horasVisible']='false';//campotexto
$datosAct['0']['porcentajeVisible']='false';//campotexto
$datosAct['0']['fechaNaciVisible']='false';//campotexto

$datosAct['0']['causaVisible']='false';//lista
$datosAct['0']['incidenciaVisible']='false';//lista
$datosAct['0']['AltaHojaRosaVisible']='false';//lista
$datosAct['0']['tipoVisible']='false';//lista tipo
$datosAct['0']['distanciaVisible']='false';//lista
$datosAct['0']['distanciaMFAoMUHVisible']='false';//lista
$datosAct['0']['gradoParentescoVisible']='false';//lista
$datosAct['0']['aportaJustificanteVisible']='false';//lista 09/04/2010
$pcclase=substr($datos['0']['motivo'],0,strpos($datos['0']['motivo'],'|'));
  switch ($pcclase){
          case 'IT':
                    $datosAct['0']['causaVisible']='true';//lista
                    $datosAct['0']['incidenciaVisible']='true';//lista
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    break;
          case 'MAD':
                    $datosAct['0']['incidenciaVisible']='true';//lista
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    break;
           case 'RJM':
                    $datosAct['0']['tipoVisible']='true';//lista tipo minusvalia
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    break;
           case 'RJP':
                    $datosAct['0']['incidenciaVisible']='true';//lista
                    $datosAct['0']['horasVisible']='true';//campotexto
                    $datosAct['0']['porcentajeVisible']='true';//campotexto
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    break;
           case 'RJC':
                    $datosAct['0']['horasVisible']='true';//campotexto
                    $datosAct['0']['porcentajeVisible']='true';//campotexto
                    break;
           case 'RJ5':
                    $datosAct['0']['incidenciaVisible']='true';//lista
                    break;
           case 'RJV':
                    $datosAct['0']['horasVisible']='true';//campotexto
                    $datosAct['0']['porcentajeVisible']='true';//campotexto
                    break;
           case 'RJL':
           case 'AJL':
           case 'FPO':
                    $datosAct['0']['fechaNaciVisible']='true';//campotexto
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    break;
           case 'RJG':
                    $datosAct['0']['horasVisible']='true';//campotexto
                    $datosAct['0']['porcentajeVisible']='true';//campotexto
                    $datosAct['0']['fechaNaciVisible']='true';//campotexto
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    break;
           case 'EXA':
           case 'RJE':
           case 'VV6':
           case 'AAP':
           case 'ADI':
           case 'VAC':
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    break;
           case 'EGP':
                    $datosAct['0']['distanciaVisible']='true';//lista
                    $datosAct['0']['gradoParentescoVisible']='true';//lista
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    $datosAct['0']['indicarFamiliarVisible']='true';//campotexto
                    break;
           case 'MFA':
                    $datosAct['0']['distanciaMFAoMUHVisible']='true';//lista
                    break;
           case 'MUH':
                    $datosAct['0']['distanciaMFAoMUHVisible']='true';//lista
                    break;
           case 'DEF':
                    $datosAct['0']['distanciaVisible']='true';//lista
                    $datosAct['0']['gradoParentescoVisible']='true';//lista
                    $datosAct['0']['reqJustificanteVisible']='true';//campotexto
                    $datosAct['0']['indicarFamiliarVisible']='true';//campotexto
                    break;
           case 'ENC':
                    $datosAct['0']['aportaJustificanteVisible']='true';//lista radio button
                    break;
           case 'TDO':
                    $datosAct['0']['distanciaVisible']='true';//lista
                    break;
          default:
```

```
                                        break;
                    }


                    if(IgepSession::dameRol()=='P_TRAMITA'){
                              $retorno=ClasesDePermisos::f_esSolicitable($pcclase);
                              if($retorno['0']=='1'){
                                        $datosAct['0']['AltaHojaRosaVisible']='true';//lista
                              }
                    }







                    //Reajuste y visibilidad de  max y min cuando son vacios
                    $pdiasMaximo = $datos[0]['maximo'];
                    $pdiasMinimo = $datos[0]['minimo'];

                    if($pdiasMinimo!=0 and ($pdiasMaximo!=999999 AND $pdiasMaximo!=0) ){
                              $datosAct['0']['minimoVisible']='true';//campotexto
                              $datosAct['0']['maximoVisible']='true';//campotexto
                              $datosAct['0']['unidadVisible']='true';//campotexto
                    }else{
                              if($pdiasMinimo==0 and ($pdiasMaximo!=999999 AND $pdiasMaximo!=0) ){
                                        $datosAct['0']['maximoVisible']='true';//campotexto
                                        $datosAct['0']['unidadVisible']='true';//campotexto
                              }
                              if($pdiasMinimo!=0 and ($pdiasMaximo==999999 OR $pdiasMaximo==0) ){
                                        $datosAct['0']['minimoVisible']='true';//campotexto
                                        $datosAct['0']['unidadVisible']='true';//campotexto
                              }
                    }

                    //Vuelve a hacer la búsqueda y pone los datos artificiales creados antes
                    $this->refreshSearch();
                    $this->setResultadoBusqueda($datosAct);

                    IgepDebug::setDebug(DEBUG_USER,'FIN volver(): <pre>'.print_r($datosAct,true).'</pre>');
                    $actionForward = $objDatos->getForward('correcto');
                    return $actionForward;
}




function saltoDeVentana($objDatos, $objSalto){ //confirmar()
          //Limpiamos variable para poder guardar y imprimir/guardar
          if (IgepSession::existeVariable("AltaSolicPermisosConfirmar","listadoSolicitudesPermisosYLicencias")){
                    IgepSession::borraVariable("AltaSolicPermisosConfirmar","listadoSolicitudesPermisosYLicencias");
          }

          /*Validaciones*/
          $fechaNaci=null;
          //Si volvemos de un salto estamos en otro tipo de operacion
          if(is_null($objDatos->getValue('DNI',actualizar))){
                    $operacion='seleccionar';
                    $interesado = $objDatos->getValue('DNI',seleccionar);
                    if($interesado==''){$this->showMensaje('APL-48');return -1;}
                    $periodo = $objDatos->getValue('periodo',seleccionar);
                    $pgradoParentesco = $objDatos->getValue('gradoParentesco',seleccionar);
                    $fechaIni = $objDatos->getValue('fechaIni',seleccionar);
                    $fechaFin = $objDatos->getValue('fechaFin',seleccionar);
                    $motivo = $objDatos->getValue('motivo',seleccionar);
                    if($motivo==''){$this->showMensaje('APL-04');return -1;}
                    $idMotivo = substr($motivo,0,strpos($motivo,'|'));
                    $descMotivo = substr($motivo,strpos($motivo,'|')+1);
                    if(empty($fechaNaci)AND($idMotivo=='RJL' OR $idMotivo=='AJL' OR $idMotivo=='RJP' OR $idMotivo=='RJG')){
                              $this->showMensaje('APL-41');return -1;
                    }
                    if($idMotivo=='MFA' OR $idMotivo=='MUH' ){
                              $pdistancia = $objDatos->getValue('distanciaMFAoMUH',seleccionar);
                    }
                    else{
                              $pdistancia = $objDatos->getValue('distancia',seleccionar);
                    }
                    $causa = $objDatos->getValue('causa',seleccionar);
                    $tipoMinus = $objDatos->getValue('tipo',seleccionar);
                    $var_horas=$objDatos->getValue('horas',seleccionar);
                    $porcentaje= $objDatos->getValue('porcentaje',seleccionar);
                    $generarIncidencia=$objDatos->getValue('incidencia',seleccionar);
                    $AltaHojaRosa=$objDatos->getValue('AltaHojaRosa',seleccionar);
                    $indicarFamiliar = $objDatos->getValue('indicarFamiliar',seleccionar);
                    $observaciones = $objDatos->getValue('observaciones',seleccionar);
                    $aportaJustificante = $objDatos->getValue('aportaJustificante',seleccionar);

          }else{
                    $operacion='actualizar';
                    $interesado = $objDatos->getValue('DNI',actualizar);
                    if($interesado==''){$this->showMensaje('APL-48');return -1;}
                    $periodo = $objDatos->getValue('periodo',actualizar);
                    $pgradoParentesco = $objDatos->getValue('gradoParentesco',actualizar);
                    $fechaIni = $objDatos->getValue('fechaIni',actualizar);
                    $fechaFin = $objDatos->getValue('fechaFin',actualizar);
                    $motivo = $objDatos->getValue('motivo',actualizar);
```

```
		if($motivo==''){$this->showMensaje('APL-04');return -1;}
		$idMotivo = substr($motivo,0,strpos($motivo,'|'));
		$descMotivo = substr($motivo,strpos($motivo,'|')+1);
		$fechaNaci = $objDatos->getValue('fechaNaci',actualizar);
		if(empty($fechaNaci)AND($idMotivo=='RJL' OR $idMotivo=='AJL' OR $idMotivo=='RJG')){
			$this->showMensaje('APL-41');return -1;
		}
		if($idMotivo=='MFA' OR $idMotivo=='MUH'){
			$pdistancia = $objDatos->getValue('distanciaMFAoMUH',actualizar);
		}
		else{
			$pdistancia = $objDatos->getValue('distancia',actualizar);
		}
		$causa = $objDatos->getValue('causa',actualizar);
		$tipoMinus = $objDatos->getValue('tipo',actualizar);
		$var_horas=$objDatos->getValue('horas',actualizar);
		$porcentaje= $objDatos->getValue('porcentaje',actualizar);
		$generarIncidencia=$objDatos->getValue('incidencia',actualizar);
		$AltaHojaRosa=$objDatos->getValue('AltaHojaRosa',actualizar);
		$indicarFamiliar = $objDatos->getValue('indicarFamiliar',actualizar);
		$observaciones = $objDatos->getValue('observaciones',actualizar);
		$aportaJustificante = $objDatos->getValue('aportaJustificante',actualizar);//09/04/2010
}


IgepDebug::setDebug(DEBUG_USER,"Incidencia".print_r($generarIncidencia,true));



//Pasamos DNI a nregpgv
$cons="SELECT nregpgv FROM vper_ocupacion where nregpgv like '".$interesado."__'";
$res=$this->consultar($cons);
$interesado=$res['0']['nregpgv'];


//Fecha fin no puede ser menor q fecha inicio si no es nula
if(!empty($fechaFin)){
		if (gvHidraTimestamp::cmp($fechaIni, $fechaFin) < 0 ){
			$this->showMensaje('APL-03');
			return -1;
		}
}
//Fecha Ini es obligatoria
if(empty($fechaIni)){
		$this->showMensaje('APL-39');
		return -1;

}



//confirmar()
$altaModificada ='N';
$finReduc='N';
//Fin confirmar()

$tipo = ClasesDePermisos::f_esSolicitable($idMotivo);
if($tipo['0']<0){
		$this->showMensaje('APL-GENERICO',array($tipo['error']));
		return -1;
}



if(($tipo['0']==1) and $AltaHojaRosa=='N'){
		$tipo = 'S';//Solicitud
		// se añade &$aportaJustificante como ultimo parámetro en f_altaSolicitud

$datosSoliPer=Solicitudes::f_altaSolicitud($interesado,$periodo,$fechaIni,$fechaFin,$observaciones,$fechaNaci,

			$altaModificada,$estado='PR',$ffirmaResp=null,$FirmadoPorALR=null,

			$ffirmaPer=null,$FirmadaPorALP=null,$psituacion='A',

			$totalDias,$totalDiasLaboVac,$totalDiasLabo,$var_dias_naturales,

			$var_dias_habiles, $resultado,$idMotivo,$pdistancia,$listaSolicitudesAnulacion,
			$listaSolicitudesAnulacionResueltas,$pgradoParentesco,$aportaJustificante);


}else{//tipo = 0
		$tipo = 'P';//Permiso


$datosSoliPer=Permisos::f_altaPermiso($interesado,$idMotivo,$periodo,$fechaIni,$fechaFin,$observaciones,$causa,

		$tipoMinus,$porcentaje,$generars,$finReduc,$fechaNaci,$tipo,

		$resultado,$var_horas);


}

if($datosSoliPer<0){
		//En $resultado tendremos el mensaje de error tipo P
		$this->showMensaje('APL-GENERICO',$resultado['mensaje']==''?array("Error no
                                                                controlado"):array($resultado['mensaje']));
		IgepDebug::setDebug(DEBUG_USER,'saltoDeVentana sale con mensaje');
		return -1;
```

```
        }



        //Enviamos todos los datos al panel confirmar
        $operacion='visibles';
        $datosSalto = $objDatos->getAllTuplas($operacion);

        //nuevo
        $datosSalto['0']['resultado'] = $resultado;
        //nuevo
        $datosSalto['0']['tipoAlta'] = $tipo;
        $datosSalto['0']['AltaHojaRosa'] = $AltaHojaRosa;
        $datosSalto['0']['totalDias'] = $totalDias;
        $datosSalto['0']['totalDiasLaboVac'] = $totalDiasLaboVac;
        $datosSalto['0']['totalDiasLabo'] = $totalDiasLabo;
        $datosSalto['0']['var_dias_naturales'] = $var_dias_naturales;
        $datosSalto['0']['var_dias_habiles'] = $var_dias_habiles;

        $datosSalto['0']['listaSolicitudesAnulacion'] = $listaSolicitudesAnulacion;
        $datosSalto['0']['listaSolicitudesAnulacionResueltas'] = $listaSolicitudesAnulacionResueltas;
        //Mezclamos con los datos que vienen de negocio alta,validar, leer_validacion,etc...
        $datosSalto['0']['datosSoliPer']=$datosSoliPer;



        if($objSalto->getId()=='saltoConfirmar'){
                $objSalto->setClase('AltaSolicPermisosConfirmar');
                $objSalto->setAccion('buscar');
                //Enviamos todo los datos del panel
                $objSalto->setParam('arrayParams',$datosSalto);
                //Configuramos la acciÃ³n de retorno
                $objSalto->setAccionRetorno('volver');
                return 0;
         }
         return -1;
}
        /* ------------------------- FIN SALTO -------------------------  */



}
?>
```

Tpl – Alta de solicitudes y permisos

```
{CWVentana tipoAviso=$smty_tipoAviso  codAviso=$smty_codError  descBreve=$smty_descBreve  textoAviso=$smty_textoAviso
                                                                                    onLoad=$smty_jsOnLoad}

{CWBarra customTitle=$smty_customTitle usuario=$smty_usuario codigo=$smty_codigo}
          {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}

<!-- ***************** PANEL edi ***********************-->
{CWPanel id="edi" tipoComprobacion="envio" action="operarBD" method="post" estado="on"
                                        claseManejadora="AltaSolicPermisos" accion="modificar"}
{CWBarraSupPanel titulo="Alta de solicitud de permisos y licencias"}

          {CWBotonTooltip imagen="04" titulo="Restaurar valores" action="buscar" }
{/CWBarraSupPanel}
{CWContenedor}
          {CWFichaEdicion id="FichaEdicion" datos=$smty_datosFicha numPagInsertar="1"}
                    {CWFicha}
                              {CWCampoTexto nombre="solicitables" size="1" editable="false" oculto="true"
                                        textoAsociado="Solicitables"}
                              <!--Es tramitador-->
                              {if $smty_acciones eq 1}
                                        {CWCampoTexto nombre="usuNif" mascara="nnnnnnnn" longitudMaxima="8" size="8"
                                                  editable="false" textoAsociado="Usuario"}
                                        {CWCampoTexto nombre="usuNreg"  size="11" editable="false" oculto="true"
                                                  textoAsociado="Oculto"}
                                        {CWCampoTexto nombre="usuNombre" longitudMaxima="60" size="50" editable="false"}

                                        <br><br>
                                        {CWCheckBox nombre="depuertos" dataType=$dataType_AltaSolicPermisos.depuertos
                                                  editable="true" actualizaA="dispara_accion_interfaz"
                                                  textoAsociado="Usuarios de puertos (sustituciones)"}
                                        <br>
                                        {CWCampoTexto nombre="DNI" dataType=$dataType_AltaSolicPermisos.DNI size="8"
                                                  editable="true" textoAsociado="Interesado"
                                                  actualizaA="dispara_accion_interfaz"}
                                        {CWBotonTooltip imagen="13" titulo="Busqueda Interesado" funcion="abrirVS"
                                                  actuaSobre="DNI" formActua="edi" panelActua="FichaEdicion"
                                                  claseManejadora="AltaSolicPermisos"}
                                        {CWCampoTexto nombre="Nombre" longitudMaxima="60" size="50" editable="false"}

                              {/if}
                              <!--No es tramitador-->
                              {if $smty_acciones eq 0}
                                        {CWCampoTexto nombre="DNI" obligatorio = "true" longitudMaxima="8" size="8"
                                                  editable="false" textoAsociado="Interesado"}
                                        {CWCampoTexto nombre="Nombre" longitudMaxima="60" size="50" editable="false"}
                              {/if}
                              <br>
                              <br>
                              <br>
                              <!--Esta lista  debería tener un actualizaA doble (actualizaA="causa,motivo"), pero como no
                              podemos cargaremos las listas dependientes y las ocultaremos-->
                              {CWLista nombre="motivo"  actualizaA="dispara_accion_interfaz"
                                        datos=$defaultData_AltaSolicPermisos.motivo
                                        editable="true" textoAsociado="Motivo"  dataType=$dataType_
                                        AltaSolicPermisos.motivo obligatorio = true}    
                              {CWCampoTexto nombre="reqJustificante"
                                        visible=$smty_datosFicha.$smty_iteracionActual.reqJustificanteVisible
                                        size="25" editable="false"}

                              <br>
                              <br>
                              {CWCampoTexto visible=$smty_datosFicha.$smty_iteracionActual.minimoVisible size="3"
                                        nombre="minimo"  editable="false" textoAsociado=" Mínimo"}
                               
                              {CWCampoTexto visible=$smty_datosFicha.$smty_iteracionActual.maximoVisible size="3"
                                        nombre="maximo"  editable="false" textoAsociado="Máximo"}
                               
                              {CWCampoTexto visible=$smty_datosFicha.$smty_iteracionActual.unidadVisible size="14"
                                        nombre="unidad"  editable="false" textoAsociado="Unidad"}
                               
                              <br><br>
                              {CWCampoTexto nombre="fechaIni" size="14" actualizaA="dispara_accion_interfaz"
                                        textoAsociado="Comienzo" dataType=$dataType_AltaSolicPermisos.fechaIni}
                               
                              {CWCampoTexto nombre="fechaFin" actualizaA="dispara_accion_interfaz" size="14"
                                        textoAsociado="Fin" dataType=$dataType_AltaSolicPermisos.fechaFin}
                               
                              {CWLista nombre="periodo" actualizaA="dispara_accion_interfaz"
                                        datos=$defaultData_AltaSolicPermisos.periodo
                                        editable="true" textoAsociado="Periodo"}
                              <br><br>
                              {CWCampoTexto nombre="fechaNaci" visible=$smty_datosFicha.$smty_iteracionActual.fechaNaciVisible
                                        size="14" actualizaA="dispara_accion_interfaz"
                                        textoAsociado="Fecha nacimiento del menor"
                                        dataType=$dataType_AltaSolicPermisos.fechaNaci}
                              <br><br>
                              {CWCampoTexto visible=$smty_datosFicha.$smty_iteracionActual.horasVisible nombre="horas"
                                        size="4" actualizaA="dispara_accion_interfaz" obligatorio="true"
                                        textoAsociado="Tiempo reducción" editable="true"
                                        dataType=$dataType_AltaSolicPermisos.horas}
                              {CWCampoTexto visible=$smty_datosFicha.$smty_iteracionActual.porcentajeVisible
                                        nombre="porcentaje" size="3" mascara="nn" obligatorio="true"
                                        textoAsociado="Porcentaje reducciÃ³n" editable="false"}
                              <br>
                              {CWLista visible=$smty_datosFicha.$smty_iteracionActual.distanciaVisible nombre="distancia"
                                        actualizaA="dispara_accion_interfaz"
                                        datos=$defaultData_AltaSolicPermisos.distancia
```

```
                                    obligatorio="true"  editable="true" textoAsociado="Distancia"}
                        <br>
                        {CWLista visible=$smty_datosFicha.$smty_iteracionActual.distanciaMFAoMUHVisible
                                    nombre="distanciaMFAoMUH" actualizaA="dispara_accion_interfaz"
                                    datos=$defaultData_AltaSolicPermisos.distanciaMFAoMUH
                                    obligatorio="true"  editable="true" textoAsociado="Distancia"}
                        <br>
                        {CWLista visible=$smty_datosFicha.$smty_iteracionActual.gradoParentescoVisible
                                    nombre="gradoParentesco" actualizaA="dispara_accion_interfaz"
                                    datos=$defaultData_AltaSolicPermisos.gradoParentesco
                                    obligatorio="true"  editable="true" textoAsociado="Grado de
                                    parentesco"}
                           
                        {CWCampoTexto nombre="indicarFamiliar"
                                    visible=$smty_datosFicha.$smty_iteracionActual.indicarFamiliarVisible
                                    size="27" editable="false"}
                        <br><br>
                        {CWAreaTexto nombre="observaciones"  dataType=$dataType_AltaSolicPermisos.observaciones
                                    longitudMaxima="500" cols="50" rows="2" textoAsociado="Observaciones"}
                        <br>
                        {CWCheckBox valor=$defaultData_AltaSolicPermisos.AltaHojaRosa
                                    visible=$smty_datosFicha.$smty_iteracionActual.AltaHojaRosaVisible
                                    nombre="AltaHojaRosa" editable="true"
                                    dataType=$dataType_AltaSolicPermisos.AltaHojaRosa textoAsociado="Alta
                                    papel"}
                        <br>
                        <br>
                        {CWLista visible=$smty_datosFicha.$smty_iteracionActual.aportaJustificanteVisible
                                    nombre="aportaJustificante"
                                    datos=$defaultData_AltaSolicPermisos.aportaJustificante obligatorio="true"
                                    dataType=$dataType_AltaSolicPermisos.aportaJustificante
                                    editable="true" textoAsociado="Adjunta justificante"}
                        <br>
                        {CWLista visible=$smty_datosFicha.$smty_iteracionActual.tipoVisible nombre="tipo"
                                    datos=$defaultData_AltaSolicPermisos.tipo obligatorio="true"
                                    dataType=$dataType_AltaSolicPermisos.tipo editable="true"
                                    textoAsociado="Tipo de minusvalía"}
                        <br>
                        {CWCheckBox valor=$defaultData_AltaSolicPermisos.incidencia
                                    visible=$smty_datosFicha.$smty_iteracionActual.incidenciaVisible
                                    nombre="incidencia" editable="true"
                                    dataType=$dataType_AltaSolicPermisos.incidencia
                                    textoAsociado="Generar incidencia"}
                          
                        {CWLista visible=$smty_datosFicha.$smty_iteracionActual.causaVisible nombre="causa"
                                    datos=$defaultData_AltaSolicPermisos.causa editable="true"
                                    textoAsociado="Causa"}
                        <br>

                {/CWFicha}
                {CWPaginador enlacesVisibles="3"}
        {/CWFichaEdicion}
{/CWContenedor}
{CWBarraInfPanel}
        {CWBoton imagen="22" id="saltoConfirmar" texto="Continuar" class="boton" accion="saltar"}
        {CWBoton imagen="42" texto="Cancelar" class="boton" accion="cancelar" action="cancelarEdicion"}
{/CWBarraInfPanel}
/CWPanel}

<!-- ***************** PESTAÃ‘AS ************************-->
{CWContenedorPestanyas}
        {CWPestanya tipo="edi" estado="on"}
{/CWContenedorPestanyas}
{/CWMarcoPanel}
{/CWVentana}
```

Views - Alta de solicitudes y permisos

```php
<?php


	//Creamos una pantalla
	$comportamientoVentana= new IgepPantalla();

	$panel1 = new IgepPanel('AltaSolicPermisos',"smty_datosFicha");
	//$panel1->activarModo("fil","estado_fil");
	//Panel listado que no usamos pq sale el mensaje de sin datos
	//$panel1->activarModo("lis","estado_lis");
	$panel1->activarModo("edi","estado_edi");
	$comportamientoVentana->agregarPanel($panel1);



	//Para ocultar o mostrar campos nada mÃ¡s entrar a la pantalla
	if(IgepSession::dameRol()=='P_TRAMITA')
		$s->assign("smty_acciones",1);
	else
		$s->assign("smty_acciones",0);


	$s->display('solicitudes/AltaSolicPermisos.tpl');

?>
```

Clase Manejadora – Alta de solicitudes y permisos Confirmar

```php
<?php



class AltaSolicPermisosConfirmar extends gvHidraForm_DB{

	var $listadoSolicitudesPermisosYLicencias;
	var $lanzarInforme;

	//En este atributo será el informe Jasper
	var $ij_solicitudes;

function AltaSolicPermisosConfirmar(){

	$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
	//Las tablas sobre las que trabaja
	$nombreTablas= array('vper_ocupacion','tper_permisos');
	parent::__construct($g_dsn,$nombreTablas);

	//Completamos con los datos que vienen del salto
	$objSalto = IgepSession::damePanel('saltoIgep');
	$datos = $objSalto->getParams();



	$interesado = $datos['arrayParams']['0']['DNI'];

	//La select que mostramos en la ficha
	$this->setSelectForSearchQuery( "SELECT " .
			" substr(nregpgv, 1,8) as \"DNI\" " .
			", nregpgv as \"nregpgv\" " .
			", dapell1 ||' '|| dapell2 ||', '|| dnombre as \"Nombre\"" .
			" FROM vper_ocupacion where nregpgv like '".$interesado."__'");

	/*Añadimos los Matching - Correspondecias campoTPL <-> campoBD*/
	$this->addMatching("DNI","nregpgv","vper_ocupacion");
	$this->addMatching("nregpgv","nregpgv","vper_ocupacion");
	$this->addMatching("Nombre","dnombre","vper_ocupacion");


}//Fin de Constructor


function accionesParticulares($str_accion, $objDatos) {
	switch($str_accion){
		case "guardarSolicitudPermiso":
				$actionForward = $this->guardarSolicitudPermiso($objDatos,$nsolicitud=null);
			break;
		case "guardarImprimirSolicitudPermiso":
				$actionForward = $this->guardarImprimirSolicitudPermiso($objDatos);
			break;
	}
	return $actionForward;
}


function guardarSolicitudPermiso($objDatos,& $nsolicitud){
	//Control de guardar imprimir
	if (IgepSession::existeVariable("AltaSolicPermisosConfirmar","listadoSolicitudesPermisosYLicencias")){
		$this->showMensaje('APL-GENERICO',array("Ya se guardÃ³ la solicitud"));
		$actionForward = $objDatos->getForward('incorrecto');
		return $actionForward;
	}

	//Recuperamos los datos e implementamos la insercción según sea permiso o solicitud
	$objSalto = IgepSession::damePanel('saltoIgep');
	$datos = $objSalto->getParams();
	$datos = $datos['arrayParams']['0'];
	$motivo=$datos['motivo'];
	$cclase= substr($motivo,0,strpos($motivo,'|'));
	$totalDias = $datos['totalDias'];
	$totalDiasLaboVac = $datos['totalDiasLaboVac'];
	$totalDiasLabo = $datos['totalDiasLabo'];
	$diasNaturales = $datos['var_dias_naturales'];
	$diasHabiles = $datos['var_dias_habiles'];
	$AltaHojaRosa = $datos['AltaHojaRosa'];
	$distancia = $datos['distancia'];
	$distanciaMFAoMUH = $datos['distanciaMFAoMUH'];
	$gradoparentesco = $datos['gradoParentesco'];
	$listaSolicitudesAnulacion = $datos['listaSolicitudesAnulacion'];
	$listaSolicitudesAnulacionResueltas = $datos['listaSolicitudesAnulacionResueltas'];
	//preInsert

	if(($datos['tipoAlta']=='solicitud' or  $datos['tipoAlta']=='S') and $AltaHojaRosa=='N'){

		//Utilizar la tabla TPER_SOLICITUDES para insertar los datos de la SOLICITUD
		$tabla='TPER_SOLICITUDES';
	//f_preGrabarAltaSolicitud()
	//Antes de insertar la solicitud en TPER_SOLICITUDES se deberá generar la secuencia (sper_solicitud) para el nº de
	solicitud.

		$nsolicitud = $this->calcularSecuenciaBD('sper_solicitud');

		if($nsolicitud<0){
			$this->showMensaje('APL-GENERICO',array("Error al calcular NÂº de solicitud"));
			$actionForward = $objDatos->getForward('incorrecto');
			return $actionForward;
```

```
		}

		$datos['datosSoliPer']['solicitud']['nsolicitud']=$nsolicitud;
//Fin f_preGrabarAltaSolicitud()
}else{
		//Utilizar la tabla TPER_PERMISOS para insertar los datos del PERMISO
		$tabla='TPER_PERMISOS';
}
//Valores por defecto
//Solicitudes

if(isset($datos['datosSoliPer']['solicitud'])){
		$datos['datosSoliPer']['solicitud']['altamodificada']='N';

		if ($cclase=='ENC')
			$datos['datosSoliPer']['solicitud']['justificante']=$datos['aportaJustificante'];
		else
			$datos['datosSoliPer']['solicitud']['justificante']='N';


		if ($cclase=='EGP' or $cclase=='DEF' or $cclase=='TDO' ){

			$datos['datosSoliPer']['solicitud']['distancia']=$datos['distancia'];
			$datos['datosSoliPer']['solicitud']['gradoparentesco']=$datos['gradoParentesco'];
		}


		if ($cclase=='MFA' or $cclase=='MUH' ){
			$datos['datosSoliPer']['solicitud']['distanciaMFAoMUH']=$datos['distanciaMFAoMUH'];
		}

		$datos['datosSoliPer']['solicitud']['nsolanulacion']=$nsolicitud;



}
//Fin preInsert


//Control de transacción
$g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
$conOracle = new IgepConexion($g_oracle_dsn);
$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
$conPostgres = new IgepConexion($g_dsn, true);
$conPostgres->empezarTransaccion();
$conOracle->empezarTransaccion();
$error = 0;



//Limpiamos tags HTML del aviso
$avisoSinHtmlGuion = str_replace ("</LI>","\n",$datos['datosSoliPer']['solicitud']['avisos']);
$avisoSinHtmlGuion = str_replace ("<UL>","",$avisoSinHtmlGuion);
$avisoSinHtmlGuion = str_replace ("</UL>","",$avisoSinHtmlGuion);
$avisoSinHtmlGuion = str_replace ("</LI>","\n",$avisoSinHtmlGuion);
$avisoSinHtmlGuion = str_replace ("<LI>","- ",$avisoSinHtmlGuion);
$avisoSinHtmlGuion = str_replace ("<br>","",$avisoSinHtmlGuion);
$datos['datosSoliPer']['solicitud']['avisos']= $avisoSinHtmlGuion;



IgepDebug::setDebug(DEBUG_USER,"Datos: <pre>".print_r($datos['datosSoliPer'],true)."</pre>");



$conPostgres->prepararOperacion($datos['datosSoliPer']['solicitud']['observaciones']);
$conPostgres->prepararOperacion($datos['datosSoliPer']['permiso']['observaciones']);
switch ($tabla){
case "TPER_SOLICITUDES":

	if(empty($datos['datosSoliPer']['solicitud']['fecha_fin']))
				unset($datos['datosSoliPer']['solicitud']['fecha_fin']);
	$indices = array_keys($datos['datosSoliPer']['solicitud']);

	$insert="INSERT INTO TPER_SOLICITUDES (fsolicitud, nregpgv, cclase, periodo,";
		if (strlen($datos['datosSoliPer']['solicitud']['observaciones'])>1)
					$insert=$insert."observaciones,";
		$insert=$insert."fecha_ini,justificante,fecha_fin,situacion,cestado,";
		if (strlen($datos['datosSoliPer']['solicitud']['avisos'])>1)
					$insert=$insert."avisos,";
		if (strlen($datos['datosSoliPer']['solicitud']['avisosusuario'])>1)
					$insert=$insert."avisosusuario,";
		if (is_object($datos['datosSoliPer']['solicitud']['fnacimientomenor']))
					$insert=$insert."fnacimientomenor,";
		if (strlen($datos['datosSoliPer']['solicitud']['altamodificada'])>0)
					$insert=$insert."altamodificada,";

		if (!empty($datos['datosSoliPer']['solicitud']['requierejustificante']))
					$insert=$insert."requierejustificante,";
		if (!empty($datos['datosSoliPer']['solicitud']['distancia']))
					$insert=$insert."distancia,";
		if (!empty($datos['datosSoliPer']['solicitud']['distanciaMFAoMUH']))
					$insert=$insert."distancia,";
		if (!empty($datos['datosSoliPer']['solicitud']['gradoparentesco']))
					$insert=$insert."gradoparentesco,";
		$insert=$insert."nsolicitud) VALUES(" .
		'".$conPostgres->prepararFecha($datos['datosSoliPer']['solicitud']['fsolicitud'])."',
		'".$datos['datosSoliPer']['solicitud']['nregpgv']."',
		'".$datos['datosSoliPer']['solicitud']['cclase']."',
		 ".$datos['datosSoliPer']['solicitud']['periodo']."," ;
		if (strlen($datos['datosSoliPer']['solicitud']['observaciones'])>1)
```

```
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['observaciones']."',";
        $insert=$insert."'".$conPostgres->prepararFecha($datos['datosSoliPer']['solicitud']['fecha_ini'])."',
        '".$datos['datosSoliPer']['solicitud']['justificante']."'," .
        '".$conPostgres->prepararFecha($datos['datosSoliPer']['solicitud']['fecha_fin'])."',
        '".$datos['datosSoliPer']['solicitud']['situacion']."',
        '".$datos['datosSoliPer']['solicitud']['cestado']."',";
        if (strlen($datos['datosSoliPer']['solicitud']['avisos'])>1)
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['avisos']."',";
        if (strlen($datos['datosSoliPer']['solicitud']['avisosusuario'])>1)
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['avisosusuario']."',";
        if (is_object($datos['datosSoliPer']['solicitud']['fnacimientomenor']))
                $insert=$insert."'".$conPostgres-
                        >prepararFecha($datos['datosSoliPer']['solicitud']['fnacimientomenor'])."',";
        if (strlen($datos['datosSoliPer']['solicitud']['altamodificada'])>0)
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['altamodificada']."',";
        if (!empty($datos['datosSoliPer']['solicitud']['requierejustificante']))
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['requierejustificante']."',";
        if (!empty($datos['datosSoliPer']['solicitud']['distancia']))
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['distancia']."',";
        if (!empty($datos['datosSoliPer']['solicitud']['distanciaMFAoMUH']))
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['distanciaMFAoMUH']."',";
        if (!empty($datos['datosSoliPer']['solicitud']['gradoparentesco']))
                $insert=$insert."'".$datos['datosSoliPer']['solicitud']['gradoparentesco']."',";
        $insert=$insert."'".$datos['datosSoliPer']['solicitud']['nsolicitud']."')";


        $res = $conPostgres->operar($insert);
        if($res<0){
                $error=1;
                $this->showMensaje('APL-GENERICO',array("Error insertando solicitud"));
                $actionForward = $objDatos->getForward('incorrecto');
                return $actionForward;
        }
        break;

case "TPER_PERMISOS":


        $insert="INSERT INTO $tabla " .
                "(nregpgv,cclase,periodo,";
        if (strlen($datos['datosSoliPer']['permiso']['observaciones'])>1)
                        $insert=$insert."observaciones,";
        $insert=$insert."fecha_ini,justificante";
        if($datos['datosSoliPer']['permiso']['causa']!='')$insert=$insert.",causa";
        if($datos['datosSoliPer']['permiso']['horas_reduccion']!='')$insert=$insert.",horas_reduccion";
        if($datos['datosSoliPer']['permiso']['porcentaje']!='')$insert=$insert.",porcentaje_reduccion";
        if($datos['datosSoliPer']['permiso']['fnacimientomenor']!='')$insert=$insert.",fnacimientomenor";
        if (strlen($datos['datosSoliPer']['permiso']['tipo_minusvalia'])>0)$insert=$insert.",tipo_minusvalia";
        $insert=$insert.",cpro,cmun";
        if($datos['datosSoliPer']['permiso']['fecha_fin']!='')$insert=$insert.",fecha_fin";
        $insert=$insert.")
        VALUES ('".$datos['datosSoliPer']['permiso']['nregpgv']."',
        '".$datos['datosSoliPer']['permiso']['cclase']."',
        '".$datos['datosSoliPer']['permiso']['periodo']."'," ;
        if(strlen($datos['datosSoliPer']['permiso']['observaciones'])>1)
                $insert=$insert."'".$datos['datosSoliPer']['permiso']['observaciones']."',";
        $insert=$insert."'".$conPostgres->prepararFecha($datos['datosSoliPer']['permiso']['fecha_ini'])."'," ;
        if($datos['datosSoliPer']['permiso']['justificante']!='')
                $insert=$insert."'".$datos['datosSoliPer']['permiso']['justificante']."'," ;
        if($datos['datosSoliPer']['permiso']['causa']!='')
                $insert=$insert."'".$datos['datosSoliPer']['permiso']['causa']."',";
        if($datos['datosSoliPer']['permiso']['horas_reduccion']!='')
                $insert=$insert."'".$datos['datosSoliPer']['permiso']['horas_reduccion']."',";
        if($datos['datosSoliPer']['permiso']['porcentaje']!='')
                $insert=$insert."'".$datos['datosSoliPer']['permiso']['porcentaje']."',";
        if($datos['datosSoliPer']['permiso']['fnacimientomenor']!='')$insert=$insert."'".$conPostgres-
                        >prepararFecha($datos['datosSoliPer']['permiso']['fnacimientomenor'])."',";
        if (strlen($datos['datosSoliPer']['permiso']['tipo_minusvalia'])>0)
                $insert=$insert."'".$datos['datosSoliPer']['permiso']['tipo_minusvalia']."',";
        $insert=$insert."'".$datos['datosSoliPer']['permiso']['cpro']."',
        '".$datos['datosSoliPer']['permiso']['cmun']."'";
        if($datos['datosSoliPer']['permiso']['fecha_fin']!='')$insert=$insert.",'".$conPostgres-
                        >prepararFecha($datos['datosSoliPer']['permiso']['fecha_fin'])."'";

        $insert=$insert.");";
        $res=$conPostgres->operar($insert);

        if($res<0){
                $error=1;
                $this->obj_errorNegocio->limpiarError();
                $this->showMensaje('APL-GENERICO',array("Error insertando permiso"));
                $actionForward = $objDatos->getForward('incorrecto');
                return $actionForward;
        }
        break;
}


//postInsert()
IgepDebug::setDebug(DEBUG_USER,'INI postInsert');

if(($datos['tipoAlta']=='solicitud' or $datos['tipoAlta']=='S')  and $AltaHojaRosa=='N') {

   if($datos['datosSoliPer']['solicitud']['cclase']=='ADI' OR $datos['datosSoliPer']['solicitud']['cclase']=='VAC'){
        if($datos['datosSoliPer']['vacyAdiCompletos']=='S'){
                $update="UPDATE TPER_SOLICITUDES SET cestado='PR'
                        WHERE cestado='PC' and
                              periodo = ".$datos['datosSoliPer']['solicitud']['periodo']." and
                              nregpgv = '".$datos['datosSoliPer']['solicitud']['nregpgv']."'";
                $res=$conPostgres->operar($update);
                if($res<0){
                        $error=1;
```

```
                                $this->showMensaje('APL-GENERICO',array("Error actualizando solicitud(PR)"));
                                $actionForward = $objDatos->getForward('incorrecto');
                                return $actionForward;
                        }
                }
        }


        if(is_array($listaSolicitudesAnulacion))foreach ($listaSolicitudesAnulacion as $nsolicitudAnula){
                // Para insertar una anulación (que antes era un alta) en tper_solicitudesmodificadas no debe de estar
                // asociada a otra anulación. Si relacionasemos un alta con todas las anulaciones, luego no podríamos
                // controlar bien el orden de las firmas, y como consecuencia, podríamos pasarnos del máximo de días

                $solAsociadaEsAnulacion ="
                        SELECT s2.nsolicitudmodifica
                        FROM tper_solicitudesmodificadas s2, tper_solicitudes an
                        WHERE an.nsolicitud = s2.nsolicitud AND
                            s2.nsolicitudmodifica = $nsolicitudAnula
                           AND
                           (
                           NOT ((an.cestado in ('NR','NI')) OR
                                (an.situacion = 'A' and an.cestado='NP' and an.ffirmaresolucion is not null) OR
                                (an.situacion = 'B' and an.cestado='AP' and an.ffirmaresolucionanula is not null))
                           ) AND
                           an.situacion IN ('X','B')";
                $res=$this->consultar($solAsociadaEsAnulacion);
                if(count($res)==0)
                SolicitudesModificadas::f_altaSolicitudesModificadas($datos['datosSoliPer']['solicitud']['nsolicitud'],
                                                                     $nsolicitudAnula, 'N',$conPostgres);

        }

        if(is_array($listaSolicitudesAnulacionResueltas))foreach ($listaSolicitudesAnulacionResueltas as
                                                                   $nsolicitudAnula){
                // Para insertar una anulación (que antes era un alta) en tper_solicitudesmodificadas no debe de estar
                // asociada a otra anulación. Si relacionasemos un alta con todas las anulaciones, luego no podríamos
                // controlar bien el orden de las firmas, y como consecuencia, podríamos pasarnos del máximo de días

                $solAsociadaEsAnulacion ="
                        SELECT s2.nsolicitudmodifica
                        FROM tper_solicitudesmodificadas s2, tper_solicitudes an
                        WHERE an.nsolicitud = s2.nsolicitud AND
                            s2.nsolicitudmodifica = $nsolicitudAnula
                           AND
                           (
                           NOT ((an.cestado in ('NR','NI')) OR
                                (an.situacion = 'A' and an.cestado='NP' and an.ffirmaresolucion is not null) OR
                                (an.situacion = 'B' and an.cestado='AP' and an.ffirmaresolucionanula is not null))
                           ) AND
                           an.situacion IN ('X','B')";
                         $res=$this->consultar($solAsociadaEsAnulacion);
                if(count($res)==0)
                SolicitudesModificadas::f_altaSolicitudesModificadas($datos['datosSoliPer']['solicitud']['nsolicitud'],
                                                                     $nsolicitudAnula, 'S',$conPostgres);
         }



}
//Fin f_postGrabarAltaSolicitud

if($datos['incidencia']=='S'){//Solo tiene valor si es permiso
        $generarIncidencia=Permisos::f_generarIncidencia($conPostgres,
                                                         $conOracle,$datos['datosSoliPer']['permiso']['nregpgv'],
                                                         $datos['datosSoliPer']['permiso']['cclase'],
                                                         $datos['datosSoliPer']['permiso']['fecha_ini']);

        if(($generarIncidencia['0']<0) and (!empty($generarIncidencia['error']))){//Hay error de prohibición
                $error=1;
                $this->showMensaje('APL-GENERICO',array($generarIncidencia['error']));
                $actionForward = $objDatos->getForward('incorrecto');
                return $actionForward;
        }
}

//Enviar mensaje de correo de postGrabarAltaSolicitud
if(($datos['tipoAlta']=='solicitud' or $datos['tipoAlta']=='S' and $AltaHojaRosa=='N')  ){

        $usuNoFicha = UsuariosDePersonal::f_noFichan($datos['datosSoliPer']['solicitud']['nregpgv']);
        if(!$usuNoFicha){
                //Obtener el centro, departamento y grupo al que pertenece el usuario que realiza la solicitud
                $grupo=Personal::f_obtenerGrupo($datos['datosSoliPer']['solicitud']['nregpgv']);
                if(empty($grupo['centro']) or is_null($grupo) or count($grupo)==0 ){
                        $error = 1;
                        $this->showMensaje('APL-34');
                        $actionForward = $objDatos->getForward('incorrecto');
                        return $actionForward;
                }

                // Obtenemos el centro, departamento y grupo de lo obtenido en "grupo"
                $var_centro = $grupo['centro'];
                $var_departamento = $grupo['departamento'];
                $var_grupo = $grupo['seccion'];

                //Si el usuario que realiza la solicitud es responsable de firma no suplente no se envía ningún correo,
                // ya que solo el podrá firmar sus solicitudes
                if( Firmantes::f_esResponsableNoSuplente($datos['datosSoliPer']['solicitud']['nregpgv'],
                                                         $var_centro,$var_departamento,$var_grupo) ){
                        // No se envia correo
                        $enviocorreo=false;
                }
```

```
        else{
            IgepDebug::setDebug(DEBUG_USER,"datos<pre>".print_r($datos,true)."</pre>");
            //Obtener los responsables de ese grupo
            $destinatarios = Firmantes::f_dameResponsables($grupo,$datos['datosSoliPer']['solicitud']['fecha_ini'],
                                                           $datos['datosSoliPer']['solicitud']['fsolicitud'],'N');
            $enviocorreo=true;
        }
    }
    else{
            // Obtener el grupo al que pertenecen los usuarios que no fichan
            $var_ccentrab = UsuariosDePersonal::f_dameCentroTrab($datos['datosSoliPer']['solicitud']['nregpgv'],
                                                                                                    $ccentrab);
            if($var_ccentrab<0){
                    $error=1;
                    $this->showMensaje('APL-GENERICO',array("Error consultando centro de trabajo del usuario"));
                    $actionForward = $objDatos->getForward('incorrecto');
                    return $actionForward;
            }


            // Permitir que en un centro donde no se fiche, el responsable de firma pueda ser alguien
             que no fiche(f_anularSolicitud)

            UsuariosDePersonal::f_obtenerCdgCserv($datos['datosSoliPer']['solicitud']['nregpgv'],$pcdg,$pcserv);

            Firmantes::f_obtenerFirmanteOEmpleadoGrupoNoFichan($ccentrab,
                                                           $datos['datosSoliPer']['solicitud']['nregpgv'],
                                                           $pcdg, $pcserv, $ficha, $firmanteGrupoNoFichan);

            if(!empty($firmanteGrupoNoFichan) or $firmanteGrupoNoFichan!='' ){
                 if ($ficha=='S'){
                      $grupoNoFichan = Personal::f_obtenerGrupo($firmanteGrupoNoFichan);
                      if ($grupoNoFichan!='') {
                                  // Se obtienen los responsables del grupo correspondiente
                                  $destinatarios=Firmantes::f_dameResponsables($grupoNoFichan,
                                  $datos['datosSoliPer']['solicitud']['fecha_ini'],
                                  $datos['datosSoliPer']['solicitud']['fsolicitud'],'N');
                      }
                      else{
                               $this->showMensaje('APL-32');
                               return -1;
                      }
                  }
                  else{
                      $destinatarios=$firmanteGrupoNoFichan;
                  }

            }
            else{
                  //dar mensaje
                  $this->showMensaje('APL-94');
                  return -1;
            }

            $enviocorreo=true;

    }


    if ($enviocorreo) {

    //-------------------------------------------------------------------------------------------------------------------
                      //Se envía correo a los responsables de firma

    //-------------------------------------------------------------------------------------------------------------------

            //Enviar correo a los responsables
            if(empty($destinatarios)){
                      $error=1;
                      $this->showMensaje('APL-35');
                      $actionForward = $objDatos->getForward('incorrecto');
                      return $actionForward;
            }

            $mensaje = "";

            //Limpiamos tags HTML del aviso
            $avisoSinHtml = str_replace ("</LI>","\n",$datos['datosSoliPer']['solicitud']['avisosusuario']);
            $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
            $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
            $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
            $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
            $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);

            // Añadimos los parámetros cclase, totalDias y totalDiasLabo


            if($datos['datosSoliPer']['solicitud']['cclase'] == 'VAC')
                      $totalDiasLabo = $totalDiasLaboVac;


            if( is_null($totalDias) && is_null($totalDiasLabo)){
                      $totalDias = $diasNaturales;
                      $totalDiasLabo = $diasHabiles;
            }
            //Generar mensaje a enviar en el correo
            Solicitudes::f_GenerarMensajeCorreo($datos['datosSoliPer']['solicitud']['nregpgv'],$datos['Nombre'],
                      $datos['datosSoliPer']['solicitud']['cclase'],substr($datos['motivo'],
                      strpos($datos['motivo'],'|')+1),
                      $datos['periodo'],$datos['datosSoliPer']['solicitud']['fecha_ini'],
                      $datos['datosSoliPer']['solicitud']['fecha_fin'],
                      $datos['datosSoliPer']['solicitud']['observaciones'],
```

```
                        "",$avisoSinHtml, "S","N","N","N","N","N","N",$totalDias, $totalDiasLabo,
                        $datos['datosSoliPer']['solicitud']['nsolicitud'],$mensaje);
                        $retorno = Solicitudes::f_enviarCorreo($datos['datosSoliPer']['solicitud']['nregpgv'],
                                                $destinatarios,$mensaje,'PERSONAL - Alta de solicitud');

                        //Mensaje de aviso al volver al panel original desde el que saltamos
                if(!$retorno){
                        $obj_mensaje = new IgepMensaje('APL-GENERICO-AVISO',array("Problemas al enviar el correo, la
                                                                        solicitud si que se dio de alta."));
                        //Metemos el mensaje en la clase saltadora
                        IgepSession::guardaVariable('AltaSolicPermisos','obj_mensaje',$obj_mensaje);
                        $actionForward = $objDatos->getForward('correcto');

                        //Control de transacción
                        $conPostgres->acabarTransaccion($error);
                        $conOracle->acabarTransaccion($error);

                        return $actionForward;
                }
        }


        }//Fin enviar mensaje de correo

        //Fin postInsert

        //Control de transacción
        $conPostgres->acabarTransaccion($error);
        $conOracle->acabarTransaccion($error);

        //Llegados a este punto todo fue bien
        if(($datos['tipoAlta']=='solicitud' or $datos['tipoAlta']=='S' and $AltaHojaRosa=='N')  ){
                $obj_mensaje = new IgepMensaje('APL-36',array($datos['Nombre'],$datos['DNI'],
                                                        $datos['datosSoliPer']['solicitud']['nsolicitud']));
                //Metemos el mensaje en la clase saltadora
                IgepSession::guardaVariable('AltaSolicPermisos','obj_mensaje',$obj_mensaje);
                $actionForward = $objDatos->getForward('correcto');
        }else{//Es un permiso con o sin incidencia
                $obj_mensaje = new IgepMensaje('APL-37',array($datos['Nombre'],$datos['DNI'],
                                                                        $generarIncidencia['errorAviso']));
                //Metemos el mensaje en la clase saltadora
                IgepSession::guardaVariable('AltaSolicPermisos','obj_mensaje',$obj_mensaje);
                $actionForward = $objDatos->getForward('correcto');
        }
        return $actionForward;
}



function guardarImprimirSolicitudPermiso($objDatos){

        //Recuperamos los datos e implementamos la insercción según sea permiso o solicitud
        $objSalto = IgepSession::damePanel('saltoIgep');
        $datos = $objSalto->getParams();
        $datos = $datos['arrayParams']['0'];


        //Guardamos solicitudPermiso
        $nsolicitud =0;//recogemos el numero de solicitud para hacer el reporte
        $actionForward = $this->guardarSolicitudPermiso($objDatos,$nsolicitud);

        //control de error Si ha habido algun error, no se lanza el informe jasper
        if($actionForward->_name=='incorrecto'){
                return $actionForward;
        }

        //Es un permiso
        if(isset($datos['datosSoliPer']['permiso'])){
             if($datos['datosSoliPer']['codJustificante']=='S'){
                //Creamos el objeto que manejarÃ¡ el informe
                $informeJ = new InformeJasper('solicitudPermisoYLicencia');
                //Especificamos la fuente de datos, en este caso, una BD relacional
                $informeJ->setDataSourceType('sgbd');
                //Fijamos la disposición del informe incrustado en la ventana
                $informeJ->setDisposition('inline');
                //Especificamos los parámetros relativos a la conexión con la BD relacional
                //Si coinciden con los del DSN de IGEP, podemos importarlos con:
                $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
                $informeJ->importPearDSN($g_dsn);
                //Fijamos el fichero jasper que nos hace de plantilla
                $informeJ->setJasperFile('./plantillasJasper/listadoPermisos.jasper');

                $queryPermiso="SELECT substr(u.nregpgv, 1,8) as \"nregpgv\",
                   u.dnombre, u.dapell1 || ' ' || CASE WHEN u.dapell2 is null THEN ' ' ELSE u.dapell2 END as \"apellido\",
                                p.fecha_ini, p.fecha_fin, p.observaciones, dclase, o.dservv,
                                CASE WHEN u.creljur ='FC' THEN 'Funcionari de carrera'
                                WHEN u.creljur ='FI' THEN 'Interíno'
                                WHEN u.creljur ='FV' THEN 'Eventual'
                                WHEN u.creljur ='CL' or u.creljur ='CT' THEN 'Laboral' ELSE '' END as \"reljur\"
                        FROM tper_cod_permisos c, vcmn_organoso o, vper_ocupacion u, tper_permisos p
                        WHERE p.cclase = c.cclase and
                           p.nregpgv = u.nregpgv AND
                           o.cdgo = u.cdg AND
                           o.cservo = u.cserv AND
                           p.nregpgv = '".$datos['datosSoliPer']['permiso']['nregpgv']."' AND
                           p.cclase = '".$datos['datosSoliPer']['permiso']['cclase']."' AND
                           p.fecha_ini = '".$this->obj_conexion-
                                        >prepararFecha($datos['datosSoliPer']['permiso']['fecha_ini'])."'";
```

```
                //Añado los parametros
                $informeJ->addParam('query', $queryPermiso, 'String');
                $res = $this->consultar($queryPermiso);

                if ((!is_array($res) && $res==-1)OR(count($res)==0)) {   //Hay un error en la query o no hay datos

                  unset($informeJ);//Borramos el objeto...
                  unset($this->listadoSolicitudesPermisosYLicencias);//Borramos posibles anteriores objetos
                  $actionForward = $objDatos->getForward('correcto');
                  return $actionForward;

                }

                //Asignamos el informe como variable de clase para poder
                //acceder a él desde el fichero views donde realizaremos "la ejecución"
                $this->listadoSolicitudesPermisosYLicencias = $informeJ;
                $this->ij_solicitudes = $informeJ;
                $this->lanzarInforme = true;
                $actionForward = $objDatos->getForward('correcto');
                return $actionForward;

            }
        //Si la solicitud requiere adjuntar un justificante o es LIP(Licencia por interes particular), deberá imprimirse la
        solicitud
        }elseif( $datos['datosSoliPer']['codJustificante']=='S'
                                    OR        $datos['datosSoliPer']['solicitud']['cclase']== 'LIP'
                                    OR $datos['datosSoliPer']['permiso']['cclase']== 'LIP'
                                    OR $datos['datosSoliPer']['solicitud']['requierejustificante']=='S'){

        //Comprobamos si existe la solicitud antes de imprimir.(nsolicitud viene de una secuencia)
        $res=$this->consultar("Select count(nsolicitud) as cuenta from tper_solicitudes where nsolicitud=$nsolicitud");
                if($res[0]['cuenta']==0){
                        unset($this->listadoSolicitudesPermisosYLicencias);//Borramos posibles anteriores objetos
                        $actionForward = $objDatos->getForward('correcto');
                        return $actionForward;
                }
                //Creamos el objeto que manejará el informe
                $informeJ = new InformeJasper('solicitudPermisoYLicencia');
                //Especificamos la fuente de datos, en este caso, una BD relacional
                $informeJ->setDataSourceType('sgbd');
                //Fijamos la disposición del informe incrustado en la ventana
                $informeJ->setDisposition('inline');
                //Especificamos los parámetros relativos a la conexión con la BD relacional
                //Si coinciden con los del DSN de IGEP, podemos importarlos con:
                $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
                $informeJ->importPearDSN($g_dsn);
                // Si la solicitud requiere adjuntar un justificante o es resuelta por la DGAA
                ('LIP','LEF','LPP','EFP','AIN'),
                // deberá imprimirse la solicitud
                //Fijamos el fichero jasper que nos hace de plantilla
                //  cambio de  cclase not in ('LIP','LEF','LPP','EFP','AIN')
                $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');
                foreach($permisosDGAA as $index=>$tupla){

                        $var_cclases=$var_cclases."'".$permisosDGAA[$index]['cclase']."',";

                }
                $var_cclases=rtrim($var_cclases,",");
                $var_tupla=$datos['datosSoliPer']['solicitud']['cclase'];
                $pos=strpos ($var_cclases,$var_tupla);
                if ($pos > 0)
                        $informeJ->setJasperFile('./plantillasJasper/listadoSolicitudPermisosYLicenciasDGAA.jasper');
                else
                        $informeJ->setJasperFile('./plantillasJasper/listadoSolicitudPermisosYLicencias.jasper');

                //Añado los parametros
                $informeJ->addParam('nsolicitud', $nsolicitud, 'integer');
                //Asignamos el informe como variable de clase para poder
                //acceder a él desde el fichero views donde realizaremos "la ejecución"
                $this->listadoSolicitudesPermisosYLicencias = & $informeJ;
                $this->ij_solicitudes = $informeJ;
                $this->lanzarInforme = true;

                $actionForward = $objDatos->getForward('correcto');
                return $actionForward;
          }

        //Avisamos si ya se ha guardado la solicitud
        if($actionForward->_name=='incorrecto'){
                return $actionForward;
        }

        $actionForward = $objDatos->getForward('correcto');
        return $actionForward;
}


}//Fin AltaSolicPermisosConfirmar

?>
```

Tpl – Alta de solicitudes y permisos Confirmar

```
{CWVentana tipoAviso=$smty_tipoAviso  codAviso=$smty_codError  descBreve=$smty_descBreve  textoAviso=$smty_textoAviso}

{CWBarra usuario=$smty_usuario codigo=$smty_codigo customTitle=$smty_customTitle}
        {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}

        <!-- ***************** PANEL edi ***********************-->
        {CWPanel id="edi" tipoComprobacion="envio" action="operarBD" method="post" estado="on"
                                                                claseManejadora="AltaSolicPermisosConfirmar"}
                {CWBarraSupPanel titulo="Alta de solicitud de permisos y licencias"}

                {/CWBarraSupPanel}
                {CWContenedor}
                        {CWFichaEdicion id="FichaEdicion" datos=$smty_datosFicha numPagInsertar="1"
                                                                        accion=$smty_operacionFicha}
                        {CWFicha}

                                {CWCampoTexto nombre="DNI" longitudMaxima="8" size="8" editable="false"
                                                                        textoAsociado="Interesado"}
                                {CWCampoTexto nombre="Nombre" longitudMaxima="60" size="50" editable="false"}
                                {CWCampoTexto nombre="nregpgv"  size="11" editable="false" oculto="true"
                                                                        textoAsociado="Oculto"}
                                <br>
                                <br>
                                <table style="border-style: none; width: 100%;" cellpadding="2" cellspacing="0">
                                        <tbody><tr class="barraInfPanel">
                                                <td class="paginador">
                                                ATENCIÓN:
                                                </td>
                                        </tr>
                                        </tbody>
                                </table>
                                <br>
                                {$smty_mensajeAlta}<br>
                                <br>
                                <table style="border-style: none; width: 100%;" cellpadding="2" cellspacing="0">
                                        <tbody><tr class="barraInfPanel">
                                                <td class="paginador">
                                                AVISOS:
                                                </td>
                                        </tr>
                                        </tbody>
                                </table>
                                {$smty_avisos}<br>
                        {/CWFicha}
                        {CWPaginador enlacesVisibles="1"}
                        {/CWFichaEdicion}
                {/CWContenedor}
                {CWBarraInfPanel}
                        {CWBoton imagen="48" texto="Volver" class="boton" accion="volver"}   

                        {if $smty_justificante eq 'S' AND $smty_aportaJustificante eq 'S'}

                                {CWBoton imagen="06" texto="Guardar e Imprimir" class="boton" accion="particular"
                                        action="guardarImprimirSolicitudPermiso" mostrarEspera = "Guardando ..."}
                        {else}
                                {CWBoton imagen="41" texto="Guardar" class="boton" accion="particular"
                                        action="guardarSolicitudPermiso" mostrarEspera = "Guardando ..."}
                        {/if}
                {/CWBarraInfPanel}
        {/CWPanel}


        <!-- ****************** PESTAÃ‘AS ************************-->
        {CWContenedorPestanyas}
                {CWPestanya tipo="edi" estado="on"}
        {/CWContenedorPestanyas}
{/CWMarcoPanel}
{/CWVentana}
```

Views – Alta de solicitudes y permisos Confirmar

```php
<?php

//Creamos una pantalla
$comportamientoVentana= new IgepPantalla();
$panel1 = new IgepPanel('AltaSolicPermisosConfirmar',"smty_datosFicha");
$panel1->activarModo("fil","estado_fil");
$panel1->activarModo("edi","estado_edi");
$comportamientoVentana->agregarPanel($panel1);


//Los datos del interesado se sacan en buscar.

//Completamos con los datos que vienen del salto
$objSalto = IgepSession::damePanel('saltoIgep');
$datos = $objSalto->getParams();
$datos=$datos['arrayParams']['0'];



$s->assign('smty_mensajeAlta',$datos['datosSoliPer']['mensajeAlta']);
//$s->assign('smty_avisos',$datos['datosSoliPer']['mensajeA']);
$s->assign('smty_avisos',$datos['resultado']['mensajeHTML']);

//InitForm

//Si la solicitud requiere adjuntar un justificante o es LIP(Licencias por interes particular),
//deberá imprimirse la solicitud, de ahí que se visualice el botón "Grabar e Imprimir Solicitud"
if(isset($datos['datosSoliPer']['codJustificante'])){
          //Tambien pido justificante si requierejustificante='S'
          if($datos['datosSoliPer']['codJustificante']=='S' OR $datos['datosSoliPer']['solicitud']['cclase']=='LIP'
                                    OR $datos['datosSoliPer']['solicitud']['requierejustificante']=='S')
                    $s->assign('smty_justificante','S');
          else
                    $s->assign('smty_justificante','N');
          // Si se trata de 'ENC' y no se aportaJustificante (N), en la siguiente pantalla se mostrará el botón 'Guardar'
          // en vez del 'Guardar e Imprimir'
          if($datos['datosSoliPer']['solicitud']['cclase']=='ENC') {

                    if(isset($datos['datosSoliPer']['aportaJustificante']))
                              $s->assign('smty_aportaJustificante',$datos['datosSoliPer']['aportaJustificante']);
          }
          else
                    $s->assign('smty_aportaJustificante','S');

}


$s->display('solicitudes/AltaSolicPermisosConfirmar.tpl');

?>
```

6.2. Anulación de solicitudes y permisos

Clase Manejadora - Anular solicitudes y permisos

```
<?php

class AnularSolicitudes extends gvHidraForm_DB{

        function AnularSolicitudes(){


        $nombreTablas= array('tper_solicitudes');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        parent::__construct($g_dsn,$nombreTablas);

        $this->setSelectForSearchQuery( "SELECT
                        CASE WHEN s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado='AP' and s.fanulaper
                                is null THEN 'Autorizada'
                        WHEN s.ffirmaresolucionanula is not null and s.situacion ='A' and s.cestado='AP' and s.fanulaper
                                is not null THEN 'No autorizada' ELSE e.destado END as \"destado\" ,
                                     s.situacion,s.nsolicitud,p.dclase,s.fecha_ini,s.fecha_fin, CASE WHEN (s.avisos is
                                null or s.avisos = '') THEN 'images/nada.gif' ELSE 'images/ok.gif' END as
                                \"img_avisos\", s.cestado, s.fanulaper, s.ffirmaresolucion, s.ffirmaresolucionanula
                        FROM TPER_SOLICITUDES s,tper_cod_permisos p, TPER_ESTADOSOL e");
        $this->setWhereForSearchQuery( "s.cclase = p.cclase and
                           s.cestado = e.cestado and
                           s.situacion = 'A' and
                           (s.cestado = 'PT' or s.cestado = 'PV' or s.cestado = 'PN' or (s.cestado = 'AP' and
                            s.fanulaper is null) or s.cestado = 'PP') and
                           (s.periodo=to_char(current_date,'yyyy') or s.periodo=to_char(current_date - INTERVAL '15
                            days','yyyy'))");
        $this->setOrderByForSearchQuery("s.situacion asc, e.destado asc, s.fecha_ini desc");

        /*Añadimos los Matching - Correspondecias campoTPL <-> campoBD*/

        $this->addMatching("nsolicitud","nsolicitud","tper_solicitudes");

        //Indicamos la Clave Primaria
        $this->addCamposClave(array("nsolicitud"));


        //Checkbox
        $checkDepuertos = new IgepCheckBox('depuertos');
        $checkDepuertos->setValueChecked('S');
        $checkDepuertos->setValueUnchecked('N');
        $this->addCheckBox($checkDepuertos);

        /*Ventanas de Seleccion*/
        //Usuarios interesados
        $interesados = new IgepVentanaSeleccion('DNI','INTERESADOS',array("DNI","Nombre"));
        $interesados->setDependencia(array('depuertos'),array('vper_ocupacion.depuertos'));
        $this->addVentanaSeleccion($interesados);

        /*Acciones de Interfaz*/
        $this->addAccionInterfaz('DNI','validaDNI');
        $this->addAccionInterfaz('depuertos','validaDNI');

        }//Fin de Constructor

        /************************** VALIDACIONES *********************************/


function preIniciarVentana($objDatos) {
        // El usuario se cargará de la siguiente manera:

        $usuarioConectado = IgepSession::dameDatosUsuario();
        $this->nregpgv = $usuarioConectado['nrp'];
        $this->dni = substr($usuarioConectado['nrp'],0,-2);
        $datosUsuario = $this->consultar('SELECT nregpgv,dnombre, dapell1, dapell2 FROM vper_ocupacion WHERE
                                                nregpgv=\''.$usuarioConectado['nrp'].'\'');
        $this->datosUsuario = $datosUsuario[0]['dapell1'].' '.$datosUsuario[0]['dapell2'].', '.$datosUsuario[0]['dnombre'];
        $this->informacionUsuarioActivo = substr($datosUsuario[0]['nregpgv'],0,-2).'-'.$datosUsuario[0]['dnombre'].'
                                        '.$datosUsuario[0]['dapell1'].' '.$datosUsuario[0]['dapell2'];
        //Segun el rol del usuario puede o no ver las solicitudes de otras personas
        if(IgepSession::dameRol()=='P_TRAMITA'){
          $this->esPersonal = 1;
          $this->informacionUsuarioConectado = $this->informacionUsuarioActivo;
          // Cambiamos la condición del where: Si el usuario es tramitador de personal, se deja anular solicitudes del
          periodo siguiente.
          $this->setWhereForSearchQuery( "s.cclase = p.cclase and
                           s.cestado = e.cestado and
                           s.situacion = 'A' and
                           (s.cestado = 'PT' or s.cestado = 'PV' or s.cestado = 'PN' or (s.cestado = 'AP' and
                            s.fanulaper is null) or s.cestado = 'PP') and
                           (s.periodo=to_char(current_date,'yyyy') or s.periodo=to_char(current_date - INTERVAL '15
                            days','yyyy') or s.periodo=to_char(current_date + INTERVAL '1 year','yyyy'))");
        }
        else{
          $this->esPersonal = 0;
          $filtro = $this->str_where." and  s.nregpgv = '".$this->nregpgv."' ";
          $select = $this->str_select.' WHERE '.$filtro.' ORDER BY '.$this->str_orderBy;
          $res = $this->consultar($select);
          $this->str_whereFiltro = ' WHERE '.$filtro;
          if($res!=-1){
                  if(count($res)>0){
                           foreach($res as $index=> $tupla){
                                   $color = 'none';
```

```
                                        if(($tupla['situacion']=='A' AND $tupla['cestado']=='AP' AND
                                                empty($tupla['fanulaper']) AND !empty($tupla['ffirmaresolucion']))
                                            OR
                                            ($tupla['situacion']=='A' AND $tupla['cestado']=='AP' AND !
                                             empty($tupla['fanulaper']) AND !empty($tupla['ffirmaresolucionanula']))
                                            OR
                                            ($tupla['situacion']=='B' AND $tupla['cestado']=='AP' AND !
                                             empty($tupla['ffirmaresolucionanula'])))

                                        $color='sugerencia';
                                    $objDatos->setRowColor($res[$index],$color);
                                }
                                $this->setResultadoBusqueda($res);
                        }else{
                                $mensaje = new IgepMensaje('APL-45');
                                IgepSession::guardaMensaje('principal',$mensaje);
                                return -1;
                        }
                    }
                }
                return 0;
}


function preBuscar($objDatos) {

                if($objDatos->getValue('DNI')!=''){
                        $this->dni = $objDatos->getValue('DNI');
                        //Pasamos DNI a nregpgv
                        $cons="SELECT nregpgv FROM vper_ocupacion where nregpgv like '".$this->dni."__'";
                        $res=$this->consultar($cons);
                        $this->nregpgv =$res['0']['nregpgv'];
                        //Cambiamos informaciÃ³n de usuario activo
                        $datosUsuario = $this->consultar('SELECT nregpgv,dnombre, dapell1, dapell2 FROM vper_ocupacion WHERE
                                                                                        nregpgv=\''.$this->nregpgv.'\'');
                        $this->datosUsuario = $datosUsuario[0]['dnombre'].' '.$datosUsuario[0]['dapell1'].'
                                                                                '.$datosUsuario[0]['dapell2'];
                        $this->informacionUsuarioActivo = substr($datosUsuario[0]['nregpgv'],0,-
                                2).'-'.$datosUsuario[0]['dnombre'].' '.$datosUsuario[0]['dapell1'].'
                                                                                '.$datosUsuario[0]['dapell2'];
                }
                $this->setParametrosBusqueda("s.nregpgv='".$this->nregpgv."'");

                $listaDePuertos = $this->getDefaultData('depuertos');
                $listaDePuertos['seleccionado']= $objDatos->getValue('depuertos');
                $this->addDefaultData('depuertos',$listaDePuertos);

                return 0;
}//Fin de PreBuscar


function postBuscar($objDatos){

                // Se deberán de mostrar solicitudes resaltadas con color verde cuando estén en las situaciones
                siguientes:
                // - La solicitud estÃ¡ autorizada (situacion = A, estado = AP, fanulaper nula, ffirmaresolucion no nula o
                //situacion = A, estado = AP, fanulaper no nula, ffirmaresolucionanula no nula   o
                //situacion = B, estado = AP, ffirmaresolucionanula no nula) :
                // Mostrar con fondo verde
                $m_datos = $objDatos->getAllTuplas();
                foreach($m_datos as $index=> $tupla){
                        $color = 'none';
                        if(($tupla['situacion']=='A' AND $tupla['cestado']=='AP' AND empty($tupla['fanulaper']) AND !
                                        empty($tupla['ffirmaresolucion']))
                                OR
                                ($tupla['situacion']=='A' AND $tupla['cestado']=='AP' AND !empty($tupla['fanulaper'])
                                        AND !empty($tupla['ffirmaresolucionanula']))
                                OR
                                ($tupla['situacion']=='B' AND $tupla['cestado']=='AP' AND !
                                        empty($tupla['ffirmaresolucionanula'])) )

                                $color='sugerencia';
                        $objDatos->setRowColor($m_datos[$index],$color);
                }
                $objDatos->setAllTuplas($m_datos);
                return 0;
}


function saltoDeVentana($objDatos, $objSalto){

        $objDatos->setOperacion('seleccionar');
        $m_datos = $objDatos->getAllTuplas();
        if($m_datos==0){
            $this->showMensaje('APL-10');
            return -1;
        }
        $objSalto->setClase('DetalleSolicitud');
        $objSalto->setAccion('buscar');
        $objSalto->setAccionRetorno('volverDeDetalleSolicitud');
        $objSalto->setParam('visibilidad','FirmaResponsable');
        $objSalto->setParam('entradaMenu','responsable');
        $perfil = IgepSession::dameRol();
        if ($perfil!='P_USUARIO')
           $objSalto->setParam('mostrarValidacion','1');
        $solicitudes = array();
        foreach($m_datos as $solicitud)
            array_push($solicitudes,$solicitud['nsolicitud']);
        $objSalto->setParam('solicitudes',$solicitudes);
```

```
        return 0;
    }


function accionesParticulares($str_accion, $objDatos) {
        switch ($str_accion)
        {
            case 'volverDeDetalleSolicitud':
                $actionForward = $objDatos->getForward('correcto');
                return $actionForward;
                break;
            case 'anularSolicitud':
                $objDatos->setOperacion('seleccionar');
                $nsolicitud = $objDatos->getValue('nsolicitud');
                if($nsolicitud!=''){
                    $res = $this->f_anularSolicitud($nsolicitud);
                    if($res==0)
                              $this->showMensaje('APL-15',array($nsolicitud));
                    else{
                        if($res==-1){
                              $actionForward = $objDatos->getForward('incorrecto');
                              return $actionForward;
                         }
                        if($this->obj_mensaje==null)
                            $this->showMensaje('APL-16',array($nsolicitud));
                    }
                    $this->refreshSearch();
                }
                else
                    $this->showMensaje('APL-21');
                $actionForward = $objDatos->getForward('correcto');
                return $actionForward;
            default:
                return -1;
        }
    }


//Validación DNI del interesado
function validaDNI($objDatos){
        $campoDisparador = $objDatos->getTriggerField();
        $nif = $objDatos->getValue('DNI');
        if($nif!='' AND $campoDisparador=='DNI'){
            $depuertos = $objDatos->getValue('depuertos');
            $res = $this->consultar( "SELECT nregpgv, dapell1 ||' '|| dapell2 ||', '|| dnombre as \"Nombre\" FROM " .
                    "vper_ocupacion WHERE nregpgv like '".$nif."__' and vper_ocupacion.depuertos='".$depuertos."'");
            if(count($res)>0){
                $objDatos->setValue('Nombre',$res[0]['Nombre']);
                $objDatos->setValue('nregpgv',$res[0]['nregpgv']);
            }
            else {
                $objDatos->setValue('Nombre','');
                $objDatos->setValue('nregpgv','');
                $objDatos->setValue('DNI','');
                $this->showMensaje('APL-01',array($nif));
                return -1;
            }
        }
        else{
            $objDatos->setValue('Nombre','');
            $objDatos->setValue('nregpgv','');
            $objDatos->setValue('DNI','');
        }
        return 0;
} //Fin de validaDNI


function f_anularSolicitud($pnsolicitud){

        //Recuperamos los datos de la BD de nuevo. Esto permite evitar error de concurrencia
        //f_listaSolicitudesAnular

        $res = $this->consultar("SELECT s.nregpgv,CASE WHEN s.ffirmaresolucion is not null and s.situacion ='A' and
                                        s.cestado='AP' and s.fanulaper is null THEN 'Autorizada'
                                WHEN s.ffirmaresolucionanula is not null and s.situacion ='A' and s.cestado='AP' and
                                        s.fanulaper is not null THEN 'No autorizada'ELSE e.destado END as \"destado\" ,
                                        s.situacion,s.nsolicitud,p.dclase,s.fecha_ini,s.fecha_fin,s.fnacimientomenor,
                                        s.cestado,s.ffirmaresp, s.ffirmaper, s.periodo, s.fsolicitud, o.dnombre,
                                        o.dapell1, o.dapell2, s.observaciones, s.avisos, s.justificante,
                                        o.cpro, o.cmun, s.cclase, s.fnacimientomenor, s.nrpresp, s.nrpper, s.nrpanularesp,
                                        s.fanularesp, s.motivonoautresp, s.nrpanulaper, s.fanulaper, s.motivonoautper,
                                        s.fresolucion, s.fresolucionanula, s.valida, s.validaanula, s.revisada, s.distancia,
                                        s.ffirmaresolucion, s.numresol, s.nrptramitador, s.fvalidatramitador, s.nrprevisor,
                                        s.fvalidarevisor,s.nrpgeneraresol, s.nrpnotificaresol
                                FROM TPER_SOLICITUDES s,tper_cod_permisos p, TPER_ESTADOSOL e, vper_ocupacion o
                                WHERE s.cclase = p.cclase and s.cestado = e.cestado and o.nregpgv = s.nregpgv and
                                        s.situacion = 'A' and (s.cestado = 'PT' or s.cestado = 'PV' or s.cestado = 'PN' or
                                                (s.cestado = 'AP' and s.fanulaper is null)  or s.cestado = 'PP') and
                                        s.nsolicitud=".$pnsolicitud."
                                ORDER BY s.situacion asc, e.destado asc, s.fecha_ini asc",array(
                                        'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA, 'fecha_fin'=>TIPO_FECHA,
                                        'fecha_fin'=>TIPO_FECHA,'fnacimientomenor'=>TIPO_FECHA, 'ffirmaresp'=>TIPO_FECHA,
                                        'ffirmaper'=>TIPO_FECHA,'fsolicitud'=>TIPO_FECHA,
                                        'fnacimientomenor'=>TIPO_FECHA,'fanularesp'=>TIPO_FECHA,
                                        'fanulaper'=>TIPO_FECHA,'fresolucion'=>TIPO_FECHA,'fresolucionanula'=>TIPO_FECHA,
                                        'ffirmaresolucion'=>TIPO_FECHA,'fvalidatramitador'=>TIPO_FECHA,
                                        'fvalidarevisor'=>TIPO_FECHA,)) );

          if($res==-1 or count($res)==0){
```

```
      $this->showMensaje('APL-13');
      return -1;
    }
    $solicitud= $res[0];

    //Creamos la conexion a la BD
    $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
    $conPostgres = new IgepConexion($g_dsn, true);
    $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
    $conOracle = new IgepConexion($g_oracle_dsn);
    $conPostgres->empezarTransaccion();
    $conOracle->empezarTransaccion();

    $error = 0;
    $listaSolicitudesAnulacion = null;
    $listaSolicitudesAnulacionResueltas = null;

    // Comprobar que antes de anular o borrar una solicitud de VAC, no se quedan los ADI consecutivos a Ã©sta sin anular
    $pcclase = $res[0]['cclase'];
    $pnregpgv = $res[0]['nregpgv'];
    $pfechaIni = $res[0]['fecha_ini'];
    $pfechaFin = $res[0]['fecha_fin'];
    $pperiodo = $res[0]['periodo'];

    $mensaje = Solicitudes::f_anularBorrarVAC($pnregpgv,$pcclase,$pfechaIni,$pfechaFin,$pperiodo,'S');

    if($mensaje!='0'){
              $this->showMensaje('APL-GENERICO',array($mensaje));
              return -1;
    }


    //Si fecha de inicio < fecha del día, comprobaremos que existan fichajes o incidencias para poderla anular:
    $varHoy = new gvHidraTimestamp();
    $varHoy->setTime(0,0,0);

    // inicialización de noficha
    $noficha=false;

    //Sacamos de esta condición el mostrar los mensajes APL-GENERICO y el APL-14 para ponerlos en otra condición
    if(gvHidraTimestamp::cmp($varHoy, $solicitud['fecha_ini']) < 0){

              //Comprobamos que la persona ficha, ya que hay personas que están en la aplicación de personal(puertos,
              vigilantes,
              //y no fichan
              $ultimoFichaje= Personal::f_obtenerUltimoFichaje($solicitud['nregpgv']);
              if(!empty($ultimoFichaje)){
                   $noficha=false;
               }
               else
                  $noficha=true;
    }

    // 26/03/2010 INICIO : Los usuarios de personal pueden anular solicitudes aunque ya estén disfrutadas: Los marcamos
    como si no ficharan
    if(IgepSession::dameRol()=='P_TRAMITA' OR IgepSession::dameRol()=='P_REVISOR'OR IgepSession::dameRol()=='P_RESPPER')
    {
              $noficha=true;
     }

  // Sólo cuando la solicitud acabe en el pasado y fichamos, realizamos la comprobación de si hay fichajes o no
    if(gvHidraTimestamp::cmp($varHoy, $solicitud['fecha_ini']) < 0 AND
                                        gvHidraTimestamp::cmp($varHoy, $solicitud['fecha_fin']) < 0 ){

              if ($noficha==false) {
                        $hayFichaje = true;
                        //Comprobamos que se haya fichado o tenido incidencia en un intervalo de fechas
                        $hayFichaje = Personal::f_existeAlgunFichajeoIncidencia($solicitud['fecha_ini'],
                                             $solicitud['fecha_fin'], $solicitud['nregpgv'],$resultado);
                        if($hayFichaje<0){
                                  $this->showMensaje('APL-GENERICO',array($resultado['mensaje']));
                          return -1;
                        }
                        //Fin f_obtenerFichajesIncidencias
                         if(!$hayFichaje){
                                  $this->showMensaje('APL-14',array($pnsolicitud));
                                  return -1;
                         }
              }
    }



     // Inicialización de las variables con información de la solicitud a anular original
    $var_fecha_ini =  $solicitud['fecha_ini'];
    $var_fecha_fin =   $solicitud['fecha_fin'];
    $var_estado_original =   $solicitud['cestado'];
    $var_ffirmaresp =   $solicitud['ffirmaresp'];
    $var_FirmadoPorALR =   $solicitud['nrpresp'];
    $var_ffirmaper =   $solicitud['ffirmaper'];
    $var_FirmadaPorALP =   $solicitud['nrpper'];
    $var_periodo =   $solicitud['periodo'];
    $var_fnacimientomenor =   $solicitud['fnacimientomenor'];
    $var_fsolicitud =   $solicitud['fsolicitud'];
    $var_observaciones = $solicitud['observaciones'];
    $var_nrpresp = $solicitud['nrpresp'];
    $var_nrpper = $solicitud['nrpper'];
    $var_nrpanularesp = $solicitud['nrpanularesp'];
    $var_fanularesp =   $solicitud['fanularesp'];
    $var_motivonoautresp = $solicitud['motivonoautresp'];
    $var_nrpanulaper = $solicitud['nrpanulaper'];
    $var_fanulaper =   $solicitud['fanulaper'];
```

```
$var_motivonoautper = $solicitud['motivonoautper'];
$var_fresolucion =   $solicitud['fresolucion'];
$var_fresolucionanula =   $solicitud['fresolucionanula'];
$var_valida = $solicitud['valida'];
$var_validaanula = $solicitud['validaanula'];
$var_revisada = $solicitud['revisada'];
$var_justificante = $solicitud['justificante'];
$var_distancia = $solicitud['distancia'];


$var_ffirmaresolucion = $solicitud['ffirmaresolucion'];
$var_numresol = $solicitud['numresol'];
$var_nrptramitador = $solicitud['nrptramitador'];
$var_fvalidatramitador = $solicitud['fvalidatramitador'];
$var_nrprevisor = $solicitud['nrprevisor'];
$var_fvalidarevisor = $solicitud['fvalidarevisor'];
$var_nrpgeneraresol = $solicitud['nrpgeneraresol'];
$var_nrpnotificaresol = $solicitud['nrpnotificaresol'];



$var_gradoparentesco = $solicitud['gradoparentesco'];


//f_fin = MIN(sysdate-1, fecha_fin);
$fechaComp1 = new gvHidraTimestamp("yesterday");
$fechaComp1->setTime(0,0,0);
if($fechaComp1<$var_fecha_fin)
        $f_fin = $fechaComp1;
else
        $f_fin = $var_fecha_fin;
if(!$noficha){
        $fechas= Personal::f_IntervalosNoFichados($var_fecha_ini, $f_fin, $solicitud['nregpgv'],$resultado);
        if($fechas<0){
                $this->showMensaje('APL-GENERICO',array($resultado['mensaje']));
                return -1;
        }
}

else
        $fechas=array();

// Con este flag indicamos si hemos modificado la solicitud de anulación
$anulacion_modificada = false;

if(count($fechas)==0){
  // No se ha disfrutado ningún dia, se puede anular la solicitud
  // Anula la solicitud actualizando su estado si la solicitud está firmada o
  // borra el alta de solicitud si no está firmada
  if($var_estado_original == 'AP'){
      $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                SET   situacion = 'B', cestado = 'PR'
                WHERE nsolicitud = ".$solicitud['nsolicitud']);
  }
  if($var_estado_original == 'PN' OR $var_estado_original == 'PP' OR $var_estado_original == 'PT' OR
        $var_estado_original == 'PV'){
      $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                SET   situacion = 'X', cestado = 'PR'
                WHERE nsolicitud = ".$solicitud['nsolicitud']);
  }
  if($res!=-1){ //Si no hay error

        $nofichan = UsuariosDePersonal::f_noFichan($solicitud['nregpgv']);
        if(!$nofichan){
             //Obtener el centro, departamento y grupo al que pertenece el usuario que realiza la anulación de
                solicitud
             $grupo = Personal::f_obtenerGrupo($solicitud['nregpgv']);
             if(count($grupo)==0){
                  $this->showMensaje('APL-32');
                  return -1;
             }

             // Obtenemos el centro, departamento y grupo de lo obtenido en "grupo"
             $var_centro = $grupo['centro'];
             $var_departamento = $grupo['departamento'];
             $var_grupo = $grupo['seccion'];

             // Si el usuario que realiza la solicitud es responsable de firma no suplente no se envía ningún
             // correo, ya que solo el podrí firmar sus anulaciones de solicitud

             if(Firmantes::f_esResponsableNoSuplente($solicitud['nregpgv'],$var_centro,$var_departamento,
                                                                                        $var_grupo)){
                  // No se envia correo
                  $enviocorreo=false;
             }
             else{
                  //Obtener los responsables de ese grupo


                  $destinatarios=Firmantes::f_dameResponsables($grupo,$var_fecha_ini,
                                                        $solicitud['fsolicitud'],'N');
                  $enviocorreo=true;
             }
        }
        else{
                // Obtener el grupo al que pertenecen los usuarios que no fichan
                // f_dameCentroTrab
                //Permitir que en un centro donde no se fiche, el responsable de firma pueda ser alguien que no
                fiche (f_anularSolicitud)
                $var_ccentrab = UsuariosDePersonal::f_dameCentroTrab($solicitud['nregpgv'], $ccentrab);

                UsuariosDePersonal::f_obtenerCdgCserv($datos['datosSoliPer']['solicitud']['nregpgv'],
```

```
                                                                                                $pcdg,$pcserv);
                        Firmantes::f_obtenerFirmanteOEmpleadoGrupoNoFichan($ccentrab, $solicitud['nregpgv'],$cdg,
                                                                $cserv, $ficha, $firmanteGrupoNoFichan);

                        if(!empty($firmanteGrupoNoFichan) or $firmanteGrupoNoFichan!='' ){
                                if ($ficha=='S'){
                                        $grupoNoFichan = Personal::f_obtenerGrupo($firmanteGrupoNoFichan);
                                        if ($grupoNoFichan!='') {
                                          // Se obtienen los responsables del grupo correspondiente

                                          $destinatarios=Firmantes::f_dameResponsables($grupoNoFichan,
                                                        $var_fecha_ini,$solicitud['fsolicitud'],'N');
                                        }
                                        else{
                                          $this->showMensaje('APL-32');
                                          return -1;
                                        }
                                }
                                else{
                                        $destinatarios=$firmanteGrupoNoFichan;
                                }

                        }
                        else{
                                //dar mensaje
                                $this->showMensaje('APL-94');
                                return -1;
                        }


                        $enviocorreo=true;

                }
                if ($enviocorreo) {

                //-------------------------------------------------------------------------------------
                                        //Se envía correo a los responsables de firma

                //-------------------------------------------------------------------------------------

                $mensaje = "";

                //Limpiamos tags HTML del aviso
                $avisoSinHtml = str_replace ("</LI>","\n",$solicitud['avisos']);
                $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
                $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
                $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
                $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
                $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);

                //Generar mensaje a enviar en el correo
                Solicitudes::f_GenerarMensajeCorreo($solicitud['nregpgv'],
                       $solicitud['dapell1'] . " " . $solicitud['dapell2'] . ", " . $solicitud['dnombre'],
                       $solicitud['cclase'], $solicitud['dclase'], $var_periodo, $var_fecha_ini, $var_fecha_fin,
                       $solicitud['observaciones'],'',$avisoSinHtml, "N","N","N","N","N","N",
                                "N", null, null, $solicitud['nsolicitud'], $mensaje);

                //Enviar correo a los responsables
                Solicitudes::f_enviarCorreo($solicitud['nregpgv'],$destinatarios,$mensaje,"PERSONAL - Anulación de
                                                                                                solicitud");
                }


        } // fin if Si no hay error
        else{ // hay error
            //Error al actualizar en tper_solicitudes
            $error=1;
        }
    }//Fin if: No se ha consumido nada
    else{ // count($fechas)!=0
        // Se ha disfrutado algún dia, habrá que modificar la parte de la solicitud que aún no se ha disfrutado,
        // y crear tantas solicitudes nuevas como intervalos que ya se hayan disfrutado
        //f_esDePuertos: Comprobamos si es usuario de puertos
        $res2 = $this->consultar('SELECT nregpgv
                                FROM VPER_OCUPACION
                                WHERE nregpgv = \''.$solicitud['nregpgv'].'\' AND
                                depuertos = \'S\'');
        if($res2[0]['nregpgv']!='')
            $esDePuertos = TRUE;
        else
            $esDePuertos = FALSE;


        if(!$esDePuertos){
                //conexion oracle
                $resFtoma = $conOracle->consultar('SELECT ftpos as "ftpos"
                                                FROM tper_puestos_trab_admon
                                                WHERE nregpgv = \''.$solicitud['nregpgv'].'\' and fbajpad is
                                                        null',array('DATATYPES'=>array('ftpos'=>TIPO_FECHA)));
                $ftoma = $resFtoma[0]['ftpos'];
                $esDePuertos1 = 'N';
      }
      else{
                $esDePuertos1 = 'S';
                $ftoma = new gvHidraTimestamp('2200/01/01');
                $ftoma->setTime(0,0,0);
      }
      $diasTrabajados = null;
      $diasTrabajadosVV6 = null;
      UsuariosDePersonal::f_obtenerDiasTrabajados($solicitud['nregpgv'], $var_periodo, $esDePuertos1,
                                                        $ftoma, $diasTrabajados, $diasTrabajadosVV6);
```

```
//Insertamos en un array de fechas, aquellos intervalos donde se haya fichado
$fechas_fichadas = Personal::f_IntervalosFichados($var_fecha_ini, $f_fin,$solicitud['nregpgv'],$resultado);
if($fechas_fichadas<0){

          $this->showMensaje('APL-GENERICO',array($resultado['mensaje']));
               return -1;
}

// ACTUALIZACIÓN DE LA SOLICITUD CON EL INTERVALO A ANULAR

// Si la solicitud a anular termina en una fecha posterior a la del dÃa actual, entonces se deberÃ¡ modificar
// la solicitud a anular para que empiece el dÃa siguiente y termine en la fecha final original
$varHoy = new gvHidraTimestamp();
$varHoy->setTime(0,0,0);

if(gvHidraTimestamp::cmp($varHoy, $var_fecha_fin) >= 0){
     if($var_estado_original == 'AP')
          $var_situacion='B';
     elseif($var_estado_original == 'PP' OR $var_estado_original == 'PN' OR $var_estado_original == 'PT' OR
                                                                                $var_estado_original == 'PV')
          $var_situacion='X';

     $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                              SET  fecha_ini = current_date ,
                                   fecha_fin = '".$conPostgres->prepararFecha($var_fecha_fin)."',
                                   altamodificada = 'S',
                                   situacion = '".$var_situacion."',
                                   cestado = 'PR'
                              WHERE nsolicitud = ".$solicitud['nsolicitud']);
     if($res!=-1){
          // Con este flag indicamos que ya hemos modificado la solicitud de anulaciÃ³n
          $anulacion_modificada = true;

          $anulacion_futura_modificada = true;

          // ACTUALIZACIÃ“N DE FECHAS DE LA SOLICITUD A ANULAR EN LA TABLA DE PERMISOS
     }
     else{//Error al actualizar TPER_SOLICITUDES
          $error=1;
     }
 }//Fin if (fecha_fin>=current_date)
 $nulo=null;
 // CREACIÓN DE NUEVAS SOLICITUDES CON INTERVALOS YA DISFRUTADOS EN EL PASADO
 $cuenta_fechas = count($fechas);
 for($i=0; $i<=$cuenta_fechas-1;$i+=2){
      if($anulacion_modificada){
          $resultado = null;

          $datosSolicitud=Solicitudes::f_altaSolicitud($solicitud['nregpgv'],$var_periodo,$fechas[$i],$fechas[$i+1],
                           $var_observaciones,$solicitud['fnacimientomenor'],
                           $altaModificada='S',$var_estado_original,$var_ffirmaresp,$var_FirmadoPorALR,
                           $var_ffirmaper,$var_FirmadaPorALP,$psituacion='A',$nulo,$nulo,
                           $nulo,$nulo,$nulo,$resultado,$solicitud['cclase'],$var_distancia,
                           $listaSolicitudesAnulacion, $listaSolicitudesAnulacionResueltas,
                           $var_gradoparentesco, $var_justificante);


          if($datosSolicitud<0){
              //Parche hasta que se arregle en igep el tema de poner intro en los mensajes
              $mensaje = str_replace("\n",'',$resultado['mensaje']);
              $this->showMensaje('APL-17',array($pnsolicitud,$mensaje));
              $error=1;
              break;
          }
          else{
              $var_solicitud = $datosSolicitud['solicitud'];

              //Añadimos los campos que faltan a la estrucutra $var_solicitud

              $var_solicitud['fsolicitud'] = $var_fsolicitud;
              $var_solicitud['observaciones'] =$var_observaciones;
              $var_solicitud['nrpresp'] =$var_nrpresp;
              $var_solicitud['nrpper'] =$var_nrpper;
              $var_solicitud['nrpanularesp'] = $var_nrpanularesp;
              $var_solicitud['fanularesp'] = $var_fanularesp;
              $var_solicitud['motivonoautresp'] = $var_motivonoautresp;
              $var_solicitud['nrpanulaper'] = $var_nrpanulaper;
              $var_solicitud['fanulaper'] = $var_fanulaper;
              $var_solicitud['motivonoautper'] = $var_motivonoautper;
              $var_solicitud['fresolucion'] = $var_fresolucion;
              $var_solicitud['fresolucionanula'] = $var_fresolucionanula;
              $var_solicitud['valida'] = $var_valida;
              $var_solicitud['validaanula'] = $var_validaanula;
              $var_solicitud['revisada'] = $var_revisada;
              $var_solicitud['nsolanulacion'] = $pnsolicitud;

              $var_solicitud['ffirmaresolucion'] = $var_ffirmaresolucion ;
              $var_solicitud['numresol'] = $var_numresol;
              $var_solicitud['nrptramitador'] = $var_nrptramitador;
              $var_solicitud['fvalidatramitador'] = $var_fvalidatramitador;
              $var_solicitud['nrprevisor'] = $var_nrprevisor;
              $var_solicitud['fvalidarevisor'] = $var_fvalidarevisor;
              $var_solicitud['nrpgeneraresol'] = $var_nrpgeneraresol;
              $var_solicitud['nrpnotificaresol'] = $var_nrpnotificaresol;



              //Antes de insertar la solicitud en TPER_SOLICITUDES se deberá generar la secuencia (sper_solicitud)
              //para el nº de solicitud.
              $nuevaSolicitud = $conPostgres->calcularSecuenciaBD('sper_solicitud');

              //Transformo fechas a formato BD
```

```
        $var_solicitud['fsolicitud'] = $conPostgres->prepararFecha($var_solicitud['fsolicitud']);
        $var_solicitud['fecha_ini']=$conPostgres->prepararFecha($var_solicitud['fecha_ini']);
        $var_solicitud['fecha_fin']=$conPostgres->prepararFecha($var_solicitud['fecha_fin']);
        $var_solicitud['ffirmaresp']=$conPostgres->prepararFecha($var_solicitud['ffirmaresp']);
        $var_solicitud['ffirmaper']=$conPostgres->prepararFecha($var_solicitud['ffirmaper']);
        $var_solicitud['fanularesp']=$conPostgres->prepararFecha($var_solicitud['fanularesp']);
        $var_solicitud['fanulaper']=$conPostgres->prepararFecha($var_solicitud['fanulaper']);
        $var_solicitud['fresolucion']=$conPostgres->prepararFecha($var_solicitud['fresolucion']);
        $var_solicitud['fresolucionanula']=$conPostgres->prepararFecha($var_solicitud['fresolucionanula']);
        $var_solicitud['fnacimientomenor']=$conPostgres->prepararFecha($var_solicitud['fnacimientomenor']);

        $var_solicitud['ffirmaresolucion']=$conPostgres->prepararFecha($var_solicitud['ffirmaresolucion']);
        $var_solicitud['fvalidatramitador']=$conPostgres->prepararFecha($var_solicitud['fvalidatramitador']);
        $var_solicitud['fvalidarevisor']=$conPostgres->prepararFecha($var_solicitud['fvalidarevisor']);


        if($nuevaSolicitud!=-1){
                  if($var_solicitud['fecha_fin']=='')
                      unset($var_solicitud['fecha_fin']);
                  if($var_solicitud['ffirmaresp']=='')
                      unset($var_solicitud['ffirmaresp']);
                  if($var_solicitud['ffirmaper']=='')
                      unset($var_solicitud['ffirmaper']);
                  if($var_solicitud['fanularesp']=='')
                      unset($var_solicitud['fanularesp']);
                  if($var_solicitud['fanulaper']=='')
                      unset($var_solicitud['fanulaper']);
                  if($var_solicitud['fresolucion']=='')
                      unset($var_solicitud['fresolucion']);
                  if($var_solicitud['fresolucionanula']=='')
                      unset($var_solicitud['fresolucionanula']);
                  if($var_solicitud['fnacimientomenor']=='')
                      unset($var_solicitud['fnacimientomenor']);
                  if($var_solicitud['valida']=='')
                      unset($var_solicitud['valida']);
                                                  if($var_solicitud['validaanula']=='')
                      unset($var_solicitud['validaanula']);
                  if($var_solicitud['revisada']=='')
                      unset($var_solicitud['revisada']);
                  if($var_solicitud['motivonoautresp']=='')
                      unset($var_solicitud['motivonoautresp']);
                  if($var_solicitud['motivonoautper']=='')
                      unset($var_solicitud['motivonoautper']);
                  if($var_solicitud['nrpanulaper']=='')
                      unset($var_solicitud['nrpanulaper']);
                  if($var_solicitud['nrpanularesp']=='')
                      unset($var_solicitud['nrpanularesp']);
                  if($var_solicitud['nrpper']=='')
                      unset($var_solicitud['nrpper']);
                  if($var_solicitud['nrpresp']=='')
                      unset($var_solicitud['nrpresp']);
                  if($var_solicitud['avisos']=='')
                      unset($var_solicitud['avisos']);

                  // 16/04/2010 INICIO : AÃ±adir campos que faltaba incluir para el alta modificada
                  if($var_solicitud['ffirmaresolucion']=='')
                      unset($var_solicitud['ffirmaresolucion']);
                  if($var_solicitud['numresol']=='')
                      unset($var_solicitud['numresol']);
                  if($var_solicitud['nrptramitador']=='')
                      unset($var_solicitud['nrptramitador']);
                  if($var_solicitud['fvalidatramitador']=='')
                      unset($var_solicitud['fvalidatramitador']);
                  if($var_solicitud['nrprevisor']=='')
                      unset($var_solicitud['nrprevisor']);
                  if($var_solicitud['fvalidarevisor']=='')
                      unset($var_solicitud['fvalidarevisor']);
                  if($var_solicitud['nrpgeneraresol']=='')
                      unset($var_solicitud['nrpgeneraresol']);
                  if($var_solicitud['nrpnotificaresol']=='')
                      unset($var_solicitud['nrpnotificaresol']);
                  // 16/04/2010 FIN
                  if($var_solicitud['nsolanulacion']=='')
                            unset($var_solicitud['nsolanulacion']);

                  $indices = array_keys($var_solicitud);
                  $res = $conPostgres->operar('INSERT INTO TPER_SOLICITUDES
                                                (nsolicitud,'.implode($indices,',').')
                                        VALUES(\''.$nuevaSolicitud.'\',\''.implode($var_solicitud,"','").'\') ');
                if($res==-1)
                      $error=1;
        }
        else
            $error=1;
    }
}//Fin if($anulacion_modificada)
else{

    // La solicitud no se ha modificado aÃºn. La modificamos para el intervalo actual
    $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                    SET  fecha_ini = '".$conPostgres->prepararFecha($fechas[$i])."',
                         fecha_fin = '".$conPostgres->prepararFecha($fechas[$i+1])."',
                         altamodificada = 'S',
                         situacion = 'A',
                         cestado = '".$var_estado_original."'
                    WHERE nsolicitud = ".$solicitud['nsolicitud']);
    if($res==-1){
            $error=1;
            break;
    }
    $anulacion_modificada=true;
}//Fin else
```

```
    }//Fin For

$nulo=null;
if($error==0){
          // CREACIÓN DE NUEVAS SOLICITUDES DE ANULACIÓN CON INTERVALOS NO DISFRUTADOS EN EL PASADO
          $cuenta_fechas_fichadas = count($fechas_fichadas);
          for($i=0; $i<=$cuenta_fechas_fichadas-1;$i+=2){
                if($var_estado_original == 'AP')
                    $var_situacion='B';
                elseif($var_estado_original == 'PP' OR $var_estado_original == 'PN' OR $var_estado_original == 'PT'
                                              OR $var_estado_original == 'PV')
                    $var_situacion='X';


                if(gvHidraTimestamp::cmp($varHoy, $fechas_fichadas[$i+1] + 1) >= 0 && $anulacion_futura_modificada){

                    // Modificamos la anulación del futuro para concatenarla con el intervalo actual que está en
                    el pasado
                    $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                                    SET  fecha_ini = '".$conPostgres->prepararFecha($fechas_fichadas[$i])."'
                                    WHERE nsolicitud = ".$solicitud['nsolicitud']);
                    if($res===-1){
                            $error=1;
                            break;
                    }

                }
                else{
                    $datosSolicitud=Solicitudes::f_altaSolicitud($solicitud['nregpgv'],$var_periodo,
                                     $fechas_fichadas[$i],$fechas_fichadas[$i+1],
                                     $var_observaciones,$solicitud['fnacimientomenor'],$altaModificada='S','PR',
                                     $var_ffirmaresp,$var_FirmadoPorALR,$var_ffirmaper,$var_FirmadaPorALP,
                                     $var_situacion,$nulo,$nulo,$nulo,$nulo,$nulo,$resultado,
                                     $solicitud['cclase'],$var_distancia,$listaSolicitudesAnulacion,
                                     $listaSolicitudesAnulacionResueltas,$var_justificante);

                    if($datosSolicitud<0){
                              $mensaje = str_replace("\n",'',$resultado['mensaje']);
                              $this->showMensaje('APL-17',array($pnsolicitud,$mensaje));
                              $error=1;
                              break;
                    }
                    else {

                              $var_solicitud = $datosSolicitud['solicitud'];

                              //Antes de insertar la solicitud en TPER_SOLICITUDES se deberá generar la secuencia
                              (sper_solicitud)
                              //para el nº de solicitud.
                              $nuevaSolicitud = $conPostgres->calcularSecuenciaBD('sper_solicitud');


                              //Añadir campos que faltaba incluir para el alta modificada
                              //Añadimos los campos que faltan a la estrucutra $var_solicitud

                              $var_solicitud['fsolicitud'] = $var_fsolicitud;
                              $var_solicitud['observaciones'] =$var_observaciones;
                              $var_solicitud['nrpresp'] =$var_nrpresp;
                              $var_solicitud['nrpper'] =$var_nrpper;
                              $var_solicitud['nrpanularesp'] = $var_nrpanularesp;
                              $var_solicitud['fanularesp'] = $var_fanularesp;
                              $var_solicitud['motivonoautresp'] = $var_motivonoautresp;
                              $var_solicitud['nrpanulaper'] = $var_nrpanulaper;
                              $var_solicitud['fanulaper'] = $var_fanulaper;
                              $var_solicitud['motivonoautper'] = $var_motivonoautper;
                              $var_solicitud['fresolucion'] = $var_fresolucion;
                              $var_solicitud['fresolucionanula'] = $var_fresolucionanula;
                              $var_solicitud['valida'] = $var_valida;
                              $var_solicitud['validaanula'] = $var_validaanula;
                              $var_solicitud['revisada'] = $var_revisada;
                              $var_solicitud['nsolanulacion'] = $pnsolicitud;
                              $var_solicitud['ffirmaresolucion'] = $var_ffirmaresolucion ;
                              $var_solicitud['numresol'] = $var_numresol;
                              $var_solicitud['nrptramitador'] = $var_nrptramitador;
                              $var_solicitud['fvalidatramitador'] = $var_fvalidatramitador;
                              $var_solicitud['nrprevisor'] = $var_nrprevisor;
                              $var_solicitud['fvalidarevisor'] = $var_fvalidarevisor;
                              $var_solicitud['nrpgeneraresol'] = $var_nrpgeneraresol;
                              $var_solicitud['nrpnotificaresol'] = $var_nrpnotificaresol;


                              //Transformo fechas a formato BD
                        $var_solicitud['fsolicitud'] = $conPostgres->prepararFecha($var_solicitud['fsolicitud']);
                        $var_solicitud['fecha_ini']=$conPostgres->prepararFecha($var_solicitud['fecha_ini']);
                        $var_solicitud['fecha_fin']=$conPostgres->prepararFecha($var_solicitud['fecha_fin']);
                        $var_solicitud['ffirmaresp']=$conPostgres->prepararFecha($var_solicitud['ffirmaresp']);
                        $var_solicitud['ffirmaper']=$conPostgres->prepararFecha($var_solicitud['ffirmaper']);
                        $var_solicitud['fanularesp']=$conPostgres->prepararFecha($var_solicitud['fanularesp']);
                        $var_solicitud['fanulaper']=$conPostgres->prepararFecha($var_solicitud['fanulaper']);
                        $var_solicitud['fresolucion']=$conPostgres->prepararFecha($var_solicitud['fresolucion']);
                        $var_solicitud['fresolucionanula']=$conPostgres-
                                                          >prepararFecha($var_solicitud['fresolucionanula']);
                        $var_solicitud['fnacimientomenor']=$conPostgres-
                                                          >prepararFecha($var_solicitud['fnacimientomenor']);

                        //Añadir campos que faltaba incluir para el alta modificada
                        $var_solicitud['ffirmaresolucion']=$conPostgres-
                                                          >prepararFecha($var_solicitud['ffirmaresolucion']);
                        $var_solicitud['fvalidatramitador']=$conPostgres-
                                                          >prepararFecha($var_solicitud['fvalidatramitador']);
                        $var_solicitud['fvalidarevisor']=$conPostgres-
                                                          >prepararFecha($var_solicitud['fvalidarevisor']);
```

```
                                    if($nuevaSolicitud!=-1){
                                        if($var_solicitud['fecha_fin']=='')
                                            unset($var_solicitud['fecha_fin']);
                                        if($var_solicitud['ffirmaresp']=='')
                                            unset($var_solicitud['ffirmaresp']);
                                        if($var_solicitud['ffirmaper']=='')
                                            unset($var_solicitud['ffirmaper']);
                                        if($var_solicitud['fanularesp']=='')
                                            unset($var_solicitud['fanularesp']);
                                        if($var_solicitud['fanulaper']=='')
                                            unset($var_solicitud['fanulaper']);
                                        if($var_solicitud['fresolucion']=='')
                                            unset($var_solicitud['fresolucion']);
                                        if($var_solicitud['fresolucionanula']=='')
                                            unset($var_solicitud['fresolucionanula']);
                                        if($var_solicitud['fnacimientomenor']=='')
                                            unset($var_solicitud['fnacimientomenor']);
                                        if($var_solicitud['valida']=='')
                                            unset($var_solicitud['valida']);
                                                                        if($var_solicitud['validaanula']=='')
                                            unset($var_solicitud['validaanula']);
                                        if($var_solicitud['revisada']=='')
                                            unset($var_solicitud['revisada']);
                                        if($var_solicitud['motivonoautresp']=='')
                                            unset($var_solicitud['motivonoautresp']);
                                        if($var_solicitud['motivonoautper']=='')
                                            unset($var_solicitud['motivonoautper']);
                                        if($var_solicitud['nrpanulaper']=='')
                                            unset($var_solicitud['nrpanulaper']);
                                        if($var_solicitud['nrpanularesp']=='')
                                            unset($var_solicitud['nrpanularesp']);
                                        if($var_solicitud['nrpper']=='')
                                            unset($var_solicitud['nrpper']);
                                        if($var_solicitud['nrpresp']=='')
                                            unset($var_solicitud['nrpresp']);
                                        if($var_solicitud['avisos']=='')
                                            unset($var_solicitud['avisos']);

                                        // 16/04/2010 INICIO : Añadir campos que faltaba incluir para el alta modificada
                                        if($var_solicitud['ffirmaresolucion']=='')
                                            unset($var_solicitud['ffirmaresolucion']);
                                        if($var_solicitud['numresol']=='')
                                            unset($var_solicitud['numresol']);
                                        if($var_solicitud['nrptramitador']=='')
                                            unset($var_solicitud['nrptramitador']);
                                        if($var_solicitud['fvalidatramitador']=='')
                                            unset($var_solicitud['fvalidatramitador']);
                                        if($var_solicitud['nrprevisor']=='')
                                            unset($var_solicitud['nrprevisor']);
                                        if($var_solicitud['fvalidarevisor']=='')
                                            unset($var_solicitud['fvalidarevisor']);
                                        if($var_solicitud['nrpgeneraresol']=='')
                                            unset($var_solicitud['nrpgeneraresol']);
                                        if($var_solicitud['nrpnotificaresol']=='')
                                            unset($var_solicitud['nrpnotificaresol']);

                                        // 16/04/2010 FIN
                                        if($var_solicitud['nsolanulacion']=='')
                                            unset($var_solicitud['nsolanulacion']);

                                        $indices = array_keys($var_solicitud);
                                        $res = $conPostgres->operar('INSERT INTO TPER_SOLICITUDES
                                                                    (nsolicitud,'.implode($indices,',').')
                                              VALUES(\''.$nuevaSolicitud.'\',\''.implode($var_solicitud,"','").'\') ');
                                        if($res==-1)
                                            $error=1;
                                    }
                                    else
                                        $error=1;
                                }//else

                            }//else

                        }//for
                    }
                }//Fin de else: se ha consumido algo
                $conPostgres->acabarTransaccion($error);
                $conOracle->acabarTransaccion($error);
                return $error;
            } //Fin f_anularSolicitud

        }//Fin AnularSolicitudes
        ?>
```

Tpl – Anular solicitudes y permisos

```
{CWVentana tipoAviso=$smty_tipoAviso  codAviso=$smty_codError  descBreve = $smty_descBreve  textoAviso=$smty_textoAviso
                                                                                    onLoad=$smty_jsOnLoad}
{CWBarra customTitle=$smty_customTitle usuario=$smty_usuario codigo=$smty_codigo}
```

```
        {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}

<!--********** PANEL fil *****************-->
        {CWPanel id="fil" action="buscar" method="post" estado="$estado_fil"  claseManejadora="AnularSolicitudes"}
                {CWBarraSupPanel titulo="AnulaciÃ³n de solicitudes de permisos y licencias"}
                        {CWBotonTooltip imagen="04" titulo="Limpiar campos" funcion="limpiar" actuaSobre="ficha"}
                {/CWBarraSupPanel}
                {CWContenedor}
                        {CWFicha}
                                <br>
                                {if $smty_esPersonal eq 1}
                                        {CWCheckBox nombre="depuertos"
                                                dataType=$dataType_AnularSolicitudes.depuertos
                                                valor=$defaultData_AnularSolicitudes.depuertos editable="true"
                                                actualizaA="dispara_accion_interfaz" textoAsociado="Usuarios de
                                                puertos (sustituciones)"}
                                        <br/><br/>
                                        {CWCampoTexto nombre="DNI"  maxlength="8" size="8" editable="true"
                                                textoAsociado="Interesado" actualizaA="x" value="$smty_dni"
                                                obligatorio="true"}
                                        {CWBotonTooltip imagen="13" titulo="Busqueda Interesado" funcion="abrirVS"
                                                actuaSobre="DNI" formActua="fil" panelActua="fil"
                                                claseManejadora="ConsultaSolicitudesACargoResponsable"}
                                         {CWCampoTexto nombre="Nombre" size="40" editable="false"
                                                                                value="$smty_usuario1"}
                                        {*CWCampoTexto nombre="nregpgv" oculto="true" value="$smty_nregpgv"*}
                                {else}
                                         {CWCampoTexto nombre="Nombre" size="40" value="$smty_usuario1"
                                                                editable="false" textoAsociado="Interesado"}
                                        {*CWCampoTexto nombre="nregpgv" oculto="true" value="$smty_nregpgv"*}

                                {/if}
                                <br><br>
                        {/CWFicha}
                {/CWContenedor}
                {CWBarraInfPanel}
                        {CWBoton imagen="50" texto="Buscar" class="boton" accion="buscar"}
                {/CWBarraInfPanel}
        {/CWPanel}

<!-- ***************** PANEL lis ***********************-->
        {CWPanel id="lis" action="borrar" method="post" estado="$estado_lis" claseManejadora="AnularSolicitudes"}
                {CWBarraSupPanel titulo="AnulaciÃ³n de solicitudes de permisos y licencias"}
                        {CWBotonTooltip imagen="detalle" titulo="Ver detalle" funcion="saltar" actuaSobre="ficha"
                                                                                        id="DetalleSolicitud"}
                        {CWBotonTooltip imagen="04" titulo="Refrescar" funcion="modificar" actuaSobre="ficha"
                                                                                action="AnularSolicitudes__buscar"}

                {/CWBarraSupPanel}
                {CWContenedor}
                        {if $smty_mostrarUsuario eq 1}
                                <p align="center" class="tabla_titulo">Usuario: {$smty_usuarioConectado}</p>
                        {/if}
                        <p align="center" class="tabla_titulo">Interesado: {$smty_usuarioActivo}</p>
                        {CWTabla conCheck="true" conCheckTodos="false"  id="Tabla1" numPagInsertar="0"
                                numFilasPantalla="15" datos=$smty_datosTabla seleccionUnica="true"}
                                {CWFila tipoListado="false"}
                                        {CWImagen nombre="img_avisos" textoAsociado=" " src="images/nada.gif"}
                                        {CWCampoTexto nombre="nsolicitud" editable="false" size="6"
                                                                                textoAsociado="NÂº sol."}
                                        {CWCampoTexto nombre="dclase" editable="false" size="40"
                                                                                textoAsociado="Motivo"}

                                        {CWCampoTexto nombre="fecha_ini" editable="false" size="10"
                                                                                textoAsociado="Fecha inicio"}
                                        {CWCampoTexto nombre="fecha_fin" editable="false" size="10"
                                                                                textoAsociado="Fecha fin"}
                                        {CWCampoTexto nombre="destado" editable="false" size="30"
                                                                                textoAsociado="Estado"}
                                        {CWCampoTexto nombre="situacion" editable="false" oculto=true}
                                        {CWCampoTexto nombre="cestado" editable="false" oculto=true}

                                        {CWCampoTexto nombre="fanulaper" editable="false" oculto=true}
                                        {CWCampoTexto nombre="ffirmaresolucion" editable="false" oculto=true}
                                        {CWCampoTexto nombre="ffirmaresolucionanula" editable="false" oculto=true}
                                {/CWFila}
                                {CWPaginador enlacesVisibles="3"}
                        {/CWTabla}
                {/CWContenedor}
                {CWBarraInfPanel}
                        <img src="images/soliAut.gif"/> 
                        {CWBoton imagen="35" id="Autorizar" texto="Anular" class="boton" accion="particular"
                                                        action="AnularSolicitudes__anularSolicitud"}
                        {CWBoton imagen="41" texto="Guardar" class="boton" accion="guardar"}
                        {CWBoton imagen="42" texto="Cancelar" class="boton" accion="cancelar"}
                {/CWBarraInfPanel}
        {/CWPanel}
<!-- ***************** PESTAÃ‘AS ************************-->
        {CWContenedorPestanyas}
                {CWPestanya tipo="fil" estado=$estado_fil}
                {CWPestanya tipo="lis" estado=$estado_lis}
        {/CWContenedorPestanyas}
{/CWMarcoPanel}
{/CWVentana}
```

Views – Anular solicitudes y permisos

```php
<?php


$comportamientoVentana= new IgepPantalla();

$panel = new IgepPanel('AnularSolicitudes',"smty_datosTabla");
$panel->activarModo("fil","estado_fil");
$panel->activarModo("lis","estado_lis");
$comportamientoVentana->agregarPanel($panel);


$tramitador= IgepSession::dameVariable('AnularSolicitudes','esPersonal');
$s->assign('smty_mostrarUsuario',$tramitador);

//Incorporamos los datos del usuario
$usuario = IgepSession::dameVariable('AnularSolicitudes','datosUsuario');
$s->assign('smty_usuario1',$usuario);


$usuarioConectado = IgepSession::dameVariable('AnularSolicitudes','informacionUsuarioConectado');
$s->assign('smty_usuarioConectado',$usuarioConectado);

$usuario = IgepSession::dameVariable('AnularSolicitudes','informacionUsuarioActivo');
$s->assign('smty_usuarioActivo',$usuario);

$nregpgv = IgepSession::dameVariable('AnularSolicitudes','nregpgv');
$s->assign('smty_nregpgv',$nregpgv);

$dni = IgepSession::dameVariable('AnularSolicitudes','dni');
$s->assign('smty_dni',$dni);


$s->assign('smty_esPersonal',IgepSession::dameVariable('AnularSolicitudes','esPersonal'));
$s->display('solicitudes/AnularSolicitudes.tpl');
?>
```

6.3. Baja de solicitudes y permisos

Clase Manejadora - Baja de solicitudes y permisos

```
<?php

                                        /*CLASE ESPECIFICA DE LA VENTANA QUE MANEJA BorraSolicitudes*/

class BorraSolicitudes extends gvHidraForm_DB{

    function BorraSolicitudes(){

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');

          $nombreTablas= array('tper_solicitudes','vper_ocupacion' );

          parent::__construct($g_dsn,$nombreTablas);

          //La select que mostramos en el panel de listado

           $this->setSelectForSearchQuery("SELECT tper_solicitudes.nregpgv, tper_estadosol.destado,CASE WHEN
tper_solicitudes.situacion='A' AND (tper_solicitudes.cestado<> 'AP' OR tper_solicitudes.fanulaper is null ) THEN
tper_cod_permisos.dclase ELSE 'ANULA '|| tper_cod_permisos.dclase END as \"dclase\",tper_solicitudes.situacion as
\"codigoSituacion\",tper_solicitudes.nsolicitud,tper_solicitudes.fecha_ini,tper_solicitudes.fecha_fin,tper_solicitudes.period
o,tper_solicitudes.cclase,tper_solicitudes.avisos,tper_solicitudes.valida,tper_solicitudes.cestado,tper_solicitudes.revisada
                                                        FROM tper_solicitudes,tper_cod_permisos, tper_estadosol");

          $this->setWhereForSearchQuery("tper_solicitudes.cclase = tper_cod_permisos.cclase and
                                         tper_solicitudes.cestado = tper_estadosol.cestado and
                                         tper_solicitudes.cestado in ('PR', 'PC')");

          $this->setOrderByForSearchQuery("tper_solicitudes.situacion asc, tper_estadosol.destado asc,
                                         tper_solicitudes.fecha_ini asc");

          /*Añadimos los Matching - Correspondecias campoTPL <-> campoBD*/

          $this->addMatching("nsolicitud","nsolicitud","tper_solicitudes");
          $this->addMatching("valida","valida","tper_solicitudes");


          //Definicion de los tipos de los campos

          $fecha_ini =  new gvHidraDate(false);
          $fecha_ini->setDayOfWeek('short');

          //$fecha_ini->setCalendar(true);
          $this->addFieldType('fecha_ini',$fecha_ini);

          //Definicion de los tipos de los campos
          $fecha_fin =  new gvHidraDate(false);
          $fecha_fin->setDayOfWeek('short');
          //$fecha_ini->setCalendar(true);
          $this->addFieldType('fecha_fin',$fecha_fin);



          //Indicamos la Clave Primaria
          $this->addCamposClave(array("nsolicitud"));

          /*Ventanas de Seleccion*/
          //Usuarios interesados
          $interesados = new IgepVentanaSeleccion('DNI','INTERESADOS',array("DNI","Nombre"));
          $interesados->setDependencia(array('depuertos'),array('vper_ocupacion.depuertos'));
          $this->addVentanaSeleccion($interesados);

          /*Acciones de Interfaz*/
          $this->addAccionInterfaz('depuertos','validaDNI');
          $this->addAccionInterfaz('DNI','validaDNI');

          //Checkbox
          $checkDepuertos = new IgepCheckBox('depuertos');
          $checkDepuertos->setValueChecked('S');
          $checkDepuertos->setValueUnchecked('N');
          $this->addCheckBox($checkDepuertos);

          }//Fin de Constructor


/************************* VALIDACIONES *********************************/

          //Validación DNI del interesado

          function validaDNI($objDatos){

          $campoDisparador = $objDatos->getTriggerField();
          $nif = $objDatos->getValue('DNI');

          if($nif!='' AND $campoDisparador=='DNI'){
            $depuertos = $objDatos->getValue('depuertos');
            $res = $this->consultar( "SELECT nregpgv, dapell1 ||' '|| dapell2 ||', '|| dnombre as \"Nombre\" FROM " .
                    "vper_ocupacion WHERE nregpgv like '".$nif."__' and vper_ocupacion.depuertos='".$depuertos."'");
          if(count($res)>0){
                $objDatos->setValue('Nombre',$res[0]['Nombre']);
                //$objDatos->setValue('nregpgv',$res[0]['nregpgv']);
           }
           else {
                $objDatos->setValue('Nombre','');
                //$objDatos->setValue('nregpgv','');
```

```
                $objDatos->setValue('DNI','');
                $this->showMensaje('APL-01',array($nif));
                return -1;
            }
        }
        else{
            $objDatos->setValue('Nombre','');
            //$objDatos->setValue('nregpgv','');
            $objDatos->setValue('DNI','');
        }
        return 0;
    } //Fin de validaDNI



        // incluir en la funcion prebuscar el check de puertos si es S o N

    function preBuscar($objDatos){

        $listaDePuertos = $this->getDefaultData('depuertos');
        $listaDePuertos['seleccionado']= $objDatos->getValue('depuertos');
        $this->addDefaultData('depuertos',$listaDePuertos);

        if($objDatos->getValue('DNI')!=''){
          $this->dni = $objDatos->getValue('DNI');
          //Pasamos DNI a nregpgv
                        $cons="SELECT nregpgv FROM vper_ocupacion where nregpgv like '".$this->dni."__'";
                        $res=$this->consultar($cons);
                        $this->nregpgv =$res['0']['nregpgv'];

          //Cambiamos información de usuario activo
          $datosUsuario = $this->consultar('SELECT nregpgv,dnombre, dapell1, dapell2 FROM vper_ocupacion WHERE
                        nregpgv=\''.$this->nregpgv.'\'');
          $this->datosUsuario = $datosUsuario[0]['dnombre'].' '.$datosUsuario[0]['dapell1'].'
                        '.$datosUsuario[0]['dapell2'];
          $this->informacionUsuarioActivo = substr($datosUsuario[0]['nregpgv'],0,-2).'-'.$datosUsuario[0]['dnombre'].'
                        '.$datosUsuario[0]['dapell1'].' '.$datosUsuario[0]['dapell2'];
        }

        $this->setParametrosBusqueda("TPER_SOLICITUDES.nregpgv='".$this->nregpgv."'");
        return 0;

        }


    function preIniciarVentana($objDatos) {

        // El usuario se cargará de la siguiente manera:
        // Seleccionaremos de la vista VPER_OCUPACION el DNI,nombre y apellidos de la   persona cuyo nregpgv coincida
        // con el "nrp" del usuario conectado en la sesiÃ³n (en TCOM_USUARIOS).
        $usuarioConectado = IgepSession::dameDatosUsuario();
        $this->nregpgv = $usuarioConectado['nrp'];
        $this->dni = substr($usuarioConectado['nrp'],0,-2);
        $datosUsuario = $this->consultar('SELECT nregpgv,dnombre, dapell1, dapell2 FROM vper_ocupacion WHERE
                                    nregpgv=\''.$usuarioConectado['nrp'].'\'');
        $this->datosUsuario = $datosUsuario[0]['dnombre'].' '.$datosUsuario[0]['dapell1'].' '.$datosUsuario[0]['dapell2'];
        $this->informacionUsuarioActivo = substr($datosUsuario[0]['nregpgv'],0,-2).'-'.$datosUsuario[0]['dnombre'].'
                                '.$datosUsuario[0]['dapell1'].' '.$datosUsuario[0]['dapell2'];

        //Segun el rol del usuario puede o no ver las solicitudes de otras personas
        if(IgepSession::dameRol()=='P_TRAMITA'){
          $this->esPersonal = 1;
          $this->informacionUsuarioConectado = $this->informacionUsuarioActivo;

        }
        else{

          $this->esPersonal = 0;

          $select = "SELECT tper_solicitudes.nregpgv,tper_estadosol.destado,CASE WHEN tper_solicitudes.situacion='A' AND
(tper_solicitudes.cestado<>'AP' OR tper_solicitudes.fanulaper is null) THEN  tper_cod_permisos.dclase ELSE 'ANULA ' ||
tper_cod_permisos.dclase END as \"dclase\",tper_solicitudes.situacion as
\"codigoSituacion\",tper_solicitudes.nsolicitud,tper_solicitudes.fecha_ini,tper_solicitudes.fecha_fin,tper_solicitudes.period
o,tper_solicitudes.cclase,tper_solicitudes.avisos,tper_solicitudes.cestado,tper_solicitudes.valida,tper_solicitudes.revisada

                    FROM tper_solicitudes,tper_cod_permisos, tper_estadosol
                    WHERE tper_solicitudes.cclase = tper_cod_permisos.cclase and
                           tper_solicitudes.cestado = tper_estadosol.cestado and
                           tper_solicitudes.cestado in ('PR', 'PC')and  TPER_SOLICITUDES.nregpgv = '".$this->nregpgv."'
                    ORDER BY tper_solicitudes.situacion asc, tper_estadosol.destado asc, tper_solicitudes.fecha_ini asc";

          $res = $this->consultar($select,array('DATATYPES'=>array('fecha_ini'=> TIPO_FECHA,'fecha_fin'=> TIPO_FECHA)));
          $this->str_whereFiltro = "  WHERE tper_solicitudes.cclase = tper_cod_permisos.cclase and
                                        tper_solicitudes.cestado = tper_estadosol.cestado and
                                        tper_solicitudes.cestado in ('PR', 'PC')and
                                TPER_SOLICITUDES.nregpgv = '".$this->nregpgv."'";
          if(count($res)>0){

                foreach($res as $index=> $tupla){

                        if($tupla['cestado']=='PC'){
                        if($tupla['avisos']!='')      $res[$index]['img_avisos'] = 'images/conAvisosY PC.gif';
                                else                  $res[$index]['img_avisos'] = 'images/PC.gif';
                        }
                        elseif($tupla['avisos']!='') $res[$index]['img_avisos'] = 'images/ok.gif';
            else
                $res[$index]['img_avisos'] = 'images/nada.gif';
                                $color = 'none';
                                if(trim($tupla['situacion'])=='ANULACIÃ“N')
                                        $color = 'aviso';
```

```
                }
                IgepComunicaUsuario::setRowColor($res[$index],$color);
                // $objDatos->setRowColor($res[$index],$color);
                $this->setResultadoBusqueda($res);

        }
        else{
                $mensaje = new IgepMensaje('APL-53');
                IgepSession::guardaMensaje('principal',$mensaje);
                return -1;
        }



                }
                return 0;
        }




function accionesParticulares($str_accion, $objDatos) {

        switch ($str_accion) {

        //mod para volver de detalle
        case 'volverDeDetalleSolicitud':
             $actionForward = $objDatos->getForward('correcto');
             break;

        // fin mod

        case 'borrarSolicitud':

             $objDatos->setOperacion('seleccionar');
             $m_datos = $objDatos->getAllTuplas();
             $errores = 0;
             $hayAnuladas = 0;
             $hayBorradas = 0;
             $pmensaje="";

             $solicitudesBorradas = array();
             $solicitudesAnuladas = array();
             $this->obj_conexion->empezarTransaccion();
             //04/10/2010 INICIO: Guardamos informaciÃ³n de la solicitud a borrar para guardarla posteriormente
             //nsolicitud,nregpgv,cclase,fechaIni,fechaFin ya estÃ¡n recogidos en $m_datos y posteriormente en
             //   $solicitud['nsolicitud']....

             //Para informar del php desde donde se realiza la llamada y el nombre del método
             $proceso = 'BorraSolicitudes.php ; f_borrarSolicitud';
             //$proceso = 'Fichero '.__FILE__.'; Método: '.__METHOD__.'; Línea: '.__LINE__;

             foreach($m_datos as $solicitud){

                  $res=$this>f_borrarSolicitud($solicitud['nsolicitud'],$solicitud['codigoSituacion'],$solicitud['nregpgv'],
                      $solicitud['periodo'],$solicitud['cclase'],$solicitud['fecha_ini'],$solicitud['fecha_fin'],
                      $solicitudesAnuladas,$solicitudesBorradas,$solicitud['valida'],
                      $solicitud['revisada'],$pmensaje,$solBorrada);

                  if($res<0  )
                           $errores++;
                  elseif($res==0  and $solBorrada ){

                  //02/11/2010 INICIO TAREA 6910 Auditar borrado de solicitudes. Sólo cuando se borre, se auditará.

                      SolicitudesBorradas::f_altaSolicitudesBorradas($solicitud['nsolicitud'],$solicitud['codigoSituacion'],
                            $solicitud['nregpgv'],$solicitud['cclase'],$solicitud['fecha_ini'],$solicitud['fecha_fin'],
                            $proceso,$this->obj_conexion);

                      if ($solicitud['cclase']=='VV6' or $solicitud['cclase']=='AAP'){

                      //Recalculamos el ordinal de los días solicitados de VV6 o AAP por un interesado en un determinado
                      //periodo, al haber anulado un VV6 o AAP

                            SolicitudesYPermisos::f_recalcularOrdinalBorrado($solicitud['nregpgv'],$solicitud['periodo'],
                                                                             $solicitud['cclase'] );

                      }
                  }

             }


             $this->obj_conexion->acabarTransaccion($errores);
             if($errores==0){
                  if ($pmensaje==""){
                  // INICIO añadimos $pmensaje
        //        $this->showMensaje('APL-24',array($pmensaje));
        //
                  $str_solicitudesBorradas = implode($solicitudesBorradas,', ');
                  $str_solicitudesAnuladas = implode($solicitudesAnuladas,', ');

                           $hayAnuladas=strlen($str_solicitudesAnuladas);
                           $hayBorradas=strlen($str_solicitudesBorradas);

                           //Mensaje en funcion de si hay anuladas o borradas o ambas a la vez
                           if (($hayAnuladas>0) and ($hayBorradas>0)){

                                    $this->showMensaje('APL-11',array($str_solicitudesBorradas,$str_solicitudesAnuladas));
                                    }
```

```
                        elseif ($hayAnuladas>0){
                                $this->showMensaje('APL-24',array($str_solicitudesAnuladas));
                                }
                        elseif ($hayBorradas>0){
                                $this->showMensaje('APL-25',array($str_solicitudesBorradas));
                                }
                                        }
                                $this->showMensaje('APL-80',array($solicitud['nsolicitud'],$pmensaje));
            }
            elseif ($res===-2)
                $this->showMensaje('APL-GENERICO',array($pmensaje));

            else
                $this->showMensaje('APL-12');
            $this->refreshSearch();
            $actionForward = $objDatos->getForward('correcto');
            return $actionForward;
            break;
        default:
            return -1;
    }
    return $actionForward;
}


function f_borrarSolicitud($pnsolicitud,$psituacion,$var_nregpgv,$var_periodo,$var_cclase,$var_fecha_ini,$var_fecha_fin,
      &$solicitudesAnuladas,&$solicitudesBorradas,$var_valida,$var_revisada,&$pmensaje,&$solBorrada){

      //Conexion
      $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');


      // Comprobar que antes de anular o borrar una solicitud de VAC, no se quedan los ADI consecutivos a ésta sin anular

      $mensaje = Solicitudes::f_anularBorrarVAC($var_nregpgv,$var_cclase,$var_fecha_ini,$var_fecha_fin,$var_periodo,'N');

      if($mensaje!='0'){
                $pmensaje=$mensaje;
                return -2;
      }

      // Auditar borrado de solicitudes
      $solBorrada = false;

      $nulo=null;

      if($psituacion=='X'){
                if (($var_valida=='S') and ($var_revisada=='S')) {
                        $res = $this->operar("UPDATE TPER_SOLICITUDES SET   situacion = 'A', cestado = 'PP' WHERE
                                                                                nsolicitud = ".$pnsolicitud);

                        array_push($solicitudesAnuladas,$pnsolicitud);
                }
                elseif (($var_valida=='N') and ($var_revisada=='S')){
                        $res = $this->operar("UPDATE TPER_SOLICITUDES SET   situacion = 'A', cestado = 'PN' WHERE
                                                                                nsolicitud = ".$pnsolicitud);

                        array_push($solicitudesAnuladas,$pnsolicitud);
                        }
                elseif ($var_revisada=='N'){

                        $res = $this->operar("UPDATE TPER_SOLICITUDES SET   situacion = 'A', cestado = 'PV' WHERE
                                                                                nsolicitud = ".$pnsolicitud);

                        array_push($solicitudesAnuladas,$pnsolicitud);
                        }
                elseif ($var_valida==$nulo) { // var_valida es nulo
                        $res = $this->operar("UPDATE TPER_SOLICITUDES SET   situacion = 'A', cestado = 'PT' WHERE
                                                                                nsolicitud = ".$pnsolicitud);

                        array_push($solicitudesAnuladas,$pnsolicitud);
                        }

      }
      elseif($psituacion=='B'){
                        $res = $this->operar("UPDATE TPER_SOLICITUDES SET   situacion = 'A', cestado = 'AP' WHERE
                                                                                nsolicitud = ".$pnsolicitud);

                        array_push($solicitudesAnuladas,$pnsolicitud);

      }

      else{

      // Si la solicitud de alta borrada tenía anulaciones asociadas, también se borran,
      // antes que la solicitud. Si no se hace antes falla el delete porque existencia de clave foranea

                $res = $this->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS WHERE nsolicitudmodifica = ".$pnsolicitud);

                $res = $this->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS WHERE nsolicitud = ".$pnsolicitud);



                //Borra la solicitud ya que es un alta
                $res = $this->operar("DELETE FROM TPER_SOLICITUDES WHERE nsolicitud = ".$pnsolicitud);
                if($res<0)return $res;

                //Auditar borrado de solicitudes
                $solBorrada = true;

                        array_push($solicitudesBorradas,$pnsolicitud);

      }

      //  Si se borra una solicitud de anulacion que está siendo modificada por otras
      // solicitudes de alta, se deberán borrar las solicitudes de alta que la modifican
      $pmensaje="";
```

```
        if(($psituacion=='X') or ($psituacion=='B')){
                SolicitudesModificadas::f_dameSolicitudesQueModifican($pnsolicitud,'N', $listaSolAsociadas);
                // Recorremos todas las solicitudes asociadas a la anulación que estamos borrando ($pnsolicitud) que se
                //deberan borrar automaticamente
                $textoSolAsociada="";
                foreach($listaSolAsociadas as $solicitudQueModifica ){
                        $res = $this->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS WHERE nsolicitud =
                                                        ".$solicitudQueModifica['nsolicitud']);
                        $res = $this->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS WHERE nsolicitudmodifica
                                                        = ".$solicitudQueModifica['nsolicitud']);
                        $res = $this->operar("DELETE FROM TPER_SOLICITUDES WHERE nsolicitud =
                                                        ".$solicitudQueModifica['nsolicitud']);

                        //Auditar borrado de solicitudes
                        $solBorrada = true;


                        $dd=$solicitudQueModifica['nsolicitud'];
                        $textoSolAsociada = ($textoSolAsociada.$dd.", ");

                }

                if ($textoSolAsociada!=''){

                $pmensaje= ("Al haber borrado la anulaciÃ³n ".$pnsolicitud.", tambien se han borrado las altas que la
                                                                                        modifican (".$textoSolAsociada.")");
                }
        }

        // Se comprueba si se han pedido ya todos los días de vacaciones despuÃ©s del borrado, solo para el alta.
        // Si no fuera así , se actualizarÃ¡n las solicitudes restantes al estado pendiente de completar.

        $var_estado = Solicitudes:: f_calculaEstadoPCoPR($var_cclase,$var_nregpgv,$var_periodo);
        if ($var_estado=='PC'){

                        // CAMBIO FECHA 18/07/2007 CCLASE IN ('VAC','ADI')
                $res = $this->operar("UPDATE TPER_SOLICITUDES SET  cestado='PC' WHERE ((nregpgv='$var_nregpgv') and
                (periodo= '$var_periodo') and (cclase IN ('VAC','ADI')) and (cestado= 'PR') and (situacion= 'A')) ");
        }

        return $res  ;


    }   // FIN FUNCION f_borrarSolicitud


    // funcion que muestra el detalle de una solicitud

    function saltoDeVentana($objDatos, $objSalto){

        $objDatos->setOperacion('seleccionar');
        $m_datos = $objDatos->getAllTuplas();
        if($m_datos==0){
            $this->showMensaje('APL-10');
            return -1;
        }
        $objSalto->setClase('DetalleSolicitud');
        $objSalto->setAccion('buscar');
        $objSalto->setAccionRetorno('volverDeDetalleSolicitud');
        $solicitudes = array();
        foreach($m_datos as $solicitud)
            array_push($solicitudes,$solicitud['nsolicitud']);
        $objSalto->setParam('solicitudes',$solicitudes);
        $objSalto->setParam('visibilidad','ConsultarSolicitudesACargoResponsable_NoPersonal');
        return 0;
    }


    function postBuscar($objDatos){

        //Añadimos el color a las filas
        $m_datos = $objDatos->getAllTuplas();

        if(count($m_datos)>0){
                                foreach($m_datos as $index=> $tupla){
                                        if($tupla['cestado']=='PC'){
                                                if($tupla['avisos']!='')
                                $m_datos[$index]['img_avisos'] = 'images/conAvisosY PC.gif';
                                else
                                $m_datos[$index]['img_avisos'] = 'images/PC.gif';
                                        }
                                        elseif($tupla['avisos']!='')
                $m_datos[$index]['img_avisos'] = 'images/ok.gif';
                else
                $m_datos[$index]['img_avisos'] = 'images/nada.gif';
                                        $color = 'none';
                                        if(trim($tupla['situacion'])=='ANULACIÓN')
                                                $color='aviso';
                $objDatos->setRowColor($m_datos[$index],$color);
          }
        }
        $objDatos->setAllTuplas($m_datos);

    return 0;
    }

}//Fin BorraSolicitudes

?>
```

Tpl – Borrar solicitudes y permisos

```
{CWVentana tipoAviso=$smty_tipoAviso  codAviso=$smty_codError  descBreve = $smty_descBreve  textoAviso=$smty_textoAviso
                                                                                    onLoad=$smty_jsOnLoad}
{CWBarra customTitle=$smty_customTitle usuario=$smty_usuario codigo=$smty_codigo}
        {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}

<!--********** PANEL fil ******************-->
        {CWPanel id="fil" action="buscar" method="post" estado="$estado_fil"  claseManejadora="BorraSolicitudes"}
                {CWBarraSupPanel titulo="Borrado de solicitudes de permisos y licencias"}
                        {CWBotonTooltip imagen="04" titulo="Limpiar campos" funcion="limpiar" actuaSobre="ficha"}
                {/CWBarraSupPanel}
                {CWContenedor}
                        {CWFicha}
                                {if $smty_esPersonal eq 1}

                                {CWCheckBox nombre="depuertos" dataType=$dataType_BorraSolicitudes.depuertos
                                        editable="true" textoAsociado="Usuarios de puertos
                                        (sustituciones)" actualizaA="x"}
                                <br><br>
                                {CWCampoTexto nombre="DNI"  maxlength="8" size="8" editable="true"
                                        textoAsociado="Interesado" actualizaA="x" value="$smty_dni"
                                        obligatorio="true"}
                                {CWBotonTooltip imagen="13" titulo="Busqueda Interesado" funcion="abrirVS"
                                        actuaSobre="DNI" formActua="fil" panelActua="fil"
                                        claseManejadora="BorraSolicitudes"}
                                 {CWCampoTexto nombre="Nombre" size="40" editable="false" value="$smty_usuario1"}
                                {*CWCampoTexto nombre="nregpgv" oculto="true" value="$smty_nregpgv"*}

                                {else}
                                 {CWCampoTexto nombre="Nombre" size="40" value="$smty_usuario1" editable="false"
                                        textoAsociado="Interesado"}
                                        {*CWCampoTexto nombre="nregpgv" oculto="true" value="$smty_nregpgv"*}



                                {/if}
                        {/CWFicha}
                {/CWContenedor}
                {CWBarraInfPanel}
                        {CWBoton imagen="50" texto="Buscar" class="boton" accion="buscar"}
                {/CWBarraInfPanel}
        {/CWPanel}

<!-- ***************** PANEL lis ***********************-->


        {CWPanel id="lis" action="editar" method="post" estado=$estado_lis claseManejadora="BorraSolicitudes"}
                {CWBarraSupPanel titulo="Borrado de solicitudes de permisos y licencias"}
                        {CWBotonTooltip imagen="detalle" titulo="Ver detalle" funcion="saltar" actuaSobre="ficha"
                                                                                id="DetalleSolicitud"}
                        {*CWBotonTooltip imagen="04" titulo="Refrescar" funcion="modificar" actuaSobre="ficha"
                                                                        action="BorraSolicitudes__buscar"*}
                {/CWBarraSupPanel}
                {CWContenedor}
                {if $smty_mostrarUsuario eq 1}
                        <p align="center" class="tabla_titulo">Usuario: {$smty_usuarioConectado}</p>
                {/if}
                <p align="center" class="tabla_titulo">Interesado: {$smty_usuarioActivo}</p>


                {CWTabla conCheck="true" conCheckTodos="true"  id="Tabla1" numPagInsertar="0"
                                                numFilasPantalla="12" datos=$smty_datosTabla}
                        {CWFila tipoListado="false"}
                                {CWImagen nombre="img_avisos" textoAsociado=" " src="images/nada.gif"}
                                {CWCampoTexto nombre="nsolicitud" size="6" editable="false"
                                                                        textoAsociado="NÂº sol."}
                                {CWCampoTexto nombre="dclase" size="49" editable="false"
                                                                        textoAsociado="Motivo"}
                                {CWCampoTexto nombre="fecha_ini" size="10" editable="false"
                                textoAsociado="Fecha inicio" dataType=$dataType_BorraSolicitudes.fecha_ini}
                                {CWCampoTexto nombre="fecha_fin" size="10" editable="false"
                                textoAsociado="Fecha fin" dataType=$dataType_BorraSolicitudes.fecha_fin}
                                {CWCampoTexto nombre="destado" size="30" editable="false"
                                                                        textoAsociado="Estado"}
                                {*CWCampoTexto nombre="situacion" size="10" editable="false"
                                                                        textoAsociado="SituaciÃ³n"*}
                                {CWCampoTexto nombre="codigoSituacion" oculto="true"}

                                {CWCampoTexto nombre="periodo" oculto="true"}
                                {CWCampoTexto nombre="cclase" oculto="true"}
                                {CWCampoTexto nombre="nregpgv" oculto=true }
                                {CWCampoTexto nombre="situacion" oculto="true"}
                                {CWCampoTexto nombre="valida" oculto="true"}
                                {CWCampoTexto nombre="revisada" oculto="true"}
                        {/CWFila}

                        {CWPaginador enlacesVisibles="3"}
                {/CWTabla}
                {/CWContenedor}
                {CWBarraInfPanel}
                        {* <img src="images/anulacion.gif"/> *}
                        {CWBoton imagen="62" id="Borrar" texto="Borrar" class="boton" accion="particular"
                                                        action="BorraSolicitudes__borrarSolicitud"}
                {/CWBarraInfPanel}
        {/CWPanel}


<!-- ***************** PANEL edi ***********************-->
<!-- ***************** NO UTILIZADO Y SUSTITUIDO POR UNA PANTALLA DE MENSAJE ***********************-->
```

```
        {CWPanel id="edi" tipoComprobacion="envio" action="$smty_operacionFichaBorraSolicitudes" method="post"
                 estado="$estado_edi" claseManejadora="BorraSolicitudes"  accion=$smty_operacionFichaBorraSolicitudes}
                 {CWBarraSupPanel titulo="Borrado de solicitudes de permisos y licencias"}
                          {CWBotonTooltip imagen="04" titulo="Limpiar campos" funcion="limpiar" actuaSobre="ficha"}

                 {/CWBarraSupPanel}
                 {CWContenedor}
                          {CWFichaEdicion id="FichaEdicion" datos=$smty_datosFicha numPagInsertar="1"}
                                   {CWFicha}
                                   {/CWFicha}
                                   {CWPaginador enlacesVisibles="3"}
                          {/CWFichaEdicion}
                 {/CWContenedor}
                 {CWBarraInfPanel}
                          {CWBoton imagen="41" texto="Guardar" class="boton" accion="guardar"}
                          {CWBoton imagen="42" texto="Cancelar" class="boton" accion="cancelar" action="cancelarEdicion"}
                 {/CWBarraInfPanel}
        {/CWPanel}

<!-- ***************** PESTAÃ‘AS ************************-->
        {CWContenedorPestanyas}
        <!--Es tramitador-->
                 {if $smty_esPersonal eq 1}
                          {CWPestanya tipo="fil" estado=$estado_fil}
                          {CWPestanya tipo="lis" estado=$estado_lis}
                          {CWPestanya tipo="edi" estado=$estado_edi}
        <!--No es tramitador-->
                 {else}
                          {CWPestanya tipo="lis" estado=$estado_lis}
                          {CWPestanya tipo="edi" estado=$estado_edi}
                 {/if}
        {/CWContenedorPestanyas}
{/CWMarcoPanel}
{/CWVentana}
```

Views – Borrar solicitudes y permisos

```php
<?php


$comportamientoVentana= new IgepPantalla();

$panel = new IgepPanel('BorraSolicitudes',"smty_datosTabla","smty_datosFicha");
$panel->activarModo("fil","estado_fil");
$panel->activarModo("lis","estado_lis");
$panel->activarModo("edi","estado_edi");
$comportamientoVentana->agregarPanel($panel);

//Para ocultar o mostrar campos nada mÃ¡s entrar a la pantalla
//$esTramitador = IgepSession::dameVariable('BorraSolicitudes','esTramitador');
//$s->assign("smty_tramitador",$esTramitador);

//Asignamos smty_nregpgv
//$s->assign("smty_nregpgv",IgepSession::dameVariable('BorraSolicitudes','nregpgv'));

//Incorporamos los datos del usuario
//$usuario = IgepSession::dameVariable('BorraSolicitudes','informacionUsuarioActivo');
//$s->assign('smty_usuarioActivo',$usuario);

// NACHO

$tramitador= IgepSession::dameVariable('BorraSolicitudes','esPersonal');
$s->assign('smty_mostrarUsuario',$tramitador);

//Incorporamos los datos del usuario
$usuario = IgepSession::dameVariable('BorraSolicitudes','datosUsuario');
$s->assign('smty_usuario1',$usuario);


$usuarioConectado = IgepSession::dameVariable('BorraSolicitudes','informacionUsuarioConectado');
$s->assign('smty_usuarioConectado',$usuarioConectado);

$usuario = IgepSession::dameVariable('BorraSolicitudes','informacionUsuarioActivo');
$s->assign('smty_usuarioActivo',$usuario);

$nregpgv = IgepSession::dameVariable('BorraSolicitudes','nregpgv');
$s->assign('smty_nregpgv',$nregpgv);

$dni = IgepSession::dameVariable('BorraSolicitudes','dni');
$s->assign('smty_dni',$dni);

//*************** mod *****************
$esPersonal=IgepSession::dameVariable('BorraSolicitudes','esPersonal');
$s->assign('smty_esPersonal',$esPersonal);



$s->display('solicitudes/BorraSolicitudes.tpl');
?>
```

6.4. Consulta: Control de presencia

Clase Manejadora– Consulta: Control de presencia

```php
<?php

class ControlPresencia extends gvHidraForm_DB{


	var $lanzarInforme;
	//En este atributo será el informe Jasper
	var $lisControlPresencia;

	//propiedad que nos indica si se muestra el filtro o no
	var $verFiltro = false;

	public function __construct() {

		$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
		$g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');

		//Las tablas sobre las que trabaja
		$nombreTablas= array('twps_vfichajes');
		parent::__construct($g_oracle_dsn,$nombreTablas);

		$str_selectEditar = "SELECT twps_vfichajes.personal, twps_vfichajes.fecha, ";
		$str_selectEditar .= "to_char(twps_vfichajes.hora,'HH24:MI') as \"HORA\" ";
		$str_selectEditar .= "FROM twps_vfichajes ";

		$this->setSelectForEditQuery($str_selectEditar);

		$this->addMatching("personal","personal","twps_vfichajes");
		$this->addMatching("fecha","fecha","twps_vfichajes");

		$interesados = new IgepVentanaSeleccion('DNI','INTERESADOS_HISTORICO',array("DNI","Nombre"),$g_dsn);
		$interesados->setDependencia(array('depuertos'),array('vper_ocupacion_historico.depuertos'));
		$this->addVentanaSeleccion($interesados);

		$this->addAccionInterfaz('DNI','validaDNI');
		$this->addAccionInterfaz('depuertos','validaDNI');

		$checDePuertos = new IgepCheckBox('depuertos');
		$checDePuertos->setValueChecked('S');
		$checDePuertos->setValueUnchecked('N');
		$this->addCheckBox($checDePuertos);

		$listaMeses = new IgepLista('listaMeses');
		$listaMeses->addOpcion('M',' Mes actual<br/>');
		$listaMeses->addOpcion('A',' Mes anterior');
		$listaMeses->setSelected('M');
		$listaMeses->setRadio(true);
		$this->addLista($listaMeses);

		//NO usamos fecha como objeto porque las operaciones ya estan hechas con llamadas primitivas.

}//Fin de constructor


public function preIniciarVentana($objDatos) {

	$perfil = IgepSession::dameRol();
	$opcionMenu = $objDatos->getValue('opcionMenu');


	$usuarioConectado = IgepSession::dameDatosUsuario();
	$esTitulaPersonal = Firmantes::f_esResponsablePersonalTitular($usuarioConectado['nrp']);

	if($perfil == 'P_RESPPER' AND $esTitulaPersonal  AND ($opcionMenu=='FPC'))
		$this->verFiltro = true;
	else
		$this->verFiltro = false;

	return 0;
}


public function preBuscar($objDatos) {


	if($this->verFiltro) {

		$this->accionListar = $objDatos->getValue('listaMeses');
		//Pasamos DNI a nregpgv
		$nregpgv = $objDatos->getValue('DNI');
		$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
			$conexion=new IgepConexion($g_dsn, true);
			$cons="SELECT nregpgv FROM vper_ocupacion_historico where nregpgv like '".$nregpgv."__'";
			$res=$conexion->consultar($cons);
			$this->nregpgv=$res['0']['nregpgv'];
		if($this->nregpgv=='') {
			$this->showMensaje('APL-48');
			return -1;
		}
	}
	else {
		$this->accionListar = $objDatos->getValue('accionListar');
```

```
                $datos = IgepSession::dameDatosUsuario();
                $this->nregpgv = $datos['nrp'];
        }
        return 0;
    }


public function postBuscar($objDatos) {

        $datos = $this->consultaFichajes($objDatos,'buscar');

        $objDatos->setAllTuplas($datos);
        return 0;
}//Fin de PostBuscar



public function consultaFichajes($objDatos, $operacion) {

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);

        $select = "SELECT dapell1 ||' '|| dapell2 ||', '|| dnombre as \"nombre\" , substr(nregpgv, 1,8) as \"dni\" " ;

        $select .= " FROM vper_ocupacion ";
        $select .= " WHERE nregpgv ='".$this->nregpgv."'";
        $res = $con->consultar($select);
        if($res==-1){
                return -1;
        }

        $m_datos = $res;
        // Array para guardar los datos para la operación Búsqueda
        $datos = array();
        // Array para guardar los datos que interesan para el listado
        $listado = array();
        // Variable en la que guardaremos el color del fondo correspondiente a cada fila
        $color = 'none';

        $this->dni = $m_datos[0]['dni'];
        $this->usuario = $m_datos[0]['nombre'];

        // Establecemos el intervalo de fechas a consultar
        // Comprobamos si venimos desde el menú inicial, x lo tanto tendremos la variable d qué mes consultar
        if ($this->accionListar != '')
                $mesConsulta = $this->accionListar;
        else
        // Si hemos dado al botón refrescar tendremos q utilizar la acción guardada d antes.
                $mesConsulta = $this->mesConsultado;
        switch ($mesConsulta) {
                case 'M': // Mes actual
                        $fechaInicio = date("01/".date('m/Y'));
                        //$ultimo = date("t",mktime(0, 0, 0, date('n'), 1, date('Y')));
                        $ultimo = date("d",mktime(0, 0, 0, date('n'), date('d'), date('Y')));
                        $ultimoDia = $ultimo."/";
                        $fechaFin = date($ultimoDia.date('m/Y'));
                break;
                case 'A': // Mes anterior
                        $mes = date('m');

                        if (strlen($mes) == 1)
                                $mes = '0'.$mes;
                        if ($mes == '01'){
                                $mesAnyo = '12/'.(date('Y') - 1);
                        }
                        else{
                                $mes = date('m') - 1;
                                if (strlen($mes) == 1)
                                        $mes = '0'.$mes;
                                $mesAnyo = $mes.'/'.(date('Y'));
                        }
                        $fechaInicio = date("01/".$mesAnyo);
                        $fechaFin = date("d/m/Y",mktime(0,0,0,date("m"),date("d","01".date("m/Y")) - 1, date("Y")));
                break;
        }

        // Nos guardamos q mes hemos consultado x si luego se pulsa el botón refrescar
        $this->mesConsultado = $mesConsulta;
        //Obtenemos el nº de tarjeta a partir del nº de registro personal

        $tarjeta = Personal::f_dameTarjeta($this->nregpgv);

        //Conexión a ORACLE
        $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
        $conexionOracle = new IgepConexion($g_oracle_dsn);
        // utilizo to_char para quitar la hora que viene con la fecha
        $select = "SELECT to_char(fecha,'dd/mm/yyyy') as \"fecha\", horario FROM twps_validacion";
        // fecha --> dd/mm/yyyy

        $select .= " WHERE personal = '".$tarjeta."' and horario <> 0 ";
        $select .= " and fecha BETWEEN to_date('".$fechaInicio."','dd-mm-yyyy') and to_date('".$fechaFin."','dd-mm-yyyy')";

        $select .= " ORDER BY fecha";
        $diasConsulta = $conexionOracle->consultar($select);
        // Guardamos en variables la fecha inicio y fin
        $this->fechaInicio = $fechaInicio;
        $this->fechaFin = $fechaFin;


        ////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
                    //   OBTENER LOS FICHAJES DEL MES A CONSULTAR

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
$sumaSaldo = 0;
$sumHorasTrabajadas = 0;
$sumHorasExceso = 0;
$phayContadores = 0;
$hayFichajes = 0;
$hayIncidencias = 0;
$fechaConsulta = "Fichajes desde ".$fechaInicio." hasta ".$fechaFin;
for($i=0;$i<count($diasConsulta);$i++)  {
          $datos[$i]['nombre'] = $m_datos[0]['nombre'];
          $datos[$i]['dni'] = $m_datos[0]['dni'];
          $datos[$i]['personal'] = $tarjeta;
          $datos[$i]['fecha'] = $diasConsulta[$i]['fecha']; // Guardamos la fecha sin hora
          $datos[$i]['hayDetalle'] = 0; // por defecto NO hay detalle

          ////////////// DATOS PARA EL LISTADO ////////////////////////////
          $listado[$i]['nombre'] = $m_datos[0]['nombre'];
          $listado[$i]['dni'] = $m_datos[0]['dni'];
          $listado[$i]['fechaConsulta'] = $fechaConsulta;
          $listado[$i]['fecha'] = $diasConsulta[$i]['fecha']; // Guardamos la fecha sin hora

          $listado[$i]['ent1'] = '-';
          $listado[$i]['sal1'] = '-';
          $listado[$i]['ent2'] = '-';
          $listado[$i]['sal2'] = '-';
          $listado[$i]['ent3'] = '-';
          $listado[$i]['sal3'] = '-';
          $listado[$i]['ent4'] = '-';
          $listado[$i]['sal4'] = '-';
          $listado[$i]['descripcion'] = ''; // quitamos el - por defecto para que no salga en el listado
          $listado[$i]['incidencia'] = '-';
          $listado[$i]['color'] = '-';


          // Obtenemos los fichajes
          // Tanto fichajes en el reloj normal como en el de incidencias (aux)
          $selectFichajes = "SELECT to_char(f.hora,'HH24:MI') as \"HORA\", f.causa, ";
          $selectFichajes .= "CASE WHEN f.causa = 0 THEN '' ELSE c.descripcion END DESCRIPCION, ";
          $selectFichajes .= "f.funcion, f.aux, c1.descripcion as \"DESCAUX\"  ";
          $selectFichajes .= "FROM twps_vfichajes f, twps_causas c, twps_causas c1 ";
          $selectFichajes .= "WHERE f.personal = '".$tarjeta."' and f.fecha =
                              to_date('".$diasConsulta[$i]['fecha']."','dd-mm-yyyy') and ";
          $selectFichajes .= "f.causa = c.codigo and f.aux = c1.codigo ";
          $selectFichajes .= "ORDER BY f.hora ";
          $fichajes = $conexionOracle->consultar($selectFichajes);
          $hayFichajes = 0;
          if (count($fichajes) > 0)
                    $hayFichajes = 1;

          // Obtener los contadores
          $datos[$i]['horasTrabajadas'] = '00:00';
          $datos[$i]['horasExceso'] = '00:00';
          $datos[$i]['saldoDiario'] = '00:00';

          $listado[$i]['horasTrabajadas'] = '00:00';
          $listado[$i]['horasExceso'] = '00:00';
          $listado[$i]['saldoDiario'] = '00:00';

          // HAY FICHAJE PARA ESE DÍA???
          $phayContadores = ValidacionHorario::f_existeContador($datos[$i]['fecha'], $horasTrabajadas, $horasExceso,
                              $saldoDiario, $sumaHorasTrabajadas, $sumaHorasExceso, $sumaSaldoDiario, $tarjeta);
          if ($phayContadores != 1)
                    $phayContadores = 0;
          $sumaSaldo = $sumaSaldoDiario + $sumaSaldo;


          //  PASO DE LOS MINUTOS A FORMATO HORA

          // Horas Trabajadas
          $sumHorasTrabajadas = $sumHorasTrabajadas + $horasTrabajadas;
          $horasTrabajadas = $this->formatoHoras($horasTrabajadas);
          // Horas Exceso
          $sumHorasExceso = $sumHorasExceso + $horasExceso;
          $horasExceso = $this->formatoHoras($horasExceso);
          // Saldo Diario
          $saldoDiario = $this->formatoHoras($saldoDiario);

          $datos[$i]['horasTrabajadas'] = $horasTrabajadas;
          $datos[$i]['horasExceso'] = $horasExceso;
          $datos[$i]['saldoDiario'] = $saldoDiario;

          $listado[$i]['horasTrabajadas'] = $horasTrabajadas;
          $listado[$i]['horasExceso'] = $horasExceso;
          $listado[$i]['saldoDiario'] = $saldoDiario;



          // OBTENER INCIDENCIAS QUE NO SON DE DÃ A COMPLETO
          $select = "SELECT CASE
                         WHEN i.incidencia IN (0,1,2,3,4,5,6) THEN ' '
                         WHEN i.incidencia IN (11,12,13,15,16,20) THEN ic.descripcion
                                                                  ELSE 'Desconocida ('||i.incidencia||')'
                                                                  END incidencia, ";
          $select .= "i.causa, ";
          $select .= "CASE WHEN i.causa = 0 THEN '' ELSE c.descripcion END desc_causa ";
          $select .= "FROM twps_vincidencias i, twps_incidencias ic, twps_causas c ";
          $select .= "WHERE i.personal = '".$tarjeta."' and i.incidencia NOT IN (5,16,17,18,21,22,24) and ";
          $select .= "fecha = to_date('".$datos[$i]['fecha']."','dd-mm-yyyy') and i.incidencia = ic.codigo and
                                                          i.causa = c.codigo ";
          $incidencias = $conexionOracle->consultar($select);
```

```
		if (count($incidencias) > 0)
			$hayIncidencias = 1;
		else
			$hayIncidencias = 0;

////////////////////////////////////////////////////////////////////////////////////////////////////////////
		$color='none'; // Por defecto una fila no tiene color
		$listado[$i]['descripcion'] = '';
		// Añadimos $hayFichajes == 0 para que salgan las fichadas del día ya que la única condición que cambia es
		//esa. De esta forma entra en el else  // DÃ A NORMAL DE TRABAJO
		 if ( ($phayContadores == 0) && ($hayIncidencias == 0) && ($hayFichajes == 0))  {

////////////////////////////////////////////////////////////////////////////////////////////////////////////
					// DÍA FESTIVO NO TRABAJADO
		 }
		 elseif ( ($hayIncidencias == 1) && ($phayContadores == 1) && ($hayFichajes == 0) ) {

////////////////////////////////////////////////////////////////////////////////////////////////////////////
					// DÍA ENTERO DE PERMISO
			for($j=0;$j<count($incidencias);$j++){ // Para cada fichaje

				$datos[$i]['incidencia'] .= $incidencias[$j]['INCIDENCIA']."
								".$incidencias[$j]['DESC_CAUSA']."  ";
				if ($incidencias[$j]['CAUSA'] == 1){
					// INCIDENCIA NO JUSTIFICADA SE MUESTRA EN ROJO
					$listado[$i]['incidencia'] = $datos[$i]['incidencia'];
					$color='error';
					$datos[$i]['hayDetalle'] = 1; // Marcamos q sÃ hay detalle
					$listado[$i]['color'] = 'rojo';
				}
				else{
					$listado[$i]['incidencia'] = $datos[$i]['incidencia'];
					$listado[$i]['color'] = 'blanco';
				}
			}

		}

////////////////////////////////////////////////////////////////////////////////////////////////////////////
		else  {		// DÃ A NORMAL DE TRABAJO

			$indiceFichadas = 0;

			for($j=0;$j<count($fichajes);$j++) {	// Para cada fichaje

				$txtIncidencia = "Salida por motivo de ";
				$indiceFichadas++;

				switch ($indiceFichadas) {
					////////////////////////////////////////////////
					// fichadas de ENTRADA
					case 1:
						$datos[$i]['ent1'] = $fichajes[$j]['HORA'];
						$listado[$i]['ent1'] = $fichajes[$j]['HORA'];
						break;

					case 3:
						$datos[$i]['ent2'] = $fichajes[$j]['HORA'];
						$listado[$i]['ent2'] = $fichajes[$j]['HORA'];
						break;
					case 5:
						$datos[$i]['ent3'] = $fichajes[$j]['HORA'];
						$listado[$i]['ent3'] = $fichajes[$j]['HORA'];
						break;
					case 7:
						$datos[$i]['ent4'] = $fichajes[$j]['HORA'];
						$listado[$i]['ent4'] = $fichajes[$j]['HORA'];
						break;

					////////////////////////////////////////////////
					// fichadas de SALIDA

					case 2: // Salida 1
						$datos[$i]['sal1'] = $fichajes[$j]['HORA'];
						$listado[$i]['sal1'] = $fichajes[$j]['HORA'];
						if ( ($fichajes[$j]['CAUSA'] <> 0) && (($fichajes[$j]['FUNCION']
						== 4) ||  ($fichajes[$j]['FUNCION'] == 2) ||
						($fichajes[$j]['FUNCION'] == 0)) ){

							// POR INCIDENCIA
							$datos[$i]['sal1'] = $datos[$i]['sal1']."*";
							$listado[$i]['sal1'] = $datos[$i]['sal1'];
							$datos[$i]['hayDetalle'] = 1; // Marcamos q sí hay
											detalle
							$listado[$i]['hayDetalle'] = 1; // Marcamos q sí hay
											detalle
							$listado[$i]['color'] = 'blanco';
							if ($fichajes[$j]['AUX'] != 0)
								$listado[$i]['descripcion'] .=
								"*".$txtIncidencia.$fichajes[$j]['DESCAUX']."\n
								";
							else
								$listado[$i]['descripcion'] .=
								"*".$txtIncidencia.$fichajes[$j]['DESCRIPCION']
								."\n";
						}
						else
							if ($fichajes[$j]['AUX'] <> 0)  {
								// POR TIEMPO
								$datos[$i]['sal1'] = $datos[$i]['sal1']."*";
								$listado[$i]['sal1'] = $datos[$i]['sal1'];
```

```
                                        $datos[$i]['hayDetalle'] = 1; // Marcamos q sí
                                                                        hay detalle
                                        $listado[$i]['hayDetalle'] = 1; // Marcamos q
                                                                        sí hay detalle
                                        $listado[$i]['color'] = 'blanco';

                                        $listado[$i]['descripcion'] .=
                                        "*".$txtIncidencia.$fichajes[$j]['DESCAUX']."\n
                                        ";
                                }
                        break;
                case 4:
                        $datos[$i]['sal2'] = $fichajes[$j]['HORA'];
                        $listado[$i]['sal2'] = $fichajes[$j]['HORA'];
                        if ( ($fichajes[$j]['CAUSA'] <> 0) && (($fichajes[$j]['FUNCION']
                        == 4) ||  ($fichajes[$j]['FUNCION'] == 2) ||
                        ($fichajes[$j]['FUNCION'] == 0) ) ){
                                // Por incidencia
                                $datos[$i]['sal2'] = $datos[$i]['sal2']."**";
                                $listado[$i]['sal2'] = $datos[$i]['sal2'];
                                $datos[$i]['hayDetalle'] = 1; // Marcamos q sí hay
                                                                        detalle
                                $listado[$i]['hayDetalle'] = 1; // Marcamos q sí hay
                                                                        detalle
                                $listado[$i]['color'] = 'blanco';
                                if ($fichajes[$j]['AUX'] != 0)
                                        $listado[$i]['descripcion'] .=
                                        "**".$txtIncidencia.$fichajes[$j]['DESCAUX']."\
                                        n";
                                else
                                        $listado[$i]['descripcion'] .=
                                        "**".$txtIncidencia.$fichajes[$j]['DESCRIPCION'
                                        ]."\n";
                        }
                        else
                                if ($fichajes[$j]['AUX'] <> 0) {
                                        //Por tiempo
                                        $datos[$i]['sal2'] = $datos[$i]['sal2']."**";
                                        $listado[$i]['sal2'] = $datos[$i]['sal2'];
                                        $datos[$i]['hayDetalle'] = 1; // Marcamos q sí
                                                                        hay detalle
                                        $listado[$i]['hayDetalle'] = 1; // Marcamos q
                                                                        sí hay detalle
                                        $listado[$i]['color'] = 'blanco';
                                        $listado[$i]['descripcion'] .=
                                        "**".$txtIncidencia.$fichajes[$j]['DESCAUX']."\
                                        n";
                                }
                        break;
                case 6:
                        $datos[$i]['sal3'] = $fichajes[$j]['HORA'];
                        if ( ($fichajes[$j]['CAUSA'] <> 0) && (($fichajes[$j]['FUNCION']
                        == 4) ||  ($fichajes[$j]['FUNCION'] == 2) ||
                        ($fichajes[$j]['FUNCION'] == 0) ) ){
                                // Por incidencia
                                $datos[$i]['sal3'] = $datos[$i]['sal3']."*";
                                $listado[$i]['sal3'] = $datos[$i]['sal3'];
                                $datos[$i]['hayDetalle'] = 1; // Marcamos q sí hay
                                                                        detalle
                                $listado[$i]['hayDetalle'] = 1; // Marcamos q sí hay
                                                                        detalle
                                $listado[$i]['color'] = 'blanco';
                                if ($fichajes[$j]['AUX'] != 0)
                                        $listado[$i]['descripcion'] .=
                                        "*".$txtIncidencia.$fichajes[$j]['DESCAUX']."\n
                                        ";
                                else
                                        $listado[$i]['descripcion'] .=
                                        "*".$txtIncidencia.$fichajes[$j]['DESCRIPCION']
                                        ."\n";
                        }
                        else
                                if ($fichajes[$j]['AUX'] <> 0){
                                        //Por tiempo
                                        $datos[$i]['sal3'] = $datos[$i]['sal3']."*";
                                        $listado[$i]['sal3'] = $datos[$i]['sal3'];
                                        $datos[$i]['hayDetalle'] = 1; // Marcamos q sí
                                                                        hay detalle

                                        $listado[$i]['hayDetalle'] = 1; // Marcamos q
                                                                        sí hay detalle
                                        $listado[$i]['color'] = 'blanco';
                                        $listado[$i]['descripcion'] .=
                                        "*".$txtIncidencia.$fichajes[$j]['DESCAUX']."\n
                                        ";
                                }
                        break;
                case 8:
                        $datos[$i]['sal4'] = $fichajes[$j]['HORA'];
                        if ( ($fichajes[$j]['CAUSA'] <> 0) && (($fichajes[$j]['FUNCION']
                        == 4) ||  ($fichajes[$j]['FUNCION'] == 2) ||
                        ($fichajes[$j]['FUNCION'] == 0) ) ){
                                // Por incidencia
                                $datos[$i]['sal4'] = $datos[$i]['sal4']."**";
                                $listado[$i]['sal4'] = $datos[$i]['sal4'];
                                $datos[$i]['hayDetalle'] = 1; // Marcamos q sí hay
                                                                        detalle
                                $listado[$i]['hayDetalle'] = 1; // Marcamos q sí hay
                                                                        detalle
                                $listado[$i]['color'] = 'blanco';
                                if ($fichajes[$j]['AUX'] != 0)
```

```
                                                                        $listado[$i]['descripcion'] .=
                                                                        "**".$txtIncidencia.$fichajes[$j]['DESCAUX']."\
                                                                        n";
                                                                else
                                                                        $listado[$i]['descripcion'] .=
                                                                        "**".$txtIncidencia.$fichajes[$j]['DESCRIPCION'
                                                                        ]."\n";
                                                        }
                                                        else
                                                                if ($fichajes[$j]['AUX'] <> 0){
                                                                        //Por tiempo
                                                                        $datos[$i]['sal4'] = $datos[$i]['sal4']."**";
                                                                        $listado[$i]['sal4'] = $datos[$i]['sal4'];
                                                                        $datos[$i]['hayDetalle'] = 1; //Marcamos q
                                                                                                      síhay detalle

                                                                        $listado[$i]['hayDetalle'] = 1; //Marcamos q sí
                                                                                                        hay detalle
                                                                        $listado[$i]['color'] = 'blanco';
                                                                        $listado[$i]['descripcion'] .=
                                                                        "**".$txtIncidencia.$fichajes[$j]['DESCAUX']."\
                                                                        n";
                                                                }
                                                        break;
                                                } // switch fichadas
                                        }// for
                                        for($j=0;$j<count($incidencias);$j++) {    // Para cada incidencia

                                                if ($incidencias[$j]['CAUSA'] == 1){
                                                        $listado[$i]['descripcion'] .= " ".$incidencias[$j]['INCIDENCIA']."
                                                                                        ".$incidencias[$j]['DESC_CAUSA'];

                                                        // INCIDENCIA NO JUSTIFICADA SE MUESTRA EN ROJO
                                                        $color='error';
                                                        $datos[$i]['hayDetalle'] = 1; // Marcamos q sÃ hay detalle
                                                        $listado[$i]['color'] = 'rojo';
                                                }
                                                else{
                                                        $listado[$i]['descripcion'] .= $incidencias[$j]['INCIDENCIA']."
                                                                                        ".$incidencias[$j]['DESC_CAUSA']." (Primer
                                                                                        fichaje)";
                                                        $datos[$i]['hayDetalle'] = 1;
                                                        $listado[$i]['color'] = 'blanco';
                                                }

                                        }
                                } // else
                                //Fijamos color a la fila
                                $objDatos->setRowColor($datos[$i],$color);
                        } // for $diasConsulta

                        $sumaSaldo = $this->formatoHoras($sumaSaldo);
                        $this->saldo = $sumaSaldo;
                        $sumHorasTrabajadas = $this->formatoHoras($sumHorasTrabajadas);
                        $this->horasTrabajadas = $sumHorasTrabajadas;
                        $sumHorasExceso = $this->formatoHoras($sumHorasExceso);
                        $this->horasExceso = $sumHorasExceso;

                        for($i=0;$i<count($diasConsulta);$i++)  {
                                $listado[$i]['totalSaldo'] = $this->saldo;
                                $listado[$i]['totalHorasTrabajadas'] = $this->horasTrabajadas;
                                $listado[$i]['totalHorasExceso'] = $this->horasExceso;
                        }

                        if ($operacion == 'buscar'){

                                return $datos;
                        }
                        elseif ($operacion == 'listar'){

                                return $listado;
                        }
}// consultaFichajes


////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////
// TRATAMIENTO DE LOS MINUTOS PARA FORMATEARLOS COMO HH:MM
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////
private function formatoHoras($horas) {

        $signo = ' ';
        if ($horas < 0){
                $horas = $horas*(-1);
                $signo = '-';
        }
        $minutos = ($horas % 60);
        if (strlen($horas % 60) < 2)
                $minutos = "0".($horas % 60);
        $horas = intval($horas/60);
        if (strlen($horas) == 1)
                $horas = '0'.$horas;
        $horas = $signo.$horas.":".$minutos;
        return($horas);
}
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////
//          CONSULTA  DETALLE

                                                                                                                              //
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////
function preEditar($objDatos) {

          $m_datos = $objDatos->getAllTuplas();
          for($i=0;$i<count($m_datos);$i++){
                    if ($m_datos[$i]['hayDetalle'] != 1){
                              $fecha = $m_datos[$i]['fecha'];
                              $this->showMensaje('APL-42',array($fecha));
                              return -1;
                    }
          }
          return 0;
}



function postEditar($objDatos) {

          $m_datos = $objDatos->getAllTuplas();
          $fecha = $m_datos[0]['FECHA'];
          $fecha = split(' ',$fecha);
          $fecha = $fecha[0];
          // Guardamos en variables la fecha inicio y fin
          $this->dia = $fecha;
          $tarjeta = $m_datos[0]['PERSONAL'];

          //Conexión a ORACLE
          $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
          $conexionOracle = new IgepConexion($g_oracle_dsn);
          $m_datos_nuevos = array();

          $selectFichajes = "SELECT to_char(f.hora,'HH24:MI') as \"HORA\", f.causa, ";
          $selectFichajes .= "CASE WHEN f.causa = 0 THEN '' ELSE c.descripcion END DESCRIPCION, ";
          $selectFichajes .= "f.funcion, f.aux, c1.descripcion as \"DESCAUX\"  ";
          $selectFichajes .= "FROM twps_vfichajes f, twps_causas c, twps_causas c1 ";
          $selectFichajes .= "WHERE f.personal = '".$tarjeta."' and f.fecha = to_date('".$fecha."','dd-mm-yyyy') ";
          $selectFichajes .= " and f.causa = c.codigo and f.aux = c1.codigo ";
          $fichajes = $conexionOracle->consultar($selectFichajes);
          $hayFichajes = 0;
          if (count($fichajes) > 0)
                    $hayFichajes = 1;

          // Obtener las incidencias que no son de día completo
          $selectIncidencias = "SELECT CASE  WHEN i.incidencia IN (0,1,2,3,4,5,6) THEN ' '
                                             WHEN i.incidencia IN (11,12,13,15,16,20) THEN ic.descripcion
                                                      ELSE 'Desconocida ('||i.incidencia||')'
                                                      END incidencia, ";
          $selectIncidencias .= "i.causa, ";
                    $selectIncidencias .= "CASE WHEN i.causa = 0 THEN '' ELSE c.descripcion END desc_causa ";
          $selectIncidencias .= "FROM twps_vincidencias i, twps_incidencias ic, twps_causas c ";
          $selectIncidencias .= "WHERE i.personal = '".$tarjeta."' and i.incidencia NOT IN (5,16,17,18,21,22,24) and ";
          $selectIncidencias .= "fecha = to_date('".$fecha."','dd-mm-yyyy') and i.incidencia = ic.codigo and i.causa =
                                                                                                                  c.codigo ";
          $incidencias = $conexionOracle->consultar($selectIncidencias);
          $hayIncidencias = 0;
          if (count($incidencias) > 0)
                    $hayIncidencias = 1;

          $phayContadores = ValidacionHorario::f_existeContador($fecha, $horasTrabajadas, $horasExceso, $saldoDiario,
                                        $sumaHorasTrabajadas, $sumaHorasExceso, $sumaSaldoDiario, $tarjeta);
          if ($phayContadores != 1)
                    $phayContadores = 0;

          // Añadimos $hayFichajes == 0 para que salgan las fichadas del día ya que la única condición que cambia es esa. De
          // esta forma entra en el else  // DÃ A NORMAL DE TRABAJO
          if ( ($phayContadores == 0) && ($hayIncidencias == 0) && ($hayFichajes == 0))  {
                    // Día festivo no trabajado
          }
          elseif ( ($phayContadores == 1) && ($hayIncidencias == 1)  && ($hayFichajes == 0) ){
                    for($j=0;$j<count($incidencias);$j++)  {   // Para cada incidencia

                              }
                    }
                    else {
                              // Día normal de trabajo
                              $horasTrabajadas = $this->formatoHoras($horasTrabajadas);
                              $horasExceso = $this->formatoHoras($horasExceso);
                              $saldoDiario = $this->formatoHoras($saldoDiario);

                              $m_datos_nuevos[0]['horastrabajadas'] = $horasTrabajadas;
                              $m_datos_nuevos[0]['horasexceso'] = $horasExceso;
                              $m_datos_nuevos[0]['saldodiario'] = $saldoDiario;

                              //$incidencia = '';
                              $fichada = 0;

                              for($j=0;$j<count($incidencias);$j++) {  // Para cada incidencia

                                        if ($incidencias[$j]['CAUSA'] == 1){
                                                  $m_datos_nuevos[0]['incidencia'] .= "  ".$incidencias[$j]['INCIDENCIA']."
                                                                                      ".$incidencias[$j]['DESC_CAUSA'];
                                        }

                                        else{
                                                  $m_datos_nuevos[0]['incidencia'] .= $incidencias[$j]['INCIDENCIA']."
                                                                      ".$incidencias[$j]['DESC_CAUSA']." (Primer fichaje)";
```

```
				}
			}

			for($i=0;$i<count($fichajes);$i++){
				$x = $i + 1;
				$numero = $x % 2;
				if ($numero != 0)
				{
					$fichada++;
					$m_datos_nuevos[0]['ent'.$fichada] = $fichajes[$i]['HORA'];
				}
				else
				{
					$m_datos_nuevos[0]['sal'.$fichada] = $fichajes[$i]['HORA'];
					$signo = '';
					switch($x)
					{
						case 2:
							$signo = '*';
						break;
						case 4:
							$signo = '**';
						break;
						case 6:
							$signo = '+';
						break;
						case 8:
							$signo = '++';
						break;
					} //switch
					$txtIncidencia = " ".$signo."Salida por motivo de ";
					if ( ($fichajes[$i]['CAUSA'] <> 0)
						&& (($fichajes[$i]['FUNCION'] == 4) || ($fichajes[$i]['FUNCION']
						== 2) || ($fichajes[$i]['FUNCION'] == 0) ) )
					{
						// Por incidencia
						$m_datos_nuevos[0]['sal'.$fichada] =
								$m_datos_nuevos[0]['sal'.$fichada].$signo;
						$m_datos_nuevos[0]['incidencia'] .= "\n
								".$txtIncidencia.$fichajes[$i]['DESCRIPCION'];
					} //if
					elseif ($fichajes[$i]['AUX'] != 0)
					{
						// Por tiempo
						$m_datos_nuevos[0]['sal'.$fichada] =
								$m_datos_nuevos[0]['sal'.$fichada].$signo;
						$m_datos_nuevos[0]['incidencia'] .= "\n
								".$txtIncidencia.$fichajes[$i]['DESCAUX'];
					}
				}
			}
		}
		$objDatos->setAllTuplas($m_datos_nuevos);
		return 0;
}


///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
///////////////////////////////////////////////////////////
//                                                                                         LISTADO  CONSULTA  DE  FICHAJES

                                                                                //
///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
///////////////////////////////////////////////////////////
public function accionesParticulares($str_accion, $objDatos) {

	switch($str_accion){
		case "listadoFichajes":
			$actionForward = $this->listadoFichajes($objDatos);
			break;
	}
	$actionForward = $objDatos->getForward('correcto');
	return $actionForward;
}


public function listadoFichajes($objDatos) {

	$objDatos->setOperacion('external');

	$listado = $this->consultaFichajes($objDatos,'listar');
	if (count($listado) > 0){

		//Creamos el objeto que manejará el informe
		$informeJ = new InformeJasper('ControlPresencia');
		//Especificamos la fuente de datos, en este caso, una BD relacional
		$informeJ->setDataSourceType('xml');
		//Fijamos la disposición del informe incrustado en la ventana
		$informeJ->setDisposition('inline');
		//Enlazamos la matriz
		$informeJ->addDBResultSet($listado, array('idResultSet'=>'listado'));
		//Fijamos el fichero jasper que nos hace de plantilla
		$informeJ->setJasperFile('./plantillasJasper/consultasFichajes.jasper');

		// Asignamos el informe como variable de clase para poder
		//acceder a él desde el fichero views donde realizaremos "la ejecución"
		$this->lisControlPresencia = $informeJ;
		$this->lanzarInforme = true;
		$actionForward = $objDatos->getForward('mostrarListado');
```

```
                $this->openWindow($actionForward);
        }
        else{
                unset($this->lisControlPresencia);//Borramos el objeto...
                $this->showMensaje('APL-Listado');//Mostramos un mensaje
        }
}


//Validación DNI del interesado
public function validaDNI($objDatos) {

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);
        $campoDisparador = $objDatos->getTriggerField();
        $nif = $objDatos->getValue('DNI');
        if($nif!='' AND $campoDisparador=='DNI'){
                $depuertos = $objDatos->getValue('depuertos');
                $res = $con->consultar( "SELECT nregpgv, dapell1 ||' '|| dapell2 ||', '|| dnombre as \"Nombre\" FROM " .
                                        "vper_ocupacion_historico WHERE nregpgv like '".$nif."__' and
                                        vper_ocupacion_historico.depuertos='".$depuertos."'");
                if(count($res)>0){
                        $objDatos->setValue('Nombre',$res[0]['Nombre']);
                        $objDatos->setValue('nregpgv',$res[0]['nregpgv']);
                 }
                else {
                        $objDatos->setValue('Nombre','');
                         $objDatos->setValue('nregpgv','');
                         $objDatos->setValue('DNI','');
                        $this->showMensaje('APL-01',array($nif));
                        return -1;
                }
        }
        else{
            $objDatos->setValue('Nombre','');
            $objDatos->setValue('DNI','');
        }
        return 0;
} //Fin de validaDNI

}//Fin de la clase ControlPresencia
?>
```

Tpl– Consulta: Control de presencia

```
{CWVentana tipoAviso=$smty_tipoAviso  codAviso=$smty_codError  descBreve = $smty_descBreve  textoAviso=$smty_textoAviso
                                                                                    onLoad=$smty_jsOnLoad}
{CWBarra customTitle=$smty_customTitle usuario=$smty_usuario codigo=$smty_codigo}
        {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}


{if $smty_verFiltro eq 1}
        <!--************ PANEL fil ******************-->
        {CWPanel id="fil" action="buscar" method="post" estado="$estado_fil" claseManejadora="ControlPresencia"}
                {CWBarraSupPanel titulo="Fichadas del personal de la CIT"}
                        {CWBotonTooltip imagen="04" titulo="Restaurar valores" funcion="restaurar" actuaSobre="ficha"}
                {/CWBarraSupPanel}
                {CWContenedor}
                        {CWFicha}
                        <br/>
                        <table  cellpadding="5" width="90%">
                        <tr>
                        <td class="formularios" width="3%">
                                 
                        </td>
                        <td class="formularios"  colspan="2" width="97%">
                        {CWCheckBox nombre="depuertos" editable="true" textoAsociado="Usuarios de puertos
                                (sustituciones)" actualizaA="x" dataType=$dataType_ControlPresencia.depuertos}
                        <br/><br/>
                        {CWCampoTexto nombre="DNI"  maxlength="8" size="8" editable="true" textoAsociado="Personal"
                                                                                                actualizaA="x"}
                        {CWBotonTooltip imagen="13" titulo="Busqueda Interesado" funcion="abrirVS" actuaSobre="DNI"
                                        formActua="fil" panelActua="fil" claseManejadora="ControlPresencia"}
                         {CWCampoTexto nombre="Nombre" size="50" editable="false"}

                        </td>
                        </tr>
                        <tr>
                        <td colspan="2">
                                 
                        </td>
                        </tr>
                        <tr>
                        <td class="formularios">
                                 
                        </td>
                        <td class="formularios" width="20%">
                        <fieldset style="  border-bottom-width: thin; border-color: silver">
                        <legend align='left' style="font-weight:bold;">Fichadas</legend>
                        {CWLista nombre="listaMeses" editable="true" datos=$defaultData_ControlPresencia.listaMeses
                                                        dataType=$dataType_ControlPresencia.listaMeses}
                        </fieldset>
                        </td>
                        <td class="formularios" width="77%">
                                 
                        </td>
                        </tr>
                        </table>
                        <br/>
                        {/CWFicha}
                {/CWContenedor}
                {CWBarraInfPanel}
                        {CWBoton imagen="50" texto="Buscar" class="boton" accion="buscar" busquedaEsLarga="true"}
                {/CWBarraInfPanel}
        {/CWPanel}

{/if}

<!-- ***************** PANEL lis ***********************-->
        {*CWPanel id="lis" action="listadoFichajes" method="post" estado="$estado_lis" claseManejadora="ControlPresencia"*}
        {CWPanel id="lis" method="post" estado="$estado_lis" claseManejadora="ControlPresencia"}
                {CWBarraSupPanel titulo="Fichajes desde $smty_inicio hasta el $smty_fin"}
                        {CWBotonTooltip imagen="02" titulo="Ver detalle" actuaSobre="ficha"  action="editar"}
                        {CWBotonTooltip imagen="04" titulo="Refrescar" funcion="modificar" actuaSobre="tabla"
                                                                                                action="buscar"}
                {/CWBarraSupPanel}
                {CWContenedor}
                        <table align="center" width=50% cellspacing=1 cellpadding=1 border=0>
                                <tr>
                                        <td colspan="3" class="tabla_titulo">
                                                Usuario: {$smty_dni} - {$smty_usuarioActivo}
                                        </td>
                                </tr>
                                <tr>
                                        <td class="datos_cabecera" align="left">
                                                Horas Trabajadas: {$smty_horasTrabajadas}
                                        </td>
                                        <td class="datos_cabecera" align="left">
                                                Horas Exceso: {$smty_horasExceso}
                                        </td>
                                        <td class="datos_cabecera" align="left">
                                                Saldo: {$smty_saldo}
                                        </td>
                                </tr>
                        </table>
                        {CWTabla conCheck="true" seleccionUnica="true" id="Tabla1" numPagInsertar="0"
                                                                numFilasPantalla="23" datos=$smty_datosTabla}
                                {CWFila tipoListado="false"}
                                        {CWCampoTexto nombre="fecha" editable="false" size="12"
                                                                                textoAsociado="Fecha"}
                                        {CWCampoTexto nombre="ent1" editable="false" size="8" textoAsociado="Ent1"}
                                        {CWCampoTexto nombre="sal1" editable="false" size="7" textoAsociado="Sal1"}
                                        {CWCampoTexto nombre="ent2" editable="false" size="7" textoAsociado="Ent2"}
                                        {CWCampoTexto nombre="sal2" editable="false" size="7" textoAsociado="Sal2"}
```

```
                                                {CWCampoTexto nombre="ent3" editable="false" size="7" textoAsociado="Ent3"}
                                                {CWCampoTexto nombre="sal3" editable="false" size="7" textoAsociado="Sal3"}
                                                {CWCampoTexto nombre="ent4" editable="false" size="7" textoAsociado="Ent4"}
                                                {CWCampoTexto nombre="sal4" editable="false" size="7" textoAsociado="Sal4"}
                                                {CWCampoTexto nombre="horasTrabajadas" editable="false" size="5"
                                                                                        textoAsociado="H. trabajadas"}
                                                {CWCampoTexto nombre="horasExceso" editable="false" size="5"
                                                                                        textoAsociado="H. exceso"}
                                                {CWCampoTexto nombre="saldoDiario" editable="false" size="5"
                                                                                        textoAsociado="Saldo diario"}
                                                {CWCampoTexto nombre="incidencia" editable="false" size="50"
                                                                                        textoAsociado="Incidencia"}
                                                {CWCampoTexto nombre="personal" oculto="true"}

                                                {CWCampoTexto nombre="hayDetalle" oculto="true"}
                                                {CWCampoTexto nombre="nregpgv" oculto=true}
                                        {/CWFila}
                                        {CWPaginador enlacesVisibles="3"}
                                {/CWTabla}
                        {/CWContenedor}
                        {CWBarraInfPanel}
                                        <img src="images/incidencia_sinjust.gif"/>
                                                 
                                        {CWBoton imagen="06" texto="Ver detalle mes" class="boton" accion="particular"
                                                                                        action="listadoFichajes" }
                                                     
                                                     
                                                     
                                                     
                                                     

                        {/CWBarraInfPanel}
                {/CWPanel}


<!-- ****************** PANEL edi ***********************-->
        {CWPanel id="edi" tipoComprobacion="envio" action="operarBD" method="post" estado="$estado_edi"
                                        claseManejadora="ControlPresencia"  accion=$smty_operacionFichaControlPresencia}
                {CWBarraSupPanel titulo="Detalle del fichaje del $smty_dia"}
                {/CWBarraSupPanel}
                {CWContenedor}
                        {CWFichaEdicion id="FichaEdicion" datos=$smty_datosFicha numPagInsertar="5"}
                                {CWFicha}
                                <table width=98% cellspacing=0 cellpadding=0 border=0>
                                  <tr class="formularios">
                                        <td colspan="4" class="tabla_titulo">
                                                Usuario: {$smty_dni} - {$smty_usuarioActivo}
                                        </td>
                                        </tr>
                                  <tr>
                                        <td colspan="4">  </td>
                                        </tr>
                                  <tr class="formularios">
                                        <td>
                                        {CWCampoTexto nombre="ent1" editable="false" size="5" textoAsociado="Ent1"}
                                          
                                        {CWCampoTexto nombre="sal1" editable="false" size="5"textoAsociado="Sal1"}
                                        </td>
                                        <td>
                                        {CWCampoTexto nombre="ent2" editable="false" size="5" textoAsociado="Ent2"}
                                          
                                        {CWCampoTexto nombre="sal2" editable="false" size="5" textoAsociado="Sal2"}
                                        </td>
                                        <td>
                                        {CWCampoTexto nombre="ent3" editable="false" size="5" textoAsociado="Ent3"}
                                          
                                        {CWCampoTexto nombre="sal3" editable="false" size="5" textoAsociado="Sal3"}
                                        </td>
                                        <td>
                                        {CWCampoTexto nombre="ent4" editable="false" size="5" textoAsociado="Ent4"}
                                          
                                        {CWCampoTexto nombre="sal4" editable="false" size="5" textoAsociado="Sal4"}
                                        </td>
                                </tr>
                                <tr>
                                        <td colspan="4">  </td>
                                </tr>
                                <tr class="formularios">
                                        <td>
                                        {CWCampoTexto nombre="horastrabajadas" editable="false" size="5"
                                                                                        textoAsociado="Horas trabajadas"}
                                        </td>
                                        <td>
                                        {CWCampoTexto nombre="horasexceso" editable="false" size="5"
                                                                                        textoAsociado="Horas exceso"}
                                        </td>
                                        <td colspan="2">
                                        {CWCampoTexto nombre="saldodiario" editable="false" size="5"
                                                                                        textoAsociado="Saldo diario"}
                                        </td>
                                </tr>
                                <tr>
                                        <td>  </td>
                                </tr>
                                <tr>
                                        <td colspan="4" class="formularios">
                                         {CWAreaTexto nombre="incidencia" editable="false" cols="80" rows="2"}
                                        </td>
                                </tr>
                                <tr>
                                        <td colspan="4">
                                                  
```

```
                                        </td>
                                    </tr>
                                    <tr class="formularios">
                                        <td>
                                            {CWCampoTexto nombre="personal" oculto="true"}
                                            {CWCampoTexto nombre="fecha" oculto="true"}
                                        </td>
                                    </tr>
                                    </table>
                                    {/CWFicha}
                                    {CWPaginador enlacesVisibles="3"}
                                {/CWFichaEdicion}
                        {/CWContenedor}
                        {CWBarraInfPanel}
                        {/CWBarraInfPanel}
                {/CWPanel}

<!-- ***************** PESTA?S ************************-->
                {CWContenedorPestanyas}
{if $smty_verFiltro eq 1}
                        {CWPestanya tipo="fil" estado=$estado_fil}
{/if}
                        {CWPestanya tipo="lis" estado=$estado_lis}
                        {CWPestanya tipo="edi" estado=$estado_edi}
                {/CWContenedorPestanyas}
{/CWMarcoPanel}
{/CWVentana}
```

Views– Consulta: Control de presencia

```php
<?php
//Creamos una pantalla
$comportamientoVentana= new IgepPantalla();

//Creamos un panel
$panel1 = new IgepPanel('ControlPresencia',"smty_datosTabla","smty_datosFicha");
//Activamos las pestanyas que necesitamos
$panel1->activarModo("fil","estado_fil");
$panel1->activarModo("lis","estado_lis");
$panel1->activarModo("edi","estado_edi");

// Datos que aparecerÃ¡n en la cabecera
$dni = IgepSession::dameVariable('ControlPresencia','dni');
$s->assign('smty_dni',$dni);
$usuario = IgepSession::dameVariable('ControlPresencia','usuario');
$s->assign('smty_usuarioActivo',$usuario);
$fInicio = IgepSession::dameVariable('ControlPresencia','fechaInicio');
$s->assign('smty_inicio',$fInicio);
$fFin = IgepSession::dameVariable('ControlPresencia','fechaFin');
$s->assign('smty_fin',$fFin);
$dia = IgepSession::dameVariable('ControlPresencia','dia');
$s->assign('smty_dia',$dia);
$saldo = IgepSession::dameVariable('ControlPresencia','saldo');
$s->assign('smty_saldo',$saldo);
$hTrabajadas = IgepSession::dameVariable('ControlPresencia','horasTrabajadas');
$s->assign('smty_horasTrabajadas',$hTrabajadas);
$hExceso = IgepSession::dameVariable('ControlPresencia','horasExceso');
$s->assign('smty_horasExceso',$hExceso);

//Para saber si visualizamos el filtro
$verFiltro = IgepSession::dameVariable('ControlPresencia','verFiltro');
if($verFiltro)
          $s->assign('smty_verFiltro',1);
else
          $s->assign('smty_verFiltro',0);


//Agregamos el panel a la ventana
$comportamientoVentana->agregarPanel($panel1);
//Realizamos el display
$s->display('consultas/p_controlPresencia.tpl');
?>
```

6.5. Consulta: Días de licencia por asuntos propios y vacaciones

Clase Manejadora – Consulta: Días de licencia por asuntos propios y vacaciones

```php
<?php

/*CLASE ESPECIFICA DEL PANEL Q MANEJA ConsultaDiasLicenciaAsuntosPropiosyVacaciones*/

class ConsultaDiasLicenciaAsuntosPropiosyVacaciones extends gvHidraForm_DB{


function ConsultaDiasLicenciaAsuntosPropiosyVacaciones(){
          /*manejador de conexión*/
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');

          //Las tablas sobre las que trabaja
          $nombreTablas= array('vper_permisos_anyo','tper_personas');
          parent::__construct($g_dsn,$nombreTablas);

          // con el "nrp" del usuario conectado en la sesión (en TCOM_USUARIOS).
          $usuarioConectado = IgepSession::dameDatosUsuario();
          $datosUsuario = $this->consultar('SELECT nregpgv,dnombre, dapell1, dapell2 FROM vper_ocupacion
                                                  WHERE nregpgv=\''.$usuarioConectado['nrp'].'\'');
          $this->datosUsuario = $datosUsuario[0];
          $this->informacionUsuarioActivo =substr($datosUsuario[0]['nregpgv'], 0,8).'-'.$datosUsuario[0]['dnombre'].'
                                                  '.$datosUsuario[0]['dapell1'].' '.$datosUsuario[0]['dapell2'];

          $nregpgv = $this->datosUsuario['nregpgv'];


}//Fin constructor de la clase ConsultaDiasLicenciaAsuntosPropiosyVacaciones




function preBuscar($objDatos) {
          $this->accion = $objDatos->getValue('accion');
          return 0;
}


//Select para obtener el valor de M_DIASADI actualmente 2007
function postBuscar($objDatos) {
          $accion = $this->accion;
          $nregpgv = $this->datosUsuario['nregpgv'];
          $totalVAC=0;$totalVV6=0;$totalAAP=0;$totaldiasVAC=0;$totaldiasVV6=0;$totaldiasADI=0;$totaldiasAAP=0;
          $anyodiasADI=0;$totalADI=0;
          $res = $this-> f_consultarVV6yAAPyVACyADI($accion,$nregpgv,$totalVAC,$totalVV6,$totalADI,$totalAAP,$totaldiasVAC,
                                                  $totaldiasVV6,$totaldiasADI,$totaldiasAAP,$anyodiasADI,$mensaje);

          //Fijamos el resultado de la consulta
          $objDatos->setAllTuplas($res);



          //  CABECERA DE LA TPL EN FUNCION DE LA ACCION
          if ($accion=='P'){
                    $titulo_tpl="Consulta de dias de licencia por asuntos propios y vacaciones autorizados del periodo
                                                                                                                  actual";
          }
          else{
                    $titulo_tpl="Consulta de dias de licencia por asuntos propios y vacaciones autorizados del periodo
                                                                                                                  anterior";
          }
          // Paso a variables de clase para smarty
          $this->totalVV6=$totalVV6;
          $this->totalAAP=$totalAAP;
          $this->totalVAC=$totalVAC;
          $this->totalADI=$totalADI;

          $this->totaldiasVV6=$totaldiasVV6;
          $this->totaldiasAAP=$totaldiasAAP;
          $this->totaldiasVAC=$totaldiasVAC;
          $this->totaldiasADI=$totaldiasADI;

          // Para evitar que salga un -1 en pendientes cuando el total pedido es 23, ya que el máximo son 22, restaremos 22 en
          lugar de 23
          if (isset($totaldiasVAC) AND isset($totalVAC))
                    if ($totalVAC == 23) $this->pendVAC=$totaldiasVAC-22;
                    else $this->pendVAC=$totaldiasVAC-$totalVAC;
          elseif (isset($totaldiasVAC) AND !isset($totalVAC)) $this->pendVAC=$totaldiasVAC;

          if ($totaldiasADI === '?' AND $totalADI === '?') {
                    $this->totaldiasADI= 'Pdte. F. PÃºblica';
                    $this->totalADI= '0';
                    $this->pendADI='';
          }
          elseif (isset($totaldiasADI) AND isset($totalADI)) $this->pendADI=$totaldiasADI-$totalADI;

          if (isset($totaldiasVV6) AND isset($totalVV6)) $this->pendVV6=$totaldiasVV6-$totalVV6;
          elseif (isset($totaldiasVV6) AND !isset($totalVV6)) $this->pendVV6=$totaldiasVV6;

          if (isset($totaldiasAAP) AND isset($totalAAP)) $this->pendAAP=$totaldiasAAP-$totalAAP;
          elseif (isset($totaldiasAAP) AND !isset($totalAAP)) $this->pendAAP=$totaldiasAAP;
```

```
		$this->mensaje=$mensaje;
		$this->titulo_tpl=$titulo_tpl;

		return 0;
}



function f_consultarVV6yAAPyVACyADI($accion,$nregpgv,&$totalVAC,&$totalVV6,&$totalADI,&$totalAAP,&$totaldiasVAC,
				&$totaldiasVV6,&$totaldiasADI,&$totaldiasAAP,&$anyodiasADI,&$mensaje){
		$g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
		$conexion=new IgepConexion($g_oracle_dsn);
		$nulo=null;


		// componer el periodo para la consulta según el parametro accion (P:periodo actual; A: periodo anterior)


		if ($accion=='P'){
			// Estamos en el periodo actual

			//FFF Modificacion para usar objetos fecha en vez de cadenas.
			$var_periodo_objeto=new gvHidraTimestamp(date('Y'));
			$var_periodo=$var_periodo_objeto->format('Y');



			PermisosValidacion::f_calcularTotalDiasVACyADIyVV6yAAP($nregpgv,$var_periodo,$totaldiasVAC,
					$totaldiasADI,$anyodiasADI,$totaldiasVV6,$totaldiasAAP,$accion);



			$res = $this->consultar("SELECT CASE WHEN vper_permisos_anyo.cclase='VV6' THEN 'ASUNTOS PROPIOS'
					WHEN vper_permisos_anyo.cclase='VAC' THEN 'VACACIONES'
					WHEN vper_permisos_anyo.cclase='ADI' THEN 'DIAS ADICIONALES VACACIONES'
					WHEN vper_permisos_anyo.cclase='AAP' THEN 'DIAS ADICIONALES ASUNTOS PARTICULARES'
					ELSE vper_permisos_anyo.cclase END as \"var_motivo\", vper_permisos_anyo.fecha_ini,
					     vper_permisos_anyo.fecha_fin, vper_permisos_anyo.dias
					FROM vper_permisos_anyo WHERE vper_permisos_anyo.nregpgv ='$nregpgv' AND
					     vper_permisos_anyo.periodo = '$var_periodo'
					ORDER BY periodo desc,CASE WHEN vper_permisos_anyo.cclase='VV6' THEN 'A'
					                           WHEN vper_permisos_anyo.cclase='VAC' THEN 'AAA'
					                             WHEN vper_permisos_anyo.cclase='ADI' THEN 'AAAA'
					                              WHEN vper_permisos_anyo.cclase='AAP' THEN 'AA'
					                                   END,fecha_ini asc");
			// SI NO EXISTEN REGISTROS MENSAJE
			if (count($res[0])==0) {
				// Mostramos en la cabecera los totales obtenidos para el periodo actual
				//$totaldiasVAC,$totaldiasVV6,$totaldiasAAP,$anyodiasADI,$totaldiasADI
				// Tambien mostramos la siguiente información: "No ha tenido ningún dia de permiso o de licencia
				por asuntos propios o vacaciones en el periodo indicado"
				//$this->showMensaje("No ha tenido ningún día de permiso o de licencia por asuntos propios o
				vacaciones en ",$var_periodo);
				$mensaje = "No ha tenido ningún día de permiso o de licencia por asuntos propios o vacaciones en
					el periodo ".$var_periodo;
			}
			else {

				// CALCULAMOS LOS TOTALES DE VV6 , AAP, VAC Y ADI del periodo correspondiente

				$res2 = $this->consultar("SELECT case when sum(dias) is null then 0 else sum(dias) end as
							\"totalVAC\"
						FROM vper_permisos_anyo WHERE nregpgv = '$nregpgv' and cclase = 'VAC'
									and periodo = '$var_periodo'");
				if($res2!=-1)
					$totalVAC = $res2[0]['totalVAC'];
				$res2 = $this->consultar("SELECT case when sum(dias) is null then 0 else sum(dias) end as
							\"totalVV6\"
						FROM vper_permisos_anyo WHERE nregpgv = '$nregpgv' and cclase = 'VV6'
									and periodo = '$var_periodo'");
				if($res2!=-1)
					$totalVV6 = $res2[0]['totalVV6'];
				$res2 = $this->consultar("SELECT case when sum(dias) is null then 0 else sum(dias) end as
							\"totalADI\"
						FROM vper_permisos_anyo WHERE nregpgv = '$nregpgv' and cclase = 'ADI'
									and periodo = '$var_periodo'");
				if($res2!=-1)
					$totalADI = $res2[0]['totalADI'];

				$res2 = $this->consultar("SELECT case when sum(dias) is null then 0 else sum(dias) end as
							\"totalAAP\"
					 FROM vper_permisos_anyo WHERE nregpgv = '$nregpgv' and cclase = 'AAP'
									and periodo = '$var_periodo'");
				if($res2!=-1)
					$totalAAP = $res2[0]['totalAAP'];


			}

			// Devuelvo mas valores
			return $res;

		}

}
?>
```

Tpl – Consulta: Días de licencia por asuntos propios y vacaciones

```
{CWVentana tipoAviso=$smty_tipoAviso  codAviso=$smty_codError  descBreve=$smty_descBreve  textoAviso=$smty_textoAviso
                                                                                    onLoad=$smty_jsOnLoad}

{CWBarra customTitle=$smty_customTitle usuario=$smty_usuario codigo=$smty_codigo}
        {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}

        <!--*********** PANEL fil ******************-->
        {CWPanel id="fil" action="buscar" method="post" estado="$estado_fil"
                                        claseManejadora="ConsultaDiasLicenciaAsuntosPropiosyVacaciones"}
                {CWBarraSupPanel titulo="Consulta de días de licencia por asuntos propios y vacaciones autorizados"}
                        {CWBotonTooltip imagen="01" titulo="Insertar registros" funcion="insertar" actuaSobre="ficha"
                                        action="ConsultaDiasLicenciaAsuntosPropiosyVacaciones__nuevo"}
                        {CWBotonTooltip imagen="04" titulo="Restaurar valores" funcion="restaurar" actuaSobre="ficha"}
                {/CWBarraSupPanel}
                {CWContenedor}
                        {CWFicha}
                                {CWCampoTexto nombre="var_motivo" size="45" editable="false" textoAsociado="Motivo"}

                                {CWCampoTexto nombre="fechaini" size="10" editable="false" textoAsociado="Fecha
                                                                                                inicio"}
                                {CWCampoTexto nombre="fechafin" size="10" editable="false" textoAsociado="Fecha fin"}
                                {CWCampoTexto nombre="dias" size="3" editable="false" textoAsociado="Días"}

                        {/CWFicha}
                {/CWContenedor}
                {CWBarraInfPanel}
                        {CWBoton imagen="50" texto="Buscar" class="boton" accion="buscar" }
                {/CWBarraInfPanel}
        {/CWPanel}

        <!-- ****************** PANEL lis ***********************-->
        {CWPanel id="lis" tipoComprobacion="envio" action="operarBD" method="post" estado="$estado_lis"
                                        claseManejadora="ConsultaDiasLicenciaAsuntosPropiosyVacaciones"}
                {CWBarraSupPanel titulo=$smty_titulo_tpl}
                {/CWBarraSupPanel}
                {CWContenedor}
                <p align="center" class="tabla_titulo">Usuario: {$smty_usuarioActivo}</p>

                <table cellpadding="1" class="tablaInsertar" width="98%">
                        <!--tr>
                                <td width="14%" rowspan = "2" class="formularios tabla_titulo"> </td>
                                <td width="42%" colspan = "3" class="formularios tabla_titulo"> {$smty_anyo}</td>
                                <td width="42%" colspan = "3" class="formularios
                                                                tabla_titulo"> {$smty_anyoADI}</td>
                        </tr-->
                        <tr>
                                <td width="14%" class="formularios tabla_titulo"> </td>
                                <td width="14%" class="formularios tabla_titulo">Disponibles</td>
                                <td width="14%" class="formularios tabla_titulo">Autorizados</td>
                                <td width="14%" class="formularios tabla_titulo">Pendientes</td>
                                <td width="14%" class="formularios tabla_titulo">Adicionales disponibles</td>
                                <td width="14%" class="formularios tabla_titulo">Adicionales autorizados</td>
                                <td width="14%" class="formularios tabla_titulo">Adicionales pendientes</td>
                        </tr>
                        <tr>
                                <td width="14%" class="formularios tabla_titulo">Vacaciones</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totaldiasVAC}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totalVAC}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_pendVAC}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totaldiasADI}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totalADI}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_pendADI}</td>
                        </tr>
                        <tr>
                                <td width="14%" class="formularios tabla_titulo">Asuntos propios</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totaldiasVV6}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totalVV6}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_pendVV6}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totaldiasAAP}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_totalAAP}</td>
                                <td width="14%" align="center" class="formularios tablaNoEdi">
                                 {$smty_pendAAP}</td>
                        </tr>
                </table>

                {if $smty_mensaje <> ''}
                <table cellpadding="1" class="tablaInsertar" width="98%">
                        <tr>
                                <td width="14%" colspan = "7" align="center" class="formularios tablaNoEdi">
                                 {$smty_mensaje}</td>
                        </tr>
                </table>
                {/if}
```

```
                    {CWTabla conCheck="false" conCheckTodos="false" id="Tabla1" numPagInsertar="1"
                                                                    numFilasPantalla="15" datos=$smty_datosTabla}
                              {CWFila tipoListado="false"}
                                        {CWCampoTexto nombre="var_motivo" size="45" editable="false"
                                                                                textoAsociado="Motivo"}
                                        {CWCampoTexto nombre="fecha_ini" size="10" editable="false"
                                                                              textoAsociado="Fecha inicio"}
                                        {CWCampoTexto nombre="fecha_fin" size="10" editable="false"
                                                                                textoAsociado="Fecha fin"}
                                        {CWCampoTexto nombre="dias" size="3" editable="false" textoAsociado="Días"}

                              {/CWFila}
                              {CWPaginador enlacesVisibles="3"}
                    {/CWTabla}
          {/CWContenedor}
          {CWBarraInfPanel}
                    {CWBoton imagen="41" texto="Guardar" class="boton" accion="guardar"}
                    {CWBoton imagen="42" texto="Cancelar" class="boton" accion="cancelar"}
          {/CWBarraInfPanel}
{/CWPanel}

<!-- ***************** PESTAÃ‘AS ************************-->
{CWContenedorPestanyas}
          {*CWPestanya tipo="fil" estado=$estado_fil*}
          {CWPestanya tipo="lis" estado=$estado_lis}
{/CWContenedorPestanyas}
{/CWMarcoPanel}
{/CWVentana}
```

Views – Consulta: Días de licencia por asuntos propios y vacaciones

```php
<?php


//Creamos una pantalla
$comportamientoVentana= new IgepPantalla();

//Creamos un panel
$panel1 = new IgepPanel('ConsultaDiasLicenciaAsuntosPropiosyVacaciones',"smty_datosTabla");
//Incorporamos los datos del usuario
$usuario = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','informacionUsuarioActivo');
$s->assign('smty_usuarioActivo',$usuario);
//Incorporamos el tipo de accion a realizar aÃ±o actual o anterior
$titulo_tpl=IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','titulo_tpl');
$s->assign('smty_titulo_tpl',$titulo_tpl);
//Incorporamos el mensaje a mostrar en caso de no haber registros
$mensaje=IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','mensaje');
$s->assign('smty_mensaje',$mensaje);



$l5 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totaldiasVV6');
$s->assign('smty_totaldiasVV6',$l5);
$l6 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totaldiasAAP');
$s->assign('smty_totaldiasAAP',$l6);
$l7 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totaldiasVAC');
$s->assign('smty_totaldiasVAC',$l7);
$l8 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totaldiasADI');
$s->assign('smty_totaldiasADI',$l8);

$l1 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totalVV6');
$s->assign('smty_totalVV6',$l1);
$l2 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totalAAP');
$s->assign('smty_totalAAP',$l2);
$l3 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totalVAC');
$s->assign('smty_totalVAC',$l3);
$l4 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','totalADI');
$s->assign('smty_totalADI',$l4);

$l11 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','pendVAC');
$s->assign('smty_pendVAC',$l11);
$l12 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','pendADI');
$s->assign('smty_pendADI',$l12);
$l13 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','pendVV6');
$s->assign('smty_pendVV6',$l13);
$l14 = IgepSession::dameVariable('ConsultaDiasLicenciaAsuntosPropiosyVacaciones','pendAAP');
$s->assign('smty_pendAAP',$l14);

//Activamos las pestañas que necesitamos
$panel1->activarModo("fil","estado_fil");
$panel1->activarModo("lis","estado_lis");
//Agregamos el panel a la ventana
$comportamientoVentana->agregarPanel($panel1);
//Realizamos el display
$s->display('consultas/ConsultaDiasLicenciaAsuntosPropiosyVacaciones.tpl');
?>
```

6.6. Consulta: Estado de mis solicitudes

Clase Manejadora– Consulta: Estado de mis solicitudes

```
<?php

class ConsultaEstadoMisSolicitudes extends gvHidraForm_DB{

function ConsultaEstadoMisSolicitudes(){

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');

        $nombreTablas= array('s');
        parent::__construct($g_dsn,$nombreTablas);

        $this->setSelectForSearchQuery("SELECT s.nsolicitud, s.fecha_ini,s.fecha_fin,c.dclase, " .
                                       "CASE " .
                                       " WHEN s.situacion ='A' and (s.cestado<>'AP' or s.fanulaper is null) THEN c.dclase
                                                ELSE 'ANULA ' || c.dclase  END AS \"dclase\" ,".  /
                                       "CASE " .
                                       " WHEN s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado='AP' and
                                                                                s.fanulaper is null THEN 'Autorizada' ".
                                       " WHEN s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado='NP' THEN 'No
                                                                                                    autorizada' ".
                                       " WHEN s.ffirmaresolucionanula is not null and s.situacion ='B' and s.cestado='AP'
                                                                                                    THEN 'Autorizada' ".
                                       " WHEN s.ffirmaresolucionanula is not null and s.situacion ='A' and s.cestado='AP' and
                                                                    s.fanulaper is not null      THEN 'No autorizada' ".
                                                                " ELSE e.destado " .
                                       " END as \"destado\", situacion, s.cestado, fanulaper,ffirmaresolucion,
                                                ffirmaresolucionanula,avisos,s.cclase, s.requierejustificante " .

                                       " FROM TPER_SOLICITUDES s,tper_cod_permisos c, TPER_ESTADOSOL e ");
        $this->setWhereForSearchQuery("s.cestado = e.cestado and s.cclase = c.cclase ");
        $this->setOrderByForSearchQuery("s.fecha_ini desc,s.fecha_fin desc");


        $this->addMatching('fil_periodo','periodo','s');

        $listaTipoSolicitud = new IgepLista('fil_solicitudes');
        $listaTipoSolicitud->setRadio(true);
        $listaTipoSolicitud->addOpcion('pendientes',' Pendientes<br/>');
        $listaTipoSolicitud->addOpcion('firmadas',' Autorizadas<br/>');
        $listaTipoSolicitud->addOpcion('todas',' Todas');
        $listaTipoSolicitud->setSelected('todas');
        $this->addLista($listaTipoSolicitud);

        //declaración de los campos fecha en el panel de búsqueda
        $fil_fecha_ini =  new gvHidraDate(false);
        $fil_fecha_ini->setCalendar(true);
        $fil_fecha_ini->setDayOfWeek('short');
        $fil_fecha_ini->enableInputMask(true);
        $this->addFieldType('fil_fechaIni',$fil_fecha_ini);
        $fil_fecha_fin =  new gvHidraDate(false);
        $fil_fecha_fin->setCalendar(true);
        $fil_fecha_fin->setDayOfWeek('short');
        $fil_fecha_fin->enableInputMask(true);
        $this->addFieldType('fil_fechaFin',$fil_fecha_fin);
        //declaración de los campos fecha en el panel lis
        $fecha_ini =  new gvHidraDate(false);
        $fecha_ini->setDayOfWeek('short');
        $fecha_ini->enableInputMask(false);
        $fecha_ini->enableServerValidation(false);
        $this->addFieldType('fecha_ini',$fecha_ini);

        $fecha_fin =  new gvHidraDate(false);
        $fecha_fin->setDayOfWeek('short');
        $fecha_fin->enableInputMask(false);
        $fecha_fin->enableServerValidation(false);
        $this->addFieldType('fecha_fin',$fecha_fin);


        //El periodo a mostrar en el desplegable será por defecto el actual.
        //Otro valor posible serÃ¡ el periodo anterior cuando la fecha sea hasta el 1 de febrero,
        //o el periodo posterior si la fecha es despuÃ©s del 1 de febrero.
        $fechaActual = date('d/m/Y');
        $fechaPartida = explode('/',$fechaActual);
        $listaTipoSolicitud = new IgepLista('fil_periodo');
        $listaTipoSolicitud->addOpcion('',' ');
        $listaTipoSolicitud->addOpcion($fechaPartida[2]-1,' '.($fechaPartida[2]-1));
        $listaTipoSolicitud->addOpcion($fechaPartida[2],' '.$fechaPartida[2]);
        $listaTipoSolicitud->addOpcion($fechaPartida[2]+1,' '.($fechaPartida[2]+1));
        $listaTipoSolicitud->setSelected($fechaPartida[2]);
        $this->addLista($listaTipoSolicitud);

        $listaMotivo = new IgepLista('fil_motivo','MOTIVO');
        $listaMotivo->addOpcion('',' ');
        $this->addLista($listaMotivo);

        // El usuario se cargará de la siguiente manera:
        // Seleccionaremos de la vista VPER_OCUPACION el DNI,nombre y apellidos de la   persona cuyo nregpgv coincida
        // con el "nrp" del usuario conectado en la sesiÃ³n (en TCOM_USUARIOS).
        $usuarioConectado = IgepSession::dameDatosUsuario();
```

```
        $datosUsuario = $this->consultar('SELECT nregpgv,dnombre, dapell1, dapell2 FROM vper_ocupacion WHERE
                                                                                nregpgv=\''.$usuarioConectado['nrp'].'\'');
        $this->datosUsuario = $datosUsuario[0];
        $this->informacionUsuarioActivo = substr($datosUsuario[0]['nregpgv'], 0,8).'-'.$datosUsuario[0]['dnombre'].'
                                                '.$datosUsuario[0]['dapell1'].' '.$datosUsuario[0]['dapell2'];
        $this->keepFilterValuesAfterSearch(true);
}//Fin de Constructor


function preBuscar($objDatos){
        // En función de si se quieren visualizar las solicitudes pendientes de firma, se utiliza una select u otra, en la
        // que se filtra por los parÃ¡metros de entrada, siendo solo obligatorio el periodo

        $pcclasedclase = $objDatos->getValue('fil_motivo');
        $pcclasedclasePartida= explode('|',$pcclasedclase);
        $pcclase = $pcclasedclasePartida[0];

        $pfechaIni = $objDatos->getValue('fil_fechaIni');
        $pfechaFin = $objDatos->getValue('fil_fechaFin');


        $pPeriodo = $objDatos->getValue('fil_periodo');
        $ptipoSolicitudes = $objDatos->getValue('fil_solicitudes');

        $pnregpgv =$this->datosUsuario['nregpgv'];
            $str_where = '';
        if($pcclase!='')
                $str_where = "s.cclase = '".$pcclase."'";

        if($pnregpgv!=''){
                if($str_where!='')
                        $str_where.=' AND ';
                $str_where.= " s.NREGPGV = '".$pnregpgv."' ";
        }



        if($pfechaIni!=null AND $pfechaFin!=null){
                 if($str_where!='')
                        $str_where.=' AND ';
        $str_where.= "(s.fecha_ini <= '".$this->obj_conexion->prepararFecha($pfechaFin)."') AND (s.fecha_fin >= '".$this-
                                                >obj_conexion->prepararFecha($pfechaIni)."' OR s.fecha_fin is null ) ";
        }
        if($pfechaIni!=null AND $pfechaFin==null){
                if($str_where!='')
                        $str_where.=' AND ';
                $str_where.= "(s.fecha_fin >= '".$this->obj_conexion->prepararFecha($pfechaIni)."' OR s.fecha_fin is
                                                                                                        null ) ";
        }
        if($pfechaIni==null AND $pfechaFin!=null){
                if($str_where!='')
                        $str_where.=' AND ';
                $str_where.= "(s.fecha_ini <= '".$this->obj_conexion->prepararFecha($pfechaFin)."'  ) ";
        }



        if($ptipoSolicitudes=='pendientes' || $ptipoSolicitudes=='firmadas' ){
                if($str_where!='')
                        $str_where.=' AND ';
                if($ptipoSolicitudes=='pendientes')
                $str_where.=" NOT((s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado ='AP' and s.fanulaper
                        is null) OR  (s.ffirmaresolucion is not null and  s.situacion ='A' and s.cestado ='NP')
                        OR  (s.ffirmaresolucionanula is not null and  s.situacion ='B' and s.cestado ='AP')
                        OR  (s.ffirmaresolucionanula is not null and  s.situacion ='A' and s.cestado ='AP' and
                        s.fanulaper is not null))";
                elseif($ptipoSolicitudes=='firmadas')
                $str_where.="((s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado ='AP' and s.fanulaper is
                        null) OR  (s.ffirmaresolucion is not null and  s.situacion ='A' and s.cestado ='NP')
                        OR  (s.ffirmaresolucionanula is not null and  s.situacion ='B' and s.cestado ='AP')
                        OR  (s.ffirmaresolucionanula is not null and  s.situacion ='A' and s.cestado ='AP' and
                        s.fanulaper is not null))";
        }
        if($str_where!=''){
                $str_where=' ('.$str_where.') ';
                $this->setParametrosBusqueda($str_where);
        }
        if($pcclase!=''){
                $fechaactual = new gvHidraTimeStamp();
                $fechaactual->setTime(0,0,0);
                SolicitudesYPermisos::f_diasPermiso($pPeriodo, $fechaactual, $fechaactual, $pcclase, $total_dias,
                                                $total_diaslabo, $total_dias_act, $total_diaslabo_act,
                                                'S',$null1,$null2, $pnregpgv, $pcclase);

                $ptotal_dias = $total_dias - $total_dias_act;
                $ptotal_diaslabo = $total_diaslabo - $total_diaslabo_act;

                //  Además le restamos el total de días de anulación solicitados para la clase de permiso seleccionada
                SolicitudesYPermisos::f_calcularDiasPermisoAnyoAnulacion($pnregpgv, $pPeriodo, $pcclase,
                                        $totalDiasLaboAnula, $totalDiasAnula, $nsolicitudesAnula, $fechasAnulacion,
                                        $nsolicitudesAnulaResueltas);

                $ptotal_dias -= $totalDiasAnula;
                $ptotal_diaslabo -= $totalDiasLaboAnula;


        }
        else{
                $ptotal_dias = 0;
```

```
                $ptotal_diaslabo = 0;
        }


        //echo "$ptotal_dias,$ptotal_diaslabo,$informacionDiasHabiles,$informacionDiasNaturales";die;
        $this->informacionDiasHabiles = $ptotal_diaslabo;
        $this->informacionDiasNaturales = $ptotal_dias;
        $this->periodoActual = $pPeriodo;
        $this->pcclase = $pcclase;

      return 0;
}


function postBuscar($objDatos){

        $m_datos = $objDatos->getAllTuplas();

        $ptipoSolicitudes = $objDatos->getValue('fil_solicitudes');

        if(count($m_datos)>0){
                foreach($m_datos as $index=> $tupla){


                        // - La solicitud está pendiente de completar y tiene avisos (estado PC y avisos no es nula):
                        //   Mostrar la imagen de con avisos y pendiente de completar (conAvisosY PC.gif) en la parte
                             izquierda de la pantalla
                        // - La solicitud está pendiente de completar (estado PC):
                        //   Mostrar la imagen de pendiente de completar (PC.gif) en la parte izquierda de la
                             pantalla
                        // - La solicitud tiene avisos (avisos no es nula) y no estÃ¡ pendiente de completar (estado
                             diferente de PC):
                        //   Mostrar la imagen de avisos (ok.gif) en la parte izquierda de la pantalla
                        if($tupla['cestado']=='PC'){
                                if($tupla['avisos']!='')
                                        $m_datos[$index]['img_avisos'] = 'images/conAvisosY PC.gif';
                                else
                                        $m_datos[$index]['img_avisos'] = 'images/PC.gif';
                                        }
                        elseif($tupla['avisos']!='')
                                $m_datos[$index]['img_avisos'] = 'images/ok.gif';
                        else
                                $m_datos[$index]['img_avisos'] = 'images/nada.gif';

                        // Añadimos el color a las filas
                        // Se deberán de mostrar solicitudes resaltadas con color verde cuando estén en las situaciones
                           siguientes:

                        // - La solicitud está autorizada (situacion = A, estado = AP, fanulaper nula, ffirmaresolucion
                             no nula o
                        // situacion = A, estado = AP, fanulaper no nula, ffirmaresolucionanula no nula   o
                        // situacion = B, estado = AP, ffirmaresolucionanula no nula) :
                        // Mostrar con fondo verde
                        $color='none';

                        if(($tupla['situacion']=='A' AND $tupla['cestado']=='AP' AND empty($tupla['fanulaper']) AND !
                                                        empty($tupla['ffirmaresolucion']))
                          OR ($tupla['situacion']=='A' AND $tupla['cestado']=='AP' AND !empty($tupla['fanulaper']) AND !
                                                        empty($tupla['ffirmaresolucionanula']))
                          OR ($tupla['situacion']=='B' AND $tupla['cestado']=='AP' AND !
                                                        empty($tupla['ffirmaresolucionanula']))

                                        )

                                $color='sugerencia';



                        $objDatos->setRowColor($m_datos[$index],$color);
                }
        $objDatos->setAllTuplas($m_datos);
        }
        return 0;
}


function saltoDeVentana( $objDatos, $objSalto){

        $objDatos->setOperacion('seleccionar');
        $m_datos = $objDatos->getAllTuplas();
        if($m_datos==0){
                  $this->showMensaje('APL-10');
                return -1;
        }
        $objSalto->setClase('DetalleSolicitud');
        $objSalto->setAccion('buscar');
        $objSalto->setAccionRetorno('refrescarDatos');
        $solicitudes = array();
        foreach($m_datos as $solicitud)
                  array_push($solicitudes,$solicitud['nsolicitud']);
        $objSalto->setParam('solicitudes',$solicitudes);
        ClasesDePermisos::f_dameDatosClasePermiso($m_datos[0]['cclase'],$dclase,$justificante);


        $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');

        foreach($permisosDGAA as $index=>$tupla){

                $var_cclases=$var_cclases."'".$permisosDGAA[$index]['cclase']."',";
```

```
            }
            $var_cclases=rtrim($var_cclases,",");
            $var_tupla=$tupla['cclase'];
            $pos=strpos ($var_cclases,$var_tupla);


            //Si situacion='A' and estado='NR'
            if(trim($m_datos[0]['situacion'])=='A' and $m_datos[0]['cestado']=='NR')
                    $objSalto->setParam('visibilidad','ConsultaEstadoMisSolicitudes_noAutorizadaPorResponsable');
            // También se incluye el caso en que esté marcado requierejustificante

            elseif(($justificante=='S') or ($pos>0) OR $m_datos[0]['requierejustificante']=='S')

                    $objSalto->setParam('visibilidad','ConsultaEstadoMisSolicitudes_conJustificante');
            else
                    $objSalto->setParam('visibilidad','ConsultarSolicitudesACargoResponsable_NoPersonal');

            $perfil = IgepSession::dameRol();
            if ($perfil!='P_USUARIO')
                    $objSalto->setParam('mostrarValidacion','1');
            return 0;
}


function accionesParticulares($str_accion, $objDatos) {
        switch ($str_accion)
        {
            case 'volverDeDetalleSolicitud':
                $actionForward = $objDatos->getForward('correcto');
                break;
            case 'refrescarDatos':
                $actionForward = $objDatos->getForward('correcto');
                break;

            default:
                return -1;
        }
        return $actionForward;
    }
}//Fin ConsultaEstadoMisSolicitudes
?>
```

Tpl– Consulta: Estado de mis solicitudes

```
{CWVentana tipoAviso=$smty_tipoAviso codAviso=$smty_codError descBreve=$smty_descBreve textoAviso=$smty_textoAviso
                                                                                                    onLoad=$smty_jsOnLoad}

{CWBarra customTitle=$smty_customTitle usuario=$smty_usuario codigo=$smty_codigo}
        {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}

        <!--*********** PANEL fil ******************-->
        {CWPanel id="fil" action="buscar" method="post" estado="$estado_fil" claseManejadora="ConsultaEstadoMisSolicitudes"}
                {CWBarraSupPanel titulo="Consulta del estado de mis solicitudes"}
                        {CWBotonTooltip imagen="04" titulo="Restaurar valores" funcion="limpiar" actuaSobre="ficha" }
                {/CWBarraSupPanel}
                {CWContenedor}
                        {CWFicha}
                        <table  cellpadding="5" width="80%">
                        <tr>
                                <td class="formularios" width="3%">
                                         
                                </td>
                                <td class="formularios" colspan="2">
                                         
                                </td>
                        </tr>
                        <tr>
                                <td class="formularios" width="3%">
                                         
                                </td>
                                <td class="formularios" colspan="2">
                                        {CWLista nombre="fil_periodo" editable="true" textoAsociado="Periodo"
                                                datos=$defaultData_ConsultaEstadoMisSolicitudes.fil_periodo
                                                value=$defaultData_ConsultaEstadoMisSolicitudes.fil_periodo}
                                </td>
                        </tr>
                        <tr>
                                <td class="formularios" width="3%">
                                         
                                </td>
                                <td class="formularios">
                                        {CWCampoTexto nombre="fil_fechaIni"  maxlength="10" size="10"
                                                editable="true" textoAsociado="Fecha inicio"
                                                dataType=$dataType_ConsultaEstadoMisSolicitudes.fil_fechaIni
                                                value=$defaultData_ConsultaEstadoMisSolicitudes.fil_fechaIni}
                                </td>
                                <td class="formularios">
                                        {CWCampoTexto nombre="fil_fechaFin"  maxlength="10" size="10"
                                                editable="true" textoAsociado="Fecha fin"
                                                dataType=$dataType_ConsultaEstadoMisSolicitudes.fil_fechaFin
                                                value=$defaultData_ConsultaEstadoMisSolicitudes.fil_fechaFin}
                                </td>
                        </tr>
                        </table>
                        <table  cellpadding="5" width="80%">
                        <tr>
                                <td class="formularios" width="3%">
                                         
                                </td>
                                <td class="formularios" width="70%">
                                        {CWLista nombre="fil_motivo" editable="true" textoAsociado="Motivo"
                                                datos=$defaultData_ConsultaEstadoMisSolicitudes.fil_motivo}
                                <br/><br/>
                                </td>
                                <td class="formularios" width="27%">
                                <fieldset style="  border-bottom-width: thin; border-color: silver">
                                <legend align='left' style="font-weight:bold;">Solicitudes</legend>
                                {CWLista nombre="fil_solicitudes" editable="true"
                                                datos=$defaultData_ConsultaEstadoMisSolicitudes.fil_solicitudes
                                                dataType=$dataType_ConsultaEstadoMisSolicitudes.fil_solicitudes}
                                </fieldset>
                                </td>
                        </tr>
                        </table>
                        <br/>
                        {/CWFicha}
                {/CWContenedor}
                {CWBarraInfPanel}
                        {CWBoton imagen="50" texto="Buscar" class="boton" accion="buscar" }
                {/CWBarraInfPanel}
        {/CWPanel}


        <!-- ****************** PANEL lis ***********************-->
        {CWPanel id="lis" action="editar" method="post" estado=$estado_lis claseManejadora="ConsultaEstadoMisSolicitudes"}
                {CWBarraSupPanel titulo="Consulta del estado de mis solicitudes"}
                        {CWBotonTooltip imagen="detalle" titulo="Ver detalle" funcion="saltar" actuaSobre="ficha"
                                                                                        id="DetalleSolicitud"}
                        {CWBotonTooltip imagen="04" titulo="Refrescar" funcion="modificar" actuaSobre="ficha"
                                                                action="ConsultaEstadoMisSolicitudes__refrescarDatos"}
                {/CWBarraSupPanel}
                {CWContenedor}
                        <p align="center" class="tabla_titulo">Interesado: {$smty_usuarioActivo}</p>

                        {if $smty_clasePermiso <> ''} <!-- Si se ha seleccionado un motivo(tipo) de permiso, se muestra
                                                        los dias habiles y naturales gastados para ese permiso-->
                        <p align="center" class="tabla_titulo">
                                Total días hábiles solicitados {$smty_periodo}: {$smty_usuarioDiasHabiles}
```

```
                                                
                                        Total días naturales solicitados {$smty_periodo}: {$smty_usuarioDiasNaturales}
                                    </p>

                                    {/if} <!-- FIN -->

                                    {CWTabla conCheck="true" seleccionUnica="true" id="Tabla1" numPagInsertar="0"
                                                                        numFilasPantalla="15" datos=$smty_datosTabla}
                                        {CWFila tipoListado="false"}
                                            {CWImagen nombre="img_avisos" textoAsociado=" " src="images/nada.gif"}
                                            {CWCampoTexto nombre="nsolicitud" size="6" editable="false"
                                                                                    textoAsociado="NÂº sol."}
                                            {CWCampoTexto nombre="dclase" size="45" editable="false"
                                                                                    textoAsociado="Motivo"}

                                            {CWCampoTexto nombre="fecha_ini" size="10" editable="false"
                                                                    textoAsociado="F. inicio"
                                                    dataType=$dataType_ConsultaEstadoMisSolicitudes.fecha_ini}
                                            {CWCampoTexto nombre="fecha_fin" size="10" editable="false"
                                                                    textoAsociado="F. fin"
                                                    dataType=$dataType_ConsultaEstadoMisSolicitudes.fecha_fin}
                                            {CWCampoTexto nombre="destado" size="27" editable="false"
                                                                                    textoAsociado="Estado"}
                                            {CWCampoTexto nombre="cestado" oculto="true"}
                                            {CWCampoTexto nombre="justificante" oculto="true"}
                                            {CWCampoTexto nombre="situacion" oculto="true"}
                                            {CWCampoTexto nombre="cclase" oculto="true"}
                                            {CWCampoTexto nombre="requierejustificante" oculto="true"}
                                        {/CWFila}

                                        {CWPaginador enlacesVisibles="3"}
                                    {/CWTabla}
                                {/CWContenedor}
                                {CWBarraInfPanel}
                                <img src="images/soliAut.gif"/> 
                                {/CWBarraInfPanel}
                        {/CWPanel}

                        <!-- ***************** PESTAÑAS ************************-->
                        {CWContenedorPestanyas}
                                {CWPestanya tipo="fil" estado=$estado_fil}
                                {CWPestanya tipo="lis" estado=$estado_lis}
                        {/CWContenedorPestanyas}
                {/CWMarcoPanel}
                {/CWVentana}
```

Views– Consulta: Estado de mis solicitudes

```php
<?php

$comportamientoVentana= new IgepPantalla();

$panel = new IgepPanel('ConsultaEstadoMisSolicitudes',"smty_datosTabla");
$panel->activarModo("fil","estado_fil");
$panel->activarModo("lis","estado_lis");
$comportamientoVentana->agregarPanel($panel);


//Incorporamos los datos del usuario
$usuario = IgepSession::dameVariable('ConsultaEstadoMisSolicitudes','informacionUsuarioActivo');
$s->assign('smty_usuarioActivo',$usuario);

//Total dias habiles solicitados (en el caso de haber marcado un determinado motivo o tipo de permiso)
$DiasHabiles = IgepSession::dameVariable('ConsultaEstadoMisSolicitudes','informacionDiasHabiles');
$s->assign('smty_usuarioDiasHabiles',$DiasHabiles);

//Total dias naturales solicitados (en el caso de haber marcado un determinado motivo o tipo de permiso)
$DiasNaturales = IgepSession::dameVariable('ConsultaEstadoMisSolicitudes','informacionDiasNaturales');
$s->assign('smty_usuarioDiasNaturales',$DiasNaturales);

//Para ocultar o mostrar los dias habiles y naturales, segÃºn se haya seleccionado o no un determinado permiso
$clasePermiso = IgepSession::dameVariable('ConsultaEstadoMisSolicitudes','pcclase');
$s->assign("smty_clasePermiso",$clasePermiso);

//periodo actual
$Periodo = IgepSession::dameVariable('ConsultaEstadoMisSolicitudes','periodoActual');
$s->assign('smty_periodo',$Periodo);
//$s->assign('smty_periodo',date('Y'));


$s->display('consultas/ConsultaEstadoMisSolicitudes.tpl');
?>
```

Clase Manejadora – Consulta: DetalleSolicitud

```
<?php

class DetalleSolicitud extends gvHidraForm_DB{

    function DetalleSolicitud(){

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');

        $nombreTablas= array('tper_solicitudes');
        parent::__construct($g_dsn,$nombreTablas);

        //Comprobamos los datos del salto
        $objSalto = IgepSession::damePanel('saltoIgep');
        if(is_object($objSalto))
            $this->parametrosBusqueda = $objSalto->getParams();

        if($this->parametrosBusqueda['mostrarValidacion']==1){
            $estado = "CASE
                                            WHEN s.cestado = 'PT'
                                                    THEN e.destado || ' (pendiente tramitador)'
                                WHEN s.cestado = 'PV' and s.valida = 'S' and s.revisada='N' and s.situacion='A'
                                                    then e.destado || ' (vÃ¡lida por tramitador)'
                                            WHEN s.cestado = 'PV' and s.valida = 'N' and s.revisada='N' and
                                                    s.situacion='A'
                                                    then e.destado || ' (no vÃ¡lida por tramitador)'
                                            WHEN s.cestado = 'PV' and s.validaanula = 'S' and s.situacion='B'
                                                    then e.destado || ' (vÃ¡lida por tramitador)'
                                            WHEN s.cestado = 'PV' and s.validaanula = 'N' and s.situacion='B'
                                                    then e.destado || ' (no vÃ¡lida por tramitador)'
                                WHEN s.cestado = 'PP' THEN e.destado || ' (vÃ¡lida por revisor)'
                                  WHEN s.cestado = 'PN' THEN e.destado || ' (no vÃ¡lida por revisor)'
                                  WHEN s.cestado = 'PC' THEN e.destado || ' y pendiente de completar'

                                        WHEN s.fresolucion is null and s.situacion ='A' and s.cestado='AP' and s.fanulaper is
                                                                                                                        null
                                        THEN e.destado || ' (vÃ¡lida por personal)'
                                WHEN s.ffirmaresolucion is null and fresolucion is not null and s.situacion ='A' and
                                                                s.cestado='AP'and s.fanulaper is null
                                        THEN e.destado || ' (pendiente de firma positiva)'
                                WHEN s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado='AP' and s.fanulaper is
                                                                null
                                        THEN 'Autorizada'

                                WHEN s.fresolucion is null and s.situacion ='A' and s.cestado='NP'
                                        THEN e.destado || ' (no vÃ¡lida por personal)'
                                WHEN s.ffirmaresolucion is null and fresolucion is not null and s.situacion ='A' and
                                                                                                        s.cestado='NP'
                                        THEN e.destado || ' (pendiente de firma negativa)'
                                WHEN s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado='NP'
                                        THEN 'No autorizada'

                                WHEN s.fresolucionanula is null and s.situacion ='B' and s.cestado='AP'
                                        THEN e.destado || ' (vÃ¡lida por personal)'
                                WHEN s.ffirmaresolucionanula is null and fresolucionanula is not null and s.situacion ='B'
                                                                                                and s.cestado='AP'
                                        THEN e.destado || ' (pendiente de firma positiva)'
                                WHEN s.ffirmaresolucionanula is not null and s.situacion ='B' and s.cestado='AP'
                                        THEN 'Autorizada'

                                WHEN s.fresolucionanula is null and s.situacion ='A' and s.cestado='AP' and s.fanulaper is not
                                                                                                                        null
                                        THEN e.destado || ' (no vÃ¡lida por personal)'
                                WHEN s.ffirmaresolucionanula is null and fresolucionanula is not null and s.situacion ='A'
                                                and s.cestado='AP' and s.fanulaper is not null
                                        THEN e.destado || ' (pendiente de firma negativa)'
                                WHEN s.ffirmaresolucionanula is not null and s.situacion ='A' and s.cestado='AP'
                                                and s.fanulaper is not null
                                        THEN 'No autorizada'
                                ELSE e.destado
                                END as \"destado\"";
        }
        else
            $estado = "CASE WHEN s.cestado = 'PC' THEN e.destado || ' y pendiente de completar'
                                        WHEN s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado='AP' and
                                                                                                        s.fanulaper is null
                                                THEN 'Autorizada'
                                        WHEN s.ffirmaresolucion is not null and s.situacion ='A' and s.cestado='NP'
                                                THEN 'No autorizada'
                                        WHEN s.ffirmaresolucionanula is not null and s.situacion ='B' and s.cestado='AP'
                                                THEN 'Autorizada'
                                        WHEN s.ffirmaresolucionanula is not null and s.situacion ='A' and s.cestado='AP'
                                                                and s.fanulaper is not null
                                                THEN 'No autorizada'
                                        ELSE e.destado END as \"destado\"";

        //Segun el perfil del usuario mostramos unos avisos u otros
        $perfil = IgepSession::dameRol();
        if($perfil=='P_RESPPER' OR $perfil=='P_TRAMITA' OR $perfil=='P_REVISOR')
                    $avisos = 's.avisos';
        else
          $avisos = 's.avisosusuario';


        $str_select="SELECT u.nregpgv, substr(u.nregpgv,0,length(u.nregpgv)-1)||' - '||u.dapell1 || ' ' ||
```

```
            CASE WHEN u.dapell2 is null THEN ' ' ELSE u.dapell2 END||', '||u.dnombre as \"solicitante\", s.situacion
            as \"situacionsolicitud\", s.nsolicitud,s.fsolicitud, s.periodo, c.dclase, s.fecha_ini,
            s.fecha_fin, s.observaciones, $avisos as \"avisos\", s.motivonoautresp,s.motivonoautper,
            $estado, s.justificante,s.pendientenuevojustificante as \"justificante_correcto\",
            s.requierejustificante as \"requierejustificante\",
            s.ffirmaresp,
            u1.dapell1 || ' ' || CASE WHEN u1.dapell2 is null THEN ' ' ELSE u1.dapell2 END || ', ' ||
                          u1.dnombre as \"firmaresp\",
            s.fvalidatramitador,
            u5.dapell1 || ' ' || CASE WHEN u5.dapell2 is null THEN ' ' ELSE u5.dapell2 END || ', ' ||
                                                          u5.dnombre as \"validatramitador\",
            s.fvalidarevisor, u7.dapell1 || ' ' || CASE WHEN u7.dapell2 is null THEN ' ' ELSE u7.dapell2 END
                                                    || ', ' || u7.dnombre as \"validarevisor\",
            s.ffirmaper,  u2.dapell1 || ' ' || CASE WHEN u2.dapell2 is null THEN ' ' ELSE u2.dapell2 END ||
                                                          ', ' || u2.dnombre as \"firmaper\",
            s.fresolucion,s.numresol, u9.dapell1 || ' ' || CASE WHEN u9.dapell2 is null THEN ' ' ELSE
            u9.dapell2 END || ', ' || u9.dnombre as \"generaresol\",
            CASE WHEN s.ffirmaresolucion is null THEN 'N' ELSE 'S' END as \"firmaresolucion\",
            u11.dapell1 || ' ' || CASE WHEN u11.dapell2 is null THEN ' ' ELSE u11.dapell2 END || ', ' ||
                                                          u11.dnombre as \"notificaresol\",
            s.fanularesp, u3.dapell1 || ' ' || CASE WHEN u3.dapell2 is null THEN ' ' ELSE u3.dapell2 END ||
                                                          ', ' || u3.dnombre as \"anularesp\",
            s.fvalidaanulatramitador, u6.dapell1 || ' ' || CASE WHEN u6.dapell2 is null THEN ' ' ELSE
                                        u6.dapell2 END || ', ' || u6.dnombre as \"validaanulatramitador\",
            s.fvalidaanularevisor, u8.dapell1 || ' ' || CASE WHEN u8.dapell2 is null THEN ' ' ELSE
                                        u8.dapell2 END || ', ' || u8.dnombre as \"validaanularevisor\",
            s.fanulaper,  u4.dapell1 || ' ' || CASE WHEN u4.dapell2 is null THEN ' ' ELSE u4.dapell2 END ||
                                                          ', ' || u4.dnombre as \"anulaper\",
            s.cestado,s.gradoparentesco,s.distancia ,s.fnacimientomenor,s.fresolucionanula,
            s.numresolanula,
            u10.dapell1 || ' ' || CASE WHEN u10.dapell2 is null THEN ' ' ELSE u10.dapell2 END || ', ' ||
                                                          u10.dnombre as \"generaresolanula\",
            CASE WHEN s.ffirmaresolucionanula is null THEN 'N' ELSE 'S' END as \"firmaresolucionanula\",
            u12.dapell1 || ' ' || CASE WHEN u12.dapell2 is null THEN ' ' ELSE u12.dapell2 END || ', ' || u12.dnombre
                                                          as \"notificaresolanula\",
            CASE WHEN s.situacion='A' THEN 'ALTA' ELSE 'ANULACIÃ“N' END as \"situacion\", c.cclase as \"cclase\", ''
                     as \"diasSinJustificanteENC\", '' as \"diasConJustificanteENC\"
            FROM tper_cod_permisos c, TPER_ESTADOSOL e, vper_ocupacion_historico u,
                     tper_solicitudes s LEFT JOIN vper_ocupacion_historico u1 ON s.nrpresp = u1.nregpgv
                     LEFT JOIN vper_ocupacion_historico u5 ON s.nrptramitador = u5.nregpgv
                     LEFT JOIN vper_ocupacion_historico u7 ON s.nrprevisor = u7.nregpgv
                     LEFT JOIN vper_ocupacion_historico u2 ON s.nrpper = u2.nregpgv
                     LEFT JOIN vper_ocupacion_historico u9 ON s.nrpgeneraresol = u9.nregpgv
                     LEFT JOIN vper_ocupacion_historico u11 ON s.nrpnotificaresol = u11.nregpgv
                     LEFT JOIN vper_ocupacion_historico u3 ON s.nrpanularesp = u3.nregpgv
                     LEFT JOIN vper_ocupacion_historico u6 ON s.nrpanulatramitador = u6.nregpgv
                     LEFT JOIN vper_ocupacion_historico u8 ON s.nrpanularevisor = u8.nregpgv
                     LEFT JOIN vper_ocupacion_historico u4 ON s.nrpanulaper = u4.nregpgv
                     LEFT JOIN vper_ocupacion_historico u10 ON s.nrpgeneraresolanula = u10.nregpgv
                     LEFT JOIN vper_ocupacion_historico u12 ON s.nrpnotificaresolanula = u12.nregpgv";

$this->setSelectForSearchQuery($str_select);
$this->setWhereForSearchQuery("s.cclase = c.cclase and
                s.nregpgv = u.nregpgv and
                s.cestado = e.cestado");
$this->setOrderByForSearchQuery("s.nsolicitud");

$this->addMatching('nsolicitud','nsolicitud','tper_solicitudes');
$this->addMatching('justificante','justificante','tper_solicitudes');
$this->addMatching('justificante_correcto','pendientenuevojustificante','tper_solicitudes');
$this->addMatching('requierejustificante','requierejustificante','tper_solicitudes');
$this->addMatching('observaciones','observaciones','tper_solicitudes');

            //Definicion de los tipos de los campos
$fecha_ini =  new gvHidraDate(false);
            $fecha_ini->setDayOfWeek('short');
            $fecha_ini->enableInputMask(false);
            $fecha_ini->enableServerValidation(false);
            $this->addFieldType('fecha_ini',$fecha_ini);

$fecha_fin =  new gvHidraDate(false);
            $fecha_fin->setDayOfWeek('short');
            $fecha_fin->enableInputMask(false);
            $fecha_fin->enableServerValidation(false);
            $this->addFieldType('fecha_fin',$fecha_fin);

$observaciones =  new gvHidraString(false,500);
            $this->addFieldType('observaciones',$observaciones);

            //
            $fecha_nacimientomenor =  new gvHidraDate(false);
            $fecha_nacimientomenor->setDayOfWeek('short');
            $fecha_nacimientomenor->enableInputMask(false);
            $fecha_nacimientomenor->enableServerValidation(false);
            $this->addFieldType('fnacimientomenor',$fecha_nacimientomenor);

            $listaGrado = new IgepLista('gradoparentesco');
            $listaGrado->addOpcion("1","1er grado");
            $listaGrado->addOpcion("2","2Âº grado");
            $this->addLista($listaGrado);

            $listaDistancia= new IgepLista('distancia');
            $listaDistancia->addOpcion("C1","A no mÃ¡s de 100 Km");
            $listaDistancia->addOpcion("C2","A no mÃ¡s de 375 Km");
            $listaDistancia->addOpcion("L1","A mÃ¡s de 100 Km");
            $listaDistancia->addOpcion("L2","A mÃ¡s de 375 Km");
            $this->addLista($listaDistancia);

            //justificante
            $checkJustificante = new IgepCheckBox('justificante');
            $checkJustificante->setValueChecked('S');
            $checkJustificante->setValueUnchecked('N');
```

```
                    $checkJustificante->setChecked(false);
                    $this->addCheckBox($checkJustificante);

                    //requiere justificante
                    $checkRequiereJustificante = new IgepCheckBox('requierejustificante');
                    $checkRequiereJustificante->setValueChecked('S');
                    $checkRequiereJustificante->setValueUnchecked('N');
                    $checkRequiereJustificante->setChecked(false);
                    $this->addCheckBox($checkRequiereJustificante);


                    //        justificante Correcto  T 5622
                    $checkJustificanteCorrecto = new IgepCheckBox('justificante_correcto');
                    $checkJustificanteCorrecto->setValueChecked('S');
                    $checkJustificanteCorrecto->setValueUnchecked('N');
                    $checkJustificanteCorrecto->setChecked(false);
                    $this->addCheckBox($checkJustificanteCorrecto);

                    //firmaresolucion
                    $checkFirmaresolucion = new IgepCheckBox('firmaresolucion');
                    $checkFirmaresolucion->setValueChecked('S');
                    $checkFirmaresolucion->setValueUnchecked('N');
                    $checkFirmaresolucion->setChecked(false);
                    $this->addCheckBox($checkFirmaresolucion);

                    //firmaresolucionanula
                    $checkFirmaresolucionanula = new IgepCheckBox('firmaresolucionanula');
                    $checkFirmaresolucionanula->setValueChecked('S');
                    $checkFirmaresolucionanula->setValueUnchecked('N');
                    $checkFirmaresolucionanula->setChecked(false);
                    $this->addCheckBox($checkFirmaresolucionanula);


          $this->addAccionInterfaz('justificante','textoJustificante');

    }//Fin de Constructor



function textoJustificante($objDatos){


    // Incluir de forma automatica la palabra JUSTIFICANTE en el campo
    // Observaciones en el caso de que se seleccion la casilla justificante.

    $s_observaciones=$objDatos->getvalue('observaciones');
    if  (substr($s_observaciones, 0, 14)=="JUSTIFICANTE. "){
          $hayTextoJustificante="S";
    } else{
          $hayTextoJustificante="N";
    }
    $check_justificante=$objDatos->getvalue('justificante');
    if (($check_justificante=='S') and ( $hayTextoJustificante=="N")){
          $s_observaciones="JUSTIFICANTE. ".$s_observaciones;
          $objDatos->setvalue('observaciones',$s_observaciones);
    }elseif (($check_justificante=='N') and $hayTextoJustificante=="S") {
          $s_observaciones=substr($s_observaciones, 14, strlen($s_observaciones));
          $objDatos->setvalue('observaciones',$s_observaciones);
    }



   return 0;


}



function preBuscar($objDatos){

        if(count($this->parametrosBusqueda)==0)
            return -1;
        $solicitudes = $this->parametrosBusqueda['solicitudes'];
        $str_sol = implode(',',$solicitudes);
        $str_where=' s.nsolicitud in ('.$str_sol.')';
        $this->setParametrosBusqueda($str_where);
        return 0;
}



function postBuscar($objDatos){

          $m_datos = $objDatos->getAllTuplas();

          // Mostrar días sin justificante y con justificante de Enfermedad Común para el interesado
          $perfil = IgepSession::dameRol();
          if($perfil=='P_TRAMITA' OR $perfil=='P_RESPONSA' OR $perfil=='P_RESPPER')
                    $this->v_mostrarJustificanteEnc = array();

          $this->v_mostrarSolModificada = array();

          $this->fresolucionnumresol = array();
          $this->v_mostrarFechaNacimiento = array();
```

```
$this->v_mostrarDistancia = array();
$this->v_mostrarGradoParentesco = array();
$this->v_grado=array();
$opcionMenu = IgepSession::dameVariable('ConsultaSolicitudesACargoResponsable','opcionMenu');
foreach($m_datos as $indice=>$tupla){
          // TAREA 6031 09/02/2010 Mostrar el nº de resolución en el detalle de las solicitudes con resolución
          positiva
          if($perfil=='P_REVISOR' OR    $perfil=='P_RESPPER' OR $perfil=='P_TRAMITA') {

                    if(($tupla['situacionsolicitud']=='B' and $tupla['cestado']=='AP' and !
                              is_null($tupla['fanulaper']))
                              OR ($tupla['situacionsolicitud']=='A' and $tupla['cestado']=='AP' and
                              is_null($tupla['fanulaper'])) ){

                              $this->fresolucionnumresolanula =array();

                              if (!empty($tupla['fresolucion']) and $tupla['fresolucion']!='' and
                                        isset($tupla['fresolucion']) ){

                                                  if ($opcionMenu == 'DGA'){
                                                            $m_datos[$indice]['fresolucion'] = $tupla['fresolucion'];
                                                            array_push($this->fresolucionnumresol,1);

                                                  }
                                                  else {
                                                            $m_datos[$indice]['fresolucion'] =
                                                                      $tupla['fresolucion'].' - '.$tupla['numresol'];
                                                            array_push($this->fresolucionnumresol,1);
                                                  }
                              }
                              else{
                                        array_push($this->fresolucionnumresol,0);
                              }

                              if (!empty($tupla['fresolucionanula']) and $tupla['fresolucionanula']!='' and
                                                  isset($tupla['fresolucion'])){
                                        if ($opcionMenu == 'DGA'){
                                                  $m_datos[$indice]['fresolucionanula'] =
                                                                                      $tupla['fresolucionanula'];
                                                  array_push($this->fresolucionnumresolanula,1);
                                        }
                                        else{
                                                  $m_datos[$indice]['fresolucionanula'] =
                                                                                $tupla['fresolucionanula'].' -
                                                                                '.$tupla['numresolanula'];
                                                  array_push($this->fresolucionnumresolanula,1);
                                        }
                              }
                    else{
                              array_push($this->fresolucionnumresolanula,0);
                    }
                    }

          }
                    // FIN  TAREA 6031 09/02/2010

          if($perfil=='P_TRAMITA' OR $perfil=='P_RESPONSA' OR $perfil=='P_RESPPER'){
                    if($tupla['cclase']=='ENC'){
                              $diasSinJustificanteENC = '';
                              $diasConJustificanteENC = '';
                              Solicitudes::f_dameDiasENC($tupla['periodo'], $tupla['nregpgv'],
                                                  $diasSinJustificanteENC, $diasConJustificanteENC);
                              $m_datos[$indice]['diasSinJustificanteENC'] = $diasSinJustificanteENC;
                              $m_datos[$indice]['diasConJustificanteENC'] = $diasConJustificanteENC;
                              array_push($this->v_mostrarJustificanteEnc,1);
                    }
                    else
                              array_push($this->v_mostrarJustificanteEnc,0);
          }
          //Si es 'VAC' o 'ADI' mostramos el campo nsolmodificada
          if($tupla['cclase']=='VAC' OR $tupla['cclase']=='ADI')
                    array_push($this->v_mostrarSolModificada,1);
          else
                    array_push($this->v_mostrarSolModificada,0);




          if($tupla['cclase']=='RJL' OR $tupla['cclase']=='AJL')
                    array_push($this->v_mostrarFechaNacimiento,1);
          else
                    array_push($this->v_mostrarFechaNacimiento,0);


          if($tupla['cclase']=='EGP' or $tupla['cclase']=='DEF' OR $tupla['cclase']=='TDO'){

                    array_push($this->v_mostrarDistancia,1);
                    if ($tupla['distancia']=='C')
                              $m_datos[$indice]['distancia']='C1';
                    else
                              $m_datos[$indice]['distancia']='L1';
          }elseif ($tupla['cclase']=='MFA' OR $tupla['cclase']=='MUH'){
                    array_push($this->v_mostrarDistancia,1);
                    if ($tupla['distancia']=='C')
                              $m_datos[$indice]['distancia']='C2';
                    else
                              $m_datos[$indice]['distancia']='L2';
          }
          else
                    array_push($this->v_mostrarDistancia,0);
```

```
                if($tupla['cclase']=='EGP' OR $tupla['cclase']=='DEF')
                        array_push($this->v_mostrarGradoParentesco,1);

                else
                        array_push($this->v_mostrarGradoParentesco,0);


                $var_solicitudesmodificadas = array();
                $listaSolModificaNoResueltas = null;
                SolicitudesModificadas::f_dameSolicitudesModificadas($tupla['nsolicitud'],'N',
                                                                        $listaSolModificaNoResueltas);
                foreach($listaSolModificaNoResueltas as $solModificadaNoResuelta)

                        array_push($var_solicitudesmodificadas,$solModificadaNoResuelta['nsolicitudmodifica']);

                $listaSolModificaResueltas = null;
                SolicitudesModificadas::f_dameSolicitudesModificadas($tupla['nsolicitud'], 'S',
                                                                        $listaSolModificaResueltas);
                foreach($listaSolModificaResueltas as $solModificadaResuelta)
                        array_push($var_solicitudesmodificadas,$solModificadaResuelta['nsolicitudmodifica']);

                $m_datos[$indice]['nsolmodificada'] = implode(',',$var_solicitudesmodificadas);
        }// foreach

        $objDatos->setAllTuplas($m_datos);

          //Segun la visibilidad cambiamos la accion
        $visibilidad = $this->parametrosBusqueda['visibilidad'];
        if($visibilidad=='ConsultarSolicitudesACargoResponsable_Modificacion'){
            $this->v_botonImprimir = array();
            $this->v_activarJustificante = array();
            $this->v_activarRequiereJustificante = array();
            $this->actualizaCampos = 1;
            //Comprobamos si tenemos que activar el campo justificante y el botÃ³n imprimir para cada una de las fichas.
            foreach($m_datos as $tupla){
                    $dclase = '';
                    $justificante = '';
                    ClasesDePermisos::f_dameDatosClasePermiso($tupla['cclase'],$dclase,$justificante);

                    // Para sacar el botón Imprimir cuando sea una de las tramitadas por la DGAA
                    $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');

                    foreach($permisosDGAA as $index=>$tuplaDGAA){

                            $var_cclases=$var_cclases."'".$permisosDGAA[$index]['cclase']."',";

                    }
                    $var_cclases=rtrim($var_cclases,",");
                    $var_tupla=$tupla['cclase'];
                    $pos=strpos ($var_cclases,$var_tupla);


                    //Si tiene justificante OR si lo requiere OR si es una sol. tramitada por la DGAA OR  es VAC o ADI con
                    cambio.
                    if(($justificante=='S' OR $pos>0 OR $tupla['requierejustificante']=='S' )
                            OR ( $tupla['cclase']=='VAC' OR $tupla['cclase']=='ADI' )
                             AND ( Solicitudes::f_esCambioDeSolicitud($tupla['nsolicitud']) )){

                            array_push($this->v_botonImprimir,1);
                            array_push($this->v_activarJustificante,1);
                    }
                    else{
                            array_push($this->v_botonImprimir,0);
                            array_push($this->v_activarJustificante,0);
                    }


                    if($tupla['cclase']=='ENC'){
                            array_push($this->v_activarRequiereJustificante,1);
                    }
                    else{
                            array_push($this->v_activarRequiereJustificante,0);
                    }

            }


            //Puede estar activo el botÃ³n imprimir
            $this->botonImprimir = 1;
            $this->botonEnDesacuerdo=0;
            //no mostramos la parte de personal
            $this->visualizarPartePersonal = 1;
        }
        elseif($visibilidad=='ConsultarSolicitudesACargoResponsable_PersonalConsulta'){
            //Comprobamos si tenemos que activar el botÃ³n imprimir para cada una de las fichas.
            $this->v_botonImprimir = array();
            foreach($m_datos as $tupla){
                    $dclase = '';
                    $justificante = '';
                    ClasesDePermisos::f_dameDatosClasePermiso($tupla['cclase'],$dclase,$justificante);
                    // También se incluye el caso en que esté marcado requierejustificante
                    if($justificante=='S' OR $tupla['requierejustificante']=='S'){

                            array_push($this->v_botonImprimir,1);
                    }
                    else{
                            array_push($this->v_botonImprimir,0);
                    }
            }
            //No puede actualizar el campo justificante
            $this->actualizaCampos = 0;
            //Puede que este el campo imprimir activo
```

```
        $this->botonImprimir = 1;
        $this->botonEnDesacuerdo=0;
        //no mostramos la parte de personal
        $this->visualizarPartePersonal = 1;
    }
    elseif($visibilidad=='ConsultarSolicitudesACargoResponsable_PersonalConsultaCITDGA'){
        $this->v_botonImprimir = array();

        //Comprobamos si tenemos que activar el campo justificante y el botón imprimir para cada una de las fichas.
        foreach($m_datos as $tupla){
                $dclase = '';
                $justificante = '';
                ClasesDePermisos::f_dameDatosClasePermiso($tupla['cclase'],$dclase,$justificante);

                // Para sacar el botón Imprimir cuando sea una de las tramitadas por la DGAA
                $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');
                foreach($permisosDGAA as $index=>$tuplaDGAA)
                                                $var_cclases=$var_cclases."'".$permisosDGAA[$index]['cclase']."',";
                $var_cclases=rtrim($var_cclases,",");
                $var_tupla=$tupla['cclase'];
                $pos=strpos ($var_cclases,$var_tupla);


                //Si tiene justificante OR si lo requiere OR si es una sol. tramitada por la DGAA OR  es VAC o ADI con
                                                                                                                cambio.
                if(($justificante=='S' OR $pos>0 OR $tupla['requierejustificante']=='S' )
                                OR ( $tupla['cclase']=='VAC' OR $tupla['cclase']=='ADI' )
                                AND ( Solicitudes::f_esCambioDeSolicitud($tupla['nsolicitud']) )){
                                array_push($this->v_botonImprimir,1);
                }
                else{
                                array_push($this->v_botonImprimir,0);
                }
        }

      //No puede actualizar el campo justificante
        $this->actualizaCampos = 0;
        //Puede que este el campo imprimir activo
        $this->botonImprimir = 1;
        $this->botonEnDesacuerdo=0;
        //si mostramos la parte de personal
        $this->visualizarPartePersonal = 1;
    }
    elseif($visibilidad=='ConsultarSolicitudesACargoResponsable_NoPersonal'){
        //No hace nada
        $this->actualizaCampos = 0;
        //No puede Imprimir
        $this->botonImprimir = 0;
        $this->botonEnDesacuerdo=0;
        //no mostramos la parte de personal
        $this->visualizarPartePersonal = 0;
    }
    elseif($visibilidad=='FirmaResponsable'){
        //SegÃºn el perfil podrÃ¡ ver o no la parte de abajo
        if($this->parametrosBusqueda['entradaMenu']=='responsable')
                $this->visualizarPartePersonal = 0;
        else
                                $this->visualizarPartePersonal = 1;
        //No hace nada
        $this->actualizaCampos = 0;
        //No puede Imprimir
        $this->botonImprimir = 0;
        $this->botonEnDesacuerdo=0;
    }
    elseif($visibilidad=='ConsultaEstadoMisSolicitudes_noAutorizadaPorResponsable'){
        //No hace nada
        $this->actualizaCampos = 0;
        //No puede Imprimir
        $this->botonImprimir = 0;
        //Tiene el boton de en desacuerdo
        $this->botonEnDesacuerdo=1;
        //no mostramos la parte de personal
        $this->visualizarPartePersonal = 0;
    }
    elseif($visibilidad=='ConsultaEstadoMisSolicitudes_lip'){
        //Puede Imprimir
        $this->botonImprimir = 1;
        //Comprobamos si tenemos que activar el botÃ³n imprimir para cada una de las fichas.
        $this->v_botonImprimir = array(1);
        //No hace nada
        $this->actualizaCampos = 0;
        //Tiene el boton de en desacuerdo
        $this->botonEnDesacuerdo=0;
        //no mostramos la parte de personal
        $this->visualizarPartePersonal = 0;
    }
    elseif($visibilidad=='ConsultaEstadoMisSolicitudes_conJustificante'){
        //Comprobamos si tenemos que activar el botÃ³n imprimir para cada una de las fichas.
        $this->v_botonImprimir = array();
        foreach($m_datos as $tupla){
                $dclase = '';
                $justificante = '';
                ClasesDePermisos::f_dameDatosClasePermiso($tupla['cclase'],$dclase,$justificante);
                // 01/09/2009 INICIO cambio de  cclase not in ('LIP','LEF','LPP','EFP','AIN')
                $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');

                foreach($permisosDGAA as $index=>$tuplaDGAA){

                                $var_cclases=$var_cclases."'".$permisosDGAA[$index]['cclase']."',";

                }
                $var_cclases=rtrim($var_cclases,",");
```

```
                $var_tupla=$tupla['cclase'];
                $pos=strpos ($var_cclases,$var_tupla);

                if($justificante=='S' OR ($pos>0) OR $tupla['requierejustificante']=='S'){
                        array_push($this->v_botonImprimir,1);
                }
                else{
                        array_push($this->v_botonImprimir,0);
                }
            }

            $this->actualizaCampos = 0;
            //Puede Imprimir
            $this->botonImprimir = 1;
            //Tiene el boton de en desacuerdo
            $this->botonEnDesacuerdo=0;
            //no mostramos la parte de personal
            $this->visualizarPartePersonal = 0;
        }
        return 0;
}



function accionesParticulares($str_accion, $objDatos) {

        switch ($str_accion)
        {
            case 'imprimirSolicitud':
                IgepDebug::setDebug(DEBUG_USER,"entro");
                $objDatos->setOperacion('seleccionar');
                $m_datos = $objDatos->getAllTuplas();
                $nsolicitud = $objDatos->getValue('nsolicitud');
                if (is_null($nsolicitud)){//Estoy editando
                        $this->noSaltarParaImprimir=true;
                        $this->ShowMensaje("APL-92");
                        $actionForward = new ActionForward('gvHidraValidationError');
                        $actionForward->put('IGEPclaseManejadora','DetalleSolicitud');
                        return $actionForward;
                }
                $this->noSaltarParaImprimir=false;
                break;

            case 'enDesacuerdo':
                $objDatos->setOperacion('seleccionar');
                $nsolicitud = $objDatos->getValue('nsolicitud');
                $res = $this->operar('UPDATE TPER_SOLICITUDES
                                SET cestado=\'NI\'
                                WHERE nsolicitud = '.$nsolicitud);
                if($res==0)
                    $obj_mensaje = new IgepMensaje('APL-23');
                else
                    $obj_mensaje = new IgepMensaje('APL-22');
                //Metemos el mensaje en la clase
                IgepSession::guardaVariable('ConsultaEstadoMisSolicitudes','obj_mensaje',$obj_mensaje);
                $actionForward = $objDatos->getForward('correcto');
                return $actionForward;
            default:
                return -1;
        }

        //Creamos el objeto que manejará el informe
        $informeJ = new InformeJasper('solicitudPermisoYLicencia');

        //Especificamos la fuente de datos, en este caso, una BD relacional
        $informeJ->setDataSourceType('sgbd');
          //Incrustamos el pdf en el navegador
        $informeJ->setDisposition('inline');
        //Especificamos los parámetros relativos a la conexión con la BD relacional
        //Si coinciden con los del DSN de IGEP, podemos importarlos con:
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $informeJ->importPearDSN($g_dsn);


        //Fijamos el fichero jasper que nos hace de plantilla

          $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');
          foreach($permisosDGAA as $index=>$tuplaDGAA){

                    $var_cclases=$var_cclases."'".$permisosDGAA[$index]['cclase']."',";

          }
          $var_cclases=rtrim($var_cclases,",");
          $var_tupla=$m_datos[0]['cclase'];
          $pos=strpos ($var_cclases,$var_tupla);

        if ($pos > 0)
          $informeJ->setJasperFile('./plantillasJasper/listadoSolicitudPermisosYLicenciasDGAA.jasper');
        else
            $informeJ->setJasperFile('./plantillasJasper/listadoSolicitudPermisosYLicencias.jasper');

        $informeJ->addParam('nsolicitud', $nsolicitud, 'Integer');

        //Asignamos el informe como variable de clase para poder
        //acceder a él desde el fichero views donde realizaremos "la ejecución"
        $this->listadoSolicitudesPermisosYLicencias = $informeJ;
        $this->openWindow($objDatos->getForward('listado'));
        $this->showMensaje('APL-52',array($nsolicitud));
        $actionForward = $objDatos->getForward('correcto');
        return $actionForward;
}
```

```
function postModificar($objDatos){

        $m_datos = $objDatos->getAllTuplas();
        foreach($m_datos as $tupla){
            $res = $this->consultar("SELECT nsolicitud
                                FROM tper_permisos
                                WHERE nsolicitud = ".$tupla['nsolicitud']);
            if(count($res[0])>0){
                $this->operar("UPDATE tper_permisos
                                 SET justificante = '".$tupla['justificante']."'
                                                                                , observaciones = '".$tupla['observaciones']."'

                                 WHERE nsolicitud = ".$tupla['nsolicitud']);
            }
        }
        if($this->noSaltarParaImprimir){
          //Detectamos que se ha querido imprimir y por tanto guardamos pero NO volvemos al listado.
          $this->refreshSearch();
          return $objDatos->getForward('gvHidraSuccessNoJump');
        }
        return 0;
    }

}//Fin DetalleSolicitud
?>
```

Tpl – Consulta: DetalleSolicitud

```
{CWVentana tipoAviso=$smty_tipoAviso  codAviso=$smty_codError  descBreve=$smty_descBreve  textoAviso=$smty_textoAviso
                                                                                                    onLoad=$smty_jsOnLoad}

{CWBarra customTitle=$smty_customTitle usuario=$smty_usuario codigo=$smty_codigo}
        {CWMenuLayer name="$smty_nombre" cadenaMenu="$smty_cadenaMenu"}
{/CWBarra}
{CWMarcoPanel conPestanyas="true"}

<!-- ****************** PANEL edi ************************-->
{CWPanel id="edi" tipoComprobacion="envio" action="modificar" method="post" estado="on"
                                        claseManejadora="DetalleSolicitud" accion=$smty_accionActualizar}
{CWBarraSupPanel titulo="Detalle de solicitud de permisos y licencias"}
{/CWBarraSupPanel}
{CWContenedor}
        {CWFichaEdicion id="FichaEdicion" datos=$smty_datosFicha numPagInsertar="1"
                                                        accion=$smty_accionActualizar}
        {CWFicha}

        <table  cellpadding="5" width="95%">
        <tr>
                <td class="formularios" width="3%">
                                         
                </td>
                <td class="formularios" colspan="2">
                        {CWCampoTexto nombre="solicitante"  size="60" editable="false" textoAsociado="Interesado"}
                </td>
                <td class="formularios" >
                        {CWCampoTexto nombre="situacion"  size="10" editable="false" textoAsociado="Situación"}
                </td>
        </tr>
        <tr>
                <td class="formularios" width="3%">
                         
                </td>
                <td class="formularios">
                        {CWCampoTexto nombre="nsolicitud"  size="6" editable="false" textoAsociado="N° solicitud"}
                        {if $smty_v_mostrarSolModificada.$smty_iteracionActual eq 1}
                                  
                                {CWCampoTexto nombre="nsolmodificada"  size="6" editable="false"
                                                                        textoAsociado="Solicitud cambiada"}
                        {/if}
                </td>
                <td class="formularios">
                        {CWCampoTexto nombre="fsolicitud"  size="10" editable="false" textoAsociado="Fecha solicitud"}
                </td>
                <td class="formularios">
                        {CWCampoTexto nombre="periodo"  size="5" editable="false"
                                                                textoAsociado="Periodo"}
                </td>
        </tr>
        <tr>
                <td class="formularios" width="3%">
                         
                </td>
                <td class="formularios">
                        {CWCampoTexto nombre="dclase"  size="50" editable="false"  textoAsociado="Motivo"}
                        {CWCampoTexto nombre="cclase" oculto="true"}
                </td>
                <td class="formularios">
                        {CWCampoTexto nombre="fecha_ini"  size="10" editable="false" textoAsociado="Fecha inicio"
                                                                dataType=$dataType_DetalleSolicitud.fecha_ini}
                </td>
                <td class="formularios">
                        {CWCampoTexto nombre="fecha_fin"  size="10" editable="false" textoAsociado="Fecha fin"
                                                                dataType=$dataType_DetalleSolicitud.fecha_fin}
                </td>
        </tr>
        <tr>
                <td class="formularios" width="3%">     </td>
                        {if $smty_v_mostrarFechaNacimiento.$smty_iteracionActual eq 1}
                           <td class="formularios" colspan="3">
                                {CWCampoTexto nombre="fnacimientomenor"  size="10"
                                editable="false" textoAsociado="Fecha nacimiento"
                                dataType=$dataType_DetalleSolicitud.fnacimientomenor}
                           </td>
                        {/if}
                        {if $smty_v_mostrarDistancia.$smty_iteracionActual eq 1 and
                                $smty_v_mostrarGradoParentesco.$smty_iteracionActual eq 1}
                           <td class="formularios">
                                {CWLista nombre="distancia"  size="20" editable="false"
                                        textoAsociado="Distancia"
                                        dataType=$dataType_DetalleSolicitud.distancia}
                           </td>
                           <td class="formularios" colspan="2">
                                {CWLista nombre="gradoparentesco"  size="10" editable="false"
                                        textoAsociado="Grado parentesco"
                                        dataType=$dataType_DetalleSolicitud.gradoparentesco}

                           </td>
                        {elseif $smty_v_mostrarGradoParentesco.$smty_iteracionActual eq 1}
                          <td class="formularios" colspan="3">
```

```
                                {CWLista nombre="gradoparentesco"  size="10" editable="false"
                                        textoAsociado="Grado parentesco"
                                        dataType=$dataType_DetalleSolicitud.gradoparentesco}

                        </td>
                        {elseif $smty_v_mostrarDistancia.$smty_iteracionActual eq 1}
                        <td class="formularios" colspan="3">
                                {CWLista nombre="distancia"  size="20" editable="false"
                                        textoAsociado="Distancia"
                                        dataType=$dataType_DetalleSolicitud.distancia}


                        </td>
                        {/if}
</tr>
<tr>
        <td class="formularios" width="3%">
                 
        </td>
        <td class="formularios" colspan="2">
                {if $smty_actualizaCampos eq 1}
                        <b>Observaciones: </b>{CWAreaTexto nombre="observaciones" longitudMaxima="500"
                                                cols="50" rows="2" editable="true"}
                {else}
                        <b>Observaciones: </b>{CWAreaTexto nombre="observaciones" longitudMaxima="500"
                                                cols="50" rows="2" editable="false"}
                {/if}
        </td>
        <td class="formularios" align="center">
                {if $smty_botonImprimir eq 1}
                        {if $smty_v_botonImprimir.$smty_iteracionActual eq 1}

                                <fieldset style="  border-bottom-width: thin; border-color: silver">
                                <legend align='center' style="font-weight:bold;">Imprimir solicitud</legend>
                                {CWBotonTooltip imagen="06" titulo="Imprimir" funcion="modificar"
                                                                        actuaSobre="ficha"
                                                action="DetalleSolicitud__imprimirSolicitud"}
                                </fieldset>
                        {else}
                                         
                        {/if}
                {else}
                         
                {/if}
        </td>
</tr>
<tr>
        <td class="formularios" width="3%">
                 
        </td>
        <td class="formularios" colspan="3">
                <b>Avisos: </b>{CWAreaTexto nombre="avisos" editable="false" rows="2" cols="86"}
        </td>
</tr>
<tr>
        <td class="formularios" width="3%">
                 
        </td>
        <td class="formularios" colspan="2">
                {CWCampoTexto nombre="destado"  size="60" editable="false"
                                                        textoAsociado="Estado"}
        </td>
        <td class="formularios">
                {if $smty_v_activarRequiereJustificante.$smty_iteracionActual eq 1}
                        {CWCheckBox nombre="requierejustificante" editable="true" actualizaA="true"
                                                textoAsociado="Requiere justificante"
                                dataType=$dataType_DetalleSolicitud.requierejustificante}
                {else}
                         
                {/if}
        </td>
</tr>
<tr>
        <td class="formularios" width="3%">
                 
        </td>
                {if $smty_actualizaCampos eq 1}
                        {if $smty_v_activarJustificante.$smty_iteracionActual eq 1}
                                <td class="formularios">
                                        {CWCheckBox nombre="justificante_correcto" editable="false"
                                                                actualizaA="true" textoAsociado="Pendiente
                                                                justificante correcto"
                                        dataType=$dataType_DetalleSolicitud.justificante_correcto}
                                </td>
                                <td class="formularios" colspan="2">
                                        {CWCheckBox nombre="justificante" editable="true"
                                                                actualizaA="true" textoAsociado="Adjunta
                                        justificante" dataType=$dataType_DetalleSolicitud.justificante}
                                </td>
                        {else}
                                <td class="formularios">
                                        {CWCheckBox nombre="justificante_correcto" editable="false"
                                                                actualizaA="true" textoAsociado="Pendiente
                                        justificante correcto"
                                        dataType=$dataType_DetalleSolicitud.justificante_correcto}
                                </td>
                                <td class="formularios" colspan="2">
                                        {CWCheckBox nombre="justificante" editable="false"
                                                                textoAsociado="Adjunta justificante"
                                                dataType=$dataType_DetalleSolicitud.justificante}
                                </td>
                        {/if}
```

```
                    {else}
                              <td class="formularios">
                                        {CWCheckBox nombre="justificante_correcto" editable="false"
                                                  actualizaA="true" textoAsociado="Pendiente justificante correcto"
                                                  dataType=$dataType_DetalleSolicitud.justificante_correcto}
                              </td>
                              <td class="formularios" colspan="2">
                                        {CWCheckBox nombre="justificante" editable="false" textoAsociado="Adjunta
                                                  justificante" dataType=$dataType_DetalleSolicitud.justificante}
                              </td>
                    {/if}
          </td>
</tr>
{if $smty_visualizarPartePersonal eq 1}
{if $smty_v_mostrarJustificanteEnc.$smty_iteracionActual eq 1}
<tr>
          <td class="formularios" width="3%">
                     
          </td>
          <td class="formularios">
                    <b>Total días sin justificante: </b>{CWCampoTexto nombre="diasSinJustificanteENC"  size="4"
                                                                                                        editable="false"}
          </td>
          <td class="formularios" colspan="2">
                    <b>Total días con justificante: </b>{CWCampoTexto nombre="diasConJustificanteENC"  size="4"
                                                                                                        editable="false"}
          </td>
</tr>
{/if}
{/if}
</table>
{if $smty_visualizarPartePersonal eq 1}
<fieldset style="  border-bottom-width: thin; border-color: silver">
<legend align='left' style="font-weight:bold;">TrÃ¡mites de alta</legend>
<table  cellpadding="5" width="95%">
<tr>
          <td class="formularios" width="3%">
                     
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="ffirmaresp"  size="10" editable="false" textoAsociado="Fecha"}
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="firmaresp"  size="50" editable="false" textoAsociado="Responsable"}
          </td>
</tr>
<tr>
          <td class="formularios" width="3%">
                     
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="fvalidatramitador"  size="10" editable="false" textoAsociado="Fecha"}
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="validatramitador"  size="50" editable="false" textoAsociado="Validado
                                                                                                        por"}
          </td>
</tr>
<tr>
          <td class="formularios" width="3%">
                     
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="fvalidarevisor"  size="10" editable="false" textoAsociado="Fecha"}
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="validarevisor"  size="50" editable="false" textoAsociado="Revisado por"}
          </td>
</tr>
<tr>
          <td class="formularios" width="3%">
                     
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="ffirmaper"  size="10" editable="false" textoAsociado="Fecha"}
          </td>
          <td class="formularios">
                    {CWCampoTexto nombre="firmaper"  size="50" editable="false" textoAsociado="Resp. personal"}
          </td>
</tr>
<tr>
          <td class="formularios" width="3%">
                     
          </td>

          {if $smty_fresolucionnumresol.$smty_iteracionActual eq 1}
             {if $smty_opcionMenu eq DGA}
              <td class="formularios">
                {CWCampoTexto nombre="fresolucion"  size="13" editable="false" textoAsociado="Fecha resoluciÃ³n"}
              </td>
             {else}
              <td class="formularios">
                {CWCampoTexto nombre="fresolucion" size="13" editable="false" textoAsociado="Fecha resolución - N°"}
              </td>
             {/if}
          {else}
              <td class="formularios">
               {CWCampoTexto nombre="fresolucion"  size="10" editable="false" textoAsociado="Fecha resoluciÃ³n"}
              </td>
          {/if}
          <td class="formularios">
                    {CWCampoTexto nombre="generaresol"  size="50" editable="false" textoAsociado="Generada por"}
```

```
		</td>
</tr>
<tr>
		<td class="formularios" width="3%">
			 
		</td>
		<td class="formularios">
			{CWCheckBox nombre="firmaresolucion" editable="false" textoAsociado="Firma resolución"
				dataType=$dataType_DetalleSolicitud.firmaresolucion}
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="notificaresol"  size="50" editable="false" textoAsociado="Notificada por"}
		</td>
</tr>
</table>
</fieldset>
<fieldset style="  border-bottom-width: thin; border-color: silver">
<legend align='left' style="font-weight:bold;">TrÃ¡mites de anulaciÃ³n</legend>
<table  cellpadding="5" width="95%">
<tr>
		<td class="formularios" width="3%">
			 
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="fanularesp"  size="10" editable="false" textoAsociado="Fecha"}
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="anularesp"  size="50" editable="false" textoAsociado="Responsable"}
		</td>
</tr>
<tr>
		<td class="formularios" width="3%">
			 
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="fvalidaanulatramitador"  size="10" editable="false" textoAsociado="Fecha"}
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="validaanulatramitador"  size="50" editable="false" textoAsociado="Validado
				por"}
		</td>
</tr>
<tr>
		<td class="formularios" width="3%">
			 
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="fvalidaanularevisor"  size="10" editable="false" textoAsociado="Fecha"}
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="validaanularevisor"  size="50" editable="false" textoAsociado="Revisado
				por"}
		</td>
</tr>
<tr>
		<td class="formularios" width="3%">
			 
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="fanulaper"  size="10" editable="false" textoAsociado="Fecha"}
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="anulaper"  size="50" editable="false" textoAsociado="Resp. personal"}
		</td>
</tr>
<tr>
		<td class="formularios" width="3%">
			 
		</td>
		{if $smty_fresolucionnumresolanula.$smty_iteracionActual eq 1}
		<td class="formularios">
			{CWCampoTexto nombre="fresolucionanula"  size="10" editable="false" textoAsociado="Fecha
				resolución - N°"}
		</td>
		{else}
		<td class="formularios">
			{CWCampoTexto nombre="fresolucionanula"  size="10" editable="false" textoAsociado="Fecha
				resolución"}
		</td>
		{/if}
		<td class="formularios">
			{CWCampoTexto nombre="generaresolanula"  size="50" editable="false" textoAsociado="Generada
				por"}
		</td>
</tr>
<tr>
		<td class="formularios" width="3%">
			 
		</td>
		<td class="formularios">
			{CWCheckBox nombre="firmaresolucionanula" editable="false" textoAsociado="Firma resolución"
				dataType=$dataType_DetalleSolicitud.firmaresolucionanula}
		</td>
		<td class="formularios">
			{CWCampoTexto nombre="notificaresolanula"  size="50" editable="false" textoAsociado="Notificada
				por"}
		</td>
</tr>
</table>
</fieldset>
{if $smty_verMotivoNoAutorizacion eq 1}
<table  cellpadding="5" width="95%">
```

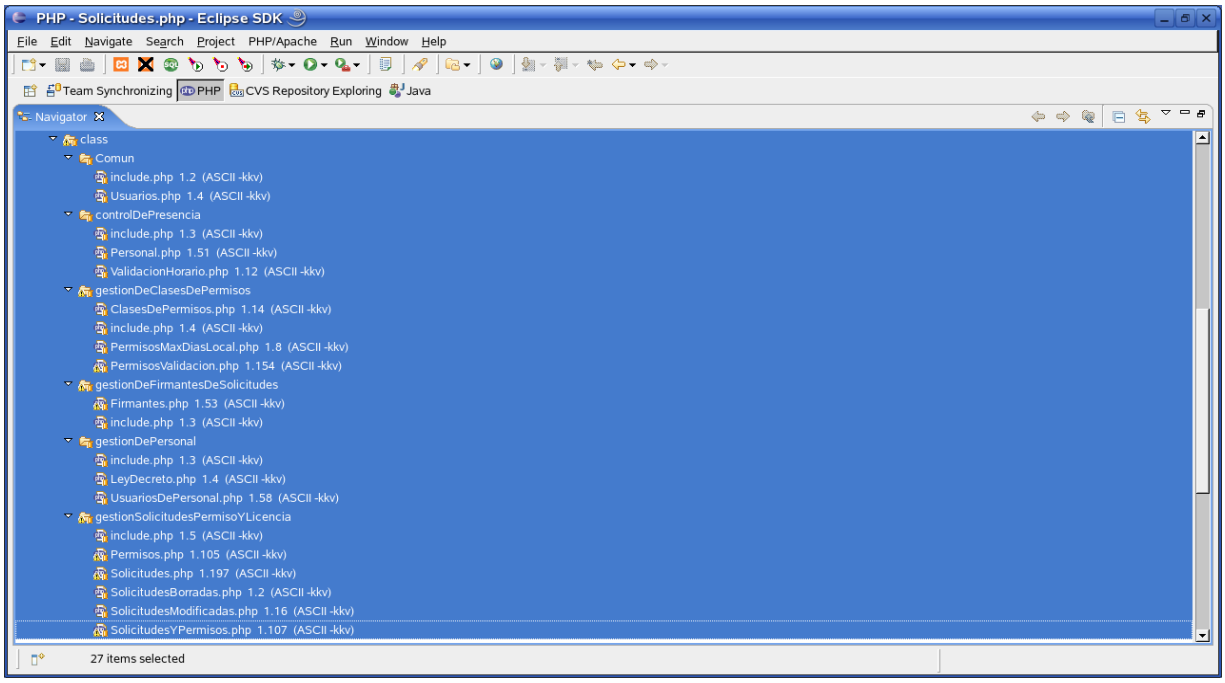
```
          <tr>
                    <td class="formularios" width="3%">
                               
                    </td>
                    <td class="formularios" colspan="3">
                              <b>Motivo de la no conformidad: </b>{CWAreaTexto nombre="motivonoautper" editable="false"
                                                                                                        rows="2" cols="80"}
                    </td>
          </tr>
          </table>
          {/if}
          {/if}
          {/CWFicha}
          {CWPaginador enlacesVisibles="3"}
       {/CWFichaEdicion}
     {/CWContenedor}
     {CWBarraInfPanel}
          {if $smty_botonEnDesacuerdo eq 1}
                    {CWBoton imagen="51" texto=" En desacuerdo" class="boton" accion="particular"
                                                                                  action="DetalleSolicitud__enDesacuerdo"}

          {/if}
          {if $smty_actualizaCampos eq 1}
                    {CWBoton imagen="41" texto="Guardar" class="boton" accion="guardar"}
                    {CWBoton imagen="42" texto="Cancelar" class="boton" accion="cancelar"
                                                                                  action="DetalleSolicitud__cancelarEdicion"}
          {else}
                    {CWBoton id="volver" imagen="48" texto=" Volver" class="boton" accion="volver"}
          {/if}
     {/CWBarraInfPanel}
     {/CWPanel}


          <!-- ****************** PESTAÑAS ************************-->
     {CWContenedorPestanyas}
          {CWPestanya tipo="edi" estado="on"}
     {/CWContenedorPestanyas}
{/CWMarcoPanel}
{/CWVentana}
```

Views – Consulta: DetalleSolicitud

```php
<?php

$comportamientoVentana= new IgepPantalla();

$panel = new IgepPanel('DetalleSolicitud',"smty_datosFicha");
$panel->activarModo("lis","estado_lis");
$comportamientoVentana->agregarPanel($panel);

//Visibilidad del botÃ³n imprimir
$botonImprimir = IgepSession::dameVariable('DetalleSolicitud','botonImprimir');
$s->assign('smty_botonImprimir',$botonImprimir);

$botonImprimir = IgepSession::dameVariable('DetalleSolicitud','v_botonImprimir');
$s->assign('smty_v_botonImprimir',$botonImprimir);


$actualizaCampos = IgepSession::dameVariable('DetalleSolicitud','actualizaCampos');
$s->assign('smty_actualizaCampos',$actualizaCampos);

$activarJustificante = IgepSession::dameVariable('DetalleSolicitud','v_activarJustificante');
$s->assign('smty_v_activarJustificante',$activarJustificante);

$activarRequiereJustificante = IgepSession::dameVariable('DetalleSolicitud','v_activarRequiereJustificante');
$s->assign('smty_v_activarRequiereJustificante',$activarRequiereJustificante);


$v_mostrarJustificanteEnc = IgepSession::dameVariable('DetalleSolicitud','v_mostrarJustificanteEnc');
$s->assign('smty_v_mostrarJustificanteEnc',$v_mostrarJustificanteEnc);

$v_mostrarSolicitudModificada = IgepSession::dameVariable('DetalleSolicitud','v_mostrarSolModificada');
$s->assign('smty_v_mostrarSolModificada',$v_mostrarSolicitudModificada);

//
$v_mostrarFechaNacimiento = IgepSession::dameVariable('DetalleSolicitud','v_mostrarFechaNacimiento');
$s->assign('smty_v_mostrarFechaNacimiento',$v_mostrarFechaNacimiento);

$v_mostrarDistancia = IgepSession::dameVariable('DetalleSolicitud','v_mostrarDistancia');
$s->assign('smty_v_mostrarDistancia',$v_mostrarDistancia);


$v_mostrarGradoParentesco = IgepSession::dameVariable('DetalleSolicitud','v_mostrarGradoParentesco');
$s->assign('smty_v_mostrarGradoParentesco',$v_mostrarGradoParentesco);


//

$botonEnDesacuerdo = IgepSession::dameVariable('DetalleSolicitud','botonEnDesacuerdo');
$s->assign('smty_botonEnDesacuerdo',$botonEnDesacuerdo);

$visualizarPartePersonal = IgepSession::dameVariable('DetalleSolicitud','visualizarPartePersonal');
$s->assign('smty_visualizarPartePersonal',$visualizarPartePersonal);

$fresolucionnumresol = IgepSession::dameVariable('DetalleSolicitud','fresolucionnumresol');
$s->assign('smty_fresolucionnumresol',$fresolucionnumresol);


$opcionMenu = IgepSession::dameVariable('ConsultaSolicitudesACargoResponsable','opcionMenu');
$s->assign('smty_opcionMenu',$opcionMenu);

$fresolucionnumresolanula = IgepSession::dameVariable('DetalleSolicitud','fresolucionnumresolanula');
$s->assign('smty_fresolucionnumresolanula',$fresolucionnumresolanula);

if($actualizaCampos==1)
    $s->assign('smty_accionActualizar','modificar');
else
    $s->assign('smty_accionActualizar','nada');

if(IgepSession::dameRol()=='P_USUARIO')
          $s->assign('smty_verMotivoNoAutorizacion',0);
else
          $s->assign('smty_verMotivoNoAutorizacion',1);

$s->display('consultas/p_detalleSolicitud.tpl');
?>
```

6.7. Clases

Las distintas clases manejadoras, se apoyan en llamadas a ciertas clases, las cuales se describen a contunuación:



- Clase Permisos:

```
class Permisos{

    public function Permisos(){
    }


     public function f_altaPermiso($nregpgv,$clase,$pPeriodo,$ppfechaIni,$ppfechaFin,$pObservaciones,$pCausa,$pTipoMinusvalia,


          $pPorcentajeReduccion,$pGenInci,$pFinReduc,$ppFnacimientomenor,$tipo,&$resultado,$pHorasReduccion){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_altaPermiso');
          if ($ppfechaIni!=null)
                    $pfechaIni = clone $ppfechaIni;
          if ($ppfechaFin!=null)
                    $pfechaFin = clone $ppfechaFin;
          if ($ppFnacimientomenor!=null)
                    $pFnacimientomenor = clone $ppFnacimientomenor;

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);

          // Validación del permiso
          switch ($clase){
                    case 'IT':
                              if(empty($pCausa)){
                                        $resultado ['tipo'] = 'P';
                                        $resultado ['mensaje'] = "La causa es obligatoria";
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
                                        return -1;
                              }
                    break;
                    case 'RJM':
                              if(empty($pTipoMinusvalia)){
                                        $resultado ['tipo'] = 'P';
                                        $resultado ['mensaje'] = "El tipo de minusvalía es obligatorio.";
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
                                        return -1;
                              }
                    break;
                    case 'RJP':
                    case 'RJG':
                    case 'RJC':
                    case 'RJV':
                              if(empty($pHorasReduccion)){
                                        $resultado ['tipo'] = 'P';
                                        $resultado ['mensaje'] = "El tiempo de reducción en horas es obligatorio.";
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
                                        return -1;
                              }

                              if(empty($pPorcentajeReduccion)){
                                        $resultado ['tipo'] = 'P';
                                        $resultado ['mensaje'] = "El porcentaje de reducción es obligatorio. Debería indicar
                                                  el tiempo de reducción, y abandonar el campo, para que el
                                                  cálculo del porcentaje de reducción tenga efecto.";
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
                                        return -1;
                              }
                    break;
          }

          // En el interfaz de usuario se podría confirmar o cancelar la solicitud actual.
          // Si se confirma, se pasa a dar de alta

          $ret= Permisos::f_validarPermiso($resultado, $var_dias_naturales, $var_dias_habiles, $var_ultimo,$pPeriodo,

                    $pfechaIni, $pfechaFin, $total_dias, $total_dias_labo, $pFnacimientomenor,$clase,$nregpgv);

          if ($ret<0){
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
                    return -1;
          }


          //Mostrar la información del historial de su solicitud.
          //Obtener la descripción del permiso que se acaba de solicitar
          $ret=ClasesDePermisos::f_dameDatosClasePermiso($clase, $var_desc_clase, $var_justificante,$clase,$nregpgv);
          if ($ret<0){
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
                    return -1;
          }
          //Mostrar la siguiente información al pulsar el botón "Confirmar"
          if(!empty($pfechaFin)){
                    // Obtenemos la información de la clase de permiso para saber si trata días hábiles o naturales
                    $minDias="";$maxDias="";$diasNaturales="";$unidad="";
                    PermisosValidacion::f_LeerMinMaxDias($clase,$minDias,$maxDias,$diasNaturales,$unidad);
                    if ($diasNaturales=='S'){
                              $mensaje=" naturales";
                              $var_dias=$var_dias_naturales;
                    }else{
                              $mensaje=" hÃ¡biles";
                              $var_dias=$var_dias_habiles;
                    }

                    if(!empty($var_ultimo)){
                              $mensajeAlta=$var_desc_clase." del ".$pfechaIni->formatUser()." al ".$pfechaFin->formatUser().":
                                        ".$var_dias." días ".$mensaje.", de ".$pPeriodo;
```

```
                        if ($diasNaturales=='S'){
                                $mensajeAlta=$mensajeAlta."<blockquote>".$var_desc_clase.", días ".$mensaje.
                                        "<blockquote>solicitados:".$total_dias."</blockquote></blockquote>";
                        }else{
                                $mensajeAlta=$mensajeAlta."<blockquote>".$var_desc_clase.", días ".$mensaje.
                                        "<blockquote>solicitados: ".$total_dias_labo."</blockquote></blockquote>";
                        }
                        $mensajeAlta=$mensajeAlta."Fecha de inicio de la Ãºltima solicitud de ".$var_desc_clase. ":
                                ".$var_ultimo->formatUser(). "<br>";
                }else {
                        $mensajeAlta=$var_desc_clase." del ".$pfechaIni->formatUser()." al ".$pfechaFin->formatUser().":
                                ". $var_dias. " días ".$mensaje.", de ".$pPeriodo. ".";
                }
        }else{
                if(!empty($var_ultimo)){
                        $mensajeAlta=$var_desc_clase." del ".$pfechaIni->formatUser().", de ".$pPeriodo. ".";
                        $mensajeAlta=$mensajeAlta."Fecha de inicio de la Última solicitud de
                                ".$var_desc_clase. ": ".$var_ultimo->formatUser(). "<br>";
                }
                else $mensajeAlta=$var_desc_clase." del ".$pfechaIni->formatUser().", de ".$pPeriodo. ".";
        }

        //Se da de alta el permiso

        $permiso=SolicitudesYPermisos::f_altaSoliPer($nregpgv,$clase,$pPeriodo,$pfechaIni,$pfechaFin,$pObservaciones,
                                                $pFnacimientomenor);
        //Tras dar de alta parte de los datos de la solicitud, pasaremos a actualizarla

        switch ($clase){
                case 'IT':
                        $permiso['causa']=$pCausa;
                        $datosValidarPermiso['generarIncidencia']=$pGenInci;
                        break;
                case 'RJM':
                        $permiso['tipo_minusvalia']=$pTipoMinusvalia;
                        $permiso['aviso_agenda']=$pFinReduc;
                        break;
                case 'RJG':
                        $permiso['porcentaje']=$pPorcentajeReduccion;
                        $permiso['horas_reduccion']=$pHorasReduccion;
                        $permiso['aviso_agenda']=$pFinReduc;
                        break;
                case 'RJE':
                        $permiso['aviso_agenda']=$pFinReduc;
                        break;
                case 'MAD':
                        $datosValidarPermiso['generarIncidencia']=$pGenInci;
                        break;
                case 'RJP':
                        $permiso['porcentaje']=$pPorcentajeReduccion;
                        $permiso['horas_reduccion']=$pHorasReduccion;
                        $permiso['aviso_agenda']=$pFinReduc;
                        $datosValidarPermiso['generarIncidencia']=$pGenInci;
                        break;
                case 'RJ5':
                        $permiso['porcentaje']=31;
                        $permiso['horas_reduccion']='2:15';
                        $permiso['aviso_agenda']=$pFinReduc;
                        $datosValidarPermiso['generarIncidencia']=$pGenInci;
                        break;
                case 'RJC':
                        $permiso['porcentaje']=$pPorcentajeReduccion;
                        $permiso['horas_reduccion']=$pHorasReduccion;
                        $permiso['aviso_agenda']=$pFinReduc;
                        break;
                case 'RJV':
                        $permiso['porcentaje']=$pPorcentajeReduccion;
                        $permiso['horas_reduccion']=$pHorasReduccion;
                        $permiso['aviso_agenda']=$pFinReduc;

                        break;

        }



        //Tomamos la provincia y municipio que tenga el usuario interesado en la vista VPER_OCUPACION
        $cproCmun = $con->consultar("SELECT cpro, cmun FROM vper_ocupacion WHERE nregpgv = '".$nregpgv."'");
        if ($cproCmun==-1 || count($cproCmun)==0) {
                IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
                return -1;
        }
        $permiso['cpro']=$cproCmun['0']['cpro'];
        $permiso['cmun']=$cproCmun['0']['cmun'];
        $datosValidarPermiso['mensajeAlta']=$mensajeAlta;
        $datosValidarPermiso['codJustificante']=$var_justificante;
        $datosValidarPermiso['permiso']=$permiso;
        IgepDebug::setDebug(DEBUG_USER,'FIN f_altaPermiso');
        return $datosValidarPermiso;
}


    function f_validarPermiso(&$resultado,&$pdias,&$pdiaslabo,&$pultimo,$periodo,$ppfechaIni,$ppfechaFin,

        &$ptotaldias,&$ptotaldiaslabo,$pfnacimientoMenor,$clase,$nregpgv){

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_validarPermiso');
        if ($ppfechaIni!=null)
                $pfechaIni = clone $ppfechaIni;
        if ($ppfechaFin!=null)
                $pfechaFin = clone $ppfechaFin;
```

```
        if ($pfnacimientoMenor!=null)
                $fnacimientoMenor = clone $pfnacimientoMenor;
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);


        $datosSolicitud= SolicitudesYPermisos::f_validarSoliPer($periodo,$pfechaIni,$pfechaFin,$clase,$resultado,
                        $diasTrabajados,$pdias, $pdiaslabo, $tipo='P', $pultimo,$pdiasadi,$ptotaldias,$ptotaldiaslabo,
                        $diasTrabajadosVV6,$nregpgv, $claseLlamada=$clase);
        if ($datosSolicitud<0){
                IgepDebug::setDebug(DEBUG_USER,'FIN f_validarPermiso');
                return -1;
        }
        //Obtener la descripción del permiso
        ClasesDePermisos::f_dameDatosClasePermiso($clase,$var_desc_clase, $var_justificante);

        //Comprobar que no existe ya otro permiso para la misma clase y con alguna fecha del intervalo que coincida
        // con el intervalo solicitado

        // Pasamos el código de cálculo de solapamiento a una función


                SolicitudesYPermisos::f_calcularSolapamientoSoliPer($resultado,$pfechaIni,$pfechaFin,$var_desc_clase,
                                                                $phaySolapamiento,'S',$clase,$nregpgv );

        if($phaySolapamiento){
                IgepDebug::setDebug(DEBUG_USER,'FIN f_validarPermiso');
                return -1;
        }

        //Sacamos nrp de la sesión
        //Añadir  a llamada f_leerValidacion el último parametro nregpgvQueSolicita, que es la persona que da de alta las
        solicitudes
        $datos = IgepSession::dameDatosUsuario();
        $nregpgvQueSolicita = $datos[nrp];

        $datosValidacion=PermisosValidacion::f_leerValidacion($clase,$pfechaIni,$pfechaFin,$periodo,$diasTrabajados,

                                $ptotaldiaslabo,$nulo=null,$ptotaldias, $nregpgv,$nulo=null,$nulo=null,$resultado,

                                $var_desc_clase,$var_justificante,$nulo=null,$nulo=null,$nulo=null,$nulo=null,

                                $nulo=null,$tipo='P', $fnacimientoMenor,$diasTrabajadosVV6,$pdias, $pdiaslabo,

                                $nulo=null,$nulo=null,$nulo=null,$nulo=null,$nulo=null,$nulo=null,$reqJustificante,

                                $aportaJustificante,$nulo=null,$nulo=null,$gradoParentesco,$nregpgvQueSolicita);


        if ($datosValidacion<0){
                $resultado['tipo']='P';
                $resultado['mensaje'];//= "Error al calcular los dÃas de permiso";
                IgepDebug::setDebug(DEBUG_USER,'FIN f_validarPermiso');
                return -1;
        }

        IgepDebug::setDebug(DEBUG_USER,'FIN f_validarPermiso');
        return 0;
        }//Fin f_validarPermiso



public function f_generarPermiso(&$conPostgres,&$conOracle,$pnregpgv,$pcpro,$pcmun,$pcclase,$ppfechaInicio,

$ppfechaFin,$pPeriodo,$pPersona,$justificante,$observaciones,$pnsolicitud,$ppfnacimientomenor,$pdiasTrabajados){

        IgepDebug::setDebug(DEBUG_USER,"INI f_generarPermiso");
        if ($ppfechaInicio!=null)
                $pfechaInicio = clone $ppfechaInicio;
        if ($ppfechaFin!=null)
                $pfechaFin = clone $ppfechaFin;
        if ($ppfnacimientomenor!=null)
                $pfnacimientomenor = clone $ppfnacimientomenor;


    ////////////////////////////////////////////////////////////////////
    //SE DEBERÃ A COMPROBAR LO DE AÑADIR A LA AGENDA//////////////////////
    ////////////////////////////////////////////////////////////////////

    //(Cod 53)
    ////////////////////////////////////////////////////////////////////////////////
    // Esta comprobación que añade a la agenda sólo se realizará tras la firma de personal autorizándola
    // SE DEBERÍA MIRAR EN f_altaPermiso, además de en f_generarPermiso. Aunque realmente no se tiene que
    // hacer en f_altaPermiso, porque realmente no hace falta en el alta, ya que solo se añade a la agenda
    // cuando se genera el permiso correspondiente de las solicitudes de VAC, LIP o ADI, que nunca se dan
    // de alta por f_altaPermiso, sino por f_generarPermiso
    ////////////////////////////////////////////////////////////////////////////////////////////////////
    //Solo se tienen en cuenta las LIP para generar la incidencia en la agenda
    if($pcclase=='LIP'){
        // ------------------------------------------------------------------------------------------------
        // Calculo del total días solicitados de LICENCIA SIN RETRIBUCIÓN (LIP)
        // ------------------------------------------------------------------------------------------------
        $nulo = null;
        $totalMesesNat = null;
        $mesesNat = null;
        $totalRestoDias = null;
        $restoDias = null;


        $ret=SolicitudesYPermisos::f_mesesNaturalesPermiso($pPeriodo,$pfechaInicio,$pfechaFin,$pcclase,$totalMesesNat,
```

```
                                $mesesNat,$totalRestoDias,$restoDias,'P','ALTA',$pnregpgv,$claseLlamada='LIP');


    //Comprobar si procede devolución proporcional de retribuciones para los permisos 'VAC' y 'LIP'
    if($totalMesesNat >=1){
       //Vacaciones y Licencia por interés particular de más de un mes
        //Se calcula la parte proporcional de días de vacaciones (y adicionales) que no puede disfrutar porque
        //corresponden a los permisos 'LIP'
        // ------------------------------------------------------------------------------------------------------
        // Cálculo de días mínimo y máximo a solicitar por permisos y solicitudes
        // Los días para VV6 y VAC se reducen proporcionalmente si no se ha trabajado todo el año
        // ------------------------------------------------------------------------------------------------------
        $nulo=null;


        // Cambiamos pcclase por 'VAC' para que maxdias tenga el valor máximo de VAC y no de LIP
        PermisosValidacion::f_CalcularMinMaxDias('VAC',$pPeriodo,$pcmun,$pcpro, $diasMinimo,$diasMaximo,
            $pdiasTrabajados,$diasNaturales,$maxdias,$var_total_dias,$var_desc_tipodias,$nulo,$nulo,$nulo,$nulo,
            $nulo, $nulo, $nulo, $pnregpgv, $noEsLaboral,$diasMaximoLocal);
        // ------------------------------------------------------------------------------------------------------
        // Total días laborables solicitados de VACACIONES (VAC)
        // ------------------------------------------------------------------------------------------------------
        $nulo=null;
        SolicitudesYPermisos::f_diasPermiso($pPeriodo, $pfechaInicio, $pfechaFin, $pcclase,$total_dias_vac,
                               $total_diaslabo_vac, $nulo, $nulo, 'P', $nulo, $nulo, $pnregpgv, 'VAC');



       // Aquí se hace el cálculo final que hará que aparezca o no el mensaje en la agenda

        if($total_diaslabo_vac > $maxdias - $maxdias/12 * ($totalMesesNat + $totalRestoDias/30)){
            $msg = '';
            //f_addAgenda
            //Formamos el mensaje que vamos a insertar en la agenda
            $mensaje = "Procede devolución proporcional de retribuciones. Solicitados:"
            . $total_diaslabo_vac . " días laborables de vacaciones, y ".$totalMesesNat." meses y ".$totalRestoDias.
            " días naturales no retribuídos. El nº de días laborables de vacaciones que pierde por los
            días solicitados no retribuídos es " . ($maxdias)/12 * ($totalMesesNat + $totalRestoDias/30) . " " .
            $msg;
            $msg = "A ".$pnregpgv.", ".$pPersona." ".$mensaje;
            $msg_ins = substr($msg,0,149);

            //conexiÃ³n a ORACLE
            $res = $conOracle->consultar("select count(numero) as \"numero\"
                                from tper_agenda
                                where texto='".$msg_ins."' and usuario_c='PER'");
            if($res[0]['numero']==0){
                $res = $conOracle->operar("insert into tper_agenda (numero, usuario_c, usuario_d, fecha, texto )
                                  values ( sper_agenda.NextVal, 'PER', null, sysdate, '".$msg_ins."' )");
                if($res==-1){
                      IgepDebug::setDebug(DEBUG_USER,"FIN ERROR f_generarPermiso");
                    return -1;
                }
                $resultado = $mensaje;
            }
        }
    }
}
// Grabamos en la tabla de personal TPER_PERMISOS la solicitud autorizada
// Se inserta después de añadir en la agenda los posibles mensajes porque si se hiciera antes,
// se sumarían dos veces los días de LIP que se están solicitando

$conPostgres->prepararOperacion($observaciones);
$conPostgres->prepararOperacion($justificante);


$pcclase_RESTAURAR=$pcclase;
$pnregpgv_RESTAURAR=$pnregpgv;

empty($pnregpgv)?$pnregpgv = "null":$pnregpgv="'".$pnregpgv."'";
empty($pcclase)?$pcclase = "null":$pcclase="'".$pcclase."'";
empty($pfechaInicio)?$str_pfechaInicio = "null":$str_pfechaInicio="'".$str_pfechaInicio=$conPostgres-
  >prepararFecha($pfechaInicio)."'";
empty($pfechaFin)?$str_pfechaFin = "null":$str_pfechaFin="'".$conPostgres->prepararFecha($pfechaFin)."'";
empty($observaciones)?$observaciones = "null":$observaciones="'".$observaciones."'";
empty($pPeriodo)?$pPeriodo = "null":$pPeriodo;
empty($pcpro)?$pcpro = "null":$pcpro="'".$pcpro."'";
empty($pcmun)?$pcmun = "null":$pcmun="'".$pcmun."'";
empty($justificante)?$justificante = "null":$justificante="'".$justificante."'";
empty($pnsolicitud)?$pnsolicitud = "null":$pnsolicitud;
empty($pfnacimientomenor)?$str_pfnacimientomenor = "null":$str_pfnacimientomenor="'".$conPostgres-

  >prepararFecha($pfnacimientomenor)."'";

$res = $conPostgres->operar("INSERT INTO tper_permisos
        (nregpgv,cclase,fecha_ini,fecha_fin,observaciones,periodo,cpro,cmun,justificante,nsolicitud,fnacimientomenor)
        VALUES
        ($pnregpgv,$pcclase,$str_pfechaInicio,$str_pfechaFin,$observaciones,$pPeriodo,$pcpro,$pcmun,$justificante,$pn
            solicitud,$str_pfnacimientomenor)");


  if($res!=-1) {
            // Generamos la incidencia para nóminas de algunos permisos

            $pcclase=$pcclase_RESTAURAR;
            $pnregpgv=$pnregpgv_RESTAURAR;

            if ( ($pcclase == 'ALU')  OR  ($pcclase=='LIP') OR ($pcclase=='LEF') OR ($pcclase=='AIN')  ) {
```

```
                        $generarIncidencia = Permisos::f_generarIncidencia($conPostgres,$conOracle,$pnregpgv, $pcclase,
                                                                           $pfechaInicio);

                        $res = $generarIncidencia['0'];
                        if($generarIncidencia['0']<0) {//Hay error

                                if(!empty($generarIncidencia['error'])) {     //Es un error, lanzamos la excepcion

                                        throw new Exception($generarIncidencia['error']);
                                        }
                                }
                        }

        }



        $result['0'] = $res;
        $result['errorAviso'] = $generarIncidencia['errorAviso'];
        $result['error'] = $generarIncidencia['error'];

    IgepDebug::setDebug(DEBUG_USER,"FIN f_generarPermiso");
    return $result;
}//Fin f_generarPermiso



 public function f_generarIncidencia(&$conPostgres,&$conOracle,$nregpgv,$cclase,$pfecha_ini){


        IgepDebug::setDebug(DEBUG_USER,"INI f_generarIncidencia");

        $fecha_ini = clone $pfecha_ini;

        //Este procedimiento será llamado cada vez que se quiere añadir una incidencia.
        //Se encarga de obtener los datos necesarios para la incidencia e insertarlos en tper_incidencias.
        //Comprobamos, antes de generar la incidencia, que el permiso existe

        $var_horasReduccion=0;

        $sel1="select CASE
                        WHEN causa = 'C' THEN 'Enfermedad común - Seguridad Social'
                        WHEN causa = '1' THEN 'Enfermedad común - Muface'
                        WHEN causa = 'P' THEN 'Accidente de trabajo/Enfermedad profesional - Seguridad Social'
                        WHEN causa = '2' THEN 'Accidente de trabajo/Enfermedad profesional - Muface'
                        WHEN causa = 'L' THEN 'Accidente no laboral - Seguridad Social'
                        WHEN causa = '3' THEN 'Accidente no laboral - Muface'
                        ELSE observaciones END AS observaciones, fecha_fin, horas_reduccion
        FROM tper_permisos
        WHERE nregpgv ='".$nregpgv."' and cclase ='".$cclase."' and  fecha_ini ='".$conPostgres-prepararFecha($fecha_ini) ";



        $res1=$conPostgres->consultar($sel1,array( 'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));

        ClasesDePermisos::f_dameDatosClasePermiso($cclase,$var_desc_clase, $var_justificante);

        if($res1<0){
                  $retorno['0']=-1;
                  $retorno ['error'] = "<ul><li>Error consultando el permiso ".$var_desc_clase." con fecha de inicio
                                       ".$pfecha_ini->format('d/m/Y').", antes de generar la incidencia para la aplicación
                                       de nóminas. Avise al Servicio de informÃ¡tica.<br/></li></ul>";
                  IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 1");
                  return $retorno;
        }
        if(count($res1)==0){ // Si no encuentra registros
                  $retorno['0']=-1;
                  $retorno ['errorAviso'] = "<ul><li>No se puede generar la incidencia para la aplicación de nóminas
                                             porque no existe el permiso ".$var_desc_clase." con fecha de inicio
                                      ".$pfecha_ini->format('d/m/Y').". Avise al Servicio de informática.<br/></li></ul>";
                  IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 2");
                  return $retorno;
        }


        $var_horasReduccion = $res1['0']['horas_reduccion'];


        if ($cclase=='RJ5'){
                  $var_porcentaje=25;
                  }

        else{
                  $var_porcentaje=UsuariosDePersonal::f_damePorcentajeReduccion($var_horasReduccion,$nregpgv,$phorasSemana);
        }



        if($fecha_ini!=null)
                        $var_fecha_baja = $fecha_ini;
        if($res1['0']['fecha_fin']!=null)
                        $var_fecha_alta = clone $res1['0']['fecha_fin'];
        $var_motivo = $res1['0']['observaciones'];


        //Asignamos códigos a las clases de permisos
        $varCodigo=null;$varEstado=null;
        Permisos::f_dameDatosIncidencia($cclase,$varCodigo,$varEstado);
```

```
// Si no existe incidencia, se inserta
$pestadoIncidencia=null;

$var_existeIncidencia = Permisos::f_existeIncidencia($varCodigo,$nregpgv,$fecha_ini,$pestadoIncidencia);


// si pestadoIncidencia es nulo // no encuentra registros . No existe la incidencia, entonces se inserta
if ($pestadoIncidencia==null){

          if($varCodigo == '4180')
                    $fecha_alta = "null";
          else
                    $fecha_alta = "'".$conOracle->prepararFecha($var_fecha_alta)."'";

          //Insertamos la incidencia
          $ins="INSERT INTO tper_incidencias (codigo,nregpgv,fecha_efecto,fecha_alta,fecha_baja,estado,
                    grupo_inicial,grupo_actual,jornada_inicial,jornada_actual,creljur,num_trienio,
                    grupo_trienio,ccomped_inicial,ccomped_final,ccompe4_inicial,ccompe4_final,importe,
                    descripcion,cplaza_inicial,cplaza_final,cpro,motivo,cclase,programa_inicial,programa_final)
                    VALUES ('".$varCodigo."','".$nregpgv."','".$conOracle->prepararFecha($fecha_ini)."',
                              ".$fecha_alta.",'".$conOracle->prepararFecha($var_fecha_baja)."'," .
                              "'$varEstado',null,null,null,'".$res1['0']['porcentaje_reduccion']."',null,null,null,
                              null,null,null,null,null,null,null,null,null,'".$res1['0']['observaciones']."',
                              '".$cclase."',null,null)";


          $ins=$conOracle->operar($ins);

          if($ins<0){
              $retorno['0']=-1;
              $retorno ['error'] = "Error insertando la incidencia ".$varCodigo." con N° de registro personal
                    '".$nregpgv."' y fecha ".$pfecha_ini->format('d/m/Y').". Avise al Servicio de informática.";
              IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 3");
              return $retorno;
          }
          else {
              $retorno['0']=-1;
              if (!is_null($var_fecha_alta))
                    $retorno ['errorAviso'] = "<ul><li>Ha sido registrada con éxito la incidencia, para la
                                        aplicación de nóminas, del permiso ".$var_desc_clase." entre
                                        el ".$pfecha_ini->format('d/m/Y'). " y el
                                        ".$var_fecha_alta->format('d/m/Y').".<br/></li></ul>";
              else
                    $retorno ['errorAviso'] = "<ul><li>Ha sido registrada con Ã©xito la incidencia, para la
                                        aplicación de nóminas, del permiso ".$var_desc_clase." con fecha
                                        de inicio ".$pfecha_ini->format('d/m/Y').".<br/></li></ul>";
                    IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 4");

          }

}

elseif(($varCodigo != '4180') or (($varCodigo = '4180') and (is_null($res1['0']['fecha_fin'])))) {

          if ($pestadoIncidencia[0]['ESTADO']<>$varEstado and $pestadoIncidencia[0]['ESTADO']!='R'){

                    $retorno['0']=-1;
                    $retorno ['errorAviso'] = "No se ha generado la incidencia ".$varCodigo." con N° de registro
                                        personal '".$nregpgv."' y fecha de inicio ".$pfecha_ini->format('d/m/Y').",
                                        porque ya se ha creado.";
                    IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 5");


                              return $retorno;
          }else {
          // Actualizamos la incidencia

                    $upp="UPDATE tper_incidencias SET fecha_alta='".$conOracle->prepararFecha($var_fecha_alta)."',
                              jornada_actual=".$var_porcentaje.",motivo='".$var_motivo."',cclase='".$cclase."'
                         WHERE codigo=".$varCodigo." and nregpgv='".$nregpgv."' and
                                                  fecha_baja='".$conOracle->prepararFecha($pfecha_ini)."'";

                    $upp=$conOracle->operar($upp);

          }


}

if($cclase=='IT' AND !is_null($res1['0']['fecha_fin'])){


          // Si no existe la incidencia , se inserta
          $var_asignar = Permisos::f_existeIncidencia('4181',$nregpgv,$fecha_ini,$pestadoIT);


          if ($var_asignar<0){
                    $retorno['0']=-1;
                    $retorno ['error'] = "Error consultando la incidencia '4181' con N° de registro personal
                                        '".$nregpgv."' y fecha ".$fecha_ini->format('d/m/Y').".
                                        Avise al Servicio de informática.";
                    IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 6");
                    return $retorno;
          }
          if($pestadoIT == null or $pestadoIT == ''){
          //No encuentra registros. No existe la incidencia, entonces se inserta
          //Insertamos la incidencia
          $ins="INSERT INTO tper_incidencias(codigo,nregpgv,fecha_efecto,fecha_alta,fecha_baja,estado,grupo_inicial,
                                                  "
                    grupo_actual,jornada_inicial,jornada_actual,creljur,num_trienio,grupo_trienio,
```

```
                                        ccomped_inicial,ccomped_final,ccompe4_inicial,ccompe4_final,importe,descripcion,

                                        cplaza_inicial,cplaza_final,cpro, motivo,cclase,programa_inicial,programa_final)
                                VALUES ('4181','".$nregpgv."','".$conOracle->prepararFecha($fecha_ini)."',
                                        '".$conOracle->prepararFecha($var_fecha_alta)."',
                                        '".$conOracle->prepararFecha($var_fecha_baja)."','$varEstado',null,null,null,null,
                                        null,null,null,null,null,null,null,null,null,null,null,null,null,
                                        ".$res1['0']['observaciones']."','".$cclase."',null,null)";
                                        // 28/04/2010 FIN [#5628]
                        $ins=$conOracle->operar($ins);
                        if($ins<0){
                                $retorno['0']=-1;
                                $retorno ['error'] = "Error insertando la incidencia '4181' con Nº de registro personal
                                        '".$nregpgv."' y fecha ".$pfecha_ini->format('d/m/Y').".
                                                Avise al Servicio de informática.";
                                IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 7");
                                return $retorno;
                        }else {
                                $retorno['0']=-1;
                                if (!is_null($var_fecha_alta))
                                        $retorno ['errorAviso'] = "<ul><li>Ha sido registrada con éxito la incidencia, para la
                                                        aplicación de nóminas, del permiso ".$var_desc_clase."
                                                        entre el ".$pfecha_ini->format('d/m/Y'). " y el
                                                        ".$var_fecha_alta->format('d/m/Y').".<br/></li></ul>";
                                else
                                        $retorno ['errorAviso'] = "<ul><li>Ha sido registrada con éxito la incidencia, para la
                                                        aplicación de nóminas, del permiso ".$var_desc_clase."
                                                        con fecha de inicio ".$pfecha_ini->format('d/m/Y')."
                                                        .<br/></li></ul>";
                                IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 8");


                        }
                                        // 28/04/2010 INICIO [#5628]
                                        // Si el estdo ha cambiado , se indica que la incidencia ya se ha creado
                }else {
                        if ($pestadoIT<>$varEstado){
                                $retorno['0']=-1;
                                $retorno ['errorAviso'] ="No se ha generado la incidencia '".$varCodigo."' con Nº de registro
                                                personal '".$nregpgv."' y fecha de inicio
                                                ".$pfecha_ini->format('d/m/Y').", porque ya se ha creado.";
                                IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 9");
                                return $retorno;
                        }
                        else {
                        // Actualizamos la Incidencia
                                $upp="UPDATE tper_incidencias SET
                                                fecha_alta='".$fecha_alta."',motivo='".$var_motivo."',cclase='".$cclase."'
                                                WHERE codigo = '4181' and nregpgv='".$nregpgv."' and
                                                                fecha_baja='".$fecha_ini."'";
                                $upp=$conOracle->operar($upp);

                        }
                }
                }
                $retorno['0']=0;
                $retorno ['error'] = "";
                IgepDebug::setDebug(DEBUG_USER,"FIN f_generarIncidencia 10");
                return $retorno;
                }//Fin f_generarIncidencia



        function f_existeIT($ppdia,$pnregpgv){

                IgepDebug::setDebug(DEBUG_USER,"INI f_existeIT");
                $pdia = clone $ppdia;

                $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
                $conexion = new IgepConexion($g_dsn, true);
                $pdia=$conexion->prepararFecha($pdia);
                $res = $conexion->consultar("SELECT fecha_ini,nregpgv
                                FROM tper_permisos
                                WHERE nregpgv = '".$pnregpgv."' and
                                    fecha_ini <= '".$pdia."' and
                                    (fecha_fin >= '".$pdia."' or fecha_fin is null) and
                                cclase = 'IT'");
                IgepDebug::setDebug(DEBUG_USER,"FIN f_existeIT");
                if(count($res[0]))
                    return true;
                else
                    return false;
        }


//  Me dice si el permiso tiene asociada un nº de solicitud o no.
        function f_existeSolicitud($pnregpgv, $cclase,$pfecha_ini){
                IgepDebug::setDebug(DEBUG_USER,"INI f_existeSolicitud");

                $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
                $conexion = new IgepConexion($g_dsn, true);
                $pfecha_ini=$conexion->prepararFecha($pfecha_ini);

                $res = $conexion->consultar("SELECT nsolicitud
                                FROM tper_permisos
                                WHERE nregpgv = '".$pnregpgv."' and
                                    cclase = '".$cclase."' and
                                    fecha_ini = '".$pfecha_ini."' ");

                IgepDebug::setDebug(DEBUG_USER,"FIN f_existeSolicitud");
                if(empty($res[0][nsolicitud])or($res[0][nsolicitud]==''))
```

```
            return false;
        else
            return true;

    }


    function f_consultarPermisos($pnregpgv,$pperiodo,&$pfini,&$pffin,&$pcclase,&$pnpermisos){

        IgepDebug::setDebug(DEBUG_USER,"INI f_consultarPermisos");
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);

        $var_query="SELECT p.observaciones,p.periodo,p.fecha_ini,p.fecha_fin,c.dclase
                    FROM tper_permisos p, tper_cod_permisos c
                    WHERE p.cclase = c.cclase and
                    p.NREGPGV = $pnregpgv ";
        // Condición a añadir de periodo a buscar

        if(!is_null($pperiodo)|| $pperiodo!='')
            $var_query.=" and p.periodo = $pperiodo ";
        // Condición a añadir de clase de permiso a buscar
        if (!is_null($pcclase)|| $pcclase!='')
            $var_query .= " and p.cclase = $pcclase ";

        // Condiciones a añadir de intervalo de fechas a buscar
        if((!is_null($pfini) ) AND (!is_null($pffin)))
            $var_query .=  " and (p.fecha_ini <= '".$conexion->prepararFecha($pffin)."' ) and (p.fecha_fin >=
                        '".$conexion->prepararFecha($pfini)."' or p.fecha_fin is null) ";
        elseif((!is_null($pfini)) AND (is_null($pffin)))
            $var_query .= " and (p.fecha_fin >= '".$conexion->prepararFecha($pfini)."' or p.fecha_fin is null) ";
            elseif( (is_null($pfini)) AND (!is_null($pffin)))
                $var_query .= " and p.fecha_ini <= '".$conexion->prepararFecha($pffin)."' ";

        // Por último se añade la expresión de ordenación
        $var_query .= " ORDER BY p.fecha_ini DESC, p.fecha_fin DESC;";


        // Se ejecuta la query var_observaciones,var_periodo,var_fechaini,var_fechafin,var_motivo
        $pnpermisos =
$conexion->consultar($var_query,array('DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
        IgepDebug::setDebug(DEBUG_USER,"FIN f_consultarPermisos");
        if(count($pnpermisos))
            return true;
         else
            return false;
    }



    function f_consultarDetallePermiso($pnregpgv,$pcclase,$ppfecha_ini,&$psolicitante,&$pmotivo,&$pperiodo,&$pfechafin,

                                       &$pcausa,&$ptipo_minusvalia, &$pporcentaje, &$pjustificante, &$pobservaciones,

                                       &$pdias, &$pdiaslabo, &$pultimo, &$ffirmaresolucion){

        IgepDebug::setDebug(DEBUG_USER,"INI f_consultarDetallePermiso");

        $pfecha_ini = clone $ppfecha_ini;

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);


        $query="SELECT   u.dnombre || ', ' || u.dapell1 || ' ' || u.dapell2 as solicitante,c.dclase, p.periodo,p.fecha_fin,
                        p.causa, p.tipo_minusvalia,p.porcentaje_reduccion,p.justificante,p.observaciones,s.ffirmaresolucion
                FROM tper_permisos p left outer join tper_solicitudes s on p.nsolicitud = s.nsolicitud,
                     tper_cod_permisos c, vper_ocupacion_historico u
                WHERE p.CCLASE = c.CCLASE and
                                       u.nregpgv = p.nregpgv and
                                            p.NREGPGV = $pnregpgv and
                                       p.cclase = $pcclase and
                                       p.fecha_ini = '".$conexion->prepararFecha($pfecha_ini)."'
                                       ORDER BY p.fecha_ini DESC, p.fecha_fin DESC";
       $res=$conexion->consultar($query,array('DATATYPES'=>array('fecha_fin'=>TIPO_FECHA,'ffirmaresolucion'=>TIPO_FECHA,)));
        $res=$res['0'];
        $pperiodo=$res['periodo'];
        $psolicitante=$res['solicitante'];
        $pmotivo=$res['dclase'];
        $pperiodo=$res['periodo'];
        $pfechafin=clone $res['fecha_fin'];
        $pcausa=$res['causa'];
        $ptipo_minusvalia=$res['tipo_minusvalia'];
        $pporcentaje=$res['porcentaje_reduccion'];

        //Obtención de los días naturales, hábiles y el primer día del último periodo solicitado previamente
        SolicitudesYPermisos::f_diasPermiso($pperiodo,$pfechaini , $pfechafin, $pcclase,$ptotaldias, $ptotaldiaslabo,
                                            $pdias, $pdiaslabo, 'P', $pultimo, $ptotaldiasadi, $pnregpgv, $pcclase);
        IgepDebug::setDebug(DEBUG_USER,"FIN f_consultarDetallePermiso");
        if(count($res))
            return true;
         else
            return false;

    }

    function f_obtieneFechaFin($pfechaNacimientoMenor){
```

```
		$pfecha=clone $pfechaNacimientoMenor;
		$pfecha->addYears(13);
		$pfecha->subDays(1);


		return $pfecha;
	}


	function f_dameDatosIncidencia($pcclase,&$pcodigo,&$pestado){

		switch($pcclase) {
			case 'IT':
				$pcodigo = '4180';
				$pestado='I';
				break;
			case 'MAD':
				$pcodigo = '9971';
				$pestado='P';
				break;
			case 'RJ5':
			case 'RJV':
			case 'RJP':
				$pcodigo = '9974';
				$pestado='P';
				break;
			case 'ALU':
				$pcodigo = '9972';
				$pestado='P';
				break;
			case 'LIP':
				$pcodigo = '9973';
				$pestado='P';
				break;
			case 'LEF':
				$pcodigo = '9975';
				$pestado='P';
				break;
			case 'AIN':
				$pcodigo = '9976';
				$pestado='P';
				break;
		}

		return true ;
	}


	function f_existeIncidencia($pcodigo,$pnregpgv,$pfechaini,&$pestado){

		// conexion a oracle
		$g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
		$conOracle = new IgepConexion($g_oracle_dsn);
		$cosulta_estado=" SELECT estado FROM tper_incidencias WHERE codigo= ".$pcodigo." AND nregpgv = '".$pnregpgv."' AND
				fecha_efecto =  '".$conOracle->prepararFecha($pfechaini)."' ";
		$pestado=$conOracle->consultar($cosulta_estado);



	return true;
	}


	public function f_borrarPermiso($conPostGress,$conOracle,$pnregpgv,$pcclase,$pfechaini,&$avisoNominas){

		IgepDebug::setDebug(DEBUG_USER,'INICIO f_borrarPermiso');

		if(is_object($pfechaini))
			$pfechaini = clone $pfechaini;

		// Si el permiso es alguno de los que se genera incidencia para nóminas y el estado de la incidencia No ha cambiado,

		// se borra la incidencia y el permiso.

		$var_asignar = permisos::f_dameDatosIncidencia($pcclase,$codigo,$estado);
		$avisoNominas=false;

		if (!is_null($codigo)){
			// Si existe la incidencia y no se ha modificado el estado , la borramos
			$var_existe = permisos::f_existeIncidencia($codigo,$pnregpgv,$pfechaini,$pestadoIncidencia);

			if ($pestadoIncidencia[0]['ESTADO']==$estado OR $pestadoIncidencia[0]['ESTADO']=='R' ){
				// Borramos la incidencia
				$borra_incidencia="DELETE FROM tper_incidencias WHERE codigo='".$codigo."' and
					nregpgv='".$pnregpgv."' and fecha_baja='".$conOracle->prepararFecha($pfechaini)."'";
				$borra_incidencia=$conOracle->operar($borra_incidencia);

				if ($borra_incidencia ==-1) {
					IgepDebug::setDebug(DEBUG_USER,'FIN f_borrarPermiso');
					return -1;
				}
				if ($codigo=='4180'){
					// Borramos la incidencia de alta de la IT
					$borra_incidencia="DELETE FROM tper_incidencias WHERE codigo='4181' and
					nregpgv='".$pnregpgv."' and fecha_baja='".$conOracle->prepararFecha($pfechaini)."'";
					$borra_incidencia=$conOracle->operar($borra_incidencia);
					if ($borra_incidencia ==-1){
						IgepDebug::setDebug(DEBUG_USER,'FIN f_borrarPermiso');
						return -1;
```

```
                                        }
                                }
                        }
                        else{
                                $avisoNominas=true;
                        }
                }

                // Borramos de la tabla de personal TPER_PERMISOS el permiso cuya solicitud de anulacion ha sido autorizada o
                // bien el permiso que no se ha disfrutado
                $borra_incidencia="DELETE FROM tper_permisos WHERE nregpgv='".$pnregpgv."' and cclase='".$pcclase."' and
                                                fecha_ini='".$conPostGress->prepararFecha($pfechaini)."'";
                $borra_incidencia=$conPostGress->operar($borra_incidencia);
                IgepDebug::setDebug(DEBUG_USER,'FIN f_borrarPermiso');
                if ($borra_incidencia ==-1) return -1;
                return 0;

        }

}//Fin clase Permisos
?>
```

- Clase Solicitudes:

```
class Solicitudes{

    public function Solicitudes(){
    }

public function f_altaSolicitud($nregpgv,$pPeriodo,$ppfecha_ini,$ppfecha_fin,$pObservaciones,$ppFnacimientomenor,

          $pAltamodificada,$pEstado,$pffirmaresp,$FirmadoPorALR,$pffirmaper,$FirmadaPorALP,$psituacion,&$total_dias,
          &$dias_labo_vac,&$total_dias_labo,&$var_dias_naturales,&$var_dias_habiles,&$resultado,$clase,$pdistancia,
          &$listaSolicitudesAnulacion, &$listaSolicitudesAnulacionResueltas,$pgradoParentesco, &$aportaJustificante ){


          IgepDebug::setDebug(DEBUG_USER,'INICIO f_altaSolicitud');
          $pfecha_ini = clone $ppfecha_ini;
          if ($ppfecha_fin!=null)
                    $pfecha_fin = clone $ppfecha_fin;
          if ($ppFnacimientomenor!=null)
                    $pFnacimientomenor = clone $ppFnacimientomenor;
          if ($pffirmaresp!=null)
                    $ffirmaresp = clone $pffirmaresp;
          if ($pffirmaper!=null)
                    $ffirmaper = clone $pffirmaper;



          if($clase=='EGP' or $clase=='DEF'){
                    $gradoParentesco=$pgradoParentesco;
                    $distancia=$pdistancia;
                    if(is_null($gradoParentesco)||empty($gradoParentesco)||$gradoParentesco==''){
                              $resultado['tipo']='P';
                              $resultado['mensaje']="El grado de parentesco es obligatorio.";
                              IgepDebug::setDebug(DEBUG_USER,'FIN error 1(gradoParentesco obligatorio) f_altaSolicitud');
                              return -1;
                    }

                    if(is_null($distancia)||empty($distancia)||$distancia==''){
                              $resultado['tipo']='P';
                              $resultado['mensaje']="La distancia es obligatoria.";
                              IgepDebug::setDebug(DEBUG_USER,'FIN error 1(distancia obligatoria) f_altaSolicitud');
                              return -1;
                    }
          }

          if($clase=='MFA' or $clase=='MUH' or $clase=='TDO'){
                    $distancia=$pdistancia;
                    if(is_null($distancia)||empty($distancia)||$distancia==''){
                              $resultado['tipo']='P';
                              $resultado['mensaje']="La distancia es obligatoria.";
                              IgepDebug::setDebug(DEBUG_USER,'FIN error 1(distancia obligatoria) f_altaSolicitud');
                              return -1;
                    }
          }


          $validarSolicitud= Solicitudes::f_validarSolicitud($resultado ,$var_vacyadicompletos  ,$var_dias_naturales,

                                        $var_dias_habiles ,$dias_adi ,$dias_labo_adi ,$dias_vac,

                                        $dias_labo_vac ,$var_ultimo  ,$var_totaldiasadi,

                                        $total_dias, $total_dias_labo,$pFnacimientomenor,

                                        'S',$clase,$nregpgv,$pPeriodo,$pfecha_ini,$pfecha_fin,$anular,

                                        $pObservaciones, $pdistancia, $resultadoUsuario,

                                        $listaSolicitudesAnulacion, $listaSolicitudesAnulacionResueltas,

                                        $totalDiasAnula,$totalDiasLaboAnula,$reqJustificante,$aportaJustificante,

                                        $pdistancia,$pgradoParentesco);


          if ($validarSolicitud<0){//Miramos en la llamada el valor de las variables $resultado*
                    IgepDebug::setDebug(DEBUG_USER,'FIN error 2(f_validarSolicitud) f_altaSolicitud');
                    return -1;
          }

          $datosValidarSolicitud['aportaJustificante'] = $aportaJustificante;

          //Construimos mensaje de la pantalla de confirmar

          //Si el estado de la solicitud es 'PC' (Pendiente de completar), también se indicará¡
          //que queda pendiente pedir días de vacaciones y/o adicionales para que la solicitud pase a firma de responsable
          //Esto se hace para que el usuario confirme su solicitud antes de que se grabe en la bb.dd.
          //Obtener la descripción del permiso

          ClasesDePermisos::f_dameDatosClasePermiso($clase,$var_desc_clase,$var_justificante);

          //Si se están pidiendo vacaciones, el total de días solicitados correcto estÃ¡ en dias_labo_vac, ya que la variable
          // total_dias_labo puede tener p. ej. 23 días para un mes natural, cuando en realidad son 22
          if(is_object($var_ultimo))
                    $var_ultimo= $var_ultimo->format('d/m/Y');
```

```
if(empty($pfecha_fin)){
        if(!empty($var_ultimo)){
                $mensajeAlta=$var_desc_clase." del ".$pfecha_ini.", de ".$pPeriodo.
                        "<blockquote>Fecha de inicio de la última solicitud de " .$var_desc_clase.":
                        ".$var_ultimo."</blockquote>";
        }else{
                $mensajeAlta=$var_desc_clase." del ".$pfecha_ini.", de ".$pPeriodo. ".";
        }
}else{

        // Obtenemos la información de la clase de permiso para saber si trata días hábiles o naturales
        $minDias="";$maxDias="";$diasNaturales="";$unidad="";
        PermisosValidacion::f_LeerMinMaxDias($clase,$minDias,$maxDias,$diasNaturales,$unidad);
        if ($diasNaturales=='S'){
                $mensaje=" naturales";
                $var_dias=$var_dias_naturales;
        }else{
                $mensaje=" hábiles";
                $var_dias=$var_dias_habiles;
        }

        if(!empty($var_ultimo)){
                $mensajeAlta=$var_desc_clase." del ".$pfecha_ini->formatUser()." al ".$pfecha_fin-
                >formatUser().": ".$var_dias." días ".$mensaje.", de ".$pPeriodo;
                if ($diasNaturales=='S'){
                        $mensajeAlta=$mensajeAlta."<blockquote>".$var_desc_clase.", días ".$mensaje.
                        "<block quote>solicitados: ".($total_dias -$totalDiasAnula)."</blockquote>";
                }else{
                        if ($clase=='VAC'){
                        $mensajeAlta=$mensajeAlta."<blockquote>".$var_desc_clase.", días ".$mensaje.
                        "<blockquote>solicitados: ".($dias_labo_vac -$totalDiasLaboAnula)."</blockquote>";
                        }
                        else { // en este caso si que es correcto el valor de total_dias_labo
                        $mensajeAlta=$mensajeAlta."<blockquote>".$var_desc_clase.", días ".$mensaje.
                        "<blockquote>solicitados: ".($total_dias_labo -$totalDiasLaboAnula)."</blockquote>";
                        }
                }
        }else {
                $mensajeAlta=$var_desc_clase." del ".$pfecha_ini->formatUser()." al ".$pfecha_fin-
                >formatUser().": ". $var_dias. " días ".$mensaje.", de ".$pPeriodo. ".";
        }

        if($clase=='ADI' OR $clase=='VAC' OR $clase=='AAP' OR $clase=='VV6'){

                if ($pPeriodo==(date ("Y"))){
                        $accion='P';
                }
                else if ($pPeriodo==(date ("Y"))-1){
                                $accion='A';
                }

                //Obtenemos los totales de VAC, ADI, VV6 y AAP que le corresponden, y el aÃ±o para el que se
                conocen los ADI
                PermisosValidacion::f_calcularTotalDiasVACyADIyVV6yAAP($nregpgv, $pPeriodo,
                        $totaldiasVAC, $totaldiasADI, $anyodiasADI, $totaldiasVV6, $totaldiasAAP,$accion);
                if ($totaldiasADI=='?'){
                        $totaldiasADI=0;
                }

        }

        //Si a la persona le corresponden días ADI y está pidiendo VAC, mostrar también los días ADI solicitados
        if($clase == 'VAC'){
                $mensajeAlta=$mensajeAlta."<blockquote>disponibles año:
                                        ".$totaldiasVAC."</blockquote></blockquote>";
                if($totaldiasADI > 0){

                        // Además le restamos el total de días de anulacion solicitados para la clase de
                        permiso ADI
                        SolicitudesYPermisos::f_calcularDiasPermisoAnyoAnulacion($nregpgv, $pPeriodo, 'ADI',
                        $totalLaboAnula, $totalAnula, $nsolicitudesAnula, $fechasAnulacion,
                        $nsolicitudesAnulaResueltas);

                        $mensajeAlta=$mensajeAlta."<blockquote>Adicionales de vacaciones, dÃas ".$mensaje;
                        if ($diasNaturales=='S'){
                                $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($dias_adi -
                                                                $totalAnula)."</blockquote>";
                        }else {
                                $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($dias_labo_adi -
                                                                $totalLaboAnula)."</blockquote>";
                        }
                        $mensajeAlta=$mensajeAlta."<blockquote> disponibles aÃ±o ".$anyodiasADI.":
                                                        ".$totaldiasADI."</blockquote></blockquote>";
                }
        }

        //Si a la persona le corresponden días ADI y estÃ¡ pidiendo días ADI, mostrar también los días de VAC
        solicitados
        if($clase == 'ADI'){
                $mensajeAlta=$mensajeAlta."<blockquote>disponibles aÃ±o ".$anyodiasADI.":
                                                        ".$totaldiasADI."</blockquote></blockquote>";
                if($totaldiasADI > 0){

                        // Además le restamos el total de días de anulación solicitados para la clase de
                        permiso VAC
                        SolicitudesYPermisos::f_calcularDiasPermisoAnyoAnulacion($nregpgv, $pPeriodo, 'VAC',
                                                $totalLaboAnula, $totalAnula, $nsolicitudesAnula,
                                                $fechasAnulacion, $nsolicitudesAnulaResueltas);
```

```
                                $mensajeAlta=$mensajeAlta."<blockquote>Vacaciones, dÃas ".$mensaje;
                                if ($diasNaturales=='S'){
                                        $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($dias_vac -
                                                                                $totalAnula)."</blockquote>";
                                }else {
                                        $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($dias_labo_vac -
                                                                                $totalLaboAnula)."</blockquote>";
                                }
                                $mensajeAlta=$mensajeAlta."<blockquote> disponibles año:
                                                                ".$totaldiasVAC."</blockquote></blockquote>";
                        }
                }

                SolicitudesYPermisos::f_diasPermiso($pPeriodo, $pfecha_ini, $pfecha_fin, 'AAP', $diasAap, $diasLaboAap,
                                                $nulo1, $nulo2, 'S', $nulo3, $nulo4,$nregpgv,$clase);


                //Si a la persona le corresponden días AAP y está¡ pidiendo VV6, mostrar también los días AAP solicitados
                if($clase == 'VV6'){
                         $mensajeAlta=$mensajeAlta."<blockquote>disponibles año:
                                                        ".$totaldiasVV6."</blockquote></blockquote>";
                         if($totaldiasAAP > 0){

                                // Además le restamos el total de días de anulación solicitados para la clase de
                                permiso AAP
                                SolicitudesYPermisos::f_calcularDiasPermisoAnyoAnulacion($nregpgv, $pPeriodo, 'AAP',
                                                        $totalLaboAnula, $totalAnula, $nsolicitudesAnula,
                                                        $fechasAnulacion, $nsolicitudesAnulaResueltas);

                                $mensajeAlta=$mensajeAlta."<blockquote>Adicionales de asuntos particulares, días
                                                                                                        ".$mensaje;
                                if ($diasNaturales=='S'){
                                        $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($diasAap -
                                                                                $totalAnula)."</blockquote>";
                                }else {
                                        $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($diasLaboAap -
                                                                                $totalLaboAnula)."</blockquote>";
                                }
                        $mensajeAlta=$mensajeAlta."<blockquote> disponibles año:
                                                        ".$totaldiasAAP."</blockquote></blockquote>";
                        }
                }

                SolicitudesYPermisos::f_diasPermiso($pPeriodo, $pfecha_ini, $pfecha_fin, 'VV6', $diasVv6, $diasLaboVv6,
                                                $nul1, $nul2, 'S', $nul3, $nul4,$nregpgv,$clase);

                //Si a la persona le corresponden días AAP y está¡ pidiendo días AAP, mostrar también los días VV6
                solicitados
                if($clase == 'AAP'){
                        $mensajeAlta=$mensajeAlta."<blockquote>disponibles año:
                                                        ".$totaldiasAAP."</blockquote></blockquote>";
                        if($totaldiasAAP > 0){

                                // Además le restamos el total de días de anulación solicitados para la clase de
                                permiso VV6
                                SolicitudesYPermisos::f_calcularDiasPermisoAnyoAnulacion($nregpgv, $pPeriodo, 'VV6',
                                                        $totalLaboAnula, $totalAnula, $nsolicitudesAnula,
                                                        $fechasAnulacion, $nsolicitudesAnulaResueltas);

                                $mensajeAlta=$mensajeAlta."<blockquote>Asuntos propios, días ".$mensaje;
                                if ($diasNaturales=='S'){
                                        $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($diasVv6 -
                                                                                $totalAnula)."</blockquote>";
                                }else {
                                        $mensajeAlta=$mensajeAlta."<blockquote>solicitados: ".($diasLaboVv6 -
                                                                                $totalLaboAnula)."</blockquote>";
                                }
                                $mensajeAlta=$mensajeAlta."<blockquote> disponibles año:
                                                        ".$totaldiasVV6."</blockquote></blockquote>";
                        }
                }

                if(!empty($var_ultimo)){
                        $mensajeAlta=$mensajeAlta."</blockquote>Fecha de inicio de la Ãºltima solicitud de
                                                        ".$var_desc_clase. ": ".$var_ultimo. "<br>";
                }

                if ($clase == 'VAC'){
                        $mensajeAlta=$mensajeAlta."<br><br>En caso de que quiera cambiar estas vacaciones, deberá¡
                                seguir estos pasos:"
                                ."<ul><li> Si la solicitud no está firmada por su responsable, bórrela y vuelva a
                                solicitar sus vacaciones con las nuevas fechas.</li>"
                                ."<li>Si la solicitud ya está firmada por su responsable, anúlela y, a continuación,
                                realice su solicitud con las nuevas fechas, indicando la justificación del
                                cambio en 'Observaciones'.</li> "."</ul>";

                }
        }

//Este mensaje se muestra en el views mirando siempre el campo mensajeHTML
$perfil= IgepSession::dameRol();
if( $perfil=='P_TRAMITA' || $perfil== 'P_REVISOR' || $perfil== 'P_RESPPER'){
// Mostrar en mensajes de aviso el valor de la variable resultado, que se ha calculado en f_leerValidacion()
        $resultado['mensajeHTML'];
}else{
// Mostrar en mensajes de aviso el valor de la variable resultadousuario, que se ha calculado en f_leerValidacion()
        $resultado['mensajeHTML']=$resultadoUsuario['mensajeHTML'];
}


//-----------------------------------------------------------------------------------------------------------
                // Se prepara la información para el alta de la solicitud
```

```
		//-----------------------------------------------------------------------------------------------------------------


		$solicitud['fsolicitud']=new gvHidraTimestamp();
		$solicitud['fsolicitud']->setTime(0,0,0);

		//  Ponemos a 'S' el campo requierejustificante si la variable listaSolicitudesAnulacion no está vacía

		//  Ponemos a 'S' el campo requierejustificante si el parámetro reqJustificante vale 'S' y el permiso es 'ENC'
		if (($clase == 'ENC' and $reqJustificante=='S') or ($listaSolicitudesAnulacion[0] != '' and ($clase == 'VAC' or
                                                                                $clase == 'ADI')))
			$solicitud['requierejustificante'] = 'S';
		else
			$solicitud['requierejustificante'] = 'N';


		$solicitudAux=SolicitudesYPermisos::f_altaSoliPer($nregpgv,$clase,$pPeriodo,$pfecha_ini,$pfecha_fin,$pObservaciones,

                                $pFnacimientomenor);


		$solicitud=array_merge($solicitud, $solicitudAux);


		// también se añade para los días adicionales de asuntos propios AAP
		if ($clase == 'VV6' OR $clase=='AAP')

			$solicitud['observaciones'] = SolicitudesYPermisos::f_recalcularOrdinalAlta($nregpgv, $pPeriodo,
                                                                $clase,$pfecha_ini,$pfecha_fin,$pObservaciones);

			$solicitud['situacion']=$psituacion; // Alta o Anulación de solicitud
				// 23/04/2007 FIN



		//Antes de asignar el estado a la solicitud:
		// - si es VAC o ADI deberemos comprobar si realmente se ha pedido todas las vacaciones del periodo.
		// Si no se las ha pedido todas, cambiaremos el estado a 'PC' --> pendiente de completar,
		// en caso contrario, el estado será 'PR' para ésta y para todas aquellas solicitudes que estuvieran
		// en estado 'PC'
		// - para el resto, el estado serÃ¡ 'PR'
		if($clase=='ADI' OR $clase=='VAC'){
			if(!$var_vacyadicompletos){
				$solicitud['cestado']='PC';
			}else{
				$solicitud['cestado']='PR';
			}
		}
		else{
			$solicitud['cestado']=$pEstado;
		}
		//Añadimos en avisos sólo los resultados que hayan generado un aviso(A) y no una prohibición(P)
		$solicitud['avisos']=$resultado['mensaje'];


		$solicitud['avisosusuario'] = $resultadoUsuario['mensaje'];
		// 12/12/2007 FIN

		 $solicitud['altamodificada']= $pAltamodificada;//Parametro de entrada

		 if($pAltamodificada == 'S'){
			//Asignamos el valor de estado y las fechas y firmantes que se deben tomar para el alta modificada
			$solicitud['cestado']=$pEstado;
			$solicitud['ffirmaresp']= $ffirmaresp;
			$solicitud['nrpresp'] = $FirmadoPorALR;
			$solicitud['ffirmaper'] = $ffirmaper;
			$solicitud['nrpper'] = $FirmadaPorALP;
		 }

		//En esta submatriz tenemos la solicitud que se insertará finalmente
		$datosValidarSolicitud['codJustificante']=$var_justificante;
		$datosValidarSolicitud['mensajeAlta']=$mensajeAlta;

		if($var_vacyadicompletos)
			$datosValidarSolicitud['vacyAdiCompletos']='S';
		else
			$datosValidarSolicitud['vacyAdiCompletos']='N';
		$datosValidarSolicitud['solicitud']=$solicitud;

		IgepDebug::setDebug(DEBUG_USER,'FIN correcto f_altaSolicitud');
		return $datosValidarSolicitud;

}//Fin f_altaSolicitud


public function f_obtienePrimerDiaSolicitud($pfechaClone,$pperiodo,$pcpro,$pcmun,$pcclase,$nregpgv){
		IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtienePrimerDiaSolicitud');
		$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
		$con = new IgepConexion($g_dsn, true);
		// Recorremos hacia atrás los días desde pfecha hasta encontrar un día que no sea festivo ni fin de semana
		// ni adicional, y devolvemos el siguiente.

		$pfecha= clone ($pfechaClone);
		do {
			$pfecha->subDays(1);
			$var_fecha_aux=$con->prepararFecha($pfecha);
			$var_festivos_aux = $con->consultar(" SELECT fcmn_dias_festivos(to_date('$var_fecha_aux','dd/mm/yyyy')," .
                                " to_date('$var_fecha_aux','dd/mm/yyyy'),'$pcpro','$pcmun') as \"festivos\"");
```

```
                    if ($var_festivos_aux<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_obtienePrimerDiaSolicitud');
                              return -1;
                    }

                    $var_hay_equiv = $con->consultar(" SELECT count(*) as \"cuenta\" FROM TPER_SOLICITUDES" .
                                                  " WHERE nregpgv = '$nregpgv' and cclase = '$pcclase' and" .
                                                  " periodo = '$pperiodo' and " .
                                                  "'$var_fecha_aux' between fecha_ini and fecha_fin and" .


                                                  " NOT ((cestado in ('NR','NI')) OR
                                                  (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                                  (situacion = 'B' OR situacion = 'X')) ");



                    if ($var_hay_equiv<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_obtienePrimerDiaSolicitud');
                              return -1;
                    }
          }while( $var_festivos_aux['0']['festivos'] == 1 or $var_hay_equiv['0']['cuenta'] >= 1);

          //Sumamos un día
          $pfecha->addDays(1);
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtienePrimerDiaSolicitud');
          return $pfecha;
}


public function f_obtieneUltimoDiaSolicitud($pfechaClone,$pperiodo,$pcpro,$pcmun,$pcclase,$nregpgv){
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtieneUltimoDiaSolicitud');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          // Recorremos hacia adelante los días desde pfecha hasta encontrar un dia que no sea festivo ni fin de semana ni
          // adicional, y devolvemos el anterior
          if(is_null($pfechaClone))
                    return null;
          $pfecha = clone $pfechaClone;
          do{
                    $pfecha->addDays(1);//Sumamos un día
                    $var_fecha_aux=$con->prepararFecha($pfecha);

                    $var_festivos_aux = $con->consultar(" SELECT
          fcmn_dias_festivos(to_date('$var_fecha_aux','dd/mm/yyyy'),to_date('$var_fecha_aux','dd/mm/yyyy')," .
                                                  "'$pcpro','$pcmun') as \"festivos\"");
                    if ($var_festivos_aux<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_obtieneUltimoDiaSolicitud');
                              return -1;
                    }
                    $var_hay_equiv= $con->consultar(" SELECT count(*) as \"cuenta\" FROM TPER_SOLICITUDES" .
                                        " WHERE nregpgv = '$nregpgv' and " .
                                        " cclase = '$pcclase' and" .
                                        " periodo = '$pperiodo' and" .
                                        " '$var_fecha_aux' between fecha_ini and fecha_fin and" .

                                        " NOT ((cestado in ('NR','NI')) OR
                                        (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                        (situacion = 'B' OR situacion = 'X')) ");

                     if ($var_hay_equiv<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_obtieneUltimoDiaSolicitud');
                              return -1;
                      }

          }while ($var_festivos_aux['0']['festivos'] == 1 OR $var_hay_equiv['0']['cuenta'] >= 1);
          $pfecha->subDays(1);//Restamos un día
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtieneUltimoDiaSolicitud');
          return $pfecha;
}


public function f_obtienePrimerDiaSolicitudConAnulacion($pfechaClone,$pperiodo,$pcpro,$pcmun,$pcclase,$nregpgv,$panula){
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtienePrimerDiaSolicitudConAnulacion');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          // Recorremos hacia atrás los días desde pfecha hasta encontrar un día que no sea festivo ni fin de semana
          // ni adicional, y devolvemos el siguiente.

          $pfecha= clone $pfechaClone;
          do {
                    $pfecha->subDays(1);
                    $var_fecha_aux=$con->prepararFecha($pfecha);
                    $var_festivos_aux = $con->consultar(" SELECT fcmn_dias_festivos(to_date('$var_fecha_aux','dd/mm/yyyy')," .
                                                  " to_date('$var_fecha_aux','dd/mm/yyyy'),'$pcpro','$pcmun') as
                                                            \"festivos\"");
                    if ($var_festivos_aux<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN error f_obtienePrimerDiaSolicitudConAnulacion');
                              return -1;
                    }

                    if ($panula=='N')
                              $incluyeAnulacion=" OR (situacion='B' and cestado='AP' and ffirmaresolucionanula is not null) ";

                    $var_hay_equiv = $con->consultar(" SELECT count(*) as \"cuenta\" FROM TPER_SOLICITUDES" .
                                        " WHERE nregpgv = '$nregpgv' and cclase = '$pcclase' and" .
                                        " periodo = '$pperiodo' and " .
                                        "'$var_fecha_aux' between fecha_ini and fecha_fin and" .
                                        " NOT ((cestado in ('NR','NI')) OR
                                        (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null)
                                                  $incluyeanulacion) ");
```

```
                    if ($var_hay_equiv<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN error  f_obtienePrimerDiaSolicitudConAnulacion');
                              return -1;
                    }
          }while( $var_festivos_aux['0']['festivos'] == 1 or $var_hay_equiv['0']['cuenta'] >= 1);

          //Sumamos un día
          $pfecha->addDays(1);
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtienePrimerDiaSolicitudConAnulacion');
          return $pfecha;
}


public function f_obtieneUltimoDiaSolicitudConAnulacion($pfechaClone,$pperiodo,$pcpro,$pcmun,$pcclase,$nregpgv,$panula){
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtieneUltimoDiaSolicitudConAnulacion');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          // Recorremos hacia adelante los días desde pfecha hasta encontrar un día que no sea festivo ni fin de semana ni
          // adicional, y devolvemos el anterior
          if(is_null($pfechaClone))
                    return null;
          $pfecha = clone $pfechaClone;
          do{
                    $pfecha->addDays(1);//Sumamos un día
                    $var_fecha_aux=$con->prepararFecha($pfecha);

                    $var_festivos_aux = $con->consultar(" SELECT
          fcmn_dias_festivos(to_date('$var_fecha_aux','dd/mm/yyyy'),to_date('$var_fecha_aux','dd/mm/yyyy')," .
                                                  "'$pcpro','$pcmun') as \"festivos\"");
                    if ($var_festivos_aux<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_obtieneUltimoDiaSolicitudConAnulacion');
                              return -1;
                    }

                    if ($panula=='N')
                              $incluyeAnulacion= " OR (situacion='B' and cestado='AP' and ffirmaresolucionanula is not null)";

                              $var_hay_equiv= $con->consultar(" SELECT count(*) as \"cuenta\" FROM TPER_SOLICITUDES" .
                                        " WHERE nregpgv = '$nregpgv' and " .
                                        " cclase = '$pcclase' and" .
                                        " periodo = '$pperiodo' and" .
                                        " '$var_fecha_aux' between fecha_ini and fecha_fin and" .
                                        " NOT ((cestado in ('NR','NI')) OR
                                        (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null)
                                        $incluyeAnulacion) ");

                     if ($var_hay_equiv<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_obtieneUltimoDiaSolicitud');
                              return -1;
                     }

          }while ($var_festivos_aux['0']['festivos'] == 1 OR $var_hay_equiv['0']['cuenta'] >= 1);
          $pfecha->subDays(1);//Restamos un día
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtieneUltimoDiaSolicitudConAnulacion');
          return $pfecha;
}


public function f_validarSolicitud(&$resultado ,&$pVacyAdiCompletos  ,&$pdias   ,&$pdiaslabo ,&$pdias_adi ,&$pdias_labo_adi

                              ,&$pdias_vac,&$pdias_labo_vac ,&$pultimo  ,&$ptotaldiasadi,& $ptotal_dias,& $ptotal_dias_labo,

                              $pfechaNaci,$tipo,$clase,$pnregpgv,$pPeriodo,$ppfechaInicio,$pfechaFin,$anular,$observaciones,

                              $distancia,&$resultadoUsuario,&$listaSolicitudesAnulacion,&$listaSolicitudesAnulacionResueltas,

                              &$totalDiasAnula,&$totalDiasLaboAnula,&$reqJustificante,&$aportaJustificante,

                              $pdistancia,$pgradoParentesco ){


          IgepDebug::setDebug(DEBUG_USER,'INICIO f_validarSolicitud');
          if ( $pfechaNaci!=null)
                    $fechaNaci = clone $pfechaNaci;
          if ( $ppfechaInicio!=null)
                    $pfechaInicio = clone $ppfechaInicio;
          if ( $ppfechaFin!=null)
                    $pfechaFin = clone $ppfechaFin;

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);


          $datosSolicitud = SolicitudesYPermisos::f_validarSoliPer($pPeriodo,$pfechaInicio,$pfechaFin,$clase, $resultado,
                              $diasTrabajados, $pdias,  $pdiaslabo, $ptipo='S', $pultimo, $ptotaldiasadi,$ptotal_dias,
                              $ptotal_dias_labo,$diasTrabajadosVV6,$pnregpgv, $claseLlamada=$clase);
          if ($datosSolicitud<0){
                    IgepDebug::setDebug(DEBUG_USER,'FIN error 1(f_validarSoliPer) f_validarSolicitud');
                    return -1;
          }
          //Obtener la descripción del permiso
          $datosClasePer=ClasesDePermisos::f_dameDatosClasePermiso($clase,$var_desc_clase,$var_justificante);
          if ($datosClasePer<0){
                    IgepDebug::setDebug(DEBUG_USER,'FIN error 2(f_dameDatosClasePermiso) f_validarSolicitud');
                    return -1;
          }



          SolicitudesYPermisos::f_calcularSolapamientoSoliPer($resultado,$pfechaInicio,$pfechaFin,$var_desc_clase,
                                                              $phaySolapamiento,'S',$clase,$pnregpgv );
```

```
        if($phaySolapamiento){
                IgepDebug::setDebug(DEBUG_USER,'FIN error 3(f_calcularSolapamientoSoliPer) f_validarSolicitud');
                return -1;
        }
        $datosNreg= $con->consultar("SELECT  dapell1, dapell2, dnombre, cpro, cmun  FROM vper_ocupacion where nregpgv
                                                                                    ='".$pnregpgv."'");
        if($datosNreg<0)return -1;
        $dnombre=$datosNreg['0']['dnombre'];
        $apellido1=$datosNreg['0']['apell1'];
        $apellido2=$datosNreg['0']['apell2'];
        $cpro=$datosNreg['0']['cpro'];
        $cmun=$datosNreg['0']['cmun'];

        $datos = IgepSession::dameDatosUsuario();
        $nregpgvQueSolicita = $datos[nrp];

        $datosValidacion=PermisosValidacion::f_leerValidacion($clase,$pfechaInicio,$pfechaFin,$pPeriodo,$diasTrabajados,

                                $ptotal_dias_labo,$ptotaldiasadi,$ptotal_dias, $pnregpgv,$cmun,$cpro,$resultado,

                                $var_desc_clase,$var_justificante,$pVacyAdiCompletos,$pdias_adi,$pdias_labo_adi,

                                $pdias_vac,$pdias_labo_vac,$tipo='S',$fechaNaci,$diasTrabajadosVV6, $pdias,

                                $pdiaslabo,$observaciones, $resultadoUsuario,$distancia,

                                $listaSolicitudesAnulacion,$listaSolicitudesAnulacionResueltas,

                                $totalDiasAnula,$totalDiasLaboAnula,$reqJustificante,$aportaJustificante,

                                $pdistancia,$pgradoParentesco,$nregpgvQueSolicita);

        if ($datosValidacion<0){//En el alta miraremos las variables $resultado*
                IgepDebug::setDebug(DEBUG_USER,'FIN error 4(f_leerValidacion) f_validarSolicitud');
                return -1;
        }
        IgepDebug::setDebug(DEBUG_USER,'FIN correcto f_validarSolicitud');
        return 0;
}//Fin f_validarSolicitud


public function f_GenerarMensajeCorreo($pnregpgv, $pnombre ,$pclase, $pmotivo, $pperiodo, $ppfini,$ppffin ,$pobservaciones,

                                $pmotivonoaut,$pavisos, $palta, $pfirma, $pautorizacion, $pparainteresado,

                                $pmodificacion, $pfirmaanulada, $panularesolucion,$totalDias, $totalDiasLabo,

                                $pnsolicitud, &$mensaje){
        //Generar el mensaje del correo a enviar
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_GenerarMensajeCorreo');
        if($ppfini!=null)$pfini= clone $ppfini;
        if($ppffin!=null)$pffin= clone $ppffin;

        if($pfirma=='N'){
            if($pfirmaanulada=='S')
                $mensaje="Se ha anulado la firma realizada sobre ";
            else{
                    if($panularesolucion=='S')
                            $mensaje= "Se ha anulado la firma de resoluciÃ³n realizada sobre ";
                    else
                            $mensaje = "Se ha realizado ";
            }
        }
        else{
            if($pautorizacion == 'S')
                $mensaje="Se ha dado por conforme ";
            else
                $mensaje = "No se ha dado por conforme ";
        }

        if($palta=='S')
            $mensaje.="el alta ";
        else{
            if($pmodificacion == 'S')
                $mensaje.="la actualizaciÃ³n de las fechas ";
            else
                $mensaje.="la anulaciÃ³n ";
        }
        $mensaje.="de la siguiente solicitud:\n";

        // Si el mensaje no va al interesado, hay que poner los datos del interesado
        if($pparainteresado == 'N')
            $mensaje.="NREGPGV: " . $pnregpgv."\nNOMBRE: ".$pnombre."\n";

          // Si el mensaje es de una conformidad de anulación, añadimos el nº de solicitud
        if ($pfirma == 'S' AND $pautorizacion == 'S' AND $palta=='N' AND $pmodificacion == 'N')
          $mensaje.="\nNÂº SOLICITUD:\n".$pnsolicitud."\n";

        $mensaje.="\nMOTIVO: ".$pmotivo."\n".
        "PERIODO: ".$pperiodo." FECHA INICIO: ".$pfini->formatUser()." FECHA FIN: ".$pffin->formatUser()."\n";

        if(($palta=='S') AND ($pfirma=='N') AND ($pfirmaanulada=='N') AND (!empty($totalDias)) AND (!empty($totalDiasLabo))){
                    if($pclase == 'VAC' OR $pclase == 'ADI' OR $pclase == 'VV6' OR $pclase == 'AAP'){
                         $mensaje.="DÃ AS NATURALES ACUMULADOS: $totalDias  DÃ AS HÃ BILES ACUMULADOS: $totalDiasLabo \n";
                    }
        }

        if(!empty($pobservaciones))
        $mensaje.="OBSERVACIONES: ".$pobservaciones."\n";

          // No sacamos los avisos cuando es una conformidad de anulación
```

```
        if(!empty($pavisos) AND !($pfirma == 'S' AND $pautorizacion == 'S' AND $palta=='N' AND $pmodificacion == 'N'))
          $mensaje.="\nAVISOS:\n".$pavisos."\n";

        //No se ha dado por conforme el alta o la anulacion, hay que incluir el motivo de no conformidad
        if($pfirma == 'S' AND $pautorizacion == 'N')
                                $mensaje.= "\nMOTIVO DE LA NO CONFORMIDAD: ".$pmotivonoaut;
        IgepDebug::setDebug(DEBUG_USER,'FIN f_GenerarMensajeCorreo');
}


public function f_generarMensajeNotificacion($pnsolicitud_persona,$ppffirma,$ppositiva,&$mensaje){

        //Generar el mensaje de la notificaciÃ³n a enviar
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_generarMensajeNotificacion');
        $pffirma = clone $ppffirma;

        $mensaje = "Por resolución del Subsecretario de fecha " . $pffirma->formatUser();

        if($ppositiva=='negativas')
            $mensaje.= ' NO';

        $mensaje.= " le han sido autorizadas las siguientes solicitudes:\n";
        foreach($pnsolicitud_persona as $solicitud) {
            $mensaje.="- '".$solicitud['motivo']."' del ".$solicitud['fecha_ini']->format('d/m/Y').' al
                                                ".$solicitud['fecha_fin']->format('d/m/Y')."\n";
            if($ppositiva=='negativas'){
                if(empty($solicitud['motivonoautper']))
                    $mensaje.=' No autorizada por: ' . $solicitud['motivonoautresp'] . "\n";
                else
                    $mensaje.=' No autorizada por: ' . $solicitud['motivonoautper'] . "\n";
            }
        }
        IgepDebug::setDebug(DEBUG_USER,'FIN f_generarMensajeNotificacion');
    }


public function f_enviarCorreo($pRemitente,$pDestino,$pMensaje,$pAsunto){
        include_once "igep/include/IgepCorreo.php";
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_enviarCorreo');
        $dirCorreoRemitente = IgepCorreo::correoNREGPGV($pRemitente);
        if (is_array($pDestino)){
                $dirsCorreoDestino=array();
        foreach($pDestino as $nreg){
                $correo=IgepCorreo::correoNREGPGV($nreg);
                array_push($dirsCorreoDestino,$correo['0']);
        }
        }else{
          $dirsCorreoDestino=IgepCorreo::correoNREGPGV($pDestino);
        }
        IgepDebug::setDebug(DEBUG_USER,'FIN f_enviarCorreo<pre>'.print_r($dirsCorreoDestino,true).
                                                '/'.$pAsunto.'/'.$pMensaje.'/'.$dirCorreoRemitente['0'].'</pre>');
        if(empty($dirsCorreoDestino)) return false;
        return IgepCorreo::sinAnexo('no_responder@personal.gva.es',
                                                $dirsCorreoDestino,$pAsunto,$pMensaje,$dirCorreoRemitente['0'],true);
}


public function f_calculaEstadoPCoPR($var_cclase,$var_nregpgv,$var_periodo,$popcionmenu='AAA'){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_calculaEstadoPCoPR');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);

          $nulo=null;
          $fecha_actual=new gvHidraTimestamp();
          $fecha_actual->setTime(0,0,0);

          if (($var_cclase=='VAC') or ($var_cclase=='ADI')){

                    $total_dias_vac=0;$ptotaldias=0;$pdias=0;$ptotaldiaslabo=0;$pdiaslabo=0;
                    $total_diaslabo_vac=0;$total_diaslabo_vac=0;

          // ------------------------------------------------------------------------------------------------
          // Total días solicitados de VACACIONES (VAC)
          // ------------------------------------------------------------------------------------------------
          $datosVAC = SolicitudesYPermisos:: f_diasPermiso($var_periodo,$fecha_actual,$fecha_actual,
                                        'VAC', $total_dias_vac, $total_diaslabo_vac,$total_dias_vac_act,
                                        $total_diaslabo_vac_act,'S',$nulo1, $nulo2,$var_nregpgv,$var_cclase);

          //Al total de días de vacaciones le restamos los que pasamos como solicitados actualmente
          // Solo restamos del total si la clase de permiso es VAC,
          // ya que con ADI ya se han restado en f_diasPermiso
          if($var_cclase=='VAC'){
                    $total_dias_vac = $total_dias_vac - $total_dias_vac_act;
                    $total_diaslabo_vac = $total_diaslabo_vac - $total_diaslabo_vac_act;
          }

          //------------------------------------------------------------------------------------------------
                                        // Total días solicitados de días ADICIONALES (ADI)
          // ------------------------------------------------------------------------------------------------
          $datosADI = SolicitudesYPermisos:: f_diasPermiso($var_periodo,$fecha_actual,$fecha_actual,
                                                'ADI', $total_dias_adi, $total_diaslabo_adi,$total_dias_adi_act,
                                                $total_diaslabo_adi_act,'S',$nulo3,$nulo4,$var_nregpgv,$var_cclase);



          //Al total de días adicionales le restamos los que pasamos como solicitados actualmente
          // Solo restamos del total si la clase de permiso es ADI,ya que con VAC ya se han restado en f_diasPermiso
```

```
        if($var_cclase=='ADI'){
                $total_dias_adi = $total_dias_adi - $total_dias_adi_act;
                $total_diaslabo_adi = $total_diaslabo_adi - $total_diaslabo_adi_act;
        }

        //Calculamos los meses naturales para LIP, ya que se deberÃ¡ recalcular el valor de
        // maxDias de VAC cuando pedimos 1 mes o mÃ¡s de LIP
        // ------------------------------------------------
        // Total meses naturales solicitados de LICENCIA POR INTERES PARTICULAR LIP
        // ------------------------------------------------

        SolicitudesYPermisos::f_mesesNaturalesPermiso($var_periodo,$fecha_actual,$fecha_actual,'LIP',

        $totalMesesNatLIP,$mesesNatLIP,$totalRestoDiasLIP,$restoDiasLIP,'S','ALTA',$var_nregpgv,$var_cclase );

        // Al total de meses y dias de LIP le restamos los que pasamos como solicitados actualmente



        //CALCULO DE DIAS TRABAJADOS DURANTE EL AÑO

        //Comprobamos que sea o no de puertos
        $res = $con->consultar("SELECT depuertos  FROM VPER_OCUPACION WHERE nregpgv = '$nregpgv'");
        $esDeP=$res[0]['depuertos'];

        if (($esDeP=='N') or ($esDeP==$nulo)){
                $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
                // Conexion a Oracle

                $conexion=new IgepConexion($g_oracle_dsn);
                $res=$conexion->consultar("SELECT ftpos as \"ftpos\" FROM tper_puestos_trab_admon WHERE
                        nregpgv='$var_nregpgv' and fbajpad is null",array('DATATYPES'=>array('ftpos'=>TIPO_FECHA)));
                $ftoma=$res[0]['ftpos'];
                $esDePuertos='N';

        }
        else{
                $esDePuertos='S';
                $ftoma= new gvHidraTimestamp('01/01/2200');
                $ftoma->setTime(0,0,0);
        }



        UsuariosDePersonal:: f_obtenerDiasTrabajados($var_nregpgv,$var_periodo,$esDePuertos,$ftoma,
                                                        $var_diastrabajados,$nulo);


        PermisosValidacion:: f_LeerMinMaxDias('VAC', $diasMinimo, $diasMaximo, $diasNaturales,$unidad);

        //Permitir que el personal funcionario incorporado durante el año, pueda pedirse el máximo de VAC y ADI
        UsuariosDePersonal::f_dameRelJur($var_nregpgv,$creljur);
        if($var_diastrabajados == -1 OR $creljur != 'CL'){
                 $maxDias = $diasMaximo;
        }else {
                $maxDias = round( ($diasMaximo * $var_diastrabajados) / 365 );
        }


        // Obtenemos el valor del aÃ±o para el que se conocen los dias adicionales de vacaciones
        Usuarios::f_obtenerAnyoDiasADI($panyodiasADI);
        // Obtenemos los dias adicionales en caso de solicitar un permiso de vacaciones o dias adicionales
        if ($var_periodo < $panyodiasADI){
                $res = $con->consultar("SELECT diasadi_ant as \"totaldiasadi\" from tper_personas  where nregpgv =
                                                        '".$pnregpgv."'");
        } else {
                $res = $con->consultar("SELECT diasadi_act as \"totaldiasadi\" from tper_personas  where nregpgv =
                                                        '".$pnregpgv."'");
        }
        if ($res==-1){
              return -1;
        }
        $var_totaldiasadi = $res[0]['totaldiasadi'];


        if ($totalMesesNatLIP>=1){
                if (($total_diaslabo_vac >= $maxDias - $maxDias/12 *($totalMesesNatLIP+$totalRestoDiasLIP/30)) and
                                ($total_diaslabo_adi>=$var_totaldiasadi)){

                        $var_estado='PR';
                }
                else {
                        $var_estado='PC';
                }
        }
        else {
                if (($total_diaslabo_vac>=$maxDias) and ($total_diaslabo_adi>=$var_totaldiasadi))
                        $var_estado='PR';
                else
                        $var_estado='PC';
        }

        }// Cuando no pedimos VAC o ADI el estado siempre serÃ¡ PR
        else {
                $var_estado='PR';
        }
        IgepDebug::setDebug(DEBUG_USER,'FIN f_calculaEstadoPCoPR');
        return $var_estado;
}


        //Devuelve los días con y sin justificante de ENC solicitados por el interesado en el periodo actual
```

```
function f_dameDiasENC($pPeriodo,$pnregpgv,&$pdiasSinJustificante,&$pdiasConJustificante){
          //Conectamos a posgreSQL
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameDiasENC');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);

          //pdiasSinJustificante
          $res = $con->consultar("SELECT sum(s.fecha_fin - s.fecha_ini + 1 -
                              fcmn_dias_festivos(s.fecha_ini,s.fecha_fin,o.cpro,o.cmun)) as \"pdiasSinJustificante\"
                              FROM TPER_SOLICITUDES s,VPER_OCUPACION o
                              WHERE s.nregpgv = '$pnregpgv' and s.cclase = 'ENC' and s.periodo = '$pPeriodo' and
                                    s.nregpgv = o.nregpgv and( NOT ((s.cestado in ('NR','NI')) OR
                                   (s.situacion = 'A' and s.cestado='NP' and s.ffirmaresolucion is not null) OR
                                   (s.situacion = 'B' and s.cestado='AP' and s.ffirmaresolucionanula is not null))) and
                                    s.fecha_fin is not null and s.justificante = 'N'");
          if($res!=-1)
                    $pdiasSinJustificante=$res[0]['pdiasSinJustificante'];


          $res = $con->consultar("SELECT sum(s.fecha_fin - s.fecha_ini + 1 -
                              fcmn_dias_festivos(s.fecha_ini,s.fecha_fin,o.cpro,o.cmun)) as \"pdiasConJustificante\"
                              FROM TPER_SOLICITUDES s,VPER_OCUPACION o
                              WHERE s.nregpgv = '$pnregpgv' and s.cclase = 'ENC' and s.periodo = '$pPeriodo' and
                                    s.nregpgv = o.nregpgv and (  NOT ((s.cestado in ('NR','NI')) OR
                                   (s.situacion = 'A' and s.cestado='NP' and s.ffirmaresolucion is not null) OR
                                   (s.situacion = 'B' and s.cestado='AP' and s.ffirmaresolucionanula is not null))) and
                                    s.fecha_fin is not null and s.justificante = 'S'");
          if($res!=-1)
                    $pdiasConJustificante=$res[0]['pdiasConJustificante'];
          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameDiasENC');
}



          //Se genera el permiso y se envía el correo al interesado de las solicitudes de la resolución pasada por parámetros


public function f_firmaResolucion($dePuertos, $positivas, $tipoResolucion, $pfresolucion, $pnumResol, $pffirmaResolucion,
                                  $nsolicitud, & $con, & $conOra){

          //Inicializamos el flag de error
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_firmaResolucion');

          $solicitudesBorradas = '';
          $str_avisos = '';

          $fresolucion = clone $pfresolucion;
          $ffirmaResolucion = clone $pffirmaResolucion;

          $correcto = true;

          $fresolucionBD= $con->prepararFecha($fresolucion);

          $permisosCIT = ClasesDePermisos::f_clasesPermisosPorTramite('C');
          $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');
          foreach($permisosCIT as $index=>$tupla){

                    $var_cclasesCIT=$var_cclasesCIT."'".$permisosCIT[$index]['cclase']."',";

          }
          $var_cclasesCIT=rtrim($var_cclasesCIT,",");

          foreach($permisosDGAA as $index=>$tupla){

                    $var_cclasesDGAA=$var_cclasesDGAA."'".$permisosDGAA[$index]['cclase']."',";

          }
          $var_cclasesDGAA=rtrim($var_cclasesDGAA,",");


          if($positivas == 'S'){
          // Tratamos primero las anulaciones y luego las altas para evitar insertar en TPER_PERMISOS registros con la clave
          // duplicada. Es decir, primero borraremos las anulaciones de TPER_PERMISOS y luego insertaremos las altas en
          TPER_PERMISOS

          // Firma de solicitudes de anulación
                    if($tipoResolucion == 'C'){
                              $query_solicitudesAnula = "
                              SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, justificante, observaciones,
                                   fnacimientomenor, motivonoautper, avisos, nsolanulacion, cclase, situacion, cestado,
                                   fanulaper, ffirmaresolucion
                              FROM tper_solicitudes
                              WHERE
                                 fresolucionanula = '".$fresolucionBD."' and
                                 numresolanula = ".$pnumResol." and
                                 situacion ='B' and
                                 cestado ='AP' and
                                 cclase in (".$var_cclasesCIT.") and
                                 nregpgv in (select v.nregpgv from vper_ocupacion v where depuertos='".$dePuertos."')";

                    }
                    else{
                              $query_solicitudesAnula = "
                              SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, justificante, observaciones,
                                   fnacimientomenor, motivonoautper, avisos, nsolanulacion, cclase, situacion, cestado,
                                   fanulaper, ffirmaresolucion
                              FROM tper_solicitudes
                              WHERE nsolicitud = $nsolicitud and
                                 situacion ='B' and
                                 cestado ='AP' and
                                 cclase in (".$var_cclasesDGAA.")";
                    }
```

```
$solicitudesAnula = $con->consultar($query_solicitudesAnula,array(
                              'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,
                              'fecha_fin'=>TIPO_FECHA,'fnacimientomenor'=>TIPO_FECHA,)));

if(count($solicitudesAnula)>0){

    foreach($solicitudesAnula as $solicitudAnula){
          // Obtenemos la provincia y municipio del interesado
          $cpro= null; $cmun = null;
          UsuariosDePersonal::f_obtenerMunPro($solicitudAnula['nregpgv'], $cpro, $cmun);
          // Obtenemos el valor de diasTrabajados para pasarlo en f_generarPermiso(...,diasTrabajados)
          $diasTrabajados = null;
          $nulo = null;

          SolicitudesYPermisos::f_validarSoliPer($solicitudAnula['periodo'],
                    $solicitudAnula['fecha_ini'],$solicitudAnula['fecha_fin'],$solicitudAnula['cclase'],
                    $resultado,$diasTrabajados,$nulo,$nulo,'S',$nulo,$nulo,$nulo,$nulo,
                    $nulo,$solicitudAnula['nregpgv'],$solicitudAnula['cclase']);
          if($res<0){
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                    return array('0'=>-1);
          }
          // Para las anulaciones normales
          if (Is_null($solicitudAnula['nsolanulacion'])){
                    $fini = clone $solicitudAnula['fecha_ini'];
                    $ffin = clone $solicitudAnula['fecha_fin'];
          }
          else{

                    // Se busca el permiso de la solicitud original. La anulación parcial ha generado
                    varios intervalos de solicitudes
                    // pero el permiso sigue siendo único, por lo tanto el permiso tendrá como nsolicitud
                    el valor nsolanulacion
                    $query = "
                              SELECT fecha_ini, fecha_fin
                              FROM tper_permisos
                              WHERE nsolicitud = ".$solicitudAnula['nsolanulacion'];
                                        $resFechasPermiso = $con->consultar($query ,array(
                              'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
                    $fini = clone $resFechasPermiso[0]['fecha_ini'];
                    $ffin = clone $resFechasPermiso[0]['fecha_fin'];
          }
          if((gvHidraTimestamp::cmp($fini, $solicitudAnula['fecha_ini']) == 0)  AND
                    (gvHidraTimestamp::cmp($ffin, $solicitudAnula['fecha_fin']) == 0) ) {

                    //Borramos el alta de la base de datos

                    //Borramos de la tabla de personal TPER_PERMISOS el permiso cuya solicitud de
                    anulación ha sido autorizada

                    $res = Permisos::f_borrarPermiso($con, $conOra, $solicitudAnula['nregpgv'],
                                                  $solicitudAnula['cclase'], $fini,$var_avisoNominas);

                    if($res==-1){
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                              return array('0'=>-1);
                    }


                    if ($var_avisoNominas){

                              $solicitudesBorradas = $solicitudesBorradas."DNI:
                                        ".substr($solicitudAnula['nregpgv'],1,8).";
                                         motivo: ".$solicitudAnula['cclase']."; fecha inicio:
                                        ".$fini."\n";
                    }

          }
          elseif(gvHidraTimestamp::cmp($fini, $solicitudAnula['fecha_ini']) == 0){
                    // Recortamos el permiso atrasando la fecha de inicio (ha habido una anulación
                    parcial)
                    //Borramos de la tabla de personal TPER_PERMISOS el permiso cuya solicitud de
                    anulación ha sido autorizada

                    $res = Permisos::f_borrarPermiso($con, $conOra, $solicitudAnula['nregpgv'],
                                                  $solicitudAnula['cclase'], $fini,$var_avisoNominas);

                    if($res==-1){
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                              return array('0'=>-1);
                    }

                    if ($var_avisoNominas){
                              $solicitudesBorradas = $solicitudesBorradas."DNI:
                                        ".substr($solicitudAnula['nregpgv'],1,8).";
                                         motivo: ".$solicitudAnula['cclase']."; fecha inicio:
                                        ".$fini."\n";
                    }
                    // Se obtiene el nsolicitud de la solicitud de alta correspondiente al permiso que
                    estamos modificando
                    $var_fecha_fin= clone $solicitudAnula['fecha_fin'];
                    $var_fecha_fin->addDays(1);
                    $res_nsolicitud=$con->consultar("SELECT nsolicitud FROM tper_solicitudes
                                                  WHERE nregpgv = '".$solicitudAnula['nregpgv']."' and
                                                        cclase = '".$solicitudAnula['cclase']."' and
                                                  fecha_ini = '".$con->prepararFecha($var_fecha_fin)."' and
                                                  fecha_fin=  '".$con->prepararFecha($ffin)."'");
                    $var_nsolicitud=$res_nsolicitud[0]['nsolicitud'];

                    if($res_nsolicitud==-1){
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                              return array('0'=>-1);
```

```
                }


                //solicitudAnula['fecha_fin'] + 1;
                $inicio = clone     $solicitudAnula['fecha_fin'];
                $inicio->addDays(1);

                //se genera con el nsolicitud de la solicitud de alta correspondiente al permiso que
                estamos modificando
                $res = Permisos::f_generarPermiso($con,$conOra,$solicitudAnula['nregpgv'], $cpro,
                                        $cmun, $solicitudAnula['cclase'], $inicio,
                                        $solicitudAnula['fecha_fin'],
                                        $solicitudAnula['periodo'], $solicitudAnula['dnombre']
                            .' '.$solicitudAnula['apell1'].' '.$solicitudAnula['apell2'],

                            $solicitudAnula['justificante'],$solicitudAnula['observaciones'],
                            $var_nsolicitud,$solicitudAnula['fnacimientomenor'],
                            $diasTrabajados);

                if($res['0']==-1){
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                        return $res;
                }
                if(!empty($res['errorAviso'])) {
                        $str_avisos.=$res['errorAviso'] . ' ';
                }
        }
        elseif(gvHidraTimestamp::cmp($ffin, $solicitudAnula['fecha_fin']) == 0 ){
        // caso de uso
        // Recortamos el permiso adelantando la fecha de fin (ha habido una anulación parcial)

        //Borramos de la tabla de personal TPER_PERMISOS el permiso cuya solicitud de anulación ha sido
        autorizada

                $res = Permisos::f_borrarPermiso($con, $conOra, $solicitudAnula['nregpgv'],
                                $solicitudAnula['cclase'], $fini,$var_avisoNominas);

                if($res==-1){
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                        return array('0'=>-1);
                }
                if ($var_avisoNominas){
                        $solicitudesBorradas = $solicitudesBorradas."DNI:
                                        ".substr($solicitudAnula['nregpgv'],1,8).";
                                        motivo: ".$solicitudAnula['cclase']."; fecha inicio:
                                        ".$fini."\n";
                }


                //Se obtiene el nsolicitud de la solicitud de alta correspondiente al permiso que
                estamos modificando
                $var_fecha_ini= clone $solicitudAnula['fecha_ini'];
                $var_fecha_ini->subDays(1);
                $res_nsolicitud=$con->consultar("SELECT nsolicitud FROM tper_solicitudes
                                        WHERE nregpgv = '".$solicitudAnula['nregpgv']."' and
                                                cclase = '".$solicitudAnula['cclase']."' and
                            fecha_ini = '".$con->prepararFecha($fini)."'
                            and fecha_fin='".$con->prepararFecha($var_fecha_ini)."' ");
                $var_nsolicitud=$res_nsolicitud[0]['nsolicitud'];


                if($res_nsolicitud==-1){
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                        return array('0'=>-1);
                }



                //solicitudAnula['fecha_ini'] - 1;
                $fin = clone $solicitudAnula['fecha_ini'];
                $fin->subDays(1);
                //  se genera con el nsolicitud de la solicitud de alta correspondiente al permiso que
                estamos modificando
                $res =Permisos::f_generarPermiso($con,$conOra,$solicitudAnula['nregpgv'], $cpro,
                                $cmun, $solicitudAnula['cclase'], $fini, $fin,
                                $solicitudAnula['periodo'], $solicitudAnula['dnombre'].'
                                '.$solicitudAnula['apell1'].' '.$solicitudAnula['apell2'],
                                $solicitudAnula['justificante'],$solicitudAnula['observaciones'],
                        $var_nsolicitud, $solicitudAnula['fnacimientomenor'], $diasTrabajados);

                if($res['0']==-1){
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                        return $res;
                }
                if(!empty($res['errorAviso'])) {
                        $str_avisos.=$res['errorAviso'] . ' ';
                }
        }
        else{
                // Generamos dos nuevos permisos (ha habido una anulación parcial)

                //Borramos de la tabla de personal TPER_PERMISOS el permiso cuya solicitud de
                anulación ha sido autorizada

                $res = Permisos::f_borrarPermiso($con, $conOra, $solicitudAnula['nregpgv'],
                                        $solicitudAnula['cclase'], $fini,$var_avisoNominas);


                if($res==-1){
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                        return array('0'=>-1);
                }
```

```
                                if ($var_avisoNominas){
                                        $solicitudesBorradas = $solicitudesBorradas."DNI:
                                                                ".substr($solicitudAnula['nregpgv'],1,8).";
                                                        motivo: ".$solicitudAnula['cclase']."; fecha inicio:
                                                        ".$fini."\n";
                                }



                                // Se obtienen los nsolicitud de las solicitudes de alta correspondientes a los
                                permisos que estamos modificando

                                $var_fecha_ini= clone $solicitudAnula['fecha_ini'];
                                $var_fecha_ini->subDays(1);
                                $var_fecha_fin= clone $solicitudAnula['fecha_fin'];
                                $var_fecha_fin->addDays(1);

                                $res_nsolicitud1=$con->consultar("SELECT nsolicitud FROM tper_solicitudes
                                                        WHERE nregpgv = '".$solicitudAnula['nregpgv']."' and
                                                                cclase = '".$solicitudAnula['cclase']."' and
                                                fecha_ini = '".$con->prepararFecha($fini)."' and
                                                fecha_fin='".$con->prepararFecha($var_fecha_ini)."' ");
                                                $res_nsolicitud2=$con->consultar("SELECT nsolicitud FROM
                                                tper_solicitudes WHERE nregpgv = '".$solicitudAnula['nregpgv']."'
                                                and cclase = '".$solicitudAnula['cclase']."' and fecha_ini =
                                                '".$con->prepararFecha($var_fecha_fin)."' and fecha_fin='".$con-
                                                >prepararFecha($ffin)."' ");


                                $var_nsolicitud1=$res_nsolicitud1[0]['nsolicitud'];
                                $var_nsolicitud2=$res_nsolicitud2[0]['nsolicitud'];

                                if($res_nsolicitud1==-1){
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                                        return array('0'=>-1);
                                }
                                if($res_nsolicitud2==-1){
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                                        return array('0'=>-1);
                                }



                                //solicitudAnula['fecha_ini'] - 1;
                                $fin = clone $solicitudAnula['fecha_ini'];
                                $fin->subDays(1);
                                //Se genera con el primer nsolicitud de la solicitud de alta correspondiente al
                                permiso que estamos modificando
                                                $res =
                                Permisos::f_generarPermiso($con,$conOra,$solicitudAnula['nregpgv'], $cpro,$cmun,
                                $solicitudAnula['cclase'], $fini, $fin, $solicitudAnula['periodo'],
                                $solicitudAnula['dnombre'].' '.$solicitudAnula['apell1'].' '.$solicitudAnula['apell2'],
                                $solicitudAnula['justificante'],$solicitudAnula['observaciones'],$var_nsolicitud1 ,
                                $solicitudAnula['fnacimientomenor'], $diasTrabajados);

                                if($res['0']==-1){
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                                        return $res;
                                }
                                if(!empty($res['errorAviso'])) {
                                        $str_avisos.=$res['errorAviso'] . ' ';
                                }
                                //solicitudAnula['fecha_fin'] + 1;
                                $ini = clone $solicitudAnula['fecha_fin'];
                                $ini->addDays(1);
                                // Se genera con el segundo nsolicitud de la solicitud de alta correspondiente al
                                permiso que estamos modificando
                                $res = Permisos::f_generarPermiso($con,$conOra,$solicitudAnula['nregpgv'], $cpro,
                                        $cmun, $solicitudAnula['cclase'], $ini, $ffin, $solicitudAnula['periodo'],
                                        $solicitudAnula['dnombre'].' '.$solicitudAnula['apell1'].'
                                        '.$solicitudAnula['apell2'],$solicitudAnula['justificante'],
                                        $solicitudAnula['observaciones'],$var_nsolicitud2 ,
                                        $solicitudAnula['fnacimientomenor'], $diasTrabajados);

                                if($res['0']==-1){
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                                        return $res;
                                }
                                if(!empty($res['errorAviso'])) {
                                        $str_avisos.=$res['errorAviso'] . ' ';
                                }
                        }

                        if ($solicitudAnula['cclase']=='VV6' or $solicitudAnula['cclase']=='AAP'){
                        //Recalculamos el ordinal de los días solicitados de VV6 o AAp por un interesado en un
                        determinado periodo,

                        // al haber anulado un VV6 o AAP
                                SolicitudesYPermisos::f_recalcularOrdinalBorrado(
                                                                                $solicitudAnula['nregpgv'],
                                                                                $solicitudAnula['periodo'],
                                                                                $solicitudAnula['cclase']);
                        }


                    }//Fin de foreach
                }

        // Firma de solicitudes de alta
        if($tipoResolucion == 'C'){
                $query_solicitudes= "SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, justificante,
```

```
                                        observaciones, fnacimientomenor,motivonoautper, avisos, cclase, situacion,
                                        cestado, fanulaper, ffirmaresolucion
                                FROM tper_solicitudes
                                WHERE
                           fresolucion = '".$fresolucionBD."' and
                           numresol = ".$pnumResol." and
                           situacion ='A' and
                           cestado = 'AP' and
                           cclase in (".$var_cclasesCIT.") and
                           nregpgv in (select v.nregpgv from vper_ocupacion v where depuertos='".$dePuertos."')
                        UNION
                        SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, justificante, observaciones,
                               fnacimientomenor,motivonoautper, avisos, cclase, situacion, cestado,
                               fanulaper, ffirmaresolucion
                        FROM tper_solicitudes
                        WHERE
                           fresolucion = '".$fresolucionBD."' and
                           numresol = ".$pnumResol." and
                           situacion ='B' and
                           cclase in (".$var_cclasesCIT.") and
                           nregpgv in (select v.nregpgv from vper_ocupacion v where depuertos='".$dePuertos."')";
                        }
                        else{
                                $query_solicitudes= "
                        SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, justificante, observaciones,
                              fnacimientomenor,
                              motivonoautper, avisos, cclase, situacion, cestado, fanulaper, ffirmaresolucion
                        FROM tper_solicitudes
                        WHERE nsolicitud = $nsolicitud and
                           situacion ='A' and
                           cestado = 'AP' and
                           cclase in (".$var_cclasesDGAA.")
                        UNION
                        SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, justificante, observaciones,
                              fnacimientomenor,
                              motivonoautper, avisos, cclase, situacion, cestado, fanulaper, ffirmaresolucion
                        FROM tper_solicitudes
                        WHERE nsolicitud = $nsolicitud and
                           situacion ='B' and
                           cestado <> 'AP' and
                           cclase in (".$var_cclasesDGAA.")";
                        }
                $solicitudes = $con->consultar($query_solicitudes,array(
                'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,'fnacimientomenor'=>TIPO_FECH   A,)));
                foreach($solicitudes as $solicitud){
                        //Grabamos la solicitud en la base de datos de personal

                        // Obtenemos la provincia y municipio del interesado
                        $cpro= null; $cmun = null;
                        UsuariosDePersonal::f_obtenerMunPro($solicitud['nregpgv'], $cpro, $cmun);

                        // Obtenemos el valor de diasTrabajados para pasarlo en f_generarPermiso(...,diasTrabajados)
                        $diasTrabajados = null;
                        $nulo = null;$resultado = null;
                        $res = SolicitudesYPermisos::f_validarSoliPer($solicitud['periodo'],$solicitud['fecha_ini'],
                                                $solicitud['fecha_fin'],$solicitud['cclase'],$resultado,
                                                $diasTrabajados,$nulo,$nulo,'S',$nulo,$nulo,$nulo,$nulo,
                                                $nulo,$solicitud['nregpgv'],$solicitud['cclase']);
                        if($res<0){
                                IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                                return array('0'=>-1);
                        }

                        $res = Permisos::f_generarPermiso($con,$conOra,$solicitud['nregpgv'], $cpro,
                                $cmun, $solicitud['cclase'], $solicitud['fecha_ini'], $solicitud['fecha_fin'],
                                $solicitud['periodo'], $solicitud['dnombre'].' '.$solicitud['apell1'].'
                                '.$solicitud['apell2'],$solicitud['justificante'],$solicitud['observaciones'],
                                $solicitud['nsolicitud'], $solicitud['fnacimientomenor'], $diasTrabajados);

                        if($res['0']==-1){
                                IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
                                return array('0'=>-1);
                        }
                        if(!empty($res['errorAviso'])) {
                                $str_avisos.=$res['errorAviso'] . ' ';
                        }
                }

                //Enviar notificación a los usuarios que realizan las solicitudes en cada resolución
                $totalSolicitudesResolucion = $solicitudes;
                if(count($solicitudesAnula)>0){
                        foreach($solicitudesAnula as $sol){
                                array_push($totalSolicitudesResolucion,$sol);
                        }
                }

                Solicitudes::f_notificarFirmaResolucion($totalSolicitudesResolucion, $ffirmaResolucion, 'S');
        }
        // Eliminamos el envío de notificación al interesado en el caso de resoluciones negativas porque esta notificación
        se hace por escrito.

        //Aviso a nominas
        if($solicitudesBorradas!='')
                $str_avisos.="Los siguientes permisos ya borrados, tienen creadas incidencias para la aplicación de
                        nóminas. Avisar a nóminas:\n".$solicitudesBorradas;

        IgepDebug::setDebug(DEBUG_USER,'FIN f_firmaResolucion');
        $result['0'] = 0;
        $result['errorAviso'] = $str_avisos;

        return $result;
} //Fin f_firmaResolucion
```

```
public function f_anularFirmaResolucion($dePuertos, $positivas, $tipoResolucion, $fresolucion, $ffirmaResolucion,
                                        $nsolicitud, &$con, &$conOra){

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_anularFirmaResolucion');

        $str_avisos = '';
        $solicitudesBorradas = '';
        $var_avisoNominas  = null;


        $fresolucion = clone $fresolucion;
        $ffirmaResolucion = clone $ffirmaResolucion;


        $permisosCIT = ClasesDePermisos::f_clasesPermisosPorTramite('C');
        $permisosDGAA = ClasesDePermisos::f_clasesPermisosPorTramite('D');
        foreach($permisosCIT as $index=>$tupla){

                $var_cclasesCIT=$var_cclasesCIT."'".$permisosCIT[$index]['cclase']."',";
        }
        $var_cclasesCIT=rtrim($var_cclasesCIT,",");

        foreach($permisosDGAA as $index=>$tupla){

                $var_cclasesDGAA=$var_cclasesDGAA."'".$permisosDGAA[$index]['cclase']."',";

        }
        $var_cclasesDGAA=rtrim($var_cclasesDGAA,",");


        if($positivas=='S'){
        // Firma de solicitudes de alta
                if($tipoResolucion=='C'){
                      $query = "
                         SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, motivonoautper, avisos,
                                cclase
                         FROM tper_solicitudes
                         WHERE
                            fresolucion = '".$con->prepararFecha($fresolucion)."' and
                            ffirmaresolucion = '".$con->prepararFecha($ffirmaResolucion)."' and
                            situacion ='A' and
                            cestado = 'AP' and
                            cclase in (".$var_cclasesCIT.") and
                            nregpgv in (select v.nregpgv from vper_ocupacion v where depuertos='".$dePuertos."')
                         UNION
                         SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, motivonoautper, avisos,
                                cclase
                         FROM tper_solicitudes
                         WHERE
                            fresolucion = '".$con->prepararFecha($fresolucion)."' and
                            ffirmaresolucion = '".$con->prepararFecha($ffirmaResolucion)."' and
                            situacion ='B' and
                            cclase in (".$var_cclasesCIT.") and
                            nregpgv in (select v.nregpgv from vper_ocupacion v where depuertos='".$dePuertos."')";
                }
                else{
                      $query = "
                         SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, motivonoautper, avisos,
                                cclase
                         FROM tper_solicitudes
                         WHERE
                            nsolicitud = $nsolicitud and
                            situacion ='A' and
                            cestado = 'AP' and
                            cclase in (".$var_cclasesDGAA.")
                         UNION
                         SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, motivonoautper, avisos,
                                cclase
                         FROM tper_solicitudes
                         WHERE nsolicitud = $nsolicitud and
                            situacion ='B' and
                            cestado <> 'AP' and
                            cclase in (".$var_cclasesDGAA.")";
                }
                $solicitudes = $con->consultar($query,array(
                                        'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
                IgepDebug::setDebug(DEBUG_USER,'f_anularFirmaResolucion antes del borrado<pre>solicitudes:
                                        '.print_r($solicitudes,true).' &total: '.count($solicitudes[0]).'</pre>');
                if(count($solicitudes[0])>0){
                         foreach($solicitudes as $solicitud){
                         //Borramos el alta de la base de datos
                         //Borramos de la tabla de personal TPER_PERMISOS el permiso cuya solicitud de anulación ha sido
                         autorizada
                                 $res = Permisos::f_borrarPermiso($con, $conOra, $solicitud['nregpgv'],
                                                 $solicitud['cclase'], $solicitud['fecha_ini'],$var_avisoNominas);

                                 if($res==-1){
                                         IgepDebug::setDebug(DEBUG_USER,'FIN f_anularFirmaResolucion tras no poder
                                                                 borrar de tper_permisos');
                                         return array('0'=>-1);
                                 }

                                 if ($var_avisoNominas){
                                         $solicitudesBorradas = $solicitudesBorradas."DNI:
                                                         ".substr($solicitud['nregpgv'],1,8).";
                                                         motivo: ".$solicitud['cclase']."; fecha inicio:
                                                         ".$solicitud['fecha_ini']."\n";
```

```
                        }

                        // Obtenemos la descripciÃ³n y si requiere justificante la clase de permiso
                        $dclase=null; $requiereJustificante=null;
                        ClasesDePermisos::f_dameDatosClasePermiso($solicitud['cclase'], $dclase,
                                                                  $requiereJustificante);

                        // Obtenemos el nombre y apellidos del interesado
                        $dnombre=null; $dapell1=null; $dapell2=null;
                        UsuariosDePersonal::f_dameNombreApellidosInteresado($solicitud['nregpgv'], $dnombre,
                                                                            $dapell1, $dapell2);

                        $mensaje = '';

                        //Limpiamos tags HTML del aviso para gastarlo en los posibles mensajes de correo
                        $avisoSinHtml = str_replace ("</LI>","\n",$solicitud['avisos']);
                        $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);

                        //Generar mensaje a enviar en el correo
                        Solicitudes::f_GenerarMensajeCorreo($solicitud['nregpgv'],
                          $dnombre . " " . $dapell1 . " " . $dapell2,
                          $solicitud['cclase'], $dclase, $solicitud['periodo'], $solicitud['fecha_ini'],
                          $solicitud['fecha_fin'], $solicitud['observaciones'], $solicitud['motivonoautper'],
                          $avisoSinHtml,"S","N","N","S","N","N","S", null, null,
                          $solicitud['nsolicitud'], $mensaje);

                        // NOTA : El mensaje deberá ser : "Se ha anulado la firma de resolución realizada
                        sobre el alta de la siguiente solicitud: ..."
                        //Enviar correo al usuario que realiza la solicitud
                        // El usuario se cargará de la siguiente manera:
                        // Seleccionaremos de la vista VPER_OCUPACION el DNI,nombre y apellidos de la
                        persona cuyo nregpgv coincida
                        // con el "nrp" del usuario conectado en la sesiÃ³n (en TCOM_USUARIOS).
                        $usuarioConectado = IgepSession::dameDatosUsuario();
                        $datosUsuario = $con->consultar('SELECT nregpgv FROM vper_ocupacion WHERE
                                                         nregpgv=\''.$usuarioConectado['nrp'].'\'');
                        $usuarioQueSolicita = $datosUsuario[0]['nregpgv'];
                        IgepDebug::setDebug(DEBUG_USER,'f_anularFirmaResolucion (solicitudes de alta) con
                                                        mensaje:<pre>'.$mensaje.'</pre>');


                        Solicitudes::f_enviarCorreo($usuarioQueSolicita,$solicitud['nregpgv'],
                                                    $mensaje,'PERSONAL - Firma de resoluciÃ³n anulada');
                }//Fin foreach
}
// Firma de solicitudes de anulación
if($tipoResolucion=='C'){
                $query = "
                SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, justificante,
                       motivonoautper, avisos, nsolanulacion, cclase
                FROM tper_solicitudes
                WHERE
                   fresolucionanula = '".$con->prepararFecha($fresolucion)."' and
                   ffirmaresolucionanula = '".$con->prepararFecha($ffirmaResolucion)."' and
                   situacion ='B' and
                   cestado ='AP' and
                   cclase in (".$var_cclasesCIT.") and
                   nregpgv in (select v.nregpgv from vper_ocupacion v where depuertos='".$dePuertos."')";
}
else{
                        $query = "
                SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, justificante,
                       motivonoautper, avisos, nsolanulacion, cclase
                FROM tper_solicitudes
                WHERE nsolicitud = $nsolicitud and
                   situacion ='B' and
                   cestado ='AP' and
                   cclase in (".$var_cclasesDGAA.")";

}
$solicitudesAnula = $con->consultar($query ,array(
                                'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
if(count($solicitudesAnula)>0){
                foreach($solicitudesAnula as $solicitudAnula){

                        //Grabamos la solicitud en la base de datos de personal

                        // Obtenemos la provincia y municipio del interesado
                        $cpro=null; $cmun=null;
                        UsuariosDePersonal::f_obtenerMunPro($solicitudAnula['nregpgv'], $cpro, $cmun);

                        // Obtenemos la descripción y si requiere justificante la clase de permiso
                        $dclase=null; $requiereJustificante=null;
                        ClasesDePermisos::f_dameDatosClasePermiso($solicitudAnula['cclase'], $dclase,
                                                                  $requiereJustificante);

                        //Limpiamos tags HTML del aviso para gastarlo en los posibles mensajes de correo
                        $avisoSinHtml = str_replace ("</LI>","\n",$solicitudAnula['avisos']);
                        $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);


                        // Obtenemos el nombre y apellidos del interesado
                        $dnombre=null; $dapell1=null; $dapell2=null;
```

```
UsuariosDePersonal::f_dameNombreApellidosInteresado($solicitudAnula['nregpgv'],
                                                    $dnombre, $dapell1, $dapell2);

// Obtenemos el valor de diasTrabajados para pasarlo en
f_generarPermiso(...,diasTrabajados)
$diasTrabajados=null;
$res = SolicitudesYPermisos::f_validarSoliPer($solicitudAnula['periodo'],
                    $solicitudAnula['fecha_ini'],$solicitudAnula['fecha_fin'],
                    $solicitudAnula['cclase'],$resultado,$diasTrabajados,$nulo,
                    $nulo,'S',$nulo,$nulo,$nulo,$nulo,$nulo,
                    $solicitudAnula['nregpgv'],$solicitudAnula['cclase']);

if($res<0){
          IgepDebug::setDebug(DEBUG_USER,'FIN f_anularFirmaResolucion tras error en
                                                          f_validarSoliPer');
          return array('0'=>-1);
}
// Buscar altas en tper_solicitudes con mismo nsolanulacion y además:
// - con fecha de inicio del alta igual a la fecha de fin de la anulación actual +1 o
// - con fecha de fin del alta igual a la fecha de inicio de la anulación - 1
// De esta manera solo recuperamos las altas que se encuentran junto a la anulación
actual
//solicitudAnula.fecha_fin + 1;
$fecha1 = clone $solicitudAnula['fecha_fin'];
$fecha1->addDays(1);
//solicitudAnula.fecha_ini - 1;
$fecha2 = clone $solicitudAnula['fecha_ini'];
$fecha2->subDays(1);

if($solicitudAnula['nsolanulacion'] == '')
          $solicitudesAlta = null;
else
{
          $query = "
         SELECT fecha_ini, fecha_fin, nsolicitud FROM TPER_SOLICITUDES
         WHERE nsolanulacion ".$solicitudAnula['nsolanulacion'] ." and
             (fecha_ini = '".$fecha1."' or fecha_fin = '".$fecha2."')";

          $solicitudesAlta = $con->consultar($query ,array(
          'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
}


if(count($solicitudesAlta[0])==0){
// No es una anulación parcial. A la anulación solo le corresponde un permiso
          $res = Permisos::f_generarPermiso($con,$conOra,$solicitudAnula['nregpgv'],
                    $cpro,$cmun, $solicitudAnula['cclase'],
                    $solicitudAnula['fecha_ini'], $solicitudAnula['fecha_fin'],
                    $solicitudAnula['periodo'], $dnombre.' '.$dapell1.' '.$dapell2,
                    $solicitudAnula['justificante'],$solicitudAnula['observaciones'],
                    $solicitudAnula['nsolicitud'],
                    $solicitudAnula['fnacimientomenor'],$diasTrabajados);
          if($res['0']==-1){
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_anularFirmaResolucion tras
                                                    error en f_generarPermiso(1)');
                    return $res;
          }
          if(!empty($res['errorAviso'])) {
                    $str_avisos.=$res['errorAviso'] . ' ';
          }
}
else{

          // Recorremos las altas (1 o 2) que se encuentran justo antes o después de
          la anulación actual
          foreach($solicitudesAlta as $solicitudAlta){
                    // Obtenemos las fechas del permiso del alta antes de borrarlo
            $query = "
               SELECT fecha_ini, fecha_fin, nsolicitud
               FROM tper_permisos
               WHERE nregpgv = '".$solicitudAnula['nregpgv']."' and
                  cclase = '".$solicitudAnula['cclase']."' and
                  (fecha_ini = solicitudAlta.fecha_ini or fecha_fin =
                    solicitudAlta.fecha_fin)";
            $res = $con->consultar($query,array(
                              'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>
                                                                  TIPO_FECHA,)));
            $fechaIniPer = clone $res[0]['fecha_ini'];
            $fechaFinPer= clone $res[0]['fecha_fin'];
            $nSolicitudPer = $res[0]['nsolicitud'];


            //Borramos de la tabla de personal TPER_PERMISOS el permiso cuya solicitud
            de anulación ha sido autorizada
            $res = Permisos::f_borrarPermiso($con, $conOra,$solicitudAnula['nregpgv'],
                    $solicitudAnula['cclase'], $fechaIniPer,$var_avisoNominas);


            if($res==-1){
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_anularFirmaResolucion
                    después de error en borrar de tper_permisos(1)');
                    return array('0'=>-1);
            }

            if ($var_avisoNominas){
                    $solicitudesBorradas = $solicitudesBorradas."DNI:
                    ".substr($solicitudAnula['nregpgv'],1,8)."; motivo:
                    ".$solicitudAnula['cclase']."; fecha inicio: ".$fechaIniPer."\n";
            }
```

```
  // Obtenemos la fecha de inicio del permiso de la anulacin: Puede haberse
  generado ya el permiso de
  // la anulación actual, y también se ha de borrar
  $query = "
     SELECT fecha_ini, nsolicitud
     FROM tper_permisos
     WHERE nregpgv = '".$solicitudAnula['nregpgv']."' and
        cclase = '".$solicitudAnula['cclase']."' and
        (fecha_ini = solicitudAnula.fecha_ini or fecha_fin =
         solicitudAnula.fecha_fin)";
  $res = $con->consultar($query,array(
                                'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,)));
 if(count($res[0])>0){
          $fechaIniPerAnula = clone $res[0]['fecha_ini'];
                    $nSolicitudPer = $res[0]['nsolicitud'];
                    //Borramos el permiso del alta de la base de datos

                    //Borramos de la tabla de personal TPER_PERMISOS el
                    permiso cuya solicitud de anulación ha sido autorizada
                    $res = Permisos::f_borrarPermiso($con, $conOra,
                              $solicitudAnula['nregpgv'],
                              $solicitudAnula['cclase'],
                              $fechaIniPerAnula,$var_avisoNominas);


                    if($res==-1){
                       IgepDebug::setDebug(DEBUG_USER,'FIN f_anularResolucion
                              después de borrar de tper_permisos(2)');
                              return array('0'=>-1);
                    }

                    if ($var_avisoNominas){
                              $solicitudesBorradas =
                              $solicitudesBorradas."DNI:
                              ".substr($solicitudAnula['nregpgv'],1,8).";
                              motivo: ".$solicitudAnula['cclase']."; fecha
                              inicio: ".$fechaIniPerAnula."\n";
                    }

                    $solicitudAnula['nsolicitud'] = $nSolicitudPer;
}
// Se toma el mínimo de los números de solicitud del alta y la anulación
para darselo al permiso.
// De esta manera, el permiso acabará teniendo el valor de nsolicitud comÃºn
de la anulación parcial,
// ya que este valor es el mínimo de todos los nsolicitud

if($solicitudAnula['nsolicitud']<$nSolicitudPer)
          $nsolMin = $solicitudAnula['nsolicitud'];
else
          $nsolMin = $nSolicitudPer;

//solicitudAlta.fecha_ini = solicitudAnula.fecha_fin + 1
$fechaFinMasUno= clone $solicitudAnula['fecha_fin'];
if(gvHidraTimestamp::cmp($solicitudAlta['fecha_ini'],$fechaFinMasUno)==0){
// Alta que se encuentra justo despuÃ©s de la anulación
          $res = Permisos::f_generarPermiso($con,$conOra,
                              $solicitudAnula['nregpgv'],
                              $cpro,$cmun, $solicitudAnula['cclase'],
                              $solicitudAnula['fecha_ini'],$fechaFinPer,
                              $solicitudAnula['periodo'], $dnombre.'
                              '.$dapell1.' '.$dapell2,
                              $solicitudAnula['justificante'],
                              $solicitudAnula['observaciones'],
                              $nsolMin,
                              $solicitudAnula['fnacimientomenor'],
                              $diasTrabajados);
          if($res['0']==-1){
                    IgepDebug::setDebug(DEBUG_USER,'FIN
                              f_anularFirmaResolucion tras error en
                                        f_generarPermiso(2)');
                              return $res;
          }
          if(!empty($res['errorAviso'])) {
                              $str_avisos.=$res['errorAviso'] . ' ';
          }

          $solicitudAnula['fecha_fin'] = clone $fechaFinPer;
}
else{
          // solicitudAlta.fecha_fin = solicitudAnula.fecha_ini - 1
          // Alta que se encuentra justo antes de la anulación

          $res = Permisos::f_generarPermiso($con,$conOra,
                    $solicitudAnula['nregpgv'], $cpro,
                    $cmun, $solicitudAnula['cclase'], $fechaIniPer,
                    $solicitudAnula['fecha_fin'], $solicitudAnula['periodo'],
                    $dnombre.' '.$dapell1.' '.$dapell2,
                    $solicitudAnula['justificante'],
                    $solicitudAnula['observaciones'],
                    $nsolMin,$solicitudAnula['fnacimientomenor'],
                     $diasTrabajados);
          if($res['0']==-1){
                    IgepDebug::setDebug(DEBUG_USER,'FIN
                                        f_anularFirmaResolucion tras error en
                                        f_generarPermiso(3)');
                    return $res;
          }
          if(!empty($res['errorAviso'])) {
```

```
                                                $str_avisos.=$res['errorAviso'] . ' ';
                                        }
                                        $solicitudAnula['fecha_ini'] = $fechaIniPer;

                                }//else
                                }//Fin foreach
                        }//else
                        $mensaje = "";

                        //Generar mensaje a enviar en el correo
                        Solicitudes::f_GenerarMensajeCorreo($solicitudAnula['nregpgv'],
                                        $dnombre . " " . $dapell1 . " " . $dapell2,
                                        $solicitudAnula['cclase'], $dclase, $solicitudAnula['periodo'],
                                        $solicitudAnula['fecha_ini'], $solicitudAnula['fecha_fin'],
                                        $solicitudAnula['observaciones'],
                                        $solicitudAnula['motivonoautper'], $avisoSinHtml,
                                        "N","N","N","S","N","N","S", null,
                                        null,$solicitudAnula['nsolicitud'],$mensaje);


                        //Enviar correo al usuario que realiza la solicitud


                        $usuarioConectado = IgepSession::dameDatosUsuario();
                        $datosUsuario = $con->consultar('SELECT nregpgv FROM vper_ocupacion WHERE
                                                nregpgv=\''.$usuarioConectado['nrp'].'\'');
                        $usuarioQueSolicita = $datosUsuario[0]['nregpgv'];
                        IgepDebug::setDebug(DEBUG_USER,'f_anularFirmaResolucion (solicitudes de
                                        anulación) con mensaje:<pre>'.$mensaje.'</pre>');

                        Solicitudes::f_enviarCorreo($usuarioQueSolicita,$solicitudAnula['nregpgv'],
                                                $mensaje,'PERSONAL - Firma de resoluciÃ³n anulada');
                }//Fin foreach
        }// if
}//  if (count($solicitudes[0])>0) {
else{
        // Resoluciones negativas

        // Firma de solicitudes de alta
        $query = "
              SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, motivonoautper, avisos
              FROM tper_solicitudes
              WHERE
                 nsolicitud = $nsolicitud and
                 situacion ='A' and
                 cestado ='NP'";
        $solicitudes = $con->consultar($query,array(
                                'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
        if(count($solicitudes[0])>0){
                 foreach($solicitudes as $solicitud){

                        // Obtenemos la descripciÃ³n y si requiere justificante la clase de permiso
                        $dclase=null; $requiereJustificante=null;
                        ClasesDePermisos::f_dameDatosClasePermiso($solicitud['cclase'], $dclase,
                                                                $requiereJustificante);

                        // Obtenemos el nombre y apellidos del interesado
                        $dnombre=null; $dapell1=null; $dapell2=null;
                        UsuariosDePersonal::f_dameNombreApellidosInteresado($solicitud['nregpgv'], $dnombre,
                                                                $dapell1, $dapell2);

                        //Limpiamos tags HTML del aviso para gastarlo en los posibles mensajes de correo
                        $avisoSinHtml = str_replace ("</LI>","\n",$solicitud['avisos']);
                        $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
                        $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);

                        //Enviar correo al usuario que realiza la solicitud
                        $mensaje = "";

                        //Generar mensaje a enviar en el correo
                        Solicitudes::f_GenerarMensajeCorreo($solicitud['nregpgv'],
                                          $dnombre . " " . $dapell1 . " " . $dapell2,
                                          $solicitud['cclase'], $dclase, $solicitud['periodo'],
                                          $solicitud['fecha_ini'], $solicitud['fecha_fin'],
                                          $solicitud['observaciones'],$solicitud['motivonoautper'],
                                          $avisoSinHtml,"S","N","N","S","N","N","S", null,
                                          null,$solicitud['nsolicitud'],$mensaje);


                        //Enviar correo al usuario que realiza la solicitud
                        // El usuario se cargará de la siguiente manera:
                        // Seleccionaremos de la vista VPER_OCUPACION el DNI,nombre y apellidos de la
                        persona cuyo nregpgv coincida
                        // con el "nrp" del usuario conectado en la sesiÃ³n (en TCOM_USUARIOS).
                        $usuarioConectado = IgepSession::dameDatosUsuario();
                        $datosUsuario = $con->consultar('SELECT nregpgv FROM vper_ocupacion WHERE
                                                nregpgv=\''.$usuarioConectado['nrp'].'\'');
                        $usuarioQueSolicita = $datosUsuario[0]['nregpgv'];
                        Solicitudes::f_enviarCorreo($usuarioQueSolicita,
                                                $solicitud['nregpgv'],
                                                $mensaje,'PERSONAL - Firma de resoluciÃ³n anulada');
                 }
        }
        // Firma de solicitudes de anulación
        $query = "
              SELECT nsolicitud, periodo, fecha_ini, fecha_fin, nregpgv, observaciones, motivonoautper, avisos
              FROM tper_solicitudes
              WHERE
                 nsolicitud = $nsolicitud and
```

```
                        situacion ='A' and
                        cestado ='AP' and
                        fanulaper is not null";
                $solicitudesAnula = $con->consultar($query,array(
                                        'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
                if(count($solicitudesAnula[0])>0){
                        foreach($solicitudesAnula as $solicitudAnula){

                                // Obtenemos la descripción y si requiere justificante la clase de permiso
                                $dclase=null; $requiereJustificante=null;
                                ClasesDePermisos::f_dameDatosClasePermiso($solicitudAnula['cclase'], $dclase,

                                                                        $requiereJustificante);

                                // Obtenemos el nombre y apellidos del interesado
                                $dnombre=null; $dapell1=null; $dapell2=null;

                                UsuariosDePersonal::f_dameNombreApellidosInteresado($solicitudAnula['nregpgv'],
                                                        $dnombre, $dapell1, $dapell2);

                                //Limpiamos tags HTML del aviso para gastarlo en los posibles mensajes de correo
                                $avisoSinHtml = str_replace ("</LI>","\n",$solicitudAnula['avisos']);
                                $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
                                $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
                                $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
                                $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
                                $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);

                                //Enviar correo al usuario que realiza la solicitud

                                $mensaje = '';

                                //Generar mensaje a enviar en el correo
                                Solicitudes::f_GenerarMensajeCorreo($solicitudAnula['nregpgv'],
                                  $dnombre . " " . $dapell1 . " " . $dapell2,
                                  $solicitudAnula['cclase'], $dclase, $solicitudAnula['periodo'],
                                  $solicitudAnula['fecha_ini'], $solicitudAnula['fecha_fin'],
                                  $solicitudAnula['observaciones'], $solicitudAnula['motivonoautper'], $avisoSinHtml,
                                  "N","N","N","S","N","N","S", null, null,$solicitudAnula['nsolicitud'],$mensaje);

                                //Enviar correo al usuario que realiza la solicitud
                                // El usuario se cargará de la siguiente manera:
                                // Seleccionaremos de la vista VPER_OCUPACION el DNI,nombre y apellidos de la
                                persona cuyo nregpgv coincida
                                // con el "nrp" del usuario conectado en la sesión (en TCOM_USUARIOS).
                                $usuarioConectado = IgepSession::dameDatosUsuario();
                                $datosUsuario = $con->consultar('SELECT nregpgv FROM vper_ocupacion WHERE
                                                                nregpgv=\''.$usuarioConectado['nrp'].'\'');
                                $usuarioQueSolicita = $datosUsuario[0]['nregpgv'];

                                Solicitudes::f_enviarCorreo($usuarioQueSolicita,$solicitudAnula['nregpgv'],
                                                        $mensaje,'PERSONAL - Firma de resoluciÃ³n anulada');
                        }//foreach
                }//if
        }//else

        //Aviso a nominas
        if($solicitudesBorradas!='')
                $str_avisos.="Los siguientes permisos ya borrados, tienen creadas incidencias para la aplicación de
                        nóminas. Avisar a nóminas:\n".$solicitudesBorradas;

        IgepDebug::setDebug(DEBUG_USER,'FIN f_anularFirmaResolucion con true');

        $result['0'] = 0;
        $result['errorAviso'] = $str_avisos;

        return $result;
} //Fin f_anularResolucion



public function f_notificarFirmaResolucion($pnsolicitud,$ppffirma,$positiva){
        //Conectamos a Postgresql
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_notificarFirmaResolucion');
        IgepDebug::setDebug(DEBUG_USER,'pnsolicitud '.print_r($pnsolicitud,true));
        $pffirma = clone $ppffirma;

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);

        $listaNumSol = array();
        foreach($pnsolicitud as $solicitud) {
                array_push($listaNumSol,$solicitud['nsolicitud']);
        }
        reset($pnsolicitud);

        if($positiva=='S'){

                //Se deben enviar notificaciones por correo a cada persona para las que se ha firmado las solicitudes
                // de la resolución positiva
                //Primero obtenemos la lista de personas afectadas
                $query = "
                      SELECT DISTINCT nregpgv
                      FROM TPER_SOLICITUDES
                      WHERE nsolicitud in (".implode(",", $listaNumSol).")";
                $var_personas_resolucion = $con->consultar($query);
                foreach($var_personas_resolucion as $var_persona_resolucion){

                        // Obtenemos el nombre y apellidos del interesado

                        $nsolicitud_persona = array();
```

```
                        foreach($pnsolicitud as $solicitud){
                                if($solicitud['nregpgv']==$var_persona_resolucion['nregpgv']){
                                        // Obtenemos la descripciÃ³n y si requiere justificante la clase de permiso
                                        $dclase=null; $requiereJustificante=null;
                                        ClasesDePermisos::f_dameDatosClasePermiso($solicitud['cclase'], $dclase,
                                                                                $requiereJustificante);


                                        if(($solicitud['situacion'] == 'A' and ($solicitud['cestado'] != 'AP' or
                                            is_null($solicitud['fanulaper']))or($solicitud['situacion'] == 'B' and
                                            is_null($solicitud['ffirmaresolucion'])))

                                                        $nueva['motivo'] = $dclase;
                                        else
                                                        $nueva['motivo'] = 'ANULA '.$dclase;

                                        $nueva['fecha_ini'] = clone $solicitud['fecha_ini'];
                                        $nueva['fecha_fin'] = clone $solicitud['fecha_fin'];;
                                        array_push($nsolicitud_persona,$nueva);
                                }
                        }
                        //Componer el mensaje de correo con los datos obtenidos
                        $var_mensaje="";
                        Solicitudes::f_generarMensajeNotificacion($nsolicitud_persona, $pffirma, 'S', $var_mensaje);
                        //Enviamos el correo
                        // El usuario se cargará de la siguiente manera:
                        // Seleccionaremos de la vista VPER_OCUPACION el DNI,nombre y apellidos de la   persona cuyo
                        nregpgv coincida
                        // con el "nrp" del usuario conectado en la sesión (en TCOM_USUARIOS).
                        $usuarioConectado = IgepSession::dameDatosUsuario();
                        $datosUsuario = $con->consultar('SELECT nregpgv FROM vper_ocupacion
                                                WHERE nregpgv=\''.$usuarioConectado['nrp'].'\'');
                                                        $usuarioQueSolicita = $datosUsuario[0]['nregpgv'];



                        Solicitudes::f_enviarCorreo($usuarioQueSolicita,$var_persona_resolucion['nregpgv'],
                                                        $var_mensaje,'PERSONAL - Resolución positiva');
                }
        }
        else{

                //Se deben enviar notificaciones por correo a cada persona para las que se ha firmado la solicitud
                 // de la resolución negativa

                 //Primero obtenemos la lista de personas afectadas
                $query = "
                      SELECT DISTINCT nregpgv
                      FROM TPER_SOLICITUDES
                      WHERE nsolicitud in (".implode(",", $listaNumSol).")";
                $var_personas_resolucion = $con->consultar($query);
                if(count($var_personas_resolucion[0])>0){
                        foreach($var_personas_resolucion as $var_persona_resolucion){

                                // Obtenemos el nombre y apellidos del interesado
                                $nsolicitud_persona = array();
                                foreach($pnsolicitud as $solicitud){
                                        if($solicitud['nregpgv']==$var_persona_resolucion['nregpgv']){
                                                // Obtenemos la descripción y si requiere justificante la clase
                                                de permiso
                                                $dclase=null; $requiereJustificante=null;
                                                ClasesDePermisos::f_dameDatosClasePermiso(
                                                        $solicitud['cclase'], $dclase, $requiereJustificante);


                                                $nueva['motivo'] = $dclase;
                                                $nueva['fecha_ini'] = clone $solicitud['fecha_ini'];
                                                $nueva['fecha_fin'] = clone $solicitud['fecha_fin'];
                                                //Obtener el motivo de no autorización
                                                $query = "SELECT motivonoautresp, motivonoautper
                                                                FROM TPER_SOLICITUDES
                                                                WHERE nsolicitud = ".$solicitud['nsolicitud'];
                                                $motivoNoAut = $con->consultar($query);
                                                $nueva['motivonoautresp'] = $motivoNoAut[0]['motivonoautresp'];
                                                $nueva['motivonoautper'] = $motivoNoAut[0]['motivonoautper'];
                                                array_push($nsolicitud_persona,$nueva);
                                        }
                                }

                                //Componer el mensaje de correo con los datos obtenidos
                                $var_mensaje="";
                                Solicitudes::f_generarMensajeNotificacion($nsolicitud_persona,$pffirma,'N',
                                                                                        $var_mensaje);
                                //Enviamos el correo
                                // El usuario se cargará de la siguiente manera:
                                // Seleccionaremos de la vista VPER_OCUPACION el DNI,nombre y apellidos de la
                                persona cuyo nregpgv coincida
                                // con el "nrp" del usuario conectado en la sesión (en TCOM_USUARIOS).
                                $usuarioConectado = IgepSession::dameDatosUsuario();
                                $datosUsuario = $con->consultar('SELECT nregpgv FROM vper_ocupacion WHERE
                                                        nregpgv=\''.$usuarioConectado['nrp'].'\'');
                                $usuarioQueSolicita = $datosUsuario[0]['nregpgv'];

                                Solicitudes::f_enviarCorreo($usuarioQueSolicita,$var_persona_resolucion['nregpgv'],
                                                        $var_mensaje,'PERSONAL - ResoluciÃ³n negativa');
                        }
                }
        }
        IgepDebug::setDebug(DEBUG_USER,'FIN f_notificarFirmaResolucion');
        return true;
}
```

```
public static function f_conformeAnulacion($psolicitud,$pnregpvSolicitante,&$plistaSolicitudesAsociadas,
                                            &$plistaSolicitudAnulacion,&$conPostgres,$proceso){
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_conformeAnulacion');

        //Limpiamos tags HTML del aviso para gastarlo en los posibles mensajes de correo
        $avisoSinHtml = str_replace ("</LI>","\n",$psolicitud['avisos']);
        $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
        $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
        $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
        $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
        $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);

        //Inicializamos variables
        $listaSolicitudes = null;
        $error=0;
        $mensaje = '';

        // 09/01/2009 INICIO : Movemos del final de la función al principio el cálculo de plistaSolicitudesAsociadas
        // y plistaSolicitudAnulacion. Si no lo hiciesemos así, en varios casos se quedarían vacías
        // al haber borrado previamente en tper_solicitudesmodificadas

        // Si se autoriza la firma de solicitudes de anulación, comprobamos si están asociadas
        // con otras solicitudes de alta o anulación, en cuyo caso se genera una lista de solicitudes
        // que se pasa al diseño para mostrarla en un mensaje

        if($psolicitud['situacion']=='X' or $psolicitud['situacion']=='B'){
                SolicitudesModificadas::f_dameSolicitudesQueModifican($psolicitud['nsolicitud'],'N',
                                                                      $plistaSolicitudesAsociadas);
                if(count($plistaSolicitudesAsociadas[0])>0){
                        array_push($plistaSolicitudAnulacion,$psolicitud['nsolicitud']);
                }
        }


        if($psolicitud['situacion']=='X'){
                if($psolicitud['cestado']=='PR'){
                        //Generar mensaje a enviar en el correo
                        Solicitudes::f_GenerarMensajeCorreo($psolicitud['nregpgv'],$psolicitud['persona'],
                        $psolicitud['cclase'],$psolicitud['dclase'],$psolicitud['periodo'],$psolicitud['fecha_ini'],
                        $psolicitud['fecha_fin'],$psolicitud['observaciones'], $psolicitud['motivonoautresp'],
                        $avisoSinHtml,"N","S","S","S","N","N","N", null, null,$psolicitud['nsolicitud'],$mensaje);

                        //Enviar correo al usuario que realiza la solicitud
                        Solicitudes::f_enviarCorreo($pnregpvSolicitante,$psolicitud['nregpgv'],$mensaje,
                                        'PERSONAL - Solicitud de anulaciÃ³n conforme por responsable');

                        // Movemos el borrado de tper_solicitudes después del de tper_solicitudesmodificadas:
                        // Si no se hiciera así, daría error por existencia de clave foranea entre las dos tablas

                        // Si la solicitud de anulación borrada tenía altas asociadas, también se borran
                        // Borramos las asociaciones de la solicitud a borrar por nsolicitud y por nsolicitudmodifica
                        // directamente, ya que cuesta igual lanzar el delete que un select para ver si hay


                        $res = $conPostgres->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS
                                                     WHERE nsolicitudmodifica = ".$psolicitud['nsolicitud']);
                        $res1 = $conPostgres->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS
                                                      WHERE nsolicitud = ".$psolicitud['nsolicitud']);



                        if(($res==-1) or ($res1==-1))
                                $error=1;
                        else{
                                $res = $conPostgres->operar("DELETE FROM TPER_SOLICITUDES
                                                             WHERE nsolicitud = ".$psolicitud['nsolicitud']);

                                if($res==-1)
                                        $error=1;
                        }


                         if($error!=1) {
                                //Auditoria del borrado, insertando en TPER_SOLICITUDESNORRADAS
                                SolicitudesBorradas::f_altaSolicitudesBorradas($psolicitud['nsolicitud'],
                                        $psolicitud['situacion'],$psolicitud['nregpgv'],$psolicitud['cclase'],
                                        $psolicitud['fecha_ini'],$psolicitud['fecha_fin'],$proceso,$conPostgres);

                                if ($psolicitud['cclase']=='VV6' or $psolicitud['cclase']=='AAP'){
                                        //Recalculamos el ordinal de los días solicitados de VV6 o AAP por un
                                        interesado en un determinado periodo,
                                        //al haber anulado un VV6 o AAP
                                        SolicitudesYPermisos::f_recalcularOrdinalBorrado($psolicitud['nregpgv'],
                                                        $psolicitud['periodo'],$psolicitud['cclase'] );

                                }
                        }


                }//Fin de si el estado es PR
        }//Fin de situacion=='X'

        if($psolicitud['situacion']=='B'){
                if($psolicitud['cestado']=='PR'){
                        if(empty($psolicitud['ffirmaresolucion'])){
                                //Enviar correo al usuario que realiza la solicitud para indicar que se aprueba la
                                anulación
                                //Generar mensaje a enviar en el correo
```

```
                                        Solicitudes::f_GenerarMensajeCorreo($psolicitud['nregpgv'],    $psolicitud['persona'],
                                                $psolicitud['cclase'], $psolicitud['dclase'], $psolicitud['periodo'],
                                                $psolicitud['fecha_ini'], $psolicitud['fecha_fin'],
                                                $psolicitud['observaciones'], $psolicitud['motivonoautresp'],
                                                $avisoSinHtml,"N","S","S","S","N","N","N", null,
                                                null,$psolicitud['nsolicitud'],$mensaje);

                                        //Enviar correo al usuario que realiza la solicitud
                                        Solicitudes::f_enviarCorreo($pnregpvSolicitante,$psolicitud['nregpgv'],
                                                $mensaje,'PERSONAL - Solicitud de anulaciÃ³n conforme por responsable');

                                        // Movemos el borrado de tper_solicitudes después del de tper_solicitudesmodificadas:
                                        // Si no se hiciera así, daría error por existencia de clave foranea entre las dos
                                        tablas
                                        //Si la solicitud de anulación borrada tenía altas asociadas, también se borran
                                        //Borramos las asociaciones de la solicitud a borrar por nsolicitud y por
                                        nsolicitudmodifica
                                        //directamente, ya que cuesta igual lanzar el delete que un select para ver si hay


                                        $res = $conPostgres->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS
                                                        WHERE nsolicitudmodifica = ".$psolicitud['nsolicitud']);
                                        $res1 = $conPostgres->operar("DELETE FROM TPER_SOLICITUDESMODIFICADAS
                                                        WHERE nsolicitud = ".$psolicitud['nsolicitud']);


                                        if(($res==-1) or ($res1==-1))
                                                $error=1;
                                        else{
                                                // Podemos borrar la solicitud, sería equivalente a romper el papel
                                                $res = $conPostgres->operar("DELETE FROM TPER_SOLICITUDES
                                                                        WHERE nsolicitud = ".$psolicitud['nsolicitud']);
                                                if($res==-1)
                                                        $error=1;
                                        }


                                        if($error!=1) {
                                                // Auditoria del borrado, insertando en TPER_SOLICITUDESNORRADAS
                                                SolicitudesBorradas::f_altaSolicitudesBorradas($psolicitud['nsolicitud'],
                                                  $psolicitud['situacion'],$psolicitud['nregpgv'],$psolicitud['cclase'],
                                                  $psolicitud['fecha_ini'],$psolicitud['fecha_fin'],$proceso,$conPostgres);


                                                if ($psolicitud['cclase']=='VV6' or $psolicitud['cclase']=='AAP'){
                                                        //Recalculamos el ordinal de los días solicitados de VV6 o AAP por
                                                        un interesado en un determinado periodo,
                                                        //al haber anulado un VV6 o AAP
                                                        SolicitudesYPermisos::f_recalcularOrdinalBorrado(
                                                                $psolicitud['nregpgv'],$psolicitud['periodo'],
                                                                $psolicitud['cclase'] );

                                                }

                                        }
                                }
                                else{
                                        $res = $conPostgres->operar(" UPDATE TPER_SOLICITUDES
                                                SET   fanularesp = current_date, nrpanularesp =  '".$pnregpvSolicitante."',
                                                        cestado = 'PT', validaanula = null
                                                WHERE nsolicitud = ".$psolicitud['nsolicitud']);
                                }
                        }//Fin estado = PR
                        if($psolicitud['cestado']=='PP' or $psolicitud['cestado']=='PN'){
                                $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                                         SET   fanulaper = current_date, nrpanulaper = '".$pnregpvSolicitante."',cestado='AP'
                                         WHERE nsolicitud =".$psolicitud['nsolicitud']);
                                if($res==-1)
                                        $error=1;
                         }//Fin cestado==PP o PN
                } //Fin de situacion=='B'

                IgepDebug::setDebug(DEBUG_USER,'FIN f_conformeAnulacion');
                return $error;
}



public static function f_noConformeAlta($psolicitud,$pmotivonoaut,$pnregpgvSolicitante,&$conPostgres){

                //Limpiamos tags HTML del aviso para gastarlo en los posibles mensajes de correo
                $avisoSinHtml = str_replace ("</LI>","\n",$psolicitud['avisos']);
                $avisoSinHtml = str_replace ("<UL>","",$avisoSinHtml);
                $avisoSinHtml = str_replace ("</UL>","",$avisoSinHtml);
                $avisoSinHtml = str_replace ("</LI>","\n",$avisoSinHtml);
                $avisoSinHtml = str_replace ("<LI>","",$avisoSinHtml);
                $avisoSinHtml = str_replace ("<br>","",$avisoSinHtml);

                //Inicializamos valores
                $error=0;
                $mensaje = '';

                //Aqui empieza el pseudocodigo
                if($psolicitud['cestado']=='PR' or $psolicitud['cestado']=='PC'){

                        //Si la solicitud no nos viene completa buscamos
                        $res = $conPostgres->consultar("SELECT cestado
                                                FROM TPER_SOLICITUDES
                                                WHERE nsolicitud =".$psolicitud['nsolicitud']);
                        if($res==-1)
                                return 1;
```

```
                $estadoActual= $res[0]['cestado'];
                if($estadoActual!='NR'){
                        $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                                        SET   ffirmaresp = current_date, nrpresp = '".$pnregpgvSolicitante."',
                                        motivonoautresp = 'Afecta a las necesidades del servicio', cestado = 'NR'
                                        WHERE nsolicitud = ".$psolicitud['nsolicitud']);

                        if($res!=-1){

                                //Enviar correo al usuario que realiza la solicitud
                                //Generar mensaje a enviar en el correo
                                Solicitudes::f_GenerarMensajeCorreo($psolicitud['nregpgv'],
                                        $psolicitud['persona'],$psolicitud['cclase'], $psolicitud['dclase'],
                                        $psolicitud['periodo'], $psolicitud['fecha_ini'], $psolicitud['fecha_fin'],
                                        $psolicitud['observaciones'], 'Afecta a las necesidades del servicio',
                                        $avisoSinHtml,"S","S","N","S","N","N","N", null,
                                        null,$psolicitud['nsolicitud'],$mensaje);

                                //Enviar correo al usuario que realiza la solicitud
                                Solicitudes::f_enviarCorreo($pnregpgvSolicitante,$psolicitud['nregpgv'],
                                                $mensaje,'PERSONAL - Solicitud no conforme por responsable');
                        }
                        else
                                $error=1;
                }
        }
        if($psolicitud['cestado']=='PP' OR $psolicitud['cestado']=='PN'){
                $res = $conPostgres->operar("UPDATE TPER_SOLICITUDES
                                        SET   ffirmaper = current_date, nrpper = '".$pnregpgvSolicitante."',
                                            motivonoautper = '".$pmotivonoaut."', cestado = 'NP'
                                        WHERE nsolicitud = ".$psolicitud['nsolicitud']);
        }
        if($res==-1)
            $error=1;
        return $error;
}


public static function f_dameSolicitudesPendientes() {

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameSolicitudesPendientes');

          $perfil = IgepSession::dameRol();

          //Conexion
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);

          $response = array();


          if ($perfil=='P_TRAMITA') {

                    // Obtener solicitudes pendientes de validar
                    $res = $con->consultar("SELECT count(1) as \"pValidar\"
                                                 FROM TPER_SOLICITUDES s
                                                 WHERE    s.cestado IN ('PT','NI') and
                                                s.nsolicitud not in
                                                (select s2.nsolicitud from tper_solicitudesmodificadas s2,
                                                tper_solicitudes an
                                                where an.nsolicitud = s2.nsolicitudmodifica and
                                                 an.cestado in ('PT','NI','PR','PC') and
                                                 an.situacion in ('B','X') and
                                                 s2.anulacionresuelta='N'
                                                )");

                    if(isset($res[0]['pValidar']))
                              $pValidar = $res[0]['pValidar'];
                    else
                              $pValidar = 0;


                    // Obtener solicitudes pendientes de generar resolución para la firma del subsecretario:
                    $res = $con->consultar("select count(1) as \"pResolucionCITPos\" from
                                                                                (
                                                                      select s.nsolicitud
                                                                      from TPER_SOLICITUDES s
                                                                      where    s.situacion ='A' and
                                                            s.cestado = 'AP' and
                                                            s.fresolucion is null and
                                                            s.cclase not in ('LIP','LEF','LPP','EFP','AIN')

                                                                      UNION

                                                                                select s.nsolicitud
                                                                                from TPER_SOLICITUDES s
                                                                      where s.situacion ='B' and
                                                            s.cestado ='AP' and
                                                            s.fanulaper is not null and
                                                            s.fresolucionanula is null and
                                                            s.cclase not in ('LIP','LEF','LPP','EFP','AIN') and
                                                                      s.ffirmaresolucion is not null
                                                                                ) a");

                    if(isset($res[0]['pResolucionCITPos']))
                              $pResolucionCITPos = $res[0]['pResolucionCITPos'];
                    else
                              $pResolucionCITPos = 0;

                    $res = $con->consultar("select count(1) as \"pResolucionCITNeg\" from
```

```
                                                                (
                                                        select s.nsolicitud
                                                        from TPER_SOLICITUDES s
                                                        where s.situacion ='A' and
                                                s.cestado ='NP' and
                                                s.fresolucion is null and
                                                s.cclase not in ('LIP','LEF','LPP','EFP','AIN')

                                                        UNION

                                                        select s.nsolicitud
                                                        from TPER_SOLICITUDES s
                                                                where s.situacion ='A' and
                                                        s.cestado = 'AP' and
                                                s.ffirmaresolucion is not null and
                                                        s.fanulaper is not null and
                                                                s.fresolucionanula is null and
                                                s.cclase not in ('LIP','LEF','LPP','EFP','AIN')
                                                        ) a");

if(isset($res[0]['pResolucionCITNeg']))
        $pResolucionCITNeg = $res[0]['pResolucionCITNeg'];
else
        $pResolucionCITNeg = 0;


// Obtener solicitudes pendientes de generar resoluciÃ³n DGAA
$res = $con->consultar("select count(1) as \"pResolucionDGAAPos\" from
                                                                (
                                                                 select s.nsolicitud
                                                                 from TPER_SOLICITUDES s
                                                        where s.situacion ='A' and
                                                        s.cestado = 'AP' and
                                                s.fanulaper is null and
                                                s.fresolucion is null and
                                                s.cclase in ('LIP','LEF','LPP','EFP','AIN')
                                                        UNION

                                                            select s.nsolicitud
                                                        from TPER_SOLICITUDES s
                                                                where s.situacion ='B' and
                                                s.cestado <> 'AP' and
                                                s.fresolucion is null and
                                                s.cclase in ('LIP','LEF','LPP','EFP','AIN')
                                                        UNION

                                                        select s.nsolicitud
                                                        from TPER_SOLICITUDES s
                                                                where s.situacion ='B' and
                                                s.cestado ='AP' and
                                                s.fresolucionanula is null and
                                                s.cclase in ('LIP','LEF','LPP','EFP','AIN')
                                                        ) a");

if(isset($res[0]['pResolucionDGAAPos']))
        $pResolucionDGAAPos = $res[0]['pResolucionDGAAPos'];
else
        $pResolucionDGAAPos = 0;

$res = $con->consultar("select count(1) as \"pResolucionDGAANeg\" from
                                                                (
                                                        select s.nsolicitud
                                                                from TPER_SOLICITUDES s
                                                        where s.situacion ='A' and
                                                        s.cestado ='NP' and
                                                s.fresolucion is null and
                                                s.cclase in ('LIP','LEF','LPP','EFP','AIN')

                                                        UNION

                                                        select s.nsolicitud
                                                        from TPER_SOLICITUDES s
                                                        where s.situacion ='A' and
                                                        s.cestado ='AP' and
                                                s.fanulaper is not null and
                                                s.fresolucionanula is null and
                                                s.cclase in ('LIP','LEF','LPP','EFP','AIN')
                                                        ) a");

if(isset($res[0]['pResolucionDGAANeg']))
        $pResolucionDGAANeg = $res[0]['pResolucionDGAANeg'];
else
        $pResolucionDGAANeg = 0;

// Obtener solicitudes pendientes de notificaciÃ³n resoluciÃ³n CIT
$res = $con->consultar("select count(1) as \"pFResolCITPos\" from
                                                                (
                                                        select s.nsolicitud
                                                                from TPER_SOLICITUDES s
                                                        where s.situacion ='A' and
                                                s.cestado = 'AP' and
                                                s.fanulaper is null and
                                                        s.fresolucion is not null and
                                                        s.cclase not in ('LIP','LEF','LPP','EFP','AIN')
                                                          and
                                                        s.ffirmaresolucion is null

                                                            UNION

                                                        select s.nsolicitud
                                                        from TPER_SOLICITUDES s
                                                        where s.situacion ='B' and
```

```
                                                    s.cestado <> 'AP' and
                                                    s.fresolucion is not null and
                                                    s.cclase not in ('LIP','LEF','LPP','EFP','AIN')
                                                      and
                                                    s.ffirmaresolucion is null

                                                      UNION

                                                      select s.nsolicitud
                                                      from TPER_SOLICITUDES s
                                                      where s.situacion ='B' and
                                                    s.cestado ='AP' and
                                                    s.fresolucionanula is not null and
                                                    s.cclase not in ('LIP','LEF','LPP','EFP','AIN')
                                                      and
                                                    s.ffirmaresolucionanula is null
                                                      ) a");

if(isset($res[0]['pFResolCITPos']))
          $pFResolCITPos = $res[0]['pFResolCITPos'];
else
          $pFResolCITPos = 0;

$res = $con->consultar("select count(1) as \"pFResolCITNeg\" from
                                                                (
                                                                 select s.nsolicitud
                                                           from TPER_SOLICITUDES s
                                                           where s.situacion ='A' and
                                                    s.cestado ='NP' and
                                                    s.fresolucion is not null and
                                                                  s.cclase not in
                                                   ('LIP','LEF','LPP','EFP','AIN') and
                                                           s.ffirmaresolucion is null

                                                           UNION

                                                               select s.nsolicitud
                                                           from TPER_SOLICITUDES s
                                                           where s.situacion ='A' and
                                                           s.cestado ='AP' and
                                                    s.fanulaper is not null and
                                                    s.fresolucionanula is not null and
                                                           s.cclase not in
                                                   ('LIP','LEF','LPP','EFP','AIN') and
                                                    s.ffirmaresolucionanula is null
                                                           ) a");

if(isset($res[0]['pFResolCITNeg']))
          $pFResolCITNeg = $res[0]['pFResolCITNeg'];
else
          $pFResolCITNeg = 0;


// Obtener solicitudes pendientes de notificación resolución DGAA
$res = $con->consultar("select count(1) as \"pFResolDGAAPos\" from
                                                                (
                                                                select s.nsolicitud
                                                           from TPER_SOLICITUDES s
                                                                where s.situacion ='A' and
                                                    s.cestado = 'AP' and
                                                    s.fanulaper is null and
                                                    s.fresolucion is not null and
                                                    s.cclase in ('LIP','LEF','LPP','EFP','AIN') and
                                                    s.ffirmaresolucion is null
                                                           UNION

                                                           select s.nsolicitud
                                                           from TPER_SOLICITUDES s
                                                           where s.situacion ='B' and
                                                    s.cestado <> 'AP' and
                                                    s.fresolucion is not null and
                                                    s.cclase in ('LIP','LEF','LPP','EFP','AIN') and
                                                    s.ffirmaresolucion is null
                                                           UNION

                                                                   select s.nsolicitud
                                                           from TPER_SOLICITUDES s
                                                           where s.situacion ='B' and
                                                    s.cestado ='AP' and
                                                    s.fresolucionanula is not null and
                                                    s.cclase in ('LIP','LEF','LPP','EFP','AIN') and
                                                    s.ffirmaresolucionanula is null
                                                    ) a");

if(isset($res[0]['pFResolDGAAPos']))
          $pFResolDGAAPos = $res[0]['pFResolDGAAPos'];
else
          $pFResolDGAAPos = 0;

$res = $con->consultar("select count(1) as \"pFResolDGAANeg\" from
                                                                (
                                                                select s.nsolicitud
                                                           from TPER_SOLICITUDES s
                                                           where s.situacion ='A' and
                                                           s.cestado ='NP' and
                                                    s.fresolucion is not null and
                                                    s.cclase in ('LIP','LEF','LPP','EFP','AIN') and
                                                    s.ffirmaresolucion is null
                                                           UNION

                                                           select s.nsolicitud
                                                           from TPER_SOLICITUDES s
```

```
                                                                        where s.situacion ='A' and
                                                                s.cestado ='AP' and
                                                                s.fanulaper is not null and
                                                                s.fresolucionanula is not null and
                                                                s.cclase in ('LIP','LEF','LPP','EFP','AIN') and
                                                                s.ffirmaresolucionanula is null
                                                                        ) a");

                if(isset($res[0]['pFResolDGAANeg']))
                        $pFResolDGAANeg = $res[0]['pFResolDGAANeg'];
                else
                        $pFResolDGAANeg = 0;

                $response['pValidar'] = $pValidar;
                $response['pResolucionCITPos'] = $pResolucionCITPos;
                $response['pResolucionCITNeg'] = $pResolucionCITNeg;
                $response['pResolucionDGAAPos'] = $pResolucionDGAAPos;
                $response['pResolucionDGAANeg'] = $pResolucionDGAANeg;
                $response['pFResolCITPos'] = $pFResolCITPos;
                $response['pFResolCITNeg'] = $pFResolCITNeg;
                $response['pFResolDGAAPos'] = $pFResolDGAAPos;
                $response['pFResolDGAANeg'] = $pFResolDGAANeg;
        }

        if (($perfil=='P_RESPPER') OR ($perfil=='P_REVISOR')) {

                // Obtener pendientes de firmar como responsable
                $res = $con->consultar("select count(1) as \"pFirmaRespPersonal\"
                                from TPER_SOLICITUDES s
                                where s.cestado IN ('PP','PN') and s.nsolicitud not in
                                    (select s2.nsolicitud from tper_solicitudesmodificadas s2, tper_solicitudes an
                                                        where an.nsolicitud = s2.nsolicitudmodifica and
                                                                an.cestado not in ('AP','NP') and
                                                                an.situacion in ('B','X') and
                                                                s2.anulacionresuelta = 'N'  )");

                if(isset($res[0]['pFirmaRespPersonal']))
                        $pFirmaRespPersonal = $res[0]['pFirmaRespPersonal'];
                else
                        $pFirmaRespPersonal = 0;

                // Obtener pendientes de revisar
                $res = $con->consultar("select count(1) as \"pRevisar\"
                                from TPER_SOLICITUDES s
                                where s.cestado = 'PV' and s.nsolicitud not in
                                     (select s2.nsolicitud from tper_solicitudesmodificadas s2, tper_solicitudes an
                                                        where    an.nsolicitud = s2.nsolicitudmodifica and
                                                        an.cestado in ('PV','PT','NI','PR','PC') and
                                                        an.situacion in ('B','X') and
                                                        s2.anulacionresuelta = 'N'  )");

                if(isset($res[0]['pRevisar']))
                        $pRevisar = $res[0]['pRevisar'];
                else
                        $pRevisar = 0;

                $response['pFirmaRespPersonal'] = $pFirmaRespPersonal;
                $response['pRevisar'] = $pRevisar;
        }

        IgepDebug::setDebug(DEBUG_USER,'FIN f_dameSolicitudesPendientes');
        return $response;
}


public function f_pendJustifCorrecto($pnsolicitud,$pvalida) {

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_pendJustifCorrecto');
        //Conexion
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);

        if($pvalida=='S') {

                $res = $con->operar("
                                UPDATE TPER_SOLICITUDES
                                SET   justificante = 'N', pendientenuevojustificante = 'S'
                                WHERE nsolicitud = ".$pnsolicitud);

        }
        else {

                $res = $con->operar("
                                UPDATE TPER_SOLICITUDES
                                SET   cestado = 'PT',
                                valida = CASE situacion WHEN 'A' then null ELSE valida END,
                                validaanula = CASE situacion WHEN 'B' then null ELSE validaanula END,
                                revisada= null, justificante = 'N', pendientenuevojustificante = 'S',
                                nrptramitador = CASE situacion WHEN 'A' THEN null ELSE nrptramitador END,
                                nrprevisor = CASE situacion WHEN 'A' THEN null ELSE nrprevisor END,
                                nrpanulatramitador = CASE situacion WHEN 'B' THEN null ELSE nrpanulatramitador END,
                                nrpanularevisor = CASE situacion WHEN 'B' THEN null ELSE nrpanularevisor END
                                WHERE nsolicitud = ".$pnsolicitud);
        }

        if($res!=-1)
                $res = 0;

        IgepDebug::setDebug(DEBUG_USER,'FIN f_pendJustifCorrecto');
        return $res;
}
```

```
public function f_dameDiasMaximoEGPoDEF($pdistancia,$pgradoParentesco,&$pdiasMaximo){

        //Generar el mensaje de la notificación a enviar
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameDiasMaximoEGPoDEF');
        if ($pdistancia=='L' and $pgradoParentesco=='1'){
          $pdiasMaximo = 6;
          return 0;
        }
        if ($pdistancia=='C' and $pgradoParentesco=='1'){
          $pdiasMaximo = 4;
          return 0;
        }
        if ($pdistancia=='L' and $pgradoParentesco=='2'){
          $pdiasMaximo = 5;
          return 0;
        }
        if ($pdistancia=='C' and $pgradoParentesco=='2'){
          $pdiasMaximo = 3;
          return 0;
        }

        IgepDebug::setDebug(DEBUG_USER,'FIN f_dameDiasMaximoEGPoDEF');
          return -1;

    }


public function f_dameDiasMaximoMFAoMUHoTDO($pcclase,$pdistancia,&$pdiasMaximo){

        //Generar el mensaje de la notificación a enviar
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameDiasMaximoMFAoMUHoTDO');
        if ($pdistancia=='L'){

          if ($pcclase=='TDO'){
                    $pdiasMaximo = 3;
          }
          else{
                    $pdiasMaximo = 2;
          }
          return 0;

        }
        if ($pdistancia=='C' ){

          if ($pcclase=='TDO'){
                    $pdiasMaximo = 2;
          }
          else{
                    $pdiasMaximo = 1;
          }
          return 0;//18/05/2010 INICIO
        }

        IgepDebug::setDebug(DEBUG_USER,'FIN f_dameDiasMaximoMFAoMUHoTDO');
          return -1;

}


function f_dameVecesPermiso($codClase,$pnregpgv, $panual, $pperiodo, &$totalVecesPermiso){

          IgepDebug::setDebug(DEBUG_USER,"INI f_dameVecesPermiso");

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);

          if ($panual == 'N'){
                    $res = $conexion->consultar("SELECT count(*)
                                                 FROM tper_solicitudes
                                                 WHERE cclase = '".$codClase."' and nregpgv = '".$pnregpgv."'and not
                                                 ((ffirmaresolucionanula is not null and situacion ='B' and cestado='AP')
                                                 or (ffirmaresolucion is not null and situacion ='A' and cestado='NP')) ");
          }else{
                    $res = $conexion->consultar("SELECT count(*)
                                                 FROM tper_solicitudes
                                                 WHERE cclase = '".$codClase."' and nregpgv = '".$pnregpgv."'
                                                 and not ((ffirmaresolucionanula is not null and situacion ='B' and
                                        cestado='AP') or (ffirmaresolucion is not null and situacion ='A' and cestado='NP'))
                                                 and periodo = $pperiodo");
           }
          $totalVecesPermiso=$res[0][count];

          IgepDebug::setDebug(DEBUG_USER,"FIN f_dameVecesPermiso -> totalVecesPermisos=".$res[0][count]);
          if(empty($res[0][count])or($res[0][count]==''))
                    return false;
           else
          return true;
   }


public function f_anularBorrarVAC($pnregpgv,$pcclase,$pfechaIni,$pfechaFin,$pperiodo,$panular) {

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_anularBorrarVAV');
          //Conexion
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          // Si se anulan o borran VAC comprobar que:
          // 1.- se dispone de ADI. Si es así,
          // 2.- ha solicitado alguno de los ADI que tiene disponibles y no está anulado. SI es así,
          // 3.- los ADI solicitados están pegados a las VAC que estoy anulando o borrando. Para ello:
```

```
//      3.a.- llamar a f_obtienePrimerDiaSolicitud() con ADI y la fecha inicio de las VAC
//       3.b.- llamar a f_obtieneUltimoDiaSolicitud() con ADI y la fecha fin de las VAC
//      3.c.- con el intervalo de fechas que devuelven, comprobar si hay ADI entre esas fechas
//         3.c.1.- para cada uno de ellos, ver si cumplen la regla 7. Para ello:
//           3.c.1.1.- llamar a f_obtienePrimerDiaSolicitud() con ADI y la fecha inicio de los ADI
//             3.c.1.2.- llamar a f_obtieneUltimoDiaSolicitud() con ADI y la fecha fin de los ADI
//           3.c.1.3.- con el intervalo de fechas que devuelven, comprobar que la duraciÃ³n no sea inferior a 7 dÃas

if($pcclase=='VAC') {

          // 1.- Comprobar si se dispone de dÃas ADI
          $res = $this->consultar(" SELECT diasadi_act FROM tper_personas      WHERE nregpgv = '".$pnregpgv."' ");
          if($res==-1) return 1;
                    $totalDiasAdi = $res[0]['diasadi_act'];

          if(!empty($totalDiasAdi) AND $totalDiasAdi!='' AND count($totalDiasAdi) > 0){

                    // 2.- Comprobar si ha solicitado alguno de los ADI que tiene disponibles y no estÃ¡ anulado
                    $resAnular = $this->consultar("
                              SELECT fecha_ini,fecha_fin    FROM tper_solicitudes s
                              WHERE nregpgv = '".$pnregpgv."' and cclase ='ADI' and periodo =
                                                  '".$pperiodo."' and situacion = 'A' and
                                        (s.cestado ='PT' or s.cestado ='PV' or s.cestado ='PN' or
                                        (s.cestado ='AP' and s.fanulaper is null) or s.cestado ='PP') ");

                    $resBorrar = $this->consultar("
                              SELECT fecha_ini,fecha_fin    FROM tper_solicitudes s
                              WHERE nregpgv = '".$pnregpgv."' and cclase ='ADI' and periodo =
                                                  '".$pperiodo."' and situacion = 'A' and
                                        (s.cestado ='PC' or s.cestado ='PR') ");

                    if($resAnular==-1) return 1;
                    if($resBorrar==-1) return 1;

                    $anularADI = false;
                    $borrarADI = false;

                    $solicitudesAnular= $resAnular[0];
                    $solicitudesBorrar= $resBorrar[0];

                    if (count($solicitudesAnular)>0) $anularADI = true;

                    if (count($solicitudesBorrar)>0) $borrarADI = true;


                    if ($anularADI or $borrarADI){ // Devuelve registros en cualquiera de las 2 selects
                              UsuariosDePersonal::f_obtenerMunPro($pnregpgv,$cpro,$cmun);
                              // 3.- Comprobar si los ADI solicitados estÃ¡n pegados a las VAC que estoy anulando
                              $fechaIniSolVAC = Solicitudes::f_obtienePrimerDiaSolicitud($pfechaIni, $pperiodo,
                                                                  $cpro, $cmun,'ADI',$pnregpgv);
                              $fechaFinSolVAC = Solicitudes::f_obtieneUltimoDiaSolicitud($pfechaFin, $pperiodo,
                                                                  $cpro, $cmun,'ADI',$pnregpgv);

                              // 3.c.- Comprobamos si hay ADI entre fechaIniSolVAC Y fechaFinSolVAC
                              $fechaIniSolVAC=$con->prepararFecha($fechaIniSolVAC);
                              $fechaFinSolVAC=$con->prepararFecha($fechaFinSolVAC);

                              $solicitudesADI = $this->consultar("
                                        SELECT periodo,fecha_ini,fecha_fin
                                        FROM tper_solicitudes
                                        WHERE cclase='ADI' and nregpgv = '".$pnregpgv."' and (fecha_ini <=
                                        to_date('$fechaFinSolVAC','dd/mm/yyyy')) and (fecha_fin >=
                                        to_date('$fechaIniSolVAC','dd/mm/yyyy') or fecha_fin is null ) ",array(
                                        'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA, 'fecha_fin'=>TIPO_FECHA, )) );


                              // 3.c.1.- Para cada uno de ellos, ver si cumplen la regla 7
                              foreach($solicitudesADI as $solicitudADI){


$fechaIniSol=Solicitudes::f_obtienePrimerDiaSolicitud($solicitudesADI[0]['fecha_ini'],
                                                  $solicitudesADI[0]['periodo'], $cpro, $cmun,'ADI',$pnregpgv);
                              $fechaFinSol=Solicitudes::f_obtieneUltimoDiaSolicitud($solicitudesADI[0]['fecha_fin'],
                                                  $solicitudesADI[0]['periodo'], $cpro, $cmun,'ADI',$pnregpgv);

                                        if (gvHidraTimestamp::cmp($fechaFinSol, $fechaIniSol)/(24*60*60)+1 <7){

                                                  if ($panular == 'N') $mensaj = 'borrar';
                                                  else      $mensaj = 'anular';

                                                  if ($anularADI and $borrarADI) $mensa = 'anular y borrar';
                                                            elseif ($anularADI) $mensa = 'anular';
                                                            elseif ($borrarADI) $mensa = 'borrar';

                                                  $mensaje.="Antes de ".$mensaj." sus vacaciones, deberá ".$mensa."
                                                  los días adicionales solicitados junto a sus vacaciones.";
                                                  return $mensaje;
                                        }

                              }

                    }
          }
}

IgepDebug::setDebug(DEBUG_USER,'FIN f_anularBorrarVAC');
$mensaje='0';
return $mensaje;
}
```

```
function f_esCambioDeSolicitud($pnsolicitud){

          IgepDebug::setDebug(DEBUG_USER,"INI f_esCambioDeSolicitud");

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);

          $res = $conexion->consultar("
                              SELECT
                                        count(avisos) as avisos
                              FROM
                                        TPER_SOLICITUDES
                              WHERE
                                        nsolicitud = $pnsolicitud and avisos LIKE '%Esta solicitud se considera
                                                                  una modificación de las siguientes solicitudes:%'");

          if($res[0]['avisos']!=0){
                    IgepDebug::setDebug(DEBUG_USER,"FIN true f_esCambioDeSolicitud" );
                    return true;
          }else{
                    IgepDebug::setDebug(DEBUG_USER,"FIN false f_esCambioDeSolicitud" );
                    return false;
          }
}



          /**
           * Devuelve una lista de solicitudes disfrutadas o no por el interesado, en un periodo determinado.
           * En caso de estar anuladas o no estar autorizadas, que no tengan fecha de resoluciÃ³n.
           *  @param $pnregpgv,$pperiodo,$pcclase,$listaSolicitudes
           * @return int
           */
public function f_dameSolicitudesValidas($pnregpgv,$pperiodo,$pcclase,&$listaSolicitudes){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameSolicitudesValidas');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);

          $listaSolicitudes = $this->consultar("  SELECT nsolicitud,fecha_ini,fecha_fin,observaciones
                                                  FROM tper_solicitudes
                                                  WHERE  nregpgv = '".$pnregpgv."' and periodo= '".$pperiodo."'
                                                            and cclase='".$pcclase."'  and NOT ((cestado in ('NR','NI')) OR
                                                            (situacion = 'A' and cestado='NP' and ffirmaresolucion is not
                                                            null) OR (situacion = 'B' and cestado='AP' and
                                                            ffirmaresolucionanula is not null))
                                                  ORDER BY fecha_ini");


          if($listaSolicitudes==-1) return array();

          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameSolicitudesValidas');

          return 0;
}


}
?>
```

– **Clase Personal:**

```php
<?php

class Personal{

    var $tarjeta;

    function Personal($ptarjeta){
        $this->tarjeta = $ptarjeta;
    }


    /**
    * f_dameTarjeta: Obtener la tarjeta de un usuario
    */
    function f_dameTarjeta($pnregpgv){
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameTarjeta');
        $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
        $conexion = new IgepConexion($g_oracle_dsn);
        $res = $conexion->consultar("SELECT codigo as \"codigo\"
                            FROM twps_personal
                            WHERE dnisin ='".$pnregpgv."' and fechabaja is null");
        IgepDebug::setDebug(DEBUG_USER,'FIN f_dameTarjeta');
        return $res[0]['codigo'];
    }



    /**
    * f_usuariosSuplenteActivo: Devuelve lo usuarios que pertenecen al grupo de firma del usuario validado, sólo si éste está
        preparado para firmar solicitudes en la actualidad, es decir, está activo como suplente de firma
    */
    function f_usuariosSuplenteActivo($pnregpgv,$perfil,&$usuariosValidos){
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_usuariosSuplenteActivo');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);
        if ($perfil=='P_RESPPER')
          $usuariosValidos = Personal::f_dameUsuariosValidos($pnregpgv,'S','S');
          else
                    $usuariosValidos = Personal::f_dameUsuariosValidos($pnregpgv,'N','S');

        if(count($usuariosValidos)==0 OR $usuariosValidos==-1 ){


          if (!Firmantes::f_esResponsablePersonalTitular($pnregpgv))
          {
                    // Volvemos a realizar la llamada pero con el último parámetro a 'N'
                    if ($perfil=='P_RESPPER')
                              $usuariosValidos = Personal::f_dameUsuariosValidos($pnregpgv,'S','N');
                    else
                              $usuariosValidos = Personal::f_dameUsuariosValidos($pnregpgv,'N','N');

                    if(count($usuariosValidos)>0){
                    // El firmante no estÃ¡ activo y deberÃ¡ activarse
                              return -1;
                    }
                                        else{
                    // El firmante no existe en la tabla de firmantes. Se tendrá que dar de alta para poder firmar.
                    return -2;
                                        }
          }
        }
        else{
          if (!Firmantes::f_esFirmanteTitular($pnregpgv))
          {
                    $sql = null;
                    Firmantes::f_listaGruposFirmaSuplente($pnregpgv,$sql);
                    $gruposFirmaSuplente = $con->consultar($sql);
                    if(is_array($gruposFirmaSuplente) AND count($gruposFirmaSuplente)==1 AND
                              $gruposFirmaSuplente[0]['activo']=='N'  AND $perfil=='P_RESPONSA')
                                        {
                                                  // El firmante no está activo y deberá activarse
                                                  return -1;
                                        }
          }
          }
        if(Firmantes::f_esResponsableDePuertos($pnregpgv)){
            //Ejecutamos dameUsuariosPuertos
            $usuariosPuertos = $con->consultar('SELECT nregpgv as "nregpgv"
                            FROM VPER_OCUPACION
                            WHERE depuertos = \'S\'');
            if($usuariosValidos==-1)
                $usuariosValidos = array();
            foreach($usuariosPuertos as $usuarioPuerto)
                array_push($usuariosValidos,$usuarioPuerto['nregpgv']);
        }
        IgepDebug::setDebug(DEBUG_USER,'FIN f_usuariosSuplenteActivo');
    }
```

```
/**
* f_dameUsuariosValidos: Personas que pertenecen al grupo de firma al que el usuario (UsuariosQueSolicitan) puede firmar
*/
function f_dameUsuariosValidos($pnregpgv,$ptitulares,$pactivo){

    IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameUsuariosValidos');
    $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
    // Obtenemos los grupos de firma que puede firmar el firmante pnregpgv
    $grupos = Firmantes::f_obtenerGruposFirma($pnregpgv,$pactivo);
    if($grupos<0) return $grupos;

    // Para cada uno de los grupos obtenidos buscamos en las tablas de Control de presencia,
    // las personas que pertenecen a él, es decir, las personas a las que el firmante puede firmar:

    $conexion = new IgepConexion($g_oracle_dsn);
    $res = array();
    foreach($grupos as $grupo){
        $a_nregpgv = $conexion->consultar('SELECT dnisin as "dnisin"
                                    FROM twps_personal
                                    WHERE centro = \''.$grupo['centro'].'\' and
                                     departamento = \''.$grupo['departamento'].'\' and
                                     seccion = \''.$grupo['seccion'].'\' and fechabaja is null order by apellido1,
                                                                                    apellido2, nombre ');
        // Excluimos de la lista de usuarios a firmar, aquellos que sean responsables no suplentes, salvo
        // que el firmante sea no suplente
        foreach($a_nregpgv as $index=>$var_nregpgv){

                if($ptitulares=='N'){//Queremos excluir titulares

                        if(!(Firmantes::f_esResponsableNoSuplente($var_nregpgv['dnisin'],$grupo['centro'],
                                $grupo['departamento'],$grupo['seccion']) AND ($pnregpgv!=$var_nregpgv['dnisin'])))
                                array_push($res,$a_nregpgv[$index]["dnisin"]);
                        }else{
                                array_push($res,$a_nregpgv[$index]["dnisin"]);
                        }
                }
    }

    //Ahora debemos tener en cuenta dos casos adicionales.
    //1º las personas que no fichan y que en su centro se ficha, y somos responsables de firma de personas de su servicio
    //en el centro
    //2º las personas en cuyo centro no se ficha y somos responsable de firma,

    // Obtener las personas que no fichan y que en su centro se ficha, para los cuales pnregpgv es responsable de firma
      de otras personas de su servicio en el mismo centro

      $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
      $conexion = new IgepConexion($g_dsn, true);
      $str_var_nregpgv = implode("','",$res);
      $b_nregpgv = $conexion->consultar('SELECT v2.nregpgv as "nregpgv"
                                    FROM vper_ocupacion v1,tper_cod_centros_trab c,vper_ocupacion v2
                                    WHERE v1.nregpgv in (\''.$str_var_nregpgv.'\') and
                                     v1.ccentrab = c.ccentrab and
                                     c.ficha=\'S\' and
                                     v1.ficha=\'S\' and
                                     v2.ccentrab = v1.ccentrab and
                                     v2.cdg = v1.cdg and
                                     v2.cserv = v1.cserv and
                                     v2.ficha = \'N\'
                                    GROUP BY  v2.nregpgv      ');

      foreach($b_nregpgv as $aux)
                array_push($res,$aux['nregpgv']);

      // Obtener las personas en cuyo centro no se ficha y somos responsable de firma.
      $conexion = new IgepConexion($g_dsn, true);

      $c_nregpgv = $conexion->consultar('SELECT v.nregpgv as "nregpgv"
                                    FROM vper_ocupacion v,tper_cod_centros_trab c
                                    WHERE c.nregpgv = \''.$pnregpgv.'\' and
                                     c.ficha=\'N\' and
                                     c.ccentrab = v.ccentrab and
                                     v.ficha =\'N\' ');


      foreach($c_nregpgv as $aux)
                array_push($res,$aux['nregpgv']);

      IgepDebug::setDebug(DEBUG_USER,'FIN f_dameUsuariosValidos');
     return $res;
}




/**
* f_dameUsuariosPersonal: Personas que pertenecen al grupo de firma del firmante
*/
function f_dameUsuariosPersonal($pcentro, $pdepartamento, $pseccion){
    IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameUsuariosPersonal');
    $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
    $conexion = new IgepConexion($g_oracle_dsn);
    $res = array();

    $a_nregpgv = $conexion->consultar('SELECT dnisin as "dnisin"
                                FROM twps_personal
                                WHERE centro = \''.$pcentro.'\' and
                                 departamento = \''.$pdepartamento.'\' and
                                 seccion = \''.$pseccion.'\' and fechabaja is null order by apellido1, apellido2,
                                                                                nombre ');
    // Excluimos de la lista de usuarios a firmar, aquellos que sean responsables no suplentes, salvo
```

```
    // que el firmante sea no suplente
    foreach($a_nregpgv as $index=>$var_nregpgv){
                        array_push($res,$a_nregpgv[$index]["dnisin"]);
    }
    IgepDebug::setDebug(DEBUG_USER,'FIN f_dameUsuariosPersonal');
    return $res;
}


/**
* f_obtenerGrupo: Obtener el grupo al que pertenece el usuario
*/
function f_obtenerGrupo($pnregpgv){
    IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerGrupo');
    $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
    $conexion = new IgepConexion($g_oracle_dsn);
    $a_nregpgv = $conexion->consultar('SELECT centro as "centro", departamento as "departamento", seccion as "seccion"
                        FROM twps_personal
                        WHERE dnisin = \''.$pnregpgv.'\' and fechabaja is null');
    IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerGrupo');
    return $a_nregpgv[0];
}



function f_obtenerUltimoFichaje($pnregpgv){
    IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerUltimoFichaje');
    $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
    $conOracle = new IgepConexion($g_oracle_dsn);

    $res = $conOracle->consultar(" SELECT Ultimofichaje as \"ultimoFichaje\"
                        FROM twps_personal
                        WHERE dnisin= '".$pnregpgv."' and fechabaja is null",array(
                                                'DATATYPES'=>array('ultimoFichaje'=>TIPO_FECHAHORA,)));
    IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerUltimoFichaje');
    if(!empty($res[0]['ultimoFichaje']))
        return $res[0]['ultimoFichaje'];
    else
        return null;
}


/**
* f_existeAlgunFichajeoIncidencia: Comprobar si hay algÃºn fichaje o incidencia
*/
function f_existeAlgunFichajeoIncidencia($pfechaInicio,$pfechaFin,$nregpgv,&$resultado){
    IgepDebug::setDebug(DEBUG_USER,'INICIO f_existeAlgunFichajeoIncidencia');
    $fechaInicio= clone $pfechaInicio;
    if ($pfechaFin!=null)
      $fechaFin= clone $pfechaFin;
    // Devuelve true si hay algún fichaje o incidencia o IT. En cualquier otro caso devuelve false
    $hayFichajesoIncidenciasoIT=false;
    $var_tarjeta = Personal::f_dameTarjeta($nregpgv);
    $dias= ValidacionHorario::f_existeValidacionHorario($fechaInicio,$fechaFin,$var_tarjeta);


      if(is_null($dias['0'])){
                while(gvHidraTimestamp::cmp($fechaInicio, $fechaFin)<=0){
                        //No hay validación horario (es fin de semana, festivo o no está definido en el reloj
                        por fallo)
                        if(!ValidacionHorario::f_hayValidacion($fechaInicio, $fechaInicio, $var_tarjeta)){
                                $resultado['tipo']='P';
                                $resultado['mensaje']="No se ha podido realizar la anulación de su solicitud por
                                        fallos en la aplicación de 'Control de presencia'.";
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_existeAlgunFichajeoIncidencia');
                        return -1;
                        }
                        $fechaInicio->addDays(1);
                }
       }

   foreach($dias as $dia){
        if(ValidacionHorario::f_existeFichaje($dia,$var_tarjeta))
            $hayFichajesoIncidenciasoIT = true;
        if(ValidacionHorario::f_existeIncidencia($dia,$var_tarjeta))
            $hayFichajesoIncidenciasoIT = true;
        if(Permisos::f_existeIT($dia,$nregpgv))
            $hayFichajesoIncidenciasoIT = true;
    }
    IgepDebug::setDebug(DEBUG_USER,'FIN f_existeAlgunFichajeoIncidencia');
    return $hayFichajesoIncidenciasoIT;
}



function f_intervalosNoFichados($pfecha_ini,$pfecha_fin, $nregpgv,&$resultado){

    IgepDebug::setDebug(DEBUG_USER,'INICIO f_intervalosNoFichados');
    $dia = $fecha_ini = clone $pfecha_ini;
    if($pfecha_fin!=null)
      $fecha_fin= clone $pfecha_fin;

    $fichajeant=true;
    $contador = 0;
    $fechas = array();

    $var_tarjeta = Personal::f_dameTarjeta($nregpgv);

    while($dia<=$fecha_fin){

      $fechaVal = ValidacionHorario::f_existeValidacionHorario($dia, $dia, $var_tarjeta);
      $exiteincidencia = validacionHorario::f_existeIncidencia($dia,$var_tarjeta);
```

```
        $exiteFichaje = validacionHorario::f_existeFichaje($dia,$var_tarjeta);
        $existeIT = Permisos::f_existeIT($dia,$nregpgv);
        if(!empty($fechaVal)){
                if(!($exiteFichaje OR $exiteincidencia OR $existeIT)){
                        if($fichajeant){
                                $fechas[$contador]= clone $dia;
                                ++$contador;
                                $fichajeant = false;
                        }
                        if($dia == $fecha_fin)
                       $fechas[$contador] = clone $dia;
                }
                else {// hay fichajes en el dia
                        if(!$fichajeant){
                                $diaContador= clone $dia;
                                $diaContador->subDays(1);
                                $fechas[$contador] = clone $diaContador;
                        ++$contador;
                        $fichajeant = true;
                }
                }
        }else{// No hay validación horario (es fin de semana, festivo o no está definido en el reloj por fallo)
                if (!ValidacionHorario::f_hayValidacion($dia,$dia,$var_tarjeta )){
                        $resultado['tipo']='P';
                        $resultado['mensaje']="No se ha podido realizar la anulaciÃ³n de su solicitud por fallos en la
                                                                aplicación de 'Control de presencia'.";
                        return -1;
                }else{
                        if(gvHidraTimestamp::cmp($dia,$fecha_fin)==0 AND !$fichajeant){
                                $diaContador = clone $dia;
                                $fechas[$contador]= $diaContador;
                        }
                }
        }
          $dia->addDays(1);
      }
      IgepDebug::setDebug(DEBUG_USER,'FIN f_intervalosNoFichados');
      return $fechas;
  }


  function f_intervalosFichados($pfecha_ini,$pfecha_fin, $nregpgv,&$resultado){
      IgepDebug::setDebug(DEBUG_USER,'INICIO f_intervalosFichados');
      $dia = $fecha_ini = clone $pfecha_ini;
      if ($pfecha_fin!=null)
        $fecha_fin = clone $pfecha_fin;

      $nofichajeant=true;
      $contador =0;
      $fechas = array();

      $var_tarjeta = Personal::f_dameTarjeta($nregpgv);

      while($dia<=$fecha_fin){
          $exiteFichaje = ValidacionHorario::f_existeFichaje($dia,$var_tarjeta);
          $exiteincidencia = ValidacionHorario::f_existeIncidencia($dia,$var_tarjeta);
          $existeIT = Permisos::f_existeIT($dia,$nregpgv);
          $fechaVal = ValidacionHorario::f_existeValidacionHorario($dia, $dia, $var_tarjeta);

          if(!empty($fechaVal)){
                  if($exiteFichaje OR $exiteincidencia OR $existeIT){//Hay fichajes en el día
                  if($nofichajeant){
                            $fechas[$contador] = clone $dia;
                            $contador++;
                            $nofichajeant = false;
                       }

              if(gvHidraTimestamp::cmp($dia,$fecha_fin)==0){
                  $fechas[$contador] = clone $dia;
              }
                    }else{// no hay fichajes en el dia
                        if (!$nofichajeant){
                                        $diaContador= clone $dia;
                                $diaContador->subDays(1);
                                $fechas[$contador] = clone $diaContador;

                                        ++$contador;
                                                $nofichajeant = true;
                                        }
                  }
          }else{// No hay validación horario (es fin de semana, festivo o no está definido en el reloj por fallo
                  if(!ValidacionHorario::f_hayValidacion($dia,$dia,$var_tarjeta)){
                            $resultado['tipo']='P';
                            $resultado['mensaje']="No se ha podido realizar la anulaciÃ³n de su solicitud por fallos en la
                                                                    aplicación de 'Control de presencia'.";
                            return -1;
                  }else{
                            if((gvHidraTimestamp::cmp($dia,$fecha_fin)==0) AND !$nofichajeant){
                                $diaContador= $dia;
                                        $fechas[$contador] = clone $dia;
                            }
                                        }
          }
                            $dia->addDays(1);
      }
      IgepDebug::setDebug(DEBUG_USER,'FIN f_intervalosFichados');
      return $fechas;
  }

}
?>
```

– Clase UsuariosDePersonal:

```php
<?php

class UsuariosDePersonal{

    function UsuariosDePersonal(){
    }
    /**
     * Función que Devuelve los trienios de la persona interesada
     * @return integer
     * */


function f_obtenerTrienios($pnregpv,& $ptrienios){

          // Conexión a ORACLE
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerTrienios');
          $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
          $conexion = new IgepConexion($g_oracle_dsn);
          $res = $conexion->consultar("
                    select count(nregpgv) as \"ptrienios\"
                    from tper_trienios
                    where nregpgv = '$pnregpv' ");
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerTrienios');
         if ($res<0 )return -1;
         if (isset($res['0']['ptrienios']))$ptrienios=$res['0']['ptrienios'];
         return 0;

}



    /**
     * Obtener cpro y cmun
     *
     * **/

function f_obtenerMunPro($pnregpgv,&$pcpro,&$pcmun){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerMunPro');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          $res = $conexion->consultar("SELECT cpro, cmun
                                       FROM vper_ocupacion_historico
                                       WHERE nregpgv = '$pnregpgv'");
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerMunPro');
          if ($res<0 )return -1;
          $pcpro=$res['0']['cpro'];
          $pcmun=$res['0']['cmun'];
          return 0;
}



    /**
     * fesInterino
     * Indica si la persona es o no interina
     * @return boolean
     *
     **/

function f_dameReljur($pnregpgv, & $pcreljur){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameReljur');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          $res = $conexion->consultar(" SELECT creljur as \"creljur\"
                                       FROM VPER_OCUPACION_HISTORICO
                                       WHERE nregpgv = '$pnregpgv'");
          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameReljur');
          if ($res<0 )return -1;
          if (isset($res['0']['creljur']))$pcreljur=$res['0']['creljur'];
          return 0;
}


    /**
     * f_obtenerCdgCserv
     * Devuelve DG y servicio
     * @return boolean
     *
     **/

function f_obtenerCdgCserv($pnregpgv,&$pcdg,&$pcdser){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerCdgCserv');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          $res = $conexion->consultar("SELECT cdg, cserv
                                       FROM vper_ocupacion_historico
                                       WHERE nregpgv = '$pnregpgv'");
          $pcdg=$res['0']['cdg'];
          $pcdser=$res['0']['cserv'];
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerCdgCserv');
          if(count($res)>0)
                    return TRUE;
          else
```

```
		return FALSE;
}


	/**
	 *	Devuelve la fecha de antiguedad de la persona interesada
	 *  @return Date,integer
	 *
	 * */

function f_obtenerFechaAntiguedad($pnregpgv){

		IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerFechaAntiguedad');
		$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
		$conexion = new IgepConexion($g_dsn, true);
		$res = $conexion->consultar("
				SELECT fantig
				FROM VPER_OCUPACION_historico
				WHERE nregpgv = '$pnregpgv' AND
				creljur in ('FI','FC')",array( 'DATATYPES'=>array('fantig'=>TIPO_FECHA,)));
		IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerFechaAntiguedad');
		return $res['0']['fantig'];
}



	/**
	 *	Obtiene el nÃºmero de dÃas trabajados por el interesado en el periodo pasado por parÃ¡metro
	 *  @return Date,integer
	 *
	 * */
function f_obtenerDiasTrabajados($pnregpgv,$pperiodo,$pdePuertos,$ppftoma,&$pdiasTrabajados,&$pdiasTrabajadosVV6){

		IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerDiasTrabajados');
		if ($ppftoma!=null)
			$pftoma = clone $ppftoma;

		//Creamos la conexión
		$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
		$conPost = new IgepConexion($g_dsn, true);


		//Inicializamos los días trabajados a -1 para indicar que la persona lleva trabajando mÃ¡s de 365 días
		$pdiasTrabajados = -1;
		$pdiasTrabajadosVV6 = -1;
		if ( $pdePuertos == 'N') {
			$res = $conPost->consultar("select to_char(fcesefuturo,'dd/mm/yyyy') as fcesefuturo
					from tper_personas
					where nregpgv= '".$pnregpgv."' andzxtract(year from fcesefuturo)<='".$pperiodo."'");
			$fcesefuturo = $res[0]['fcesefuturo'];

		}

		//Ver si la persona no ha trabajado todo el año
		//La persona toma posesión después del 1 de enero del año que realiza una solicitud. Si es de puertos, el cálculo
		// a realizar es diferente

		$fechaAux=new gvHidraTimestamp("01/01/$pperiodo");
		$fechaAux->setTime(0,0,0);

		if((gvHidraTimestamp::cmp($fechaAux, $pftoma) > 0) OR $pdePuertos == 'S'OR $fcesefuturo != null){
			if($pdePuertos == 'N'){
				//Accedemos a la bd de "per"
				$g_oracle_dsn= ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
				$conOracle = new IgepConexion($g_oracle_dsn);

				$res = $conOracle->consultar("select sum( case when fbajpad is null then
							case when '".$fcesefuturo."' is null

							then to_date('3112'||to_char($pperiodo,'9999'),'ddmmyyyy')

							else to_date('". $fcesefuturo ."','dd/mm/yyyy') + 0 end
							 else fbajpad + 0 end
							-
				greatest(ftpos, to_date('0101'||to_char($pperiodo,'9999'),'ddmmyyyy')) + 1)  as \"diasTrabajados\"
				    from tper_puestos_trab_admon
				    where nregpgv = '".$pnregpgv."' and (extract(year from fbajpad)=$pperiodo or fbajpad is null)
				          and (ftpos <> fbajpad or fbajpad is null)");

				$pdiasTrabajados = $res[0]['diasTrabajados'];

				if ($pdiasTrabajados < 0 ) $pdiasTrabajados = 0;

				// Para calcular el máximo disponible de días de asuntos propios, los periodos de excedencia
				// por cuidado de hijos (ccausacese='EH') se computan como días trabajados. Utilizaremos una
				// nueva variable pdiasTrabajadosVV6 para incluir los días de excedencia por cuidado de hijos

					$pdiasTrabajadosVV6 = $pdiasTrabajados;

				// Se recorren las fechas de toma de posesión y baja durante periodo actual, y aquella del
				periodo anterior
				// que sea la última fecha de toma de posesión o baja
				$cursorExcedencias =  $conOracle->consultar("SELECT t.ftpos as \"ftpos\", t.fbajpad as
				\"fbajpad\", t.ccausacese as \"ccausacese\"
					FROM tper_puestos_trab_admon t
					WHERE t.nregpgv = '". $pnregpgv."' AND
						(( extract(year from t.fbajpad)=$pperiodo OR
						extract(year from t.ftpos)=$pperiodo)
						OR
						EXISTS (SELECT MAX(fbajpad), MAX(ftpos)
```

```
                                                                FROM tper_puestos_trab_admon
                                                                WHERE nregpgv = t.nregpgv AND
                                                                (fbajpad = t.fbajpad OR ftpos = t.ftpos) AND
                                                                ( extract(year from fbajpad)=$pperiodo - 1 OR
                                                                 extract(year from ftpos)=$pperiodo - 1)
                                                                HAVING MAX(fbajpad)> MAX(ftpos)))
                                                              ORDER BY
                                                                t.ftpos",array('DATATYPES'=>array('ftpos'=>
                                                                          TIPO_FECHA,'fbajpad'=>TIPO_FECHA )));
                                $fechaInicioExcedencia = null;


                                // Si se encuentra una baja de excedencia por cuidado de hijos, se guarda la fecha, y
                                   en la siguiente
                                // iteración se obtiene la siguiente fecha de toma de posesión, que a su vez será la
                                   del final de la
                                // excedencia. Con estas fechas se obtiene la duración de la excedencia, y se suma a
                                   pdiasTrabajadosVV6
                                foreach ($cursorExcedencias as $elementoCursor){
                                          if(!($fechaInicioExcedencia===null)){

                                                    $fechaFinExcedencia = $elementoCursor['ftpos']->subDays(1);
                                                    $pdiasTrabajadosVV6 +=  round(($fechaFinExcedencia->getTimeStamp()
                                                              - $fechaInicioExcedencia->getTimeStamp())/(24*60*60)) + 1;

                                                    $fechaInicioExcedencia = null;
                                          }
                                          if($elementoCursor['ccausacese'] == 'EH'){
                                                    $elementoCursor['fbajpad']->addDays(1);
                                                    $fechaInicioExcedencia = $elementoCursor['fbajpad']->addDays(1);
                                          }
                                }
                     }//Fin no es de puertos
                     else{

                                $pdiasTrabajados = $conPost->consultar("select sum( case when fbajpad is null then
                                to_date('3112'||to_char($pperiodo,'9999'),'ddmmyyyy') else fbajpad + 0 end -
                                greatest(ftpos, to_date('0101'||to_char($pperiodo,'9999'),'ddmmyyyy')) + 1) as
                                \"diasTrabajadosVV6\"
                                                                        from tpr2_puestos_trab_admon
                                                                        where nregpgv = '".$pnregpgv."' and
                                                                                  extract(year from fbajpad)=$pperiodo
                                                                                  and (ftpos <> fbajpad or fbajpad is null)");
                                $pdiasTrabajadosVV6 = $pdiasTrabajados[0]['diasTrabajadosVV6'];
                     } //Fin es de puertos

          if($pdiasTrabajados >= 365)
                     $pdiasTrabajados = -1;
          if($pdiasTrabajadosVV6 >= 365)
                     $pdiasTrabajadosVV6 = -1;
          } //Fin de Si ftoma>... o es de puertos

          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerDiasTrabajados');
}//fin f_obtenerDiasTrabajados



          /**
           * Obtiene el nombre y apellidos del interesado a partir del nregpgv
           *
           * Fija el directorio que contiene la informaciÃ³n de custom
           *
           * @access public
           *
           * @param string $nregpgv Cadena de entrada que indica el directorio de custom
           * @param string $dnombre Cadena de salida con el nombre del interesado
           * @param string $dapell1 Cadena de salida con el 1er apellido del interesado
           * @param string $dapell2 Cadena de salida con el 2o apellido del interesado
           * @return boolean true o false en funciÃ³n del Ã©xito o fracaso de la invocaciÃ³n
           *
          */
public function f_dameNombreApellidosInteresado ($nregpgv, &$dnombre, &$dapell1, &$dapell2){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameNombreApellidosInteresado');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          $res = $conexion->consultar("
                     SELECT dnombre, dapell1, dapell2
                     FROM   vper_ocupacion_historico
                     WHERE  nregpgv like '$nregpgv'");

          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameNombreApellidosInteresado');
          if (count($res)==0) return false;
          $dnombre = $res['0']['dnombre'];
          $dapell1  = $res['0']['dapell1'];
          $dapell2  = $res['0']['dapell2'];
          return true;
}// f_dameNombreApellidosInteresado



          /**
           * Obtiene el nombre y apellidos del interesado a partir del DNI y si es o no de puertos ('S' o 'N')
           *
           * La consulta puede devolver mÃ¡s de un registro, asÃ que devolveremos SIEMPRE el primero de ellos
           *
           * @access public
           *
           * @param string $dnombre Cadena de salida con el nombre del interesado
           * @param string $dapell1 Cadena de salida con el 1er apellido del interesado
           * @param string $dapell2 Cadena de salida con el 2o apellido del interesado
           * @param string $nregpgv Cadena de salida con el nregpgv del interesado
```

```
           * @param string $dni DNI del usuario
           * @param string $esDePuertos 'S'|'N' Caracter que indica si el usuario es o no de puertos @default 'N'
           * @return boolean true o false en funciÃ³n del Ã©xito o fracaso de la invocaciÃ³n
           *
           */
public function f_dameNombreApellidosInteresadoPorDNITipoUsuario (&$dnombre, &$dapell1, &$dapell2, &$nregpgv, $dni,
                                                                  $esDePuertos='N'){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameNombreApellidosInteresadoPorDNITipoUsuario');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          $res = $conexion->consultar("
                    SELECT    nregpgv, dnombre, dapell1, dapell2
                    FROM      vper_ocupacion_historico
                    WHERE     nregpgv like '".$dni."__' and
                              depuertos = '$esDePuertos'");

          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameNombreApellidosInteresadoPorDNITipoUsuario');
          if(count($res)==0) return false;
          $dnombre = $res['0']['dnombre'];
          $dapell1 = $res['0']['dapell1'];
          $dapell2 = $res['0']['dapell2'];
          $nregpgv = $res['0']['nregpgv'];
          return true;
}// f_dameNombreApellidosInteresadoPorDNITipoUsuario


    /**
    * f_dameUsuariosPorDGyServicio:  Devuelve los nregpgv de los empleados que pertenecen a la DG pcdg y el servicio pcserv
    */
function f_dameUsuariosPorDGyServicio($pcdg,$pcserv){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameUsuariosPorDGyServicio');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          $res = $conexion->consultar("SELECT nregpgv FROM vper_ocupacion WHERE cdg = '$pcdg' and cserv = '$pcserv'");
          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameUsuariosPorDGyServicio');

          $resultado = array();
          foreach($res as $index){
                    array_push($resultado,$index['nregpgv']);
          }

          return $resultado;
}// f_dameUsuariosPorDGyServicio



    /**
    * f_dameEmpleadosFichanPorCentro: Devuelve una lista de usuarios (nregpgv) para el centro de trabajo pasado por parámetro
    */
function f_dameEmpleadosFichanPorCentro($pccentrab,$pcdg,$pcserv){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameEmpleadosFichanPorCentro');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          // Añadimos nuevos condiciones al where (and cdg = '$pcdg' and cserv = '$pcserv')
          $res = $conexion->consultar("SELECT nregpgv FROM vper_ocupacion WHERE ccentrab = '$pccentrab' and ficha = 'S' and
                                                                  cdg = '$pcdg' and cserv = '$pcserv'");

          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameEmpleadosFichanPorCentro');

          $pnregpgv = array();
          foreach($res as $index){
                    array_push($pnregpgv,$index['nregpgv']);

          }
          return $pnregpgv;
}// f_dameEmpleadosFichanPorCentro



    //  f_damePorcentajeReduccion()
function f_damePorcentajeReduccion($pHorasReduccion,$pnregpgv,&$phorasSemana){

      IgepDebug::setDebug(DEBUG_USER,'INICIO f_damePorcentajeReduccion');
      $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
      $conexion = new IgepConexion($g_dsn, true);
      $res_horasSemana=$conexion->consultar("SELECT horas * CASE WHEN pjor IS NULL THEN 1 ELSE cast(pjor as numeric)/100
                                                                  END as \"horasSemana\" FROM vper_ocupacion
                                            WHERE nregpgv = '$pnregpgv'");
      $phorasSemana = $res_horasSemana['0']['horasSemana'];

      // Cambiado cálculo de porcentaje para convertir las horas del campo pHorasReducción de cadena de texto a numérico
      // pporcentajeReduccion = ((substr(pHorasReduccion,1,1) + substr(pHorasReduccion,2,3)/60) * 5) / var_horasSemana;

      if (($phorasSemana>0) or ($phorasSemana!=null)){
       $pporcentajeReduccion=round((((substr($pHorasReduccion,0,1)+(substr($pHorasReduccion,-2)/60))*5)/$phorasSemana)*100) ;
      }
      else {
          $pporcentajeReduccion=0;
      }

      IgepDebug::setDebug(DEBUG_USER,'FIN f_damePorcentajeReduccion');
      return $pporcentajeReduccion;


} // f_damePorcentajeReduccion()
```

```php
    /**
    * f_noFichan:  Indica si la persona no ficha
    */
function f_noFichan($pnregpgv){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_noFichan');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);
          $var_nofichan = $conexion->consultar("SELECT nregpgv
                                                FROM vper_ocupacion_historico
                                                WHERE nregpgv = '$pnregpgv' and ficha='N'");

          IgepDebug::setDebug(DEBUG_USER,'FIN f_noFichan');
          if(count($var_nofichan)>0)
                    return TRUE;
          else
                    return FALSE;
}



    /**
    * f_dameCentroTrab:  Devuelve el centro de trabajo de la persona
    */
function f_dameCentroTrab($pnregpgv, &$pccentrab){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameCentroTrab');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);

          $res = $conexion->consultar("SELECT ccentrab as \"ccentrab\"
                                       FROM VPER_OCUPACION_HISTORICO
                                       WHERE nregpgv = '$pnregpgv'");
          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameCentroTrab');
          if ($res<0 )return FALSE;
          if (isset($res['0']['ccentrab']))$pccentrab=$res['0']['ccentrab'];
          return TRUE;
}

}//Fin clase
?>
```

– Clase SolicitudesModificadas:

```php
<?php

class SolicitudesModificadas{



public static function f_actualizaSolModificadas($nsolicitud,$resueltas,&$con){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_actualizaSolModificadas');
          $solicitudes = $con->operar( " UPDATE TPER_SOLICITUDESMODIFICADAS
                                         SET anulacionresuelta = '$resueltas'
                                         WHERE nsolicitud = $nsolicitud ");
          if($solicitudes<0) return -1;
          return 0;
           IgepDebug::setDebug(DEBUG_USER,'FIN f_actualizaSolModificadas');
}



public static function f_altaSolicitudesModificadas($nsol,$nsolAnula,$resuelta,&$con){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_altaSolicitudesModificadas');
          //Insertamos la solicitudes de anulaciÃ³n modificadas por un alta o una anulaciÃ³n
          $solicitudes=$con->operar("INSERT INTO TPER_SOLICITUDESMODIFICADAS (nsolicitud,nsolicitudmodifica,anulacionresuelta)
                                         VALUES ($nsol, $nsolAnula, '$resuelta'); ");
           if($solicitudes<0) return -1;
                    return 0;
          IgepDebug::setDebug(DEBUG_USER,'FIN f_altaSolicitudesModificadas');
}


          /**
           * Devuelve las solicitudes de alta o anulación que modifican a la solicitud de anulación pasada por parámetro
           * @param integer $nsol
           * @param string $resuelta
           * @param array $listaSolicitudes
           * @return integer
           */
public static function f_dameSolicitudesQueModifican($nsol,$resuelta,&$listaSolicitudes){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameSolicitudesQueModifican');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          $listaSolicitudes = $con->consultar( "
                                         select sm.nsolicitud, s.situacion, s.cestado
                                         from tper_solicitudesmodificadas sm, tper_solicitudes s
                                         where sm.nsolicitud = s.nsolicitud and sm.nsolicitudmodifica = $nsol and
                                                                                 sm.anulacionresuelta = '$resuelta' ");
          if($listaSolicitudes<0) return -1;
          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameSolicitudesQueModifican');
          return 0;
}



          /**
           * Devuelve las solicitudes de anulación modificadas por la solicitud de alta o anulación pasada por parámetro
           * @param integer $nsol
           * @param string $resueltas
           * @param array $listaSolicitudes
           * @return integer
           */
public static function f_dameSolicitudesModificadas($nsol,$resueltas,&$listaSolicitudes){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameSolicitudesModificadas');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          $listaSolicitudes = $con->consultar( "
                    select sm.nsolicitudmodifica
                     from tper_solicitudesmodificadas sm, tper_solicitudes s
                      where sm.nsolicitud = s.nsolicitud and not (s.situacion = 'B' and s.cestado='AP' and
                             sm.anulacionresuelta='S') and sm.nsolicitud = $nsol and sm.anulacionresuelta = '$resueltas'");

          if($listaSolicitudes<0) return -1;
          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameSolicitudesModificadas');
          return 0;
}

}
?>
```

– Clase SolicitudesBorradas:

```php
<?php

class SolicitudesBorradas{



        /**
         * fAltaSolicitudesBorradas
         * @param integer $pnsolicitud
         * @param string $pnregpgv
         * @param string $pcclase
         * @param date $pfechaIni
         * @param date $pfechaFin
         * @param varchar $pproceso
         * @return integer
         */
public static function f_altaSolicitudesBorradas($pnsolicitud,$psituacion,$pnregpgv,$pcclase,$pfechaIni,$pfechaFin,
                                                                                                $pproceso,$con){

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_altaSolicitudesBorradas');
        // Insertamos la solicitudes borradas de TPER_SOLICITUDES
        // siendo UsuariosQueSolicitan.nregpgv el nregpgv del usuario que se valida, es decir, que realiza el borrado de la
           solicitud y sysdate() la fecha y hora del sistema

        $UsuariosQueSolicitan = IgepSession::dameDatosUsuario();
        $nrpborra = $UsuariosQueSolicitan['nrp'];

        $fborrado = new gvHidraTimestamp();
        $str_fborrado = "'".$con->prepararFecha($fborrado)."'";

        if($pfechaIni==null)
                $str_fechaIni = "null";
        else
                $str_fechaIni = "'".$con->prepararFecha($pfechaIni)."'";
        if($pfechaFin==null)
                $str_fechaFin = "null";
        else
                $str_fechaFin = "'".$con->prepararFecha($pfechaFin)."'";

        $solicitudes = $con->operar( " INSERT INTO TPER_SOLICITUDESBORRADAS   (nsolicitud, nregpgv, cclase, fecha_ini,
                                                                        fecha_fin, nrpborra, fborrado, proceso)
                                    VALUES ($pnsolicitud, '$pnregpgv', '$pcclase', $str_fechaIni, $str_fechaFin,
                                            '$nrpborra', $str_fborrado, '$pproceso'); ");
        if($solicitudes<0) return -1;
        return 0;
        IgepDebug::setDebug(DEBUG_USER,'FIN f_altaSolicitudesBorradas');
}

}
?>
```

– Clase SolicitudesYPermisos:

```php
<?php

class SolicitudesYPermisos{

    function SolicitudesYPermisos(){
    }

    /**
     * Función que devuelve el último día del mes
     * @return integer
     * */
function ultimoDiaMes($mesini, $anyoini){

          IgepDebug::setDebug(DEBUG_USER,'INICIO ultimoDiaMes');
          $mesini=($mesini+1)%12;
          $dia = new gvHidraTimestamp($mesini.'/01/'.$anyoini);
          $dia->setTime(0,0,0);
          $dia->subDays(1);
          IgepDebug::setDebug(DEBUG_USER,'FIN ultimoDiaMes');
          return $dia->format('d');
}


          /**
           * FunciÃ³n que devuelve FunciÃ³n que devuelve para un intervalo de fechas el número de meses naturales más los días
           * restantespara un intervalo de fechas el número de meses naturales más los días restantes
           *
           * @return integer
           */
function f_calculaMesesNaturalesMasDias($ppfechaini,$ppfechafin,&$pmesesnaturales,&$prestodias){

          // Implementar la función de manera que para un intervalo de tiempo entre
          // pfechaini y pfechafin se devuelva en pmesesnaturales el nÃºmero de meses naturales del intervalo
          // y en prestodias los días restantes

          // Para realizar el cálculo nos vamos a basar en lo que dice la legislaciÃ³n sobre los plazos de meses naturales:
          // Si el día de la fecha de inicio no existe en el mes de finalización del plazo, se toma el último día del mes

          // La idea será  primero calcular los meses naturales entre pfechaini y pfechafin. A continuación, calcular
          // el resto de días naturales, cumpliendo la regla de que si el día de pfechaini no existe en el
          // mes de pfechafin, y pfechafin ocurre en final de mes, entonces se utiliza para los cÃ¡lculos
          // el último día del mes en pfechafin, para así cumplir el criterio indicado en la legislación al respecto.



          // El algoritmo será el siguiente:
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_calculaMesesNaturalesMasDias');
          $pfechaini = clone $ppfechaini;
          $pfechafin = clone $ppfechafin;


          $var_diaini = $pfechaini->format('d');
          $var_mesini = $pfechaini->format('m');
          $var_anyoini = $pfechaini->format('Y');

          $var_diafin = $pfechafin->format('d');
          $var_mesfin = $pfechafin->format('m');
          $var_anyofin = $pfechafin->format('Y');

          // Cálculo de meses naturales
          $pmesesnaturales = $var_mesfin - $var_mesini + 12 * ($var_anyofin - $var_anyoini);

          // Calculo de resto de días naturales, ajustando el valor de meses naturales si llega el caso

          // Si el día de pfechafin es el final del mes de pfechafin entonces
          // si el día de pfechaini no existe en el mes de pfechafin
          // entonces se cuenta desde el día 1 del mes siguiente de pfechaini

          if($var_diafin == SolicitudesYPermisos::UltimoDiaMes($var_mesfin, $var_anyofin)){
                    if($var_diaini > SolicitudesYPermisos::UltimoDiaMes($var_mesfin, $var_anyofin)){
                              //var_diaini deberá cambiarse a var_diafin + 1 para que prestodias valga 0,
                              // porque ya hemos contado el mes en pmesesnaturales

                              $var_diaini = $var_diafin + 1;
                    }
          }


          $prestodias = $var_diafin- $var_diaini +1;


          if($prestodias < 0){
                    // Decrementamos los meses naturales calculados y ajustamos el resto de dÃas
                    $pmesesnaturales = $pmesesnaturales - 1;
                    $prestodias = SolicitudesYPermisos::ultimoDiaMes($var_mesini, $var_anyoini) + $prestodias;
          }elseif ($prestodias == SolicitudesYPermisos::ultimoDiaMes($var_mesfin, $var_anyofin) AND $var_diafin ==
                    $prestodias){
                    //Estamos calculando justo desde el primer día del mes inicial al último día del mes final, sumamos un mes
                    $pmesesnaturales = $pmesesnaturales + 1;
                    $prestodias = 0;
          }

          IgepDebug::setDebug(DEBUG_USER,'FIN f_calculaMesesNaturalesMasDias');
          return 0;
}
```

```
    /**
     * Obtiene el nº de meses naturales de permiso de una clase que ha disfrutado el usuario en un año,
     * devolviendo el total de meses naturales y el resto de días
     * @return int
     */
function f_calcularMesesNaturalesPermisoAnyo($pPeriodo,&$pmesesnat,&$prestodias,$pcclase,$ptipo,$paltaAnula,$pnregpgv ){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_calcularMesesNaturalesPermisoAnyo');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);

          if ($ptipo == 'S'){// Si es una solicitud

                    if($paltaAnula=='ALTA')
                              $res=$con->consultar(" SELECT fecha_ini,fecha_fin
                                                  FROM tper_solicitudes
                                                  WHERE nregpgv = '$pnregpgv' and cclase = '$pcclase' and
                                                            periodo = $pPeriodo and
                                                             (  NOT ((cestado in ('NR','NI')) OR
                                                                      (situacion = 'A' and cestado='NP' and
                                                                                ffirmaresolucion is not null) OR
                                                                      (situacion = 'B' and cestado='AP' and
                                                                                ffirmaresolucionanula is not null))

                                                              ) and
                                                            fecha_fin is not null", array(
                                                  'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));
                    else
                              $res=$con->consultar(" SELECT fecha_ini,fecha_fin
                                                  FROM tper_solicitudes
                                                  WHERE nregpgv = '$pnregpgv' and   cclase = '$pcclase' and
                                                            periodo = $pPeriodo and
                                                             (  NOT ((cestado in ('NR','NI')) OR
                                                              (situacion = 'A' and cestado='NP' and
                                                                        ffirmaresolucion is not null) OR
                                                              (situacion = 'B' and cestado='AP' and
                                                                        ffirmaresolucionanula is not null))
                                                                        ) and
                                                            situacion IN ('X','B') and
                                                            fecha_fin is not null", array(
                                                  'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));



          }
          else{ // Si es un permiso
                              $res=$con->consultar(" SELECT fecha_ini,fecha_fin
                                                  FROM tper_permisos
                                                  WHERE nregpgv = '$pnregpgv' and
                                                  cclase = '$pcclase' and
                                                  periodo = $pPeriodo and
                                                  fecha_fin is not null", array(
                                                  'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));

          }

          if($res<0)return -1;

          $pmesesnat = 0;
          $prestodias = 0;

          // Si solo hay un intervalo de LIP, no se añaden los días de resto.
          // Esto se hace para evitar que salga un mes natural cuando se piden 30 días de LIP en un mes de 31 días

          $numIntervalosLIP = 0;

          if(is_array($res))
                    foreach($res as $indice=>$valor){
                              if(!empty($valor['fecha_fin'])){
                                        SolicitudesYPermisos::f_calculaMesesNaturalesMasDias($valor['fecha_ini'],
                                                  $valor['fecha_fin'], $delta_pmesesnat, $delta_prestodias);
                              $pmesesnat += $delta_pmesesnat;
                              $prestodias += $delta_prestodias;
                              $numIntervalosLIP = $numIntervalosLIP + 1;

                              }

                    }

                    if($numIntervalosLIP > 1){
                              // Si acumulamos más de 30 días de resto, aÃ±adimos un mes natural por cada 30 días que salen de
                              resto
                              $pmesesnat += floor($prestodias/30);
                    }


                    // Se descuentan los días que se han añadido antes a los meses naturales
                    $prestodias -= floor($prestodias / 30)*30;
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularMesesNaturalesPermisoAnyo');
                    return 0;

    }
```

```
	/**
	* Obtiene el nÃºmero de meses naturales de permiso que el usuario ha disfrutado en un aÃ±o mÃ¡s los que solicita,
	* devolviendo los meses naturales y el resto de dÃas naturales
		* @return integer
		*/
function f_mesesNaturalesPermiso($pPeriodo,$ppfechaInicio,$ppfechaFin,$pcclase,&$ptotalmesesnat,&$pmesesnat,
                                        &$ptotalrestodias,&$prestodias,$ptipo,$paltaAnula,$pnregpgv,$claseLlamada ){
        //Parámetros de salida:

        // ptotalmesesnat :    Total de meses naturales ya solicitados mÃ¡s los que se están solicitando actualmente
        // pmesesnat :         Meses naturales que se están solicitando actualmente
        // ptotalrestodias :   Total del resto de días naturales ya solicitados más los que se están solicitando actualmente
        // prestodias :        Resto de días naturales que se están solicitando actualmente
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_mesesNaturalesPermiso');

        $pfechaInicio = clone $ppfechaInicio;
        if(!is_null($ppfechaFin))
                $pfechaFin = clone $ppfechaFin;

        $ret=SolicitudesYPermisos::f_calcularMesesNaturalesPermisoAnyo($pPeriodo, $ptotalmesesnat,$ptotalrestodias,$pcclase,
                                                                        $ptipo,$paltaAnula,$pnregpgv);
        if($ret<0){
                //No se han recuperado los datos
                $ptotalmesesnat = null;
                $pmesesnat = null;
                $ptotalrestodias = null;
                $prestodias = null;
                IgepDebug::setDebug(DEBUG_USER,'FIN f_mesesNaturalesPermiso');
                return -1;
        }

        //Si el permiso solicitado es el que paso como parámetro

        // Para no sumar los meses que nos estamos pidiendo cuando es una anulación
        if($claseLlamada==$pcclase AND $paltaAnula == 'ALTA'){

                //Obtenemos el total de meses naturales y del resto de días del permiso que la persona solicita
                if(!empty($pfechaFin)){
                        $ret=SolicitudesYPermisos::f_calculaMesesNaturalesMasDias($pfechaInicio, $pfechaFin, $pmesesnat,
                                                                                $prestodias);

                }
                else{
                        $pmesesnat = 0;
                        $prestodias = 0;
                }

        }
        else{
                // No se están pidiendo en este momento días para el tipo de permiso que estamos calculando los días
                // No se incrementa el total ya calculado de lo que se solicitá anteriormente
                $pmesesnat = 0;
                $prestodias = 0;
        }


        if($ret<0){
                $ptotalmesesnat = null;
                $pmesesnat = null;
                $ptotalrestodias = null;
                $prestodias = null;
                IgepDebug::setDebug(DEBUG_USER,'FIN f_mesesNaturalesPermiso');
                return -1;
        }
        else{
                // A los valores calculados, aÃ±adimos los datos obtenidos en el permiso actual
                if(is_null($pmesesnat))$pmesesnat=0;
                if(is_null($prestodias))$prestodias=0;

                $ptotalrestodias = $ptotalrestodias + $prestodias;

                // Si acumulamos más de 30 días de resto, añadimos un mes natural por cada 30 días que salen de resto
                $ptotalmesesnat = $ptotalmesesnat + $pmesesnat + floor($ptotalrestodias / 30);

                // Se descuentan los días que se han añadido antes a los meses naturales
                $ptotalrestodias = $ptotalrestodias - floor($ptotalrestodias / 30)*30;

        }
        IgepDebug::setDebug(DEBUG_USER,'FIN f_mesesNaturalesPermiso');
        return 0;

}//Fin f_mesesNaturalesPermiso


    /**
         * Alta de solicitudes de permisos y licencias, y permisos y licencias
         * @return array ()
         *
         * */
function f_altaSoliPer($nregpgv,$clase,$pPeriodo,$pfecha_Ini,$pfecha_Fin,$pObservaciones,$pFnacimientomenor){

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_altaSoliPer');
        $alta['nregpgv']=$nregpgv; //usuario que solicita
        $alta['cclase']=$clase; //motivo (clase de permiso) introducido
        $alta['periodo']=$pPeriodo; //periodo introducido
        $alta['observaciones']=$pObservaciones; // observaciones introducidas
        $alta['fecha_ini']=clone $pfecha_Ini; //fecha inicio de la solicitud
        $alta['justificante']='N';   // Tanto en el alta de solicitudes como de permisos, el campo requiere justificante
        //SI ClasesDePermisos.cclase='RJG' O ClasesDePermisos.cclase = 'RJL' O ClasesDePermisos.cclase = 'AJL'
        $alta['fnacimientomenor'] = $pFnacimientomenor;
        if($pfecha_Fin!=null)
```

```
                        $alta['fecha_fin']=clone $pfecha_Fin;
                else
                        $alta['fecha_fin']=null;
        IgepDebug::setDebug(DEBUG_USER,'FIN f_altaSoliPer');
        return $alta;
}


        /**
        * Devuelve la letra del dÃa de la semana correspondiente a la fecha pasada por parÃ¡metro
         * @return char
         *
         * */
function f_calcularDiaSemana ($pfecha){

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_calcularDiaSemana');
        $fecha = clone $pfecha;
        $fecha= $con->prepararFecha($fecha);
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);
        $dia=$con->consultar("SELECT EXTRACT(DOW FROM TIMESTAMP '$fecha')");
        IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularDiaSemana');
        return $dia['0'];
}


function f_calcularSolapamientoSoliPer(&$resultado,$ppfecha_Ini,$ppfecha_Fin,$var_desc_clase,&$phaySolapamiento,$palta,
                                                                                                $clase,$nregpgv ){

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_calcularSolapamientoSoliPer');

        $pfecha_Ini = clone $ppfecha_Ini;
        if($ppfecha_Fin!=null)
                $pfecha_Fin = clone $ppfecha_Fin;

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        //  Cambiamos la conexión a persistente para reutilizar la creada anteriormente, y así acceder a los cambios de una
        transacción
        $con = new IgepConexion($g_dsn, true);


        //Transformo a fecha para BD
        $pfecha_Ini = $con->prepararFecha($pfecha_Ini);
        $pfecha_Fin = $con->prepararFecha($pfecha_Fin);


        $phaySolapamiento = false;

        //Primero se comprueba si la solicitud actual es de reducción, en cuyo caso se puede solapar con cualquier otra
        //salvo que sea del mismo tipo
        $datosMotivo = $con->consultar("SELECT * FROM TPER_COD_PERMISOS  WHERE cclase ='".$clase."' ");

        if ($datosMotivo['0']['tipo']=='R' OR $datosMotivo['0']['tipo']=='H'){

                // Comprobar si ya existe otra solicitud del mismo tipo en las mismas fechas
                 if(is_null($pfecha_Fin)or empty($pfecha_Fin))     {

                        $var_hay_otra_solicitud= $con->consultar("SELECT COUNT(*) FROM TPER_SOLICITUDES" .
                                                " WHERE cclase = '".$clase."' and nregpgv = '".$nregpgv."'" .
                                                        " AND (fecha_fin >= '".$pfecha_Ini."' OR fecha_fin is null) and
                                                         NOT ((cestado in ('NR','NI')) OR
                                                (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR

              (situacion in ('B','X') and cclase in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')) OR
              (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null and
              cclase not in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')))");


                        if($var_hay_otra_solicitud<0){
                                return -1;
                        }
                }
                else{
                         $var_hay_otra_solicitud= $con->consultar("SELECT COUNT(*) FROM TPER_SOLICITUDES" .
                                                " WHERE cclase = '".$clase."' and nregpgv = '".$nregpgv."' and" .
                                                " not(fecha_ini > '".$pfecha_Fin."' or (fecha_fin < '".$pfecha_Ini."' and
                                                fecha_fin is not null))and NOT ((cestado in ('NR','NI')) OR

              (situacion in ('B','X') and cclase in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')) OR
              (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null and
              cclase not in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')))");

                        if($var_hay_otra_solicitud<0){
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularSolapamientoSoliPer');
                                        return -1;
                         }
                }


                if ($var_hay_otra_solicitud['0']['count']==0){//si no hay otra solicitud
                         // Ahora buscamos los solapamientos entre reducciones no solicitables (todas menos RJL) u
                        horarios
                        if(is_null($pfecha_Fin)OR empty($pfecha_Fin)){
                                $var_hay_otra_solicitud = $con->consultar(" SELECT COUNT(*)
                                        FROM TPER_PERMISOS
                                          WHERE cclase = '$clase' and nregpgv = '$nregpgv' and
                                                  (fecha_fin >= '$pfecha_Ini' or fecha_fin is null) and
                                                   ('$palta' = 'S' or fecha_ini <> '$pfecha_Ini')");
                                if ($var_hay_otra_solicitud<0){
                                        IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularSolapamientoSoliPer');
                                        return -1;
```

```
                                        }
                                }
                                else{
                                        $var_hay_otra_solicitud = $con->consultar(" SELECT COUNT(*)
                                                FROM TPER_PERMISOS
                                                 WHERE cclase = '$clase' and nregpgv = '$nregpgv' and
                                                not (fecha_ini > '$pfecha_Fin' or (fecha_fin < '$pfecha_Ini' and
                                                fecha_fin is not null)) and('$palta' = 'S' or fecha_ini <> '$pfecha_Ini')");
                                        if ($var_hay_otra_solicitud<0){
                                                IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularSolapamientoSoliPer');
                                                return -1;
                                        }
                                }

                        }

                        if($var_hay_otra_solicitud['0']['count']>0){
                                //No se puede hacer la solicitud actual ya que existe otra del mismo tipo al mismo tiempo
                                $resultado['tipo']='P';
                                $resultado ['mensaje'] = "No puede solicitar el permiso porque ya ha realizado otra" .
                                                " solicitud de ".$var_desc_clase." en el mismo intervalo de tiempo.";
                                $phaySolapamiento = true;
                                IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularSolapamientoSoliPer');
                                return 0;
                        }

        }else{

                        // Sino comprobar si ya existe otra solicitud de cualquier tipo en las mismas fechas excepto
                        // las reducciones (tipo R) u horarios (tipo H)

                        //   UsuariosInteresados.nregpgv :   usuario interesado introducido por pantalla



                        if(is_null($pfecha_Fin)or empty($pfecha_Fin)){
                        $haydatos=$con->consultar("
                                        SELECT sol.fecha_ini, sol.fecha_fin, cper.dclase
                                         FROM TPER_SOLICITUDES sol, TPER_COD_PERMISOS cper
                                         WHERE sol.cclase = cper.cclase and
                                         sol.nregpgv = '$nregpgv'and(cper.tipo not in ('R','H')) and
                                        (sol.fecha_fin >= '$pfecha_Ini' or sol.fecha_fin is null) and
                                        NOT ((sol.cestado in ('NR','NI')) OR
                                        (sol.situacion = 'A' and sol.cestado='NP' and sol.ffirmaresolucion is not null) OR

                                        (sol.situacion in ('B','X') and
                                sol.cclase in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')) OR
                                (sol.situacion = 'B' and sol.cestado='AP' and sol.ffirmaresolucionanula is not null and
                                sol.cclase not in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')))    and '$clase' <>'IT'");

                        }
                        else{
                                $haydatos=$con->consultar("
                                         SELECT sol.fecha_ini, sol.fecha_fin, cper.dclase
                                        FROM TPER_SOLICITUDES sol, TPER_COD_PERMISOS cper
                                        WHERE sol.cclase = cper.cclase and sol.nregpgv = '$nregpgv' and  (cper.tipo not in
                                        ('R','H')) and (sol.fecha_ini <= '$pfecha_Fin' ) and
                         (sol.fecha_fin >= '$pfecha_Ini' or sol.fecha_fin is null) and
                          NOT ((sol.cestado in ('NR','NI')) OR
                        (sol.situacion = 'A' and sol.cestado='NP' and sol.ffirmaresolucion is not null) OR

                        (sol.situacion in ('B','X') and
                        sol.cclase in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')) OR
                        (sol.situacion = 'B' and sol.cestado='AP' and sol.ffirmaresolucionanula is not null and
                        sol.cclase not in ('VAC','ADI','VV6','AAP','EFP','LIP','LEF','LPP','AIN')))

                    and '$clase' <>'IT'");
                        }

                        if ($haydatos<0)return -1;

                        if (count($haydatos)<=0){

                                // Se buscan solapamientos con los permisos no solicitables (excepto VAC con IT, que si se
                                permite solaparse)

                                if(is_null($pfecha_Fin)or empty($pfecha_Fin)){

                                        $haydatos=$con->consultar("
                                                SELECT per.fecha_ini, per.fecha_fin, cper.dclase
                                                 FROM TPER_PERMISOS per, TPER_COD_PERMISOS cper
                                                 WHERE per.cclase = cper.cclase and
                                                          cper.solicitable = 'N' and
                                                          per.nregpgv = '$nregpgv' and
                                                         (cper.tipo not in ('R','H')) and
                                                         (per.fecha_fin >= '$pfecha_Ini' or per.fecha_fin is null) and
                                                          not ('$clase'<>'IT' and per.cclase='IT') and
                                        ('$palta' = 'S' or per.fecha_ini <> '$pfecha_Ini' or per.cclase <> '$clase')");

                                }else{
                                        $haydatos=$con->consultar("
                                                 SELECT per.fecha_ini, per.fecha_fin, cper.dclase
                                                FROM TPER_PERMISOS per, TPER_COD_PERMISOS cper
                                                 WHERE per.cclase = cper.cclase and
                                                          cper.solicitable = 'N' and
                                                          per.nregpgv = '$nregpgv' and
                                                          (cper.tipo not in ('R','H')) and
                                                          (per.fecha_ini <= '$pfecha_Fin' ) and
                                                         (per.fecha_fin >= '$pfecha_Ini' or per.fecha_fin is null) and
                                                          not ('$clase'<>'IT' and per.cclase='IT') and
                                        ('$palta' = 'S' or per.fecha_ini <> '$pfecha_Ini' or per.cclase <> '$clase')");
                                }
```

```
                    if ($haydatos<0)return -1;
                }


                if (count($haydatos)>0){

                    if(!is_null($haydatos['0']['fecha_fin'])){
                    //No se puede hacer la solicitud actual ya que existe otra que no es de reducción durante el
                    mismo intervalo de tiempo
                    $resultado['tipo']='P';
                    $resultado ['mensaje'] ="No puede solicitar el permiso porque ya ha realizado la solicitud
                                            ".$haydatos['0']['dclase']." entre el
                                            ".$haydatos['0']['fecha_ini']." y el ".$haydatos['0']['fecha_fin']." ";
                    $phaySolapamiento = true;
                     IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularSolapamientoSoliPer');
                     return 0;

                    }else{
                    //No se puede hacer la solicitud actual ya que existe otra que no es de reducción durante el
                    mismo intervalo de tiempo
                    $resultado['tipo']='P';
                    $resultado ['mensaje'] ="No puede solicitar el permiso porque ya ha realizado la solicitud
                                            ".$haydatos['0']['dclase']." desde el ".$haydatos['0']['fecha_ini'];

                    $phaySolapamiento = true;
                    }
                }
            }
            IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularSolapamientoSoliPer');
            return 0;

}//FIN f_calcularSolapamientoSoliPer



    /**
             * Realiza las comprobaciones necesarias para ver si la solicitud es o no válida.
             * @return integer
             *
             * */
function f_validarSoliPer($pPeriodo,$ppfechaInicio,$ppfechaFin,$clase,& $resultado, & $diasTrabajados,& $pdias,
                                & $pdiaslabo, $ptipo, & $pultimo,& $ptotaldiasadi,& $ptotal_dias,&
                            $ptotal_dias_labo,&$diasTrabajadosVV6,$nreg, $claseLlamada){

            IgepDebug::setDebug(DEBUG_USER,'INICIO f_validarSoliPer');
            $pfechaInicio = clone $ppfechaInicio;
            if($ppfechaFin!=null)
            $pfechaFin = clone $ppfechaFin;

            $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
            // Cambiamos la conexión a persistente para reutilizar la creada anteriormente, y así acceder a los cambios de una
            transacción
            $con = new IgepConexion($g_dsn, true);


            // Si la fecha final introducida es menor que la que se recibe como parámetro
            // Esta acción se controla en la interfaz


            $diasPermiso = SolicitudesYPermisos::f_diasPermiso($pPeriodo,$pfechaInicio,$pfechaFin,$clase,$ptotal_dias,

                            $ptotal_dias_labo, $pdias, $pdiaslabo, $ptipo, $pultimo, $ptotaldiasadi,$nreg, $claseLlamada);

            if($diasPermiso<0){
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                    return $diasPermiso;//"Error al calcular los días de permiso"
            }


            // Ver si la persona ya está de baja, o bien no ha trabajado todo el año, teniendo en cuenta si es o no de puertos
            $esPuertos = $con->consultar("SELECT count(depuertos) FROM vper_ocupacion WHERE nregpgv = '".$nreg."' and depuertos
                                                                                                    = 'S'");
            if ($esPuertos<0){
                    IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                    return -1;
            }
            $esPuertos=$esPuertos['0']['count'];

            if ($esPuertos){$depuertos='S';


                    // Primero se comprueba si la solicitud se está pidiendo cuando el interesado está trabajando

                    $var_fIni_trabajando =  $con->consultar("select count(*) from tpr2_puestos_trab_admon
                                                where nregpgv = '$nreg' and ftpos <= '$pfechaInicio'.
                                                and (fbajpad >= '$pfechaInicio' or fbajpad is null)");
                    if ($var_fIni_trabajando<0){
                            IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                            return -1;
                    }

                    if (is_null($pfechaFin)){
                    $var_fFin_trabajando['0']['count'] = 1;
                    }else{
                             $var_fFin_trabajando = $con->consultar("select count(*) from tpr2_puestos_trab_admon
                                                        where nregpgv = '$nreg' and ftpos <= '$pfechaFin'
                                                        and (fbajpad >= '$pfechaFin' or fbajpad is null)");
                             if ($var_fFin_trabajando<0){
                                    IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
```

```
                                        return -1;
                                }

                        }

                        if ( $var_fIni_trabajando['0']['count'] == 0 OR $var_fFin_trabajando['0']['count'] == 0){
                                $resultado['tipo'] = 'P';
                                $resultado['mensaje'] = "Error, esta persona ya ha cesado. No puede dar de alta solicitudes.";
                                IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                                return -1;
                        }

                        // Si el interesado está trabajando durante el periodo solicitado,
                        // se calcula más abajo el número de días trabajados en el año
                        // La fecha de toma de posesión no es relevante para este caso,
                        // así que damos un valor de control para que no falle
                        $ftoma['0']['ftpos'] = '01/01/2200';

                }
                else{$depuertos='N';
                        //No es de puertos -> Accedemos a la bd de "per" ORACLE
                        $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
                        $conOracle = new IgepConexion($g_oracle_dsn);

                        $ftoma = $conOracle->consultar("select ftpos as \"ftpos\" from tper_puestos_trab_admon where nregpgv =
                                        '".$nreg."' and fbajpad is null",array('DATATYPES'=>array('ftpos'=>TIPO_FECHA)));
                        if ($ftoma<0){
                                IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                                return -1;
                        }
                }


                if(count($ftoma)==0){//SI no encontrado
                        $resultado['tipo'] = 'P';
                        $resultado['mensaje'] = "Error, esta persona ya ha cesado. No puede dar de alta solicitudes..";
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                        return -1;;
                }


                $ret= UsuariosDePersonal::f_obtenerDiasTrabajados($nreg, $pPeriodo,$depuertos,$ftoma=$ftoma[0]['ftpos'],
                                                                                                        $diasTrabajados,$diasTrabajadosVV6);
                if($ret<0){
                        $resultado['mensaje'] = "Error de programaciÃ³n. Consultar responsable (cod:XAX)";
                        IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                        return -1;
                }


                IgepDebug::setDebug(DEBUG_USER,'FIN f_validarSoliPer');
                return 0;
}//f_validarSoliPer



        /**
        * Obtiene el nº de días de permiso de una clase que ha disfrutado el usuario en un año,
        *  devolviendo el total, los laborables y el primer día del último disfrutado
        * @return integer
        */
function f_calcularDiasPermisoAnyoAnulacion($pnregpgv,$pperiodo,$pcclase,&$pdiaslabo,&$pdias,&$nsolicitudesAnula,

                                                                                        &$fechasAnulacion,&$nsolicitudesAnulaResueltas ){
                IgepDebug::setDebug(DEBUG_USER,'INICIO f_calcularDiasPermisoAnyoAnulacion');
                $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
                $con = new IgepConexion($g_dsn, true);


                if($pcclase=='VAC'){


                        $query="SELECT fecha_ini, fecha_fin, fcmn_dias_festivos(fecha_ini,fecha_fin,o.cpro,o.cmun) as festivos
                          FROM TPER_SOLICITUDES s,VPER_OCUPACION o
                          WHERE s.nregpgv = '$pnregpgv' and cclase = '$pcclase' and periodo = $pperiodo and s.nregpgv = o.nregpgv and
                                                                ( NOT ((cestado in ('NR','NI')) OR
                                                                (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                                                (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))
                                                                 ) and
                                                                s.situacion IN ('X','B') and
                                                                fecha_fin is not null ";
                        $regSolis = $con->consultar($query,array( 'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA)));
                        if($regSolis<0 ){
                                IgepDebug::setDebug(DEBUG_USER,'FIN error f_calcularDiasPermisoAnyoAnulacion');
                                return -1;
                        }
                        $pdiaslabo=0;
                        $pdias=0;


                        foreach ($regSolis as $regSoli){
                                $fecha_ini = clone $regSoli['fecha_ini'];
                                $fecha_fin = clone $regSoli['fecha_fin'];

                                $sumaDias = round(($regSoli['fecha_fin']->getTimestamp() - $regSoli['fecha_ini']->getTimestamp()) /
                                                                                                                        (24*60*60))  + 1;
                                $sumaDiasLabo = $sumaDias - $regSoli['festivos'];

                                //SI regSoli.fecha_ini + 1 mes - 1 día = regSoli.fecha_fin
                                $fecha_ini->addMonths(1);
                                $fecha_ini->subDays(1);
```

```
                if(gvHidraTimestamp::cmp($fecha_ini, $fecha_fin) == 0){

                // Se ha solicitado un periodo de un mes natural
                // Las vacaciones se cuentan del mÃ¡ximo dÃas hÃ¡biles posibles ya que se ha pedido un mes natural de
                vacaciones
                        PermisosValidacion::f_LeerMinMaxDias('VAC', $nulo1, $ptotal_diaslabo_vac_max, $nulo2, $nulo3);

                        if($sumaDiasLabo >= $ptotal_diaslabo_vac_max){
                                $sumaDiasLabo = $ptotal_diaslabo_vac_max;
                        }
                }

                $pdiaslabo += $sumaDiasLabo;
                $pdias += $sumaDias;
        }



        }else{
                $pdiaslabo_pdias="
                                SELECT sum(fecha_fin - fecha_ini + 1 -
                                       fcmn_dias_festivos(fecha_ini,fecha_fin,o.cpro,o.cmun)) as pdiaslabo,
                                      sum(fecha_fin - fecha_ini + 1) as pdias
                                FROM TPER_SOLICITUDES s, VPER_OCUPACION o
                                WHERE s.nregpgv = '$pnregpgv' and
                                   cclase = '$pcclase' and
                                   periodo = $pperiodo and
                                   s.nregpgv = o.nregpgv and
                                   (
                                   NOT ((cestado in ('NR','NI')) OR
                                        (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                        (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))
                                   ) and
                                   s.situacion IN ('X','B') and
                                   fecha_fin is not null";

                $res1 = $con->consultar($pdiaslabo_pdias);
                if ($res1<0){
                        IgepDebug::setDebug(DEBUG_USER,'FIN error f_calcularDiasPermisoAnyoAnulacion');
                        return -1;
                }
                $pdiaslabo=$res1[0]['pdiaslabo'];
                $pdias=$res1[0]['pdias'];
        }


        // Vamos a obtener la lista de solicitudes de anulación y las fechas de estas anulaciones
        $nsolicitudesAnula_fechasAnulacion="
                              SELECT s.nsolicitud  as nsolicitudesanula , s.fecha_ini || ' - ' || s.fecha_fin as
                                                    fechasanulacion
                              FROM TPER_SOLICITUDES s, VPER_OCUPACION o
                              WHERE s.nregpgv = '$pnregpgv' and
                                 cclase = '$pcclase' and
                                 periodo = $pperiodo and
                                 s.nregpgv = o.nregpgv and
                                 (
                                 NOT ((cestado in ('NR','NI')) OR
                                      (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                      (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))
                                 ) and
                                 s.situacion IN ('X','B') and
                                 fecha_fin is not null";

        // Vamos a obtener la lista de solicitudes de anulaciÃ³n resueltas
        $nsolicitudesAnulaResueltas="
                              SELECT s.nsolicitud as nsolicitudesanularesueltas
                              FROM TPER_SOLICITUDES s, VPER_OCUPACION o
                             WHERE s.nregpgv = '$pnregpgv' and cclase = '$pcclase' and periodo = $pperiodo and
                                 s.nregpgv = o.nregpgv and situacion = 'B' and cestado='AP' and
                                 ffirmaresolucionanula is not null and  fecha_fin is not null";


        $res2 = $con->consultar($nsolicitudesAnula_fechasAnulacion);
        if ($res2<0) return -1;
        $res3 = $con->consultar($nsolicitudesAnulaResueltas);
        if ($res3<0) return -1;

        if(!is_array($nsolicitudesAnula))
        $nsolicitudesAnula=array();
        foreach($res2 as $solicitud)
                array_push($nsolicitudesAnula,$solicitud['nsolicitudesanula']);
        if(!is_array($fechasAnulacion))
                $fechasAnulacion=array();
        foreach($res2 as $fecha)
                array_push($fechasAnulacion,$fecha['fechasanulacion']);

        if(!is_array($nsolicitudesAnulaResueltas))
                $nsolicitudesAnulaResueltas=array();
        foreach($res3 as $solicitud)
                array_push($nsolicitudesAnulaResueltas,$solicitud['nsolicitudesanularesueltas']);

        $fechasAnulacion=implode(", ",$fechasAnulacion);

        IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularDiasPermisoAnyoAnulacion');
        return 0;
}
```

```
    /**
     * Obtiene el nº de días de permiso de una clase que ha disfrutado el usuario en un año,
     * devolviendo el total, los laborables y el primer día del último disfrutado
     * @return integer
     */

function f_calcularDiasPermisoAnyo($pPeriodo,$ppfechaInicio, &$pdiaslabo, &$pdias, &$pultimo, $pcclase, $ptipo,$popcionmenu,
                                                                                                                    $pnregpgv){
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_calcularDiasPermisoAnyo');
        if($ppfechaInicio!=null)
        $pfechaInicio = clone $ppfechaInicio;

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          //Cambiamos la conexión a persistente para reutilizar la creada anteriormente, y así acceder a los cambios de una
          transacción
        $con = new IgepConexion($g_dsn, true);


        if ($ptipo == 'solicitud' or $ptipo == 'S'){

            //Para VAC recalculamos el valor de cada intervalo si se pasa del mÃ¡ximo y dura un mes natural

                    if($pcclase=='VAC'){

                        $query="SELECT fecha_ini, fecha_fin, fcmn_dias_festivos(fecha_ini,fecha_fin,o.cpro,o.cmun) as festivos
                                    FROM TPER_SOLICITUDES s,VPER_OCUPACION o
                                    WHERE s.nregpgv = '$pnregpgv' and
                                       cclase = '$pcclase' and
                                       periodo = $pPeriodo and
                                       s.nregpgv = o.nregpgv and
                                       ( ";

                        if( $popcionmenu == 'ACR'){ // Firmas de solicitudes --> Anular como responsable
                              $query.=" NOT ((cestado = 'NI') OR ";
                        }else{
                              $query.=" NOT ((cestado in ('NR','NI')) OR ";
                        }

                                      $query.=" (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                  (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))
                             ) and
                             fecha_fin is not null ";

                        $regSolis = $con->consultar($query,array(
                                     'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA)));
                        if($regSolis<0 ){
                               IgepDebug::setDebug(DEBUG_USER,'FIN error f_calcularDiasPermisoAnyo');
                               return -1;
                        }

                        $pdiaslabo=0;
                        $pdias=0;

                        if(is_array($regSolis))
                              foreach ($regSolis as $regSoli){
                                        $fecha_ini= clone $regSoli['fecha_ini'];
                                        $fecha_fin= clone $regSoli['fecha_fin'];

                                        $sumaDias = round(($fecha_fin->getTimestamp() - $fecha_ini->getTimestamp() )  /
                                                                                                                    (24*60*60))  + 1;
                                        $sumaDiasLabo = $sumaDias - $regSoli['festivos'];

                                        //SI regSoli.fecha_ini + 1 mes - 1 día = regSoli.fecha_fin
                                        $fecha_ini->addMonths(1);
                                        $fecha_ini->subDays(1);
                                        if(gvHidraTimestamp::cmp($fecha_ini, $fecha_fin) == 0){
                                        // Se ha solicitado un periodo de un mes natural
                                        // Las vacaciones se cuentan del máximo días hábiles posibles ya que se ha pedido un
                                        mes natural de vacaciones
                                                PermisosValidacion::f_LeerMinMaxDias('VAC', $nulo1, $ptotal_diaslabo_vac_max,
                                                                                                                    $nulo2, $nulo3);

                                                if($sumaDiasLabo >= $ptotal_diaslabo_vac_max){
                                                      $sumaDiasLabo = $ptotal_diaslabo_vac_max;
                                                }
                                        }

                                        $pdiaslabo += $sumaDiasLabo;
                                        $pdias += $sumaDias;
                              }

                         }else{ // pcclase <> 'VAC'
                            $query="SELECT sum( fecha_fin - fecha_ini + 1 -
                                        fcmn_dias_festivos(fecha_ini,fecha_fin,o.cpro,o.cmun)) as \"totaldiaslabo\",
                                        sum(fecha_fin - fecha_ini + 1) as \"totaldias\"
                                    FROM TPER_SOLICITUDES s,VPER_OCUPACION o
                                    WHERE o.nregpgv = '".$pnregpgv."' and cclase = '".$pcclase."' and periodo = ".$pPeriodo."
                                        and s.nregpgv = o.nregpgv AND (         ";

                                        if($popcionmenu == 'ACR') // Firmas de solicitudes --> Anular como responsable
                                        $query.="NOT ((cestado = 'NI') OR ";
                                        else
                                        $query.="NOT ((cestado in ('NR','NI')) OR ";

                                  $query.="  (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                               (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))) and
                                                                                                        fecha_fin is not null";

                                  $res = $con->consultar($query);

                                  $pdiaslabo = $res[0]['totaldiaslabo'];
                                  $pdias = $res[0]['totaldias'];
```

```
                        }


                        //Transformo a fecha para BD
                        $pfechaInicioBD = $con->prepararFecha($pfechaInicio);
                        $query = "  SELECT max(fecha_ini) as \"ultimo\"
                                    FROM TPER_SOLICITUDES s
                                    WHERE nregpgv =  '".$pnregpgv."' and cclase = '".$pcclase."' and
                                          periodo = ".$pPeriodo." and NOT ((cestado in ('NR','NI')) OR
                                         (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                         (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))
                                             and fecha_fin is not null and
                                                    fecha_ini < '" .
                        $pfechaInicioBD .  "'";

                        $res2 = $con->consultar($query,array( 'DATATYPES'=>array('ultimo'=>TIPO_FECHA,)));


                        $pultimo = $res2[0]['ultimo'];

        }else{//Es permiso
            //Transformo a fecha para BD
            $pfechaInicioBD = $con->prepararFecha($pfechaInicio);
            //pdiaslabo,pdias
            $qry="SELECT sum( fecha_fin - fecha_ini + 1 - fcmn_dias_festivos(fecha_ini,fecha_fin,o.cpro,o.cmun)) as
                    \"totaldiaslabo\", sum(fecha_fin - fecha_ini + 1) as \"totaldias\" FROM tper_permisos p, VPER_OCUPACION o
                WHERE o.nregpgv = '".$pnregpgv."' and cclase = '".$pcclase."' and periodo = ".$pPeriodo." and p.nregpgv =
                            o.nregpgv and fecha_fin is not null";
            $res = $con->consultar($qry);

            $pdiaslabo = $res[0]['totaldiaslabo'];
                  $pdias = $res[0]['totaldias'];

            //pultimo
            $qry=" SELECT max(fecha_ini) as \"ultimo\"
                                                FROM tper_permisos p
                                                WHERE nregpgv = '$pnregpgv' and
                                    cclase = '$pcclase' and
                                    periodo = $pPeriodo and
                                    fecha_fin is not null and
                                fecha_ini < '$pfechaInicioBD'";
            $res2 = $con->consultar($qry,array( 'DATATYPES'=>array('ultimo'=>TIPO_FECHA,)));

            $pultimo = $res2[0]['ultimo'];
        }


        if($res<0 || $res2<0){
            IgepDebug::setDebug(DEBUG_USER,'FIN error f_calcularDiasPermisoAnyo');
            return -1;
        }
        else{


            IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularDiasPermisoAnyo');
            return 0;
        }
    }




    /**
     * Obtiene el número de días de permiso que el usuario ha disfrutado en un año más los que solicita,
     * devolviendo el total, los laborables, los adicionales y el primer día del último de ese permiso
     * @return integer
     */
function f_diasPermiso($pPeriodo, $ppfechaInicio, $ppfechaFin, $pcclase, &$ptotaldias, &$ptotaldiaslabo, &$pdias,
                    &$pdiaslabo, $ptipo, &$pultimo, &$ptotaldiasadi, $pnregpgv, $pclaseLlamada,$popcionmenu='AAA'){
        //Parámetros de salida:

        //ptotaldias:       Total de días ya solicitados más los que se están solicitando actualmente
        //ptotaldiaslabo:   Total de días laborables ya solicitados más los que se están solicitando actualmente
        //pdias:            Días que se están solicitando actualmente
        //pdiaslabo:        Días laborables que se están solicitando actualmente
        //pultimo:          Primer día del último permiso ya solicitado previamente
        //ptotaldiasadi:    Máximo número de días adicionales de que dispone el empleado

        //Conectamos con la BD
        IgepDebug::setDebug(DEBUG_USER,"INICIO f_diasPermiso");
        if($ppfechaInicio!=null)
          $pfechaInicio = clone $ppfechaInicio;
        if($ppfechaFin!=null)
          $pfechaFin = clone $ppfechaFin;

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');

          //Cambiamos la conexión a persistente para reutilizar la creada anteriormente, y así acceder a los cambios de una
                    transacción
        $con = new IgepConexion($g_dsn, true);



        $res = SolicitudesYPermisos::f_calcularDiasPermisoAnyo($pPeriodo,$pfechaInicio, $ptotaldiaslabo, $ptotaldias,
                                                               $pultimo, $pcclase, $ptipo,$popcionmenu, $pnregpgv);
        if($res==-1){ IgepDebug::setDebug(DEBUG_USER,'FIN f_diasPermiso');return -1;}
        //Si el permiso solicitado es el que paso como parámetro
        //SI ClasesDePermisos.cclase = pcclase
        if($pcclase==$pclaseLlamada){
```

```
                //Obtenemos el total de dias y el total de dias laborables del permiso que la persona solicita
                if(!is_null($pfechaFin)){
                        //total = fechaFin - fechaIni + 1;
                         $total = (round(($pfechaFin->getTimestamp() - $pfechaInicio->getTimestamp())/ (60 * 60 * 24))
                                                                                                                +1);
                }
                else
                        $total = 0;
            // Pasamos como parámetro a fcmn_dias_festivos la provincia y municipio del usuario que realiza la solicitud
            // a través de la vista VPER_OCUPACION
            $cproCmun = $con->consultar("SELECT cpro, cmun FROM vper_ocupacion WHERE nregpgv = '".$pnregpgv."'");

            if (count($cproCmun)==0){
                IgepDebug::setDebug(DEBUG_USER,'FIN f_diasPermiso');
                return -1;
            }


            //Transformo a fecha para BD
          $pfechaInicioBD = $con->prepararFecha($pfechaInicio);
          $pfechaFinBD = $con->prepararFecha($pfechaFin);
          if(!is_null($pfechaFin))
                $res = $con->consultar("SELECT
fcmn_dias_festivos(to_date('".$pfechaInicioBD."','dd/mm/yyyy'),to_date('".$pfechaFinBD."','dd/mm/yyyy'),'".$cproCmun[0]['cpro
'].."','".$cproCmun[0]['cmun']."')");
          else
                $res = $con->consultar("SELECT
fcmn_dias_festivos(to_date('".$pfechaInicioBD."','dd/mm/yyyy'),null,'".$cproCmun[0]['cpro'].."','".$cproCmun[0]['cmun']."')");
            if ($res==-1){
                return -1;
            }
            $festivos = $res['0']['fcmn_dias_festivos'];
        }else{

            // No se están pidiendo en este momento días para el tipo de permiso que estamos calculando los días
            // No se incrementa el total ya calculado de lo que se solicitó anteriormente
            $total = 0;
            $festivos = 0;
        }
        // A los valores calculados, añadimos los datos obtenidos en el permiso actual


        $ptotaldias = $ptotaldias + $total;

                // 18/11/2008 INICIO
                if($pclaseLlamada=='VAC'){
                        //SI pfechaInicio + 1 MES - 1 DIA = pfechaFin
                        $pfechaInicioClone= clone $pfechaInicio;
                        $pfechaInicioClone->addMonths(1);
                        $pfechaInicioClone->subDays(1);
                        if(gvHidraTimestamp::cmp($pfechaInicioClone, $pfechaFin) == 0){
                        // Se ha solicitado un periodo de un mes natural
                        // Las vacaciones se cuentan del máximo días hábiles posibles ya que se ha pedido un mes natural
                        de vacaciones
                                PermisosValidacion::f_LeerMinMaxDias('VAC', $nulo1, $ptotal_diaslabo_vac_max, $nulo2,
                                                                                                                $nulo3);
                                if(($total-$festivos) >= $ptotal_diaslabo_vac_max){
                                        $ptotaldiaslabo = $ptotaldiaslabo + $ptotal_diaslabo_vac_max;
                                }else{
                                        $ptotaldiaslabo = $ptotaldiaslabo + $total - $festivos;
                                }

                        }
                        else{
                                $ptotaldiaslabo = $ptotaldiaslabo + $total - $festivos;
                        }
                }else{
                        $ptotaldiaslabo = $ptotaldiaslabo + $total - $festivos;
                }

                // 18/11/2008 FIN

        // Obtenemos los días adicionales en caso de solicitar un permiso de vacaciones o días adicionales
        if (($pcclase == 'VAC') || ($pcclase == 'ADI')){
            //[#6456] 29/04/2010   // Obtener el valor del aÃ±o para el que se conocen los dias adicionales de vacaciones.
            Usuarios::f_obtenerAnyoDiasADI($panyodiasADI);

            if ($pPeriodo < $panyodiasADI){
                     $res = $con->consultar("SELECT diasadi_ant as \"totaldiasadi\" from tper_personas  where nregpgv =
                                                                                                        '".$pnregpgv."'");
            } else {
                    $res = $con->consultar("SELECT diasadi_act as \"totaldiasadi\" from tper_personas  where nregpgv =
                                                                                                        '".$pnregpgv."'");
            }



            if ($res==-1){
                return -1;
            }
            $ptotaldiasadi = $res[0]['totaldiasadi'];
        }
        $pdias = $total;
        $pdiaslabo = $total - $festivos;
        IgepDebug::setDebug(DEBUG_USER,'FIN f_diasPermiso');
        return 0;
    }//f_diasPermiso
```

```
        /**
     * Comprobamos si existe la clase de permiso en tper_solicitudes o en tper_permisos.
     */
        // 22/12/2009 INICIO
    function f_existeSoliPer($pcclase,$psoliPer){

                IgepDebug::setDebug(DEBUG_USER,"INI f_existeSoliPer");
                $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
      $conexion = new IgepConexion($g_dsn, true);
        if ($psoliPer=='S'){
                $res = $conexion->consultar("SELECT *
                                                                FROM tper_solicitudes
                                                                WHERE cclase = '".$pcclase."' ");

                if(count($res[0]))
                        return true;

            else
                        return false;

              }
        elseif($psoliPer=='P'){
                $res = $conexion->consultar("SELECT *
                                                        FROM tper_permisos
                                                        WHERE cclase = '".$pcclase."' ");

                if(count($res[0]))
                        return true;

            else
                        return false;

        }
        IgepDebug::setDebug(DEBUG_USER,"FIN f_existeSoliPer");

}   // fin f_existeSoliPer



   /**
     * Devuelve el orden de los días solicitados en las observaciones
     */
        // 04/11/2010 INICIO
function f_calcularOrdinalEnObservaciones($ptotalDiasLabo, $pdiasHabiles, $pPeriodo){

        IgepDebug::setDebug(DEBUG_USER,"INI f_calcularOrdinalEnObservaciones");
        // Añadir a las observaciones el orden de los días solicitados
        if ($pdiasHabiles == 1)
                $observaciones = $ptotalDiasLabo. "Â° del ".$pPeriodo.". ".$observaciones;
        else
                $observaciones =($ptotalDiasLabo - $pdiasHabiles + 1)."Â° - ".$ptotalDiasLabo."Â° del ".$pPeriodo.".
                                                                                                ".$observaciones;

        return $observaciones;
        IgepDebug::setDebug(DEBUG_USER,"FIN f_calcularOrdinalEnObservaciones");
}   // fin f_calcularOrdinalEnObservaciones



    /**
     * Devuelve el orden de los días solicitados en las observaciones cuando voy a dar de alta una solicitud, o bien, se
borra una solicitud
     */
function f_dameOrdinalParaObservaciones($pcontadorDias, $pdiasHabiles, $pperiodo){

        IgepDebug::setDebug(DEBUG_USER,"INI f_dameOrdinalParaObservaciones");
        // Añadir a las observaciones el orden de los días solicitados
        if ($pdiasHabiles == 1)
                $ordinalEnObservaciones = ($pcontadorDias+1). "Â° del ".$pperiodo.". ";
        else
                $ordinalEnObservaciones =($pcontadorDias+1)."Â° - ".($pcontadorDias + $pdiasHabiles)."Â° del
                                                                                        ".$pperiodo.". ";

        return $ordinalEnObservaciones;
        IgepDebug::setDebug(DEBUG_USER,"FIN f_dameOrdinalParaObservaciones");
}


     /**
     * Devuelve el campo de observaciones modificado con el ordinal correspondiente
     */
function f_dameObservaciones($pcontadorDias, $pdiasHabiles, $pperiodo, $pobservaciones){

        IgepDebug::setDebug(DEBUG_USER,"INI f_dameObservaciones");
        // Añadir el ordinal correspondiente en el campo observaciones
        $ordinalObservaciones =  SolicitudesYPermisos::f_dameOrdinalParaObservaciones($pcontadorDias, $pdiasHabiles,
                                                                                        $pperiodo);
        $cadenaABuscar = ("Â° del " + $pperiodo + ". ");
        $cadenaABuscar1 = ("Â° - " + " " + "Â° del " + $pperiodo + ". ");
        $cadenaEncontrada = strpos($cadenaABuscar,$pobservaciones);
        $cadenaEncontrada1 = strpos($cadenaABuscar1,$pobservaciones);

        if ($cadenaEncontrada > 0)
                $observaciones = str_replace($cadenaABuscar,$ordinalObservaciones,$pobservaciones);
        elseif ($cadenaEncontrada1 > 0)
                $observaciones =str_replace($cadenaABuscar1,$ordinalObservaciones,$pobservaciones);
        else
                $observaciones = $pobservaciones;
        return $observaciones;
        IgepDebug::setDebug(DEBUG_USER,"FIN f_dameObservaciones");
}
```

```
     /**
     * Obtiene los días hábiles entre dos fechas
     */
function f_dameDiasHabiles($pnregpgv,$ppfechaIni, $ppfechaFin, &$pdiasHabiles){

          IgepDebug::setDebug(DEBUG_USER,"INICIO f_dameDiasHabiles");

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          //Cambiamos la conexión a persistente para reutilizar la creada anteriormente, y así acceder a los cambios de una
          transacción
          $con = new IgepConexion($g_dsn, true);

        //Transformo a fecha para BD
          $pfechaInicioBD = $con->prepararFecha($ppfechaIni);
          $pfechaFinBD = $con->prepararFecha($ppfechaFin);
          if(!is_null($pfechaFin)){
                    //total = fechaFin - fechaIni + 1;
                     $total = (round(($pfechaFin->getTimestamp() - $pfechaIni->getTimestamp())/ (60 * 60 * 24)) +1);
                    // Pasamos como parámetro a fcmn_dias_festivos la provincia  y municipio del usuario que realiza la
                    // solicitud a través de la vista VPER_OCUPACION
                    $cproCmun = $con->consultar("SELECT cpro, cmun FROM vper_ocupacion WHERE nregpgv = '".$pnregpgv."'");

                    if (count($cproCmun)==0){
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_diasPermiso');
                              return -1;
                    }
                    $res = $con->consultar("SELECT  fcmn_dias_festivos(to_date('".$pfechaIniBD."','dd/mm/yyyy'),
                    to_date('".$pfechaFinBD."','dd/mm/yyyy'),'".$cproCmun[0]['cpro']."','".$cproCmun[0]['cmun']."')");

                    if ($res==-1){
                     return -1;
                    }
                    $festivos = $res['0']['fcmn_dias_festivos'];
          }else{
              $total = 0;
              $festivos = 0;
          }

          $pdiasHabiles = $total - $festivos;

          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameDiasHabiles');
          return 0;
}//f_dameDiasHabiles


     /**
     * Recalcula el ordinal de los días solicitados y/o disfrutados por un interesado en un determinado periodo, cuando se
está dando de alta una solicitud
     */

function f_recalcularOrdinalAlta($pnregpgv,$pperiodo,$pcclase,$pfechaIni,$pfechaFin,$pobservaciones ){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_recalcularOrdinalAlta');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          $listaSolicitudes=array();
          // Obtenemos la lista de solicitudes válidas (en caso de estar anuladas o no estar autorizadas, que no tengan fecha
          de resolución)
          // disfrutadas o no por el interesado para un determinado periodo y clase.
          Solicitudes::f_dameSolicitudesValidas($pnregpgv, $pperiodo, $pcclase, $listaSolicitudes);

          // Recorremos las solicitudes para actualizar el campo de observaciones
          $cuenta = 0;
          $altaCalculada = 'N';

          foreach ($listaSolicitudes as $solicitud){

                    $pdiasHabiles=0;

                    if ($solicitud['fecha_ini'] <= $pfechaIni){

                              // Obtener los días hábiles de la solicitud
                              SolicitudesYPermisos::f_dameDiasHabiles($pnregpgv,$solicitud['fecha_ini'],
                                                                      $solicitud['fecha_fin'], $pdiasHabiles);
                              // Incremento el contador pero no actualizo porque está ordenado correctamente
                              $cuenta = $cuenta + $pdiasHabiles;

                    }

                    $pdiasHabiles=0;
                    if ($solicitud['fecha_ini'] > $pfechaIni){
                              if ($altaCalculada ='N'){
                                    // Obtener los días hábiles de la solicitud que estoy dando de alta
                                    SolicitudesYPermisos::f_dameDiasHabiles($pnregpgv,$pfechaIni, $pfechaFin, $pdiasHabiles);

                                    // Calculo el ordinal de la solicitud que estoy dando de alta

                                        $solicitud['observaciones']=SolicitudesYPermisos::f_dameObservaciones($cuenta,
                                                                      $pdiasHabiles, $pperiodo, $pobservaciones);
                                        $cuenta = $cuenta + $pdiasHabiles;
                                        $altaCalculada = 'S';
                              }

                              // Obtener los días hábiles de la solicitud
                              $pdiasHabiles=0;
                              SolicitudesYPermisos::f_dameDiasHabiles($pnregpgv,$solicitud['fecha_ini'],
                                                                      $solicitud['fecha_fin'], $pdiasHabiles);

                              //Añadir el ordinal correspondiente en el campo observaciones
```

```
                        $solicitud['observaciones']=SolicitudesYPermisos::f_dameObservaciones($cuenta, $pdiasHabiles,
                                                                $pperiodo, $solicitud['observaciones']);

                        //Actualizamos el resultado obtenido en el campo observaciones
                        $res = $con->operar("UPDATE tper_solicitudes SET observaciones =
                        '".$solicitud['observaciones']."' WHERE nsolicitud = '".$solicitud['nsolicitud']."' ");

                        if ($res==-1){
                            return -1;
                        }

                        //Con el nº de la solicitud actualizamos en permisos el campo de observaciones
                        $res = $con->operar("UPDATE tper_permisos SET observaciones = '".$solicitud['observaciones']."'
                                  WHERE nsolicitud = '".$solicitud['nsolicitud']."' ");

                        if ($res==-1){
                           return -1;
                        }
                                  $cuenta = $cuenta + $pdiasHabiles;
                }

        }//fin foreach

        if ($altaCalculada ='N'){
                  // Obtener los días hábiles de la solicitud que estoy dando de alta
                  $pdiasHabiles=0;
                  SolicitudesYPermisos::f_dameDiasHabiles($pnregpgv,$pfechaIni, $pfechaFin, $pdiasHabiles);
                  // Calculo el ordinal de la solicitud que estoy dando de alta
                  $solicitud['observaciones']=SolicitudesYPermisos::f_dameObservaciones($cuenta, $pdiasHabiles, $pperiodo,
                                                                $pobservaciones);
        }


        IgepDebug::setDebug(DEBUG_USER,'FIN f_recalcularOrdinalAlta');
        return observaciones;
}


    /**
    * Recalcula el ordinal de los días solicitados y/o disfrutados por un interesado en un determinado periodo, cuando se ha
borrado una solicitud
    */

function f_recalcularOrdinalBorrado($pnregpgv,$pperiodo,$pcclase ){

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_recalcularOrdinalBorrado');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);

        $listaSolicitudes=array();

        // Obtenemos la lista de solicitudes válidas (en caso de estar anuladas o no estar autorizadas, que no tengan fecha
        de resolución) disfrutadas o no por el interesado para un determinado periodo y clase.

        Solicitudes::f_dameSolicitudesValidas($pnregpgv, $pperiodo, $pcclase, $listaSolicitudes);

        // Recorremos las solicitudes para actualizar el campo de observaciones
        $cuenta = 0;

        foreach ($listaSolicitudes as $solicitud){

                  $pdiasHabiles=0;

                  // Obtener los días hábiles de cada solicitud
                  SolicitudesYPermisos::f_dameDiasHabiles($pnregpgv,$solicitud['fecha_ini'], $solicitud['fecha_fin'],
                                                                $pdiasHabiles);

                  //Añadir el ordinal correspondiente en el campo observaciones
                  $solicitud['observaciones']=SolicitudesYPermisos::f_dameObservaciones($cuenta, $pdiasHabiles, $pperiodo,
                                                                $solicitud['observaciones']);


                  //Actualizamos el resultado obtenido en el campo observaciones
                  $res = $con->operar("UPDATE tper_solicitudes SET observaciones = '".$solicitud['observaciones']."' WHERE
                                                                nsolicitud = '".$solicitud['nsolicitud']."' ");

                  if ($res==-1){
                    return -1;
                  }


                //Con el nº de la solicitud actualizamos en permisos el campo de observaciones
                  $res = $con->operar("UPDATE tper_permisos SET observaciones = '".$solicitud['observaciones']."' WHERE
                                                                nsolicitud = '".$solicitud['nsolicitud']."' ");

                  if ($res==-1){
                    return -1;
                   }
                  $cuenta = $cuenta + $pdiasHabiles;


        }//fin foreach

         IgepDebug::setDebug(DEBUG_USER,'FIN f_recalcularOrdinalBorrado');
        return 0;
}
}

?>
```

– Clase ClasesDePermisos:

```php
<?php

class ClasesDePermisos{

    function ClasesDePermisos(){
    }

    /**
    * Obtiene si la clase es solicitable o no
    * @return array (retornoFuncion, error)
    */
public static function f_esSolicitable($clase){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_esSolicitable');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          $res = $con->consultar("SELECT solicitable FROM TPER_COD_PERMISOS WHERE cclase = '".$clase."' and solicitable='S'");
          if($res<0) {
                    $retorno['0']=$res;
                    $retorno ['error'] = "Error obteniendo permiso solicitable";
          }else
                    $retorno['0'] =  count($res);
          $retorno ['error']= "";
          IgepDebug::setDebug(DEBUG_USER,'FIN f_esSolicitable');
          return $retorno;
}

    /**
     * Devuelve la descripciÃ³n de la clase de permiso y si necesita justificante o no.
     */
public static function f_dameDatosClasePermiso($pcclase,&$pdclase,&$pjustificante){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameDatosClasePermiso');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          $res = $con->consultar("SELECT dclase, justificante
                    FROM TPER_COD_PERMISOS
                    WHERE cclase = '".$pcclase."'");
          if($res!=-1){
            $pdclase = $res[0]['dclase'];
            $pjustificante = $res[0]['justificante'];
            IgepDebug::setDebug(DEBUG_USER,'FIN f_dameDatosClasePermiso');
            return 0;
          }
          IgepDebug::setDebug(DEBUG_USER,'FIN error f_dameDatosClasePermiso');
          return -1;
}

          /**
           * Devuelve las clases de permisos que se resuelven por un trÃ¡mite determinado
           */
public static function f_clasesPermisosPorTramite($tramite){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_clasesPermisosPorTramite');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);

          $res = $con->consultar("SELECT cclase, dclase
                  FROM TPER_COD_PERMISOS
                  WHERE tramite = '".$tramite."'
                  ORDER BY orden");

          IgepDebug::setDebug(DEBUG_USER,'FIN f_clasesPermisosPorTramite');
          if($res!=-1)
                    return $res;
          else
                    return -1;
}

}//Fin clase ClasesDePermisos
?>
```

– Clase PermisosValidacion:

```php
<?php

class PermisosValidacion{

    function PermisosValidacion(){
    }

    /**
     * Devuelve la informaciÃ³n de una determinada regla de validaciÃ³n
     * @param pcodigo int
     * @return array
     */
    public static function f_dameReglaValidacion($pcodigo=0){
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameReglaValidacion');
          $sql=" SELECT control, orden, condicion, mensaje FROM tper_permisosvalidacion  WHERE codigo = $pcodigo";
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);
                  $reglaValidacion = $con->consultar($sql);
          if (is_array($reglaValidacion)){
                  $ret=$reglaValidacion[0];
          }else{
                  $ret= 0;
          }
          IgepDebug::setDebug(DEBUG_USER,'FIN f_dameReglaValidacion');
                  return $ret;
    }

    /***
     * Lee el mínimo y máximo de días a disfrutar de cada clase de permiso
     * @return integer
     */
function f_LeerMinMaxDias($pcclase,& $pmindias,& $pmaxdias,& $pdiasnaturales,&$punidad){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_LeerMinMaxDias');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          // Cambiamos la conexión a persistente para reutilizar la creada anteriormente, y así acceder a los cambios de una
          transacción
          $con = new IgepConexion($g_dsn, true);


          $diasAdiMaxMin = $con->consultar("select mindias, maxdias, diasnaturales, unidad " .
                                                  "from tper_cod_permisos where cclase='$pcclase'");
          if ($diasAdiMaxMin<0) {
                  return -1;
          }
          if (is_null($diasAdiMaxMin ['0']['mindias']))
                  $pmindias = 0;
          else
                  $pmindias = $diasAdiMaxMin ['0']['mindias'];

          if (is_null($diasAdiMaxMin ['0']['maxdias']))
                  $pmaxdias=999999;
          else
                  $pmaxdias = $diasAdiMaxMin ['0']['maxdias'];

          $pdiasnaturales = $diasAdiMaxMin ['0']['diasnaturales'];
          $punidad = $diasAdiMaxMin ['0']['unidad'];
          IgepDebug::setDebug(DEBUG_USER,'FIN f_LeerMinMaxDias');
          return 0;
}



    /**
    *  f_calcularTotalDiasVACyADIyVV6yAAP
    * @return integer
    */
    function f_calcularTotalDiasVACyADIyVV6yAAP($pnregpgv,$pperiodo,&$totalDiasVAC,&$totalDiasADI,&$anyoDiasADI,
                                                    &$totalDiasVV6,&$totalDiasAAP,$accion){
          IgepDebug::setDebug(DEBUG_USER,'INI f_calcularTotalDiasVACyADIyVV6yAAP');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conPos = new IgepConexion($g_dsn, true);
          $g_oracle_dsn = ConfigFramework::getConfig()->getDSN('g_oracle_dsn');
          $conOracle=new IgepConexion($g_oracle_dsn);
          //Obtenemos la fecha de toma de posesión del interesado para luego obtener los días trabajados
          //f_esDePuertos: Comprobamos si es usuario de puertos
          $res2 = $conPos->consultar('SELECT nregpgv
                                  FROM VPER_OCUPACION
                                  WHERE nregpgv = \''.$solicitud['nregpgv'].'\' AND
                                  depuertos = \'S\'');
          if($res2[0]['nregpgv']!='')
                  $esDePuertos = 'S';
          else
                  $esDePuertos = 'N';


          if($esDePuertos=='N'){

              $res=$conOracle->consultar("select ftpos as \"ftpos\"
                                  from tper_puestos_trab_admon
                            where nregpgv = '$pnregpgv' and fbajpad is null",array('DATATYPES'=>array('ftpos'=>TIPO_FECHA)));
              $ftoma=$res['0']['ftpos'];
```

```
		}else{
			$ftoma = new gvHidraTimestamp('2200/01/01');
			$ftoma->setTime(0,0,0);
		}


		UsuariosDePersonal:: f_obtenerDiasTrabajados($pnregpgv, $pperiodo, $esDePuertos, $ftoma, $diasTrabajados,
		                                             $diasTrabajadosVV6);
		UsuariosDePersonal::f_obtenerMunPro($pnregpgv, $cpro, $cmun);

		PermisosValidacion::f_CalcularMinMaxDias('VAC', $pperiodo, $cmun, $cpro, $nulo, $nulo2, $diasTrabajados, $nulo3,
		                    $totalDiasVAC,0, 0, $nulo4, $nulo5, 0, $diasTrabajadosVV6, $nulo6, $nulo7, $nulo8,  $pnregpgv,
		                    $noEsLaboral,$diasMaximoLocal);



		$varHoy1 = new gvHidraTimestamp();
		$varHoy1->setTime(0,0,0);

		$varHoy2 = new gvHidraTimestamp();
		$varHoy2->setTime(0,0,0);

		$ret=SolicitudesYPermisos::f_mesesNaturalesPermiso($pperiodo, $varHoy1,varHoy2,'LIP',$totalMesesNatLIP,$mesesNatLIP,

		                            $totalRestoDiasLIP,$restoDiasLIP,'S',$paltaAnula='ALTA',$pnregpgv,' ');

		if ($ret<0)return -1;

		if ($totalMesesNatLIP > 0)
			$totalDiasVAC = round($totalDiasVAC - $totalDiasVAC/12 * ($totalMesesNatLIP + $totalRestoDiasLIP/30));



		UsuariosDePersonal::f_obtenerTrienios($pnregpgv,$trienios);

		PermisosValidacion::f_CalcularMinMaxDias('VV6', $pperiodo, $cmun, $cpro, $nul1, $nul2, $diasTrabajados, $nul3,
		                    $totalDiasVV6,0, 0, $nul4, $nul5, 0, $diasTrabajadosVV6, $nul6, $nul7, $nul8, $pnregpgv,
		                    $noEsLaboral,$diasMaximoLocal);

		PermisosValidacion::f_CalcularMinMaxDias('AAP', $pperiodo, $cmun, $cpro, $nu1, $nu2, $diasTrabajados, $nu3,
		                    $totalDiasAAP,0, 0, $nu4, $nu5, $trienios, $diasTrabajadosVV6, $nu6, $nu7, $nu8, $pnregpgv,
		                    $noEsLaboral,$diasMaximoLocal);


		//Obtener el valor del año para el que se conocen los días adicionales de vacaciones
		Usuarios::f_obtenerAnyoDiasADI($anyoDiasADI);

		$anyoHoyObj=new gvHidraTimestamp(date('Y'));
		$anyoHoy=$anyoHoyObj->format('Y');

		if ($anyoDiasADI != $anyoHoy and $accion == 'P') {
			$totalDiasADI = '?';
		}elseif ($anyoDiasADI != $anyoHoy and $accion == 'A') {
			$res = $conPos->consultar("SELECT diasadi_act FROM tper_personas WHERE nregpgv = '$pnregpgv'");
			if (empty($res['0']['diasadi_act'])) $totalDiasADI = 0;
			else $totalDiasADI = $res['0']['diasadi_act'];
		}elseif ($anyoDiasADI == $anyoHoy and $accion == 'A') {
			$res = $conPos->consultar("SELECT diasadi_ant FROM tper_personas WHERE nregpgv = '$pnregpgv'");
			if (empty($res['0']['diasadi_ant'])) $totalDiasADI = 0;
			else $totalDiasADI = $res['0']['diasadi_ant'];
		}else {
			$res = $conPos->consultar("SELECT diasadi_act FROM tper_personas WHERE nregpgv = '$pnregpgv'");
			if (empty($res['0']['diasadi_act'])) $totalDiasADI = 0;
			else $totalDiasADI = $res['0']['diasadi_act'];
		}


		IgepDebug::setDebug(DEBUG_USER,'FIN f_calcularTotalDiasVACyADIyVV6yAAP');
		return 0;
}



	/**
	* Evaluar las reglas de validaciÃ³n
	      * @params
	      * @return integer
	      *
	      * */
function f_leerValidacion($codClase,$pfechaIni,$pfechaFin,$periodo,$diasTrabajados,$totalDiasLabo,$totalDiasAdi,
                          $totalDias,$nregpgv,$cmun,$cpro,& $resultado,$descClase,$justificante,&
                          $vacyAdiCompletos,& $totalDiasAdiSolicitados,& $totalDiasLaboAdiSolicitados,&
                          $totalDiasVac,& $totalDiasLaboVac,$tipo, $pfNacimientoMenor,$diasTrabajadosVV6, $dias,
                          $diasLabo,$observaciones, & $resultadoUsuario,$distancia, &$listaSolicitudesAnulacion,
                          &$listaSolicitudesAnulacionResueltas,&$totalDiasAnula,&$totalDiasLaboAnula,
                          &$reqJustificante,&$aportaJustificante,$distancia,$gradoParentesco,
                          $nregpgvQueSolicita){


		IgepDebug::setDebug(DEBUG_USER,'INI f_leerValidacion');

		if ($pfechaIni!=null)
			$fechaIni = clone $pfechaIni;
		if ($pfechaFin!=null)
			$fechaFin = clone $pfechaFin;
		if ($pfNacimientoMenor!=null)
			$fNacimientoMenor = clone $pfNacimientoMenor;

		$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
```

```
		$con = new IgepConexion($g_dsn, true);


		// Cálculo de dias mínimo y máximo a solicitar por permisos y solicitudes
		// Los días para VV6 y VAC se reducen proporcionalmente si no se ha trabajado todo el aÃ±o


		// Obtenemos fecha de antiguedad

		//Los trienios se calculan para AAP y no para VV6
		if($codClase == 'AAP'){

			// Obtenemos los trienios
			//LimitesDePermisos.ClasesDePermisos.UsuariosInteresados.f_obtenerTrienios(nregpgv,trienios);
			$ret = UsuariosDePersonal::f_obtenerTrienios($nregpgv,$trienios);


		}else{
			// Valor por defecto
			$trienios = 0;
		}
		// Añadimos los parámetros diasValidados, dias y diasLabo
		$retdias=PermisosValidacion::f_CalcularMinMaxDias($codClase, $periodo, $cmun, $cpro,
				$diasMinimo, $diasMaximo,$diasTrabajados, $diasNaturales, $maxDias, $totalDias, $totalDiasLabo,
				$totalDiasValidados, $descTipoDias, $trienios, $diasTrabajadosVV6, $diasValidados, $dias,
				$diasLabo,$nregpgv, $noEsLaboral,$diasMaximoLocal);


		if ($retdias<0) {
			IgepDebug::setDebug(DEBUG_USER,'FIN error1 f_leerValidacion');
			return -1;
		}

		// Cálculo de total de días de licencia interes particular acumulados en 3 aÃ±os
		if ($codClase == 'LIP' OR $codClase == 'EFP' ){
				//
-----------------------------------------------------------------------------------------------
				// CÃ¡lculo de total de días de licencia por interÃ©s particular acumulados en 3 años o
				// del total de días de licencia por estudios acumulados en 5 aÃ±os
				//
-----------------------------------------------------------------------------------------------

			$totalMesesAcumulados = 0;
			$totalRestoDiasAcumulados = 0;
			// Añadimos las nuevas variables para anulaciones a usar en las reglas de validación
			$mesesAnula = 0;
			$restoDiasAnula = 0;


			if($codClase == 'LIP')
				$numeroAnyos = 3;
			else
				$numeroAnyos = 5;

			for($anyo=1;$anyo<=$numeroAnyos;$anyo++){

				$ret=SolicitudesYPermisos::f_mesesNaturalesPermiso($periodo - $anyo +1,
					$fechaIni,$fechaFin,$codClase,
					$totalMesesNatAux,$mesesNatAux,$totalRestoDiasAux,$restoDiasAux,
					$tipo,'ALTA',$nregpgv,$claseLlamada=$codClase );
				if ($ret<0){
					IgepDebug::setDebug(DEBUG_USER,'FIN error2 f_leerValidacion');
					return -1;
				}
				$ret=SolicitudesYPermisos::f_mesesNaturalesPermiso($periodo - $anyo +1,
					$fechaIni,$fechaFin,$codClase,
					$totalMesesNatAuxAnula,$mesesNatAuxAnula,$totalRestoDiasAuxAnula,
				$restoDiasAuxAnula,$tipo,'ANUL',$nregpgv,$claseLlamada=$codClase );
				if ($ret<0){
					IgepDebug::setDebug(DEBUG_USER,'FIN error3 f_leerValidacion');
					return -1;
				}

				if($anyo == 1){
					$totalMesesAcumulados += $totalMesesNatAux;
					$totalRestoDiasAcumulados += $totalRestoDiasAux;

					// Obtenemos el valor de las nuevas variables
					$mesesAnula += $totalMesesNatAuxAnula;
					$restoDiasAnula += $totalRestoDiasAuxAnula;

				}else{
					// Para los periodos anteriores al actual no debemos sumar lo que se está solicitando
					ahora, por
					// lo tanto descontamos lo que devuelve en mesesNat y restoDias

					$totalMesesAcumulados += $totalMesesNatAux - $mesesNatAux;
					$totalRestoDiasAcumulados += $totalRestoDiasAux - $restoDiasAux;


					// Obtenemos el valor de las nuevas variables
					$mesesAnula += $totalMesesNatAuxAnula - $mesesNatAuxAnula;
					$restoDiasAnula += $totalRestoDiasAuxAnula - $restoDiasAuxAnula;

				}
			}

			// Se añaden tantos meses naturales como veces tengamos 30 días de resto
			$totalMesesAcumulados += floor($totalRestoDiasAcumulados / 30);

			// Se eliminan tantos días de resto como se hayan añadido a los meses naturales
```

```
                $totalRestoDiasAcumulados -= floor($totalRestoDiasAcumulados / 30)*30;

                // Se añaden tantos meses naturales como veces tengamos 30 días de resto
                $mesesAnula += floor($restoDiasAnula / 30);

                // Se eliminan tantos días de resto como se hayan aÃ±adido a los meses naturales
                $restoDiasAnula -= floor($restoDiasAnula / 30)*30;

                if($codClase == 'LIP'){
                        // Averiguamos si el interesado es interino o no, creando la nueva variable esInterino
                        UsuariosDePersonal::f_dameRelJur($nregpgv,$creljur);
                        if ($creljur == 'FI') $esInterino=TRUE;
                        else $esInterino=FALSE;

                }

        }//Fin if lip o efp

        if($codClase == 'LIP' OR $codClase == 'LEF' OR $codClase == 'LPP'){
                        //
-----------------------------------------------------------------------------------------------
                        // Cálculo de total de días laborables entre dos periodos de licencias sin retribución
                        //
-----------------------------------------------------------------------------------------------

                $fechaUltimaLicencia=$con->consultar("
                                                SELECT MAX(fecha_fin) as \"ultim\"
                                                FROM TPER_SOLICITUDES
                                                WHERE cclase in ('LIP','LEF','LPP') and
                                                   nregpgv = '$nregpgv' and
                                                     fecha_fin < '".$con->prepararFecha($fechaIni)."'",array(
                                                                'DATATYPES'=>array('ultim'=>TIPO_FECHA,)));
                if ($fechaUltimaLicencia<0){
                        IgepDebug::setDebug(DEBUG_USER,'FIN error4 f_leerValidacion');
                        return -1;
                }
                if($fechaUltimaLicencia['0']['ultim']!=null)
                        $fechaUltimaLicencia= clone $fechaUltimaLicencia['0']['ultim'];
                else
                        $fechaUltimaLicencia = null;

                if(!is_null($fechaUltimaLicencia)){
                        $intervaloUltimaLicencia = round(($fechaIni->getTimestamp() - $fechaUltimaLicencia-
                                                                >getTimestamp())  / (24*60*60)) -1 ;

                        $fechaUltimaLicenciaBD = $con->prepararFecha($fechaUltimaLicencia);
                        $fechaIniBD = $con->prepararFecha($fechaIni);
                        $festivos =$con->consultar("
                                                SELECT
                                                fcmn_dias_festivos(date '$fechaUltimaLicenciaBD' + INTERVAL '1
                                                                                                        day',
                                                date '$fechaIniBD' - INTERVAL '1 day', '$cpro', '$cmun') as
                                                                                                        \"festivos\"");
                        if ($festivos<0){
                                  IgepDebug::setDebug(DEBUG_USER,'FIN error5 f_leerValidacion');
                                  return -1;
                        }
                        $festivos=$festivos['0']['festivos'];
                        $intervaloUltimaLicencia -= $festivos;
                }else{
                        // Damos un valor elevado para que no salte la regla cuando no se ha pedido ninguna licencia
                        anteriormente
                        $intervaloUltimaLicencia = 100;
                }
        }
        if($codClase == 'LPP' OR $codClase == 'AIN'){
                $ret=SolicitudesYPermisos::f_mesesNaturalesPermiso($periodo ,$fechaIni,$fechaFin,$codClase,
                $totalMesesNat,$mesesNat,$totalRestoDias,$restoDias,$tipo,'ALTA',$nregpgv,$claseLlamada=$codClase );
                if ($ret<0){
                        IgepDebug::setDebug(DEBUG_USER,'FIN error6 f_leerValidacion');
                        return -1;
                }

                $ret=SolicitudesYPermisos::f_mesesNaturalesPermiso($periodo,$fechaIni,$fechaFin,$codClase,
                $totalMesesNatAnula,$mesesNatAnula,$totalRestoDiasAnula,$restoDiasAnula,$tipo,'ANUL',
                $nregpgv,$claseLlamada=$codClase );
                if ($ret<0){
                        IgepDebug::setDebug(DEBUG_USER,'FIN error7 f_leerValidacion');
                        return -1;
                }
        }

                // ------------------------------------------------------------------------------------------------
                // Cálculo de intervalo máximo de días naturales de un permiso de vacaciones,
                // incluyendo días adicionales, festivos y fines de semana
                // ------------------------------------------------------------------------------------------------
        if($fechaIni!=null)
                $fechaIniSol = clone $fechaIni;
        if($fechaFin!=null)
                $fechaFinSol = clone $fechaFin;


        if ($codClase == 'VAC'){

                $fechaIniSol = Solicitudes::f_ObtienePrimerDiaSolicitud($fechaIni,$periodo, $cpro, $cmun,'ADI', $nregpgv);
                $fechaFinSol = Solicitudes::f_ObtieneUltimoDiaSolicitud($fechaFin,$periodo, $cpro, $cmun,'ADI', $nregpgv);


                if($fechaIniSol<0 OR $fechaFinSol<0){
                        IgepDebug::setDebug(DEBUG_USER,'FIN error8 f_leerValidacion');
                        return -1;
                }
```

```
	}
	// Realizo un cálculo similar al anterior, pero para días ADI

		// ------------------------------------------------------------------------------------------------------
		// Cálculo de intervalo máximo de días naturales de un permiso de días adicionales de vacaciones
		// incluyendo días de vacaciones normales, festivos y fines de semana
		// ------------------------------------------------------------------------------------------------------
	if($fechaIni!=null)
		$fechaIniAdi = clone $fechaIni;
	if($fechaFin!=null)
		$fechaFinAdi = clone $fechaFin;

	if($codClase=='ADI'){
		$fechaIniSol = Solicitudes::f_ObtienePrimerDiaSolicitud($fechaIni,$periodo, $cpro, $cmun,'VAC', $nregpgv);
		$fechaFinSol = Solicitudes::f_ObtieneUltimoDiaSolicitud($fechaFin,$periodo, $cpro, $cmun,'VAC', $nregpgv);
		//Calculamos el valor de totalDiasVac y totalDiasLaboVac para saber si el estado debe ser PC o PR


	}
		// Cálculo total días laborables solicitados de VACACIONES, días ADICIONALES y LICENCIAS INTERES
		PARTICULAR
	if($codClase == 'VAC' or $codClase == 'LIP' or $codClase == 'ADI'){
		//Obtener información del total de dias de VAC y LIP de la persona

			//
-------------------------------------------------------------------------------------------
			// Total dóas solicitados de VACACIONES (VAC)
			//
-------------------------------------------------------------------------------------------

		// Cambiamos fechaIniVac y fechaFinVac por fechaIni y fechaFin que no incluyan los días ADI
		$diasVAC=SolicitudesYPermisos::f_diasPermiso($periodo, $fechaIni, $fechaFin, 'VAC',
			 $totalDiasVac, $totalDiasLaboVac, $diasVac, $diasLaboVac, $tipo, $nulo1, $nulo2,
			$nregpgv,$codClase);
		if($diasVAC<0){
			return $diasVAC;
		}


		// El recálculo de totalDiasLaboVac y totalDiasVac no hace falta porque ya se hace
		//  dentro de f_diasPermiso se añade el cálculo de maxDiasVac

		// Hay que averiguar el valor de maxDiasVac para VAC para usarlo en la regla de validación 9
		$ret = PermisosValidacion::f_LeerMinMaxDias('VAC',
						$diasMinimoVac,$diasMaximoVac,$diasNaturales,$unidad);
		if ($ret<0) {
			IgepDebug::setDebug(DEBUG_USER,'FIN error9 f_leerValidacion');
			return -1;
		}
		//Permitir que el personal funcionario incorporado durante el año, pueda pedirse el máximo de
		VAC y ADI
		UsuariosDePersonal::f_dameRelJur($nregpgv,$creljur);
		if($diasTrabajados == -1 OR $creljur != 'CL'){
			$maxDiasVac = $diasMaximoVac;
			$noEsLaboral = TRUE;
		}else{
			$maxDiasVac = round( ($diasMaximoVac * $diasTrabajados) / 365 );
			$noEsLaboral = FALSE;
		}


			//
-------------------------------------------------------------------------------------------
			// Total meses naturales solicitados de LICENCIA POR INTERES PARTICULAR (LIP)
			//
-------------------------------------------------------------------------------------------

		// Para LIP usaremos las variables añadiendoles el sufijo 'LIP'


		$retorno= SolicitudesYPermisos::f_mesesNaturalesPermiso($periodo, $fechaIni, $fechaFin,
					'LIP', $totalMesesNatLIP, $mesesNatLIP, $totalRestoDiasLIP, $restoDiasLIP,
					$tipo,'ALTA', $nregpgv,$claseLlamada=$codClase);
		if($retorno<0){
			IgepDebug::setDebug(DEBUG_USER,'FIN error10 f_leerValidacion');
			return -1;
		}
		// El recálculo de totalMesesNatLIP y totalRestoDiasLIP

	 ------------------------------------------------------------------------------------------
			// Total días solicitados de días ADICIONALES (ADI)
			//
-------------------------------------------------------------------------------------------
		$diasVACcompletos=SolicitudesYPermisos::f_diasPermiso($periodo, $fechaIni, $fechaFin,'ADI'
			,$totalDiasAdiSolicitados,$totalDiasLaboAdiSolicitados,$nulo1,$nulo2,$tipo,
			$nulo3,$nulo4,$nregpgv, $codClase);



	} //Fin cálculo del total de dias de VAC, ADI y LIP de la persona



	// Calcular el número de días para años bisiestos
	$duracionAnyo = 365;
	if($codClase == 'RJL' OR $codClase == 'AJL'){
		if(date('m/d',$fNacimientoMenor->format('m/d/Y')) < '03/01'){
			if((date('Y',$fNacimientoMenor->format('m/d/Y'))%4) == 0){
				$duracionAnyo = 366;
			}elseif ((date('Y',$fechaFin->format('m/d/Y'))%4) == 0){
```

```
                                $duracionAnyo = 366;
                        }
                }
}


if( $codClase == 'ADI'){
        //Obtener el valor del año para el que se conocen los días adicionales de vacaciones
        Usuarios::f_obtenerAnyoDiasADI($anyoDiasADI);
}


if(SolicitudesYPermisos::f_calcularDiasPermisoAnyoAnulacion($nregpgv, $periodo, $codClase,
     $totalDiasLaboAnula, $totalDiasAnula, $nsolicitudesAnula, $fechasAnulacion, $nsolicitudesAnulaResueltas)<0){
                        IgepDebug::setDebug(DEBUG_USER,'FIN error11 f_leerValidacion');
                        return -1;
}


if($diasNaturales == 'S')
        $diasAnula = $totalDiasAnula;
else
        $diasAnula = $totalDiasLaboAnula;


//  Hacer el cálculo de solicitudes y fechas de anulación para los periodos anteriores cuando
//  Se solicitan LIP y LFP
if ($codClase=='LIP' || $codClase == 'EFP'){

        if ($codClase == 'LIP')
                $numeroAnyos = 3;
        else
                $numeroAnyos = 5;

        for($anyo=1;$anyo<=$numeroAnyos-1;$anyo++){

                $nsolicitudesAnulaOtrosPeriodos = array();
                $fechasAnulacionOtrosPeriodos = array();
                $nsolicitudesAnulaResueltasOtrosPeriodos = array();

                if(SolicitudesYPermisos::f_calcularDiasPermisoAnyoAnulacion($nregpgv, $periodo -
                $anyo, $codClase,$nulo, $nulo, $nsolicitudesAnulaOtrosPeriodos,
                        $fechasAnulacionOtrosPeriodos, $nsolicitudesAnulaResueltasOtrosPeriodos)<0) {
                        IgepDebug::setDebug(DEBUG_USER,'FIN error12 f_leerValidacion');
                        return -1;
                }

                // Acumulamos las anulaciones de los periodos anteriores sobre las del periodo actual
                foreach($nsolicitudesAnulaOtrosPeriodos as $solicitud)
                        array_push($nsolicitudesAnula,$solicitud);
                $fechasAnulacion .= " " . $fechasAnulacionOtrosPeriodos;
                foreach($nsolicitudesAnulaResueltasOtrosPeriodos as $solicitud)
                        array_push($nsolicitudesAnulaResueltas,$solicitud);

                $nsolicitudesAnula = array_unique($nsolicitudesAnula);

        }


}


// Si el permiso de ENC se ha solicitado para 1 o más dias en el año, se devuelve 'S' en el parámetro
reqJustificante
// Mantenemos el >=1, aunque no tenga mucho sentido, por si vuelven a querer que sea para 3 o más días en
el año
if ($codClase == 'ENC' and $totalDiasValidados >= 1){
        $reqJustificante='S';
}
else{
        $reqJustificante='N';
}


if ($codClase=='EGP' || $codClase == 'DEF'){
        if(Solicitudes::f_dameDiasMaximoEGPoDEF($distancia, $gradoParentesco, $diasMaximo)<0) {
                IgepDebug::setDebug(DEBUG_USER,'FIN error13 f_leerValidacion');
                return -1;
        }

}

if ($codClase=='MFA' || $codClase == 'MUH' || $codClase == 'TDO'){
        if(Solicitudes::f_dameDiasMaximoMFAoMUHoTDO($codClase, $distancia, $diasMaximo)<0) {
                IgepDebug::setDebug(DEBUG_USER,'FIN error14 f_leerValidacion');
                return -1;
        }

}


$pnregpgv = $nregpgv;
if ($codClase=='MAT')
        if(Solicitudes::f_dameVecesPermiso($codClase,$pnregpgv,'N', $periodo, $totalVecesPermiso)<0){
                IgepDebug::setDebug(DEBUG_USER,'FIN error15 f_leerValidacion');
                return -1;
        }

if ($codClase=='VAC')
        if(Solicitudes::f_dameVecesPermiso('ADI',$pnregpgv,'S', $periodo, $totalVecesPermisoAnual)<0) {
```

```
                    IgepDebug::setDebug(DEBUG_USER,'FIN error16 f_leerValidacion');
                    return -1;
          }


// Comprobar si se han pedido ya todos los días de vacaciones y adicionales.
// El resultado de la comprobación se devuelve en VacyAdiCompletos
$vacyAdiCompletos = false;
if ($codClase == 'VAC' OR $codClase== 'ADI'){

          // Hay que averiguar el valor de maxdias para VAC si se están pidiendo ahora los adicicionales
          if($codClase=='ADI'){
                    $diasAdiMaxMin = PermisosValidacion::f_LeerMinMaxDias('VAC', $diasMinimo,
                                                                          $diasMaximo,$diasNaturales,$unidad);
                    if ($diasAdiMaxMin<0) {
                              IgepDebug::setDebug(DEBUG_USER,'FIN error17 f_leerValidacion');
                              return -1;
                    }
                    //Permitir que el personal funcionario incorporado durante el año, pueda pedirse el
                    máximo de VAC y ADI
                    UsuariosDePersonal::f_dameRelJur($nregpgv,$creljur);
                    if($diasTrabajados == -1 OR $creljur != 'CL'){
                              $maxDias = $diasMaximo;
                              $noEsLaboral = TRUE;
                    }else{
                              $maxDias = round( ( $diasMaximo * $diasTrabajados) / 365 );
                              $noEsLaboral = FALSE;
                    }
          }



          $diasVACcompletos=0;


          if($diasVACcompletos<0){
                    return $diasVACcompletos;//"Error al calcular los días de permiso"
          }
          if ($totalMesesNatLIP >=1){
                    //  Se redondea el valor de las operaciones
           if($totalDiasLaboVac >= round($maxDias - $maxDias/12 * ($totalMesesNatLIP + $totalRestoDiasLIP/30))
                              AND ($totalDiasLaboAdiSolicitados >= $totalDiasAdi)){

                    $vacyAdiCompletos = true;
            }

          }else{
                    // Calculamos como antes
                    if($totalDiasLaboVac >= $maxDias AND $totalDiasLaboAdiSolicitados >= $totalDiasAdi){
                    $vacyAdiCompletos = true;
                    }
          }

}

// Generación de los mensajes de aviso o prohibición interactivamente
// Se recorren las reglas de validación por orden,
// evaluando aquellas que cumplan los criterios de tipo de permiso y periodo

$var_regla_validacion =$con->consultar("SELECT orden, control, condicion, mensaje, mensajev, articulo,
                                        codigoleydecreto, codigo .
                                  FROM TPER_PERMISOSVALIDACION
                                  WHERE (cclase = '$codClase' or cclase is null) and
                                          (cpro = '".$cpro."' or cpro is null) and
                                          (cmun = '".$cmun."' or cmun is null) and
                                          (periodo = '$periodo' or periodo is null) and
                                          activo = 'S'
                                  order by orden");
if($var_regla_validacion<0){
          IgepDebug::setDebug(DEBUG_USER,'FIN error18 f_leerValidacion');
          return -1;
}

////////////////////////////////////////////////////////////////////////
//Normalización de variables para poder usarse en las reglas de validación


if(!isset($restoDiasAnula)OR empty ($restoDiasAnula) OR is_null($restoDiasAnula))
          $restoDiasAnula=0;
if(!isset($mesesAnula)OR empty ($mesesAnula) OR is_null($mesesAnula))
          $mesesAnula=0;
if(!isset($diasAnula)OR empty ($diasAnula) OR is_null($diasAnula))
          $diasAnula=0;
if(!isset($intervaloUltimaLicencia)OR empty ($intervaloUltimaLicencia) OR is_null($intervaloUltimaLicencia))
          $intervaloUltimaLicencia=0;
if(!isset($totalDiasValidados)OR empty ($totalDiasValidados) OR is_null($totalDiasValidados))
          $totalDiasValidados=0;
if(!isset($fNacimientoMenor)OR empty ($fNacimientoMenor) OR is_null($fNacimientoMenor))
          $fNacimientoMenor=new gvHidraTimestamp('01/01/1970');
if(!isset($totalDiasLaboVac)OR empty ($totalDiasLaboVac) OR is_null($totalDiasLaboVac))
          $totalDiasLaboVac=0;
if(!isset($totalDiasLabo)OR empty ($totalDiasLabo) OR is_null($totalDiasLabo))
          $totalDiasLabo=0;
if(!isset($totalDiasLaboAdiSolicitados)OR empty ($totalDiasLaboAdiSolicitados) OR
                                                                  is_null($totalDiasLaboAdiSolicitados))
          $totalDiasLaboAdiSolicitados=0;
if(!isset($totalRestoDias)OR empty ($totalRestoDias) OR is_null($totalRestoDias))
          $totalRestoDias=0;
if(!isset($diasMinimo)OR empty ($diasMinimo) OR is_null($diasMinimo))
          $diasMinimo=0;
```

```
if(!isset($diasMaximo) OR is_null($diasMaximo))
          $diasMaximo=99999;
if(!isset($diasMaximoLocal) OR is_null($diasMaximoLocal))
          $diasMaximoLocal=99999;
if(!isset($maxDiasVac)OR empty ($maxDiasVac) OR is_null($maxDiasVac))
          $maxDiasVac=99999;
if(!isset($maxDias)OR is_null($maxDias))
          $maxDias=99999;
if(!isset($totalDiasAdi)OR empty ($totalDiasAdi) OR is_null($totalDiasAdi))
          $totalDiasAdi=0;
if(!isset($totalMesesNat)OR empty ($totalMesesNat) OR is_null($totalMesesNat))
          $totalMesesNat=0;
if(!isset($totalMesesNatLIP)OR empty ($totalMesesNatLIP) OR is_null($totalMesesNatLIP))
          $totalMesesNatLIP=0;
if(!isset($totalRestoDiasLIP)OR empty ($totalRestoDiasLIP) OR is_null($totalRestoDiasLIP))
          $totalRestoDiasLIP=0;
$hoy= new gvHidraTimeStamp();//Para usar una referencia al dÃa actual en las reglas de validaciÃ³n
$hoy->setTime(0,0,0);


//Fin Normalización




//Almacena en la variable  $user si es P_TRAMITA
$rol = IgepSession::dameRol();

$mensajeError=" ";
$mensajeAviso=" ";
$mensajeAvisoUsuario=" ";
$abortar=$detener=false;
$articuloLeyODecreto=array();

$listaSolicitudesAnulacion=array();
$listaSolicitudesAnulacionResueltas=array();


IgepDebug::setDebug(DEBUG_USER,"Reglas a evaluar:<pre>".print_r($var_regla_validacion,true)."</pre>");
foreach ($var_regla_validacion as $indice => $regla){
          $condicion=null;unset($condicion);


          IgepDebug::setDebug(DEBUG_USER,"Regla original en BBDD (".$regla['codigo']."):".$prueba=
                                                                                $regla['condicion']);
          eval("\$salida = \"$prueba\";");
          IgepDebug::setDebug(DEBUG_USER,"Regla remplazada en PHP (".$regla['codigo']."): $salida");

          if($regla['codigo']=='19') {
                    IgepDebug::setDebug(DEBUG_USER," fNacimiento: ".$fNacimientoMenor->format('d/m/Y'));
                    if (!is_null($fechaFin))
                              IgepDebug::setDebug(DEBUG_USER," fFin: ".$fechaFin->format('d/m/Y'));
          }


          //Primer eval para remplazar variables por valores
          eval("\$eval = \"$regla[condicion]\";");




          //Evaluamos la condición de la regla
          $resultadoValidacion = eval("return ".$eval.";");

          if($resultadoValidacion===TRUE)
                    $resVal="true";
          elseif($resultadoValidacion===FALSE)
                    $resVal="false";
          IgepDebug::setDebug(DEBUG_USER,"Resultado ValidaciÃ³n de regla (".$regla['codigo']."): $resVal");


          if($resultadoValidacion){

                    if (in_array ($regla['codigo'], array('44', '45', '46', '47', '48', '49', '50', '51', '52',
                                                                                                        '53', '54'))){

                              $listaSolicitudesAnulacion= $nsolicitudesAnula;
                              $listaSolicitudesAnulacionResueltas = $nsolicitudesAnulaResueltas;

                    }


                    //Abortamos ante errores prioritarios (por ejemplo fechas nulas) Salimos SOLO con el mensaje de
                    error negativo
                    if($regla['control']=='P' AND $regla['orden']<0 ){
                              $resultado['tipo']='P';
                              $mensajeP=$regla['mensaje'];
                              eval("\$mensajeP = \"$mensajeP\";"); //Remplazamos variables del mensaje
                              $mensajeError=$mensajeError." <br> ".$mensajeP;
                              $resultado['mensaje'] = $mensajeError;
                              IgepDebug::setDebug(DEBUG_USER,'FIN f_leerValidacion: regla de prohibición negativa');
                              return -1;
                    }

                    //Caso TRAMITADOR
                    $mensajeA=$regla['mensaje'];
                    eval("\$mensajeA = \"$mensajeA\";"); //Remplazamos variables del mensaje
                    $mensajeAvisoHTML=$mensajeAvisoHTML." <LI> ".$mensajeA." </LI> ";
                    $mensajeAviso=$mensajeAviso."- ".$mensajeA."\n ";
                    //Fin caso tramitador
```

```
                        //Caso general que se ejecuta además del caso tramitador. Para todos los usuarios
                        if($regla['control']=='A'){// Acumulamos los posibles mensajes de error y aviso y continuamos
                                                                                    hasta el fin de las reglas
                            if( !is_null($regla['articulo']) AND !is_null($regla['codigoleydecreto']) ){
                                  if (!in_array ($regla['articulo']."-".$regla['codigoleydecreto'],
                                                                                    $articuloLeyODecreto)){
                                      // Añadir [articulo-leyDecreto] a la pila
                                      array_push($articuloLeyODecreto,
                                      $regla['articulo']."-".$regla['codigoleydecreto']);

                                      //Añadimos para mostrar el mensaje

                                      LeyDecreto::f_dameLeyDecreto($regla['codigoleydecreto'], $var_desc_leydecreto);
                                      eval("\$mensajeA = \"$regla[articulo]\";");
                                      eval("return ".$mensajeA.";");
                                      //$mensajeA=eval("return ".$mensajeA.";");
                                      //eval("\$mensajeA = \"$mensajeA\";"); //Remplazamos variables del mensaje
                                      $mensajeA="Ver artículo ".$mensajeA." ".$var_desc_leydecreto;
                                      $mensajeAvisoUsuarioHTML=$mensajeAvisoUsuarioHTML." <LI> ".$mensajeA." </LI> ";
                                      $mensajeAvisoUsuario=$mensajeAvisoUsuario."- ".$mensajeA."\n ";
                                  }
                             }else{
                                      $mensajeA=$regla['mensaje'];
                                      eval("\$mensajeA = \"$mensajeA\";"); //Remplazamos variables del mensaje
                                      $mensajeAvisoUsuarioHTML=$mensajeAvisoUsuarioHTML." <LI> ".$mensajeA." </LI> ";
                                      $mensajeAvisoUsuario=$mensajeAvisoUsuario."- ".$mensajeA."\n ";
                                  }
                                          //        29/06/2010 INICIO Tarea6702
                        }elseif(($regla['control']=='P' AND $rol!='P_TRAMITA' AND $rol!='P_RESPPER' AND
                                                                    $rol!='P_REVISOR') OR $nregpgvQueSolicita==$nregpgv ){
                                 //Es regla de prohibición y no es tramitador

                                  $mensajeP=$regla['mensaje'];
                                  eval("\$mensajeP = \"$mensajeP\";"); //Remplazamos variables del mensaje
                                  $mensajeError=$mensajeError." <br> ".$mensajeP;
                                  $detener= true;//Existe al menos un mensaje de Prohibicion.Detendremos la validación
                                                fuera del bucle.(acumula mensajes de error)
                        }

                }else{
                        //No Se cumple la regla, no hacemos nada
                }
        }//Fin foreach


        //Mostramos mensajes de prohibición porque al menos una regla fue de prohibición.
        if ($detener==true){
                $resultado['tipo']='P';
                $resultado['mensaje']= $mensajeError;
                IgepDebug::setDebug(DEBUG_USER,'FIN f_leerValidacion: al menos una regla fue de prohibiciÃ³n');
                return -1;
        }else{
                //TRAMITADOR
                $resultado['tipo']='A';
                //Mensaje de base de datos
                $resultado ['mensaje']=$mensajeAviso;
                //Mensaje pantalla
                $resultado ['mensajeHTML']="<UL>".$mensajeAvisoHTML."</UL>";//AÃ±adir tags HTML para hacer lista


                //TODOS
                $resultadoUsuario['tipo']='A';
                //Mensaje de base de datos
                $resultadoUsuario ['mensaje']=$mensajeAvisoUsuario;
                //Mensaje pantalla
                $resultadoUsuario ['mensajeHTML']="<UL>".$mensajeAvisoUsuarioHTML."</UL>";//AÃ±adir tags HTML para hacer
                                                                                                        lista
        }

        // Si no se están modificando anulaciones con otras altas, entonces las anulaciones no se descuentan con la variable
        diasAnula
        if(count($listaSolicitudesAnulacion)==0)
                $diasAnula=0;


        IgepDebug::setDebug(DEBUG_USER,'FIN f_leerValidacion: maxDiasVac='.$maxDiasVac. ';diasTrabajados='.$diasTrabajados);



        return 0;
}//f_leerValidacion




function f_CalcularMinMaxDias($pcclase, $pperiodo, $pcmun, $pcpro, &$pdiasMinimo, &$pdiasMaximo, $pdiastrabajados,
     &$psonDiasNaturales, &$pmaxdias, $ptotaldias, $ptotaldiaslabo, &$ptotalDiasAValidar, &$pdesc_tipodias,$ptrienios,
     $pdiasTrabajadosVV6, &$pdiasAValidar, &$pdias, &$pdiasLabo, $pnregpgv, &$pnoEsLaboral,&$pdiasMaximoLocal){

        IgepDebug::setDebug(DEBUG_USER,'INI f_CalcularMinMaxDias');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);

        //Protección contra parámetros nulos pasados por referencia
        $cpdiasMinimo= $pdiasMinimo;
        $cpdiasMaximo= $pdiasMaximo;
        $cpsonDiasNaturales=$psonDiasNaturales;
        $cpdiasMaximoLocal=$pdiasMaximoLocal;
```

```
        // ------------------------------------------------------------------------------------------------------
        // Cálculo de dias mínimo y máximo a solicitar por permisos
        // ------------------------------------------------------------------------------------------------------

          $ret = PermisosValidacion::f_LeerMinMaxDias($pcclase, $cpdiasMinimo, $cpdiasMaximo, $cpsonDiasNaturales,$nulo);
          if($ret<0)return $ret;


          if($cpsonDiasNaturales=='S'){
                    $ptotalDiasAValidar = $ptotaldias;
                    $pdiasAValidar = $pdias;
                    $pdesc_tipodias = 'naturales';
          }else{
                    $ptotalDiasAValidar = $ptotaldiaslabo;
                    $pdiasAValidar = $pdiasLabo;
                    $pdesc_tipodias = 'hÃ¡biles';
          }



          //En el caso de 'VV6' volvemos a calcular el máximo de días porque la información se encuentra en otro sitio
          if($pcclase=='VV6'){
                    //Comprobar en PermisosMaxDias el máximo de días por provincia de la solicitud

                    // Si los días máximos de VV6 no están definidos para el municipio se toma el valor por defecto del
                    permiso

                    $cpdiasMaximoLocal = PermisosMaxDiasLocal::f_LeerMaxDiasLocal($pcclase, $pperiodo, $pcmun, $pcpro);
                    if($cpdiasMaximoLocal<0){
                              IgepDebug::setDebug(DEBUG_USER,'FIN error1 f_CalcularMinMaxDias');
                              return -1;
                    }
                    if ($cpdiasMaximoLocal>0) $cpdiasMaximo = $cpdiasMaximoLocal;



          }

          if($pcclase == 'AAP'){
                    //Recalculo de días de asuntos propios para funcionarios con 6 o más trienios cumplidos
                    if($ptrienios == 6 OR $ptrienios == 7){
                              $cpdiasMaximo = 2;
                    }elseif($ptrienios >= 8){
                              $cpdiasMaximo = $ptrienios - 5;
                    }else
                              $cpdiasMaximo = 0;
          }



                    // ------------------------------------------------------------------------------------------------------
                    // pmaxdias se utilizará para indicar el máximo de días a solicitar para días adicionales y vacaciones
                    // con las otras clases de permisos se utilizará diasMaximo
                    // ------------------------------------------------------------------------------------------------------

          if($pcclase == 'VAC'){
                    //Si la persona no ha trabajado todo el año, se reducirán proporcionalmente
                    //Permitir que el personal funcionario incorporado durante el año, pueda pedirse el máximo de VAC
                    UsuariosDePersonal::f_dameRelJur($pnregpgv,$creljur);
                    if($pdiastrabajados == -1 OR $creljur != 'CL'){
                              $pmaxdias = $cpdiasMaximo;
                              $pnoEsLaboral = TRUE;
                    }else{
                              $pmaxdias = round( ($cpdiasMaximo * $pdiastrabajados) / 365 );
                              $pnoEsLaboral = FALSE;
                    }

          }elseif($pcclase == 'VV6'){
                    //Si la persona no ha trabajado todo el año, se reducirán proporcionalmente
                    //Permitir que el personal funcionario incorporado durante el año, pueda pedirse el máximo de VV6
                    UsuariosDePersonal::f_dameRelJur($pnregpgv,$creljur);
                    if($pdiasTrabajadosVV6 == -1 OR $creljur != 'CL'){
                              $pmaxdias = $cpdiasMaximo;
                              $pnoEsLaboral = TRUE;
                    }else{
                              $pmaxdias = round( ($cpdiasMaximo * $pdiasTrabajadosVV6) / 365 );
                              $pnoEsLaboral = FALSE;
                    }
          }else{
                    // Se define pmaxdias para que tenga valor al evaluar las reglas de validaciÃ³n
                              $pmaxdias = $cpdiasMaximo;
          }
        $pdiasMinimo= $cpdiasMinimo;
        $pdiasMaximo= $cpdiasMaximo;
        $psonDiasNaturales=$cpsonDiasNaturales;
        $pdiasMaximoLocal=$cpdiasMaximoLocal;
        IgepDebug::setDebug(DEBUG_USER,'FIN f_CalcularMinMaxDias: pmaxdias='.$pmaxdias);
        return 0;
}



    /**
     * fRecalcularTotalVACMesNaturalAnulacion
     * @param integer $pnregpgv
     * @param string $pcclase
     * @param integer $pperiodo
     * @param integer $ptotalDiaslaboVac
     * @return integer
     */
function f_recalcularTotalVACMesNaturalAnulacion($pnregpgv,$pcclase,$pperiodo,&$ptotalDiaslaboVac ){
```

```
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_recalcularTotalVACMesNaturalAnulacion');

                // ------------------------------------------------------------------------------------------------
                // Recalculo de total de días hábiles de vacaciones cuando se pide un intervalo de un mes natural
                // ------------------------------------------------------------------------------------------------
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);
        $var_periodos_solicitados ="
                        SELECT count(*) as cuenta
                        FROM TPER_SOLICITUDES
                        WHERE nregpgv = '$pnregpgv' and cclase = 'VAC' and periodo = $pperiodo and
                           NOT ((cestado in ('NR','NI')) OR
                            (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                            (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null)) and
                           situacion IN ('X', 'B')";
        $var_periodos_solicitados=$con->consultar($var_periodos_solicitados);
        if ($var_periodos_solicitados<0)return -1;
        $var_periodos_solicitados=$var_periodos_solicitados[0]['cuenta'];
        if($var_periodos_solicitados == 1){
        // Solo se ha solicitado un periodo de vacaciones y no se está pidiendo otro, comprobar si dura un mes natural
                $dia_ini_vac_dia_fin_vac = " SELECT fecha_ini as dia_ini_vac, fecha_fin as dia_fin_vac
                                        FROM TPER_SOLICITUDES
                                        WHERE nregpgv = '$pnregpgv' and cclase = 'VAC' and periodo = $pperiodo and
                                           NOT ((cestado in ('NR','NI')) OR
                                                (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                                                (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))
                                              and
                                           situacion IN ('X', 'B')";
                $dia_ini_vac_dia_fin_vac = $con->consultar($dia_ini_vac_dia_fin_vac,array(
                                                'DATATYPES'=>array('dia_ini_vac'=>TIPO_FECHA,'dia_fin_vac'=>TIPO_F
                                                                                                        ECHA,))));
                if ($dia_ini_vac_dia_fin_vac<0)return -1;
                // Solo cuando se ha solicitado un solo periodo de vacaciones ahora, se puede tomar
                // en consideración el mes como natural
                // Sumamos 1 mes y LUEGO restamos un día
                $fechaComparacion = clone $dia_ini_vac_dia_fin_vac[0]['dia_ini_vac'];
                $fechaComparacion->addMonths(1);
                $fechaComparacion->subDays(1);


                if(gvHidraTimestamp::cmp($dia_ini_vac_dia_fin_vac[0]['dia_ini_vac'], $fechaComparacion) == 0){

                        //Se ha solicitado un periodo de un mes natural.
                        //Las vacaciones se cuentan del máximo días hábiles posibles ya que se ha pedido un mes natural
                        de vacaciones
                        $ret = PermisosValidacion::f_LeerMinMaxDias('VAC', $nulo, $ptotalDiaslaboVacMax, $nulo1,$nulo2);
                        if($ret<0)return -1;

                        if($ptotalDiaslaboVac >= $ptotalDiaslaboVacMax)
                                $ptotalDiaslaboVac = $ptotalDiaslaboVacMax;

                }// Fin solicitud de un mes natural en un Ã°nico periodo

        }// Fin solicitud de un único periodo
        IgepDebug::setDebug(DEBUG_USER,'FIN f_recalcularTotalVACMesNaturalAnulacion');
        return 0;
}



function f_RecalcularTotalVACMesNatural($pcclase,$pPeriodo,$ppfechaini,$ppfechafin,$pcmun,$pcpro,&$ptotal_diaslabo_vac,
                                                                                                        $pnregpgv){
        // ------------------------------------------------------------------------------------------------
        // Recalculo de total de días hábiles de vacaciones cuando se pide un intervalo de un mes natural
        // ------------------------------------------------------------------------------------------------
        IgepDebug::setDebug(DEBUG_USER,'INI f_RecalcularTotalVACMesNatural');

        if ($ppfechaini!=null)
                        $pfechaini = clone $ppfechaini;
        if ($pfechafin!=null)
                        $pfechafin = clone $ppfechafin;

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);
        if($pcclase=='VAC' OR $pcclase=='LIP'){ //Vacaciones y Días Adicionales
          $res = $con->consultar("SELECT count(*) as \"periodos_solicitados\"
              FROM TPER_SOLICITUDES
              WHERE nregpgv = '".$pnregpgv."' and cclase = 'VAC' and periodo = ".$pPeriodo." and" .
                        " NOT ((cestado in ('NR','NI')) OR
              (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
              (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))  ");

          if($res==-1)
              return -1;
          $var_periodos_solicitados = $res[0]['periodos_solicitados'];
          if($var_periodos_solicitados==1 AND $pcclase!='VAC'){
              // Solo se ha solicitado un periodo de vacaciones y no se está pidiendo otro, comprobar si dura un mes
                  natural
              $res = $con->consultar("SELECT fecha_ini, fecha_fin
                        FROM TPER_SOLICITUDES
                        WHERE nregpgv = '".$pnregpgv."' and cclase = 'VAC' and periodo = ".$pPeriodo." and
                     NOT ((cestado in ('NR','NI')) OR
                        (situacion = 'A' and cestado='NP' and ffirmaresolucion is not null) OR
                        (situacion = 'B' and cestado='AP' and ffirmaresolucionanula is not null))
                     ",array( 'DATATYPES'=>array('fecha_ini'=>TIPO_FECHA,'fecha_fin'=>TIPO_FECHA,)));

              if($res==-1)
                  return -1;
              $dia_ini_vac = clone $res[0]['fecha_ini'];
              $dia_fin_vac = clone $res[0]['fecha_fin'];
```

```
            }elseif($var_periodos_solicitados == 0 AND $pcclase == 'VAC'){
                $dia_ini_vac = clone $pfechaini;
                $dia_fin_vac = clone $pfechafin;
            }

            if(($var_periodos_solicitados == 1 AND  $pcclase != 'VAC') OR ($var_periodos_solicitados == 0 AND $pcclase ==
                                                                                                        'VAC')){
                    // Solo cuando se ha solicitado un solo periodo de vacaciones, anteriormente o ahora, se puede tomar
                    // en consideración el mes como natural

                     //Sumamos 1 mes y LUEGO restamos un día
                    $fechaComparacion = clone $dia_ini_vac;
                    $fechaComparacion->addMonths(1);
                    $fechaComparacion->subDays(1);

                    if(gvHidraTimestamp::cmp($dia_fin_vac, $fechaComparacion) == 0){
                              // Se ha solicitado un periodo de un mes natural.

                              // Las vacaciones se cuentan del máximo días hábiles posibles ya que se ha pedido un mes natural
                              de vacaciones
                              $ret = PermisosValidacion::f_LeerMinMaxDias('VAC', $nulo, $ptotal_diaslabo_vac_max,
                                                                                                        $nulo1,$nulo2);
                              if($ret<0)return $ret;

                              if($ptotal_diaslabo_vac >= $ptotal_diaslabo_vac_max){
                                        $ptotal_diaslabo_vac = $ptotal_diaslabo_vac_max;
                              }
                    }// Fin solicitud de un mes natural en un único periodo

            }// Fin solicitud de un único periodo

          }//Fin cÃ¡lculo del total de dias de VAC cuando se solicitan VAC o LIP


          IgepDebug::setDebug(DEBUG_USER,'FIN f_RecalcularTotalVACMesNatural');
}//Fin f_RecalcularTotalVACMesNatural


}//Fin clase PermisosValidacion
?>
```

– Clase Firmantes:

```
<?php

class Firmantes{

    var $f_inicio;
    var $f_fin;
    var $suplente;


          /**
           * Devuelve los grupos de firma en los que un responsable de firma es suplente
           * @param integer $nregpgv
           * @param string $sql
           * @return integer
           * */
function f_listaGruposFirmaSuplente($pnregpgv,&$sql){
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_listaGruposFirmaSuplente');
          $sql ="SELECT   sup.codcentro, sup.centro, sup.coddepartamento, sup.departamento, sup.codseccion, sup.seccion,
                                    p.dnombre || ' ' || p.dapell1 || ' ' || p.dapell2 as titular,p.nregpgv as codtitular,
                                    sup.activo
                                              FROM
                                    (
                                       SELECT
                                          c.codigo as codcentro, c.descripcion as centro,
                                          d.codigo as coddepartamento, d.descripcion as departamento,
                                          s.codigo as codseccion, s.descripcion as seccion,
                                          max(tit.nregpgv) as titular, f.activo
                                       FROM
                                          twps_centros c, twps_departamentos d, twps_secciones s,
                                       TPER_FIRMANTES f left outer join TPER_FIRMANTES tit
                                       ON   tit.centro = f.centro and
                                          tit.departamento = f.departamento and
                                          tit.seccion = f.seccion and
                                          tit.suplente = 'N'
                                       WHERE
                                          f.nregpgv = '$pnregpgv' and
                                          f.suplente = 'S' and
                                          (current_date >= f.f_inicio and current_date <= f.f_fin or f.f_fin is null)
                                          and c.codigo = f.centro
                                          and d.codigo = f.departamento
                                          and s.codigo = f.seccion
                                       group by c.codigo, c.descripcion, d.codigo, d.descripcion, s.codigo, s.descripcion,
                                    f.activo
                                    ) sup
```

```
                                    left outer join
                                    VPER_OCUPACION P
                                    on p.nregpgv = sup.titular";
          IgepDebug::setDebug(DEBUG_USER,'FIN f_listaGruposFirmaSuplente');
          return 0;
 }


    /**
    * f_esResponsableNoSuplente: Devuelve si el usuario es responsable de firma no suplente
    */
function f_esResponsableNoSuplente($p_nregpgv, $pcentro, $pdepartamento, $pseccion){
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_esResponsableNoSuplente');
        //Se llama para saber si el usuario es responsable de firma y no es suplente
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);

        $fecha=new gvHidraTimestamp();
        $fecha->setTime(0,0,0);
        $fechaBD = $conexion->prepararFecha($fecha);

        $res = $conexion->consultar('SELECT suplente as "suplente"
                                    FROM TPER_FIRMANTES
                                    WHERE nregpgv = \''.$p_nregpgv.'\' and
                                    centro = \''.$pcentro.'\' and
                                    departamento = \''.$pdepartamento.'\' and
                                    seccion = \''.$pseccion.'\' and
                                    (
                                        (\''.$fechaBD.'\' >= f_inicio and
                                        (\''.$fechaBD.'\' <= f_fin or f_fin is null))
                                    )');
        IgepDebug::setDebug(DEBUG_USER,'FIN f_esResponsableNoSuplente');
        if(empty($res[0]['suplente']) OR $res[0]['suplente']=='')
            return FALSE;
        else {
            if($res[0]['suplente']=='S')
                return FALSE;
            else
                return TRUE;
        }
}

    /**
     * MÃ©todo estÃ¡tico
     */
function f_esResponsableDePuertos($pnregpgv){
        //El Jefe de la División de explotación y conservación de puertos es la persona encargada de firmar
        //las solicitudes de todo el personal de puertos, es decir, es la persona responsable que en TPER_FIRMANTES
        // tenga como centro 03, departamento 01 y sección 21.
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_esResponsableDePuertos');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);
        if($pnregpgv!='')
        $nregpgv= " SELECT nregpgv as \"nregpgv\"  FROM TPER_FIRMANTES".
                  "       WHERE centro='03' and ".
                  "       departamento = '01' and ".
                  "       seccion='21' and".
                  "       nregpgv ='".$pnregpgv."' and".
                  " (suplente='N' or (suplente='S' and activo='S'))";

        $res = $conexion->consultar($nregpgv);
        IgepDebug::setDebug(DEBUG_USER,'FIN f_esResponsableDePuertos');
        if(!empty($res[0]['nregpgv']))
            return TRUE;
        else
            return FALSE;
}
    /**
     * f_activarSuplenteFirma: Activa el grupo de firma seleccionado para el suplente de firma
     * @return boolean
     */

function f_activarSuplenteFirma($pcodcentro, $pcoddepartamento, $pcodseccion, $pnregpgv,&$resultado){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_activarSuplenteFirma');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);
          // Primero se obtiene la lista de usuarios responsables titulares para enviarles un correo
          $grupo = array("centro"=>$pcodcentro,"departamento"=>$pcoddepartamento,"seccion"=>$pcodseccion);
          $current_date=new gvHidraTimestamp();
          $current_date->setTime(0,0,0);
          $destinatarios=Firmantes::f_dameResponsables($grupo, $current_date, $current_date, 'S');


          $centro ="
                        SELECT descripcion
                        FROM twps_centros
                        WHERE codigo = '$pcodcentro'";

          $departamento ="
                        SELECT descripcion
                        FROM twps_departamentos
                        WHERE codigo = '$pcoddepartamento'";

          $seccion ="
                        SELECT descripcion
                        FROM twps_secciones
                        WHERE codigo = '$pcodseccion'";

          $centro=$conexion->consultar($centro);
          $departamento=$conexion->consultar($departamento);
          $seccion=$conexion->consultar($seccion);
```

```
        $centro=$centro[0]['descripcion'];
        $departamento=$departamento[0]['descripcion'];
        $seccion=$seccion[0]['descripcion'];



        // Luego se activa el suplente temporalmente para que pueda firmar,
        // salvo que ya está activado

        // Obtenemos la fecha y hora del dia y recuperamos la de fin de activacion temporal del suplente

        $ffinactivo= new gvHidraTimestamp(); // obtiene fecha y hora del sistema



        // incluye campo f_fin_activo de siguente linea
        $consulta="SELECT activo,f_fin_activo
                FROM TPER_FIRMANTES
                WHERE nregpgv='$pnregpgv' and
                                centro = '$pcodcentro' and
                                departamento = '$pcoddepartamento' and
                                seccion = '$pcodseccion'";

        $res=$conexion->consultar($consulta);
        if ($res<0)return FALSE;
        // Si no está activado o si esta activado y ha pasado la hora de activación temporal lo activamos.
        $ffinactivo_ant=$res[0]['f_fin_activo'] ;
        $var_activo=$res[0]['activo'] ;


        if(($res[0]['activo']=='N') or (($var_activo =='S') and ($ffinactivo_ant < $ffinactivo ) and ($ffinactivo_ant !=
                                                                                                        null))) {
                //Añadimos una hora al intervalo
                $ffinactivo->modify("+1 hour");
                $ffinactivo=$ffinactivo->format("d/m/Y H:i:s");
                $update="
                                UPDATE
                                   TPER_FIRMANTES
                                SET activo = 'S',f_fin_activo = '$ffinactivo'
                                WHERE nregpgv='$pnregpgv' and
                                      centro = '$pcodcentro' and
                                      departamento = '$pcoddepartamento' and
                                      seccion = '$pcodseccion'";

                $res=$conexion->consultar($update);

                //Se envía correo a los responsables de firma titulares
                UsuariosDePersonal::f_dameNombreApellidosInteresado($pnregpgv,$dnombre, $dapell1, $dapell2);

                // Mensaje a enviar en el correo
                $mensaje = "El suplente de firma $dnombre $dapell1 $dapell2 del centro $centro, departamento $departamento
                                 y sección $seccion se ha activado temporalmente para firmar como responsable.";

                //Enviar correo a los responsables
                Solicitudes::f_enviarCorreo($pnregpgv,$destinatarios,$mensaje,"PERSONAL - Activación temporal de suplente
                                                                                                        de firma");


                $resultado="Se ha activado temporalmente como suplente de firma del centro $centro, departamento
                                                                $departamento y sección $seccion.";
                IgepDebug::setDebug(DEBUG_USER,'FIN f_activarSuplenteFirma');
                return TRUE;
                }else
                        $resultado="El suplente de firma del centro $centro, departamento $departamento y sección
                                                                        $seccion ya estaba activado.";
                IgepDebug::setDebug(DEBUG_USER,'FIN f_activarSuplenteFirma');
                return TRUE;
}

    /**
     *f_desactivarSuplenteFirma: Desactiva el grupo de firma seleccionado para el suplente de firma
     *@return boolean
     */
function f_desactivarSuplenteFirma($pcodcentro, $pcoddepartamento, $pcodseccion, $pnregpgv,&$pactivo,&$destinatarios,
                                                                                                        &$pmensaje){
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_desactivarSuplenteFirma');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
      $conexion = new IgepConexion($g_dsn, true);
        // Primero se obtiene la lista de usuarios responsables titulares
        $grupo = array("centro"=>$pcodcentro,"departamento"=>$pcoddepartamento,"seccion"=>$pcodseccion);
        $current_date=new gvHidraTimestamp();
        $current_date->setTime(0,0,0);
        $destinatarios=Firmantes::f_dameResponsables($grupo, $current_date, $current_date, 'S');

        // Luego se desactiva el suplente para que ya no pueda firmar,
        // salvo que ya está desactivado, en cuyo caso no hace nada

        $centro ="
                        SELECT descripcion
                        FROM twps_centros
                        WHERE codigo = '$pcodcentro'";

        $departamento ="
                        SELECT descripcion
                        FROM twps_departamentos
                        WHERE codigo = '$pcoddepartamento'";

        $seccion ="
                        SELECT descripcion
                        FROM twps_secciones
```

```
                    WHERE codigo = '$pcodseccion'";

        $centro=$conexion->consultar($centro);
        $departamento=$conexion->consultar($departamento);
        $seccion=$conexion->consultar($seccion);

        $pcentro=$centro[0]['descripcion'];
        $pdepartamento=$departamento[0]['descripcion'];
        $pseccion=$seccion[0]['descripcion'];

        $dnombre = '';
        $dapell1 = '';
        $dapell2 = '';
        UsuariosDePersonal::f_dameNombreApellidosInteresado($pnregpgv,$dnombre, $dapell1, $dapell2);


        $consulta="SELECT activo
                    FROM TPER_FIRMANTES
                    WHERE nregpgv='$pnregpgv' and
                          centro = '$pcodcentro' and
                          departamento = '$pcoddepartamento' and
                          seccion = '$pcodseccion'";

        $res=$conexion->consultar($consulta);
        if ($res<0)return FALSE;
        $pactivo=$res[0]['activo'];
        //inicio y fin dejamos a nula la fecha y hora de fin de activacion.
        if($pactivo=='S'){
                $update="
                        UPDATE
                           TPER_FIRMANTES
                        SET activo = 'N',f_fin_activo=null
                        WHERE nregpgv='$pnregpgv' and
                              centro = '$pcodcentro' and
                              departamento = '$pcoddepartamento' and
                              seccion = '$pcodseccion'";

                $res=$conexion->consultar($update);
                if($res<0)return FALSE;


                // Mensaje a enviar en el correo

                $pmensaje = "El suplente de firma $dnombre $dapell1 $dapell2 del centro $pcentro, departamento
                                $pdepartamento  y sección $pseccion se ha desactivado para firmar como responsable";
                IgepDebug::setDebug(DEBUG_USER,'FIN true f_desactivarSuplenteFirma');
                return TRUE;
        }else
                $pmensaje = "El suplente de firma $dnombre $dapell1 $dapell2 del centro $pcentro, departamento
                        $pdepartamento  y sección $pseccion ya estaba desactivado para firmar como responsable";
        IgepDebug::setDebug(DEBUG_USER,'FIN false f_desactivarSuplenteFirma');
        return TRUE;
}


function f_dameResponsables($pgrupo,$ppfinicio,$ppfsolicitud,$ptitular){
        //Se llama para obtener los destinatarios del correo enviado a
        //los responsables de firmar las solicitudes
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameResponsables-->Grupo:'.$pgrupo);
        $pfinicio = clone $ppfinicio;
        $pfsolicitud = clone $ppfsolicitud;

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $con = new IgepConexion($g_dsn, true);
        //Desglosamos el parámetro "pgrupo" en codcentro, coddepartamento y codsección
        $var_centro=$pgrupo['centro'];
        $var_departamento=$pgrupo['departamento'];
        $var_seccion=$pgrupo['seccion'];

        $pfinicioBD = $con->prepararFecha($pfinicio);
        $pfsolicitudBD = $con->prepararFecha($pfsolicitud);

        $nregpgv= "SELECT nregpgv  FROM TPER_FIRMANTES  WHERE centro = '$var_centro' and
                                    departamento = '$var_departamento' and
                                    seccion = '$var_seccion'  and
                                   (('$pfinicioBD' >= f_inicio and ('$pfinicioBD' <= f_fin or f_fin is null))
                                  or ('$pfsolicitudBD' >= f_inicio and ('$pfsolicitudBD' <= f_fin or f_fin is null)))
                                   and (suplente='N' or (suplente='S' and activo='S' and '$ptitular' = 'N'
        // Añadimos condición para no tener en cuenta suplentes activados después de 1 hora
                                   and (current_timestamp <= f_fin_activo or f_fin_activo is null) ))";


        $res = $con->consultar($nregpgv);
        if($res!=-1){
                $responsables = array();
                foreach($res as $responsable)
                        array_push($responsables,$responsable['nregpgv']);
                IgepDebug::setDebug(DEBUG_USER,'FIN f_dameResponsables');
                return $responsables;
        }
        IgepDebug::setDebug(DEBUG_USER,'FIN f_dameResponsables');
        return -1;
}


    /**
    * f_obtenerGrupo: Obtener el grupo o grupos a los que puede firmar un usuario
    */
function f_obtenerGruposFirma($pnregpgv,$pactivo){
       IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerGruposFirma');
       $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
```

```
        $conexion = new IgepConexion($g_dsn, true);
        $query="SELECT centro as \"centro\",departamento as \"departamento\",seccion as \"seccion\" FROM tper_firmantes f
                WHERE nregpgv = '$pnregpgv' and current_date >= f_inicio AND
               (current_date <= f_fin or f_fin is null) AND
               ((activo='S' and (current_timestamp <= f_fin_activo or f_fin_activo is null)) or ('$pactivo'='N' or
                    suplente='N'))";



        $res = $conexion->consultar($query);
        IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerGruposFirma');
        if(count($res)==0){
          $res=-1;
        }
        return $res;
}


function f_esResponsablePersonalTitular($pnregpgv){
          // Devuelve si la persona pasada por parámetro es la responsable titular del servicio de personal según esté
          establecido en Firmantes

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_esResponsablePersonalTitular');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);
        if($pnregpgv!='')

          $nregpgv= " SELECT nregpgv as \"nregpgv\"  FROM TPER_FIRMANTES".
                        WHERE centro='03' and
                        departamento = '06' and
                        seccion='37' and
                        nregpgv ='".$pnregpgv."' and
                         suplente='N' ";

        $res = $conexion->consultar($nregpgv);
        IgepDebug::setDebug(DEBUG_USER,'FIN f_esResponsablePersonalTitular');
        if(!empty($res[0]['nregpgv']))
            return TRUE;
        else
            return FALSE;
}


function f_esFirmanteTitular($pnregpgv){
          // Devuelve si la persona pasada por parámetro es firmante titular de algún grupo de firma según esté establecido en
          Firmantes

        IgepDebug::setDebug(DEBUG_USER,'INICIO f_esFirmanteTitular');
        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);
        if($pnregpgv!='')

          $nregpgv= " SELECT distinct nregpgv as \"nregpgv\"  FROM TPER_FIRMANTES
                        WHERE nregpgv ='".$pnregpgv."' and
                             suplente='N' ";

        $res = $conexion->consultar($nregpgv);
        IgepDebug::setDebug(DEBUG_USER,'FIN f_esFirmanteTitular');
        if(!empty($res[0]['nregpgv']))
            return TRUE;
        else
            return FALSE;
}


function f_esResponsableDePuertosNoSuplente($pnregpgv){
          //El Jefe de la División de explotación y conservación de puertos es la persona encargada de firmar
          //las solicitudes de todo el personal de puertos, es decir, es la persona responsable que en
          TPER_FIRMANTES
          // tenga como centro 03, departamento 01 y sección 21.
          IgepDebug::setDebug(DEBUG_USER,'INICIO f_esResponsableDePuertosNoSuplente');
          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $con = new IgepConexion($g_dsn, true);
          $query = "SELECT suplente
                      FROM TPER_FIRMANTES
                      WHERE centro='03' and
                         departamento = '01' and
                         seccion='21' and
                         nregpgv = '$pnregpgv'";
          $res = $con->consultar($query);
          IgepDebug::setDebug(DEBUG_USER,'FIN f_esResponsableDePuertosNoSuplente');
          if($res==-1)
                    return FALSE;
          else
                    $var_esResponsableDePuertosSuplente= $res[0]['suplente'];
          if(empty($var_esResponsableDePuertosSuplente))
                    return FALSE;
          else{
                    if($var_esResponsableDePuertosSuplente=='S')
                              return FALSE;
                    else
                              return TRUE;
          }
}



          /**
```

```
    * f_obtenerGrupoNoFichan:
    * Devuelve una cadena de texto con el centro, departamento y secciÃ³n de los responsables de firmar
          * en los centros de trabajo donde no se ficha y las personas que realizan sustituciones en puertos
    **/
function f_obtenerGrupoNoFichan($pccentrab,$pnregpgv,$pcdg,$pcserv,&$pgrupo){
        IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerGrupoNoFichan');

        $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
        $conexion = new IgepConexion($g_dsn, true);

        // Buscamos en tper_personas o tpr2_personas si la persona ficha.
        $query="SELECT ficha as \"ficha\"   FROM tper_personas   WHERE nregpgv = '$pnregpgv' " ;
        $fichaPersona = $conexion->consultar($query);
        if(count($fichaPersona)==0){
                    $query="SELECT ficha as \"ficha\"   FROM tpr2_personas   WHERE nregpgv = '$pnregpgv' " ;
                    $fichaPersona = $conexion->consultar($query);

        }

          // 1) Si no ficha, buscamos en tper_cod_centros_trab para ver si en el centro donde trabaja se ficha.
          if ($fichaPersona[0]['ficha']=='N'){
                    $query="SELECT ficha as \"ficha\"   FROM tper_cod_centros_trab   WHERE ccentrab = '$pccentrab' " ;
                    $fichaCentro = $conexion->consultar($query);
                    // Si se ficha, buscaremos de ese centro de trabajo algún empleado que fiche.

                    if ($fichaCentro[0]['ficha']=='S'){
                              // Añadimos nuevos parametros a la llamada
                              $listaEmpleados = UsuariosDePersonal::f_dameEmpleadosFichanPorCentro($pccentrab,$pcdg,$pcserv);

                              if (count($listaEmpleados)!=0){
                                        $pgrupo= Personal::f_obtenerGrupo($listaEmpleados[0]);
                              }


                    }
          }

           elseif (($fichaPersona[0]['ficha']=='C')){
                    $query="SELECT ficha as \"ficha\",nregpgv as \"nregpgv\"   FROM tper_cod_centros_trab   WHERE ccentrab =
                                                                                                        '$pccentrab' " ;
                    $res = $conexion->consultar($query);
                    $firmante=$res[0]['nregpgv'];
                    $fichaCentro=$res[0]['ficha'];

                    if ($fichaCentro=='N' and count($firmante)!=0){
                              $pgrupo= Personal::f_obtenerGrupo($firmante);

                    }
          }


          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerGrupoNoFichan');

          return 0;
}// FIN f_obtenerGrupoNoFichan


function f_consultarFirmantes($pcentro,$pdepartamento,$pgrupo,$pfirmante,$pfirmado){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_consultarFirmantes');

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);

          // En función de si se quieren visualizar los firmantes de un solicitante o los grupos de firma de un firmante,
          // se utiliza una select u otra, en la que se filtra por los parámetros de entrada

          if(!empty($pfirmado)){

                    // Antes de obtener el grupo de firma del firmado, comprobamos si ficha.
                    // Si no fichase, se buscará el grupo de firma de otra persona de su DG, Serv. y centro de trabajo que sí
                    que fiche,
                    // Si no ficha y tampoco en su centro de trabajo, se obtendrá el responsable de firma del centro, que será
                    lo único que se muestre en pantalla, sin dejar que se edite.

                    // El contenido de f_dameFirmantes() es lo que había aquí con alguna modificación
                    Firmantes::f_dameFirmantes($pfirmado,'S', $resultado, $var_query);

                    if($var_query==''){
                              // Para sacar la informacion en el mensaje del diseño en el caso de que el firmado no ficha
                              // o bien fiche según su centro, y en su centro de trabajo no se ficha
                              return $resultado;
                    }



        }
        else {

          $this->setSelectForSearchQuery("SELECT suplente as \"suplente\", dapell1||' '||CASE WHEN dapell2 is null THEN ''
                                                ELSE dapell2 END||', '||dnombre  as \"Nombre\",
                                                tper_firmantes.nregpgv , centro, departamento,seccion,
                                                twps_centros.descripcion as \"nomcentro\",
                                                twps_departamentos.descripcion as \"nomdepartamento\",
                                                twps_secciones.descripcion as \"nomseccion\",suplente,activo" .
                                          FROM tper_firmantes , vper_ocupacion_historico , twps_centros , twps_departamentos ,
                                                twps_secciones  ");
          $this->setWhereForSearchQuery("tper_firmantes.nregpgv = vper_ocupacion_historico.nregpgv AND tper_firmantes.centro =
                                          twps_centros.codigo AND tper_firmantes.departamento = twps_departamentos.codigo
                                           AND tper_firmantes.seccion = twps_secciones.codigo");
```

```
          $str_where = '';
          if(!empty($pcentro)){
                    if($str_where!='')
                    $str_where.=' AND ';
                    $str_where.= " tper_permisos.centro = '".$pcentro."' ";
                    if(!empty($pdepartamento)){
                              if($str_where!='')
                              $str_where.=' AND ';
                              $str_where.= " tper_permisos.departamento = '".$pdepartamento."' ";
                              if(!empty($pgrupo)){
                                        if($str_where!='')
                                                  $str_where.=' AND ';
                              $str_where.= " tper_permisos.seccion = '".$pgrupo."' ";

                              }
                    }

          }

          if(!empty($pfirmante)){
                    if($str_where!='')
                              $str_where.=' AND ';
                    $str_where.= " tper_permisos.nregpgv = '".$pfirmante."' ";

          }


        }
        $this->setOrderByForSearchQuery("twps_centros.descripcion,twps_departamentos.descripcion,twps_secciones.descripcion
                                                                                                                  ");

        //La select que editamos, es decir la que mostramos en el panel de ediciÃ³n (ficha)
          $this->setSelectForEditQuery("SELECT substr(tper_firmantes.nregpgv, 1,8) as \"DNI\",suplente as \"suplente\",
                              dapell1||' '||CASE WHEN dapell2 is null THEN '' ELSE dapell2 END||', '||dnombre  as \"Nombre\",
                              tper_firmantes.nregpgv , tper_firmantes.centro as \"fil_centro\",
                              tper_firmantes.departamento as \"fil_departamento\",tper_firmantes.seccion as \"fil_seccion\",
                              f_inicio, f_fin,suplente,activo, twps_centros.descripcion as \"nomcentro\",
                              twps_departamentos.descripcion as \"nomdepartamento\",
                              twps_secciones.descripcion as \"nomseccion\"
                    FROM tper_firmantes, vper_ocupacion_historico, twps_centros, twps_departamentos, twps_secciones ");

          $this->setWhereForEditQuery("tper_firmantes.nregpgv = vper_ocupacion_historico.nregpgv AND tper_firmantes.centro = t
                                        wps_centros.codigo AND tper_firmantes.departamento = twps_departamentos.codigo AND
                                        tper_firmantes.seccion = twps_secciones.codigo");
          $this->setOrderByForEditQuery("4");

          IgepDebug::setDebug(DEBUG_USER,'FIN f_consultarFirmantes');

          return 0;
}// FIN f_consultarFirmantes


    /**
    * f_obtenerFirmanteOEmpleadoGrupoNoFichan:
    * Devuelve un empleado del grupo en que sÃ que se ficha, o el responsable de firma del centro en el que no se ficha
          **/
function f_obtenerFirmanteOEmpleadoGrupoNoFichan($pccentrab, $pnregpgv,$pcdg, $pcserv, &$pfichaCentro, &$pfirmante){

          IgepDebug::setDebug(DEBUG_USER,'INICIO f_obtenerFirmanteOEmpleadoGrupoNoFichan');

          $g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
          $conexion = new IgepConexion($g_dsn, true);

          // Buscamos en tper_personas o tpr2_personas si la persona ficha.
          $query="SELECT ficha as \"ficha\"   FROM tper_personas   WHERE nregpgv = '$pnregpgv' " ;
          $res = $conexion->consultar($query);
          if(count($res)==0){
          $query="SELECT ficha as \"ficha\"   FROM tpr2_personas   WHERE nregpgv = '$pnregpgv' " ;
          $res = $conexion->consultar($query);
          }

          // 1) Si no ficha, buscamos en tper_cod_centros_trab para ver si en el centro donde trabaja se ficha.
          if ($res[0]['ficha']=='N'){
                    $query="SELECT ficha as \"ficha\",nregpgv as \"nregpgv\"   FROM tper_cod_centros_trab   WHERE ccentrab =
                                                                                                        '$pccentrab' " ;
                    $res2 = $conexion->consultar($query);
                    $pfirmante = $res2[0]['nregpgv'];
                    $pfichaCentro = $res2[0]['ficha'];

                    // Si se ficha, buscaremos de ese centro de trabajo algún empleado que fiche.
                    if ($res2[0]['ficha']=='S'){
                              $listaEmpleados = UsuariosDePersonal::f_dameEmpleadosFichanPorCentro($pccentrab,$pcdg,$pcserv);
                              if (count($listaEmpleados)!=0){
                                        $pfirmante=$listaEmpleados[0];
                              }
                    }
                    elseif ($res2[0]['c']=='N'){
                    $pfichaCentro = $res2[0]['ficha'];
                    $pfirmante=$res2[0]['nregpgv'];
                    return -1;
                    }
          }
          // 2) Si ficha según centro, buscamos en tper_cod_centros_trab para ver si en el centro donde trabaja no se ficha.
          elseif (($res[0]['ficha']=='C')){
                    $query="SELECT ficha as \"ficha\",nregpgv as \"nregpgv\"   FROM tper_cod_centros_trab   WHERE ccentrab =
                                                                                                        '$pccentrab' " ;
          $res3 = $conexion->consultar($query);
          $pfirmante=$res3[0]['nregpgv'];
          $pfichaCentro=$res3[0]['ficha'];

          }
          IgepDebug::setDebug(DEBUG_USER,'FIN f_obtenerFirmanteOEmpleadoGrupoNoFichan');
```

```
		return 0;
}// FIN f_obtenerFirmanteOEmpleadoGrupoNoFichan



		/**
	* f_dameFirmantes:
	* Devuelve los firmantes de una determinada persona o la query que lo obtiene
		**/
function f_dameFirmantes($pnregpgv,$pconsultaFirmantes,&$presultado,&$pquery){

		IgepDebug::setDebug(DEBUG_USER,'INICIO f_dameFirmantes');

		$g_dsn = ConfigFramework::getConfig()->getDSN('g_dsn');
		$conexion = new IgepConexion($g_dsn, true);

		// Antes de obtener el grupo de firma del firmado, comprobamos si ficha.
		// Si no fichase, se buscará el grupo de firma de otra persona de su DG, Serv. y centro de trabajo que sí que fiche,
		//  Si no ficha y tampoco en su centro de trabajo, se obtendrá el responsable de firma del centro, que será lo único
		//  que se muestre en pantalla, sin dejar que se edite.


		// Inicializamos los valores de retorno
		$presultado = array();
		$pquery = "";


		if (!UsuariosDePersonal::f_noFichan($pnregpgv)){
			$grupoFirma = Personal::f_obtenerGrupo($pnregpgv);
		}
		else{
			UsuariosDePersonal::f_obtenerCdgCserv($pnregpgv, $cdg, $cserv);
			UsuariosDePersonal::f_dameCentroTrab($pnregpgv, $ccentrab);
			Firmantes::f_obtenerFirmanteOEmpleadoGrupoNoFichan($ccentrab, $pnregpgv,$cdg, $cserv, $ficha,
												$firmanteGrupoNoFichan);

			if(!empty($firmanteGrupoNoFichan) or $firmanteGrupoNoFichan!='' ){
				if ($ficha=='S'){
					$grupoFirma = Personal::f_obtenerGrupo($firmanteGrupoNoFichan);
				}
				elseif ($ficha=='N'){

					if ($pconcultaFirmantes=='S'){
						// Para sacar la informacion en el mensaje del diseño en el caso de que el
						firmado no ficha
						// o bien fiche según su centro, y en su centro de trabajo no se ficha
						$presultado['ficha']=$ficha;
						$presultado['firmanteGrupoNoFichan']=$firmanteGrupoNoFichan;
					}
					else{
						$presultado['firmanteGrupoNoFichan']=$firmanteGrupoNoFichan;
					}


				}

			}
		}


		if(!empty($grupoFirma)){
			$var_centro = $grupoFirma[centro];
			$var_departamento = $grupoFirma[departamento];
			$var_seccion = $grupoFirma[seccion];
			// Query básica sobre la que se van a añadir las condiciones de búsqueda
			$pquery ="SELECT suplente as \"suplente\", dapell1||' '||CASE WHEN dapell2 is null THEN '' ELSE dapell2
													END||', '||dnombre  as \"Nombre\",
					tper_firmantes.nregpgv , centro, departamento,seccion,
					twps_centros.descripcion as \"nomcentro\",
					twps_departamentos.descripcion as \"nomdepartamento\",
					twps_secciones.descripcion as \"nomseccion\",suplente,activo
				FROM tper_firmantes, vper_ocupacion_historico, twps_centros, twps_departamentos, twps_secciones

				WHERE	tper_firmantes.nregpgv = vper_ocupacion_historico.nregpgv AND tper_firmantes.centro =
					twps_centros.codigo AND tper_firmantes.departamento = twps_departamentos.codigo
					AND tper_firmantes.seccion = twps_secciones.codigo
					AND tper_firmantes.centro = '$var_centro'
					AND tper_firmantes.departamento = '$var_departamento'
					AND tper_firmantes.seccion = '$var_seccion'
				ORDER BY tper_firmantes.suplente,Nombre ";




		}

		// Si no tenemos grupo de firma pero sí firmante, mostramos sólo el firmante
		else {

			if(!empty($firmanteGrupoNoFichan)){
				$pquery="SELECT 'N', h.dapell1 || ' ' || h.dapell2 || ' ' || h.dnombre as nombre,h.nregpgv,
						nulo, nulo, nulo, nulo, nulo, nulo, nulo
					FROM VPER_OCUPACION_HISTORICO h
					WHERE h.nregpgv = '$var_centro'
					ORDER BY nombre";
			}

		}
```

```
        IgepDebug::setDebug(DEBUG_USER,'FIN f_dameFirmantes');

        return 0;
}// FIN f_dameFirmantes


}
?>
```