ENTREGA SUBEQUIPO DE DISEÑO CORRECCIÓN ERRORES ITERACIÓN 1

Moisés Gautier Gómez
Francisco Javier Gómez del Olmo
Julio Ros Martínez

3 DE ABRIL DE 2013

Universidad de Granada

Contenido

Control de Versiones

- 1º Control de versión

Fecha	Versión	Descripción
3/4/2013	1.1	Entrega subequipo de diseño: Corrección de errores de la iteración 1

Modelado de requisitos

Modelado funcional – Diagramas de casos de uso

En esta etapa del modelado de requisitos se captura el propósito general del sistema:

- Se analiza qué debe hacer el sistema.
- Se obtiene una visión contextualizada del sistema.
- Identifica y delimita el sistema.
- Se determinan características, cualidades y restricciones que debe satisfacer el sistema.

Los requisitos funcionales que se han obtenido en el sistema son los siguientes:

- Un entrenador puede modificar los datos de sus alumnos relativos al grupo (comentarios, observaciones).
- El administrador es el encargado de gestionar las fichas de los alumnos, así como algunas actividades esporádicas.
- La notificación de los pagos mensuales de los alumnos se recibirá mediante un tipo de archivo procesable por nuestro sistema (sql, xml, etc).
- Hay que llevar un inventario de las equipaciones entregabas o no por cada temporada.
- Cada grupo de entrenamiento se compone de entre 15-20 alumnos y se distribuyen entre dos días de la semana a especificar.
- No puede haber dos grupos de entrenamiento a la misma hora, al mismo día y en la misma pista de la instalación.
- Un alumno solo puede jugar en el equipo al que está inscrito y puede competir en dos equipos, uno de su categoría y otro de una superior.
- La edad mínima de los alumnos tiene que ser 5 años, siendo la máxima 18 años.
- Los alumnos menores de 9 años, no tienen porque pertenecer a un equipo.
- Los alumnos mayores de 9 años tienen que competir en al menos un equipo.
- La categoría benjamín y alevín solo pueden competir y entrenar en las pistas de mini-basket de las instalaciones.
- Se hará distinción entre 2 tipos de usuario para el acceso: administrador y entrenador. Cada uno de ellos podrá visualizar una parte distinta de la aplicación dependiendo de los privilegios que tenga.
- El administrador será el encargado de introducir los datos de los entrenadores en el sistema y proporcionar los usuarios y claves asociadas a cada entrenador.
- El administrador podrá modificar las contraseñas del sistema una vez creadas las cuentas en el mismo.
- Hay 5 categorías posibles: benjamín, alevín, infantil, cadete y junior.

En nuestro sistema hemos identificado los siguientes casos de uso:

Administrador	**Entrenador**
✓ Acceso al sistema ✓ Dar de alta alumno ✓ Dar de baja alumno ✓ Modificar alumno ✓ Consultar alumno ✓ Dar de alta usuario ✓ Dar de baja usuario ✓ Modificar usuario ✓ Consultar usuario	✓ Acceso al sistema ✓ Modificar alumno ✓ Consultar alumno

El **actor administrador** será el encargado de realizar la mayor parte de operaciones del sistema para las altas, bajas y modificaciones de los datos del mismo.

El **actor entrenador** sólo podrá acceder a ciertos contenidos del sistema para consultar información o para en ciertos casos, modificar información relativa a los alumnos en el campo observaciones del mismo.

A continuación se detallan la descripción de los casos de uso del sistema.

Caso de uso: Acceso al sistema

Nombre del CU	Acceso al sistema
Resumen	El usuario accede al sistema identificándose mediante su DNI (usuario) y su clave.
Dependencias	-
Actores	Administrador, Entrenador.
Precondiciones	Que exista el usuario en el sistema.
Postcondición	El usuario identificado accede al sistema.
Curso normal	1.-El usuario accede a la ventana de login del sistema. 2.-El sistema requiere la introducción de su DNI y su clave. 3.-El usuario introduce su DNI también su clave y pulsa “Aceptar”. 4.-Si el DNI y la clave son correctos el sistema da acceso al usuario.
Cursos alternativos	4a.- Si el DNI y/o la clave no son correctos el sistema devuelve un mensaje de error y vuelve al paso 2.
Observaciones	-
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Casos de uso: Gestión de alumnos

Nombre del CU	Dar de alta alumno
Resumen	El administrador introduce un nuevo alumno en el sistema con todos los datos personales necesarios.
Dependencias	-
Actores	Administrador.
Precondiciones	El administrador debe estar identificado en el sistema.
Postcondición	Un nuevo alumno quedará registrado en el sistema.
Curso normal	1.-El administrador selecciona la opción “Dar de alta un alumno”. 2.-El sistema muestra un formulario con los campos requeridos para los datos del nuevo alumno. 3.-El administrador introduce los datos y pulsa “Aceptar”. 4.-El sistema almacena los datos personales del nuevo alumno.
Cursos alternativos	3a.- El administrador pulsa “Cancelar”. Se termina el caso de uso sin almacenar nada. 4a.- El alumno ya existe en el sistema. El sistema informa del error y vuelve al paso 2. 4b.- Falta algún dato por rellenar. El sistema informa del error y vuelve al paso 2.
Observaciones	-
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Nombre del CU	Modificar alumno
Resumen	El actor modifica los datos de un alumno del sistema.
Dependencias	Extiende al caso de uso “Consultar alumno”.
Actores	Administrador, Entrenador.
Precondiciones	El actor debe estar identificado en el sistema. El alumno a modificar debe existir en la BD.
Postcondición	Se almacenará en el sistema los cambios realizados en los datos del alumno.
Curso normal	1.-El actor modifica los datos que desea y pulsa “Aceptar”. 2.-El sistema almacena en la BD los cambios realizados.
Cursos alternativos	2.- No se puede modificar el alumno o datos introducidos erróneos. Informa de un error y termina el caso de uso.
Observaciones	-
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Nombre del CU	Eliminar alumno
Resumen	El administrador elimina los datos de un alumno existente en el sistema.
Dependencias	-
Actores	Administrador.
Precondiciones	El administrador debe estar identificado en el sistema. El alumno que se va a eliminar debe existir en el sistema.
Postcondición	Los datos del alumno seleccionado se eliminan del sistema.
Curso normal	1.-El administrador selecciona la opción “Eliminar alumno”. 2.-El sistema muestra los campos del alumno para buscar un alumno por uno de sus atributos. 3.-El administrador rellena uno de los campos del alumno para buscarlo y pulsa el botón “Aceptar”. 4.-El sistema muestra una lista con los alumnos que coinciden con la búsqueda. 5.-El administrador selecciona el alumno que desea eliminar. 6.-El sistema muestra un mensaje de confirmación de baja del correspondiente alumno. 7.-El administrador confirma la decisión pulsando “Aceptar”. 8.-El sistema elimina del sistema al alumno seleccionado.
Cursos alternativos	-
Observaciones	-
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Nombre del CU	Consultar alumno
Resumen	El usuario consulta los datos de un alumno existente en el sistema.
Dependencias	Extendido por el caso de uso “Modificar alumno”.
Actores	Administrador, Entrenador.
Precondiciones	El usuario debe estar identificado en el sistema. El alumno a consultar debe existir en el sistema.
Postcondición	Se muestran por pantalla los datos relativos al alumno seleccionado.
Curso normal	1.-El usuario selecciona la opción “Consultar alumno” 2.-El sistema muestra los campos del alumno para buscar un alumno por uno de sus atributos. 3.-El administrador rellena uno de los campos del alumno para buscarlo y pulsa “Aceptar”. 4.-El sistema muestra una lista con los alumnos que coinciden con la búsqueda. 5.- Opcionalmente (según se quieran modificar o no los datos) se inicia el caso de uso “Modificar alumno”.
Cursos alternativos	-
Observaciones	El entrenador sólo podrá modificar una parte de los datos y no la totalidad de la ficha entera del alumno.
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Casos de uso: Gestión de usuarios

Nombre del CU	Dar de alta usuario
Resumen	Esta operación permite al actor introducir en el sistema información relacionada con un nuevo usuario
Dependencias	-
Actores	Administrador
Precondiciones	La información del usuario no debe existir previamente en el sistema de información (base de datos). El actor debe estar identificado en el sistema.
Postcondición	Se crea una nueva instancia en la base de datos del sistema con la información asociada al usuario.
Curso normal	1. El administrador selecciona la opción “Dar de alta un usuario” 2. El sistema muestra un formulario con los campos a rellenar de un nuevo usuario. 3. El administrador rellena el formulario y pulsa “Aceptar”. 4. El sistema comprueba el DNI y almacena los datos de un nuevo usuario en la BD.
Cursos alternativos	3.1 El administrador pulsa cancelar. Se finaliza el caso de uso sin guardar la información. 4.1 El DNI introducido ya corresponde a un usuario almacenado en la BD. Devuelve un mensaje de error y vuelve al paso 3.
Observaciones	-
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Nombre del CU	Modificar usuario
Resumen	El actor modifica los datos de un usuario del sistema.
Dependencias	Extiende al caso de uso “Consultar usuario”.
Actores	Administrador.
Precondiciones	El actor debe estar identificado en el sistema. El usuario a modificar debe existir en la BD.
Postcondición	Se almacenará en el sistema los cambios realizados en el usuario.
Curso normal	1.-El actor modifica los datos que desea y pulsa “Aceptar”. 2.-El sistema almacena en la BD los cambios realizados.
Cursos alternativos	2.- No se puede almacenar el usuario en la BD. Informa de un error y termina el caso de uso.
Observaciones	Sólo el administrador podrá modificar alguno de los cambios del sistema.
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Nombre del CU	Eliminar usuario
Resumen	Esta operación permite al actor eliminar del sistema información relacionada con un usuario existente.
Dependencias	-
Actores	Administrador
Precondiciones	Debe de existir el usuario que se quiere eliminar para poder llevar a cabo la operación. El actor debe estar identificado en el sistema.
Postcondición	Se eliminan los datos pertinentes del usuario del sistema (base de datos) y se reorganiza el contenido.
Curso normal	1.-El administrador selecciona la opción “Eliminar usuario”. 2.-El sistema muestra los campos del usuario para buscar un usuario por uno de sus atributos. 3.-El administrador rellena uno de los campos del usuario para buscarlo y pulsa el botón “Aceptar”. 4.-El sistema muestra una lista con los usuarios que coinciden con la búsqueda. 5.-El administrador selecciona el usuario que desea eliminar. 6.-El sistema muestra un mensaje de confirmación de baja del correspondiente usuario. 7.-El administrador confirma la decisión pulsando “Aceptar”. 8.-El sistema elimina del sistema al usuario seleccionado.
Cursos alternativos	-
Observaciones	Si sólo existe un administrador, no se puede eliminar.
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Nombre del CU	Consultar usuario
Resumen	Esta operación permite al actor consultar del sistema información relacionada con un usuario existente.
Dependencias	Extendido por el caso de uso “Modificar usuario”.
Actores	Administrador.
Precondiciones	Debe de existir el usuario que se quiere consultar para poder llevar a cabo la operación. El actor debe estar identificado en el sistema.
Postcondición	El sistema muestra por salida estándar (pantalla) los resultados de la búsqueda del usuario.
Curso normal	1.-El usuario selecciona la opción “Consultar usuario” 2.-El sistema muestra los campos del usuario para buscar un usuario por uno de sus atributos. 3.-El administrador rellena uno de los campos del usuario para buscarlo y pulsa “Aceptar”. 4.-El sistema muestra una lista con los usuarios que coinciden con la búsqueda. 5.- Opcionalmente (según se quieran modificar o no los datos) se inicia el caso de uso “Modificar usuario”.
Cursos alternativos	-
Observaciones	-
Requisitos no funcionales	Cuando el sistema detecte una anomalía mostrará al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

Identificar subsistemas – Diagramas de casos de uso

A continuación se detallan los distintos diagramas de casos de uso de los que se compone el sistema analizado.

Diagrama de CU: Acceso al sistema

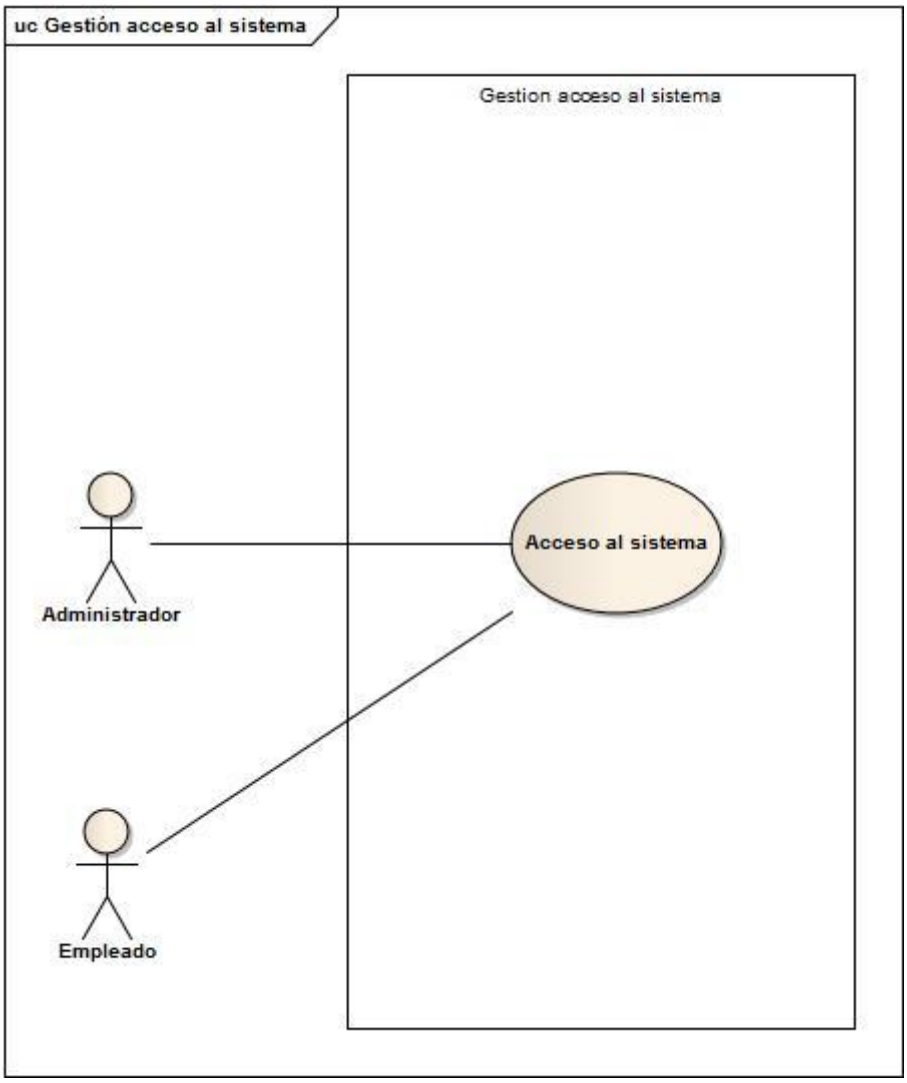


Diagrama de CU: Gestión de alumnos

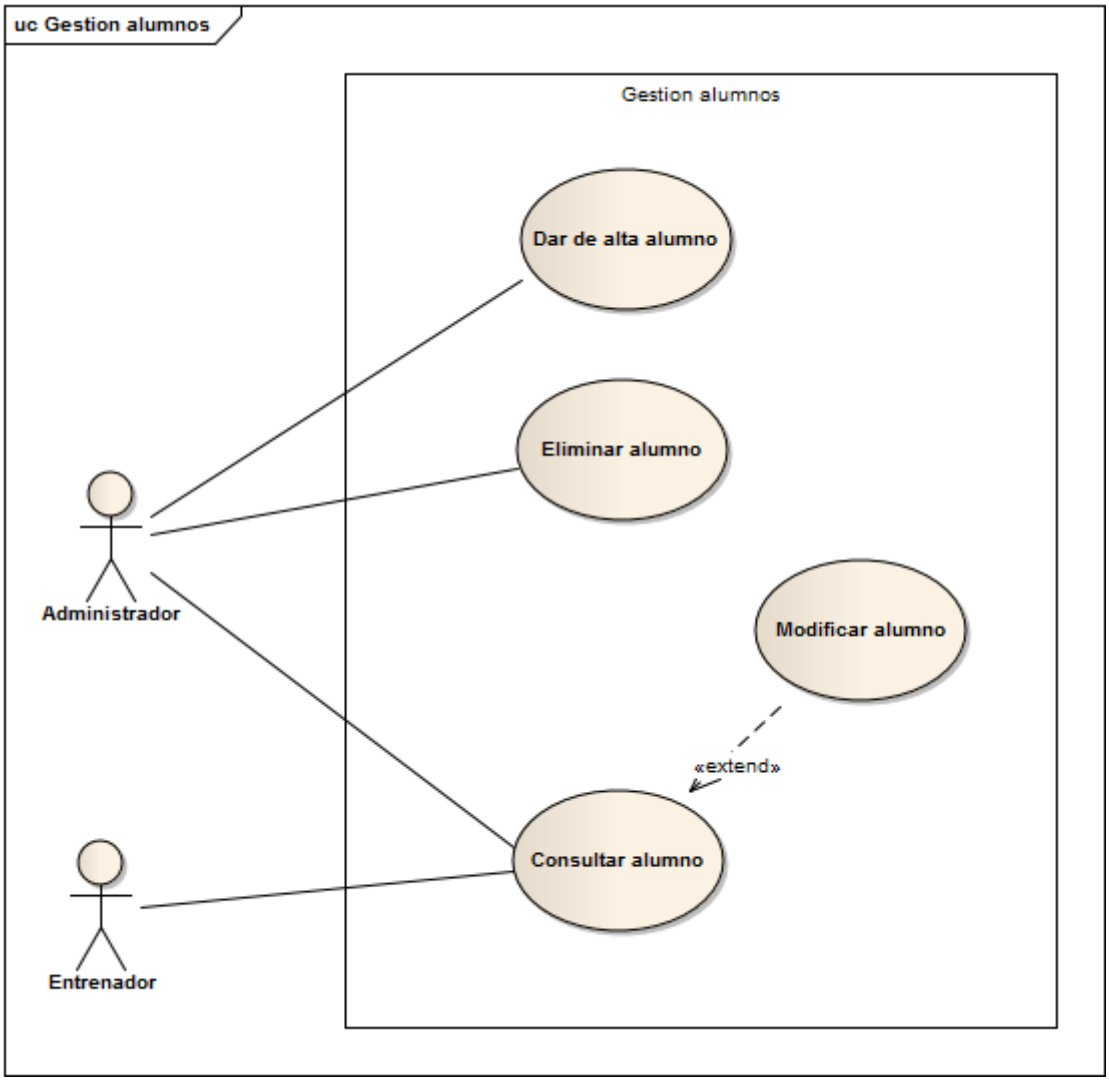
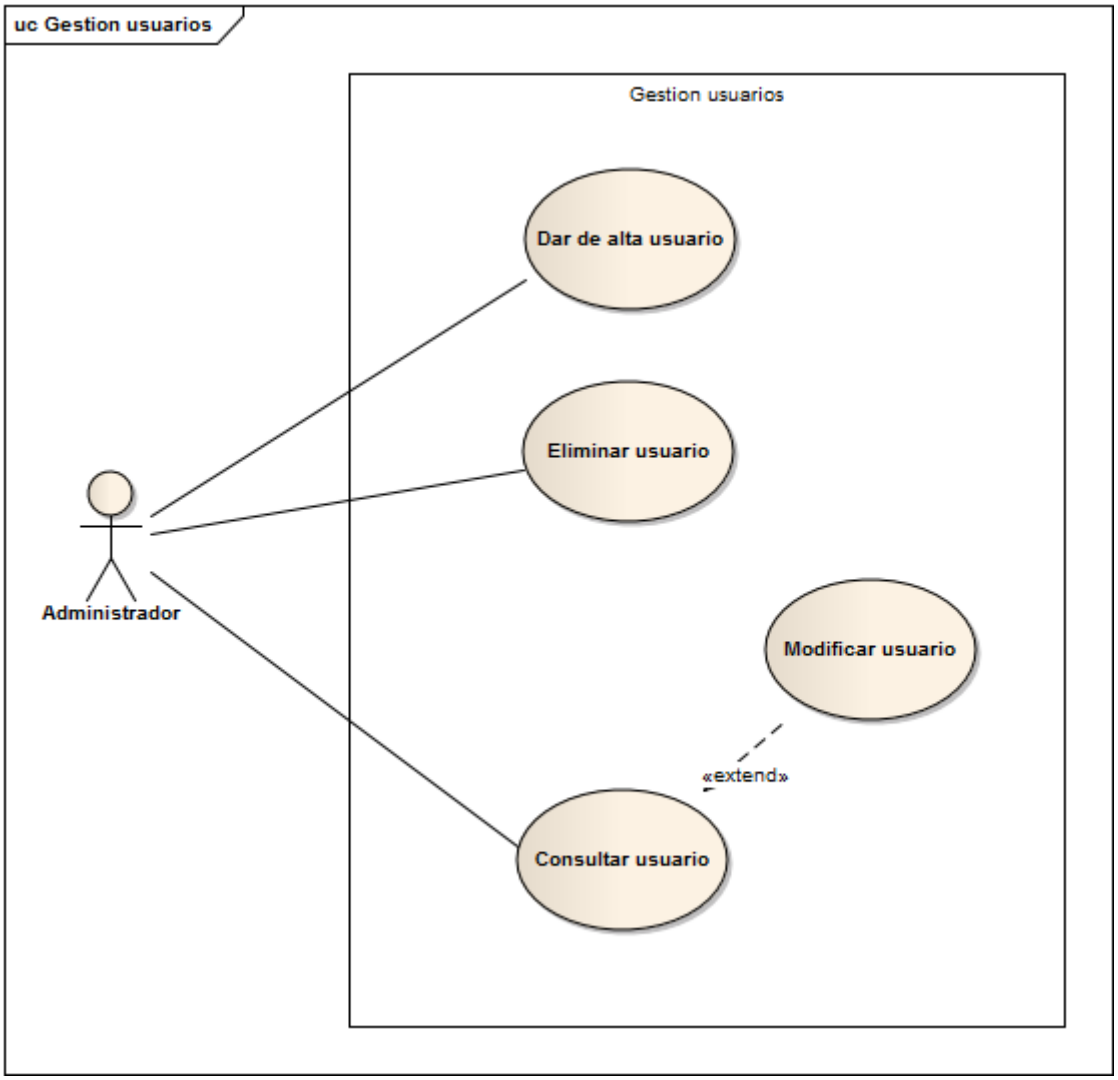


Diagrama de CU: Gestión de usuarios



Requisitos no funcionales

Estos requisitos son aquellos que describen cualidades o restricciones del sistema que no se relacionan de forma directa con el comportamiento funcional del mismo. A continuación se especifican los más importantes del sistema:

- No requiere un conocimiento específico para la utilización del software.

- La aplicación tendrá un manual de uso.

- La base de datos estará implementada en un lenguaje objeto relacional como mysql.

- La aplicación estará realizada en el lenguaje de programación Java.

- Se creara un script de tarea programada (cron) para la copia de seguridad de la base de datos debido a alguna inconsistencia del sistema.

- La interfaz debe reflejar claramente la distinción entre las distintas partes del sistema.

- La documentación del código fuente será llevada a cabo mediante la aplicación javadoc.

- El sistema se desplegara sobre una distribución Microsoft Windows.

- El administrador se encargara de tareas de mantenimiento.

- Cuando el sistema detecte una anomalía mostrara al usuario el mensaje de error pertinente y abortara la ejecución del proceso.

- El código fuente de la aplicación seguirá un estilo uniforme y normalizado para todos los módulos del mismo.

- El formato de las fechas será dd/mm/yy.

- El control de asistencia no es controlado, pero se puede incluir en la sección de observaciones de los alumnos.

Operaciones del sistema – Diagramas de secuencia

Se van a describir 2 diagramas de secuencia de 2 operaciones del sistema de los caso de uso más importantes del mismo.

Diagrama de secuencia: Dar de alta usuario

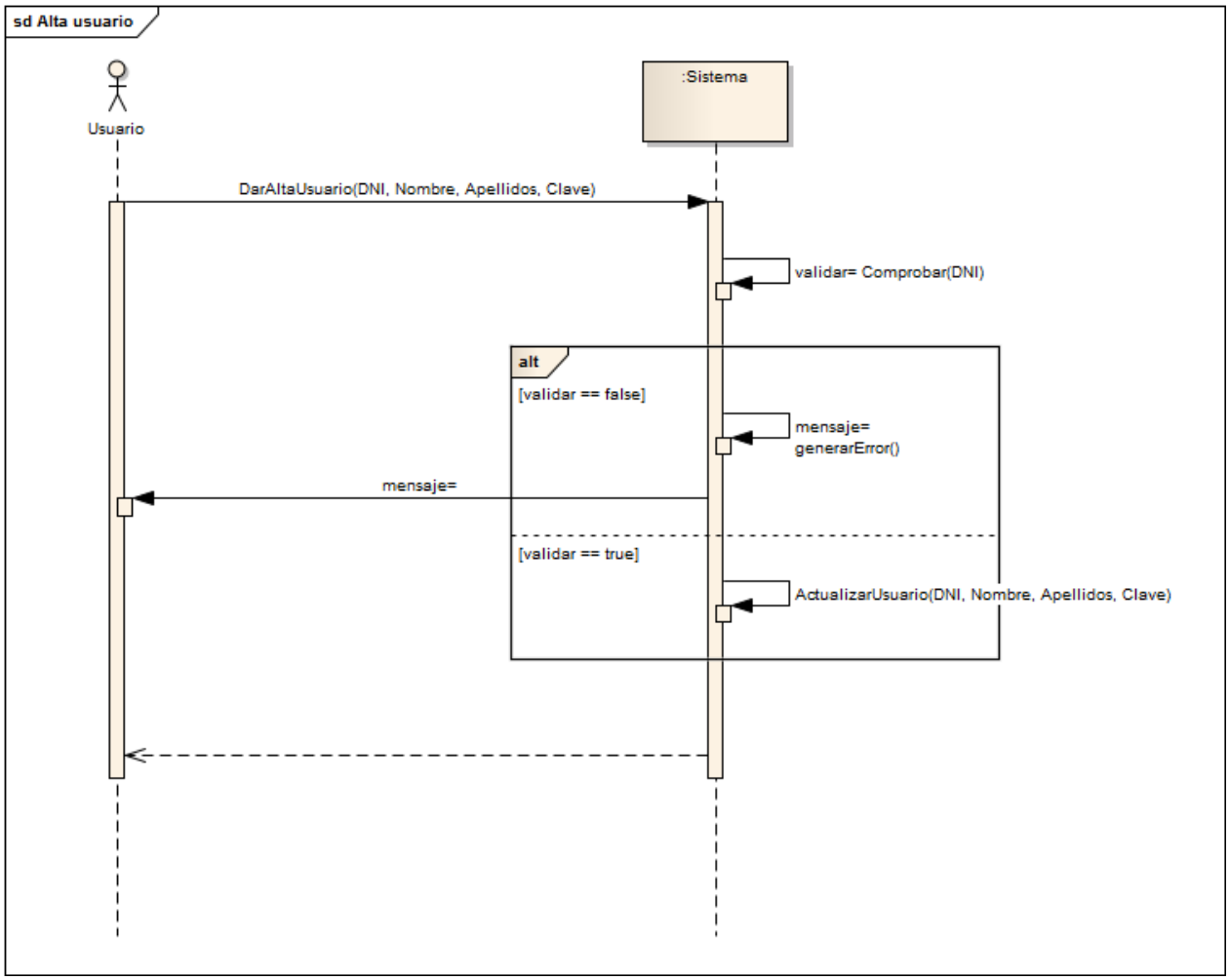
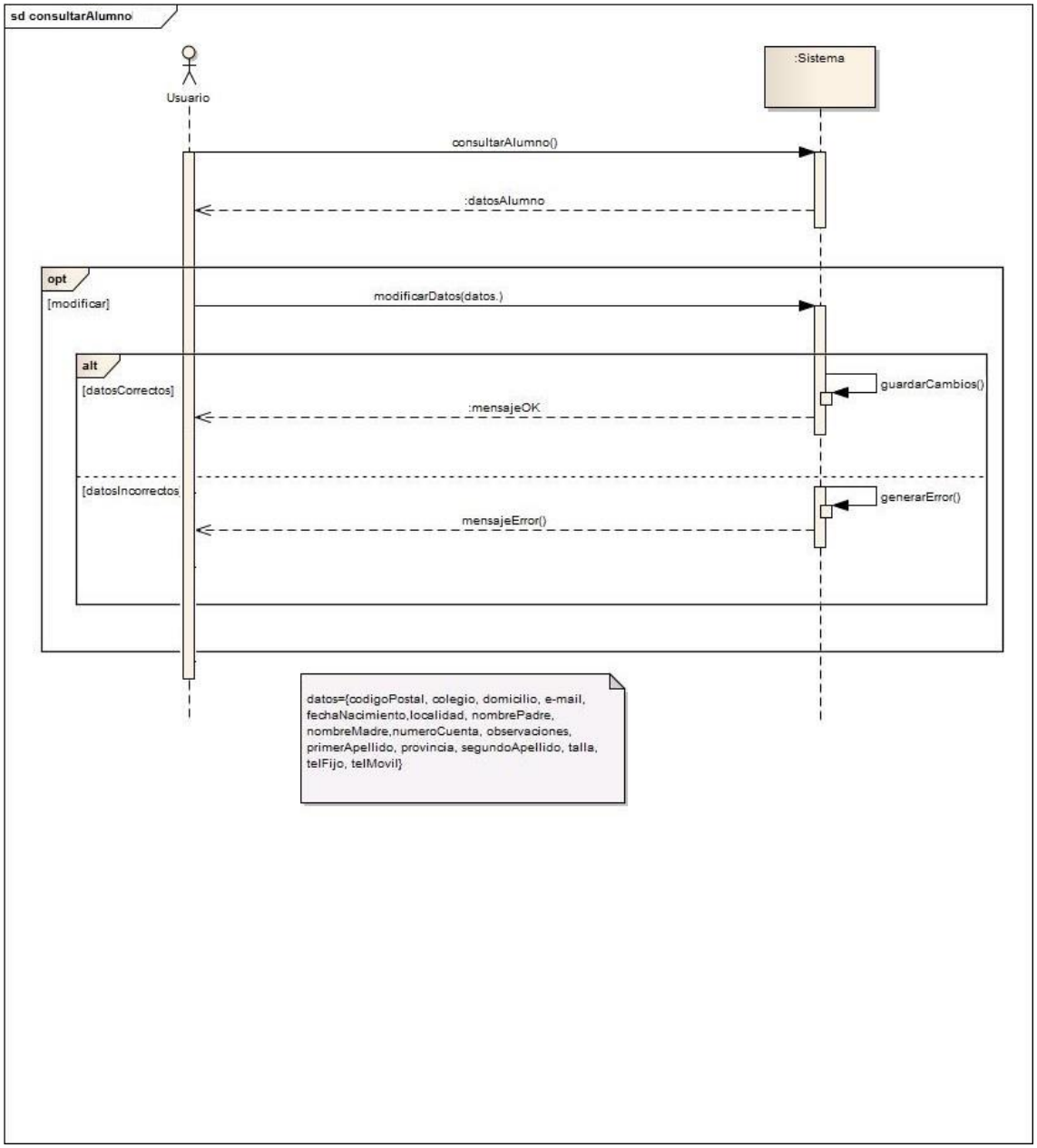


Diagrama de secuencia: Consultar alumno



Análisis

Identificar clases, atributos y relaciones

En este punto se buscan las abstracciones más significativas que identifiquen los aspectos claves del dominio del problema. Las clases conceptuales deben interrelacionarse entre sí a través de las relaciones para satisfacer las necesidades de información o de comportamiento que demandan los casos de uso y así comprender mejor el modelo.

A continuación se detallan las clases, atributos y relaciones entre las clases que se han obtenido para esta primera iteración del problema planteado.

Clase: Fundación

Atributos:

- ciudad (String): Nombre la ciudad o localidad de la fundación de baloncesto.
- codPostal (int): Código postal de la ciudad o localidad de la fundación de baloncesto.
- direccion (String): Dirección postal de la ciudad o localidad de la fundación de baloncesto.
- email (String): Correo eléctronico de la fundación.
- nombre (String): Nombre completo de la fundación de baloncesto.
- numeroCuenta (String): Número de cuenta bancario de la fundación donde se realizaran los ingresos de las mensualidades.
- telefono (int): Número de teléfono de la fundación con el prefijo de la ciudad o localidad incluido.

Relaciones:

- Asociación con la clase Equipo.

Clase: Categoría

Atributos:

- descripción (String): Breve descripción de la categoría del alumno.
- edadMaxima (int): La edad máxima posible a la que se puede acceder a la categoría.
- idCategoria (int): Número identificador de la categoría.
- tipo (String): Nombre de la categoría del alumno.

Relaciones:

- Asociación con la clase Equipo.
- Asociación con la clase Grupo.

Clase: Equipo

Atributos:

- idEquipo (int): Número identificador del equipo.

Relaciones:

- Asociación con la clase Categoría.
- Asociación con la clase Entrenador.
- Asociación con la clase Fundación.
- Asociación con la clase Alumno.
- Asociación con la clase Temporada.

Clase: Temporada

Atributos:

- curso (string): año correspondiente al curo de la temporada (ejemplo: 2012-2013 sería 12/13).
- idTemporada (int): Número identificador de la temporada.

Relaciones:

- Asociación con la clase Equipo.
- Asociación con la clase Grupo.
- Asociación con la clase Alumno.
- Asociación con la clase Temporada.

Clase: Rango (clase de asociación entre las clases Entrenador y Equipo)

Atributos:

- idRango (int): Número identificador del rango
- tipo (TipoEntrenador): Campo que describe que tipo de entrenador es para el equipo, si el primero o el segundo, mediante el tipo enumerado TipoEntrenador.

Relaciones:

- Clase de asociación entre Entrenador y Equipo.

Clase: Pago Temporada (clase de asociación entre las clases Temporada y Alumno)

Atributos:

- idPagoTemporada (int): Número identificador del pago por temporada.

Relaciones:

- Asociación con la clase Cuota precio (composición de esta última)

Clase: Cuota Precio

Atributos:

- idCuotaPrecio (int): Número identificador del precio de la cuota.
- importe (float): Valor numérico del importe de la mensualidad del alumno.
- pagado (boolean): Valor booleano que establece si la mensualidad ha sido pagada o no por el alumno. Se modificará una vez se haya recibido del banco el documento con los pagos del mes.
- fechaPago (date): Fecha en la que se realizó el pago.

Relaciones:

- Asociación con la clase Pago Temporada.
- Asociación con la clase Pago Actividad.

Clase: Alumno

Atributos:

- codigoPostal (int): Código postal de la ciudad o localidad donde resida el alumno.
- colegio (string): Colegio al que está inscrito durante el curso.
- domicilio (string): Dirección postal donde vive el alumno.
- email (string): Correo electrónico del alumno o del tutor para recibir notificaciones pertinentes de la fundación o algún otro relacionado.
- fechaNacimiento (date): Fecha de nacimiento del alumno para establecer la categoría a la que pertenece.
- idAlumno (int): Número identificador del alumno.
- localidad (string): Nombre de la ciudad o localidad donde resida el alumno.
- nombre (string): Nombre del alumno.
- nombreMadre (string): Nombre de la madre del alumno (no tiene por qué llevar asociado el apellido ya que se podría extraer de los apellidos del alumno).
- nombrePadre (string): Nombre del padre del alumno (no tiene por qué llevar asociado el apellido ya que se podría extraer de los apellidos del alumno).
- numeroCuenta (string): Número de cuenta bancario donde el alumno domiciliará las mensualidades de la fundación de baloncesto.
- observaciones (string): Campo dedicado a datos que los entrenadores del alumno quieran anotar de su rendimiento o como campo para anotaciones de cualquier índole.
- primerApellido (string): Primer apellido del alumno.

- provincia (string): Provincia a la que pertenece la ciudad o localidad del alumno durante el curso.
  segundoApellido (string): Segundo apellido del alumno.
- talla (TallaAlumno): Este campo corresponde con la talla del alumno que pertenece al tipo enumerado TallaAlumno en donde se establecen todos los valores posibles que puede tomar el campo.
- telFijo (int): Número de teléfono fijo del alumno/tutor durante el curso.
- telMovil (int): Número de teléfono móvil del alumno/tutor durante el curso.

Relaciones:

- Asociación con la clase Temporada.
- Asociación con la clase Grupo.
- Asociación con la clase Actividad.

Clase: Entrenador (clase de especialización de Usuario)

Atributos:

Relaciones:

- Asociación con la clase Equipo.
- Asociación con la clase Grupo.

Clase: Grupo Entrenamiento

Atributos:

- horarios (date): Horarios del grupo de entrenamiento en donde está inscrito el alumno durante el curso. El dato contendrá los días de entrenamiento para la categoría a la que pertenezca, así como las horas disponibles por cada dupla de días.
- idGrupo (int): Número identificador del alumno.

Relaciones:

- Asociación con la clase Entrenador.
- Asociación con la clase Categoría.
- Asociación con la clase Alumno.
- Asociación con la clase Temporada.

Clase: Usuario

Atributos:

- clave (string): Clave del usuario para acceder a la aplicación. En principio su almacenamiento podrá seguir algún tipo de cifrado seguro como MDA5.
- dni (string): DNI del usuario.
- idUsuario (int): Número identificador del usuario

- nombre (string): Nombre del usuario.
- primerApellido (string): Primer apellido del usuario.
- segundoApellido (string): Segundo apellido del usuario.
- cuentaCorriente (int): Número de cuenta bancaria del usuario (empleado) de la fundación.
- telFijo (int): Número de teléfono fijo del usuario (empleado) de la fundación.
- telMovil (int): Número de teléfono móvil del usuario (empleado) de la fundación.

Relaciones:

Clase: Pago Actividad (clase de asociación entre Alumno y Actividad)

Atributos:

- idPagoActividad (int): Número identificador del pago de la actividad.
- recibo (string): Ruta relativa o completa del archivo que contiene la imagen del recibo/pago/domiciliación del pago de la actividad.

Relaciones:

- Asociación con la clase Cuota precio (composición).

Clase: Actividad

Atributos:

- descripcion (string): Descripción breve de la actividad para consultas por el sistema.
- fechaFin (date): Fecha fin de la actividad para la temporada.
- fechaInicio (date): Fecha inicio de la actividad para la temporada.
- idActividad (int): Número identificador de la actividad.

Relaciones:

- Asociación con la clase Alumno.
- Asociación con la clase Temporada.

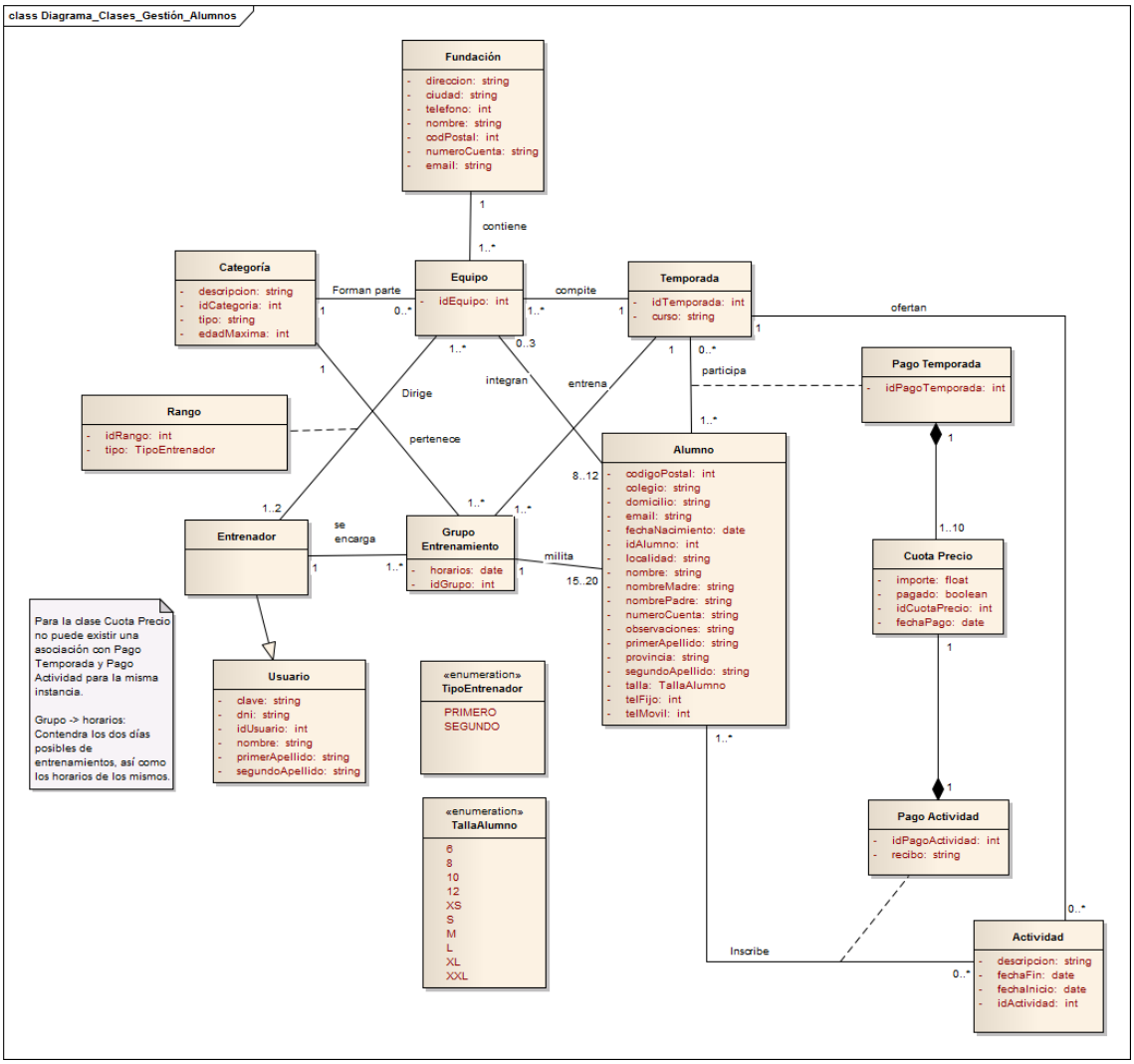
Modelado estático – Diagrama de clases

El modelo estático proporciona mecanismos para describir y representar las interrelaciones estructurales que se establecen entre las clases conceptuales de un modo visual y compacto.

- Proporciona una estructura estática de las clases conceptuales.
- Al ser un modelo de análisis sólo muestra las clases conceptuales extraídas del dominio de aplicación (dominio del problema) y sus relaciones, no componentes software pertenecientes a la solución.

- La reducida barrera que propugna la orientación a objetos entre el problema y la solución, posibilita que el modelo estático inspire la construcción del modelo de diseño.
- Definen un vocabulario común entre clientes y desarrolladores.
- Una vez que el modelo es estable, la descripción de cada clase será tan detallada como sea posible.
- Elimina la vaguedad en la definición de las clases del dominio del problema.

A continuación se describe el diagrama de clases obtenido para el problema especificado.



Modelado del comportamiento externo – Contratos

Describen el comportamiento detallado del sistema en función de los cambios de estado de los objetos cuando se invoca una operación del sistema.

Es un documento que describe lo que una operación se propone lograr, sin decir cómo se conseguirá.

- Define la especificación de una operación sin entrar en su implementación.
- Suele redactarse con un estilo declarativo.

A continuación se describen los contratos de las operaciones de tres de los casos de uso: Dar de alta un usuario, Modificar un alumno e Introducir Pago.

Contratos de Caso de uso: Dar de alta un usuario.

Nombre	CrearNuevoUsuario (DNI:string, Nombre:string, Apellidos:string, Clave:string)
Responsabilidades	Introducir un usuario en el sistema
Tipo	Sistema
Referencias cruzadas	Caso de uso “Dar de Alta Usuario”
Notas	-
Excepciones	-
Salida	-
Precondición	No existe ninguna entrada en el sistema del nuevo usuario.
Postcondición	Se crea un nuevo usuario (creación de instancia).

Nombre	comprobarUsuario (DNI:string)
Responsabilidades	Comprueba si existe o no en el sistema una entrada para el DNI pasado como parámetro a la función.
Tipo	Sistema
Referencias cruzadas	Caso de uso “Dar de Alta Usuario”
Notas	Se hace una consulta al sistema con la información del DNI del usuario (consulta sobre la base de datos). Si se encuentra una coincidencia con el DNI pasado como parámetro se devolverá true o false a la variable validar (comprobar validez del dato).
Excepciones	-
Salida	-
Precondición	Se ha creado una nueva instancia de la clase Usuario.
Postcondición	-

Nombre	generarError()
Responsabilidades	Genera un mensaje de error si el usuario comprobado no existe.
Tipo	Sistema
Referencias cruzadas	Caso de uso “Dar de Alta usuario”
Notas	La operación CreaUsuario(…) crea una instancia temporal, la cual se comprueba si ya ha sido creada anteriormente. Si ha sido creada, se manda un mensaje de error y se elimina esa instancia temporal.
Excepciones	-
Salida	-
Precondición	El usuario debe estar registrado
Postcondición	-

Nombre	ActualizarUsuario (DNI:string, nombre:string, apellidos:string, clave:string)
Responsabilidades	Actualizar la información referente a un usuario
Tipo	Sistema
Referencias cruzadas	Caso de uso “Dar de Alta Usuario”
Notas	-
Excepciones	-
Salida	-
Precondición	El usuario no debe estar registrado.
Postcondición	Guarda la información referente a un usuario.

Contratos de Caso de uso: Consultar Alumno.

Nombre	consultarAlumno()
Responsabilidades	Consultar los datos de un alumno existente en el sistema y opcionalmente poder modificarlos.
Tipo	Sistema.
Referencias cruzadas	Caso de Uso “Consultar alumno”
Notas	El sistema muestra las dos formas posibles de elegir un alumno: por su ID o mediante una lista de alumnos.
Excepciones	-
Salida	-
Precondición	Que haya al menos un alumno dado de alta en el sistema.
Postcondición	-

Nombre	modificarDatos(datos)
Responsabilidades	Modificar los datos referentes a un alumno.
Tipo	Sistema.
Referencias cruzadas	Caso de Uso “Consultar alumno”.
Notas	datos={codigoPostal, colegio, domicilio, e-mail, fechaNacimiento,localidad, nombrePadre, nombreMadre, numeroCuenta, observaciones, primerApellido, provincia, segundoApellido, talla, telFijo, telMovil}
Excepciones	-
Salida	-
Precondición	Que el alumno exista en el sistema.
Postcondición	-

Nombre	guardarCambios()
Responsabilidades	Guardar los cambios que se hayan producido en los datos del alumno.
Tipo	Sistema.
Referencias cruzadas	Caso de Uso “Consultar alumno”.
Notas	-
Excepciones	-
Salida	mensajeOK.
Precondición	Que todas las operaciones para modificar los datos del alumno se hayan efectuado correctamente.
Postcondición	Se modificaron los datos de un objeto “Alumno” que ya existía en el sistema.

Nombre	generarError()
Responsabilidades	Generar un mensaje de error debido a que los datos introducidos no sean correctos.
Tipo	Sistema.
Referencias cruzadas	Caso de Uso “Consultar alumno”.
Notas	-
Excepciones	-
Salida	mensajeError.
Precondición	-
Postcondición	-

Descomposición del sistema en subsistemas de diseño para obtener la arquitectura del sistema.

Establecer la arquitectura del sistema.

Para llevar a cabo la fase de diseño hay que establecer la arquitectura del sistema teniendo en cuenta los siguientes puntos:

- Objetivos de diseño del sistema.
- Estilos arquitectónicos posibles.
- Documentación de las etapas anteriores del proceso de desarrollo.

Subsistemas funcionales.

En el modelado de requisitos se han detectado los siguientes subsistemas funcionales:

- Subsistema GestionAlumnos.
- Subsistema GestionUsuarios.

Requerimientos no funcionales.

Existen algunas restricciones y requisitos no funcionales que ya se detallaron en la etapa de análisis:

- No requiere un conocimiento específico para la utilización del software.
- La aplicación tendrá un manual de uso.
- La base de datos estará implementada en un lenguaje objeto relacional como MySQL.
- La aplicación estará realizada en el lenguaje de programación Java.
- Se creara un script de tarea programada (cron) para la copia de seguridad de la base de datos debido a alguna inconsistencia del sistema.
- La interfaz debe reflejar claramente la distinción entre las distintas partes del sistema.
- La documentación del código fuente será llevada a cabo mediante la aplicación javadoc.

- El sistema se desplegara sobre una distribución Microsoft Windows.
- El administrador se encargara de tareas de mantenimiento.
- Cuando el sistema detecte una anomalía mostrara al usuario el mensaje de error pertinente y abortara la ejecución del proceso.
- El código fuente de la aplicación seguirá un estilo uniforme y normalizado para todos los módulos del mismo.
- El formato de las fechas será dd/mm/yy.
- El control de asistencia no es controlado, pero se puede incluir en la sección de observaciones de los alumnos.

Objetivos de diseño.

A partir de los requerimientos no funcionales se obtienen los siguientes objetivos de diseño:

- Facilidad de uso. Utilizando un entorno gráfico intuitivo, basado en ventanas, adaptable a los posibles tipos de usuarios y con un manual de usuario.
- El sistema debe facilitar la incorporación de la tecnología MySQL para el manejo de la persistencia.
- Seguridad. Debe haber mecanismos de control de acceso para todos los usuarios del sistema.
- Fiabilidad. Se espera que el sistema responda de forma satisfactoria.

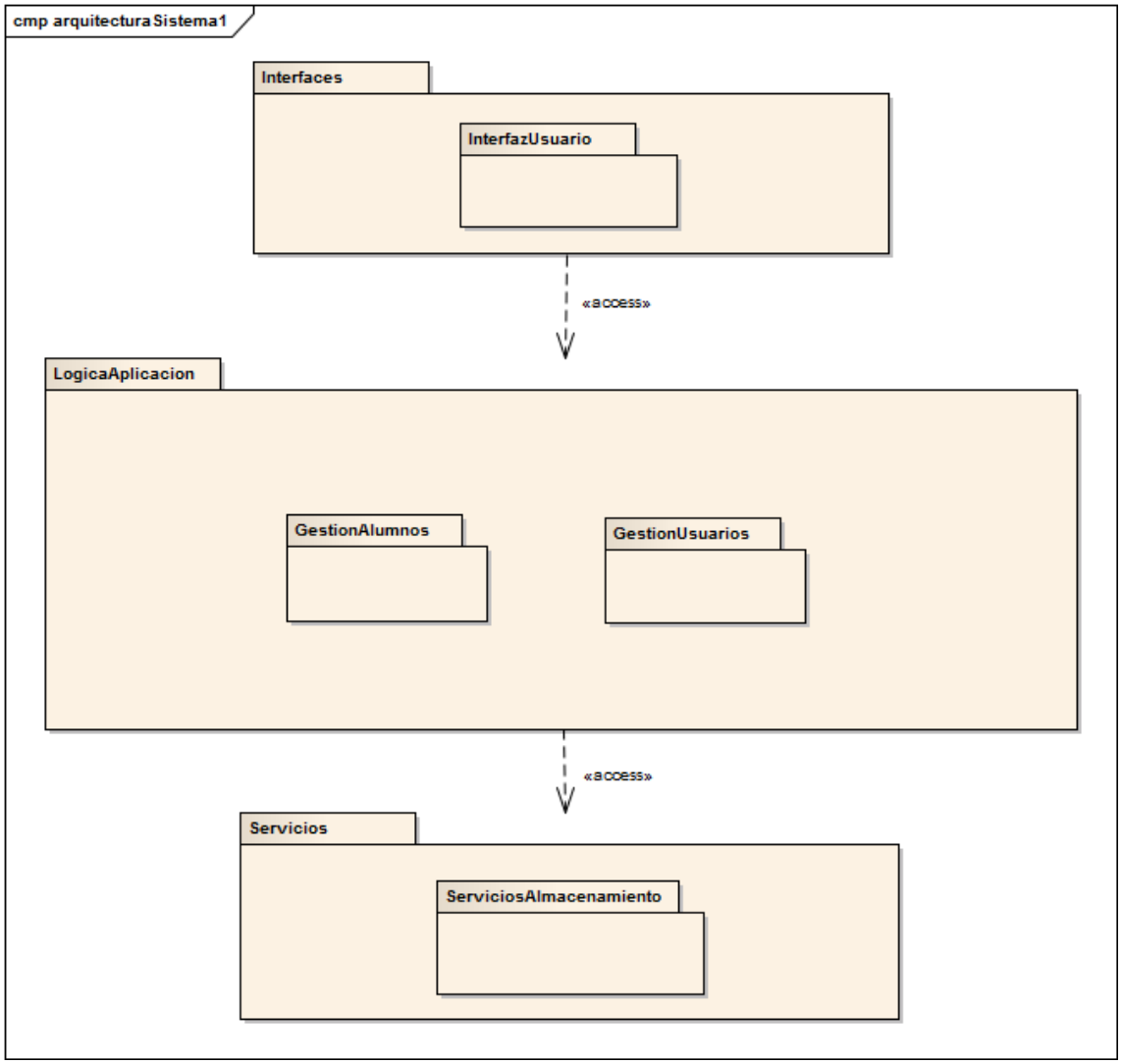
Determinación de la arquitectura.

Después de analizar los objetivos de diseño y los estilos arquitectónicos para el sistema se decide:

Utilizar una arquitectura software basada en capas cerradas que separe la aplicación en tres niveles ya que esto mejora la estructura del sistema y fomenta la flexibilidad en cuanto a sustitución de servicios o interfaz de usuario.

- Interfaz de usuario.
- Lógica de aplicación.
- Servicios

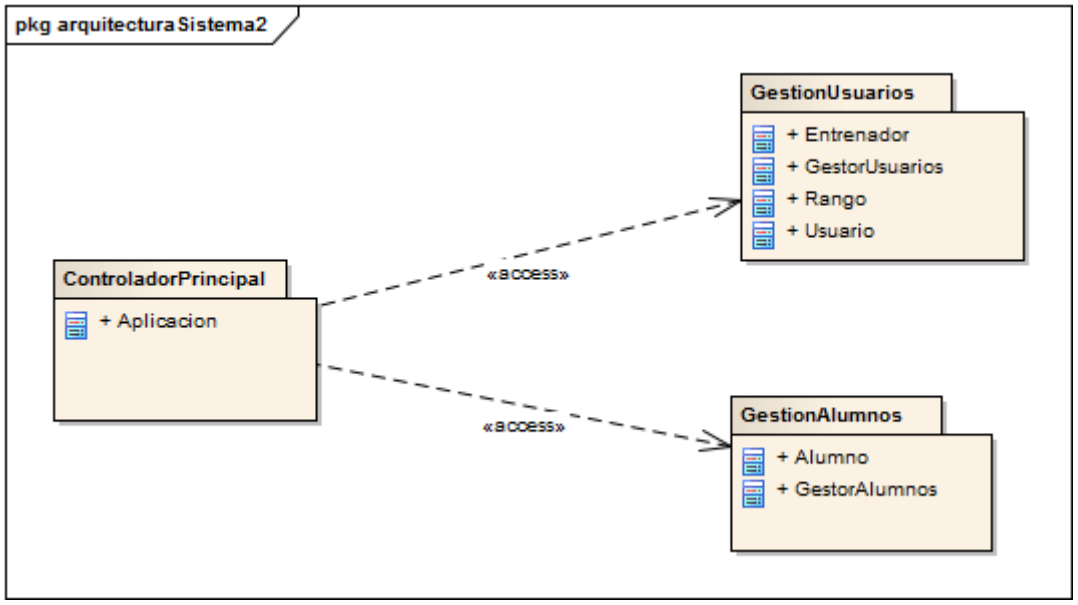
Diagrama arquitectura software de tres capas cerradas



Modelar la arquitectura: Diagramas de paquetes

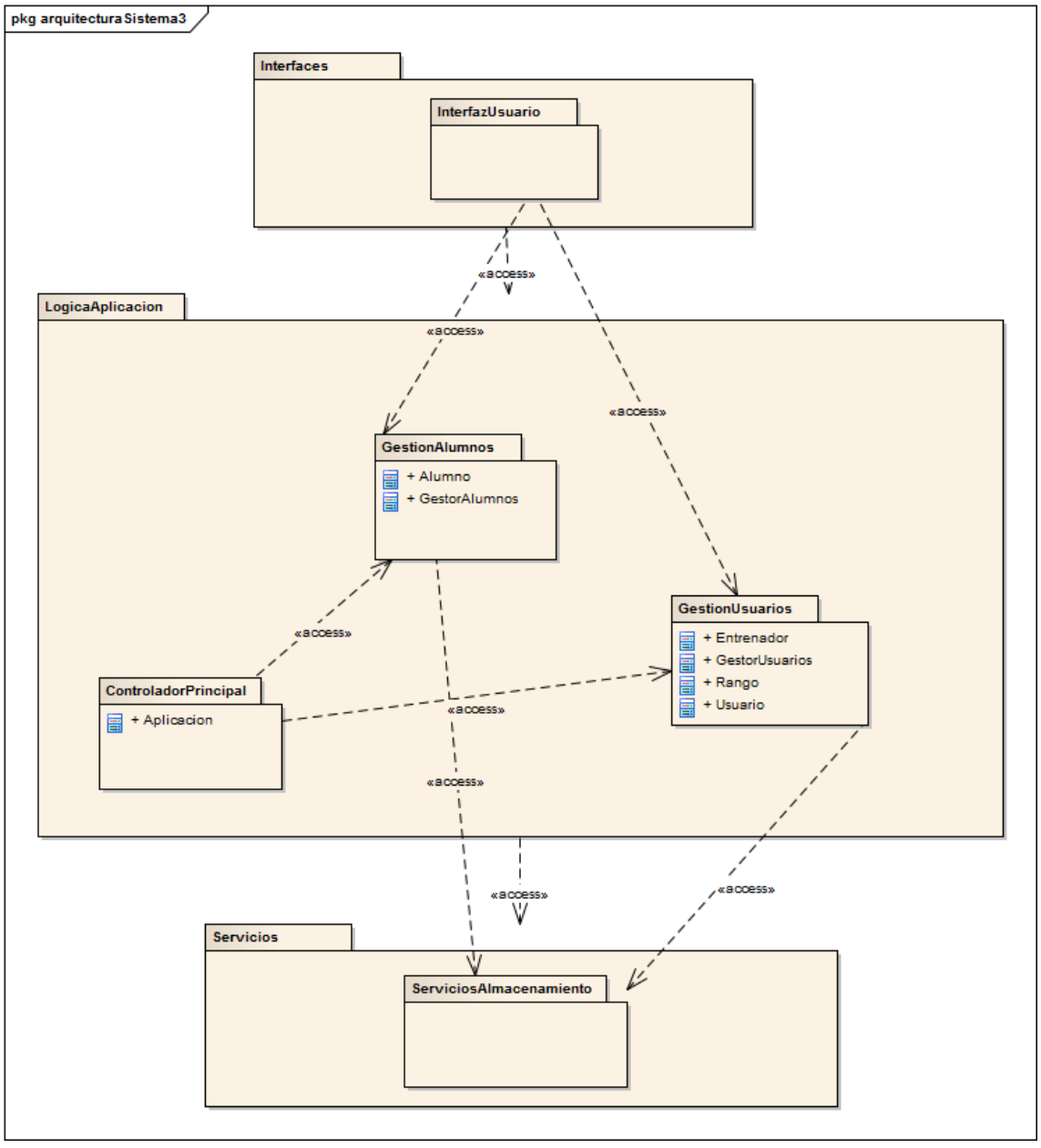
Una descomposición como la expuesta anteriormente nos lleva al diagrama de paquetes siguiente.

Diagrama de paquetes de la lógica de aplicación



A partir de la estructuración refinada de los subsistemas de la capa de lógica de aplicación, se ha diseñado una nueva arquitectura software del sistema.

Diagrama de nueva arquitectura software del sistema



Obtener Diagrama de Despliegue de Diseño.

Para obtener el diagrama de despliegue de diseño se determinan los posibles nodos del sistema, con los cuales se especifica el hardware físico sobre el que se ejecuta el sistema de software y cómo se ubica el software en el hardware. Además se especifican los componentes y las conexiones entre los distintos nodos.

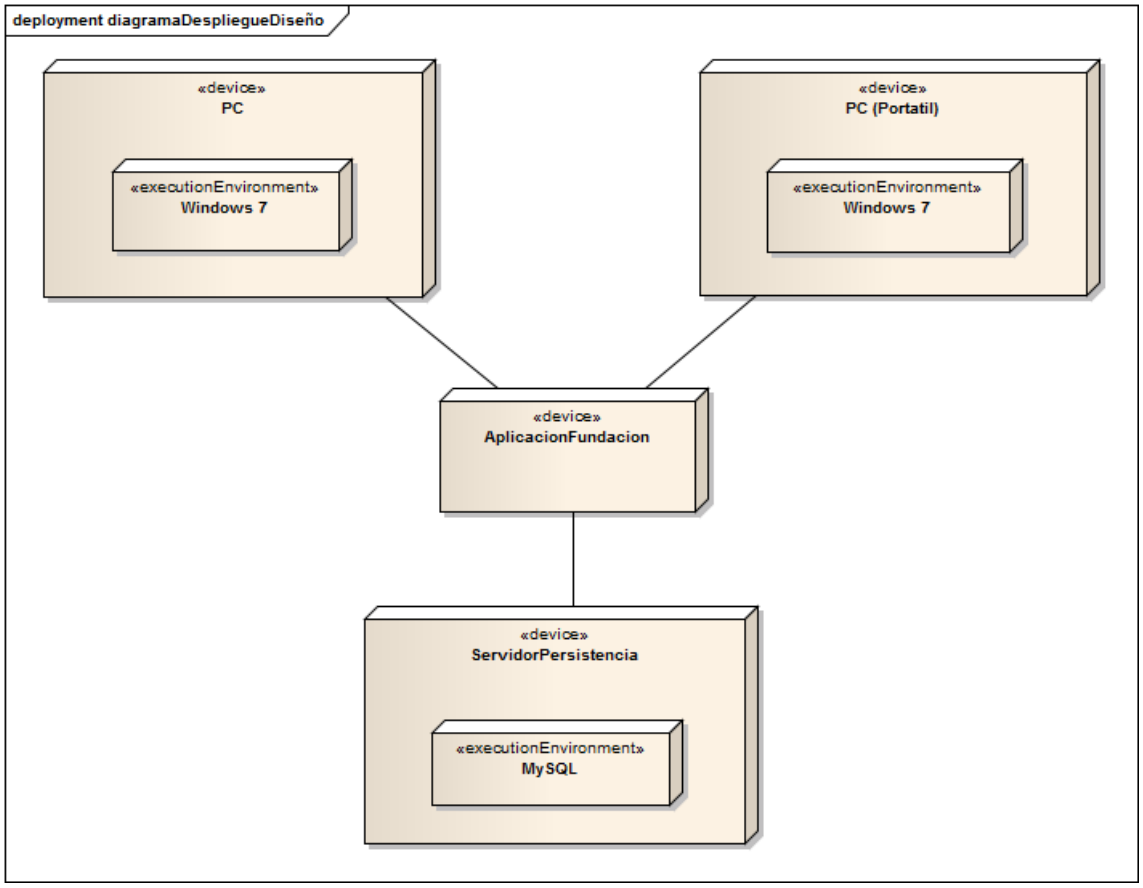
Se tienen los siguientes nodos, los cuales pueden ser dispositivos físicos o entornos de ejecución:

- PC, que es el ordenador que hay en la oficina.
- PC (Portatil), que es un portátil que pueden usar los entrenadores durante los partidos.
- Aplicación Fundación, que es la aplicación sobre la que se trabaja.
- Servidor de Persistencia, que es la tecnología que se usará para almacenamiento de datos.

Dentro de los nodos anteriores se anidan los siguientes nodos con el prototipo <<Execution Environment >> ,lo cual representa un tipo de entorno de ejecución para software.

- Execution Environment Windows 7.
- Execution Environment MySQL.

Diagrama de despliegue de diseño



Modelar Diagrama de componentes.

El diagrama de paquetes está estructurado en agrupaciones lógicas, pero es necesario definir subsistemas, es decir, cerrar las partes reutilizables del sistema en componentes. Para ello es necesario añadir interfaces.

Con el diagrama de componentes se va a definir la implementación del sistema a partir de los módulos de software y su interrelación.

Con cada componente se va a representar una parte modular de un sistema que encapsula sus contenidos y cuya manifestación es reemplazable.

Cada componente puede:

- Tener atributos y operaciones.
- Representar a cualquier cosa desde clases sencillas a aplicaciones, subsistemas y sistemas.
- Tener interfaces proporcionadas, requeridas y puertos, es decir, servicios que ofrece un componente a otro o que necesita de otro.

Las interfaces permiten a cada componente comunicarse con otros componentes. Estas son una colección de operaciones que se utilizan para especificar un servicio de un componente.

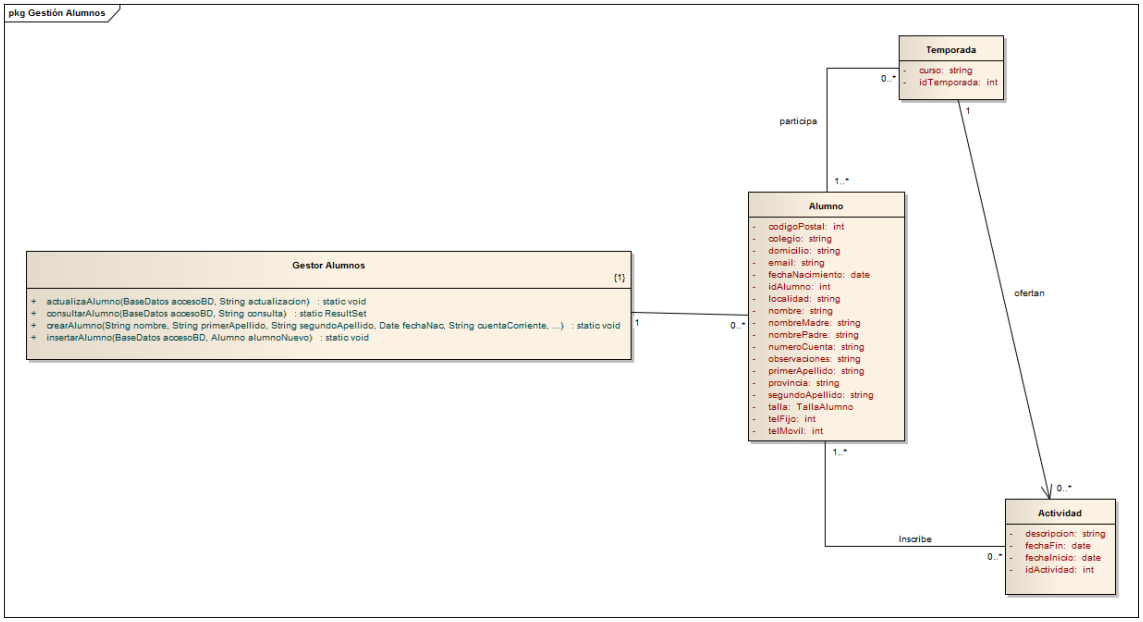
Las interfaces facilitan la sustitución y reutilización de componentes:

- Un componente puede exportar su interfaz e importar interfaces de más de un componente.
- Un componente que utiliza una interfaz determinada funcionará adecuadamente independientemente del componente que realice la interfaz.
- El componente que realiza la interfaz es siempre sustituible por un componente o conjunto de componentes que implementen dicha interfaz.
- Un componente puede utilizarse en un contexto determinado si y solo si todas sus interfaces de importación son suministradas por otros componentes.

Un puerto de componente agrupa un conjunto semánticamente cohesivo de interfaces proporcionadas y requeridas.

Aclarado todo sobre los diagramas de componentes se muestran los siguientes diagramas obtenidos.

Subsistema Gestión de alumnos



Subsistema Gestión de usuarios

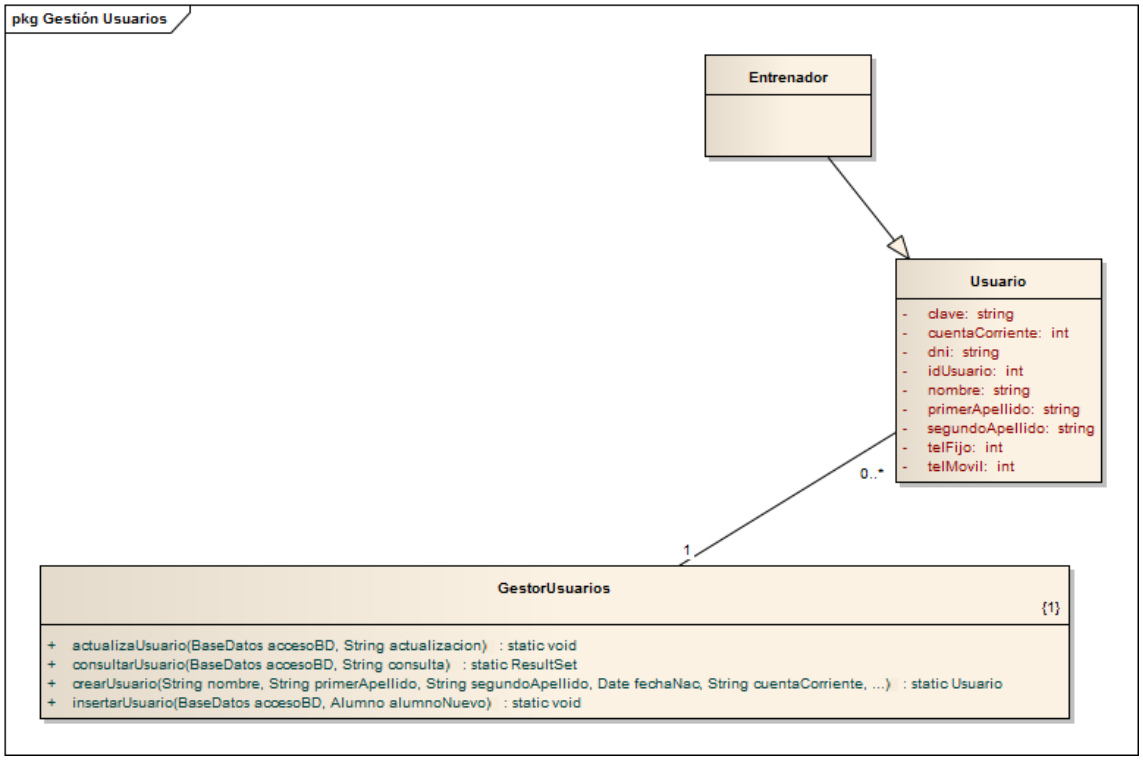
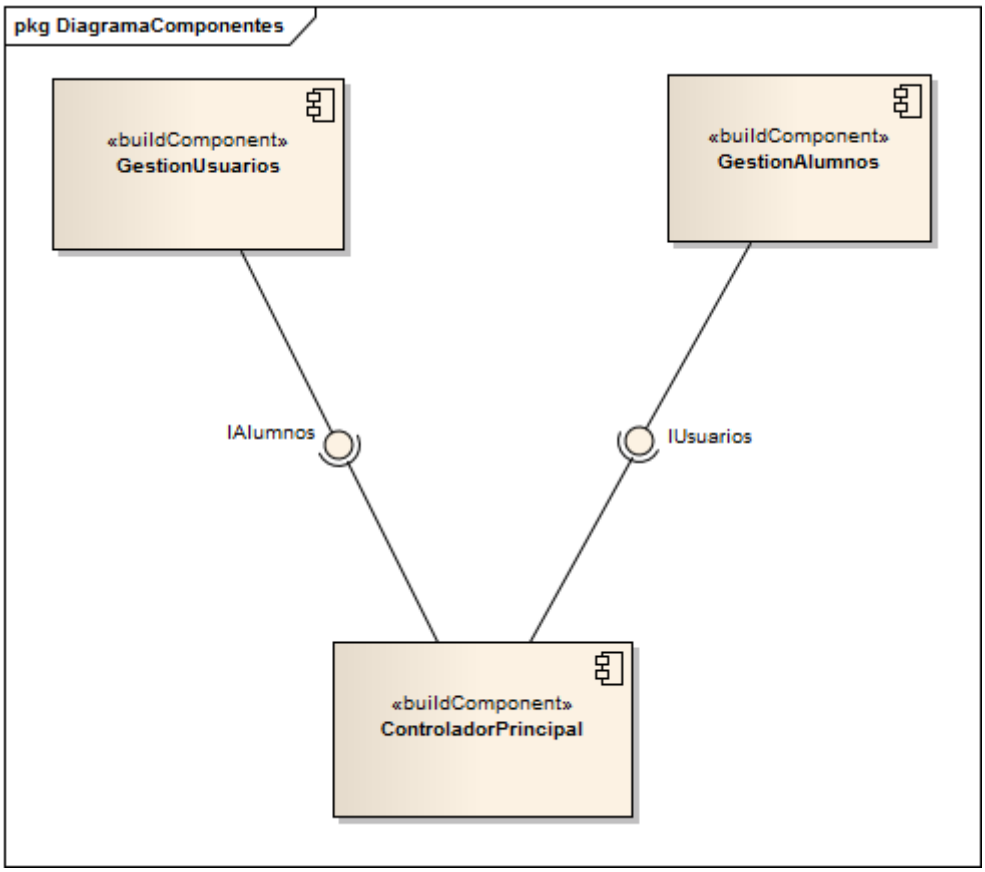
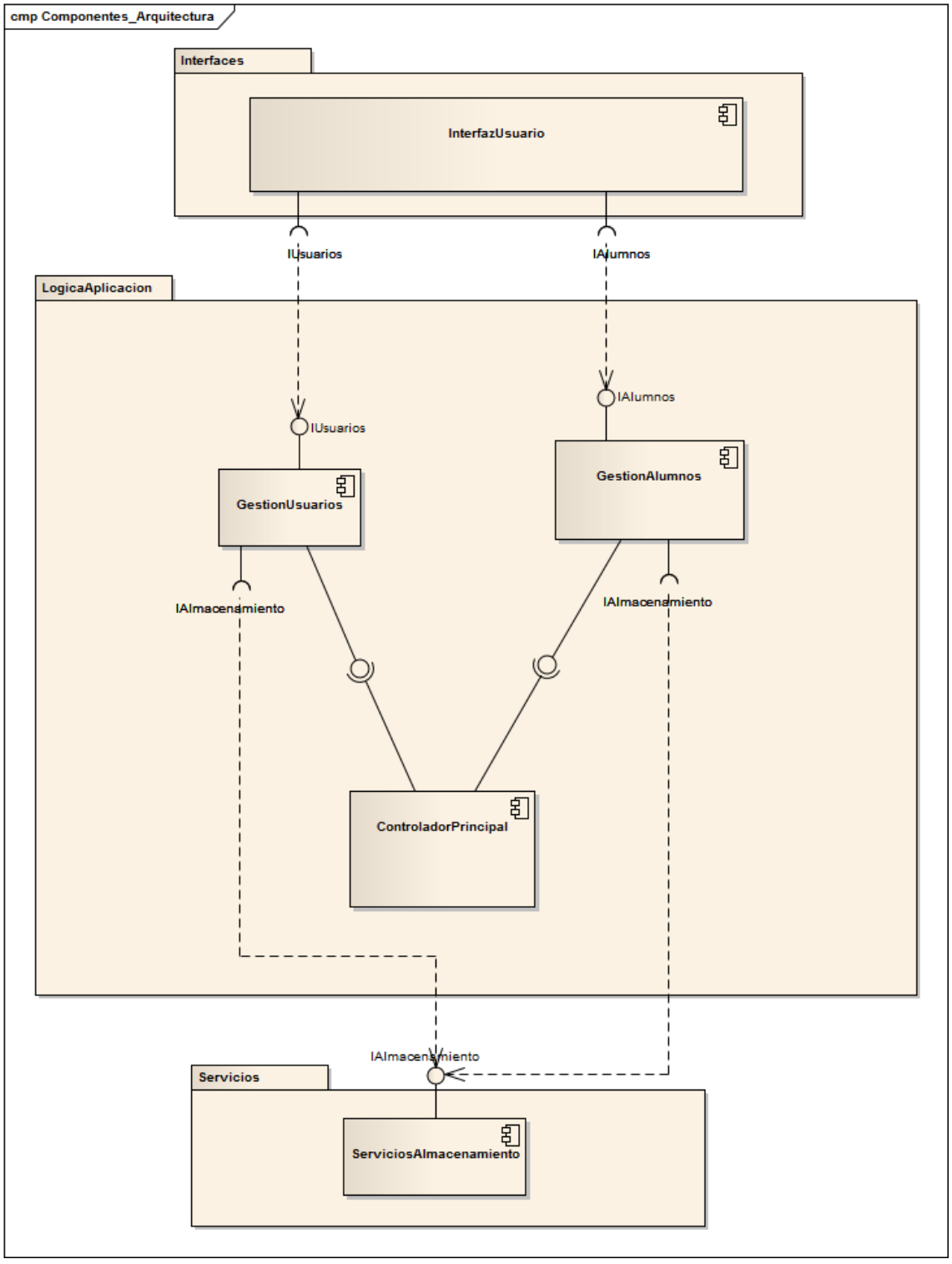


Diagrama de componentes



Componentes arquitectura



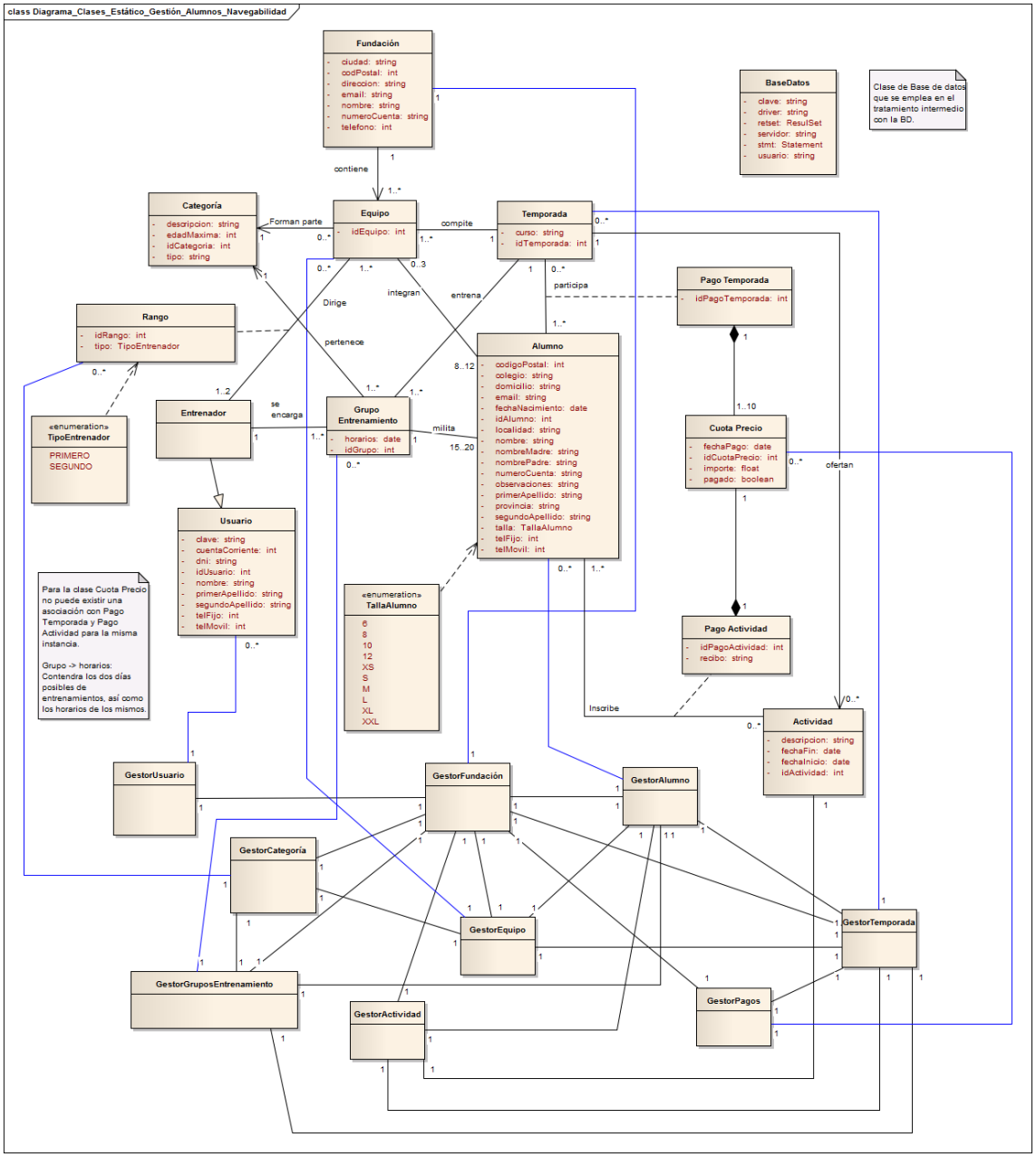
Encajar el Diagrama de Clases (obtenido anteriormente) en la arquitectura obtenida en el apartado anterior

Diagrama de análisis

Al terminar la etapa de diseño, se ha refinado suficientemente el diagrama de clases y las relaciones entre estas. También se conocen mejor los mensajes que se intercambian los objetos para realizar las tareas necesarias.

El modelo estático de análisis puede utilizarse para definir los paquetes de la capa de lógica de aplicación.

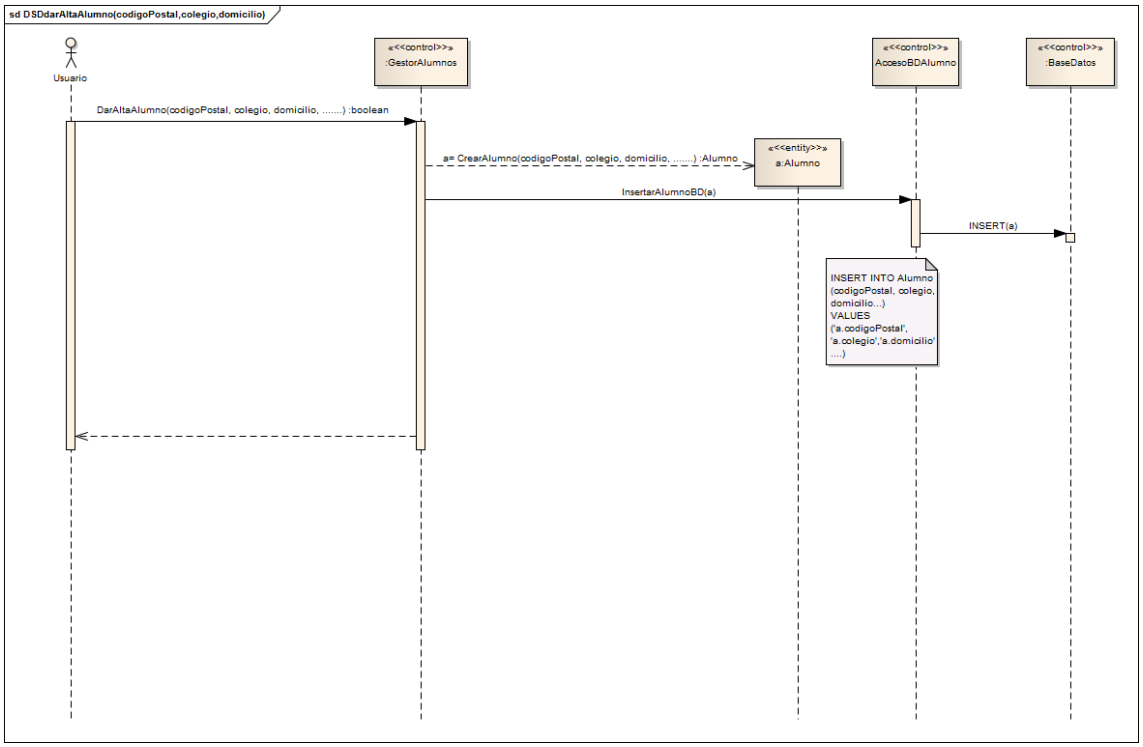
Diagrama de clases estático.



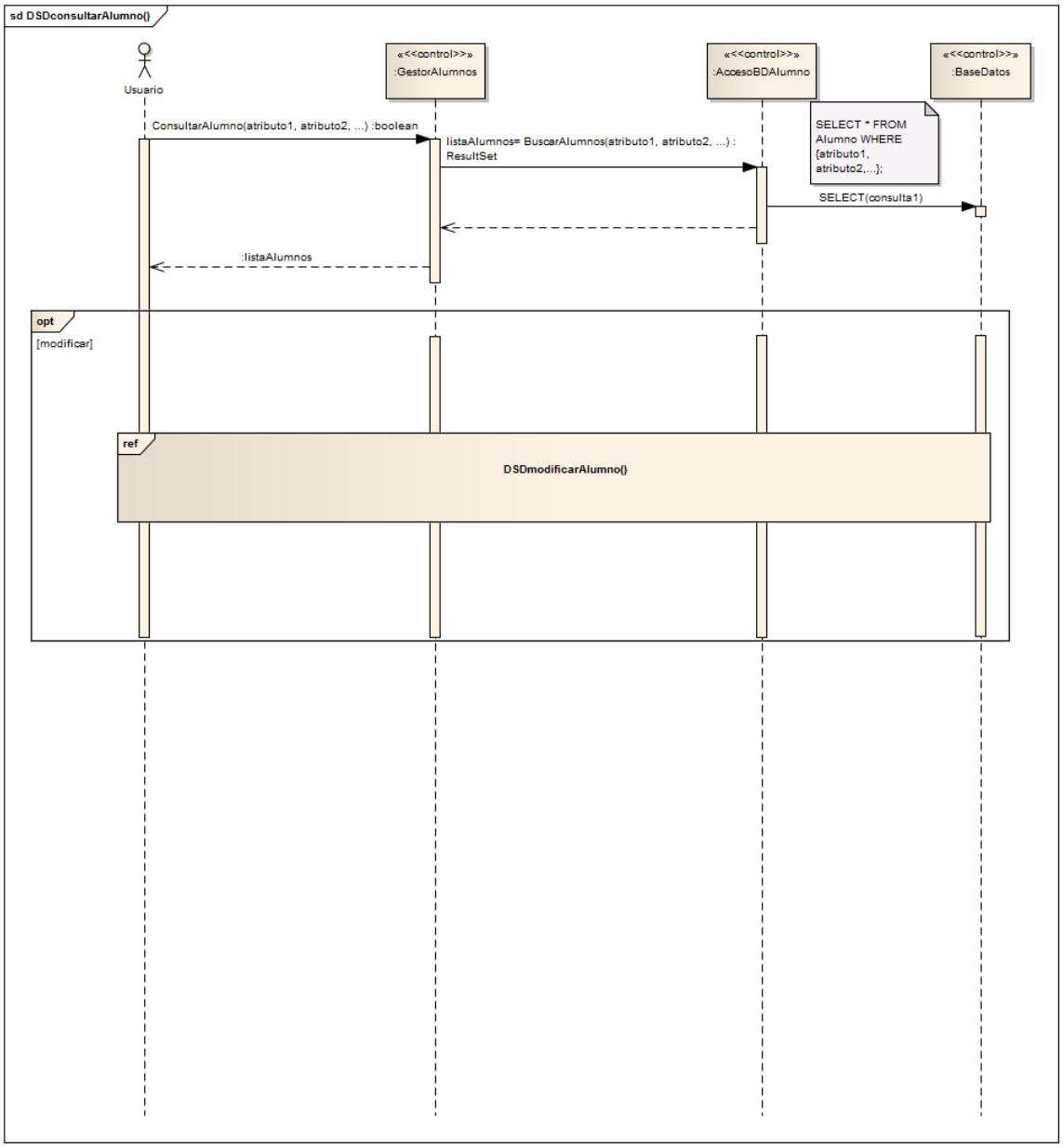
Diagramas de Secuencia del Diseño.

En esta etapa, las clases tienen ya definidas las operaciones. Además en estos diagramas se incluyen mensajes con las consultas a los objetos de control de la BD y se muestran las entidades creadas. A continuación se presentan algunos Diagramas de Secuencia de Diseño.

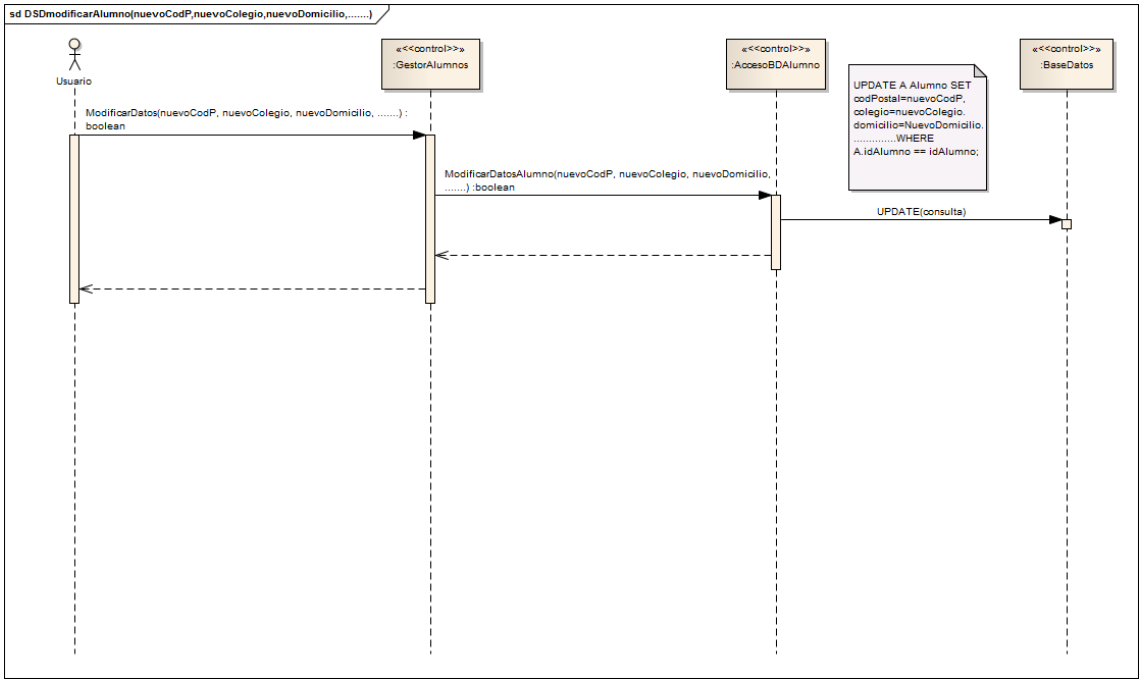
DSD Dar Alta Alumno.



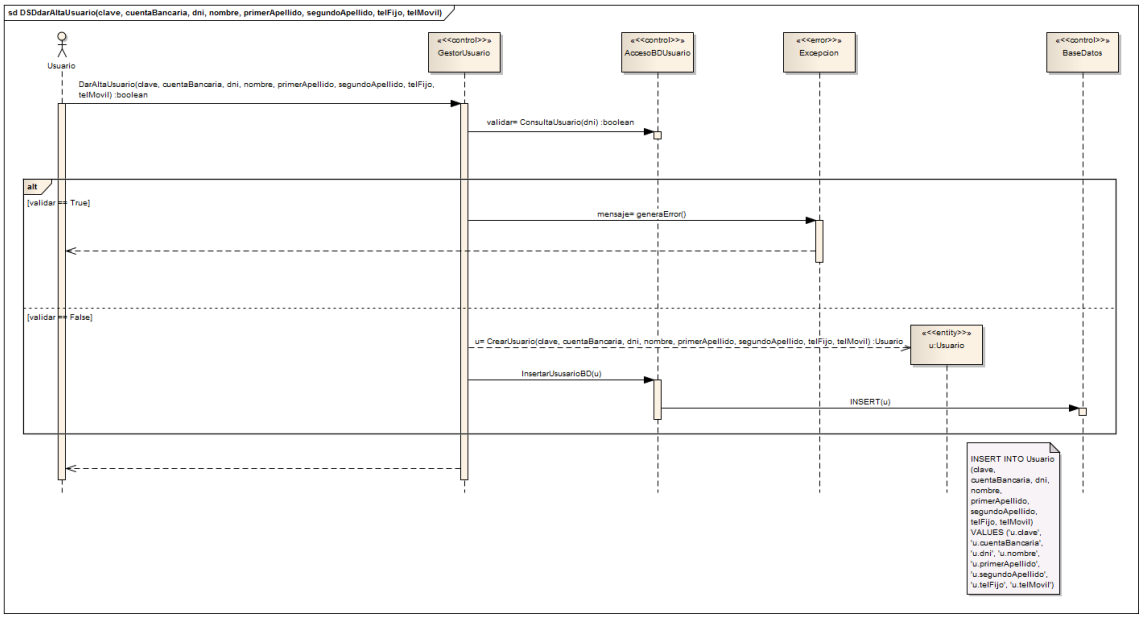
DSD Consultar Alumno



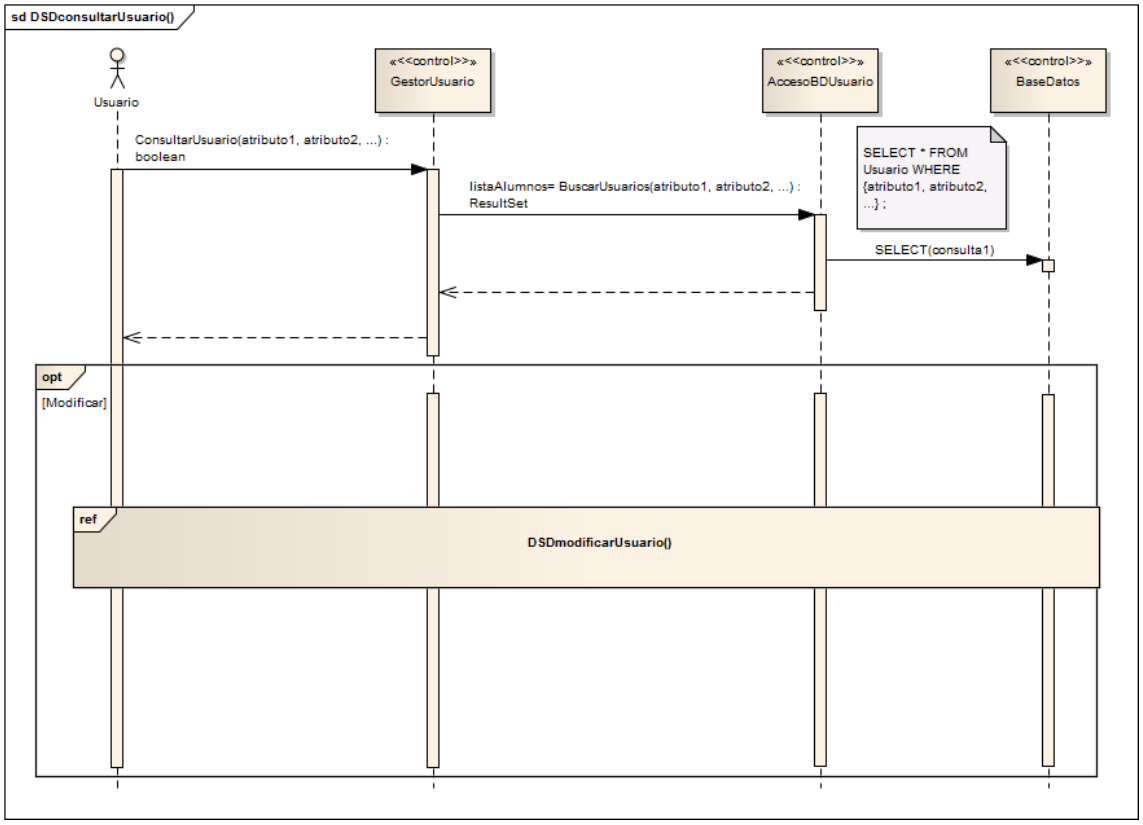
DSD Modificar Alumno



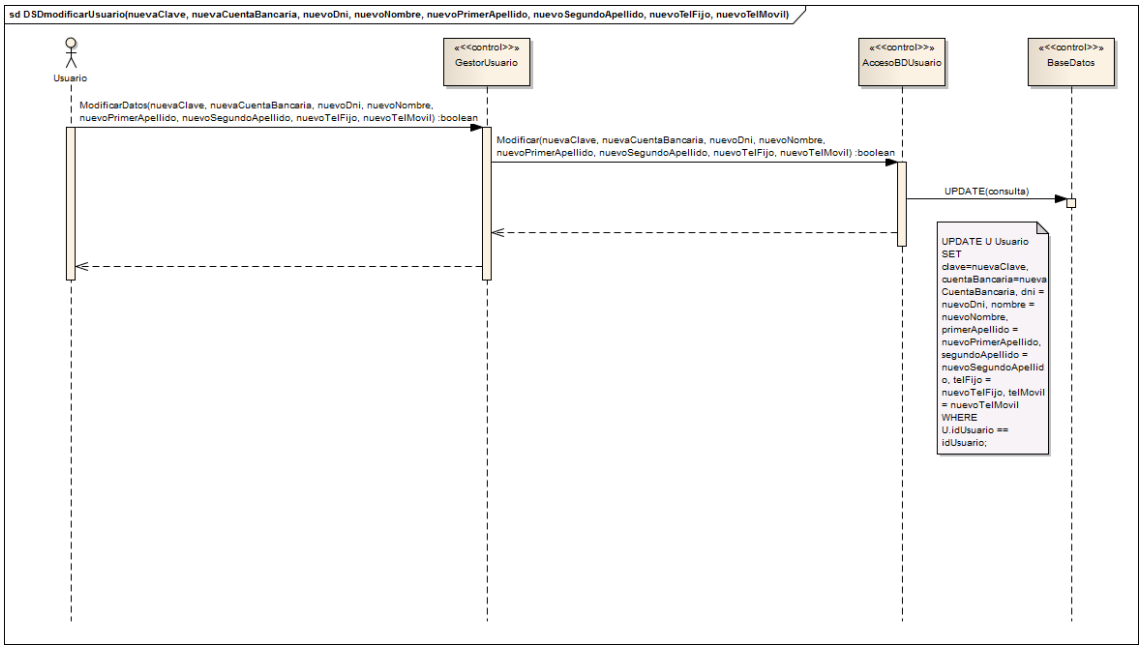
DSD Dar Alta Usuario



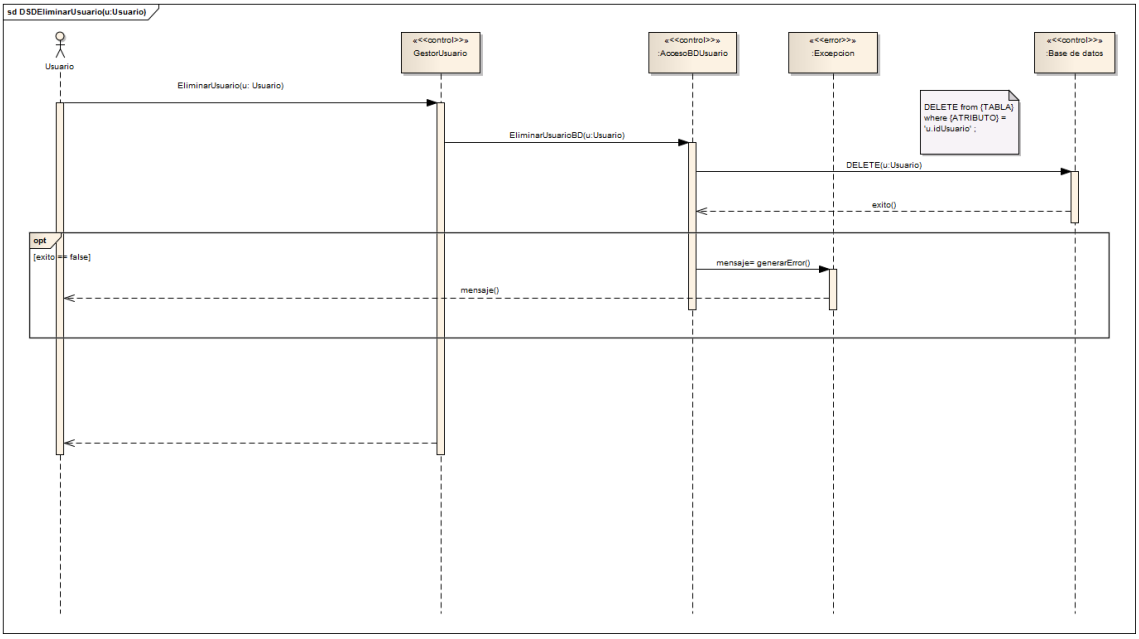
DSD Consultar Usuario



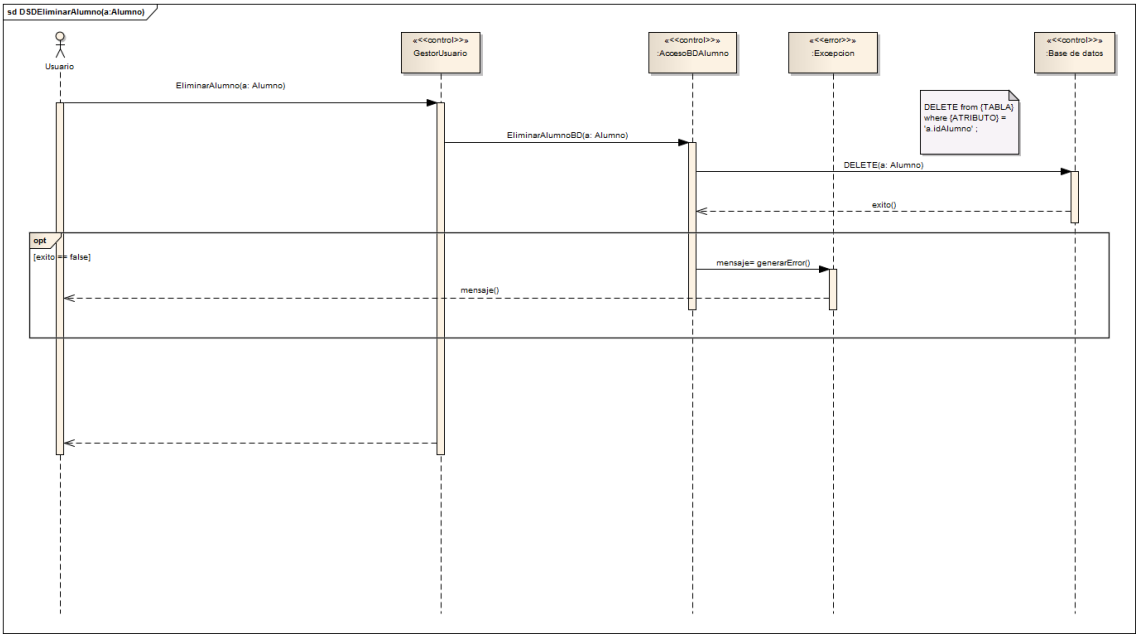
DSD Modificar Usuario



DSD Eliminar Usuario



DSD Eliminar Alumno



Anexo control de versiones

Fecha: 03/04/2013 **Versión:** 1.1

- Entrega subequipo de diseño: Corrección de errores de la iteración 1