Universidad de las Ciencias Informáticas

Facultad 6

Título: Sistema para la gestión de la información de profesores y estudiantes de la facultad 6: Desarrollo del Módulo Extensión

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Rodaisy Abella Pérez

Lorena Sánchez Rodríguez

Tutor: Ing. Yaikiel Hernández Días

Ciudad de La Habana, junio, 2009

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Rodaisy Abella Pérez

_____

Firma de la autora

Lorena Sánchez Rodríguez

_____

Firma de la autora

Ing. Yaikiel Hernández Días

_____

Firma del tutor

DATOS DE CONTACTO

Tutor:

Ing. Yaikiel Hernández Días

Universidad de las Ciencias Informáticas, Habana, Cuba.

Email: yhernandezd@uci.cu

AGRADECIMIENTOS

Agradecemos a la Universidad de las Ciencias Informáticas, a la Revolución y a Fidel, por darnos la oportunidad de formarnos y hacer de nuestros sueños una realidad.

A nuestros padres por ser guías en nuestras vidas y estar siempre allí para nosotras y por ser nuestra principal razón de ser.

A nuestros amigos por todos estos momentos lindos y tristes que pasamos en la Universidad: Yanet, Neyaris, Virgen, Evelyn, Angélica, Lino, gracias a todos por brindarnos su apoyo y cariño.

DEDICATORIA

A mis padres por confiar en mí y por haberme dado la educación con la que cuento hoy en día.

A mi hermano por ser mi ejemplo a seguir.

A mis abuelas por quererme tanto.

A mis tíos por su preocupación por mí en todos los momentos difíciles.

A mi amiguita y compañera de tesis Lorena por haber confiado en mí cuando aún no había aprobado la prueba de programación y pensó en mí para la tesis.

A una personita que me forjó en mis primeros años aquí en la UCI y que siempre vivirá en mi corazón…

Rodaisy

RESUMEN

El presente trabajo forma parte del proyecto de informatización de la facultad 6 de la Universidad de las Ciencias Informáticas, titulado: Sistema para la gestión de la información de profesores y estudiantes de la facultad 6: Módulo Extensión, el mismo aborda el análisis, diseño e implementación de funcionalidades para el control y seguimiento de las actividades extensionistas de la facultad, ya sean los festivales, los juegos deportivos y otras actividades que se realicen, proporcionándole así al vicedecano de extensión una forma más rápida y eficiente de guardar dicha información. Se analizó el funcionamiento de las actividades extensionistas para el desarrollo de este sistema y posteriormente se exponen las características del sistema.

PALABRAS CLAVES

Actividades extensionistas, festivales, gestión de la información, juegos deportivos.

TABLA DE CONTENIDOS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La ambigua y escasamente precisa denominación «Extensión Universitaria» goza en la actualidad de una extraordinaria fuerza gracias a la difusión que viene alcanzando en el ámbito universitario de todo el mundo. Afortunadamente, con independencia de la antigüedad e importancia de las diferentes universidades, cada una de ellas le presta una decidida atención, el desarrollo cultural como base de la extensión universitaria, que es el concepto de ese antiguo vocablo de Extensión Universitaria, con el que originariamente se designó a esta importante faceta de la misión universitaria.

La Vicerrectoría de Extensión Universitaria en coordinación con las facultades docentes y las organizaciones estudiantiles se encarga de organizar la actividad cultural de la universidad, el objetivo primordial está dirigido a la formación de una cultura general e integral de la comunidad en interacción con la sociedad, para ello lleva a cabo una serie de actividades culturales y deportivas, mediante proyectos de desarrollo local y presentaciones artísticas, el trabajo cultural trasciende el ámbito institucional para llegar a la comunidad. Al mismo tiempo la Dirección de Extensión Universitaria coordina diferentes actividades recreativas e instructivas: visitas a los museos más importantes de la capital, a puestas en escena de clásicos del teatro y el ballet, a ferias y bienales de arte, entre otras. El esfuerzo mayor del área cultural, compuesta por cinco departamentos: Creación Artística y Literaria, Realización, Promoción, Programación y Relaciones Públicas y una Librería, donde se vinculan los métodos tradicionales del ciclo reproductivo del arte con las facilidades que ofrecen las nuevas tecnologías de la información y las comunicaciones en la búsqueda de un hombre cada vez más comprometido con su época.

Otra de las áreas que atiende la extensión universitaria es la relacionada con la práctica de deportes que es una de las formas de esparcimiento más frecuentes dentro de la Universidad de las Ciencias Informáticas (UCI). Sus estudiantes profesores y trabajadores mantienen una vida más saludable, a partir del ejercicio físico, también la residencia ocupa un lugar importante dentro de la extensión universitaria, sus edificios y apartamentos acogen a la totalidad de los estudiantes y una gran parte de sus profesores. Siguiendo las líneas de trabajo del centro y sus principales misiones, la residencia trabaja por convertirse en un espacio esencialmente educativo, abierto a la participación y el diálogo a través de las actividades culturales, recreativas y deportivas, orientado a elevar en los estudiantes la disciplina y el compromiso con la Patria. La labor extensionista también propicia el goce espiritual de los estudiantes y trabajadores que conviven en la ciudad universitaria al promover la participación de prestigiosas agrupaciones musicales de diversos formatos, la actuación de destacados grupos

teatrales y danzarios, y la presencia de escritores e intelectuales de renombre, así como importantes figuras del deporte nacional, en la facultad no existe un sistema que sea capaz de recopilar la información de profesores y estudiantes referente a la participación en los festivales de artistas aficionados, en los juegos deportivos y otras actividades extensionista, actualmente no se logra un seguimiento y control de los artistas aficionados, deportistas, así como las diversas actividades realizadas en la beca, además la información no se almacena de manera adecuada para su posterior uso, es por ello que surge el proyecto de Informatización de la facultad, donde el módulo de extensión desarrolla este sistema para la gestión de la información de profesores y estudiantes, quedando así registrada esta información.

Por todos estos argumentos se define como **Problema Científico:**
¿Cómo gestionar los procesos de extensión universitaria de la facultad 6?

Para dar solución a este problema surge como **Objeto de Estudio:**
Proceso de gestión de la información de extensión universitaria.

Se define como **Campo de Acción:**
Proceso de gestión de la información de extensión universitaria en la facultad 6.

Se persigue como **Objetivo General:**
Desarrollar el módulo de extensión para la gestión de la información extensionista en la facultad 6.

Para cumplir este objetivo general se trazaron los siguientes **Objetivos Específicos:**

- ✓ Modelar el entorno de negocio a través del modelo de dominio.
- ✓ Identificar las funcionalidades de la aplicación Web.
- ✓ Realizar análisis y diseño de la aplicación Web.
- ✓ Implementar los componentes de la aplicación Web.

Tareas a desarrollar para asegurar el cumplimiento de los objetivos trazados:

- ✓ Revisión bibliográfica sobre aplicaciones Web vinculadas al objeto de estudio.

- ✓ Estudio de las herramientas, metodologías y tecnologías definidas para desarrollar la aplicación.
- ✓ Identificación de los conceptos principales en el área extensionista.
- ✓ Realización del levantamiento de requisitos.
- ✓ Realización del modelado de dominio.
- ✓ Realización de los diagramas correspondientes al diseño.
- ✓ Identificación de los elementos del diseño de la aplicación Web.
- ✓ Implementación de las funcionalidades de la aplicación Web.

El trabajo de diploma está estructurado de la siguiente manera: introducción, cuatro capítulos, conclusiones, recomendaciones, bibliografía, anexos y glosario.

En el **Capítulo 1 Fundamentación Teórica:** se hace un análisis del estado del arte del objeto de estudio, se investiga acerca de los sistemas informáticos vinculados al campo de acción, se fundamentan las metodologías, tecnologías y herramientas utilizadas para el desarrollo de la aplicación Web.

En el **Capítulo 2 Características del Sistema:** se define el negocio en términos de modelo de dominio y se describe la solución propuesta para la situación problémica. Se presentan las características y funcionalidades del sistema a partir de los requisitos funcionales y no funcionales capturados.

En el **Capítulo 3 Diseño del Sistema:** incluye los diagramas de clases del diseño, los diagramas de secuencia y las realizaciones de cada uno de los casos de usos del diseño. También se muestra el diagrama de despliegue.

En el **Capítulo 4 Implementación del Sistema:** incluye la programación realizada a partir de los requerimientos y los diagramas del diseño elaborados. Se muestra el diagrama de componentes correspondiente a cada uno de los casos de uso implementados, así como los fragmentos relevantes de código.

CAPÍTULO 1: FUNDAMENTACÓN TEÓRICA

Introducción

Este capítulo incluye el estado del arte del proceso de gestión de información de extensión universitaria, se describen las técnicas, tecnologías y metodologías en las que se apoya para la solución del problema, contiene una descripción de los artefactos generados por los roles definidos en el desarrollo de la aplicación, así como los patrones de caso de uso, arquitectura y diseño utilizados.

1.1 Sistema de Gestión de Información extensionista

Un Sistema de Gestión de Información puede ser definido como un conjunto de elementos que interactúan entre sí, con el fin de apoyar las actividades que se realizan en una organización o para automatizar los procesos de trabajo que se efectúan dentro de esta, y un sistema de gestión de información extensionista específicamente, se restringe a automatizar los procesos relacionados con la extensión universitaria. La puesta en marcha de un Sistema de Gestión de Información puede proporcionar numerosos beneficios, tales como la optimización del tiempo que transcurre desde la búsqueda de documentos y la reducción en gran medida de los riesgos de pérdida del documento físico original, además se puede disponer de la información de forma centralizada y rápidamente accesible.

A nivel internacional en la actualidad se puede mencionar que en España todas las Universidades están prestando desde los Vicerrectorados de Extensión Universitaria una especial atención a esta actividad. Con este fin han surgido las Universidades Internacionales (la Menéndez Pelayo y la de Andalucía) cuya programación de actividades científico-culturales pretende sobrepasar los límites nacionales internacionalizando la ciencia y la cultura en busca de la globalización del saber al servicio de la humanidad. La universidad de Verano de Lanzarote pone a disposición de sus estudiantes un sitio en el que registran todo el programa de extensión universitaria con el objetivo de promover, incentivar y acercar la vida académica universitaria entre los ciudadanos canarios y favorecer el fomento de la cultura y del conocimiento humanístico, social, científico, de sus estudiantes. En paralelo a las Universidades Internacionales, la totalidad de las Universidades españolas las clásicas y las de creación más reciente programan cursos a través de sus vicerrectorados de extensión con el deseo de cubrir objetivos en muy diversas direcciones. No existe una superestructura que fije ámbitos cerrados sino que con plena libertad se programan los cursos que se consideran más adecuados a las circunstancias e intereses de los posibles destinatarios, la Universidad de la república de Uruguay tiene un sitio web en

el cual publican información sobre las actividades extensionistas que se desarrollan, otro ejemplo es la Universidad de San Martin en Perú que tienen un sitio de extensión y proyección universitaria, en Cuba también la extensión universitaria resulta función ineludible en la vida de las universidades, y tema que concita cada vez a polémicas más agudas, un ejemplo de ello es el portal Web de la Universidad Central Martha Abreu de Villa Clara, dedicado a registrar información sobre las actividades extensionistas que se llevan a cabo en esa institución también la Universidad de Pinar del Rio tiene un sitio similar, en la UCI existe un sitio que brinda a grandes rasgos información sobre las actividades culturales y deportivas que se realizan dentro de la escuela, estos sistemas existentes en Cuba y en otras universidades fuera del país relacionados con la extensión universitaria, no satisfacen del todo el problema que existe en la facultad, respecto a la forma en que se está almacenando la información relacionada con los participantes en las actividades extensionistas, ninguno de ellos a pesar de profundizar mucho sobre el movimiento extensionista en la universidad, lleva un seguimiento de los estudiantes que participan en estos eventos, esto constituye una necesidad primordial para la facultad, ya que los datos no están registrados adecuadamente y dificulta la organización de estas actividades, con este fin se realiza este sistema, para llevar un control de los estudiantes y profesores que dan su apoyo, para ampliar el proceso de extensión universitaria en la facultad 6.

"Sin cultura no puede existir la auténtica libertad individual"

1.2 Metodología de desarrollo

Un proceso de software detallado y completo suele denominarse "metodología". Las metodologías se basan en una combinación de los modelos de procesos genéricos (cascada, evolutivo, incremental) [1]. Una metodología define con precisión los artefactos, roles y actividades involucrados, junto con prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto o guías para uso de herramientas de apoyo. Habitualmente se utiliza el término "método" para referirse a técnicas, notaciones y guías asociadas, que son aplicables a una (o algunas) actividades del proceso de desarrollo, considerando su filosofía de desarrollo, aquellas metodologías con mayor énfasis en la planificación y control del proyecto en especificación precisa de requisitos y modelado, reciben el apelativo de metodologías tradicionales, la metodología Rational Unified Process (RUP) que es más adaptable para proyectos de largo plazo es considerada una metodología tradicional, otras metodologías, denominadas Metodologías Ágiles, están más orientadas a la generación de código con ciclos muy cortos de desarrollo, se dirigen a equipos de desarrollo pequeños, hacen especial hincapié en aspectos humanos asociados al trabajo en equipo e involucran activamente al cliente en el proceso, algunas de estas metodologías son: Extreme Programming (XP) que se recomienda para proyectos de

corto plazo, la Microsoft Solution Framework (MSF) se adapta a proyectos de cualquier dimensión y de cualquier tecnología, Open Source Software Development, Scrum. Específicamente se utilizará la metodología de desarrollo RUP, es una de las metodologías más generales que existen actualmente, su finalidad no está restringida a guiar el desarrollo del *software*, sino cualquier tipo de proyecto. La estrategia de esta metodología es conseguir su objetivo por medio de orden y documentación, lo que lo convierte en el más fiel exponente de los métodos pesados. RUP define cuatro fases (inicio, elaboración, construcción y transición) y dentro de cada una de ellas el equipo de trabajo pasa por todos los flujos que son transversales a las fases, inclusive en varias iteraciones, los flujos de trabajo son modelación del negocio, requerimientos, análisis y diseño, implementación, prueba, instalación, administración del proyecto, administración de configuración y cambios y ambiente. Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las más importantes para alcanzar un grado de certificación en el desarrollo del software, está diseñada de forma que exista un entendimiento continuo y gradual de todos los implicados en el proyecto.

1.3 Herramientas de desarrollo

- Framework de desarrollo: Symfony 1.2.2.

Para abordar adecuadamente el tema que corresponde a esta sección es importante el conocimiento del significado del término *framework.* Un framework es un diseño reutilizable de todo o parte de un sistema de software, descrito por varias jerarquías de herencia de clases, generalmente algunas abstractas, y por las colaboraciones que se establecen entre las instancias de estas clases. La reutilización se produce mediante la instanciación del framework en una aplicación concreta, rellenando los puntos de variabilidad del diseño, en los que el desarrollador con reutilización incluye la funcionalidad específica de su sistema[2]. Típicamente, un *framework* puede incluir soporte de programas, bibliotecas y un lenguaje interpretado de gran importancia para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Se decidió utilizar como framework de desarrollo Symfony en su versión 1.2.2, está desarrollado completamente con PHP 5, ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel, Symfony es compatible con la mayoría de los gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server, se puede ejecutar tanto en plataformas Unix, Linux, etc. como en plataformas Windows. Fácil de instalar y configurar en la mayoría de plataformas, suficientemente flexible como para adaptarse a los casos más complejos, sigue la mayoría de mejores prácticas y patrones de diseño

para la Web, preparado para aplicaciones empresariales y adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.

- Lenguaje de programación: PHP 5

PHP no es un lenguaje de propósito general para programar cualquier aplicación, sino con un objetivo definido: La Web. Provee una sintaxis sencilla, intercalable entre los tags de HTML, y ofrece una gran facilidad para generar aplicaciones Web con contenidos dinámicos. Las aplicaciones Web escritas en PHP constan de un conjunto de scripts que interactúan con las fuentes que les proveen el contenido (bases de datos, archivos de disco, ficheros XML, etc.), procesan estos contenidos y generan una salida HTML que es la que se envía al navegador[3]. Para llevar a cabo la implementación del sistema se utilizará PHP5, distribuido bajo licencia Open Source es interpretado y de alto nivel, especialmente pensado para desarrollos Web y el cual puede ser embebido en páginas HTM. La mayoría de su sintaxis es similar a C, Java y Perl y es muy fácil de aprender. La meta de este lenguaje es permitir escribir a los creadores de páginas Web, páginas dinámicas de una manera rápida y fácil y su objetivo principal mejorar los mecanismos de la programación orientada a objetos, para solucionar las carencias de las anteriores versiones, un paso necesario para conseguir que PHP sea un lenguaje apto para todo tipo de aplicaciones y entornos, incluso los más exigentes.

- IDE de desarrollo: ECLIPSE 3.4

En español *Entorno Integrado de Desarrollo* (IDE) es una aplicación estructurada por un conjunto de herramientas que pueden ser utilizadas por un programador. Puede dedicarse exclusivamente para un lenguaje de programación o bien para varios. El IDE ECLIPSE en su versión 3.4 brinda nuevas facilidades como la disposición de un entorno mucho más flexible y profesional para controlar todo el ciclo de vida de un proceso de desarrollo. Además, brinda capacidades de refactorización del código fuente permitiendo adecuar el comportamiento externo de una función o clase sin cambiar el funcionamiento interno. Eclipse es un IDE de código abierto independiente de una plataforma, es una aplicación de cliente enriquecido, emplea *plug-ins* para proporcionar toda su funcionalidad, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Eclipse es, en el fondo, únicamente un armazón sobre la que se pueden montar herramientas de desarrollo. La arquitectura de *plug-ins* permite, además de integrar diversos lenguajes, introducir otras

aplicaciones que pueden resultar útiles durante el proceso de desarrollo, tales como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, entre otros.

- Gestor de Bases de Datos: PostgreSQL 8.2.x

Un sistema gestor de bases de datos (SGBD) es un software de propósito general que facilita el proceso de definir, construir y manipular la base de datos para diversas aplicaciones, almacena información y permite a los usuarios recuperarla y actualizarla en base a peticiones. Para los SGBD la arquitectura propuesta por el grupo del Instituto Nacional de Estándares Americano (ANSI) tiene tres niveles: el nivel interno (cercano al almacén físico de los datos), el nivel externo (cercano a los usuarios) y el nivel conceptual (nivel intermedio)[4]. El sistema gestor de bases de datos escogido para almacenar los datos de la aplicación es PostgreSQL v8.2. PostgreSQL es un sistema de base de datos relacional perteneciente al ámbito del software libre que destaca por su robustez, escalabilidad y cumplimiento de los estándares SQL. Cuenta con versiones para una amplia gama de sistemas operativos, entre ellos: Linux, Windows, Mac SX, Solaris, BSD, Tru64. Soporta ACID, o lo que es lo mismo, la realización de transacciones seguras, vistas, uniones, claves extranjeras, procedimientos almacenados, triggers, etcétera. Incluye la mayor parte de los tipos de datos especificados en los estándares SQL92 y SQL99, como: entero, numérico, booleano, char, varchar, fecha, interval o timestamp. PostgreSQL tiene otras características interesantes como son: alta concurrencia, que evita tener que bloquear una tabla cuando se está escribiendo en ella, copias de seguridad en línea, replicación asíncrona, transacciones anidadas, optimizador de consultas.

- Herramientas CASE: Visual Paradigm 6.1

Las Herramientas CASE (*Computer Aided Software Engineering*, Ingeniería de Software Asistida por Computadora) son aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo su coste en términos de tiempo y dinero[5]. Estas herramientas pueden ayudar en muchos de los aspectos del ciclo de vida de desarrollo del software, en tareas como la realización del diseño del proyecto, cálculo de costes, implementación de una parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores. La herramienta CASE utilizada es Visual Paradigm 6.1 es una herramienta diseñada para desarrollar software con programación orientada a objetos. Busca reducir la duración del ciclo de desarrollo, brindando ayuda a arquitectos, analistas, diseñadores y desarrolladores; permite el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación y tiene capacidades de ingeniería directa e inversa. Usa UML como lenguaje de modelado. Presenta una

interfaz de uso intuitiva y con muchas facilidades a la hora de modelar los diagramas que soportan la Ingeniería de Requerimientos. Una de las características más importantes de su uso es que brinda la posibilidad de sincronizar el modelo de diseño y el código en todo el ciclo de desarrollo una vez que se integra con el Eclipse, permitiendo la facilidad de programar directamente sobre el código fuente generado y a su vez actualizar el diseño con cambios que se realicen en la programación. Tiene una alta disponibilidad de múltiples versiones para cada necesidad, al igual que en las plataformas *Linux,* MacOS y *Windows.* Presenta un innovador analizador textual donde se introduce texto extraído de conversaciones con el cliente, se definen actores, entidades, casos de uso disponibles para la generación de artefactos posteriores; así mismo posee una herramienta de generación de reportes en formato PDF o HTML configurable y selectiva, se integra con entornos como Eclipse, *Hibernate* y Subversión, e importa o exporta formatos estándares de otras herramientas CASE como el *Rational Rose.* Ofrece además un entorno de creación de diagramas para UML 2.0, disponibilidad en múltiples plataformas, disponibilidad de integrarse en los principales IDEs, soporta una gama de lenguajes en la generación de código e ingeniería inversa en Java, C++, CORBA IDL, PHP, esquema de XML, Ada y Python, la generación de código soporta C #, VB .NET.

- Controlador de versiones :Subversión

Un software de sistema de control de versiones diseñado específicamente para reemplazar al popular CVS, el cual posee varias deficiencias. Es software libre bajo una licencia de tipo Apache/BSD y se le conoce también como svn por ser ese el nombre de la herramienta de línea de comandos, además tiene múltiples ventajas: Permite realizar modificaciones (incluyendo cambios a varios archivos) son atómicas, la creación de ramas y etiquetas es una operación más eficiente. Tiene costo de complejidad constante (O (1)) y no lineal (O(n)) como en CVS, permite que sólo se envíen las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos), permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.

- Lenguaje de modelado visual: Lenguaje Unificado de Modelado (UML)

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un plano del sistema, incluyendo aspectos conceptuales, tales como

procesos de negocios, funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. En fin UML es el enlace entre quien tiene la idea y el desarrollador, la comunicación es su principal objetivo. Es importante resaltar que UML es un lenguaje para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema, para documentar y construir, en otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como RUP), pero no específica en sí mismo qué metodología o proceso usar. El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos, con el uso de UML se puede desarrollar un modelado eficiente ya que en el mismo se evidencia gran simplicidad de la comunicación entre desarrolladores de software, facilidad de entendimiento y aprendizaje de sus principios, permite y viabiliza la comunicación entre trabajadores del proyecto y usuarios, además de que proporciona la estandarización de los elementos del diseño de sistemas y constituye el estándar más utilizado mundialmente.

1.4 Roles desempeñados y artefactos

El termino rol se emplea en UML para referirse a los papeles que desempeña un trabajador en los diferentes flujos de trabajo del proceso de software, un trabajador puede ocupar diferentes roles respecto a otros trabajadores es decir un arquitecto, por ejemplo puede participar en varios flujos de trabajo y asumir un rol diferente en cada uno de ellos, formalmente se define como un rol al comportamiento específico de una entidad que participa en un contexto particular[1].

En todo el proceso de elaboración de la aplicación se definieron tres roles fundamentales, analista de sistemas, diseñador e implementador.

- **Analista de sistemas**: Este rol dirige y coordina la adquisición de requisitos sintetizando la funcionalidad del sistema y delimitándolo, define las interacciones e identifica las entidades de entrada y salida que fluyen entre los actores y el sistema, y define formalmente el contexto de la operación del sistema. El analista de sistemas también define las características de rendimiento, físicas y otras características no funcionales del sistema para satisfacer las exigencias del contexto empresarial o de misión, una persona que actúe en este rol debe ser, por encima de todo, un experto en la identificación y la comprensión de problemas y oportunidades. Esto incluye la capacidad de articular las necesidades que se asocian con el

problema clave que se debe solucionar o la oportunidad a realizar, debe ser capaz también de colaborar de forma efectiva con otros miembros del equipo, debe tener grandes habilidades analíticas y una formación de base sólida en el modelado de requisitos mediante UML. El analista de sistemas también necesita habilidades de proceso de ingeniería de sistemas.

El **analista de sistemas** agrupa los roles que están involucrados fundamentalmente en la extracción e investigación de los requisitos del sistema. Este grupo está formado por los siguientes roles:

- **Diseñador de negocio**: Encargado de detallar la especificación de la organización o parte de ella. Los artefactos que realizan son: caso de uso del negocio, actor del negocio, realización de caso de uso del negocio, sistema del negocio, entidad del negocio, trabajador del negocio y eventos del negocio.

- **Analista del sistema:** Dirige y coordina el proceso de extracción de requisitos y desarrollo del modelo de casos de uso, los artefactos que realizan son: plan de gestión de requerimientos, glosario, atributos de requerimientos, especificación suplementaria, solicitudes de los stakeholder, documento visión, modelo de casos de uso, actor y Storyboard.

- **Especificador de requerimientos**: Específica los detalles de una o varias partes de la funcionalidad del sistema, describiendo uno o varios aspectos de los requisitos. Los artefactos que realizan son: casos de uso, especificación de requerimiento del software, paquete de casos de uso y requerimientos del software.

Los artefactos que se generan en esta etapa de desarrollo del proyecto son:

- *Descripción de Clases del Modelo del Dominio.*
- *Diagrama de Clases del Modelo del Dominio.*

- **Diseñador:** El diseñador identifica y define las responsabilidades, operaciones, atributos y relaciones de los elementos de diseño. El diseñador se asegura de que el diseño sea coherente con la arquitectura de software, y que esté detallado hasta un punto en que pueda proceder la implementación, un diseñador puede implementar una parte estructural del sistema (como un subsistema de implementación o de clases), o una parte funcional del sistema, como la

ejecución de guiones de uso o sus características que cruza clases/subsistemas, entre sus principales tareas se encuentran el diseño de clase, diseño del subsistema, análisis de caso de uso, diseño de caso de uso diseño de servicio, además para dirigir el diseño de la estructura de almacenamiento de datos persistentes que se utilizará en el sistema está el **Diseñador de base de datos**, el **diseñador de interfaz de usuario** coordina el diseño de la interfaz de usuario. Esto incluye recopilar los requisitos de utilización y los diseños de interfaz de usuario candidata a la creación de prototipos para cumplir estos requisitos, y el **diseñador de sistemas** que se encarga de definir y perfecciona el diseño del sistema.

Entre los artefactos que genera se encuentran:

- Clase de diseño: Es una descripción de un conjunto de objetos que comparten las mismas responsabilidades, relaciones, operaciones, atributos y semánticas.
- Paquete de diseño: Es una recopilación de clases, relaciones, ejecuciones de guión de uso, diagramas y otros paquetes. Se utiliza para estructurar el modelo de diseño dividiéndolo en componentes más pequeños.
- Subsistema de diseño: Describe una parte del sistema que encapsula comportamiento, expone un conjunto de interfaces y empaqueta otros elementos de modelo.

- **Implementador:** Este rol desarrolla los componentes de software y efectúa las pruebas de desarrollador para la integración en subsistemas más grandes, de acuerdo con los estándares adoptados de proyecto, es responsable además de los componentes de desarrollo y de prueba, de acuerdo con los estándares aprobados por el proyecto, para la integración en subsistemas más grandes. Cuando los componentes de prueba, como controladores o fragmentos para simulación, deben crearse para dar soporte a las pruebas el implementador también es responsable del desarrollo y las pruebas de los componentes de prueba y los subsistemas correspondientes.

Principales Artefactos:

- Subsistema de implementación: Consta de un conjunto de elementos de implementación. Estructura el modelo de implementación dividiéndolo en componentes más pequeños que se pueden integrar y probar separadamente.

- Elemento de implementación: Son los componentes físicos que forman una implementación, que incluyen archivos y directorios. Incluyen los archivos de código de software (origen, binario o ejecutable), los archivos de datos y los archivos de documentación.

- Prueba de desarrollador: Abarca el trabajo tradicionalmente pensado bajo las categorías siguientes: Pruebas de unidad, parte de las Pruebas de integración, y algunos aspectos de lo que se denomina Pruebas del sistema.

1.5 Patrones de Casos de Uso, Arquitectura y Diseño.

Para el adecuado desempeño de estos roles, definidos anteriormente en el desarrollo de la aplicación, es de vital importancia la adecuada utilización de los patrones de Casos de Uso, Arquitectura y Diseño. Un patrón es una descripción de un problema y su solución, recibe un nombre y puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en diversas circunstancias[6]. Los patrones son de gran importancia para el diseño y la implementación de un sistema, ya que constituyen una guía para resolver problemas comunes en la programación, al usarlos se pueden identificar de forma más eficiente las dificultades que surgen en el desarrollo de la aplicación, y que solución es la mejor para resolver dicho problema, además usar patrones permite contar con un lenguaje común entre programadores; describen problemas que ocurren una y otra vez en un contexto determinado y ofrece la solución a ese problema de forma que pueda ser utilizada varias veces sin repetir ninguna acción.

Patrones de casos de uso:

Los patrones de casos de uso utilizados son:

CRUD (Creating, Reading, Updating, Deleting): El patrón CRUD, se basa específicamente en la fusión de casos de uso simples para formar una unidad conceptual, el **CRUD** completo consta de un caso de uso, llamado ***Información CRUD*** o ***Gestionar información****[7]* modela todas las operaciones que

pueden ser realizadas sobre una parte de la información de un tipo específico, tales como creación, lectura, actualización y eliminación.

Múltiples actores (*roles comunes)***:** Los dos actores jueguen el mismo rol sobre el CU. Este rol es representado por otro actor, heredado por los actores que comparten este rol.

Para el desarrollo de la aplicación se decidió usar Symfony como Framework de desarrollo, el mismo introduce como patrón de arquitectura el MVC (Modelo- Vista -Controlador).

- **Modelo**: Representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- **Vista**: Transforma el modelo en una página web que permite al usuario interactuar con ella.
- **Controlador**: Se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

Con el uso del patrón MVC se introduce el trabajo con patrones generales de software para asignar responsabilidades (GRASP), alguno de estos son:

- **Experto:** Ante el problema de cuál es el principio fundamental en virtud del cual se asignan las responsabilidades en el diseño orientado a objetos, propone como solución asignar una responsabilidad al experto en información[6] es decir la clase que cuenta con la información necesaria para cumplir la responsabilidad; este es uno de los más utilizados en el desarrollo de la aplicación, puesto que Propel es la librería externa que utiliza Symfony para realizar su capa de abstracción en el modelo.
- **Creador:** El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento, al escogerlo como creador, se da soporte al bajo acoplamiento[6].
- **Bajo Acoplamiento:** El Bajo Acoplamiento es un principio que se debe tener presente durante las decisiones de diseño. Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño[6]. Estimula asignar una responsabilidad de modo que su colocación no incremente demasiado el acoplamiento, que produzca los resultados negativos proporcionados por un alto acoplamiento.

- **Alta Cohesión**: Como el patrón Bajo Acoplamiento, también Alta Cohesión es un principio a tener presente en todas las decisiones de diseño. Es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño, se da una alta cohesión funcional cuando los elementos de un componente (clase, por ejemplo) colaboran para producir algún comportamiento bien definido[6].
- **Controlador:** En todos los casos, si se recurre a un diseño orientado a objetos, hay que elegir los controladores que manejen esos eventos de entrada. Este patrón ofrece una guía para tomar decisiones apropiadas que generalmente se aceptan, la misma clase controlador debería utilizarse con todos los eventos sistémicos de un caso de uso, de modo que se pueda conservar la información referente al estado del caso de uso[8].

El MVC también incluye otro tipo de patrones de diseño (GOF) la estructura symfony cumple con algunos de estos patrones: Instancia única y fábrica abstracta:

- **Singleton (Instancia única):** Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia[8]. En el controlador frontal hay una llamada a sfContext una clase del nucleo de Symfony que se usa mediante getInstance(), un objeto muy útil que guarda una referencia a todos los objetos del núcleo de Symfony y puede ser accedido desde cualquier punto de la aplicación.

- **Abstract Factory (Fábrica abstracta):** Permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí haciendo transparente el tipo de familia concreta que se esté usando[8]. Cuando el frameworks necesita por ejemplo crear un nuevo objeto para una petición, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea.

- **Command:** Command encapsula una petición en un objeto, pudiéndo realizar operaciones como crear una cola de objetos, gestionarla y poder deshacer las operaciones, es un patrón de diseño de comportamiento de objetos además de que posee las ventajas de los patrones de diseño**.**

- **Decorador:** El patrón decorador (decorator) se encarga como su nombre lo indica de decorar las responsabilidades de un objeto de forma dinámica y transparentemente para sus clientes. Constituye una alternativa para la herencia al decorar responsabilidades de un subconjunto de objetos,

permite añadir o eliminar funcionalidades a objetos de forma dinámica y transparente además de evitar el estallido de subclases para poder tener todas las combinaciones de una serie de funcionalidades independientes. Opcionalmente, puede invocar métodos adicionales antes y después de la redirección, incluso puede decorarse un objeto varias veces con el mismo decorador, lo que sería imposible con la herencia, favorece la definición de interfaces y clases bases ligeras.

Conclusiones

En este capítulo se realizó un estudio teórico de la investigación lo que dió la necesidad de desarrollar este módulo, ya que puede servir para el control y seguimiento de la extensión universitaria en la facultad. Se mencionaron los patrones de casos de uso, arquitectura y diseño utilizados, los roles y artefactos definidos en el proceso de desarrollo de software, y se fundamentaron además las herramientas y la metodología a utilizar.

CAPÍTULO 2: CARTERÍSTICAS DEL SISTEMA

Introducción

En este capítulo se describe la solución propuesta utilizando los componentes del modelo de dominio. Se hace una breve descripción del problema y se específica el glosario de términos, se muestran los requerimientos funcionales y no funcionales. Además se describen los actores y casos de usos del sistema, así como el diagrama de casos de uso del mismo.

2.1 ¿Por qué modelo del dominio?

Un modelo de dominio captura los tipos más importantes de objetos que existen o los eventos que suceden en el contexto del sistema. Los objetos del dominio tienen como objetivo fundamental la comprensión y descripción de las clases más importantes del sistema.

Se utiliza modelo de dominio en el desarrollo de la aplicación debido a la poca estructuración de los procesos de gestión de la información de extensión universitaria en la facultad 6, y para poder entender el contexto en que se desarrolla el sistema es necesario definir conceptos que sea posible agrupar en un Modelo de Dominio.

2.2 Breve descripción del problema

El vicedecano de extensión controla todo lo relacionado con la extensión de la facultad, dígase festival, juegos deportivos y otras actividades extensionistas que se planifiquen. En el festival participan estudiantes y profesores de la facultad, dicha actividad consta de varias manifestaciones artísticas y estas a su vez tienen varios números artísticos, los cuales van a tener un horario de ensayo. En los juegos deportivos participan estudiantes y profesores de la facultad, estos tienen un cronograma donde el presidente de la FEU es el encargado de su realización. En las otras actividades extensionistas participan los estudiantes y profesores de la facultad.

2.3 Glosario de términos para el dominio

Vicedecano de extensión: cliente que utilizará el sistema para su control y seguimiento.

Festival: actividad que se realiza y la cual tiene números artísticos.

Juegos deportivos: actividad que se realiza y la cual tiene un cronograma.

Otras actividades extensionistas: actividades extensionistas que surjan sin una planificación previa.

Estudiantes: persona que participa en las diferentes actividades extensionistas.

Profesores: persona que participa en las diferentes actividades extensionistas.

Manifestaciones artísticas: categorías que puede tener un festival.

Números artísticos: categorías que puede tener una manifestación artística.

Horario de ensayo: horario de ensayo que tienen los números artísticos.

Cronograma: planificación de los juegos deportivos.

Presidente de la FEU: persona que se encargará de controlar el cronograma.

2.4 Representación del modelo del dominio

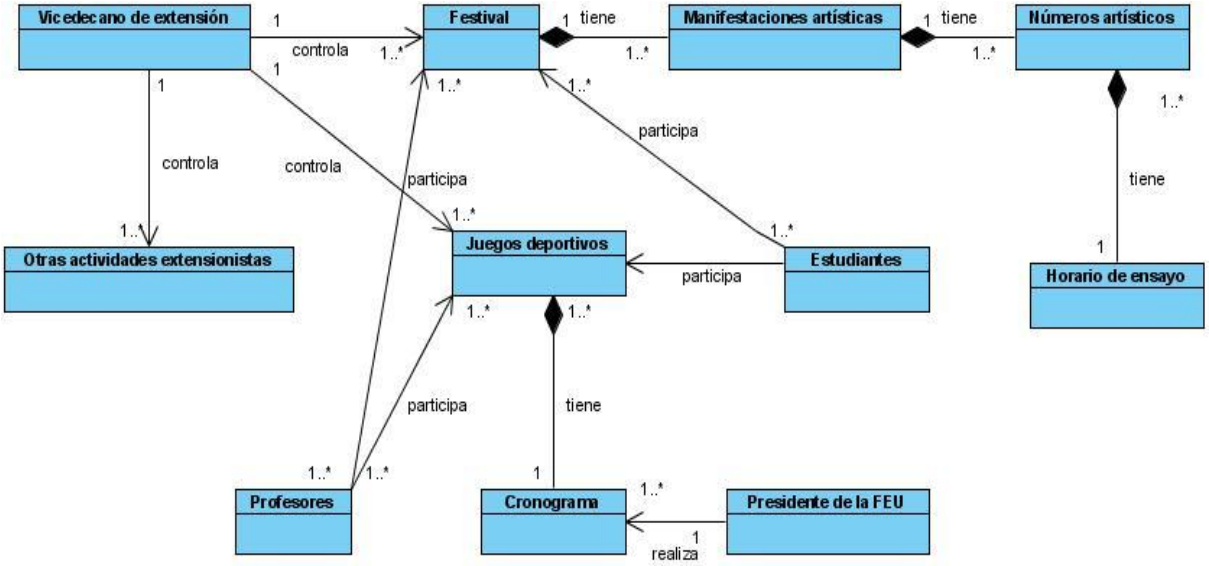


Fig. 1: Diagrama modelo del dominio.

2.5 Especificación de Requerimientos del sistema

2.5.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir.

R1: Gestionar datos de artistas en festival

R1.1: Registrar datos de artistas en festival

R1.2: Modificar datos de artista en festival

R1.3: Eliminar datos de artista en festival

R1.4: Visualizar listado de participantes en festival

R1.5 Buscar Artistas

R2: Gestionar festival

R2.1: Insertar festival

R2.2: Eliminar festival

R2.3: Modificar festival

R2.4: Visualizar listado de festivales

R3: Gestionar manifestación cultural

R3.1: Registrar manifestación cultural

R3.2: Modificar manifestación cultural

R3.3: Eliminar manifestación cultural

R3.4: Ver listado de manifestaciones culturales

R3.5: Ver listado de participantes por manifestación

R4: Gestionar número artístico

R4.1: Registrar número artístico

R4.2: Modificar número artístico

R4.3: Eliminar número artístico

R4.4: Ver listado de números artísticos

R4.5: Ver listado de participantes por número artístico

R5: Gestionar horario de ensayo

R5.1: Insertar horario

R5.2: Modificar horario

R5.3: Eliminar horario

R5.4: Visualizar horario de festival

R6: Gestionar datos de participantes en los juegos deportivos

R6.1: Registrar un nuevo participante en los juegos deportivos

R6.2: Modificar datos de participantes en los juegos deportivos

R6.3: Eliminar datos de participantes en los juegos deportivos

R6.4: Visualizar datos de participante en los juegos deportivos

R6.5: Buscar deportistas

R7: Gestionar juegos deportivos

R7.1: Insertar juegos deportivos

R7.2: Eliminar juegos deportivos

R7.3: Modificar datos de juegos deportivos

R7.4:Visualizar listado de juegos deportivos

R8: Gestionar datos de deporte

R8.1: Insertar deporte

R8.2: Modificar deporte

R8.3: Eliminar deporte

R8.4: Ver listado de deporte

R8.5: Visualizar datos de participantes por deporte

R9: Gestionar cronograma de juegos deportivos

R9.1: Registrar cronograma

R9.2: Modificar datos de cronograma

R9.3: Eliminar cronograma

R9.4 Visualizar cronograma de juegos deportivos

R10: Gestionar otras actividades extensionistas

R10.1: Insertar actividad extensionista

R10.2: Eliminar actividad extensionista

R10.3: Modificar actividad extensionista

R10.4: Visualizar actividad extensionista

2.5.2 Requisitos no funcionales

Los requisitos no funcionales responden a cualidades que el producto debe tener y las características para que este sea confiable, atractivo y seguro.

• **Apariencia o interfaz externa.**

La aplicación debe reflejar tonalidad verde.

• **Usabilidad.**

La aplicación podrá ser utilizada por personal vinculado al manejo de información de estudiantes y profesores de la facultad 6 de la UCI.

• **Portabilidad.**

El sistema podrá ser usado sobre los sistemas operativos Windows y Linux.

• **Seguridad**.

El sistema debe contar con protección contra acciones no autorizadas o que puedan afectar la integridad de los datos que se manejan en ella. El sistema de seguridad es a través de la autenticación de usuarios, a los cuales se le asignan roles y a estos a su vez se le asignan permisos para interactuar con la aplicación.

• **Políticos-culturales.**

Se empleará en la aplicación el idioma español.

• **Legales.**

Para desarrollo de la aplicación se hará uso de herramientas de software libre.

• **Software.**

Cliente:

- ✓ Cualquier sistema operativo con interfaz gráfica y red.
- ✓ Navegador Web Internet Explorer (5.5 o superior) o Mozilla firefox 2.0 o superior.

Servidor:

- ✓ Sistema Operativo Linux.
- ✓ Servidor de Base de Datos PostgreSQL 8.2.
- ✓ Servidor Web Apache 2.2 o Superior, PHP 5.1 o superior.

• **Hardware.**

Para el desarrollo:

- ✓ Sistema operativo Windows 2000 o superior.
- ✓ Navegador web Internet Explorer 5.5 o superior o Mozilla Firefox 2.0 o superior.
- ✓ Servidor web Apache 2.2 o superior.
- ✓ Servidor de Base de Datos PostgreSQL 8.2.

Para la explotación:

Cliente:

- ✓ Sistema operativo Windows 2000 o superior.
- ✓ Navegador web Internet Explorer (5.5 o superior) o Mozilla Firefox 2.0 o superior.

Servidor:

- ✓ Sistema operativo Windows 2000 o superior.
- ✓ Servidor de Base de Datos PostgreSQL 8.2.
- ✓ Servidor web Apache 2.2 o Superior, PHP 5.2 o superior.

2.6 Actores del sistema

Los actores del sistema pueden ser las personas, sistemas o hardware externo que se relacionan o interactúan con dicho sistema, los que se definen en este sistema se muestran en la siguiente tabla.

Actor	Descripción
Vicedecano de extensión	Representa el usuario del sistema que tiene la posibilidad de interactuar con todas las funcionalidades.
Presidente de la FEU	Representa el usuario del sistema que tiene la posibilidad de interactuar con alguna de las funcionalidades del mismo.
Usuario	Generalización de los actores del sistema que necesitan autenticarse.

Fig. 2 Actores del sistema

2.7 Diagrama de casos de uso del sistema.

Un diagrama de casos de usos del sistema representa gráficamente a los procesos y su interacción con los actores.

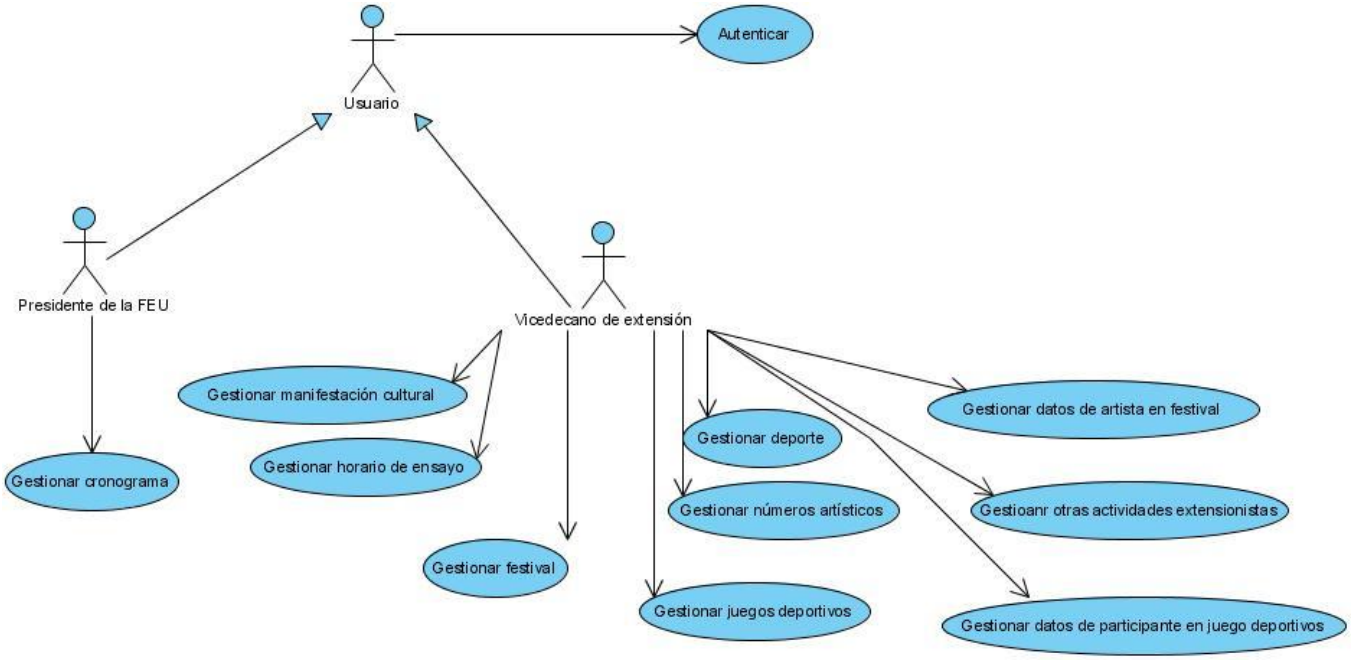


Fig. 3 Diagrama de Caso de uso del sistema

2.7.1 Matriz de trazabilidad

-->	Gestionar dat...	Gestionar fes...	Gestionar ma...	Gestionar nú...	Gestionar hor...	Gestionar dat
Registrar datos de artistas en festival	X					
Modificar datos de artista en festival	X					
Eliminar datos de artista en festival	X					
Visualizar listado de participantes en festival	X					
Buscar Artista	X					
Insertar festival		X				
Eliminar festival		X				
Modificar festival		X				
Visualizar festival		X				
Registrar manifestación			X			
Modificar manifestación			X			
Eliminar manifestación			X			
Visualizar listado de manifestaciones con sus participantes			X			
Registrar número artístico				X		
Modificar número artístico				X		
Eliminar número artístico				X		
Visualizar listado de participantes por números artísticos				X		
Insertar horario					X	
Modificar horario					X	
Eliminar horario					X	

Fig. 4 Matriz de trazabilidad

2.8 Descripción de los casos de uso del sistema

2.8.1 Descripción del caso de uso Gestionar datos de artistas en festival

Caso de Uso:	**Gestionar datos de artistas en festival**
Actores:	Vicedecano de extensión (inicia).
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Registrar datos artistas en festival, Modificar datos de artista en festival, Eliminar datos de artista en festival Visualizar listado de participantes en festival o Buscar Artistas.
Referencia:	**RF1.1, RF1.2, RF1.3, RF1.4, RF1.5.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema.
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión se dirige al menú y selecciona Participantes en festivales.	2- El sistema muestra varias opciones entre las que se encuentran: Nuevo participante, Buscar artistas y Ver participantes.
3- El Vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona "Nuevo participante" ver sección **Nuevo participante**. - Si selecciona "Buscar artistas" ver sección **Buscar artistas**. -Si selecciona "Ver participantes" ver sección **Ver participantes**.
Sección "Nuevo participante"	
Acción del Actor	**Respuesta del Sistema**

1- El Vicedecano de extensión selecciona la opción “Nuevo participante”.	2- El sistema muestra un formulario con los criterios de búsqueda (nombre, grupo, departamento, sexo, usuario) para buscar el estudiante o profesor que desea registrar en los festivales.
3- El Vicedecano de extensión busca la persona por cualquiera de los criterios de búsqueda del formulario.	4- El sistema muestra el resultado de la búsqueda realizada y la opción de registrarle la participación en festival.
5.- El Vicedecano de extensión marca registrar datos de artistas.	6- El sistema muestra un formulario con los datos que debe registrar del participante (festival en que va a participar, número artístico).
7-El Vicedecano de extensión marca aceptar.	8- El sistema guarda los cambios y muestra un mensaje al usuario “Datos de artista registrados satisfactoriamente”.
Sección “Buscar artistas”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Buscar artistas”.	2- El sistema muestra un formulario para buscar el estudiante o profesor que ha participado en algún festival, por las diferentes opciones de búsqueda (nombre, festival, manifestación, grupo docente, departamento, sexo).
3- El Vicedecano de extensión busca la persona.	4- El sistema muestra el resultado de la búsqueda realizada y las opciones de visualizar datos de artista (nombre y apellidos, sexo, grupo, correo electrónico, curso del festival en el que participó y la manifestación en que participó), modificar datos de artista y eliminar datos de artista.
3-El Vicedecano de extensión selecciona una de las opciones.	4- Si selecciona “Modificar datos de artista” ver sección **Modificar datos de artista**. - Si selecciona ¨Eliminar datos de artista ¨ ver sección **Eliminar datos de artista l**.
Sección “Ver participantes”	

Acción del Actor	Respuesta del Sistema
1- El Vicedecano de extensión selecciona la opción “Ver participantes”.	2- El sistema muestra un listado con todos los participantes en festival registrados hasta el momento.
Sección ¨Modificar datos de artista¨	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión escoge la opción de modificar datos de artista.	2-El sistema muestra un formulario con los datos de la participación en festival del artista (festival, número artístico).
3- El Vicedecano de extensión cambia los datos necesarios.	4- El sistema verifica que seleccione el festival y el número artístico.
	5- El sistema después de modificar los datos muestra un mensaje al usuario ”Se modificó la información del participante exitosamente”
Sección ¨Eliminar datos de artista¨	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión escoge la opción de eliminar datos de artista.	2-El sistema muestra un formulario con los festivales en que ha participado el artista aficionado.
3-El Vicedecano de extensión escoge el festival que quiere eliminar y marca el vínculo para eliminar los datos.	4-El sistema muestra un mensaje ”Estás seguro de querer eliminar esta participación en festival”.

Flujos Alternos	
Sección "Nuevo participante"	
Acción del Actor	**Respuesta del Sistema**
	6.1- Si el Vicedecano de extensión deja en blanco el campo de los festivales el sistema muestra un mensaje "No seleccionó el festival".
	6.2-Si el Vicedecano de extensión deja en blanco el campo de los números artísticos el sistema muestra un mensaje "No seleccionó el número artístico".
	6.1- Si el Vicedecano intenta registrarle a un participante un festival o un número artístico que ya fueron asignados a ese estudiante, el sistema muestra un mensaje "Esa persona ya tiene registrada participación en ese festival en el número artístico (título del número)".
Sección "modificar datos de artista"	
Acción del Actor	**Respuesta del Sistema**
	4.1 Si El Vicedecano de extensión no selecciona ni el número artístico ni el festival, el sistema muestra los mensajes"Seleccione un festival", "Seleccione al menos un número artístico".
Poscondiciones:	Quedan registrados los artistas con sus nuevos datos.
Prioridad:	Crítico

2.8.2 Descripción del caso de uso Gestionar festival

Caso de Uso:	Gestionar festival
Actores:	Vicedecano de extensión (inicia).
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión

	decide Insertar festival, Eliminar festival, Modificar festival o Visualizar listado de festivales.
Referencia:	RF2.1, RF2.2, RF2.3, RF2.4.
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión se dirige al menú y selecciona Festivales.	2- El sistema muestra varias opciones entre las que se encuentran: Nuevo festival y Listar festivales.
3- El Vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona "**Nuevo festival**" ver sección **Nuevo festival.** - Si selecciona "Listar festivales" ver sección **Listar festivales.**
Sección "Nuevo festival"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Nuevo festival".	2- El sistema muestra un formulario en el que pide entrar los datos del festival (curso académico).
3- El Vicedecano de extensión inserta los datos del nuevo festival.	4- El sistema verifica que llene los campos.
Sección "Listar festivales "	

Acción del Actor	Respuesta del Sistema
1- El Vicedecano de extensión selecciona la opción "Listar festivales".	2- El sistema muestra un formulario con los festivales que existen en ese momento y las opciones de editar festival y modificar festival.
3- El Vicedecano de extensión selecciona alguna de las opciones.	Si selecciona "Modificar festival" ver sección **Modificar festival**. - Si selecciona ¨Eliminar festival¨ ver sección **Eliminar festival**.
Sección ¨Modificar festival¨	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión escoge la opción de modificar festival.	2- El sistema muestra un formulario con los datos del festival seleccionado (curso académico).
3- El Vicedecano de extensión cambia los datos necesarios.	
Sección ¨Eliminar festival¨	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión selecciona el vínculo "eliminar festival"	2-El sistema muestra un mensaje "Estás seguro".
	3-El sistema muestra un mensaje "Se ha eliminado el festival".
Flujos Alternos	
Sección "Nuevo festival"	
Acción del Actor	**Respuesta del Sistema**
	3.1- Si el Vicedecano de extensión intenta registrar un festival que ya está en la base de datos se muestra un mensaje "Ya

	existe un Festival con el mismo Curso Académico“.
	4.1 Si el Vicedecano de extensión o llena los campos, el sistema muestra un mensaje “campo obligatorio”.
Poscondiciones:	Quedan registradas los nuevos festivales
Prioridad:	Crítico

2.8.3 Descripción del caso de uso Gestionar manifestación cultural

Caso de Uso:	**Gestionar manifestación cultural**
Actores:	Vicedecano de extensión(inicia)
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Registrar manifestación cultural, Modificar manifestación cultural, Eliminar manifestación cultural, Ver listado de manifestaciones culturales o Ver listado de participantes por manifestación.
Referencia:	**RF3.1, RF3.2, RF3.3, RF3.4, RF3.5.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema
Flujo Normal de Eventos	
Sección “General”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión se dirige al menú y selecciona Manifestaciones artísticas.	2- El sistema muestra varias opciones entre las que se encuentran: Nueva manifestación artística, Listar manifestaciones artísticas y Participantes por manifestación.
3- El Vicedecano de extensión selecciona una de las opciones	4- Si selecciona “Nueva manifestación artística” ver sección **Nueva manifestación artística.**

presentadas.	- Si selecciona “Listar manifestaciones artísticas” ver sección **Listar manifestaciones artísticas.** - Si selecciona “Participantes por manifestación.” Ver sección **Participantes por manifestación.**
Sección “Nueva manifestación artística”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Nueva manifestación artística”.	2- El sistema muestra un formulario en el que pide llenar los datos de la manifestación cultural a registrar (Tipo manifestación, y género).
3- El Vicedecano de extensión inserta los datos a llenar.	4- El sistema verifica que los formularios de los datos no estén en blanco.
Sección ¨Listar manifestaciones artísticas¨	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Listar manifestaciones artísticas”.	2- El sistema muestra un formulario con todas las manifestaciones creadas que existen en ese momento y las opciones de modificar manifestación artística y eliminar manifestación artística de cada una de las manifestaciones.
3-El Vicedecano selecciona una de las opciones.	4-Si selecciona “Modificar manifestación artística” ver sección **Modificar manifestación artística.** - Si selecciona “Eliminar manifestación artística.” Ver sección **Eliminar manifestación artística.**
Sección ¨ Modificar manifestación artística¨	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona	2- El sistema muestra un formulario con los datos de la manifestación seleccionada. (Tipo manifestación, género).

en el vínculo editar manifestación artística.	
3- El Vicedecano de extensión cambia los datos necesarios.	4- El sistema verifica que no deje los formularios de los datos en blanco.
Sección ¨ Eliminar manifestación artística¨	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión selecciona en el vínculo eliminar manifestación artística.	2-El sistema muestra un mensaje “Estás seguro”.
Sección “Participantes por manifestación”	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión selecciona la opción “Participantes por manifestación”.	2- El sistema muestra un formulario donde se selecciona la manifestación artística que se le quieran ver los participantes.
3-El Vicedecano de extensión selecciona una manifestación.	4-El sistema muestra los datos de los artistas aficionados que participan en esta manifestación (nombre y apellidos, tipo de persona, grupo/departamento).
Flujos Alternos	
Sección “Nueva manifestación artística”	
Acción del Actor	**Respuesta del Sistema**
	3.1-El Vicedecano de extensión intenta registrar algún género que ya se encuentra en la base de datos el sistema muestra un mensaje “Ya existe un Género con ese nombre”.

	4.1- El Vicedecano de extensión deja en blanco los formularios de los datos, el sistema muestra un mensaje "campo obligatorio" señalando los campos que lo son.
Sección "Modificar Manifestación artística"	
Acción del Actor	**Respuesta del Sistema**
	4.1- Si el Vicedecano deja datos en blanco, el sistema muestra un mensaje "campo obligatorio".
Sección "Participantes por manifestación"	
Acción del Actor	**Respuesta del Sistema**
	4.1- Si el Vicedecano selecciona una manifestación en la que no hay participantes, el sistema muestra un mensaje" No hay participantes en esa manifestación artística".
Poscondiciones:	Quedan registradas las manifestaciones con sus nuevos datos.
Prioridad:	Critico

2.8.4 Descripción del caso de uso Gestionar número artístico

Caso de Uso:	**Gestionar número artístico**
Actores:	Vicedecano de extensión(inicia)
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Registrar número artístico, Modificar número artístico, Eliminar número artístico, Ver listado de números artísticos o Ver listado de participantes por número artístico.
Referencia:	**RF4.1, RF4.2, RF4.3, RF4.4, RF4.5.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema
Flujo Normal de Eventos	
Sección "General"	

Acción del Actor	Respuesta del Sistema
1- El Vicedecano de extensión se dirige al menú y selecciona Números artísticos.	2- El sistema muestra varias opciones entre las que se encuentran: Nuevo número artístico, Listado de números artísticos y Ver participantes en números artísticos.
3- El Vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona "Nuevo número artístico" ver sección **Nuevo número artístico.** - Si selecciona "Listado de números artísticos" ver sección **Listado de números artísticos.** - Si selecciona "Ver participantes en números artísticos" ver sección **Ver participantes en números artísticos.**

Sección "Nuevo número artístico"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Nuevo número artístico".	2- El sistema muestra un formulario en el que pide llenar los datos del número artístico a registrar (género manifestación artística, festival, título del número artístico).
3- El Vicedecano de extensión inserta los datos a llenar.	4- El sistema verifica que los formularios de los datos no estén en blanco.

Sección "Listado de números artísticos"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Listado de números artísticos".	2- El sistema muestra un formulario con todos los números artísticos creados que existen en ese momento y las opciones para modificarlos y eliminarlos.
3-El Vicedecano de extensión selecciona una de las opciones	4-Si selecciona "Modificar número artístico" ver sección **Modificar número artístico.**

	- Si selecciona “Eliminar número artístico.” Ver sección **Eliminar número artístico.**
Sección ¨ Modificar número artístico ¨	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción modificar número artístico.	2- El sistema muestra un formulario con los datos del número artístico seleccionado (género manifestación artística, festival, título del número artístico).
3- El Vicedecano de extensión cambia los datos necesarios.	4- El sistema verifica que no deje los formularios de los datos en blanco.
Sección ¨ Eliminar número artístico ¨	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión selecciona eliminar número artístico.	2-El sistema muestra un mensaje “Estás seguro de querer eliminar este número artístico”.
	3-Si está seguro borra los datos del número artístico y actualiza la base de datos.
Sección “Ver participantes en números artísticos”	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión selecciona la opción Ver participantes en números artísticos.	2-El sistema muestra un formulario donde se selecciona el número artístico.
3- El Vicedecano de extensión selecciona un número	4-El sistema muestra los datos de los que participan en ese número artístico (nombre y apellidos, tipo de persona, grupo/departamento).

artístico.	
Flujos Alternos	
Sección "Nuevo número artístico"	
Acción del Actor	**Respuesta del Sistema**
	3.1- Si el vicedecano intenta registrar datos que ya están en la base de datos se muestra un mensaje "Ya existe un número artístico con ese título".
	4.1- El Vicedecano de extensión deja en blanco los formularios de los datos, el sistema muestra un mensaje "campo obligatorio¨ señalando los campos que son obligatorios.
Sección "Modificar número artístico "	
Acción del Actor	**Respuesta del Sistema**
	4.1- Si deja algún campo en blanco, el sistema muestra un mensaje "campo obligatorio "en el campo que dejo en blanco.
Sección "Ver participantes en números artísticos "	
Acción del Actor	**Respuesta del Sistema**
	4.1- Si el Vicedecano de extensión selecciona un numero artístico en no haya participado nadie, el sistema muestra un mensaje" No hay participantes "
Poscondiciones:	Quedan registrados los números artísticos con sus nuevos datos.
Prioridad:	Critico

2.8.5 Descripción del caso de uso Gestionar horario de ensayo

Caso de Uso:	**Gestionar horario de ensayo**
Actores:	Vicedecano de extensión(inicia)
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Insertar horário, Modificar horário, Eliminar horário o Visualizar

	horario de festival.
Referencia:	**RF5.1, RF5.2, RF5.3, RF5.4.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema

Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión se dirige al menú y selecciona Ensayos de festivales.	2- El sistema muestra varias opciones entre las que se encuentran: Nuevo ensayo y Ver horario de ensayo de festival.
3- El Vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona "Nuevo ensayo" ver sección **Nuevo ensayo.** - Si selecciona "Ver horario de ensayo de festival" ver sección **Ver horario de ensayo de festival**.

Sección "Nuevo ensayo"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Nuevo ensayo".	2- El sistema muestra un formulario en el que pide llenar los datos del horario a insertar (festival, fecha, hora de inicio, hora de fin, lugar, número artístico, festival, lugar).
3- El Vicedecano de extensión inserta los datos a llenar.	4-El sistema verifica que los formularios de los datos no estén en blanco.
	5- El sistema muestra un mensaje "Se ha guardado el ensayo".
Sección "Ver horario de ensayo de festival"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la	2- El sistema muestra un formulario con los festivales que existan en ese momento y su correspondiente horario de ensayo

opción ¨Ver horario de ensayo de festival¨.	
3- El Vicedecano de extensión escoge un festival y selecciona ver horario de ensayo.	4- El sistema muestra un formulario con los datos del ensayo. (Fecha del ensayo, hora de inicio, hora de fin, lugar, número artístico, festival) y la opción de editarlos y eliminarlos.
5- El Vicedecano de extensión selecciona una de las opciones.	6- Si selecciona “Modificar ensayo” ver sección **Modificar ensayo.** - Si selecciona “Eliminar ensayo.” Ver sección **Eliminar ensayo.**
Sección ¨ Modificar ensayo ¨	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión escoge un ensayo y selecciona modificar ensayo.	2-El sistema muestra los datos del ensayo que van a ser modificados (festival, numero artístico, fecha, hora de inicio, hora de fin, lugar).
3-El Vicedecano de extensión modifica los datos y guarda los cambios.	4- El sistema muestra un mensaje “Se ha guardado el ensayo’
	5- El sistema verifica que no deje los formularios de los datos en blanco.
Sección ¨ Eliminar ensayo ¨	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión escoge un ensayo y	2- El sistema guarda los cambios y actualiza la base de datos.

selecciona eliminar ensayo.	
Flujos Alternos	
Sección "Nuevo ensayo"	
Acción del Actor	**Respuesta del Sistema**
	4.1- El Vicedecano de extensión deja en blanco los formularios de los datos, el sistema muestra un mensaje "campo obligatorio".
Sección "Modificar ensayo"	
Acción del Actor	**Respuesta del Sistema**
	5.1- Si los datos se quedaron en blanco, el sistema muestra un mensaje "campo obligatorio".
Poscondiciones:	Quedan registrados los nuevos datos de los horarios.
Prioridad:	Critico

2.8.6 Descripción del caso de uso Gestionar datos de participantes en los juegos deportivos

Caso de Uso:	**Gestionar datos de participantes en los juegos deportivos**
Actores:	Vicedecano de extensión(inicia)
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Registrar un nuevo participante en los juegos deportivos, Modificar datos de participantes en los juegos deportivos, Eliminar datos de participantes en los juegos deportivos, Visualizar datos de participante en los juegos deportivos o Buscar deportista.
Referencia:	**RF6.1, RF6.2, RF6.3, RF6.4, RF6.5.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado

	en el sistema.
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión se dirige al menú y selecciona Participantes en juegos deportivos.	2- El sistema muestra varias opciones entre las que se encuentran: Ver participantes, Registrar nuevo participante y Buscar deportistas.
3- El Vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona "Registrar nuevo participante" ver sección **Registrar nuevo participante.** - Si selecciona "Buscar deportistas" ver sección **Buscar deportistas.** **-** Si selecciona "Ver participantes" ver sección **Ver participantes.**
Sección "Registrar nuevo participante"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Registrar nuevo participante".	2- El sistema muestra un formulario con los criterios de búsqueda (nombre, grupo, departamento, sexo) para buscar el estudiante que desea registrar en los juegos deportivos.
3- El Vicedecano de extensión busca la persona por cualquiera de los criterios de búsqueda del formulario.	4- El sistema muestra el resultado de la búsqueda realizada y la opción de registrarle la participación en juegos.
5- El Vicedecano de extensión pincha en el vínculo registrar participación en juegos.	6- El sistema muestra un formulario con los datos que debe registrar del participante (juego deportivo en que va a participar, deporte).
7- El Vicedecano de extensión marca aceptar.	8- El sistema guarda los cambios y actualiza la base de datos.

Sección “Buscar deportistas”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Buscar deportistas”.	2- El sistema muestra un formulario para buscar el estudiante que ha participado en algún juego deportivo, por las diferentes opciones de búsqueda (nombre, juego deportivo, deporte, grupo, sexo).
3- El Vicedecano de extensión busca la persona.	4- El sistema muestra el resultado de la búsqueda realizada y las opciones de visualizar datos de deportista (nombre y apellidos, sexo, grupo docente, correo electrónico, juego deportivo en el que participó y el deporte), modificar datos de deportista y eliminar datos de deportista.
5-El Vicedecano de extensión selecciona una de las opciones.	6- Si selecciona “Modificar datos de deportista” ver sección **Modificar datos de deportista**. - Si selecciona ¨Eliminar datos de deportista¨ ver sección **Eliminar datos de deportista**.
Sección “Modificar datos de deportista”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Modificar datos de deportista”.	2-El sistema muestra un formulario con la información correspondiente a su participación en los juegos deportivos (juegos deportivos y los deportes).
3- El Vicedecano de extensión cambia los datos necesarios.	4- El sistema verifica que no deje datos en blanco.
Sección “Eliminar datos de deportista”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la	2-El sistema muestra un formulario con los juegos deportivos del deportista seleccionado.

opción “Eliminar datos de deportista”.	
3- El Vicedecano de extensión marca el vínculo para eliminar los datos de deportista.	4-El sistema muestra los juegos deportivos en los que ha participado.
5- El Vicedecano de extensión selecciona la opción eliminar participación.	6-El sistema muestra un mensaje “Estás seguro de querer eliminar esta participación”.
	7-Si está seguro borra los datos del participante en los juegos y actualiza la base de datos.
Sección “Ver participantes”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Ver participantes”.	2- El sistema muestra un listado de todos los estudiantes que se les ha registrado participación en los juegos deportivos.
Flujos Alternos	
Sección “Registrar nuevo participante”	
Acción del Actor	**Respuesta del Sistema**
	6.1- El Vicedecano de extensión deja en blanco alguno de los campos del formulario, el sistema muestra un mensaje “campo obligatorio”.
	7.1- Si el Vicedecano intenta registrarle a un participante un juego deportivo o un deporte que ya fueron asignados a ese estudiante, el sistema muestra un mensaje “Esa persona ya tiene registrada participación en ese Juego Deportivo, practicando ese mismo deporte”.
Sección “Modificar datos de deportista”	
Acción del Actor	**Respuesta del Sistema**

	4.1- Si deja datos en blanco, el sistema muestra un mensaje “campo obligatorio”.
Sección “Ver participantes”	
Acción del Actor	**Respuesta del Sistema**
	2.1-En el listado de participantes en los juegos también están las opciones de modificar, eliminar y ver datos de participante.
	2.2- En caso de que no exista ningún participante registrado el sistema brinda la posibilidad de registrar uno nuevo.
Poscondiciones:	Quedan registrados los participantes con sus nuevos datos.
Prioridad:	Crítico

2.8.7 Descripción del caso de uso Gestionar juegos deportivos

Caso de Uso:	**Gestionar juegos deportivos**
Actores:	Vicedecano de extensión (inicia).
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Insertar juegos deportivos, Eliminar juegos deportivos, Modificar datos de juegos deportivos o Visualizar listado de juegos deportivos.
Referencia:	**RF7.1, RF7.2, RF7.3, RF7.4.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema.
Flujo Normal de Eventos	
Sección “General”	
Acción del Actor	**Respuesta del Sistema**

1- El Vicedecano de extensión se dirige al menú y selecciona Juegos deportivos.	2- El sistema muestra varias opciones entre las que se encuentran: Nuevo juego deportivo y Listar juegos deportivos.
3- El Vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona “nuevo juego deportivo” ver sección **Nuevo juego deportivo**. - Si selecciona “Listar juegos deportivos” ver sección **Listar juegos deportivos**.
Sección “Nuevo juego deportivo”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Nuevo juego deportivo”.	2- El sistema muestra un formulario en el que pide entrar datos de los juegos deportivos (curso académico).
3- El Vicedecano de extensión inserta los datos del nuevo juego deportivo.	4- El sistema verifica que los campos no estén en blanco.
Sección “Listar juegos deportivos”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Listar juegos deportivos”.	2- El sistema muestra todos los juegos deportivos que están registrados hasta el momento y la opción de modificar juego deportivo y eliminar juego deportivo.
3- El Vicedecano de extensión selecciona una de las opciones.	4-Si selecciona “Modificar juego deportivo” ver sección **Modificar juego deportivo**. - Si selecciona “Eliminar juego deportivo” ver sección **Eliminar juego deportivo.**
Sección “Modificar juego deportivo”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Modificar juego deportivo”.	2- El sistema muestra un formulario con todos los datos del juego deportivo seleccionado (curso académico).
3- El Vicedecano de extensión cambia los datos necesarios.	4- El sistema verifica que no deje datos en blanco.

	5- Si no deja los datos en blanco, el sistema guarda los cambios realizados en la base de datos.
	6-El sistema muestra un mensaje “Se ha guardado el juego deportivo”.
Sección “Eliminar juego deportivo”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Eliminar juego deportivo”.	2- El sistema muestra un mensaje “Se ha eliminado el juego deportivo”.
	3- Si está seguro borra los datos del juego deportivo y actualiza la base de datos.
Flujos Alternos	
Sección “Nuevo juego deportivo”	
Acción del Actor	**Respuesta del Sistema**
	3.1- Si el Vicedecano de extensión selecciona un juego deportivo que ya exista, el sistema muestra un mensaje “Ya existe un juego deportivo con ese curso académico”.
	4.1- El Vicedecano de extensión deja en blanco alguno de los campos, el sistema muestra un mensaje “campo obligatorio”.
Sección “Modificar juego deportivo”	
Acción del Actor	**Respuesta del Sistema**
	4.1- Si deja datos en blanco, el sistema muestra un mensaje “campo obligatorio”.
Poscondiciones:	Quedan registrados los juegos deportivos con sus nuevos datos.
Prioridad:	Crítico

2.8.8 Descripción del caso de uso Gestionar datos de deporte

Caso de Uso:	**Gestionar datos de deporte**
Actores:	Vicedecano de extensión (inicia).
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Insertar deporte, Modificar deporte, Eliminar deporte Ver listado de deporte o Visualizar datos de participantes por deporte.
Referencia:	**RF8.1, RF8.2, RF8.3, RF8.4, RF8.5.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema.

Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión se dirige al menú y selecciona Gestionar datos de deporte.	2- El sistema muestra varias opciones entre las que se encuentran: Nuevo deporte, Listar deporte y Listar participantes por deporte.
3- El vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona "Nuevo deporte" ver sección **Nuevo deporte**. - Si selecciona "Listar deportes" ver sección **Listar deportes**. - Si selecciona "Listar participantes por deporte" ver sección **Listar participantes por deporte.**
Sección "Nuevo deporte"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Nuevo deporte".	2- El sistema muestra un formulario para registrar los datos del deporte a registrar (Nombre del deporte y el sexo).

3- El Vicedecano de extensión inserta los datos.	4- El sistema verifica que el campo no esté en blanco.
Sección "Listar deportes"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Listar deportes".	2- El sistema muestra todos los deportes que están registrados hasta el momento y la opción de modificarlos y eliminarlos.
3- El Vicedecano selecciona una de las opciones.	Si selecciona **"Modificar deporte"** ver sección **Modificar deporte**. - Si selecciona **"Eliminar deporte"** ver sección **Eliminar deporte.**
Sección "Modificar deporte"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Modificar deporte".	2- El sistema muestra un formulario con todos los datos del deporte seleccionado (nombre del deporte, sexo).
3- El Vicedecano de extensión cambia los datos necesarios.	4- El sistema verifica que no deje datos en blanco.
5- El Vicedecano de extensión hace los cambios.	6- Si no deja los datos en blanco, el sistema guarda los cambios realizados en la base de datos.
	7-El sistema muestra un mensaje "Se ha guardado el deporte".
Sección "Eliminar deporte"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Eliminar deporte".	2- El sistema muestra un mensaje"Estás seguro".
	3- Si está seguro borra los datos del deporte y actualiza la base de datos.

	4- El sistema muestra un mensaje “Se ha eliminado el deporte”.
Sección “Listar participantes por deporte”	
Acción del Actor	**Respuesta del Sistema**
1-El Vicedecano de extensión selecciona la opción “Listado participantes por deporte”.	2- El sistema muestra un formulario para seleccionar el deporte.
3-El Vicedecano de extensión selecciona un deporte.	4-El sistema muestra los datos de los deportistas que participan en ese deporte (nombre participante, grupo docente).
Flujos Alternos	
Sección “Nuevo deporte”	
Acción del Actor	**Respuesta del Sistema**
	4.1- El Vicedecano de extensión deja en blanco el formulario del nombre del deporte, el sistema muestra un mensaje “campo obligatorio”.
Sección “Modificar deporte”	
Acción del Actor	**Respuesta del Sistema**
	6.1- Si deja datos en blanco, el sistema muestra un mensaje “campo obligatorio”.
Sección “Listar participantes por deporte”	
Acción del Actor	**Respuesta del Sistema**
3.1- El Vicedecano de extensión selecciona un deporte en el que no haya participado nadie.	4.1- El sistema muestra un mensaje “No hay participantes en este deporte”.
Poscondiciones:	Quedan registrados los deportes con sus nuevos datos.
Prioridad:	Simple

2.8.9 Descripción del caso de uso Gestionar cronograma de los juegos deportivos

Caso de Uso:	**Gestionar cronograma de los juegos deportivos**
Actores:	Presidente de la FEU (inicia).
Resumen:	El caso de uso se inicia cuando el Presidente de la FEU decide Registrar cronograma, Modificar datos de cronograma, Eliminar cronograma o Visualizar cronograma de juegos deportivos.
Referencia:	**RF9.1, RF9.2, RF9.3, RF9.4.**
CU asociados:	
Precondiciones:	El Presidente de la FEU debe estar autenticado en el sistema.

Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Presidente de la FEU se dirige al menú y selecciona Cronogramas de juegos deportivos.	2- El sistema muestra varias opciones entre las que se encuentran: Nuevo cronograma y Ver cronograma.
3- El Presidente de la FEU selecciona una de las opciones presentadas.	4- Si selecciona **"Nuevo cronograma"** ver sección **Nuevo cronograma**. - Si selecciona **"Ver cronograma"** ver sección **Ver cronograma**.
Sección "Nuevo cronograma"	
Acción del Actor	**Respuesta del Sistema**
1- El Presidente de la FEU selecciona la opción "Nuevo cronograma".	2- El sistema muestra un listado con los juegos deportivos registrados hasta el momento.
3- El Presidente de la FEU escoge el juego deportivo al que quiere registrarle el cronograma.	4- El sistema muestra un formulario para añadir las realizaciones de juego que conforman el cronograma.

5- El Presidente de la FEU adiciona las los datos de las realizaciones del juego.	6- El sistema guarda los datos de ese cronograma.
	7-El sistema muestra un mensaje ”Realización de juego guardada”.
Sección “Ver cronograma”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Ver cronograma”.	2- El sistema muestra un formulario con los juegos deportivos registrados y su correspondiente cronograma.
3- El Vicedecano de extensión escoge un juego deportivo y selecciona el vínculo ver cronograma.	4- El sistema muestra un formulario con los datos del cronograma. (Fecha, hora realización, lugar, equipos, deporte) y la opción de editarlos y eliminarlos.
5- El Vicedecano de extensión selecciona una de las opciones.	6-Si selecciona “Modificar cronograma” ver sección **Modificar cronograma.** - Si selecciona “Eliminar cronograma.” Ver sección **Eliminar cronograma.**
Sección “Modificar cronograma”	
Acción del Actor	**Respuesta del Sistema**
1- El Presidente de la FEU selecciona la opción “Modificar cronograma”.	2- El sistema muestra los datos de la realización de juego del cronograma (equipo1, equipo2, deporte, fecha, hora, lugar).
3- El Presidente de la FEU cambia los datos necesarios.	4- El sistema verifica que no deje datos en blanco.
	5- Si no deja los datos en blanco, el sistema guarda los cambios realizados en la base de datos.

	6-El sistema muestra un mensaje “Se ha modificado la realización de juego”.
Sección “Eliminar cronograma”	
Acción del Actor	**Respuesta del Sistema**
1- El Presidente de la FEU selecciona la opción “Eliminar cronograma”.	2-El sistema elimina las realizaciones de juego del cronograma y actualiza la base de datos
Flujos Alternos	
Sección “Nuevo cronograma”	
Acción del Actor	**Respuesta del Sistema**
	6.1- El Presidente de la FEU deja en blanco alguno de los campos con asterisco, el sistema muestra un mensaje “campo obligatorio”.
Sección “Modificar cronograma”	
Acción del Actor	**Respuesta del Sistema**
	5.1- Si deja datos en blanco, el sistema muestra un mensaje “campo obligatorio” señalando los campos que lo son.
Poscondiciones:	Quedan registrados los cronogramas con sus nuevos datos.
Prioridad:	Crítico

2.8.10 Descripción del caso de uso Gestionar otras actividades extensionistas.

Caso de Uso:	**Gestionar otras actividades extensionistas.**
Actores:	Vicedecano de extensión (inicia).
Resumen:	El caso de uso se inicia cuando el Vicedecano de extensión decide Insertar actividad extensionista, Eliminar actividad extensionista, Modificar actividad extensionista o Visualizar actividad extensionista.

Referencia:	**RF10.1, RF10.2, RF10.3, RF10.4.**
CU asociados:	
Precondiciones:	El Vicedecano de extensión debe estar autenticado en el sistema.

Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión se dirige al menú y selecciona Actividades extensionistas.	2- El sistema muestra varias opciones entre las que se encuentran: Nueva actividad extensionista y Listado de actividades extensionistas.
3- El Vicedecano de extensión selecciona una de las opciones presentadas.	4- Si selecciona "Nueva actividad extensionista" ver sección **Nueva actividad extensionista**. - Si selecciona "Listado de actividades extensionistas" ver sección **Listado de actividades extensionistas**.
Sección "Nueva actividad extensionista"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Nueva actividad extensionista".	2- El sistema muestra un formulario en el que pide entrar datos la actividad a registrar (nombre actividad, hora, responsables, fecha, comentario, lugar).
3- El Vicedecano de extensión inserta los datos de la nueva actividad.	4- El sistema verifica que los campos no estén en blanco.
Sección "Listado de actividades extensionistas"	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción "Listado de actividades extensionistas".	2- El sistema muestra las actividades extensionistas y las opciones de ver sus datos más específicos, eliminar y modificar los datos de la actividad.

3-El vicedecano selecciona una de las opciones.	4-Si selecciona “Eliminar datos” ver sección **Eliminar datos.** - Si selecciona “Modificar datos” ver sección **Modificar datos.**
Sección “Modificar datos”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Modificar datos”.	2- El sistema muestra un formulario con todos los datos de la actividad seleccionada (nombre actividad, hora, responsables, fecha, comentario, lugar).
3- El Vicedecano de extensión cambia los datos necesarios.	4- El sistema verifica que no deje datos en blanco.
	5- Si no deja los datos en blanco, el sistema guarda los cambios realizados en la base de datos.
Sección “Eliminar datos”	
Acción del Actor	**Respuesta del Sistema**
1- El Vicedecano de extensión selecciona la opción “Eliminar datos”.	2- El sistema guarda los cambios y actualiza la base de datos.
Flujos Alternos	
Sección “Nueva actividad extensionista”	
Acción del Actor	**Respuesta del Sistema**
	4.1- El Vicedecano de extensión deja en blanco alguno de los campos con asteriscos, el sistema muestra un mensaje “campo obligatorio”.
Sección “Modificar datos”	
	4.1- Si deja datos en blanco, el sistema muestra un mensaje “campo obligatorio”.

Poscondiciones:	Quedan registradas las nuevas actividades
Prioridad:	Crítico

Conclusiones

En este capítulo se definieron conceptos, que fueron relacionados mediante un diagrama de Modelo de Dominio. Se definieron además los requisitos funcionales y no funcionales, los actores así como los casos de uso con los que contará el sistema y la descripción de cada uno de ellos.

CAPÍTULO 3: DISEÑO DEL SISTEMA

Introducción

En este capítulo se modelan los artefactos correspondientes al flujo de trabajo de análisis y diseño donde se refleja a través del diseño como el sistema va a cumplir con los requerimientos definidos anteriormente, contiene algunos de los diagramas de clase del diseño realizados, también se incluye en el capítulo el diagrama de despliegue y el de secuencia**.**

3.1 Arquitectura del sistema

La arquitectura es la Vista estructural de alto nivel del sistema, que define un estilo o la combinación de estos para dar una solución. Se concentra en requerimientos no funcionales los cuales se satisfacen mediante modelado y diseño de la aplicación. Es un punto esencial para el éxito o fracaso de un proyecto.[9].Debido a la complejidad que tiene la arquitectura de software es bastante difícil identificarla en un solo modelo o diagrama es por ello que muchos de los elementos contenidos en ella se representan a través de las vistas, una vista es una presentación de un modelo, la cual es una descripción completa de un sistema desde una particular perspectiva[9] , para definir adecuadamente las vistas que se utilizan para identificar la arquitectura de software es necesario usar el modelo 4+1 este modelo define 4 vistas principales:

1. Vista Lógica (Logical View): Engloba modelo de objetos, clases, entidad – relación, etc.

2. Vista de Proceso (Process View): Incluye el modelo de concurrencia y sincronización.

3. Vista de Desarrollo (Development View): Define la organización estática del software en su entorno de desarrollo (librerías, componentes, etc.).

4. Vista Física (Physical View): Describe el modelo de correspondencia software – hardware.

A estas vistas también se incluye la vista de casos de uso, estructurada por las necesidades funcionales que tiene el sistema.

3.1.1 Aplicación de los Patrones GRASP y GoF en Symfony

El uso del framework Symfony obliga al programador aplicar patrones de diseño, a continuación se presenta una explicación detallada del uso de los patrones de diseño.

Patrón Experto

- Encapsula toda la lógica de los datos y son generadas las clases con todas las funcionalidades comunes de las entidades, las clases que realizan esta funcionalidad son las clases Peer, por ejemplo FestivalPeer y BaseFestivalPeer las mismas tienen atributos necesarios para realizar esta tarea, por lo que se puede decir que implementan las acciones directamente con la base de datos, por este motivo se aplica el patrón experto.

```
<?php


abstract class BaseFestivalPeer {

    const DATABASE_NAME = 'propel';

    const TABLE_NAME = 'festival';

    const CLASS_DEFAULT = 'lib.model.Festival';

    const NUM_COLUMNS = 2;

    const NUM_LAZY_LOAD_COLUMNS = 0;

    const ID_FESTIVAL = 'festival.ID_FESTIVAL';

    const ID_CURSO_ACADEMICO = 'festival.ID_CURSO_ACADEMICO';
```

Patrón Creador

- En las clases Actions contenidas en Symfony se encuentran las acciones definidas para la aplicación. En estas se crean los objetos de las clases que representan las entidades, por lo que de este modo la clase Actions es "creador" de dichas entidades, un ejemplo de esto es en la clase

“gestionar_festivalActions” del módulo gestionar_festival, en la acción editar se toma el Id de un festival, para modificar la información relacionada con el mismo, para esto necesita crear una instancia de la clase FestivalPeer que es la que contiene los datos del festival.

```
public function executeEditar(sfWebRequest $request)
{
    $festival = FestivalPeer::retrieveByPk($request->getParameter('id_festival'));
    if($festival == null)
    {
        $this->getUser()->setFlash('error','Ese festival no existe o fue eliminado!');
        $this->redirect('gestionar_festival/listar');
    }
    $this->form = new FestivalForm($festival);
}
```

Patrón Controlador

- En la aplicación todas las peticiones Web son manejadas por un solo controlador frontal (sfActions), que es el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la dirección entrada por el usuario, la clase sfController del núcleo de Symfony es utilizada por el controlador frontal Index.php que es el que se encarga de manejar estas peticiones.

```php
<?php


abstract class sfController
{
  protected
    $context            = null,
    $dispatcher         = null,
    $controllerClasses  = array(),
    $maxForwards        = 5,
    $renderMode         = sfView::RENDER_CLIENT;



  public function __construct($context)
  {
    $this->initialize($context);
  }


  public function initialize($context)
  {
    $this->context     = $context;
    $this->dispatcher  = $context->getEventDispatcher();


    $this->maxForwards = sfConfig::get('sf_max_forwards');
  }
```

Alta cohesión

- Symfony permite asignar responsabilidades con una alta cohesión, por ejemplo las clases Actions tiene la responsabilidad de definir las acciones para las plantillas y colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder a las properties, es decir, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas proporcionando que el software sea flexible frente a grandes cambios.

Bajo acoplamiento

- En las clases Actions se hereda solamente de sfActions para lograr un bajo acoplamiento de clases, también en la capa del modelo ya que las clases que tienen el acceso a los datos no se encuentran estrechamente relacionadas con las clases de abstracción de datos. Hay poca dependencia entre esas clases lo que permite una mayor reutilización.

Patrones GoF

Patrón Decorador

- Una aplicacion desarrollada en Symfony está estructurada por tres elementos fundamentales entre los que se encuentra la vista que está compuesta por el layaut y las plantillas de cada modulo, dentro del layout está el archivo layout.php, este archivo, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el layout, o si se mira desde el otro punto de vista, el layout decora la plantilla. Este comportamiento es una implementación del patrón de diseño llamado "decorator" explicado anteriormente.

Singleton (Instancia única)

- En el controlador frontal hay una llamada a sfContext una clase del nucleo de Symfony que se usa mediante getInstance(), un objeto muy útil que guarda una referencia a todos los objetos del núcleo de Symfony y puede ser accedido desde cualquier punto de la aplicación.

Command

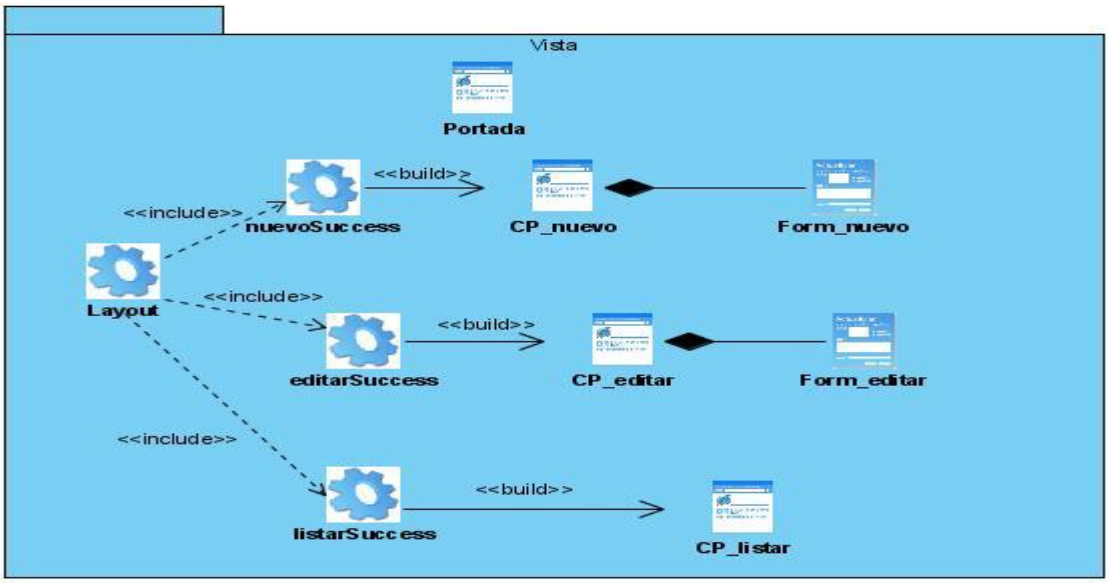
- En Symfony al usar este patrón permite facilitar el uso de parámetros de las acciones a realizar, la clase sfFrontWebController contiene el método dispatch que es el que se encarga de determinar cual módulo y acción usar en dependencia de la petición pasada por el usuario como parámetro.

```php
<?php


class sfFrontWebController extends sfWebController
{

  public function dispatch()
  {
    try
    {

      sfFilter::$filterCalled = array();


      $request    = $this->context->getRequest();
      $moduleName = $request->getParameter('module');
      $actionName = $request->getParameter('action');

      if (empty($moduleName) || empty($actionName))
      {
        throw new sfError404Exception(sprintf('Empty module and/or action after parsing the URL "%s" (%s/%s).
$request->getPathInfo(), $moduleName, $actionName));
      }


      $this->forward($moduleName, $actionName);
    }

}
```

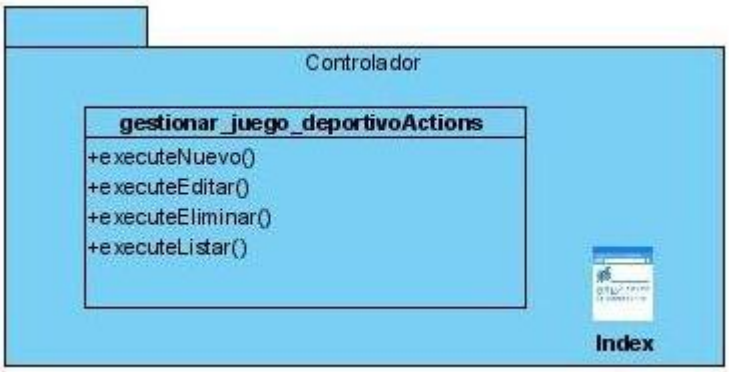
3.2 Vista lógica del sistema

La vista lógica muestra cómo la funcionalidad es diseñada en el interior del sistema, en términos de la estructura estática y comportamiento dinámico del sistema. Es un subconjunto del Modelo de Casos de Uso y puede incluir algunas realizaciones de casos de uso [10]. En la vista lógica de un sistema se trata de representar los elementos más importantes del diseño que forman parte de la arquitectura del sistema, se figuran las realizaciones de casos de uso más importantes y la organización de estos en paquetes y subsistemas. El uso de un framework que utiliza la arquitectura MVC obliga a dividir y organizar el código de acuerdo a las convenciones establecidas por el framework, en el caso específico de symfony su vista lógica está compuesta por tres elementos, la capa de la vista, el controlador y el modelo, el código correspondiente a la presentación se guarda en la vista, el código de manipulación de datos se guarda en el modelo y la lógica de procesamiento de las peticiones constituye el controlador, el principio más importante de la arquitectura MVC es la separación del código de la aplicación en estas tres capas, dependiendo de su naturaleza, a continuación se muestra como está conformada la vista lógica en el módulo cumpliendo con la estructura del framework.

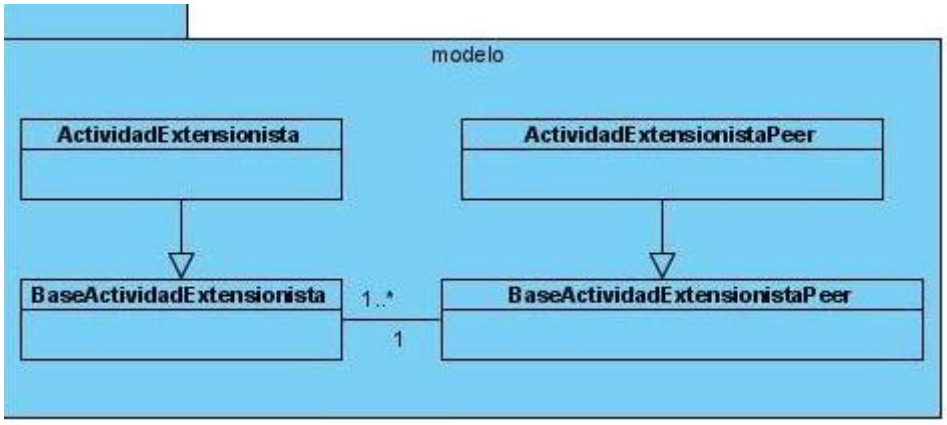
En la capa de la Vista se encuentran los elementos cuyas responsabilidades están asociadas a la presentación de los datos, aquí se ubica el Layout, que contiene la información visual común a todas las páginas. Se encuentra también el código asociado a las plantillas la cual se encarga de visualizar la información que el controlador devuelve y además contiene la lógica de la vista cuya responsabilidad es asegurar la correcta integración del Layout y la Plantilla.



En la capa de controlador se encuentran las acciones que constituyen el corazón de la aplicación, puesto que contienen toda la lógica de la aplicación. Las acciones utilizan el modelo y definen variables para la vista. Cuando se realiza una petición web en una aplicación Symfony, la URL define una acción y los parámetros de la petición.



En la capa de Modelo se localiza el código referente al acceso a datos, dichas clases se generan de forma automática en dependencia de la estructura de la base de datos, la librería Propel se encarga de esta generación automática y crea la estructura básica de las clases y genera el código necesario.



3.3 Mapa de Navegación

Los mapas de navegación en el ámbito de las aplicaciones web se definen como esquemas que dan la posibilidad de observar de forma general cuáles son las secciones en las que está dividida la aplicación, o sea representan gráficamente la ubicación en un espacio determinado de un grupo de elementos con los que van a interactuar los usuarios. La navegabilidad de la aplicación se organizó de la siguiente forma: desde la página principal se podrá acceder a las acciones correspondientes a los festivales, a las que pertenecen a los juegos deportivos y a las actividades extensionistas, una vez en ellas se puede acceder a las páginas que contienen las operaciones relacionadas con este caso de uso, por ejemplo registrar un nuevo participante, eliminarlo, modificarlo, ver sus datos, o ver un listado de todos los participantes registrados, de esta misma forma se hará para los juegos deportivos y las actividades extensionistas, la siguiente figura muestra como ha sido estructurado el mapa y la forma en que el usuario puede acceder a las diferentes acciones del módulo.

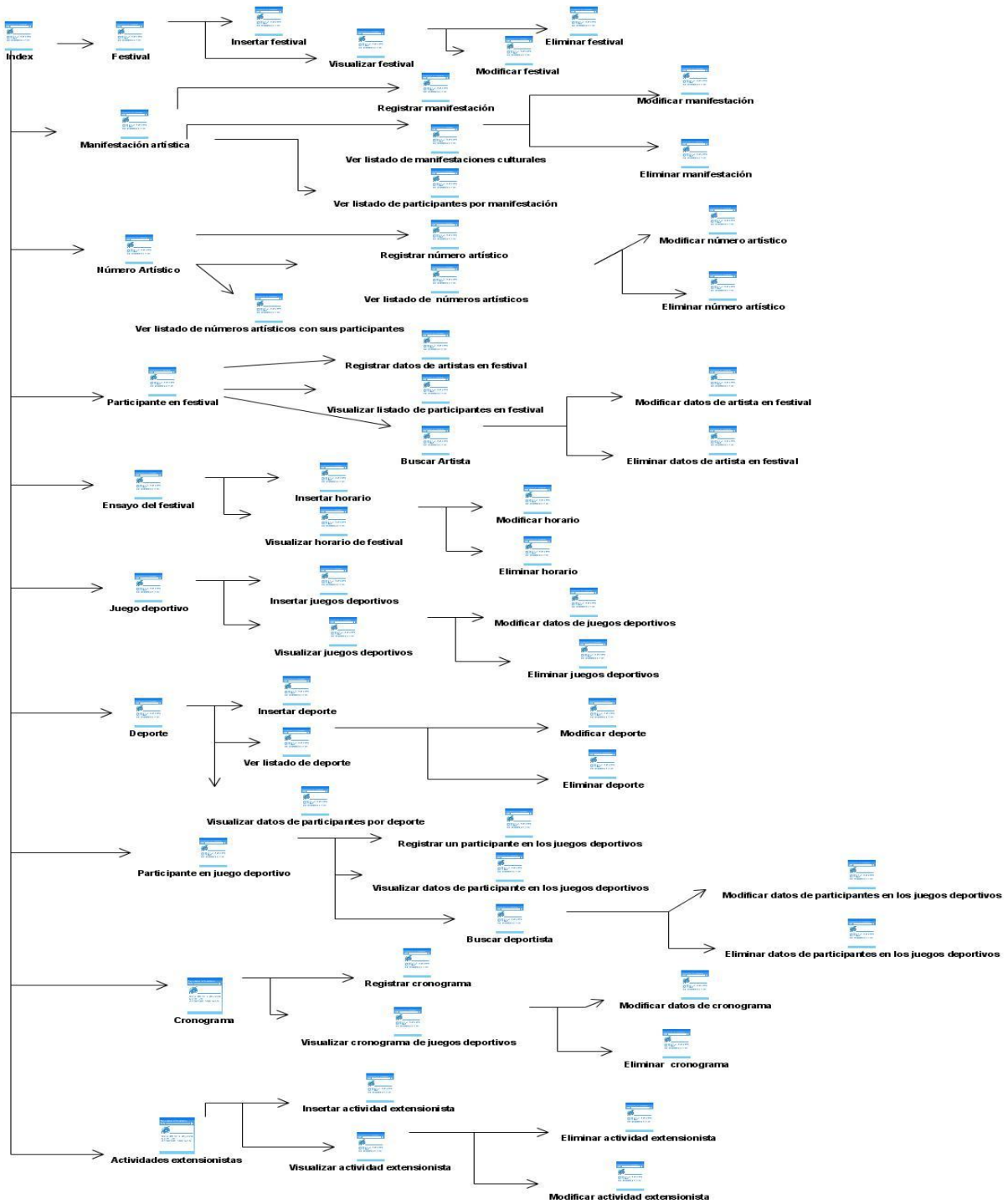


Fig. 5 Mapa de navegación

3.4 Diagrama de clases del diseño

La vista en Symfony se encarga de producir las páginas que se muestran como resultado de las acciones, está compuesta por diversas partes, estando cada una de ellas especialmente preparada para que pueda ser fácilmente modificable por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones. El sistema de extensión universitaria está estructurado a través de tres elementos: el controlador, el modelo y las vistas. El uso del framework Symfony en el desarrollo de la aplicación incluye el patrón de arquitectura Modelo-Vista-Controlador (MVC) por esta razón los diagramas de clases del diseño pertenecientes a cada caso de uso deben estar en correspondencia con la forma en que las clases del sistema, los elementos y componentes del mismo interactúan entre sí, cumpliendo siempre con el patrón arquitectónico definido por el framework. La arquitectura MVC separa el modelo y la vista, por lo que se consigue un mantenimiento más sencillo de las aplicaciones. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones, todas las solicitudes de los usuarios realizadas en las paginas clientes son gestionadas por los controladores frontales (index.php y frontend_dev.php), estos controladores frontales delegan la verdadera labor a las acciones**,** estas acciones a su vez son agrupadas en módulos. El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes, entre otras cosas, del tipo de gestor de bases de datos utilizado por la aplicación, Symfony guarda todas las clases y archivos relacionados con el modelo en el directorio lib/model/. La vista es con lo que el usuario interactúa.

A continuación se describen los paquetes y subsistemas de diseño que conforman la vista lógica:

Paquete Modelo: Este paquete es el encargado de encapsular la lógica del dominio y el acceso a datos.

Paquete Vista: En este paquete se encuentran las clases que presentan la lógica de la aplicación al usuario.

Paquete Controlador: En este paquete se agrupan las clases que implementa la lógica de la aplicación.

Subsistema Propel: Es un Modelo de Objeto Relacional (ORM, por sus siglas en inglés) que gestiona el modelo del sistema.

Subsistema Componentes Symfony: Representa las funcionalidades del Symfony que serán utilizadas para el funcionamiento del sistema.

A continuación se muestran los diagramas del diseño correspondientes a cada caso de uso.

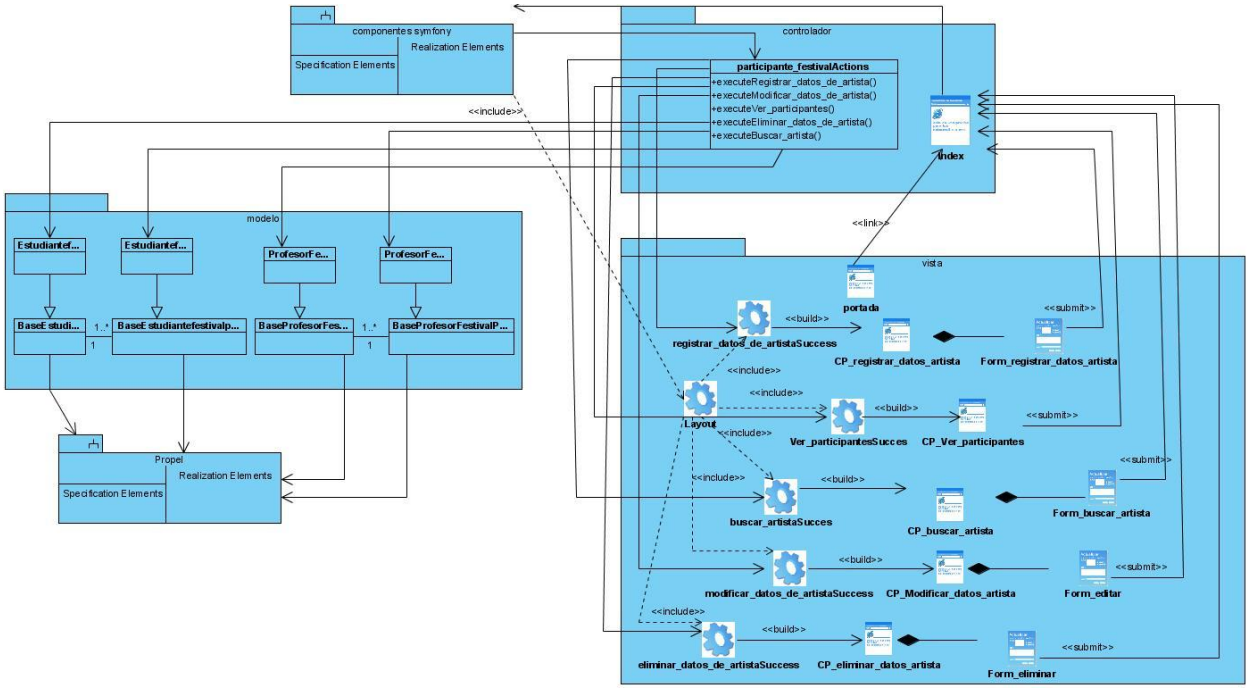


Fig. 6 Diagrama de clases del diseño: Caso de uso: Gestionar datos de participantes en festival

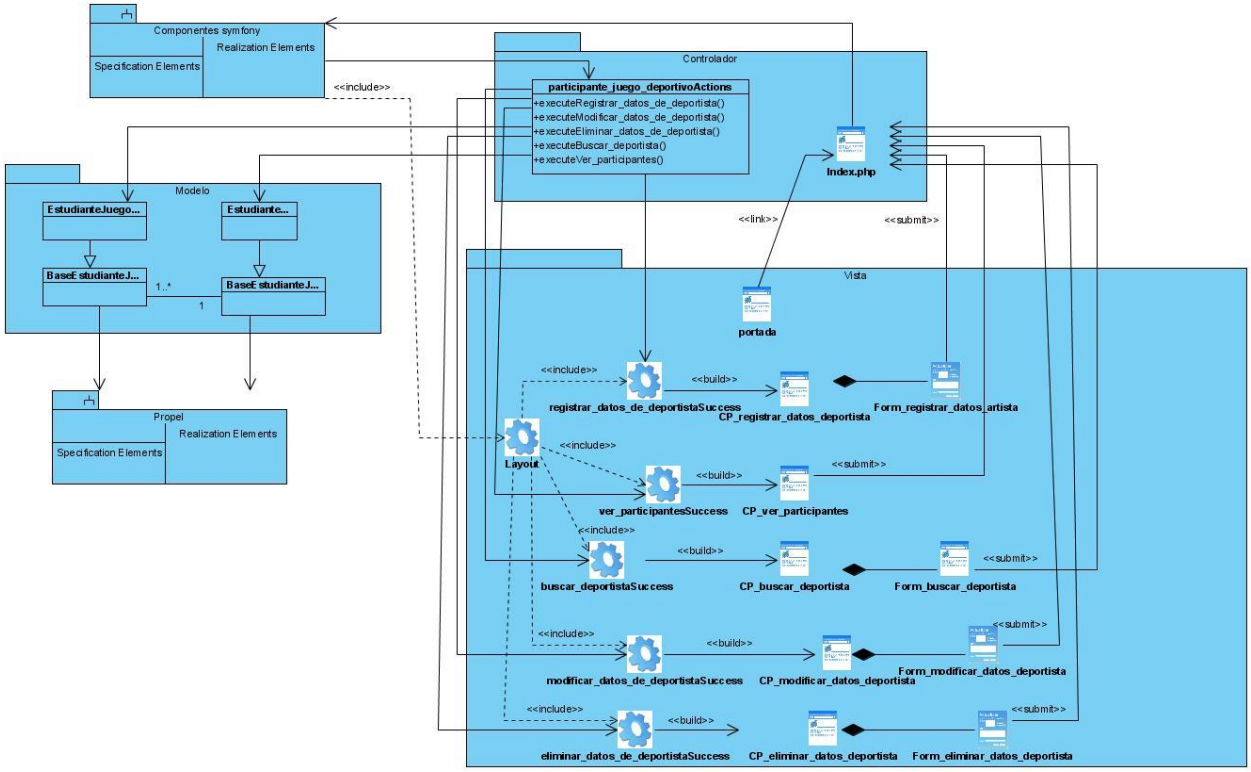


Fig. 7 Diagrama de clases del diseño: C aso de uso: Gestionar datos de participantes en juegos deportivos

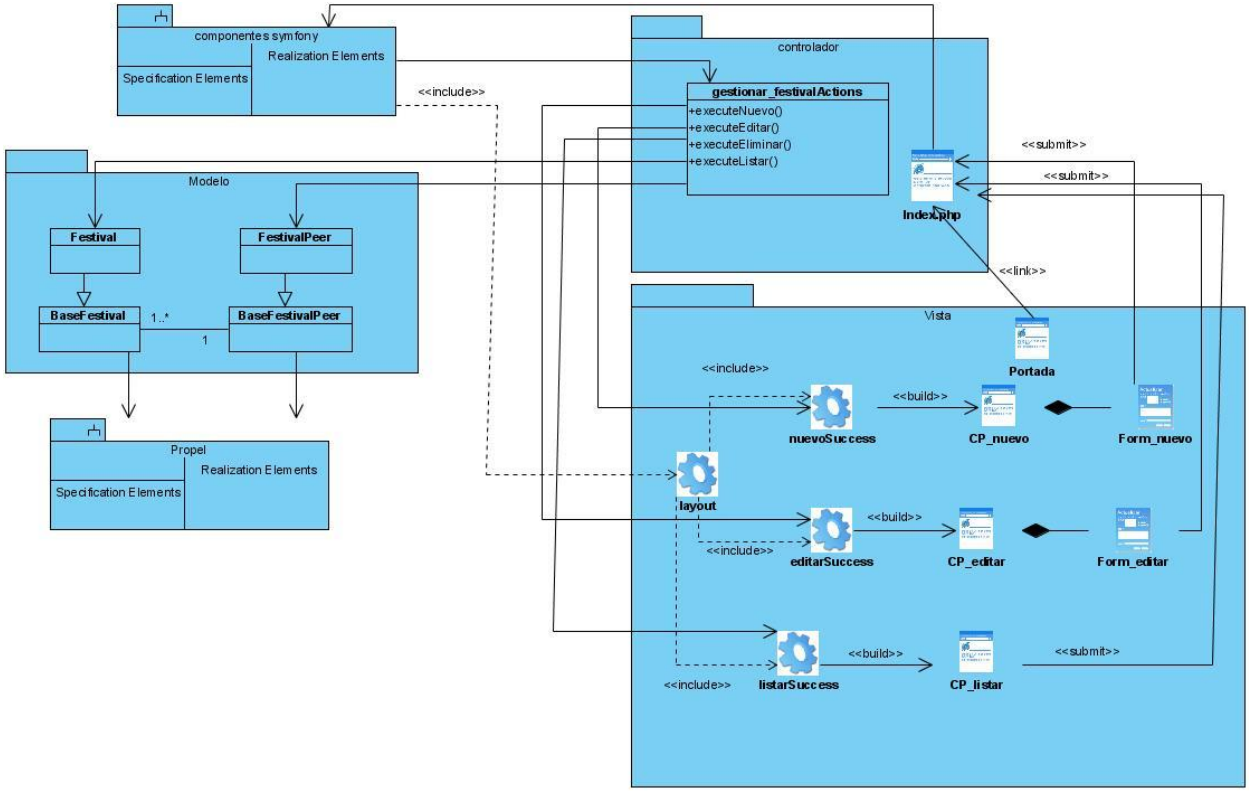


Fig. 8 Diagrama de clases del diseño: Caso de uso: Gestionar datos de festival

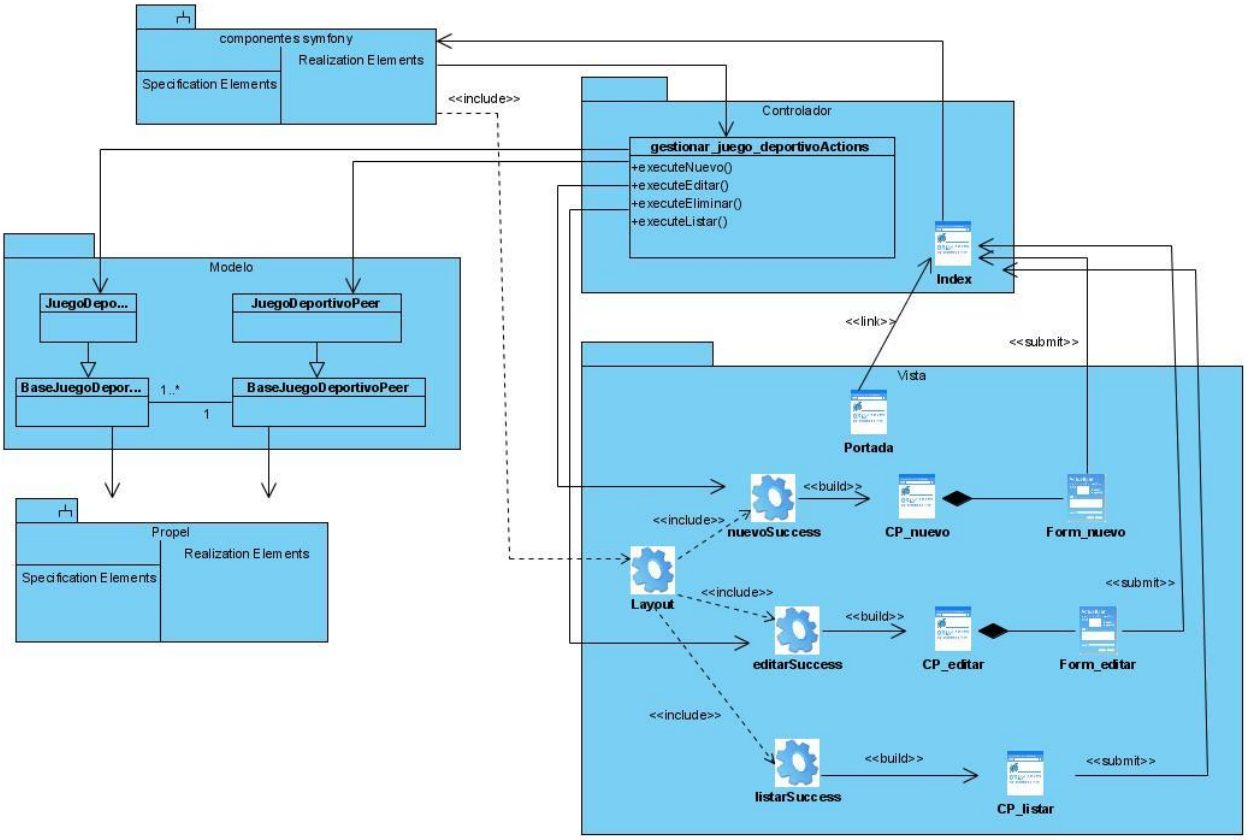


Fig. 9 Diagrama de clases del diseño: Caso de uso: Gestionar datos de juegos deportivos

3.5 Descripción de las clases del diseño

Nombre de la clase :	**participante_festivalActions**
Nombre del método:	executeRegistrar_datos_de_artista()
Descripción :	Este método se encarga de registrar los datos de los artistas.
Nombre del método:	executeModificar_datos_de_artista()
Descripción :	Este método muestra un formulario para modificar los datos de los participantes.
Nombre del método:	executeVer_datos()
Descripción :	Este método muestra el formulario con los datos específicos del artista del cual se quiere ver sus datos.

Nombre del método:	executeVer_participantes()
Descripción :	Este método muestra los participantes en festival.
Nombre del método:	executeEliminar_datos_de_artista()
Descripción :	Este método muestra un formulario para eliminar los datos de los participantes.
Nombre de la clase :	**participante_juego_deportivoActions**
Nombre del método:	executeRegistrar_datos_de_deportista()
Descripción :	Este método muestra el formulario para registrar los datos de los deportistas.
Nombre del método:	executeModificar_datos_de_deportista()
Descripción :	Esta acción muestra un formulario para modificar los datos deseados.
Nombre del método:	executeEliminar_datos_de_deportista()
Descripción :	Este método dibuja el formulario para eliminar los datos del estudiante.
Nombre del método:	executeVer_datos()
Descripción :	Este método muestra el formulario con los datos del participante en los juegos deportivos.
Nombre del método:	executeVer_participantes()
Descripción :	Este método muestra un formulario los participantes en los juegos deportivos.
Nombre de la clase :	**gestionar_manifestacion_artisticaActions**
Nombre del método:	executeNueva()
Descripción :	Este método muestra el formulario para registrar una nueva manifestación.
Nombre del método:	executeEditar()
Descripción :	Este método dibuja el formulario para editar los campos de la manifestación seleccionada.
Nombre del método:	executeEliminar()
Descripción :	Este método elimina los datos de una manifestación

	seleccionada previamente.
Nombre del método:	executeVer_manifestaciones_con_sus_participantes()
Descripción :	Este método muestra un formulario con las manifestaciones y sus participantes respectivamente.
Nombre de la clase :	**gestionar_numero_artisticoActions**
Nombre del método:	executeNuevo()
Descripción :	Este método permite registrar un nuevo número artístico.
Nombre del método:	executeEditar()
Descripción :	Este método una vez seleccionado el número artístico al que se le quiera hacer algún cambio, muestra un formulario para modificar sus datos.
Nombre del método:	executeEliminar()
Descripción :	Este método elimina los datos del número artístico que se seleccione.
Nombre del método:	executeBuscarParaAsignar()
Descripción :	Este método busca al participante que se le quiere asignar a los números artísticos registrados hasta el momento.
Nombre del método:	executeAsignar_participante()
Descripción :	Este método asigna al participante a un número artístico escogido.
Nombre de la clase :	**gestionar_ensayoActions**
Nombre del método:	executeNuevo()
Descripción :	Este método dibuja el formulario que permite registrar un nuevo horario de ensayo.
Nombre del método:	executeEditar()
Descripción :	Este método permite modificar los datos del horario seleccionado.
Nombre del método:	executeEliminar()

Descripción :	Este método elimina los datos del horario de ensayo seleccionado.
Nombre de la clase :	**gestionar_deporteActions**
Nombre del método:	executeNuevo()
Descripción :	Este método permite registrar un nuevo deporte.
Nombre del método:	executeEliminar()
Descripción :	Este método elimina los datos del deporte seleccionado.
Nombre del método:	executeEditar()
Descripción :	Este método permite modificar los datos del deporte seleccionado.
Nombre de la clase :	**gestionar_juego_deportivoActions**
Nombre del método:	executeNuevo()
Descripción :	Este método muestra el formulario para registrar un nuevo juego deportivo.
Nombre del método:	executeEditar()
Descripción :	Este método permite modificar los datos del juego deportivo seleccionado.
Nombre del método:	executeEliminar()
Descripción :	Este método elimina los datos del juego deportivo seleccionado.
Nombre del método:	executeVer_cronograma()
Descripción :	Este método muestra un formulario con los juegos deportivos y los cronogramas asociados a ellos.
Nombre del método:	executeVer_partes_diarios()
Descripción :	Este método muestra un formulario con los juegos deportivos y los partes diarios asociados a ellos
Nombre de la clase :	**gestionar_festivalActions**
Nombre del método:	executeNuevo()
Descripción :	Este método muestra un formulario que permite

	registrar un nuevo festival.
Nombre del método:	executeEditar()
Descripción :	Este método permite modificar los datos del festival seleccionado.
Nombre del método:	executeEliminar()
Descripción :	Este método elimina los datos del festival seleccionado.
Nombre del método:	executeVer_horario_de_ensayo()
Descripción :	Este método muestra un formulario con los festivales y los ensayos asociados a esos festivales.
Nombre de la clase :	**actividad_extensionistaActions**
Nombre del método:	executeNuevo()
Descripción :	Este método dibuja un formulario para crear una nueva actividad extensionista.
Nombre del método:	executeEditar()
Descripción :	Este método muestra un formulario que permite modificar los datos de la actividad.
Nombre del método:	executeEliminar()
Descripción :	Este método muestra un formulario para eliminar los datos del festival.
Nombre de la clase :	**cronograma_juego_deportivoActions**
Nombre del método:	executeNuevo()
Descripción :	Este método muestra un formulario para crear un nuevo cronograma.
Nombre del método:	executeEditar()
Descripción :	Este método muestra un formulario para editar los datos del cronograma.
Nombre del método:	ExecuteEliminar()
Descripción :	Este método dibuja un formulario con los cronogramas de los juegos deportivos que se quieren

	eliminar

3.6 Diagramas de interacción

El diagrama de secuencia es uno de los diagramas más efectivos para modelar interacción entre objetos en un sistema. Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase. Mientras que el diagrama de casos de uso permite el modelado de una vista *business* del escenario, el diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes pasados entre los objetos. Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.

A continuación se presenta el escenario Registrar datos de participante en festival correspondiente al CU Gestionar datos de participante en festival.

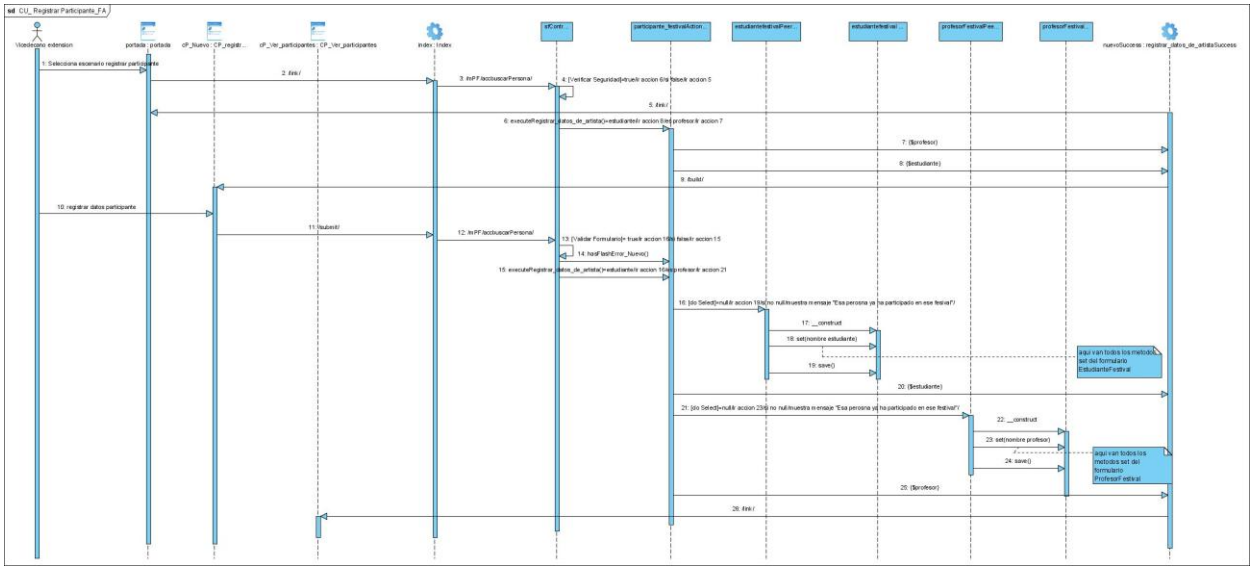


Fig. 10 Diagrama de secuencia: Caso de uso: Gestionar datos de participante en festival: Escenario: Registrar datos de participante en festival

A continuación se presenta el Registrar festival escenario correspondiente al CU Gestionar festival.

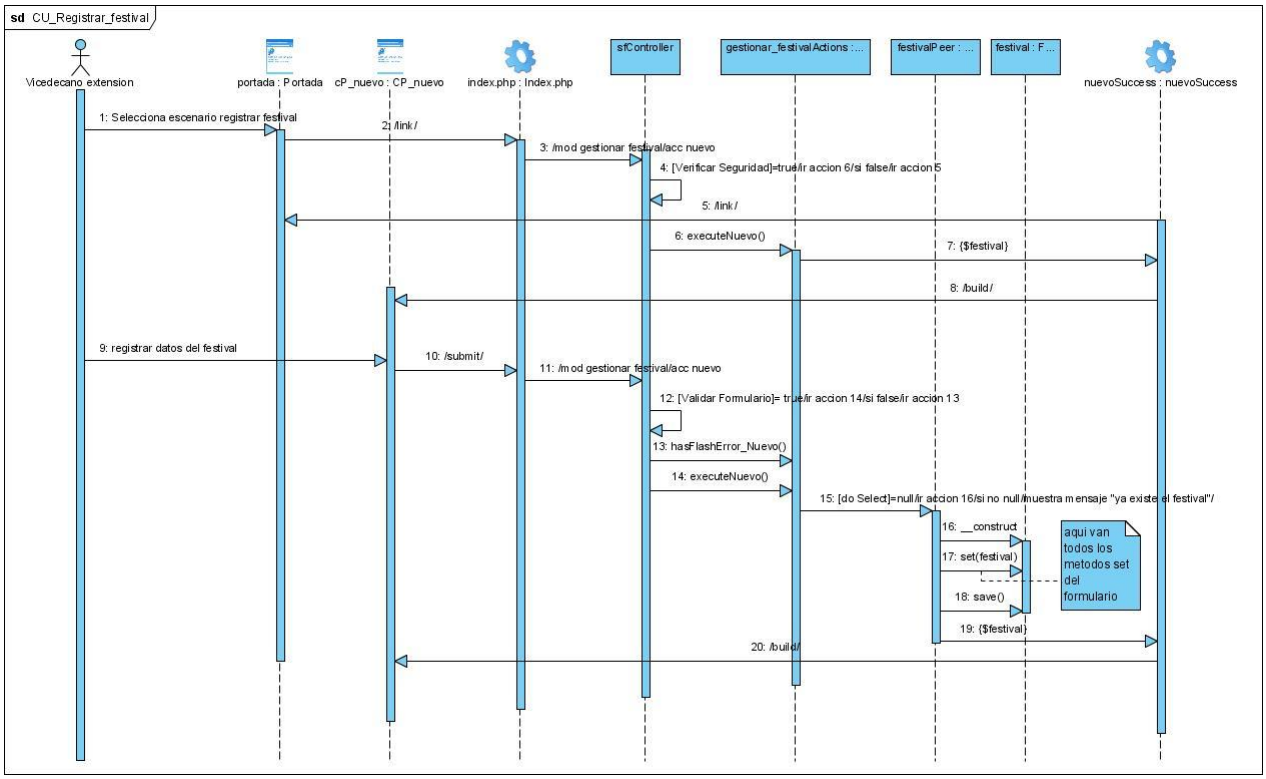


Fig. 11 Diagrama de secuencia: Caso de uso: Gestionar festival: Escenario: Registrar festival

A continuación se presenta el escenario Registrar participantes en los juegos deportivos correspondiente al CU Gestionar datos de participantes en los juegos deportivos.

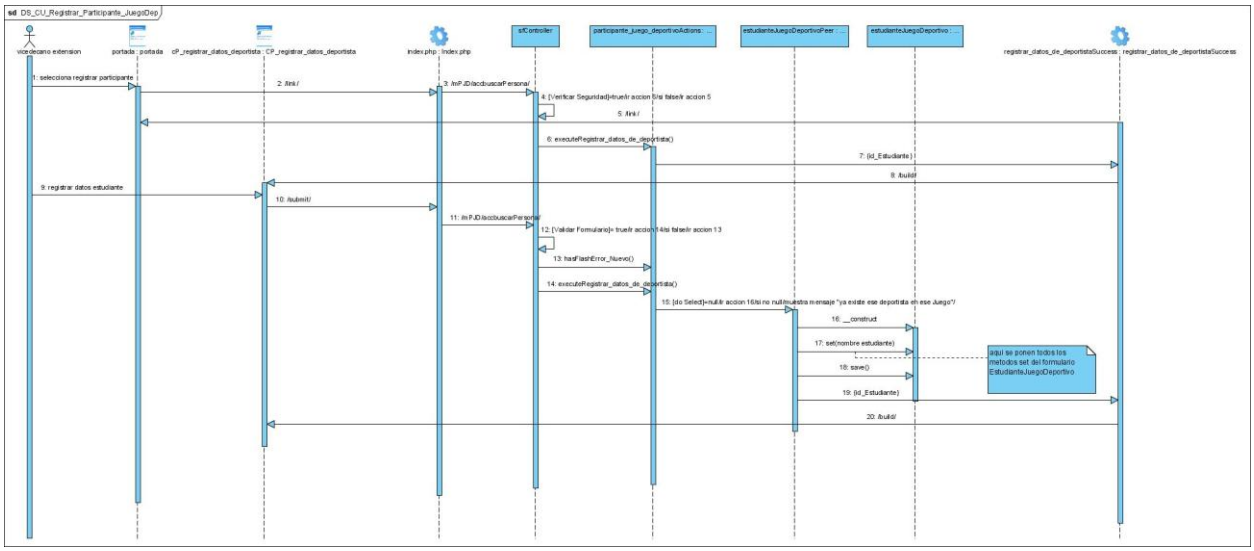


Fig. 12 Diagrama de secuencia Caso de uso: Gestionar datos de participantes en juegos deportivos: Escenario: Registrar participante en juegos deportivos

A continuación se presenta el escenario Registrar juegos deportivos correspondiente al CU Gestionar juegos deportivos.

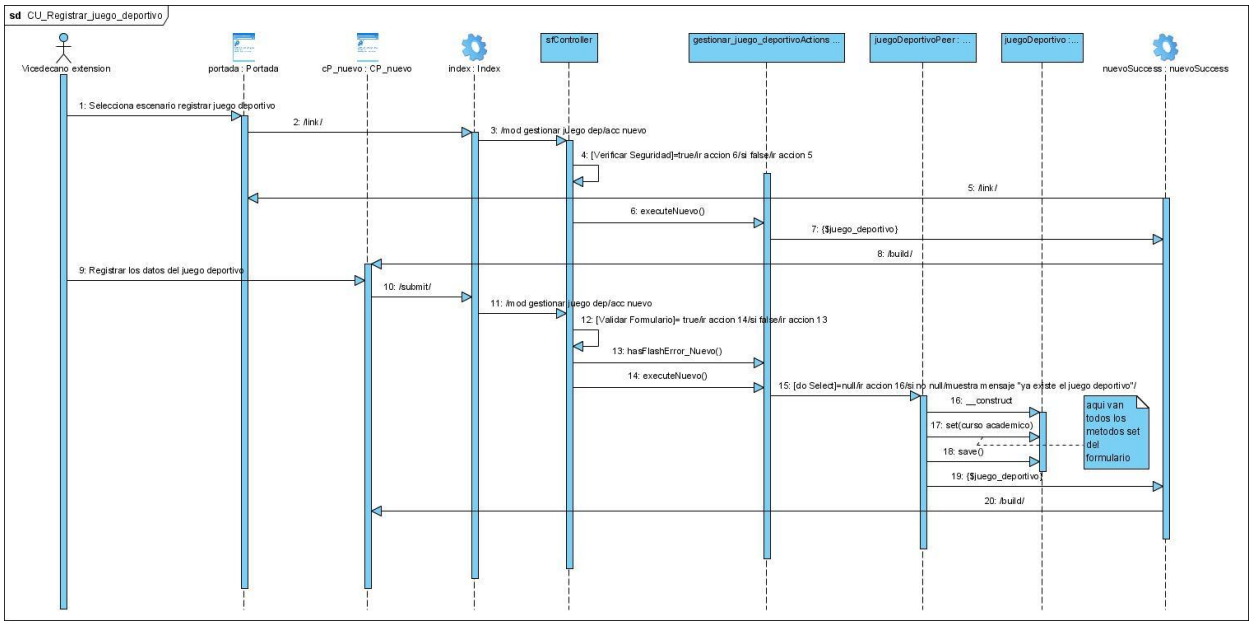


Fig. 13 Diagrama de secuencia: Caso de uso: Gestionar juegos deportivos: Escenario: Registrar juegos deportivos

3.7 Clases Persistentes. Diagrama de Clases Persistentes.

La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Por lo general las clases persistentes tienen como origen las clases clasificadas como entidad porque ellas modelan la información del sistema y el comportamiento asociado de algún fenómeno o concepto.

El diagrama de clases describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. A continuación se muestra el diagrama de clases persistentes.

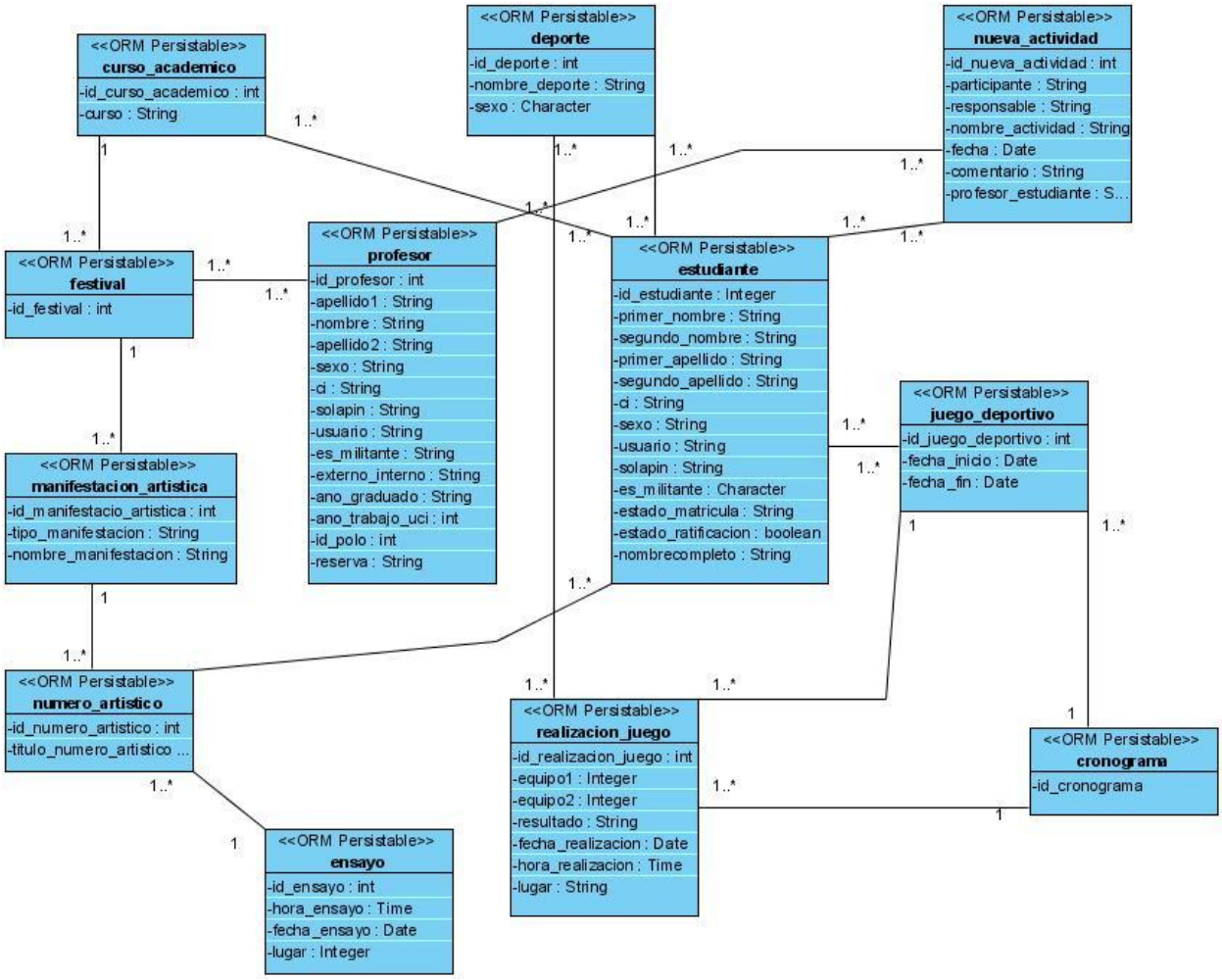


fig. 14 Diagrama de clases persistentes

3.8 Modelo de datos

El modelo de datos se describe la representación lógica y física de la información persistente del sistema. A continuación se muestra el modelo de datos.

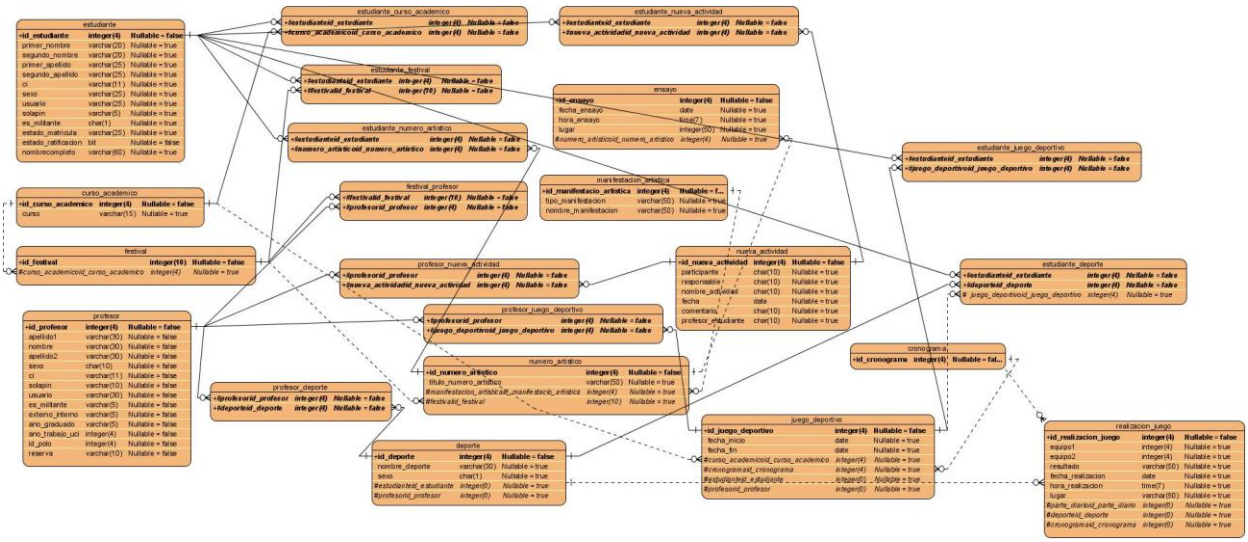


Fig. 15 Modelo de datos

3.9 Diagrama de Despliegue

Un diagrama de Despliegue muestra cómo y dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos, y la construcción interna puede ser representada por nodos o artefactos embebidos. Como los artefactos se ubican en los nodos para modelar el despliegue del sistema, la ubicación es guiada por el uso de las especificaciones de despliegue. A continuación se muestra el diagrama de despliegue correspondiente al sistema de informatización de la facultad 6, para una mejor descripción del mismo consultar la referencia al módulo de arquitectura.

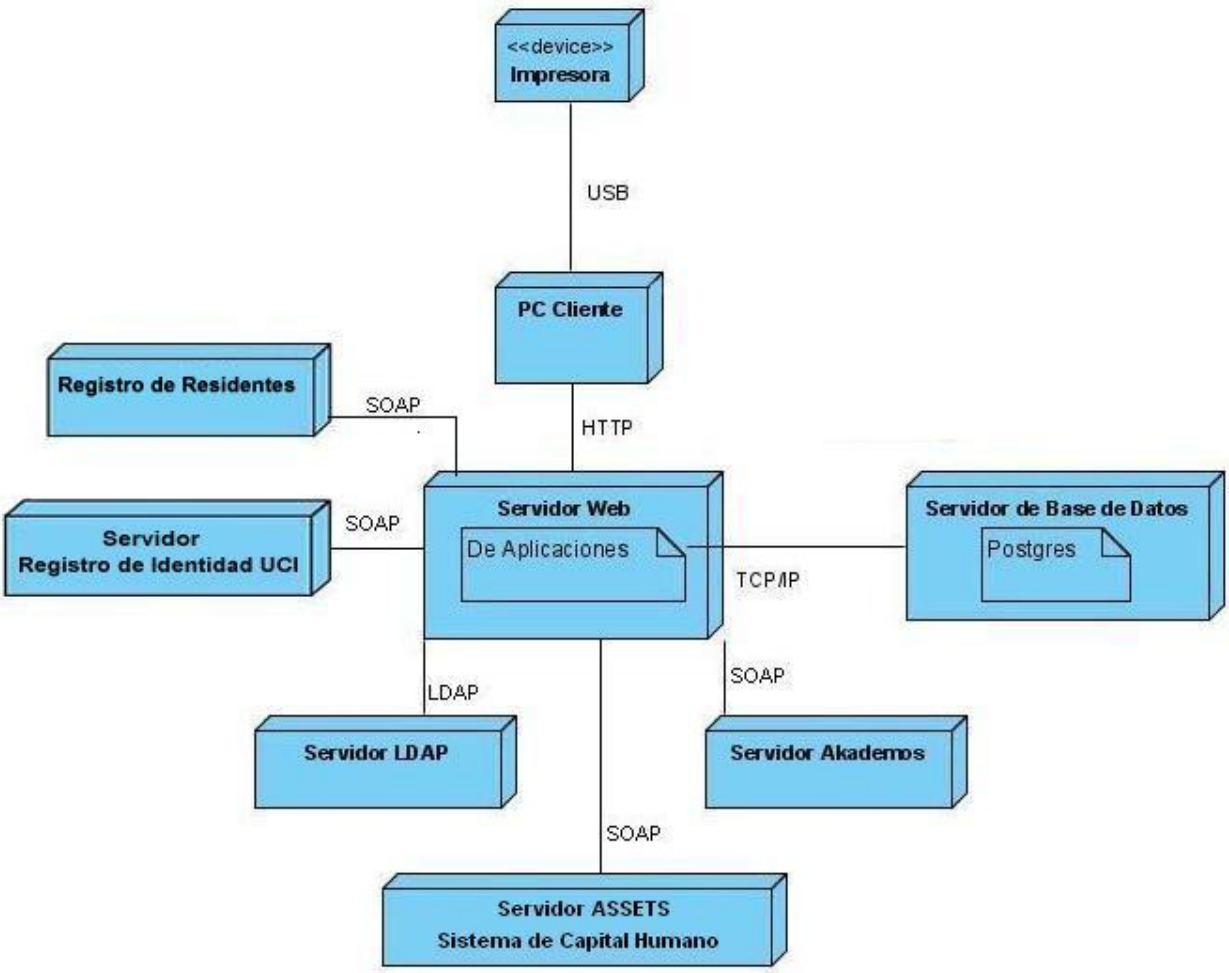


Fig. 16 Diagrama de despliegue

Conclusiones

En este capítulo se realizó los diagramas de diseño correspondientes a los casos de usos identificados así como los diagramas de secuencia. Se hizo una referencia a la vista lógica y de despliegue, se mostró el diagrama de clases persistentes y el modelo de datos, se describen las clases más importantes, así como sus atributos y tipos de datos.

CAPÍTULO 4: IMPLEMENTACIÓN DEL SISTEMA

Introducción

En el presente capítulo se modelan los artefactos correspondientes al flujo de trabajo de implementación. Se muestran fragmentos de código de los principales componentes y la descripción de su funcionamiento.

4.1 Diagramas de componentes

Los diagramas de componentes muestran los componentes *software* que constituyen una parte reusable, sus interfaces y sus interrelaciones, en muchos aspectos se puede considerar que un diagrama de componentes es un diagrama de clases a gran escala.

En los diagramas de componentes realizados, se muestran como están distribuidos según el patrón arquitectónico Modelo-vista-Controlador que utiliza Symfony como paradigma en su organización interna.

El componente sfFrontWebController o Controlador Frontal maneja todas las peticiones web, siendo el punto de entrada de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL escrita o pinchada por el usuario. El controlador frontal se encarga de despachar las peticiones, lo que implica algo más que detectar la acción que se ejecuta. De hecho, ejecuta el código común a todas las acciones. En pocas palabras el controlador frontal es el encargado de determinar qué combinación de módulo-acción se ejecutará. El paquete Vista se encarga de producir las páginas que se muestran como resultado de las acciones que se soliciten, las cuales se integran con el layout. En el paquete controlador se representan todas las acciones, estas actions.php están relacionadas con todos los archivos Success.php de la vista que contiene el código que liga la lógica de negocio con la presentación. Al acceder a las diferentes acciones lo primero que se hace es verificar si el usuario está autenticado y tiene los permisos correspondientes a la acción que desea realizar. La seguridad está dada por los módulos del sfGuardPlugin: sfGuardAuth encargado de gestionar la autenticación, sfGuardUser que gestiona los diferentes usuarios creados, sfGuardPermission que gestiona los permisos a cada usuario y sfGuardGroup gestiona todos los grupos de usuarios existentes. Dichos módulos se agrupan en el paquete sfGuard.

Los formularios en Symfony están compuestos por Widgets (campos) y Validators (validadores), cada

campo debe tener asignado un validador para comprobar los datos que vienen de las vistas y proteger de ataques al sistema. En el paquete Validators están recogidos las clases que permiten las validaciones de los formularios para cada tipo de datos. El modelo es la capa que contiene las clases: php, las Peer, las Bases y las Base Peer. Estas clases son construidas por el subsistema Propel de Symfony para el acceso a datos, permitiendo el acceso a la base de datos SIGIPE mediante el mapeo de objetos a bases de datos.

A continuación se muestran los diagramas de componentes correspondientes a algunos de los casos de uso.

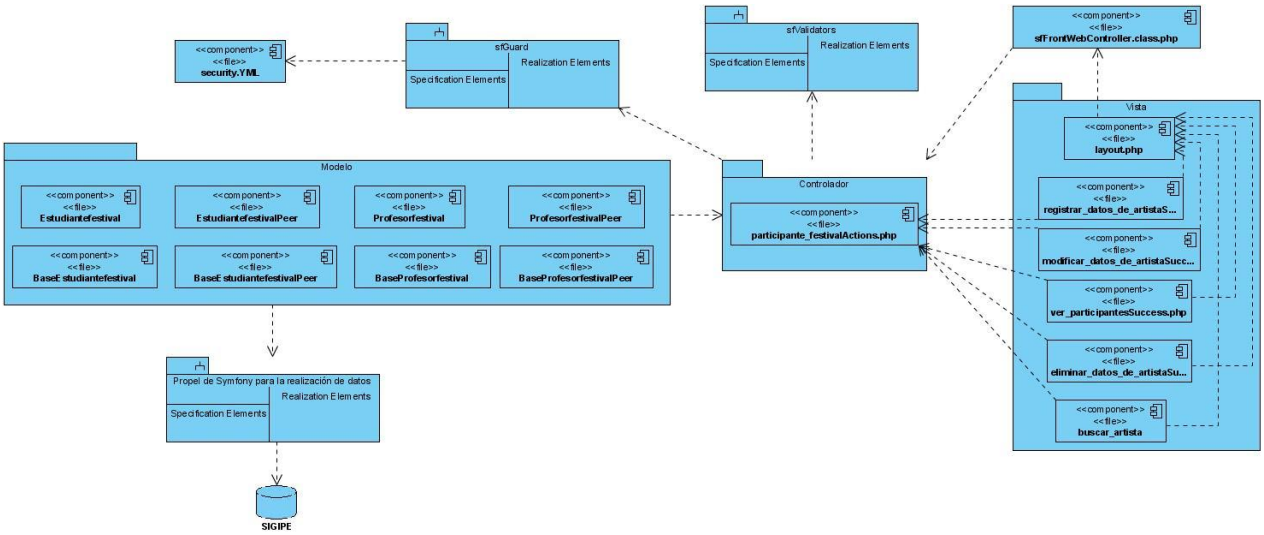


Fig. 17 Diagrama de componentes: Caso de uso: Gestionar datos de participantes en festival

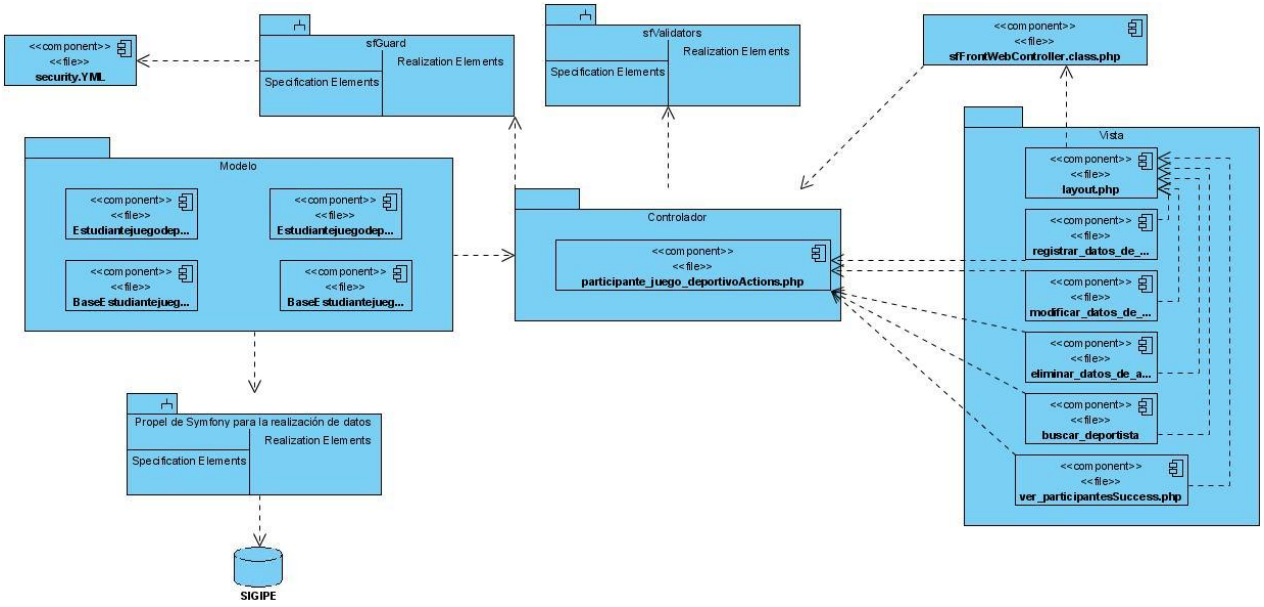


Fig. 18 Diagrama de componentes: Caso de uso: Gestionar datos de participantes en juegos deportivos

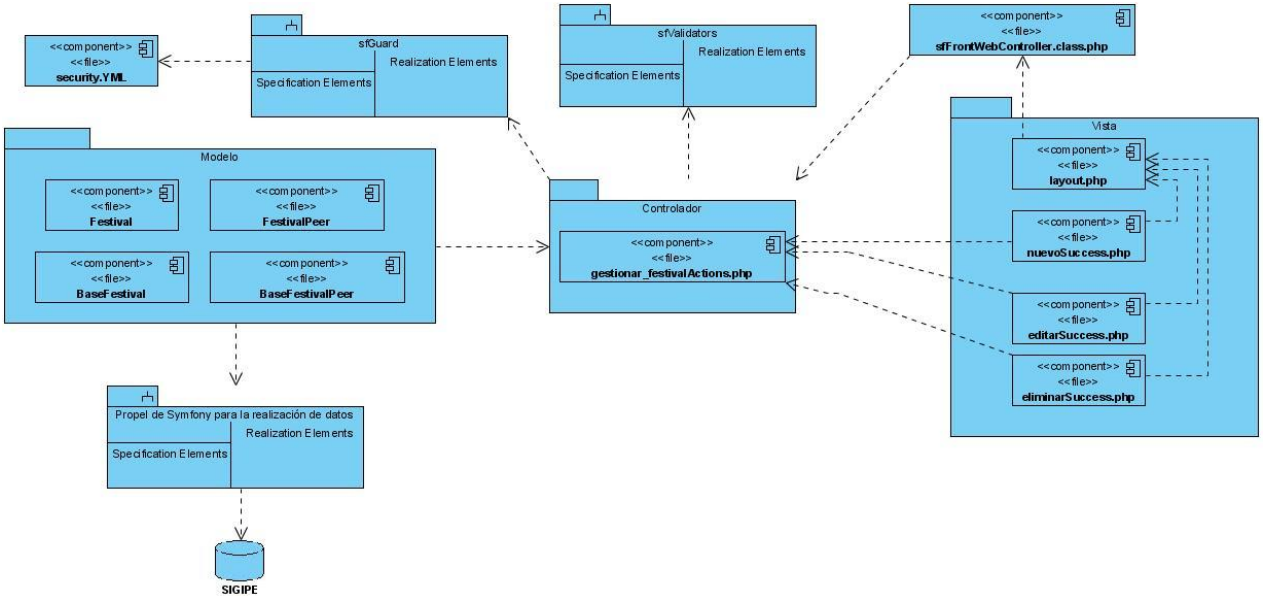


Fig. 19 Diagrama de componentes: Caso de uso: Gestionar datos de festival

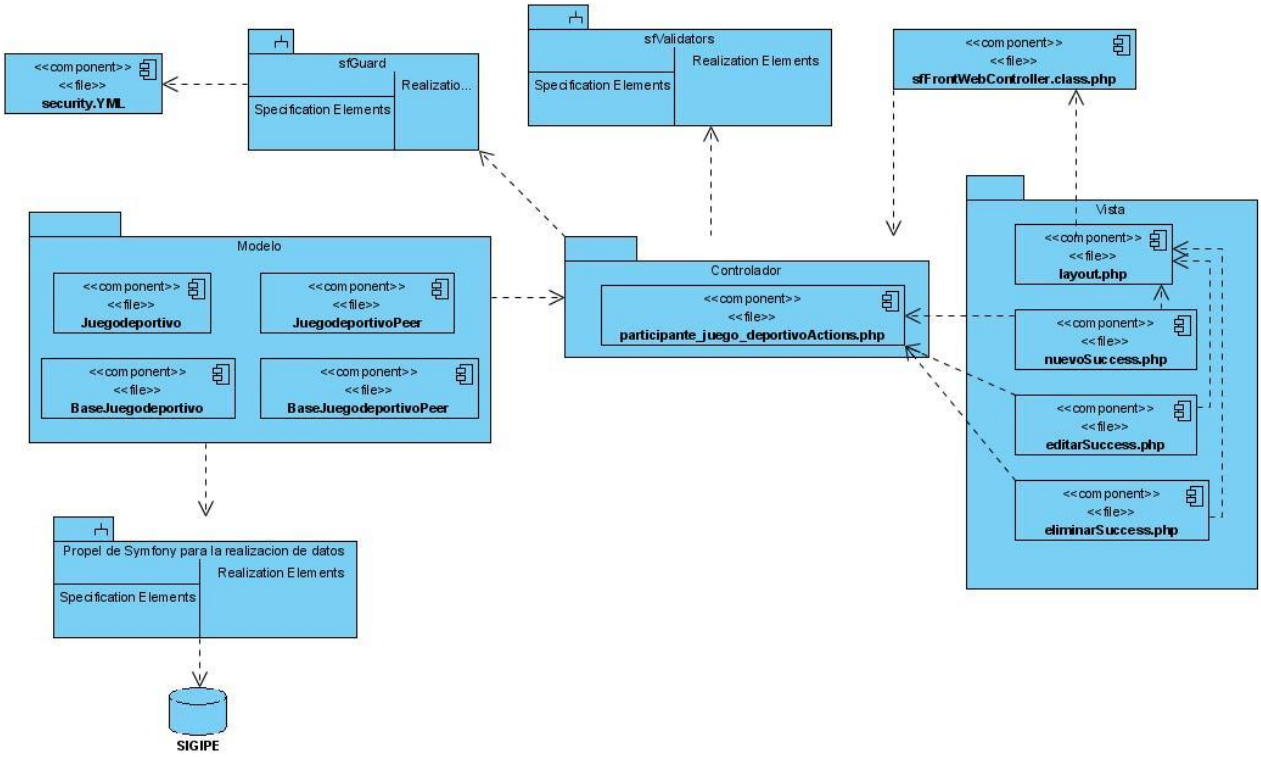


Fig. 20 Diagrama de componentes: Caso de uso: Gestionar datos de juegos deportivos

4.2 Código fuente de las principales clases

A continuación se muestran ejemplos de código perteneciente a la clase participante_festivalActions.

```
1  <?php
2
4⊕  * participante_festival actions.
11⊖ class participante_festivalActions extends sfActions
12 {
       public function executeRegistrar_datos_de_artista(sfWebRequest $request)
       {
           $this->solapinPersona = $request->getParameter('idPersona');

           $cEst = new Criteria();
           $cEst->add(EstudiantePeer::SOLAPIN,$this->solapinPersona);

           $cProfe = new Criteria();
           $cProfe->add(ProfesorPeer::SOLAPIN,$this->solapinPersona);

           $estudiante = EstudiantePeer::doSelectOne($cEst);
           $profesor = ProfesorPeer::doSelectOne($cProfe);

           if($estudiante instanceof  Estudiante)
           {
               $this->artista = $estudiante;
           }
           else $this->artista = $profesor;

           $this->festivales = FestivalPeer::doSelect(new Criteria());
           $this->numeros_artisticos = NumeroArtisticoPeer::doSelect(new Criteria());
       }
```

Fig. 21 Código fuente de la clase participante_festivalActions

Esta acción muestra un formulario para registrarle una participación en festival a una persona que puede ser estudiante o profesor para esto primeramente se obtiene el identificador de la persona (el número de su solapín) y se guarda en la variable solapinPersona, en las variables estudiante y profesor se tienen los datos de los mismos seleccionados de la clase Peer, en dependencia de si es estudiante o profesor se le asigna estos datos a la variable artista, que será utilizada en el formulario, luego se procede a buscar esa persona en la base de datos, si se encuentra la persona se muestra el formulario con los festivales y los números artísticos existentes, si no se encuentra la persona se muestra un mensaje de error.

Guardar datos participante festival

```php
<?php

 * participante_festival actions.
class participante_festivalActions extends sfActions
{

  public function executeGuardarDatos(sfWebRequest $request)
  {
      $idPersona = $request->getParameter('idPersona');
      $idFestival = $request->getParameter('festivales');
      $idNumArt = $request->getParameter('numeros_artisticos');
      $tieneLicencia = $request->getParameter('licencia_artistica')== 1 ? true:false;

      $festival = FestivalPeer::retrieveByPK($idFestival);
      $numArt = NumeroArtisticoPeer::retrieveByPK($idNumArt);
      $artista = GestionarArtistaFestival::getPersonaBySolapin($idPersona);

      try
      {
          GestionarArtistaFestival::AdicionarArtista($artista,$festival,$numArt,$tieneLicencia);
          $this->redirect('participante_festival/ver_participantes');
      }
      catch(Exception $e)
      {
          $this->error = $e->getMessage();
      }
  }
```

Fig. 22 Código fuente Código fuente de la clase participante_festivalActions

En este método una vez enviados los datos del formulario registrar datos artista, se reciben los datos id_persona, id_festival, id_numero_artistico, y si es un estudiante el campo tiene licencia, luego se buscan los objetos asociados a esos identificadores mediante los métodos retriveBYPK de las clases Peer asociadas a los datos en cuestión, después se llama al método AdicionarArtista de la clase GestionarArtistaFestival pasándole por parámetro los objetos creados anteriormente, si el método lanzó una excepción se captura y se muestra un mensaje de error, sino se redirecciona a la página ver_participantes del módulo participante festival.

4.3 Validación de la aplicación

A continuación se representa una forma de verificar la calidad del software a través de la validación de la entrada de datos obligatorios.

Conclusiones

En este capítulo se realizaron los diagramas de componentes referentes a cada caso de uso, se muestran ejemplos de código fuente así como una breve descripción de los mismos.

CONCLUSIONES GENERALES

Con este trabajo se presenta una aplicación Web para la gestión de la información extensionista en la facultad 6. Para ello se modeló el entorno de negocio a través del modelo de dominio. Se identificaron las funcionalidades de la aplicación Web levantando posteriormente los requisitos necesarios. Se realizó el diseño de la aplicación. Se implementaron todas las funcionalidades. Por lo que se puede concluir que los objetivos propuestos al inicio de este trabajo fueron cumplidos en su totalidad. Se propone una solución que apoya al proceso de gestión de la información de extensión universitaria en la facultad 6.

RECOMENDACIONES

De forma general los objetivos propuestos al inicio de este trabajo fueron cumplidos, no obstante durante el transcurso de su desarrollo, han surgido una serie de ideas y recomendaciones que podrían implementarse en futuras versiones de la solución, de manera que pueda lograrse una aplicación más útil y completa, para lo cual se recomienda:

1. Continuar el mejoramiento de este sistema, adicionándole nuevas funcionalidades, de acuerdo a las necesidades de los clientes.

2. Extender el sistema de manera que pueda ser utilizado no sólo en la facultad, sino en la universidad.

REFERENCIAS BIBLIOGRÁFICAS

1. **Jacobson, I. and G. Booch**, *El Proceso Unificado De Desarrollo De Software*. 2000.

2. *www.seamframework.org*. [En línea] 2008. [Citado el: Febrero de 10 de 2008.] from:http://www.seamframework.org/.

3. **Holzner, S.**, *PHP 5*. 2005.

4. **UCI and D. Programación**. *Conferencia1: Arquitectura de un sistema de bases de datos*. 2008.

6. **Larman, C.**, *UML y Patrones*. 1999.

7. **UCI and D.I.d. Software.** *Profundización del flujo de trabajo de requerimientos.* . in *Conferencia 5*. 2008.

8. **UCI and D.I. Software**. *Conferencia 2: Arquitectura y Patrones de diseño*. 2008.

9. *Tendencias Arquitectonicas.* **Software), UCI(Departamento Ingenieria.** 2009.

10. ***merinde.rinde.gob.ve****. [En línea] 2009. [Citado el: 1 de mayo de 2009.] http://merinde.rinde.gob.ve.*

11. Sistema para la Gestión de la Información de. **Duanis, Sotolongo y Esley, Valdez.** Habana : s.n., 2009.

BIBLIOGRAFÍA

1. **UCI and D. Programacion**. *Conferencia1: Arquitectura de un sistema de bases de datos*. 2008.

2. **Rojas, T., et al.**, *Modelo de decisión para soportar la selección de herramientas CASE*. 2000. p. 117-144.

3. **Larman, C.**, *UML y Patrones*. 1999.

4. **UCI and D.I.d. Software.** *Profundización del flujo de trabajo de requerimientos.* . in *Conferencia 5*. 2008.

5. **UCI and D.I. Software**. *Conferencia 2: Arquitectura y Patrones de diseño*. 2008.

6. **Prieto, F., et al.**, *Construcción de frameworks basada en análisis de conceptos formales y soportada por Mecanos*. 2000. p. 3647.

7. **Ortega, I.S**. (2005) *Curso de PHP 5*. **Volume***,*

8. **Zaninotto, F**., *Learn symfony*. 2007.

9. **Lellelid, H.**, *Propel - Guía de usuario.* 2004.

10. **Potencier, F. and F. Zaninotto**, *Symfony la guía definitiva.* 2008.

11. **Prieto, F., et al.**, *Mecanos y análisis de conceptos formales como soporte para la construcción de frameworks*. 2000. p. 163–175.

12. **Gallardo, D., E. Burnette, and R. MacGovern**, *Eclipse in Action.* 2003.

13. **Vaswani, V.**, *PHP and PostGreSQL.* 2002.

14. **Booch, G.**, *the Unified Modeling Language.* 2000.

15. **Nagel, W.**, *Using the Subversion Version Control System in Development Projects.* 2005.

16. **Dupré, C.**, *Source Code Revision Control with Subversion.* 2005.

17. **Garrett, J.J.**, *Ajax: A new approach to web applications*. 2005, v.

18. **Gutiérrez, J.D**., *Desarrollo Web con PHP 5 y MySQL*. 2004.

19 **Lerdorf, R.J., et al.**, *Programming Php*. 2002: O'Reilly & Associates, Inc. Sebastopol, CA, USA.

20. *www.symfony-project.org*. 2008 [cited; Available from: http://www.symfony-project.org/.

21. *www.eclipse.org*. 2004 [cited; Available from: http://www.eclipse.org.

22. *www.extension.unc.edu.ar*. 2003 [cited; Available from:http://www.extension.unc.edu.ar/.

23 *www.visual-paradigm.com*. 2008 [cited; Available from: http://www.visual-paradigm.com/.

24. http://redalyc.uaemex.mx/redalyc/pdf/615/61570402.pdf

ANEXOS

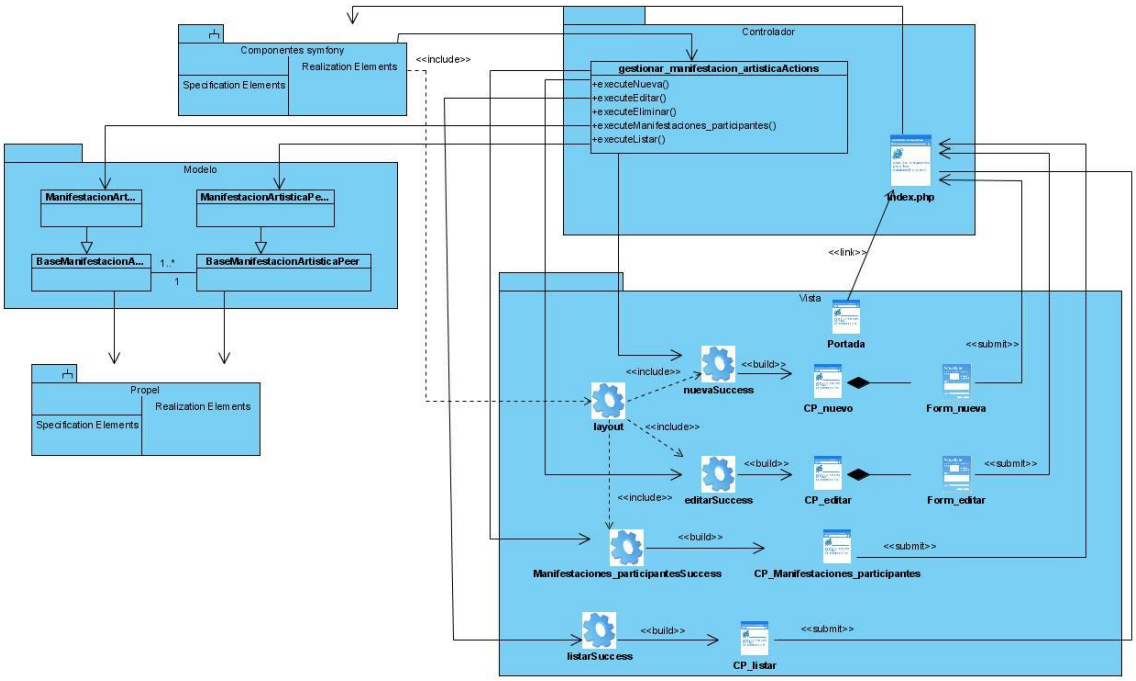


Fig. 23 Diagrama de clases del diseño: Caso de uso: Gestionar manifestación cultural.

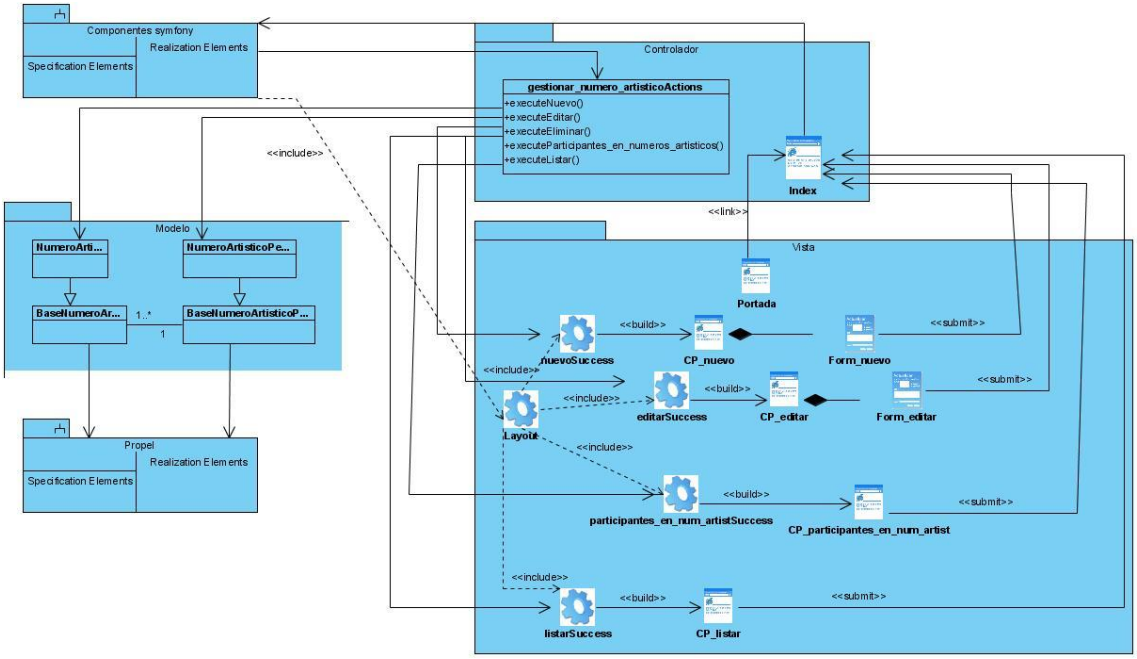


Fig. 24 Diagrama de clases del diseño: Caso de uso: Gestionar número artístico

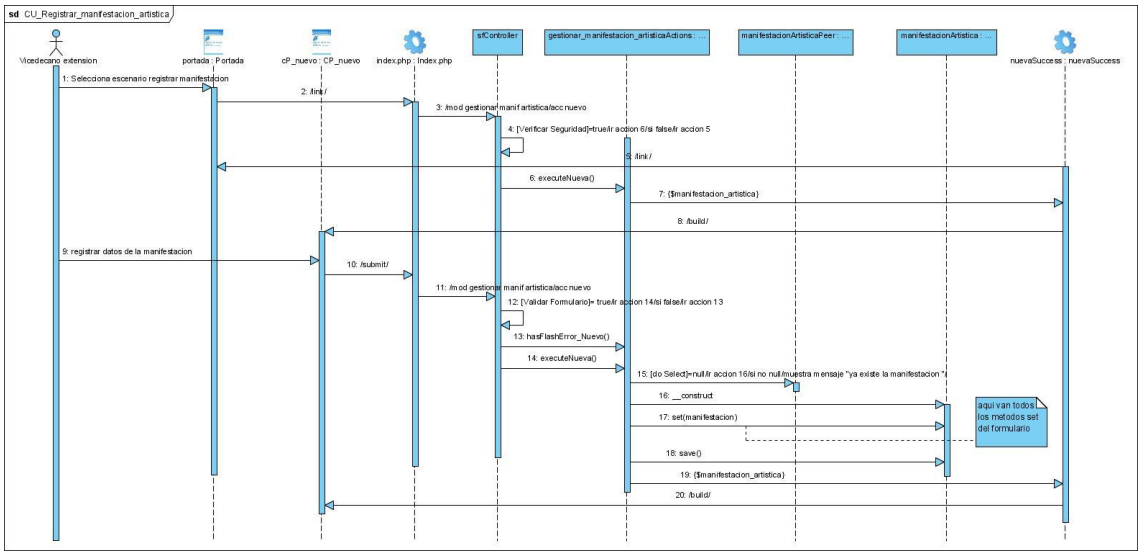


Fig. 25 Diagrama de secuencia: Caso de uso: Gestionar manifestación cultural: Escenario Registrar manifestación cultural

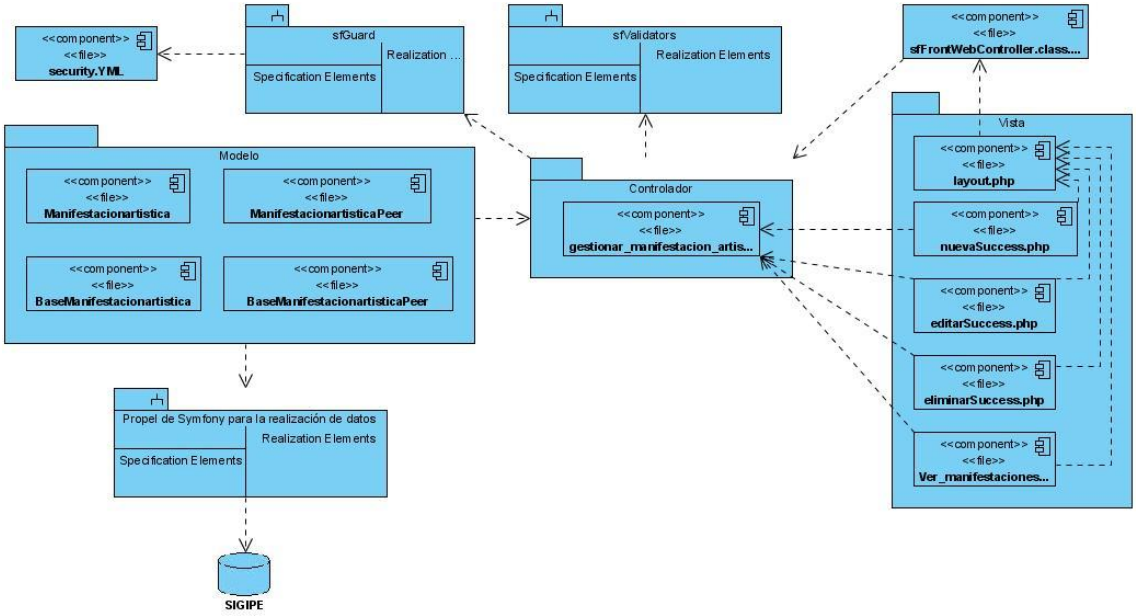


Fig. 25 Diagrama de componentes: Caso de uso: Gestionar manifestación cultural

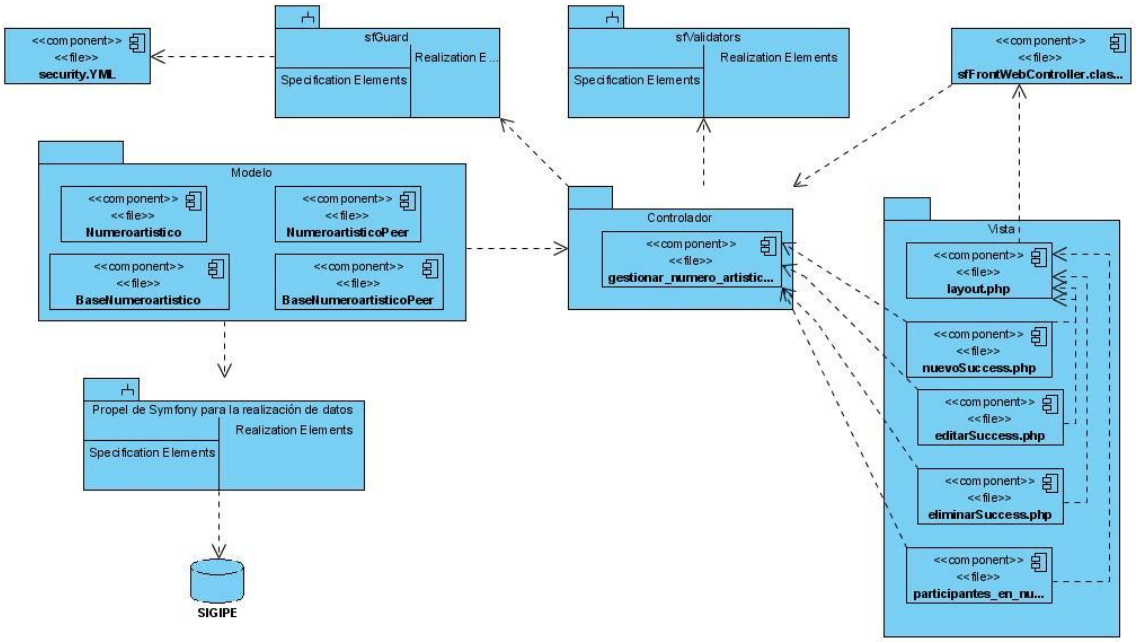


Fig. 26 Diagrama de componentes: Caso de uso: Gestionar número artístico

Fig. 27 Prototipo no funcional: Buscar personas

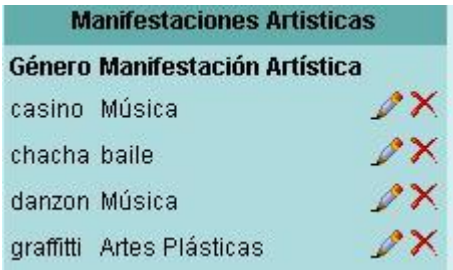


Fig. 28 Prototipo no funcional: Listado de manifestaciones artísticas

Fig. 29 Prototipo no funcional: Listado de participantes en festival

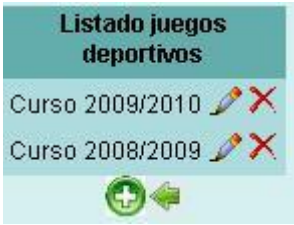


Fig. 30 Prototipo no funcional: Listado de juegos deportivos

GLOSARIO

A continuación, en orden alfabético, se muestra el significado de algunos términos usados en este documento que pueden dificultar la compresión del mismo:

ADO: Mecanismos que usan los programas de computadoras para comunicarse con las bases de datos. Sus siglas vienen dadas por su nombre en inglés ActiveX Data Objects.

AJAX: Es una técnica de desarrollo para crear aplicaciones Web, mediante la cual un grupo de acciones se ejecutan en navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano.

CUS: caso de usos del sistema.

CRUD: es el acrónimo de **Create Read Update Delete** (Crear, Leer, Actualizar, Borrar) usado para referirse a las funciones básicas en Base de Datos o en la capa de persistencia de un sistema Software.

DCD: diagrama de clases del diseño.

DS: diagrama de secuencia.

ECLIPSE: es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores.

Framework: Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

GOF: Expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas software, sus siglas vienen dadas por su nombre en inglés Gang of Four.

GRASP: Patrones generales de software para asignación de responsabilidades, sus siglas vienen dadas por su nombre en inglés General Responsibility Assignment Software Patterns.

Herramienta CASE: Aplicación informática destinada a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero, se utiliza para la modelación del sistema.

HTML: HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web.

HTTP*:* Protocolo de transferencia de hipertexto cuyas siglas vienen dadas por su nombre en inglés HyperText Transfer Protocol, es el protocolo usado en cada transacción de la Web.

IDE: Entorno Integrado de Desarrollo, es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse exclusivamente para un lenguaje de programación o bien para varios (6)

MVC: Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones Web.

PHP: Hypertext Pre-processo, es un lenguaje interpretado de propósito general ampliamente usado y que está diseñado especialmente para desarrollo Web y puede ser incrustado dentro de código HTML.

RUP: Proceso Unificado de Racional, es un proceso de desarrollo de software, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

UML: Lenguaje visual para especificar, construir y documentar un software. Sus siglas vienen dadas por su nombre en inglés Unified Modeling Language.

USB: Universal Serial Bus o Conductor Universal en Serie, es un puerto que sirve para conectar periféricos a una computadora.

XML: Extensible Markup Language («lenguaje de marcas ampliable»), es un metalenguaje extensible de etiquetas.