Universidad de las Ciencias Informáticas

Facultad 2

Sistema dataFEU v2.0

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores:

Sandra Rodríguez Angel

Reynier Arias Lemos

Tutor:

Ing. Victor Gabriel González Cardoso

Ciudad de La Habana, Abril de 2016

Año 58 de la Revolución

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los __ del mes de _____ del año 2016

_____ _____

Sandra Rodríguez Angel Reynier Arias Lemos

_____

Ing. Victor Gabriel González Cardoso

AGRADECIMIENTOS

A mis padres y a mi hermana por estar ahí en los malos y buenos momentos dándome siempre todo su cariño.

A mi familia habanera, una familia muy grande y unida que me acogió como una hija más, de ellos he aprendido muchísimo, los adoro.

A mis compañeros y amistades de la carrera con los que compartí muy buenos y divertidos momentos.

A los muy buenos profesores que tuve en estos maravillosos 5 años de la carrera.

A mi tutor, Victor muchísimas gracias por toda tu ayuda y confianza en la realización de este trabajo.

A Rey por tocarle el duro trabajo de sobrellevarme como su pareja de tesis y como su pareja de la vida, sé que fue difícil pero lo logramos juntos.

A todas las personas que de una forma u otra me ayudaron y apoyaron durante la carrera.

Sandra.

AGRADECIMIENTOS

Me gustaría agradecer su colaboración a todas las personas que han ayudado de una forma u otra en la creación de este trabajo. Sin ningún orden en particular:

A mi padre que me ha hecho un hombre de bien, una vez más me siento orgulloso de ti.

A mi madre que me ha mimado cuando he cometido errores, gracias por brindarme tu apoyo.

A Orquidia, por aconsejarme y tenerme como otro hijo más.

A mis hermanos que hoy no han podido estar aquí, aunque sé que quisieran.

A mi familia de Matanzas, José, Livania y Laura, ojala hubiera sido su hijo varón.

A Victor por permitirnos entrar a la realización de este trabajo, por sus grandes aportes y su afán de enseñarnos.

A mi grupo de clases, ustedes han sido una historia diferente cada día en el aula.

A mi compañera de tesis y prometida Sandra, gracias por todo el sacrificio que has hecho por mí. Sé que será mucho lo que nos queda por andar pero si es como hasta ahora, quiero hacerlo a tu lado.

Especialmente a mis abuelos, que sin título universitario son decanos de la vida, gracias a ellos, hoy estoy aquí, ustedes fueron mis primeros padres.

Reynier.

DEDICATORIA

A mis padres que tanto se esforzaron y se sacrificaron para que llegara hasta aquí. A ustedes por todo el amor que me han dado, por sus consejos cuando más los necesitaba y por intentar guiarme siempre por el mejor camino.

Sandra.

A mis abuelos y mis padres, nunca podre separar el amor que siento por ustedes. Ustedes me han enseñado a ser un hombre de bien y a poner a la familia por encima de todo. Gracias por su perseverancia, sus consejos y su humildad, he aprendido mucho en las buenas y malas. Ustedes son mi gran familia, siempre estaremos juntos.

Reynier.

RESUMEN

En el año 2014 surge el proyecto MAYA: “Ecosistema de aplicaciones informáticas para la evaluación integral de los estudiantes en el contexto universitario”. Este proyecto pretende integrar en una plataforma unificada aplicaciones que tributen evidencias sobre el desempeño estudiantil y contribuir a mejorar el proceso de recopilación, procesamiento y persistencia de la información que se genera. dataFEU v1.0, es una aplicación web desarrollada para gestionar los principales procesos de la Federación Estudiantil Universitaria en la UCI, la cual recoge evidencias que tributan al proceso de evaluación estudiantil. Esta solución fue implantada en la UCI en el curso 2012-2013. La arquitectura y el modelo empleado en el desarrollo de la misma hacen muy engorrosa la integración hacia el ecosistema MAYA, definiéndose como objetivo de esta investigación desarrollar una nueva versión que solvente esta situación.

En el presente informe se muestran los resultados de la implementación de la versión 2.0 de dataFEU. Esta aplicación informática se desarrolló sobre los marcos de trabajo Symfony 2 empleando PostrgreSQL como servidor de base de datos, utilizando como ORM Doctrine 2.5 y como capa de presentación Boostrap 3, ajustándose a la soberanía tecnológica por la cual aboga Cuba. Cuenta con 8 módulos y brinda información a sistemas externos mediante servicios web que garantizan la interoperabilidad del mismo.

Palabras claves: dataFEU, integración, MAYA, Symfony2.

ABSTRACT

In 2014 the project MAYA (Ecosystem applications for students comprehensive assessment in the university context) was created. This project aims to integrate into a unified platform, and contribute to improving the collection process, processing and persistence of information generated. dataFEU v1.0 is a web application developed to manage the main processes of the FEU (Federación Estudiantil Universitaria, for its acronym in Spanish) in the UCI. This solution was implemented in the UCI in 2012-2013.The architecture and the model used in the development of it make it very cumbersome the integration into the MAYA ecosystem, defined as objective of this research: develop a new version that solves this situation.

In this report shown the results of the implementation of version 2.0 of dataFEU. This software application development on frameworks Symfony 2 using PostgreSQL as database server, Doctrine 2.5 ORM and as a presentation layer Bootstrap 3 adjusting to technological sovereignty by which advocates Cuba. It has 8 modules and provides information to external systems through web services that ensure interoperability of it.

Keywords: dataFEU, integration, MAYA, Symfony2.

ÍNDICE

INTRODUCCIÓN

El significativo avance de la Tecnología de la Información y la Comunicación (TIC) a nivel mundial ha propiciado el desarrollo informático de las organizaciones. En la actualidad la mayoría de las tareas de oficina son ejecutadas y controladas por sistemas de información con el objetivo de recolectar, procesar, almacenar y distribuir datos en apoyo al control y la toma de decisiones. Las organizaciones reconocen la ventaja que supone incorporar los nuevos avances tecnológicos a sus procesos de negocio, con el fin de obtener resultados eficientes. Es por ello que constituye una prioridad mejorar la administración y desempeño organizacional a través de la incorporación de la gestión con un enfoque basado en procesos (1).

Cuba no se encuentra aislada de este proceso de informatización donde las TIC se convierte en un sector de desarrollo estratégico para la nación potenciando una economía del conocimiento. Entre las instituciones educacionales más destacadas en el desarrollo y uso de sistemas informáticos se encuentra la Universidad de las Ciencias Informáticas (UCI) creada con el objetivo de formar profesionales altamente calificados en la rama de la informática y contribuir al desarrollo de la industria de software en el país (2). Como parte de su política de informatización universitaria, esta institución dedica recursos y esfuerzos en el desarrollo de aplicaciones para automatizar sus principales procesos.

dataFEU v1.0 es una de las soluciones informáticas desarrolladas para gestionar los principales procesos de la Federación Estudiantil Universitaria (FEU) en la UCI. La aplicación web se desplegó durante el curso 2011-2012, en la Facultad 3 y se extendió luego al resto de las facultades de la institución en el curso 2013-2014. Dicha aplicación fue implementada utilizando como capa de presentación extJS en su versión 3.3 y bajo el marco de trabajo Symfony1 (3). La versión empleada del framework fue Symfony 1.4 LTS (Long Time Support) con soporte hasta el año 2012, la cual no permite su evolución a versiones superiores debido a que su estructura, arquitectura y forma de organización cambió completamente hacia un enfoque diferente.

A raíz del dinamismo implícito en la propia universidad, se comenzó a desarrollar sobre esta plataforma módulos para cubrir las necesidades de dentro y fuera de la organización estudiantil. Esto provocó que se implementaran procesos que no correspondían a la FEU gestionar. Algunos de estos módulos eran la gestión de la cuartelaría, evaluaciones de la residencia, planificación y evaluación de la guardia estudiantil, registros de cortes docentes, asignaturas a mundial y arrastres. Lo anterior desencadenó en un aumento considerable de los volúmenes de datos a manejar y una poca sistematización en la actualización de la información ajena a la organización. Se debe agregar que la aplicación dataFEU v1.0 no concebía la gestión de estos grandes volúmenes de información por lo que se ha visto ralentizado el tiempo de respuesta de la aplicación a los usuarios.

Por otro lado en el año 2014 se crea en la Facultad Introductoria de Ciencias Informáticas (FICI) el proyecto MAYA: “Ecosistema de aplicaciones informáticas para la evaluación integral de los estudiantes en el contexto universitario”, que persigue como objetivo integrar en una plataforma las aplicaciones que tributen evidencias sobre el desempeño estudiantil y así contribuir a mejorar el proceso de recopilación, procesamiento y persistencia de la información que se genera. El proyecto define que sus aplicaciones deben estar desarrolladas sobre el marco de trabajo Symfony2, utilizando como capa de presentación Boostrap y gestor de base de datos PostgreSQL. dataFEU v1.0 constituye una de esas soluciones clave, pero la arquitectura y el modelo empleado en su desarrollo hacen muy engorrosa la integración hacia al ecosistema anteriormente mencionado (4).

En dataFEU v1.0 (desarrollado en Symfony1) todos sus módulos están dispersos en bundles diferentes los cuales no fueron gestionados como dependencias del núcleo del sistema; mientras que Symfony2 propone utilizar una única aplicación para optimizar la carga y rendimiento del mismo. El primero posee una estructura de directorio muy diferente al segundo, lo que imposibilita replicar la organización de sus archivos. Todo esto sumado a que el concepto de plugins fue cambiado por la filosofía de bundles y que las dependencias de la versión 1.0 del framework no tienen soporte, hacen desechar cualquier posibilidad de mantener este sistema, que redunda en gastos de tiempo y recursos humanos.

Tomando en cuenta el contexto anterior se plantea como **problema de la investigación**: ¿Cómo mejorar la arquitectura y el modelo empleados en la versión 1.0 del Sistema dataFEU para facilitar su integración al ecosistema MAYA? Este problema se enmarca en el **objeto de estudio**: Procesos de gestión de la información en las organizaciones estudiantiles, delimitándose como **campo de acción**: Sistemas para la gestión de la información de los procesos de la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas. Para la solución de este problema se plantea como **objetivo general**: Desarrollar la versión 2.0 del Sistema dataFEU para facilitar su integración al ecosistema MAYA.

Para dar cumplimiento al problema anterior se definen como **tareas de la investigación**:

1. Descripción de los procesos de negocio asociados a la FEU aprobados en el 8vo. Congreso de la organización.
2. Realización de un estudio de ingeniería de los procesos que gestiona la versión 1.0 del Sistema dataFEU.
3. Selección de los métodos, técnicas y herramientas de desarrollo adecuadas para la implementación del software.
4. Descripción de los requisitos funcionales necesarios para dar solución al problema planteado.
5. Descripción de la integración de la aplicación con el “Ecosistema de aplicaciones informáticas para la evaluación integral de los estudiantes” de la UCI.
6. Implementación de los requisitos propuestos.

7. Validación de la solución propuesta a través de pruebas al sistema.

Para dar solución a las tareas antes mencionadas se utilizaron los siguientes **métodos** de investigación:

Métodos teóricos:

- Analítico–Sintético: se utilizará en el estudio de la base teórica, de las herramientas y tecnologías a emplear, para definir las características esenciales de las mismas y sus relaciones.
- Análisis Documental: se empleará para un mejor análisis y la sistematización de la revisión de los reglamentos y resoluciones de la FEU.
- Modelación: para la mejor comprensión del problema se realizarán abstracciones utilizando modelos que tienen una estrecha relación con el objeto que se estudia, estos modelos representan las posibles soluciones que tendrá el problema.

Métodos empíricos:

- Entrevista: permitirá la exploración y constatación del estado actual del tema investigado a dirigentes de la FEU en los diferentes niveles de dirección.

El presente trabajo está estructurado en tres capítulos en los que se encuentra el contenido distribuido de la siguiente manera:

En el **primer capítulo** se expone la fundamentación teórica de la investigación, describiéndose el Sistema dataFEU v1.0. Se realiza el estudio del estado del arte de las soluciones informáticas encontradas para la gestión de los procesos estudiantiles, a nivel internacional y nacional. Se define la metodología, lenguaje y herramientas a utilizar en el desarrollo de la solución.

El **segundo capítulo** se realiza una descripción de la solución propuesta, analizando cada uno de los artefactos y elementos para el diseño de la aplicación. Se exponen las principales características de la herramienta, su diseño, su arquitectura, los módulos que la componen y los requisitos a tener en cuenta en su desarrollo.

En el **tercer capítulo** se describen los temas correspondientes al desarrollo de la solución, comenzando por cómo se implementan los elementos del modelo de diseño, se define la estructura de la aplicación web y la construcción de la solución. Se fundamenta en la validación del sistema, empleando pruebas de caja negra mediante la realización de casos de prueba.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 INTRODUCCIÓN

En el capítulo se describen los elementos principales que fundamentan la realización del presente trabajo. Se realiza un análisis de la versión 1.0 del sistema dataFEU, así como los subsistemas y módulos que lo conforman. Se muestran los resultados del estudio de los sistemas informáticos encontrados para gestionar los procesos estudiantiles. Se describen las herramientas, metodología y tecnologías necesarias para el cumplimiento del objetivo del presente trabajo.

1.2 DATAFEU V1.0

1.2.1 ANÁLISIS DEL SISTEMA

El sistema para la gestión de los procesos de la FEU (dataFEU v1.0), surge a partir del poco control de la trayectoria estudiantil, ralentización del proceso de caracterización debido al gran volumen de datos a procesar, la no estandarización de la documentación que se aportaba por cada una de las áreas en relación al proceso y la no existencia de un archivo histórico. Esto unido a la poca fiabilidad de la información recolectada, provocaba que se comprometiera la integridad, calidad y el grado de certeza de los datos para el proceso de evaluación estudiantil (3).

Surgiendo como hobby a partir de la idea de hacer más ágil, rápido y confiable el proceso de integralidad de los estudiantes en la Universidad de las Ciencias Informáticas (3), un reducido grupo de dirigentes y estudiantes comenzaron a dedicar algunas horas de sueño y dieron los primeros pasos en Julio de 2011. Esta aplicación fue desarrollada sobre el marco de trabajo Symfony1 y utilizando como capa de presentación extJS en su versión 3.3.

Producto de las desventajas de Symfony 1.4, existían múltiples barreras a la hora de continuar con el desarrollo del software. Según Fabien Potencier: "*Symfony 1 es un poco lento, así que necesitaba encontrar la forma de arreglar esto. Además las "buenas prácticas" han evolucionado mucho durante los últimos cinco años, así que para implementar mi punto de vista era necesario comenzar desde cero*". En Symfony 2.0 es imposible actualizar los proyectos realizados con la versión anterior, a menos que se reescriban completamente, debido a su notable diferencia con las versiones anteriores, presentando una nueva estructura de directorios para los proyectos en estas versiones. Además desaparecen los plugins y se sustituyen por un elemento mucho más potente llamado bundle (5).

A dataFEU v1.0 lo definen 18 módulos agrupados en cinco subsistemas por funcionalidades comunes. Cada módulo es independiente del resto y puede ser actualizado sin que esto afecte el funcionamiento del sistema. El de mayor relevancia y responsabilidad es el destinado a gestionar lo relacionado al proceso de integralidad. A continuación se da una breve descripción de los subsistemas que lo componen:

- Subsistema común: constituye el núcleo del software e integra los componentes necesarios para su funcionamiento. Está compuesto por los módulos de Seguridad, Administración, Reportes y Configuración.
- Subsistema integralidad: agrupa la mayoría de los módulos que tributan al proceso de evaluación estudiantil: módulo Méritos, Sanciones, Cargos, Bonificaciones, Alumnos Ayudantes, Académico, Proyecto y Tareas Evaluadas.
- Subsistema eventos y actividades: permite gestionar todo lo relacionado con los eventos, su planificación, participación de los estudiantes y evaluación. Lo integra un solo módulo de Actividades.
- Subsistema de funcionamiento interno: agrupa los componentes lógicos relacionados con esta área de la organización, divididos en los módulos de Emulación, Membresía, Libro del graduado y Premios a profesores.
- Subsistema alertas y notificaciones: es el encargado de gestionar todas las notificaciones y avisos tanto vía correo como dentro del propio sistema.

Esta es una aplicación informática que monitorea el proceso de evaluación estudiantil y apoya a sus principales dirigentes en la toma de decisiones. Se centra principalmente en el proceso de integralidad y caracterización de los estudiantes registrando mediante evidencias la trayectoria de cada uno de ellos en un expediente digital (6).

A raíz del dinamismo implícito en la propia universidad se comenzó a desarrollar sobre esta plataforma módulos para cubrir las necesidades de dentro y fuera de la organización de masas. Esto hizo que se implementaran procesos que no correspondían a la FEU gestionar. Ellos son:

- Cuartelería: que incluye la planificación y evaluación.
- Evaluaciones integrales de la residencia.
- Evaluaciones del TSU.
- Guardia, que abarca la planificación y evaluación.
- Otorgar premios a profesores.
- Evaluaciones de los cortes docentes.
- Registro de estudiantes con asignaturas de arrastre y mundiales.

El sistema permite (en el caso de la residencia) que las instructoras desde sus puestos de trabajo actualicen la información y que los directivos de las diferentes áreas accedan a reportes nominales y estadísticos.

1.2.2 DESCRIPCIÓN DE LOS PROCESOS DE LA FEU

En el año 2013 se realizó el 8vo. Congreso de la FEU donde se definió el actuar de la organización y se modificaron un conjunto de procesos que esta debe llevar (7). A continuación se exponen en detalle cada uno de estos procesos aprobados en la magna cita de la organización:

- **Cargos estudiantiles**

  Son atendidos por el vicepresidente y el presidente de cada estructura u otra persona que ellos designen. Estos se guardan en un archivo histórico recogiendo la trayectoria del dirigente. Una vez concluida las elecciones es que se consideran oficiales, en su defecto deben ser aprobados por el Consejo de la FEU correspondiente. Los dirigentes pueden causar bajas por diferentes causas y motivos, esta acción solo la podrá hacer oficial el vicepresidente, previa consulta con el presidente y analizando los factores con todos los implicados. Mensualmente el dirigente recibe una evaluación de su trayectoria durante el período señalado, la cual debe ser discutida en la estructura a la que pertenece. El Consejo de la FEU de la Universidad define cada curso cuáles serán los cargos que serán ocupados en las estructuras inferiores. Para que un dirigente pase a ocupar un cargo a nivel de universidad debe transitar por las estructuras inferiores.
- **Emulación**

  Es uno de los mecanismos base para la FEU, pues a partir de ahí define políticas de estimulación y de evaluación de cada una de las estructuras inmediatamente inferiores. Cada una de ellas puede definir una emulación, estableciendo sus parámetros, la ponderación de cada uno de ellos, los actores y los que fungirán como controladores.
- **Méritos**

  Son definidos por los secretariados a cada nivel donde se generen, siendo estos los encargados de otorgarlos o darlos a conocer a instancias superiores. Se clasifican en tres tipos: reconocimientos (que abarca todos los que se entreguen por las diferentes instancias o factores), distinciones y especiales (aquellos que no entren dentro de las dos categorías anteriores). Para optar por un mérito se deben cumplir ciertos parámetros, los cuales pueden ser modificados por el Consejo de la FEU a la instancia correspondiente.
- **Sanciones**

  Están definidas en el ABC de la FEU, reglamento aprobado en el 8vo. Congreso de la organización. Una vez demostrado y analizado que el estudiante cometió una indisciplina, se procede a la imposición de la medida. Pueden aplicar sanciones todos los niveles de dirección y en el caso de la brigada lo realiza su presidente, previo consentimiento de la mayoría dentro del grupo. Las sanciones

son diferentes para los estudiantes y dirigentes estudiantiles, quienes se rigen por cláusulas especiales de tipificación de la falta.

- **Libro del graduado**

  Es una forma de recopilar los datos de los estudiantes que se encuentren en quinto año, siendo esta una de las tradiciones universitarias más antiguas. El estudiante debe entregar una página, actualizando su información personal, los cuales no pasarán por ningún nivel de certificación, el objetivo es dejar sus impresiones de la universidad y que se registre en un archivo histórico.

- **Participación en las actividades**

  Comienza con la planificación por parte del nivel de dirección que las organice. Se define el lugar, fecha, hora, los tipos de participación, las necesidades logísticas y la esfera a la cual está asociada. El miembro del secretariado de la FEU que atiende dicha esfera es el encargado de darle seguimiento y coordinar los aseguramientos materiales necesarios para su realización. Las que se ejecutan a nivel de brigada, son atendidas por el presidente quien se encargaría de la planificación y aseguramiento. Una vez que la actividad o el evento han sido realizados, el presidente de brigada archivará los participantes de su grupo teniendo en cuenta los resultados obtenidos y el tipo de participación. En dependencia del nivel donde se realicen dichas actividades, se emitirá por parte del secretariado un listado donde se certificará la asistencia de los estudiantes. La participación puede darse de tres tipos diferentes: como ponente (con la presentación de un trabajo), colaterales (en los sub-eventos que se realizan) y como organizador.

- **Alumnos ayudantes (AA)**

  Son un caso especial de membresía, que se atiende directamente por los responsables de la esfera Vida académica. Para ingresar al Movimiento de Alumnos Ayudantes (MAA) se deben cumplir un número mínimo de requisitos, además de estar validado por la brigada. Terminado el proceso de ingreso el responsable de la esfera por la cual son atendidos procede a su registro, teniendo en cuenta las funciones que realizarán, el tutor y la elaboración de su plan de trabajo mensual. En caso de que los requisitos que le dieron entrada al movimiento son perdidos, se procede a darle baja, especificando las causas que la motivaron y en previa consulta con el presidente de la estructura que lo atiende. Cada semestre los AA son ratificados, verificándose si posee los requisitos para continuar en el movimiento. Mensualmente son evaluados teniendo en cuenta la trayectoria en el período, la opinión de la brigada y del jefe de departamento por la parte administrativa.

1.3 ESTUDIO DEL ESTADO DEL ARTE

Durante la investigación se realizó una búsqueda de los sistemas que estuvieran relacionados con el objeto de estudio para conocer los referentes teóricos de la ciencia hasta el momento. A continuación se describen los sistemas encontrados:

DocCF

Aplicación desarrollada por Grupo CF Developer para automatizar los procedimientos administrativos, académicos y comerciales de las Instituciones Educativas. El objetivo de este software como herramienta, es gestionar los procesos internos y facilitar la coordinación y comunicación entre padres, alumnos, docentes y cargos directivos para ofrecer información estadística sobre dichos procesos y facilitar la toma de decisiones en el proceso de gestión de la institución (8).

Esta aplicación realiza una integración de cada módulo para generar una información centralizada de la institución.

- Módulo de Básico (gestión de Docentes y Alumnos).
- Módulo de Gestión académica.
- Módulo de Gestión Económica.
- Módulo de Gestión de Biblioteca.
- Herramientas de Gestión.
- Módulo de Gestión Web

Incluye múltiples funciones y herramientas que permiten automatizar los procedimientos administrativos, académicos y comerciales de la institución, optimizando sus recursos y disminuyendo el tiempo. Contiene más de 60 procedimientos tradicionales automatizados (asignación de horarios, matriculación, calificaciones, ausentismo, pagos, etc.) DocCF se convierte en una de las mejores herramientas para mejorar los procesos de gestión interna y facilitar la coordinación entre cargos directivos, docentes, alumnos y padres de familia con la utilización de una única y completa aplicación de gestión académica. El Módulo de Gestión Web hace posible que los padres de familia puedan estar atentos a la situación escolar de sus hijos, tales como: calificaciones, estados de cuenta, horario de exámenes, avisos, circulares, etc.

Solución informática para la gestión de la información de alumnos ayudantes en la UCI

El módulo Ayudantía forma parte del Sistema de Gestión Académica de Pregrado (SGAP) que está integrado al Sistema de Gestión Universitaria (SGU) (9) lo que permite el uso de las facilidades provistas por dicha integración. Entre estas facilidades están la reutilización de varios componentes que fueron implementados para otros sistemas, la centralización de archivos e información y la eliminación de redundancias en el código. Igualmente se comparte la relación entre los módulos de SGAP, con los que se encuentran en continua

colaboración. Esta solución informática integrada al SGAP que estandariza la gestión de información de alumnos ayudantes de la UCI en función de elevar la calidad del proceso docente-educativo en la institución.

SAS Académico

Software de gestión escolar orientado a instituciones educativas colombianas para el manejo de las inscripciones, matrículas, calificaciones, caracterización y estadísticas en tiempo record, gracias a sus potentes herramientas de trabajo en equipo (10).

Cuenta con una sólida estructura a nivel de interfaz de usuario en la cual se ha escogido cuidadosamente los controles adecuados para cada tarea y en donde se ha tenido especial cuidado con la estandarización de la nomenclatura de los menús y las ventanas. También hace énfasis en la profesionalidad del diseño gráfico, tiene en cuenta la uniformidad del color y los íconos en todas las ventanas, comentarios relevantes en las secciones que ameriten, para dar como resultado una herramienta totalmente intuitiva para el usuario.

Este programa está desarrollado en Visual FoxPro 8.0, Flash 8, y unas importantes DLLs para el manejo de la compresión de archivos y manejo de diferentes formatos de documentos y cuenta con las siguientes opciones:

- Consultas y reportes académicos.
- Caracterización de estudiantes.
- Informes descriptivos a educandos (con información de períodos anteriores y procesos de recuperación).
- Informes estadísticos por grupo.
- Libro final.
- Entorno gráfico y habilitado para correr bajo plataforma Windows.
- Diseño y entorno intuitivo.
- Capacidad de exportar a Excel, Word y PDF.
- Control de acceso parametrizable según el perfil del usuario.
- Auditoria a procesos, para determinar si la información de determinado proceso se encuentre ingresada totalmente al sistema, quien la ingreso y en qué momento fue ingresada.

Entre sus principales ventajas se destacan:

- Control de Usuarios: que permite restringir el acceso a la información a personas no autorizadas, lo cual garantiza la privacidad y seguridad de la información.
- Puede aprovechar la interfaz gráfica de Windows para incluir fotos en el programa e imprimirlas junto a los reportes de Hoja de vida del profesor y del estudiante e inclusive a los carnés estudiantiles.

- La obtención de reportes y listados es inmediata, lo cual reduce la congestión de labores y agiliza los procesos.
- El suministro de información no depende de la disponibilidad de tiempo y recursos de terceros. Todo ello estará bajo el control de la institución únicamente.

Sin dudas tanto DocCF y SAS son sistemas potentes en cuanto a la gestión de reportes. Ambos son enfocados a los estudiantes y no a los procesos, aunque debido a la licencia de software privativo, se hace difícil la obtención de la misma. El SGU de igual forma, cumple con los estándares de software libre, se enfoca en los roles adecuados de estudiante pero en la gestión de los procesos de la FEU se queda parcializado, debido a que solamente se encarga del movimiento de alumnos ayudantes siendo este solo uno de los procesos que debería gestionar. dataFEU v1.0 sería la opción ideal para darle continuidad a su desarrollo pero debido a la tecnología con la que está desarrollado se hace muy engorrosa su integración con el ecosistema MAYA.

1.4 ECOSISTEMA MAYA

El manejo de la información personal, académica, productiva, investigativa, intelectual y moral de los estudiantes, reviste vital importancia para la caracterización/evaluación integral de los estudiantes y la toma de decisiones a todos los niveles de dirección. Se añade además que la gestión de los datos de los estudiantes debe estar orientada al uso de sistemas automatizados que garanticen la disponibilidad, confidencialidad e integridad de la información y los procesos (4).

Teniendo en cuenta lo anterior se decidió implementar un ecosistema de aplicaciones informáticas que diera sustento al procesamiento de los grandes volúmenes de datos que se generan para la caracterización de los estudiantes en el contexto universitario. Es importante agregar que la Facultad Introductoria de Ciencias Informáticas(FICI) lleva a cabo un proyecto de investigación pedagógica que da sustento al ecosistema y que tiene como objetivo general: elaborar una estrategia pedagógica de orientación profesional para el primer año de la carrera de Ingeniería en Ciencias Informáticas que tribute al desarrollo de las potencialidades motivacionales, morales e intelectuales básicas de los estudiantes, para que actúen responsablemente en el proceso de su formación profesional (4).

MAYA, es el nombre que lleva este ecosistema de aplicaciones informáticas. Dentro de los sistemas que la componen están:

- **ALMA:** “Sistema para la gestión de datos primarios”, es la aplicación encargada de controlar la información base de los estudiantes, profesores, especialistas y trabajadores que interactúan con la

plataforma. Contiene un API de servicios(https://alma.uci.cu/api/docs) que garantiza un enlace de comunicación directo entre los sistemas que se integran a ella.

- **GERES:** Sistema para la gestión de la información asociada a la Residencia Estudiantil, se encarga de llevar los principales procesos como: evaluación y planificación de la cuartelería, las evaluaciones integrales del becario, las indisciplinas cometidas y las incidencias de mantenimiento. Contiene una API de servicios en la dirección https://geres.uci.cu/api/docs (11).
- **EVENTOS:** gestiona los eventos estudiantiles, relacionados con las organizaciones, el gestiona diferentes eventos dígase académicos, culturales, deportivos y talleres. Contiene una API de servicios en la dirección https://eventosuniversitarios.uci.cu/api/docs (12).
- **EDUCA:** es un sistema para la gestión de la información asociada a la estrategia educativa para gestionar los objetivos por diferentes niveles, las indisciplinas y los méritos. Contiene una API de servicios en la dirección https://educa.uci.cu/api/docs (13).

1.5 HERRAMIENTAS Y TECNOLOGÍAS

Para un correcto desarrollo e integración de las aplicaciones se definieron, por el proyecto MAYA, un conjunto de herramientas y tecnologías, las cuales serán descritas a continuación:

- **Plugin Firebug**

Mozilla Firefox, o simplemente Firefox es un navegador web libre y de código abierto desarrollado por Mozilla. Es un navegador rápido y eficiente que permite instalar varios Plugins para mejorar sus funcionalidades. Entre los plugins que se le puede incorporar está el Plugin Firebug, el cual edita, depura y monitorea CSS, HTML y JavaScript de cualquier página web, lo cual es muy útil en la programación web (14). Entre las principales ventajas se destacan que es totalmente gratuito, que lo único necesario para poder usarlo es tener Firefox y permite modificar la página en su sitio y ver el resultado inmediatamente (15).

- **Visual Paradigm para UML 8.0**

Herramienta CASE: Ingeniería de Software Asistida por Computación. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación (16).

A pesar de que no es software libre, posee una versión libre llamada Visual Paradigm for UML Comunity Edition, herramienta multiplataforma que soporta el ciclo de vida completo del desarrollo de una aplicación informática. Teniendo en cuenta lo anterior se selecciona la versión 8.0.

- **Controlador de versiones Git 1.9.1**

Software Libre distribuido bajo los términos de la Licencia Pública General GNU versión 2. Es bastante inteligente porque cuando un archivo no cambia, en lugar de guardar la misma fotografía varias veces, guarda una referencia a esa fotografía. Casi todo es guardado en Git no por nombre, sino por la suma de comprobación de sus contenidos**.** De esta forma se optimizan los recursos del sistema. Otro de sus principios es que cuando se introduce un cambio en el proyecto simplemente se añade más información al repositorio, lo que permite que sea muy difícil estropear algo por error o que no se pueda deshacer esas modificaciones (17).

- **PgAdmin III 1.18.1**

Plataforma de administración y desarrollo que se encuentre entre las más populares de código abierto para PostgreSQL. Está diseñado para responder a las necesidades de todos los usuarios de escribir consultas SQL sencillas para desarrollar base de datos complejas. La interfaz gráfica es compatible con todas las características de PostgresSQL y facilita la administración. La conexión con el servidor se puede hacer a través de TCP/IP o Unix Domain hembra (en plataformas *nix), y puede ser encriptado SSL para la seguridad. No se requieren controladores adicionales para comunicarse con el servidor de base de datos (18).

- **Netbeans 8.0**

Entorno de desarrollo para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Se encuentra en Java, pero puede servir para otros lenguajes como PHP, Groovy, HTML, C#, C++, entre otros. El mismo es un producto libre y gratuito sin restricciones de uso. El código fuente está disponible para su reutilización de acuerdo con la Common Development and Distribution License (CDDL) v1.0 and the GNU General Public License (GPL) v2 (19).

- **PostgreSQL 9.1**

Es un motor de base de datos relacional orientada a objetos y de código abierto, es potente y robusto. Tiene soporte total para transacciones, disparadores, vistas, procedimientos almacenados, almacenamiento de objetos de gran tamaño. Se destaca en ejecutar consultas complejas, consultas sobre vistas, subconsultas y joins de gran tamaño, además es multiplataforma (20).

- **Symfony 2.7.9**

Framework PHP para el desarrollo de aplicaciones y sitios web rápidos y seguros. Su código, y el de todos los componentes y librerías que incluye se publican bajo licencia MIT de software libre. Cuenta con un sitio online en el cual se puede encontrar una gran cantidad de documentación, también de forma gratuita. Propone un cambio radical tanto en la arquitectura como en la filosofía de trabajo de versiones anteriores. Está diseñado para explotar las potencialidades de PHP, lo cual supone mejoras en el rendimiento del

framework. Su arquitectura está totalmente desacoplada, esto permite eliminar aquellos componentes que no son necesarios para el desarrollo del proyecto (21).

- **Doctrine 2.4.8**

Es un asignador-objeto-relacional (ORM) para PHP que proporciona persistencia transparente de objetos PHP situados en la parte superior de una poderosa capa de abstracción de base de datos (DBAL por DataBase Abstraction Layer). Una de las características claves de Doctrine es la opción de escribir las consultas de base de datos en un dialecto SQL propio orientado a objetos llamados Lenguaje de Consulta Doctrine (DQL por Doctrine Query Language), inspirado en Hibernates HQL (22).

Dentro de las principales ventajas que aporta ORM es la reutilización, permitiendo llamar a los métodos de un objeto de datos desde varias partes de la aplicación e incluso desde diferentes aplicaciones. Además la utilización de objetos en vez de registros y de clases en vez de tablas, tiene otra ventaja: permite añadir métodos de acceso en los objetos que no tiene relación directa con una tabla (23).

- **Apache 2.4.7**

Es actualmente uno de los servidores web más utilizado. Ampliamente extensible a través de módulos y muy configurable. Entre sus principales características destaca la configuración de servidores virtuales que le permiten ejecutar, en la misma máquina, diferentes servidores para diferentes direcciones IP, diferentes nombres de máquina o diferentes puertos (24).

Este trae consigo una serie de ventajas entre las que se encuentran:

- Es un servidor altamente configurable de diseño modular.
- Es una tecnología gratuita de código abierto.
- Funciona en Linux, en otros sistemas de Unix y Windows.

Apache 2 presenta diversas características, entre ellas: un elaborado y manejable índice de directorios, un directorio de alias, negociación de contenidos, informe de errores HTTP configurable, soporte de SSL, gestión de recursos para procesos hijos, reescritura de los Localizadores Uniforme de Recursos (por sus siglas en inglés URL), comprobación de la ortografía de las URL y manuales en línea (25).

- **Boostrap 3**

Es el framework de JavaScript, CSS y HTML que se encuentra entre los más usados para el desarrollo adaptativo de proyectos para móviles en la web. Entre sus principales ventajas se encuentran: facilitar y agilizar el desarrollo de interfaces web, está hecho por personas de todos los niveles de habilidades, dispositivos de todas las formas y proyectos de todos los tamaños, permite tener extensa documentación de los elementos HTML más comunes, docenas de componentes HTML y CSS personalizados, con increíbles plugins de JQuery, escala de forma fácil y eficiente los sitios web mediante un solo código, desde teléfonos

a tablets y a escritorio con consultas de CSS. Es de código abierto, alojado, desarrollado y mantenido en Github (26).

Lenguajes

Los lenguajes de desarrollo intentan conservar una similitud con el lenguaje humano, con la finalidad de que sean más naturales a quienes los usan. Establecen un conjunto de reglas sintácticas y semánticas, las cuales rigen la estructura del programa de computación que se escribe o edita. De esta forma, permiten a los programadores o desarrolladores, poder especificar de forma precisa los datos sobre los que se van a actuar, su almacenamiento, transmisión y demás acciones a realizar bajo las distintas circunstancias consideradas (27).

- **PHP 5.4**

Lenguaje interpretado de propósito general ampliamente usado, de código abierto especialmente adecuado para desarrollo web y que puede ser incrustado dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. Es orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos. Posee capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destacándose su conectividad con PostgreSQL (28).

- **JavaScript 1.8**

Lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS), su uso en aplicaciones externas a la web, por ejemplo en documentos PDF y aplicaciones de escritorio (mayoritariamente widgets) es también significativo (29).

- **Lenguaje de Marcas de Hypertexto (por sus siglas en inglés HTML) 5**

Elemento de construcción más básico de una página web. Se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad. Le añade "marcado" a un texto estándar en español. "Hipertexto" se refiere a enlaces que conectan una página Web con otra, haciendo de la Telaraña Mundial (*World Wide Web*) lo que se conoce en la actualidad como Internet. Soporta imágenes y también otro tipo de elementos multimedia. Es el lenguaje que describe la estructura y el contenido semántico de un documento web (30).

- **Hojas de estilo en cascada (por sus siglas en inglés CSS) 3**

Lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (lenguaje de marcas extensibles). El W3C es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los navegadores. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación (31).

- **Lenguaje unificado de modelado (por sus siglas en inglés UML) 2.0**

Se trata de un lenguaje gráfico para construir, documentar, visualizar y especificar un sistema de software. Viabiliza la realización de diagramas estáticos, dinámicos, de entorno y organizativos. Ofrece un estándar para describir de la manera más sencilla y entendible para cualquier desarrollador los aspectos conceptuales tales como procesos de negocio y funciones del sistema (32).

1.6 METODOLOGÍA DE DESARROLLO DE SOFTWARE

La Programación Extrema (por sus siglas en inglés XP), es uno de los varios procesos ágiles que hace hincapié en la satisfacción del cliente. En vez de entregar el código completamente terminado en una fecha lejana en el futuro, este proceso propone entregar el código por paquetes en cada una de las iteraciones. Propicia el trabajo en equipo donde los gerentes, clientes y desarrolladores son todos socios iguales (33).
Se encarga de la mejora un proyecto de software en cinco formas esenciales; la comunicación, la sencillez, la retroalimentación, el respeto y el valor. Los programadores extremos constantemente se comunican con sus clientes y colegas programadores. Mantienen su diseño simple y limpio. Se obtienen retroalimentación probando su software a partir del primer día. Entregan el sistema a los clientes tan pronto como sea posible e implementar cambios como se sugiere. Con esta base Extreme, los programadores son capaces de responder con valentía a las necesidades cambiantes y la tecnología (34).Definiendo 5 fases fundamentales (35):

Exploración

Los clientes escriben las historias de usuario (funcionalidades con que debe contar el sistema) de lo que ellos quisieran incluir para la primera entrega. Cada plantilla describe las características que deben ser adicionadas al programa. Al mismo tiempo el equipo del proyecto se familiariza con las herramientas, la tecnología y las prácticas que utilizarán en el proyecto. La tecnología al ser usada será probada y las posibles arquitecturas para el sistema son exploradas construyendo un prototipo del sistema. La fase de exploración toma entre unas cuantas semanas a unos cuantos meses, dependiendo de que tanto los programadores conocen la tecnología.

Planeación

Configura la prioridad para las historias de usuario, contenidas en las tarjetas CRC (Clase-Responsabilidad-Colaboración, una técnica que reemplaza a los diagramas para la representación de modelos, en las que se escriben las responsabilidades) y se realiza un contrato del contenido para la primera entrega. Los programadores primero estiman cuánto esfuerzo requieren para cada historia y se hace una programación de acuerdo a esta estimación. El tiempo de la programación de la primera entrega normalmente no excede dos meses y el tiempo de la fase como tal toma un par de días.

Iteraciones

La programación que se determinó en la etapa de planeación es dividida en un número de iteraciones donde cada una tomará de una a cuatro semanas para ser implementada. La primera iteración crea la arquitectura de todo el sistema; esto es logrado seleccionando las historias que hacen cumplir la estructura para todo el sistema. El cliente decide las historias seleccionadas para cada iteración. Las pruebas funcionales creadas por los clientes son para correr al final de cada iteración. Al final de la última iteración, el sistema estará listo para ser entregado y llevarlo a producción.

Producción

Requiere pruebas extras y chequeos de la ejecución del sistema antes de que sea entregado al cliente. En ésta fase, se pueden encontrar nuevos cambios y se toma la decisión si serán incluidos en la entrega actual. Durante esta fase, las iteraciones pueden necesitar ser recortadas de tres semanas a una semana. Después que la primera entrega es producida para el uso del cliente, el proyecto XP debe mantener el sistema en producción corriendo mientras que también se estén produciendo nuevas iteraciones.

Mantenimiento

Requiere también un esfuerzo para soportar las tareas de los clientes. Así, la velocidad del desarrollo puede desacelerarse después de que el sistema está en producción. La fase de mantenimiento puede requerir incorporar nuevas personas al equipo y/o cambiar su estructura. Dentro de esta fase se llega a un estado llamado “de muerte”, que sucede cuando el cliente no tiene más historias para ser implementadas. Esto requiere que el sistema satisfaga también las necesidades en otros aspectos, como por ejemplo lo concerniente a la ejecución y la confiabilidad. Éste es el momento en el proceso XP cuando la documentación necesaria del sistema es finalmente escrita porque no habrá más cambios en la arquitectura, diseño o código. La muerte puede ocurrir si el sistema no está entregando los artefactos deseados o si se está convirtiendo muy costoso implementarlo.

Entre las principales características de XP se destacan:

- Pruebas unitarias: Se basan en las pruebas realizadas a los principales procesos, de tal manera que se adelante en algo hacia el futuro, se puedan hacer pruebas de las fallas que pudieran ocurrir.
- Refabricación: Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio (36).
- Programación en pares: Una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto mientras uno conduce, el otro consulta el mapa.

El proyecto MAYA la define como modelo de desarrollo para sus aplicaciones por brindarle la oportunidad al equipo de trabajo, la comodidad y flexibilidad a cambios que surjan en cualquier etapa del ciclo de vida de la aplicación. El grupo de desarrollo es pequeño con formación elevada y capacidad de aprender. Los clientes tienen una estrecha vinculación con el equipo de desarrollo.

1.7 CONLCUSIONES PARCIALES

A partir del análisis realizado de la herramienta dataFEU v1.0, se demostró que la arquitectura del mismo no permite la evolución e integración hacia el ecosistema MAYA. El estudio del estado del arte realizado permitió definir que no se conoce a nivel nacional e internacional un sistema para la gestión de los procesos de la Federación Estudiantil Universitaria que solventen las nuevas situaciones de la organización. Las herramientas y tecnologías seleccionadas definen el marco tecnológico para cumplir el objetivo de la investigación en ajuste a las políticas de migración a software libre por la cual aboga el país.

CAPÍTULO 2: DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

2.1 INTRODUCCIÓN

En el presente capítulo se describe el proceso de migración de datos de dataFEU v 1.0 hacia el ecosistema MAYA. Se describen los procesos de negocio de la solución propuesta. Se definen de los requisitos funcionales y no funcionales haciendo uso de técnicas de obtención de requisitos; además, se fundamenta el patrón de arquitectura, los patrones de diseños y los componentes visuales del sistema.

2.2 MIGRACIÓN DE LOS DATOS

El transcurso de los años y el avance de la tecnología unido al enorme incremento de los volúmenes de información, ha provocado que el hombre tenga la necesidad de trasladar los conocimientos que guardaba a formas de almacenamiento que le permitieran acumular mayores volúmenes de datos en el menor espacio físico posible. El desarrollo que han alcanzado las ciencias computacionales y de los medios de almacenamiento masivos, tanto magnéticos como ópticos le han permitido lograr este objetivo. Según la Real Academia Española RAE, se denomina migración: *"paso de los programas, archivos y datos de un sistema desde una determinada plataforma tecnológica a otra diferente"* (37).

dataFEU v1.0 mantiene una base de datos histórica de hace cuatro años, con datos de gran importancia necesarios en el proceso de evaluación estudiantil. Esta información abarca diferentes áreas como: la Residencia Estudiantil, la Extensión Universitaria, la Académica y la asociada a la propia organización. Es por ello que se hace necesario hacer perdurable esta información, de manera que el ecosistema MAYA dispusiera de ella.

El proceso de migración de los datos consistió en dos fases fundamentales:

Primera: la extracción y procesamiento de los datos primarios que contenía dataFEU v1.0 para cargarlos en ALMA (nombre, apellidos, foto, carnet de identidad, grupo, apartamento, municipio, provincia, dirección particular, área y facultad). Esto garantizó que la información persistiera centralizada en ALMA.

Segunda: se ocupó de los datos asociados a la organización (actividades, méritos, sanciones, alumnos ayudantes y trayectoria estudiantil) hacia la versión 2.0. Es válido mencionar que la información relacionada a otras áreas quedó almacenada en el proyecto para futuras fases donde se hiciera necesario persistir esa información en otros sistemas de gestión afines con los datos guardados.

El gestor de base de datos utilizado en dataFEU v1.0 fue MySQL mientras que los definidos en la arquitectura de MAYA corresponden a PostgreSQL. Esto unido a que los datos debían procesarse y normalizarse, obligó a desarrollar una herramienta de apoyo en lenguaje JAVA que facilitara la extracción, transformación y carga

hacia las nuevas aplicaciones (ALMA y dataFEU v2.0). Esta aplicación hecha a la medida posibilitó manejar diferentes conexiones a bases de datos, seleccionar qué datos clonar y hacer algoritmos complejos de tratamiento de información como el procesamiento de la foto desde una url hasta su almacenamiento en base64.

2.3 DESCRIPCIÓN DE LOS PROCESOS DE NEGOCIO

Trayectoria estudiantil

El proceso inicia cuando concluyen las elecciones. El vicepresidente de la estructura al nivel correspondiente o el presidente de la comisión organizadora de las elecciones en su defecto, son los encargados de registrar la información asociada al cargo y notificar a los involucrados. Mensualmente el dirigente recibe una evaluación de su trayectoria durante el período señalado la cual debe ser registrada una vez que concluya la reunión de discusión de la misma. Una vez que el dirigente cause baja por cualquier motivo es el presidente de la estructura correspondiente el responsable de actualizar estos datos.

Emulación

Puede ser creada por los secretariados de la organización a cualquiera de los niveles (universidad, facultad o brigada). La misma tiene un período de vigencia definido, que puede ser durante un curso o hasta un rango de fecha establecido. Cada una de las emulaciones establece un conjunto de moderadores los cuales son los encargados de gestionar la información correspondiente a esta, dígase: establecer parámetros, actores, otros moderadores y el acumulado final.

Méritos

Son definidos por las estructuras a nivel de facultad y de universidad. Una vez analizadas las condiciones que cumple una persona y aprobada en la estructura de dirección correspondiente, es el vicepresidente el encargado de registrar esta información. Cuando los méritos son a nivel de brigada esta responsabilidad es del presidente de la misma. El proceso concluye cuando el usuario es notificado.

Sanciones

El proceso inicia cuando se analiza un estudiante teniendo en cuenta el grado de indisciplina cometido. Luego de aprobada la sanción de la falta cometida el secretariado de la FEU al nivel correspondiente designa un miembro que registrará esta información. Concluido esto se le es notificado al estudiante de la sanción correspondiente.

Participación en las actividades

Comienza con la planificación por parte del nivel de dirección que las organice. Se define el lugar, fecha, hora, los tipos de participación, las necesidades logísticas y la esfera a la cual está asociada. Es

responsabilidad del presidente de brigada de registrar esta información en un plazo de 15 días una vez concluida la misma.

Libro del graduado

Al arribar un estudiante al quinto año de la carrera confecciona una página del libro del graduado. El secretariado a nivel de facultad es el encargado de revisar dichas páginas para la posterior conformación del libro al nivel correspondiente (facultad y/o universidad).

Alumnos Ayudantes

Se realiza una convocatoria de las ofertas de asignaturas por las que los estudiantes puedan optar. El estudiante interesado envía su solicitud al secretariado de la FEU de su facultad. Una vez recepcionadas todas las solicitudes se analizan cada uno de los estudiantes de acuerdo a los requisitos definidos en el reglamento (7). El responsable del MAA y el vicepresidente de vida académica son los encargados de registrar esta información y notificar a los estudiantes que solicitaron integrar el movimiento.

2.4 REQUERIMENTOS

Los requerimientos son propiedades o restricciones determinadas de forma precisa que deben satisfacerse (38). Describen los servicios que ha de ofrecer dataFEU v2.0 y las restricciones asociadas a su funcionamiento. Se definen en lenguaje natural, se expresan de forma individual y generalmente se enumeran. Deben ser concisos, consistentes, claros y concretos para evitar ambigüedades.

En este acápite se especifican las condiciones o capacidades que el sistema debe cumplir y las restricciones bajo las cuales debe operar, logrando un entendimiento entre el equipo de desarrollo y el cliente. Una vez definidos claramente los requisitos funcionales y no funcionales del sistema, se realiza una descripción detallada de cada uno de ellos (39). A continuación se muestran las técnicas utilizadas y el listado de los requisitos funcionales definidos para el sistema.

2.4.1 TÉCNICAS PARA LA CAPTURA DE REQUISITOS

Para realizar una eficiente captura de los requisitos se tuvieron en cuenta las siguientes técnicas:

Entrevista

Es una de las técnicas que más se utilizó, siendo de gran utilidad para la obtención de información cualitativa como opiniones o descripciones subjetivas de actividades de la FEU. Se realizaron entrevistas que cubrieron los diferentes niveles de la organización (organizadora a nivel de universidad, presidente de las facultades 2, 3 y 5, así como a 7 presidentes de brigadas).

Estudio documental

Se tomó como base el estudio de la documentación de documentos rectores de la FEU, así como las resoluciones y guías de trabajo creados por el Consejo de la FEU de la Universidad. Esto permitió llegar a una comprensión más clara de los documentos que se generan, los cuáles son imprescindibles para estos procesos que no deben ser obviados en la elaboración de la aplicación.

Tormenta de ideas

Mediante esta técnica de trabajo en grupo, se obtuvo el mayor número de ideas o soluciones a cuestiones planteadas en el menor tiempo posible, se aprovechó la capacidad creativa de las personas en el grupo de trabajo (40).

Observación de sistemas semejantes

Se realizó un estudio de soluciones semejantes, lo que permitió obtener una base de propuestas de requisitos que sirvió como idea para el desarrollo de la solución en la presente investigación (41).

2.4.2 DEFINICIÓN DE REQUISITOS FUNCIONALES

Los requerimientos funcionales (RF) son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer (42). Son capacidades o condiciones que el sistema debe cumplir, describen con detalle la función de este, sus entradas y salidas, excepciones, entre otros, estos deben estar bien determinados y ser convenientemente comprendidos tanto por los implicados para con el sistema como por los desarrolladores del mismo para que exista un entendimiento común.

Como resultado de haber aplicado las técnicas de captura de requerimientos abordadas en el subepígrafe anterior se obtuvo como un total de 96 requisitos funcionales. Entre las ventajas de la metodología XP se resalta que el cliente tiene el control sobre las prioridades de los requisitos, definiéndose en ALTA, MEDIA y BAJA (43). A continuación se exponen cada una de las agrupaciones y los requisitos funcionales que pertenecen a cada una de ellas.

Trayectoria estudiantil

No.	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF1	Insertar cargo	Se inserta por parte de un dirigente estudiantil un cargo a un estudiante de su mismo nivel o de niveles inferiores.	ALTA	BAJA
RF2	Modificar cargo	Se modifica por parte de un dirigente estudiantil un cargo a un estudiante de su mismo nivel o de niveles inferiores.	MEDIA	BAJA
RF3	Eliminar cargo	Se elimina por parte de un dirigente estudiantil un cargo a un estudiante de su mismo nivel o de niveles inferiores.	MEDIA	BAJA
RF4	Listar cargo	Se lista por parte de un dirigente estudiantil un cargo a un estudiante de su mismo nivel o de niveles inferiores.	BAJA	BAJA
RF5	Insertar esfera	Se inserta por parte del administrador las esferas de la FEU.	ALTA	BAJA
RF6	Modificar esfera	Se modifica por parte del administrador las esferas de la FEU.	MEDIA	BAJA
RF7	Eliminar esfera	Se elimina por parte del administrador las esferas de la FEU.	MEDIA	BAJA

RF8	Listar esfera	Se lista por parte del administrador las esferas de la FEU.	BAJA	BAJA
RF9	Asignar cargo a estudiante	Registrarle el cargo al estudiante para que sea reconocido como dirigente estudiantil.	ALTA	BAJA
RF10	Eliminar cargo a estudiante	Elimina el cargo a un estudiante.	MEDIA	BAJA
RF11	Modificar cargo a estudiante	Modifica el cargo a un estudiante.	MEDIA	BAJA
RF12	Listar cargos estudiantiles	Listar los cargos que ha ocupado el estudiante durante la carrera	BAJA	BAJA
RF13	Mostrar listado de dirigentes estudiantiles	Lista los dirigentes estudiantiles.	BAJA	MEDIA
RF14	Insertar tipo de baja	Inserta un tipo de baja.	ALTA	BAJA
RF15	Modificar tipo de baja	Modifica un tipo de baja.	MEDIA	BAJA
RF16	Eliminar tipo de baja	Elimina un tipo de baja.	MEDIA	BAJA
RF17	Dar baja a dirigente estudiantil	Da baja a un estudiante del cargo que ocupa.	ALTA	MEDIA
RF18	Modificar baja a dirigente estudiantil	Modifica la baja al dirigente estudiantil.	MEDIA	MEDIA
RF19	Eliminar baja a dirigente estudiantil	Elimina la baja a un dirigente estudiantil.	MEDIA	MEDIA

RF20	Mostrar evaluación de dirigente estudiantil	Muestra la evaluación de dirigente estudiantil.	BAJA	MEDIA
RF21	Evaluar dirigente	Asigna evaluación a los dirigentes.	MEDIA	MEDIA
RF22	Mostrar listado de baja de los dirigentes estudiantiles	Lista las bajas de los dirigentes estudiantiles.	BAJA	MEDIA

Gestión de la emulación

No	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF23	Insertar emulación	Inserta una nueva emulación.	ALTA	BAJA
RF24	Modificar emulación	Modifica una emulación.	MEDIA	BAJA
RF25	Eliminar emulación	Eliminar una emulación.	MEDIA	BAJA
RF26	Insertar parámetro	Inserta parámetro a la emulación.	ALTA	BAJA
RF27	Modificar parámetro	Modifica parámetro a la emulación.	MEDIA	BAJA
RF28	Eliminar parámetro	Elimina parámetro de la emulación.	MEDIA	BAJA
RF29	Listar parámetros	Lista los parámetros existentes	BAJA	BAJA
RF30	Asignar moderador a la emulación	Asigna moderador a una emulación.	MEDIA	MEDIA

RF31	Modificar moderador de la emulación	Modifica el moderador de una emulación.	MEDIA	MEDIA
RF32	Eliminar moderador de la emulación	Elimina el moderador de la emulación.	MEDIA	MEDIA
RF33	Asignar actores a la emulación	Asigna actores a la emulación.	ALTA	MEDIA
RF34	Modificar actores de la emulación	Modifica actores de la emulación.	MEDIA	MEDIA
RF35	Eliminar actores de la emulación	Elimina actores de la emulación.	MEDIA	MEDIA
RF36	Calcular acumulado de la emulación	Determina el acumulado de la emulación.	ALTA	BAJA
RF37	Calcular resultado de la emulación	Determina el resultado de la emulación.	ALTA	BAJA
RF38	Insertar estado de emulación	Inserta el estado de la emulación.	ALTA	BAJA
RF39	Modificar estado de emulación	Modifica el estado de la emulación.	MEDIA	BAJA
RF40	Eliminar estado de emulación	Elimina el estado de la emulación.	MEDIA	BAJA
RF41	Listar estados de emulación	Lista los estados de la emulación	BAJA	BAJA

Méritos

No.	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF42	Insertar tipo mérito	Inserta un tipo de mérito.	ALTA	BAJA
RF43	Modificar tipo mérito	Modifica un tipo de mérito.	MEDIA	BAJA
RF44	Eliminar tipo mérito	Elimina un tipo de mérito.	MEDIA	BAJA
RF 45	Listar tipo mérito	Lista los tipos de méritos	BAJA	BAJA
RF46	Insertar nuevo mérito	Inserta un nuevo mérito.	ALTA	BAJA
RF47	Modificar mérito	Modifica un mérito.	MEDIA	BAJA
RF48	Eliminar mérito	Elimina un mérito.	MEDIA	BAJA
RF49	Listar mérito	Lista los méritos	BAJA	BAJA
RF50	Asignar mérito a estudiante	Registrarle a un estudiante el o los méritos obtenidos por este.	ALTA	BAJA
RF51	Eliminar mérito a estudiante	Elimina un mérito a un estudiante.	MEDIA	BAJA
RF52	Asignar un mérito a varios estudiantes	Asigna un mérito a varios estudiantes.	ALTA	ALTA
RF53	Listar mérito a estudiante	Lista los méritos que le han sido asignado a cada estudiante	BAJA	BAJA

Incidencias disciplinarias

No.	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF54	Asignar sanción a estudiante	Registrarle a un estudiante una o más sanciones.	ALTA	BAJA
RF55	Eliminar sanción a estudiante	Elimina la(s) sanción(es) impuesta a un estudiante.	MEDIA	BAJA
RF56	Modificar sanción a estudiante	Modifica la sanción impuesta a un estudiante.	ALTA	BAJA
RF57	Listar sanción a estudiante	Lista las sanciones que le han sido impuestas a cada estudiante.	BAJA	BAJA
RF58	Insertar tipo sanción	Inserta un tipo de sanción.	ALTA	BAJA
RF59	Modificar tipo sanción	Modifica un tipo de sanción.	MEDIA	BAJA
RF60	Eliminar tipo sanción	Elimina un tipo de sanción.	MEDIA	BAJA
RF61	Insertar nivel	Inserta un nivel.	BAJA	BAJA
RF62	Modificar nivel	Modifica un nivel.	BAJA	BAJA
RF63	Eliminar nivel	Elimina un nivel.	BAJA	BAJA
RF64	Listar nivel	Lista los niveles.	BAJA	BAJA

Libro del graduado

No.	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF65	Insertar página en el libro de graduado	Inserta páginas en el libro de graduado.	BAJA	BAJA
RF66	Modificar página en el libro de graduado	Modifica páginas en el libro de graduado.	MEDIA	BAJA
RF67	Eliminar página en el libro de graduado	Elimina páginas en el libro de graduado.	MEDIA	BAJA
RF68	Insertar nueva graduación	Inserta una nueva graduación.	ALTA	MEDIA
RF69	Modificar graduación	Modifica una graduación.	MEDIA	MEDIA
RF70	Eliminar graduación	Elimina una graduación.	MEDIA	MEDIA
RF71	Listar graduaciones	Lista las graduaciones.	BAJA	MEDIA

Participación en las actividades

No.	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF72	Insertar actividad	Inserta una actividad.	ALTA	MEDIA
RF73	Modificar actividad	Modifica una actividad.	MEDIA	MEDIA
RF74	Eliminar actividad	Elimina una actividad.	MEDIA	MEDIA
RF75	Listar actividades	Lista las actividades.	BAJA	MEDIA
RF76	Asignar actividad a un estudiante	Asigna una actividad a un estudiante.	ALTA	MEDIA

RF77	Asignar una actividad a estudiantes	Asigna una actividad a varios estudiantes.	ALTA	ALTA
RF78	Listar participantes en actividad	Lista los participantes en una actividad.	MEDIA	MEDIA
RF79	Listar eventos	Lista los eventos.	BAJA	MEDIA

Ayudantía

No.	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF80	Listar oferta de asignaturas	Lista todas las ofertas disponibles.	BAJA	MEDIA
RF81	Insertar oferta de asignatura	Adiciona una nueva oferta.	MEDIA	MEDIA
RF82	Modificar oferta de asignatura	Edita las ofertas existentes.	MEDIA	MEDIA
RF83	Listar solicitud de ayudantía	Lista las solicitudes de ayudantía realizada por los estudiantes.	BAJA	MEDIA
RF84	Insertar solicitud de ayudantía	Adiciona una nueva solicitud para ocupar el cargo de alumno ayudante.	ALTA	MEDIA
RF85	Modificar solicitud de ayudantía	Modifica la solicitud para ocupar el cargo de alumno ayudante.	MEDIA	MEDIA
RF86	Eliminar solicitud de ayudantía	Elimina la solicitud para ocupar el cargo de alumno ayudante.	MEDIA	MEDIA

RF87	Listar solicitud de ayudantía	Lista las solicitudes.	BAJA	MEDIA
RF88	Insertar estado de solicitud	Insertar el estado que le corresponde a cada solicitud.	MEDIA	MEDIA
RF89	Modificar estado de solicitud	Modifica el estado de las solicitudes.	MEDIA	MEDIA
RF90	Eliminar estado de solicitud	Elimina el estado de solicitud.	MEDIA	MEDIA
RF91	Listar estado de solicitud	Lista los estados existentes para la solicitud.	BAJA	MEDIA

Administración y Seguridad

No.	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF92	Listar usuarios del sistema	Lista todos los usuarios que tienen acceso a la plataforma	BAJA	MEDIA
RF93	Editar usuario del sistema	Edita un conjunto de entradas a las cuales podrán tener acceso los usuarios que posean el rol.	MEDIA	ALTA
RF94	Listar trazas	Lista las trazas de los usuarios que interactúan en el sistema de forma global o individual.	ALTA	ALTA
RF95	Mostrar trazas del sistema	Muestra los detalles de una traza.	ALTA	ALTA
RF96	Eliminar usuario del sistema	Elimina un usuario del sistema por el administrador.	MEDIA	MEDIA

2.4.3 DESCRIPCIÓN DE REQUISITOS FUNCIONALES

A continuación se muestra la descripción de dos requisitos funcionales como ejemplo, el resto se encuentra en el expediente de proyecto.

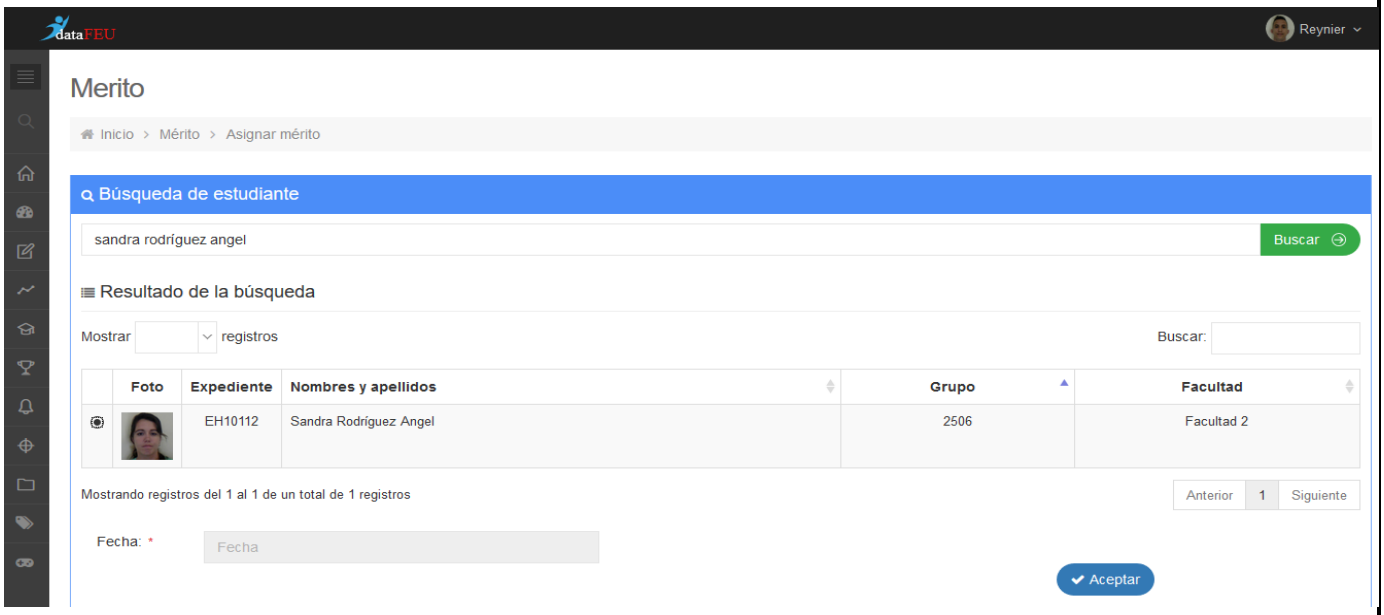
<table>
<tr><td colspan="2">Historia de usuario</td></tr>
<tr><td>Código: RF 52</td><td>Nombre historia de usuario: Asignar un mérito a varios estudiantes.</td></tr>
<tr><td colspan="2">Modificación de historia de usuario número: 2</td></tr>
<tr><td colspan="2">Referencia: RF 52 del Expediente de proyecto.</td></tr>
<tr><td>Programador: Reynier Arias Lemos.</td><td>Iteración asignada: 2</td></tr>
<tr><td>Prioridad : Alta</td><td>Puntos estimados: 4 días.</td></tr>
<tr><td>Riesgo en desarrollo: Cuidar los métodos empleados en el desarrollo del algoritmo para insertar el mérito a varios estudiantes simultáneamente.</td><td>Puntos reales: 4 días.</td></tr>
<tr><td colspan="2">Descripción:

El proceso comienza cuando se realiza la actividad o el evento que otorgue un mérito.
El presidente de brigada procede a insertar un mérito a través del menú inicio, en Méritos. En la ventana activa se listaran todas las instancias de méritos existentes. Luego se selecciona de la lista de nomencladores el mérito que se desea asignar al grupo que le corresponde a dicho presidente de brigada. Luego se selecciona el listado de los estudiantes a recibir el mérito seleccionado previamente, marcándolo por un radio-botón existente en la parte izquierda de la tabla de estudiantes y a la derecha se muestra el listado de estudiantes a los cuales ya ha sido asignado este mérito. Para finalizar se da clic en aceptar y así concluye el proceso. El sistema además mostrará un mensaje de notificación indicando que el proceso concluyó satisfactoriamente.
En caso de tener otro cargo a nivel de facultad o de universidad se prosigue buscando a los estudiantes (uno por uno) y seleccionando el botón aceptar.</td></tr>
<tr><td colspan="2">Observaciones:

Cada usuario tendrá según su rol, un nivel de acceso diferente. El presidente de brigada solo podrá asignar méritos a los estudiantes de su brigada. Los dirigentes de facultad a los usuarios de su facultad</td></tr>
</table>

y los dirigentes de universidad a todos los usuarios del sistema. Se debe de evitar dejar campos vacíos al asignar un mérito.

Prototipo de interfaz:



Historia de usuario

Código: RF 9	**Nombre historia de usuario:** Insertar un cargo
Modificación de historia de usuario número: 3	
Referencia: RF 9 del Expediente de proyecto.	
Programador: Sandra Rodríguez Ángel	**Iteración asignada:** 2
Prioridad : Alta	**Puntos estimados:** 3 días.
Riesgo en desarrollo: La no existencia de conexión con el sistema ALMA, para la obtención de datos primarios de estudiantes.	**Puntos reales:** 2 días.

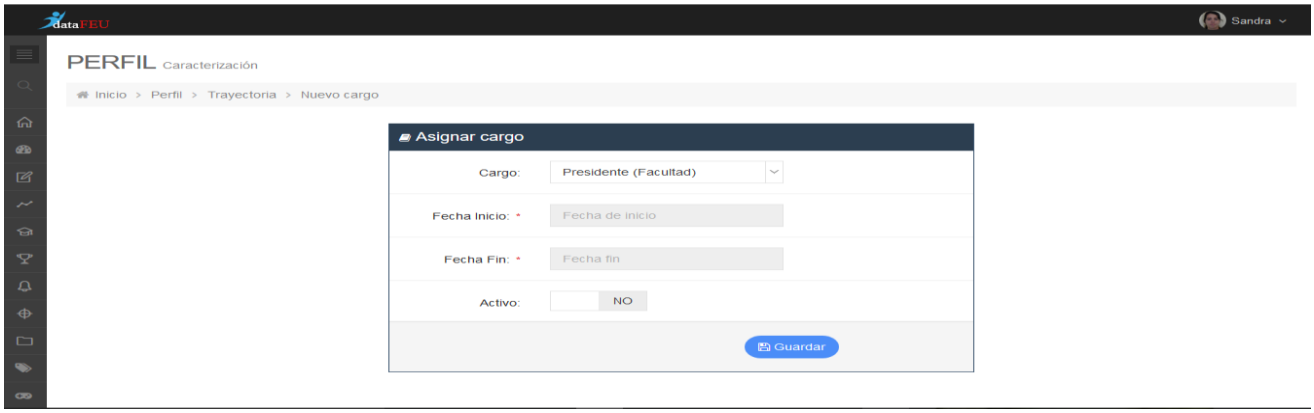
Descripción:

Una vez en el perfil del estudiante al que se le asignara el cargo se selecciona en el menú del perfil la opción de trayectoria donde mostrara una tabla con el botón Añadir, en la parte superior derecha de dicha tabla. Luego en la ventana activa se llenan los campos solicitados y se da clic en el botón guardar en la parte inferior derecha.

Observaciones:

Cada usuario tendrá según su rol, un nivel de acceso diferente. El presidente de brigada solo podrá asignar méritos a los estudiantes de su brigada. Los dirigentes de facultad a los usuarios de su facultad y los dirigentes de universidad a todos los usuarios del sistema. Se debe de evitar dejar campos vacíos al insertar un cargo.

Prototipo de interfaz:



2.4.4 REQUISITOS NO FUNCIONALES

Como su nombre sugiere, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema (42).

Usabilidad

- La aplicación debe estar dirigida a registrar la información referente al proceso de evaluación estudiantil de la FEU.
- Solo se mostrarán a los usuarios aquellas acciones o informaciones a las que por su responsabilidad o rol dentro del negocio necesitan acceder mostrando en la vista los íconos y menús correspondientes.

- Las vistas del sistema deben indicar en cada momento la acción que se está realizando así como los íconos deben estar representados por una imagen acorde a la acción que se realiza mediante el mismo.
- Para el uso del sistema se requiere una PC cliente con un navegador web, preferentemente Firefox en su versión 40 o superior y Chrome a partir de la versión 47.

Confiabilidad

- El sistema puede permanecer inactivo durante 10 minutos. Al cumplirse este término se cerrará la sesión teniendo que autenticarse el usuario nuevamente.
- El sistema de autenticación recogerá datos necesarios de los usuarios y tendrá control de identidades de los mismos.
- Debe proveer algún mecanismo seguro de encriptación y trasferencia de datos.
- Se validan los datos recibidos de otras aplicaciones externas.
- Se debe mantener una seguridad a nivel de usuarios y contraseñas codificadas para el acceso a la base de datos.
- El servidor de aplicaciones y de base de datos deberá mantener una seguridad mediante firewall para proteger el código y la información.

Soporte

- La base de datos que utilizará el sistema como medio de almacenamiento de la información estará soportada sobre un gestor de bases de datos PostgreSQL en su versión 9.1, permitiéndole interactuar con otros sistemas estableciendo vías de compatibilidad.
- Se debe desplegar en un servidor Apache con versión 2.4, que tenga CentOS en su versión 7.0.
- Debe ser capaz de obtener de sistemas externos (aplicaciones del ecosistema MAYA) información como los datos relacionados con la situación académica, así como del LDAP UCI la autenticación para acceder a la aplicación.
- Debe permitir brindar información a través de servicios web públicos a sistemas externos.

2.4.5 VALIDACIÓN DE REQUISITOS

La validación de requisitos trata de mostrar que los levantados en la fase inicial del proceso de desarrollo del sistema realmente definen el software que el cliente desea. Es importante debido a que los errores en el documento de requisitos pueden conducir a importantes costes al repetir el trabajo cuando son descubiertos durante el desarrollo o después de que el sistema esté en uso.

A decir de Somerville durante el proceso de validación de requisitos, se deben llevar a cabo verificaciones que comprenden:

- Validez: un usuario puede pensar que se necesita un sistema para llevar a cabo ciertas funciones, sin embargo, el razonamiento y el análisis pueden identificar que se requieren funciones adicionales o diferentes. Los sistemas tienen diversos puntos de vistas con diferentes necesidades y para cualquier conjunto de requisitos es inevitable un compromiso en el entorno del negocio.
- Consistencia: no deben contradecirse. Esto es, no debe haber restricciones o descripciones contradictorias en la misma función del sistema.
- Completitud: Deban definirse todas las funciones y restricciones propuestas por el usuario del sistema.
- Realismo: utilizando el conocimiento de la tecnología existente, los requisitos deben verificarse para asegurar que se pueden implementar. Estas verificaciones también deben tener en cuenta el presupuesto y la confección de agendas para el desarrollo del sistema.
- Verificabilidad: para reducir la posibilidad de discusiones entre el cliente y el contratista, los requisitos del sistema siempre deben redactarse de tal forma que sean verificables. Esto significa que se debe poder escribir un conjunto de pruebas que demuestren que el sistema a entregar cumple cada uno de los requisitos especificados.

Todos los indicadores mencionados al inicio de este epígrafe fueron seguidos, apegándose al estándar y las buenas prácticas de ingeniería de software definido por Somerville (42).

2.5 ARQUITECTURA DEL SISTEMA

La arquitectura representa un modelo relativamente pequeño, intelectualmente tratable, de la forma en que un sistema se estructura y sus componentes se entienden entre sí; este modelo es transferible a través de sistemas; en particular, se puede aplicar a otros sistemas que exhiben requerimientos parecidos y puede promover reutilización en gran escala. El diseño arquitectónico soporta reutilización de grandes componentes o incluso de framework en los que se pueden integrar componentes (44).

Según Pressman, la arquitectura de software es la representación que capacita al ingeniero del software para: analizar la efectividad del diseño para la consecución de los requisitos fijados, considerar las alternativas arquitectónicas en una etapa en la cual hacer cambios en el diseño es relativamente fácil, y reducir los riesgos asociados a la construcción del software (16).

Los patrones de arquitectura están relacionados fundamentalmente con la estructura de un sistema de software. Especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. Describen un problema particular y recurrente

del diseño, que surge en un contexto específico, y presenta un esquema genérico y probado de su solución (45).

Patrón Modelo Vista Controlador (MVC)

Symfony 2 basa su funcionamiento interno en el estilo Modelo-Vista-Controlador (MVC), uno de los más usados por los frameworks web (46), el mismo separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres capas o componentes distintos. El patrón MVC se utiliza frecuentemente en aplicaciones web, donde la vista es la página HTML que se visualiza en el navegador y el código que proporciona de datos dinámicos a la página, el modelo es el Sistema de Gestión de Base de Datos y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista, enviarlos al modelo y manejar también las respuestas del modelo y transmitirlas hacia la vista (45).

La siguiente imagen muestra lo anteriormente descrito:

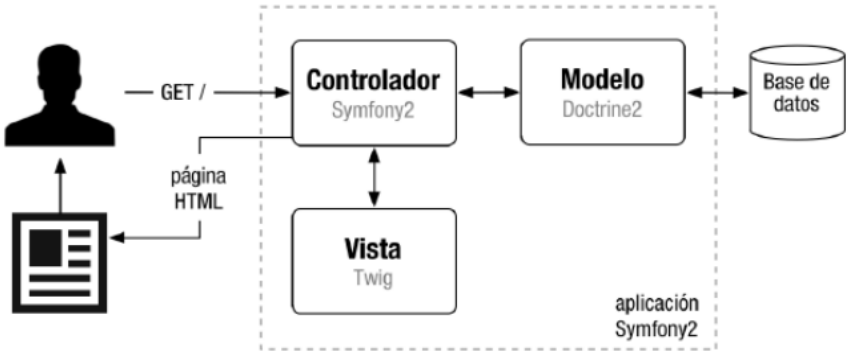


Imagen 1 Diagrama MVC según Symfony2 (46).

Modelo

El modelo es un conjunto de clases, en Symfony2 (47) se encuentra en el directorio src/AppBundle/Entity. Representan la información del mundo real que el sistema debe procesar. El modelo desconoce la existencia de las vistas y del controlador. Esta es la recepción específica del dominio de la información sobre la cual funciona la aplicación. El modelo es una forma de llamar la capa de dominio. La lógica de dominio añade significados a los datos (45).

Vista

Las vistas son el conjunto de clases que se encargan de mostrar al usuario la información contenida en el modelo. Una vista está asociada a un modelo, pudiendo existir varias vistas asociadas al mismo modelo. Una vista obtiene del modelo solamente la información que necesita para desplegar y se actualiza cada vez que el modelo del dominio cambia por medio de notificaciones generadas por el modelo de la aplicación (45). Todas las vistas en la aplicación se encuentran en el directorio app/Resources/views y cada una de estas clases posee la extensión html.twig.

Controlador

El controlador es un objeto que se encarga de dirigir el flujo del control de la aplicación debido a mensajes externos, como datos introducidos por el usuario u opciones del menú seleccionadas por él. A partir de estos mensajes, el controlador se encarga de modificar el modelo o de abrir y cerrar vistas. El controlador tiene acceso al modelo y a las vistas, pero las vistas y el modelo no conocen de la existencia del controlador (45). El framework Symfony2 define su directorio genérico en src\AppBundle\Controller encontrándose todas las clases controladoras en esta carpeta.

2.6 MÓDULOS DEL SISTEMA

Teniendo en cuenta los requisitos definidos en el epígrafe 2.2, el software se divide en ocho módulos. A continuación se describen cada uno de ellos:

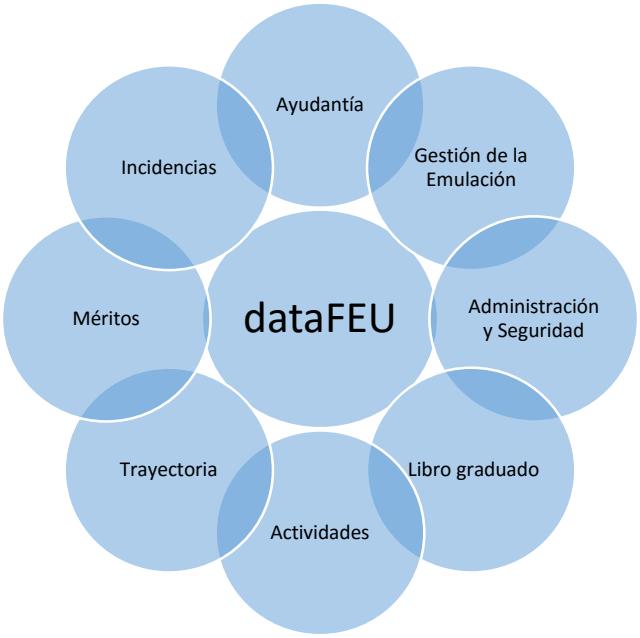


Imagen 2 Módulos del sistema.

- **Administración y seguridad:** se encarga del funcionamiento y seguridad del sistema utilizando el bundle ZTEC de Symfony2, el cual se integra con los directorios activos. Se empleó el patrón de control de acceso basado en roles (por sus siglas en inglés RBAC), el mismo es una función de seguridad, donde se establecen las funcionalidades especificas a las cuales puede acceder cada usuario. Enfoca su interés principalmente en determinar qué usuarios y qué grupos de usuarios pueden ejecutar qué tipo de operación sobre qué tipo de recurso.
- **Libro de graduado:** gestiona la página del graduado de cada estudiante que se encuentra en el año terminal de la carrera.

- **Actividades:** gestiona las actividades en el sistema (creación, modificación y eliminación) así como la asignación de los estudiantes que participaron en estas, la cual se notifica vía correo electrónico. Además genera reportes nominales y estadísticos sobre la participación de los estudiantes.
- **Trayectoria estudiantil:** gestiona los cargos (creación, modificación y eliminación), las evaluaciones y las bajas a los dirigentes estudiantiles. En caso de asignación de cargos estudiantiles al estudiante le es notificado a través de un correo electrónico.
- **Méritos:** gestiona los méritos y su asignación a los estudiantes que obtuvieron los mismos la cual se realiza mediante el envío de correo electrónico.
- **Incidencias disciplinarias:** gestiona las incidencias y su asignación a los estudiantes sancionados notificando estas a través de correos electrónicos.
- **Ayudantía:** es el encargado de lanzar las convocatorias a formar parte del movimiento de alumnos ayudantes y de gestionar la incorporación al movimiento.
- **Gestión de la emulación:** gestiona el proceso de emulación a todos los niveles, desde la brigada hasta la universidad.

2.7 PATRONES DE DISEÑO

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. Existen varios tipos de patrones de diseños entre los que se encuentra el patrón GRASP el cual se caracteriza a continuación:

GRASP es un acrónimo de General Responsibility Assignment Software Patterns (patrones generales de software para asignar responsabilidades). Estos representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones (48).

Symfony 2 evidencia el uso de varios patrones de los que concibe para que el programador se vea obligado a aplicarlos. Entre ellos se tienen:

- Experto: El objetivo de este patrón es asignar las responsabilidades a las clases que tengan la información necesaria para realizar la tarea. En Symfony 2 en las clases del modelo existen dos tipos de clases las clases repositorio y las clases entidades, estas últimas que son las clases de abstracción de los datos son las que trabajan directamente con la base de datos, son las que tienen los registros necesarios para trabajar con la base de datos, es en este vínculo donde se muestra el uso de este patrón (48).
- Creador: Este patrón asigna la responsabilidad de crear instancias de un objeto a una clase A a una clase B si cumple una de estas condiciones:

B agrega objetos de A.

B contiene objetos de A.

B registra instancias de objetos de A.

B contiene los datos de inicialización del objeto A (48).

Se utiliza en las clases del modelo para la creación de instancias que estén relacionadas mediante relaciones de uno a muchos como estrategia y objetivos, en la cual una estrategia contiene y registra instancias de objetivos.

- Controlador: Este patrón se encarga de la asignación de la responsabilidad de gestionar los eventos en un sistema a clases específicamente diseñadas para ello como: un controlador de fachada, un controlador de tareas o un controlador de casos de uso. Este se encarga de las validaciones, la seguridad, etc. El controlador no ejecuta por sí mismo estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Este patrón se pone de manifiesto en las clases cuyo nombre terminan en Controller, además la arquitectura del Symfony 2 (MVC) brinda una capa específicamente para los controladores en la que se especifica la presencia de este patrón (48).
- Alta Cohesión: Se encarga de solucionar el problema de cómo mantener el sistema con un bajo nivel de complejidad mediante la asignación de responsabilidades que garanticen una alta cohesión. Un componente con responsabilidades muy relacionadas a este y que no hace una gran cantidad de trabajo, tiene alta cohesión. Los problemas que presentan los componentes con baja cohesión son:
  - Se vuelven difíciles de mantener, entender y de reutilizar.
  - Son muy afectadas por los cambios en el sistema (48).

  Este patrón se utilizó en la asignación de las responsabilidades de los distintos controladores como ejemplo de ello el PerfilController, donde las acciones que están más fuertemente vinculadas son las que están en los controladores garantizando así una alta cohesión.

- Bajo acoplamiento: Se asignan las responsabilidades de manera que el acoplamiento sea bajo. Un componente con bajo acoplamiento no mantiene fuertes dependencias con otros componentes. Cuando este es alto se pueden presentar estos problemas**:**

  - Los cambios locales son difíciles de realizar.
  - Son difíciles de analizar de forma independiente.
  - Se vuelven difíciles de reutilizar (48).

Con el propósito de que las clases estuvieran vinculadas solo de forma imprescindible unas con otras y evitar los problemas mencionados, se utilizó este patrón. Definiendo para cada una sus respectivos métodos y atributos con el fin de que el acoplamiento entre ellas fuera débil, para lograr la reutilización y el soporte. La funcionalidad de este patrón se aplica en todas las clases desarrolladas.

2.8 CONCLUSIONES PARCIALES

A partir de los elementos estudiados en el capítulo se concluye que la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas lleva a cabo un gran número de procesos relacionados con su funcionamiento, los que pueden ser agrupados en ocho módulos. Las técnicas de obtención de requisitos seleccionadas arrojaron 96 funcionales y 14 no funcionales, siendo los mismos descritos en el capítulo utilizando la metodología XP para el desarrollo del software. A partir de la identificación de los patrones de diseño a utilizar se crearon las condiciones necesarias para enfrentarse al desarrollo de la aplicación web.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS

3.1 INTRODUCCIÓN

En el presente capítulo se analizan los aspectos relacionados con la implementación y las pruebas realizadas a la aplicación. Se describen los métodos y técnicas para la realización de las pruebas con el propósito de validar la propuesta de solución, la evaluación de su ejecución y los resultados obtenidos. Se exponen los elementos fundamentes del modelo de datos y de la integración del sistema propuesto con el resto del ecosistema MAYA.

3.2 MODELO DE DATOS

Para describir la estructura de una base de datos se define el modelo de datos, el cual se refiere a tipos de datos, sus vínculos y las restricciones que deben cumplir estos (49).

La solución propuesta cuenta con 77 tablas en general, de estas 19 son nomencladores, diferenciando unas de otras a través del prefijo Nm (para los nomencladores) y Tb (a las tablas). Se tuvo en cuenta el patrón de diseño de base de datos, llave subrogada, a partir de la generación de una llave primara única para cada entidad en vez de usar un atributo identificador en el contexto dado. Se utilizó el tipo de dato entero en las llaves primarias, lo que permite que las tablas sean más fáciles de consultar por el identificador, dado que se conoce el mismo en cada una de ellas. La notación utilizada para nombrar las tablas fue CamelCase, en su variante UpperCamelCase empleando la letra inicial de cada palabra en mayúscula.

La base de datos se encuentra en 3ra Forma Normal pues todas sus tablas contienen una llave primaria, los atributos son atómicos y no contiene dependencias funcionales transitivas ni parciales. De esta manera se evita la redundancia de los datos y los problemas que puedan ocurrir con las actualizaciones de estos en las tablas, protegiendo así la integridad de los mismos.

La imagen, muestra un fragmento del modelo Entidad-Relación donde se puede apreciar las relaciones entre las tablas que conforman el módulo Trayectoria.

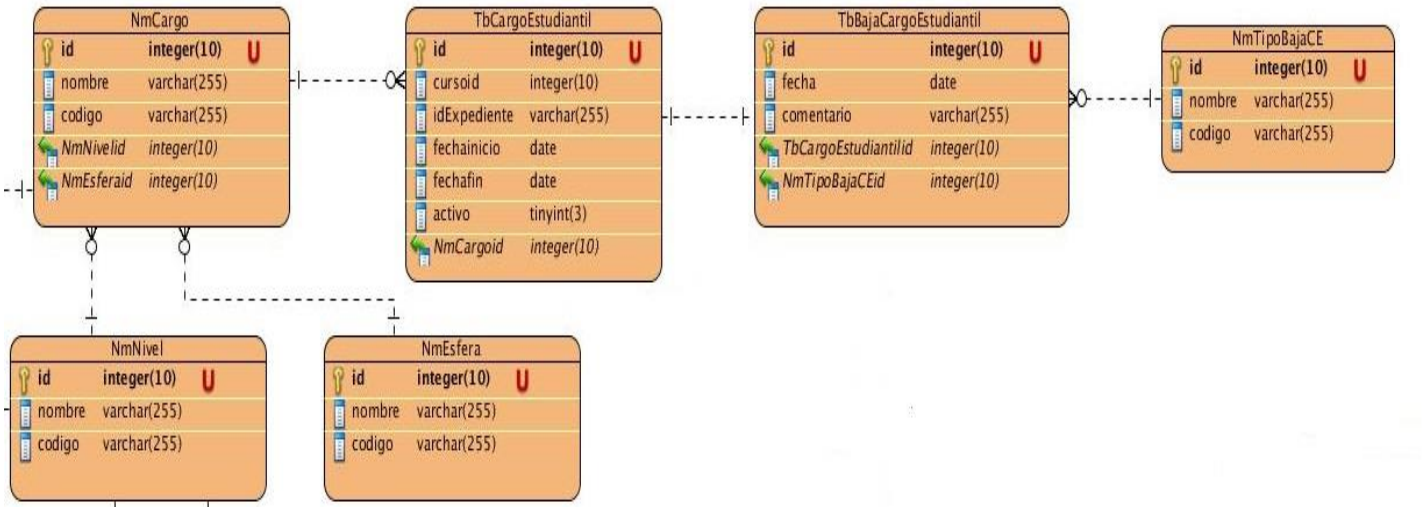


Imagen 3 Tabla Módulo Trayectoria de la Base de Datos.

3.3 INTEGRACIÓN CON OTROS SISTEMAS DE MAYA

- **Representational State Transfer**

La Transferencia de Estado Representacional (REST, por sus siglas en inglés) define una colección de principios arquitectónicos por los cuales se diseñan servicios web haciendo foco en los recursos del sistema, incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por Hypertext Transfer Protocol (HTTP) (50) hacia clientes escritos en diversos lenguajes. Confía en documentos orientados al usuario que define las direcciones de petición y las respuestas. Distingue los recursos de solo lectura (GET), de los modificables (POST, PUT, DELETE) (51).

Entre sus principales ventajas se destacan:

- Bajo consumo de recursos.
- Las instancias del proceso son creadas explícitamente.
- Generalmente fácil de construir y adoptar.

- **Simple Object Access Protocol (SOAP)**

Es un protocolo que permite la comunicación entre aplicaciones a través de mensajes por medio de Internet. Es independiente de la plataforma y del lenguaje (52). Define cómo dos objetos en diferentes procesos se comunican por medio de intercambio de datos XML. Su principal beneficio recae en ser fuertemente acoplado, lo que permite poder ser testeado y depurado antes de poner en marcha una aplicación.

LDAP es un protocolo para el acceso a un servicio de directorio que emplea una estructura de datos similar que opera sobre TCP/IP (Transfer Control Protocol / Internet Protocol). Este es el mecanismo que define la UCI para gestionar su Directorio Activo.

dataFEU v2 necesita datos de otros sistemas externos para complementar su funcionamiento. La siguiente imagen muestra cómo es su relación con las aplicaciones que integran al ecosistema MAYA, basándose en la arquitectura REST y los servicios de la arquitectura SOAP para la autenticación.

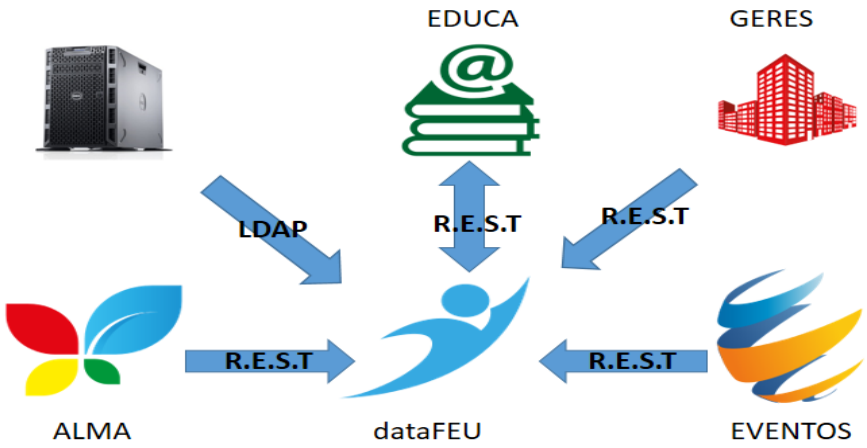


Imagen 4 Diagrama de integración.

Las siguientes tablas describen el conjunto de servicios que consume y brinda la solución propuesta.

Renderizar foto:

url	**Descripción**
http://alma.uci.cu/foto/{idExpediente}	Renderiza la foto, la petición es en forma de url. Esto posibilita que la carga de la información sea en paralelo. Entrada: idExpediente.

Tabla de servicios. Sistema para la Gestión de Datos Primarios (ALMA):

Nombre	uri	Descripción	Salida (Retorno)
Obtenercursoactual	/api/v1/curso/actual.json	Obtiene el curso actual que tiene activo el sistema.	Array con las claves: id fechaInicio

			fechaFin nombre
Obtenerpersonadadoidexpediente	/api/v1/personadadoidexpedientes/{idExpediente}.json	Obtiene una persona dado el idExpediente pasado por paráremo. Devuelve los datos generales de la persona, que puede existir o no en el curso actual. Entrada: idExpediente	<u>Array con las claves:</u> idExpediente usuario pNombre sNombre pApellido sApellido solapin sexo direccion tipoPersona municipio provincia activo En el caso que sea profesor devuelve: cargo *En caso de no encontrar datos, retorna false.*
Obtenergruposdocentes	/api/v1/gruposdocentes.json	Obtiene todos los grupos activos	<u>Array de array con las claves:</u> id codigo nombre anno facultad idFacultad
Obtenergruposdadofacultad		Obtiene todos los grupos activos y que sean de la	<u>Array de array con las claves:</u> id

		facultad entrada por parámetro. Entrada: idFacultad	codigo nombre anno facultad idFacultad *En caso de no encontrar datos, retorna false.*
Obtenerfacultades	/api/v1/facultades.json	Obtiene todas las facultades activas	Array de array con las claves: id codigo nombre siglas
Obtenerestudiantesdadogrupoy Curso	/api/v1/estudiantesdadogrupos/{idGrupo}/cursos/{idCurso}.json	Obtiene todos los estudiantes de un grupo y curso entrado por parámetro. Entrada: idGrupo, idCurso	Array de array con las claves: idExpediente usuario pNombre sNombre pApellido sApellido solapin sexo direccion tipoPersona area municipio provincia facultad apto estadoDocente curso activo

			En caso de no encontrar datos, retorna false.
Obtenerapartamentos	/api/v1/apartamentos.json	Obtiene los apartamentos	Array de array con las claves: id numero edificio idEdificio *En caso de no encontrar datos, retorna false.*
Obtenerapartamentosdadoedificio	/api/v1/apartamentodadoedificios/{idEdificio}.json	Obtiene los apartamentos dado un edificio entrado por parámetro Entrada: idEdificio	Array de array con las claves: id numero edificio idEdificio *En caso de no encontrar datos, retorna false.*
Obteneredificios	/api/v1/edificios.json	Obtiene los edificios del sistema	Array de array con las claves: id numero cantAptos *En caso de no encontrar datos, retorna false.*
Obtenerasignaturas	/api/v1/asignaturasactivas.json	Obtiene las asignaturas del plan de estudio activo en la carrera.	Array de array con las claves: id codigo nombre

			anno disciplina departamento planEstudio *En caso de no encontrar datos, retorna false.*
ObtenerCredencialSeguridadIdExpediente	/api/v1/credencialseguridadidexpedientes/{idExpediente}.json	Obtiene las credenciales de seguridad de una persona del curso activo. Son un conjunto de datos asociados a la persona.	Array de array con las claves: pNombre sNombre pApellido sApellido solapin tipoPersona idArea idFacultad activo *En caso de no encontrar datos, retorna false.*

Tabla de servicios. Sistema para la Gestión de las Estrategias Educativas (EDUCA):

Nombre	**uri**	**Descripción**	**Salida (Retorno)**
ObtenerComisionesDisciplinariaEstudiantedadoidExpedienteyCuro	/api/v1/comisionesdisciplinariasdadoidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene las comisiones disciplinarias en las que ha incurrido un estudiante. Entrada: idExpediente idCurso	Array de array con las claves: motivo fecha realizadaPor nombre *Retorna false en caso de no encontrar datos.*

ObtenerMeritosEstudiantedadoIdExpedienteyCurso	/api/v1/meritosestudiantedadoidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todos los méritos que ha recibido un estudiante dado un curso pasado por parámetro. Entrada: idExpediente idCurso	Array de array con las claves: nombre fecha tipoMerito nivel *Retorna false en caso de no encontrar datos.*
ObtenerIndisciplinasEstudiantedadoIdExpedienteyCurso	/api/v1/indisciplinasestudiantedadoidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todas las indisciplinas de las cuales ha sido objeto un estudiante dado un curso pasado por parámetro. Entrada: idExpediente idCurso	Array de array con las claves: nombreDenunciante motivo tipoIndisciplina nivel fechaDenuncia medida *Retorna false en caso de no encontrar datos.*
ObtenerProfesoresColectivoPedagogicodadoGrupoyCurso	/api/v1/profesorescolectivopedagogicodadogrupoys/{idGrupo}/cursos/{idCurso}.json	Obtiene todos los profesores del colectivo pedagógico dado un grupo. Devuelve solo los profesores del colectivo pedagógico que esté activo en el sistema en ese curso. Entrada: idGrupo idCurso	Array de array con las claves: idExpediente idAsignatura nombreAsignatura plantilla esGuia *Retorna false en caso de no encontrar datos.*

Tabla de servicios. Sistema para la Gestión de Eventos Estudiantiles:

Nombre	uri	Descripción	Salida (Retorno)
ObtenerParticipacionenEventosdadoIdExpedienteyCurso	/api/v1/participacioneventosdadoidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene los trabajos presentados en los diferentes eventos dado el idExpediente y el curso entrado por parámetro. Entrada: idExpediente, idCurso	Array de array con las claves: nombreTrabajo principal tipoTrabajo presentado fechaEvento nombreEvento resultado nivel urlDocumento *(false en caso de que no tenga)* *En caso de no tener ninguna participación retorna false.*

Tabla de servicios. Sistema para la Gestión de la información asociada a la Residencia Estudiantil de la UCI (GERES):

Nombre	uri	Descripción	Salida (Retorno)
ObtenerCuarteleriadadoIdExpedienteyCurso	/api/v1/cuarteleriasidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene las cuartelería de un estudiante dado su IdExpediente y el Curso dado por parámetro. Entrada: idExpediente, idCurso	Array de array con las claves: evaluacion fecha observaciones id *Retorna false en caso de no encontrar datos*
ObtenerEvaluacionesFrencuentesdadoIdExpedienteyCurso	/api/v1/evaluacionesfrecuentesidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todas las evaluaciones frecuentes de los estudiantes dado su idExpediente y el curso. Entrada: idExpediente,	Array de array con las claves: evaluacion fecha observaciones id

		idCurso	*Retorna false en caso de no encontrar datos*
ObtenerEvaluacionesIntegralesdadoIdExpedienteyCurso	/api/v1/evaluacionesintegralesidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todas las evaluaciones integrales de los estudiantes dado su idExpediente y el curso. Entrada: idExpediente, idCurso	Array de array con las claves: evaluacion fecha observaciones id *Retorna false en caso de no encontrar datos*
ObtenerIndisciplinasdadoIdExpedienteyCurso	/api/v1/indisciplinasidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todas las indisciplinas los estudiantes dado su idExpediente y el curso. Entrada: idExpediente, idCurso	Array de array con las claves: evaluacion fecha observaciones Id *Retorna false en caso de no encontrar datos*

Tabla de servicios. Sistema dataFEU v2.0:

Nombre	**uri**	**Descripción**	**Salida (Retorno)**
ObtenerActividadesDataFEUIdExpedienteyCurso	/api/v1/participacionactividadesdadoidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todas las actividades en las que ha participado un estudiante dado su idExpediente y el Curso	Array de array con las claves: nombre fecha nivel esfera resultado participacion *Retorna false en caso de no encontrar datos*
ObtenerMeritosDataFEUIdExpedienteyCurso	/api/v1/meritosidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todos los méritos de un estudiante dado su Idxpediente y Curso(Méritos de la FEU)	Array de array con las claves: nombre fecha nivel tipoMerito *Retorna false en caso de no encontrar datos*
ObtenerIndisciplinasDataFEUIdExpedienteyCurso	/api/v1/incidenciasidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene todas las incidencias de un estudiante dado su IdExpediente y el	Array de array con las claves: motivo fecha

		Curso(Indisciplinas de la FEU)	nivel tipoSancion *Retorna false en caso de no encontrar datos*
ObtenerTrayectoriaFEUIdExpedienteyCurso	/api/v1/trayectoriaestudiantilidexpedientes/{idExpediente}/cursos/{idCurso}.json	Obtiene la trayectoria de un estudiante dado su IdExpediente y el Curso(Cargos en la FEU)	Array de array con las claves: cargo nivel fechaInicio fechaFin *Retorna false en caso de no encontrar datos*

3.4 PRUEBAS DE SOFTWARE

Con el fin de entregar al cliente un producto de alta calidad, se diseñan una serie de casos de pruebas que tienen una gran probabilidad de detectar errores. Para la elaboración de estos casos de pruebas, existen un conjunto de técnicas, las cuales brindan una guía para comprobar la lógica interna de los componentes y que verifiquen los datos de entrada y salida del sistema, con el propósito de detectar errores y comprobar el rendimiento del mismo (16).

3.4.1 NIVELES DE PRUEBA

Son diferentes ángulos de verificar y validar un producto de software. Existen diferentes niveles que se le aplican al producto, entre ellas se mencionan las utilizadas en la presente investigación (53):

Pruebas unitarias: Se le realiza a pequeñas unidades o fragmentos de código, que no son más que los métodos o funciones del sistema. Estas pruebas revelan al igual que otras la presencia de errores en el sistema, pero no informa de errores en la integración de las partes, ni errores de rendimiento o problemas derivados del sistema sobre el que está ejecutándose el programa. El objetivo de las pruebas unitarias es el aislamiento de partes del código y la demostración de que estas partes no contienen errores.

Pruebas de sistema: Se realiza cuando el probador no tiene acceso al código fuente, que es lo más frecuente. Estas pruebas se realizan en muchos casos para estudiar el comportamiento del sistema bajo determinada condiciones.

Pruebas de integración: Es ejecutada para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso. Se prueba un paquete o un conjunto de paquetes del modelo de implementación. Estas pruebas descubren errores o incompletitud en las especificaciones de las interfaces de los paquetes. Esta prueba debe ser responsabilidad de

desarrolladores y de independientes, sin solaparse las pruebas. Es el proceso de combinar y probar múltiples componentes juntos. El objetivo es tomar los componentes probados en unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño.

Se llama integración incremental cuando el programa se construye y se prueba en pequeños segmentos en los que los errores son más fáciles de aislar y corregir, es más probable que se pueda probar completamente las interfaces y se puede aplicar un enfoque de prueba sistemática.

Hay dos estrategias de integración incremental:

- Integración Descendente (Top-Down)

Se integran los módulos moviéndose hacia abajo por la jerarquía de control. Comenzando por el módulo principal, los módulos subordinados se van incorporando a la estructura bien, en forma primero en profundidad, que integra todos los módulos de un camino de control principal de la estructura, o primero en anchura, que incorpora todos los módulos directamente subordinados a cada nivel, moviéndose por la estructura de forma horizontal.

- Integración Ascendente (Bottom-Up)

Empieza la construcción y la prueba con los módulos de los niveles más bajos de la estructura del programa. Dado que los módulos se integran de abajo hacia arriba, el proceso requerido de los módulos subordinados a un nivel dado siempre está disponible y se elimina la necesidad de resguardos.

A medida que la integración progresa disminuye la necesidad de controladores de prueba diferentes. La selección de una estrategia de integración depende de las características del software y de la planificación del proyecto (54).

3.4.2 TIPOS DE PRUEBAS

Las pruebas realizadas son las encargadas de determinar el grado de conformidad de los clientes, por lo que es recomendable que estas sean ejecutadas por un conjunto de especialistas, para determinar el mayor número de errores, que en un final es el principal objetivo. Existen dos tipos de pruebas fundamentales que son las de Caja Blanca y las de Caja Negra.

- **Métodos de prueba de caja blanca**

Son pruebas estructurales. Conociendo el código y siguiendo su estructura lógica, se pueden diseñar pruebas destinadas a comprobar que el código hace correctamente lo que el diseño de bajo nivel indica y otras que demuestren que no se comporta adecuadamente ante determinadas situaciones (16).
Uno de los métodos de prueba de caja blanca es el camino básico, la cual consiste en calcular la complejidad ciclomática de un fragmento de código. Cuando se usa el camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y facilita un límite superior para el número de pruebas que se deben realizar. Esta complejidad se puede calcular de tres formas:

- El número de regiones del grafo de flujo coincide con la complejidad ciclomática.
- La complejidad ciclomática, V (G), de un grafo de flujo G, se define como: V (G)=A-N+2. Donde A es el número de aristas del grafo de flujo y N es el número de nodos del mismo.
- La complejidad ciclomática, V (G), de un grafo de flujo G también se define como V (G) = P+1. Donde P es el número de nodos predicados contenidos en el grafo de flujo G.

- **Métodos de prueba de caja negra**

Las pruebas de Caja Negra son pruebas funcionales. Se parte de los requisitos funcionales, a muy alto nivel, para diseñar pruebas que se apliquen sobre el sistema sin necesidad de conocer como está construido por dentro (caja negra). Las pruebas se realizan empleando un determinado conjunto de datos de entrada y observando las salidas que se producen para determinar si la función se está desempeñando correctamente por el sistema bajo prueba. Las herramientas básicas son observar la funcionalidad y contrastar con la especificación (16).

Una de las técnicas más usadas en pruebas de caja negra es la de partición equivalente, la cual divide el dominio de entrada de un programa en clases de datos, a partir de las cuales deriva los casos de prueba. Cada una de estas clases de equivalencia representa a un conjunto de estados válidos o inválidos para las condiciones de entrada.

Las pruebas de caja negra pretenden encontrar estos tipos de errores:

- Funciones incorrectas o ausentes.
- Errores en la interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

3.4.3 DISEÑO DE CASOS DE PRUEBA

A continuación se presenta en una tabla el diseño de caso de prueba para los requisitos que fueron descritos en el epígrafe 2.4.2. El resto de los casos de pruebas se encuentran en el expediente de proyecto.

Escenario	Descripción	Nombre actividad	Respuesta del sistema	Flujo central
RF 72 Insertar una actividad	Se inserta una nueva actividad al listado existente situando esta en tiempo y espacio.	Marcha por el 1ro de mayo	Agrega una nueva actividad a la lista de actividades	Una vez situados en la ventana de actividad se da clic en el botón Nueva y en la página que aparece ingresar los datos requeridos y por último hacer clic en el botón Guardar.
RF 77 Asignar actividad a estudiantes	Se asigna una o varias actividades a un grupo de estudiantes	Marcha por el 1ro de mayo	Asigna una actividad a un grupo de estudiante	Una vez situados en la ventana de actividad se da clic en el botón Asignar actividad y en dependencia de nivel del acceso será la interfaz que vera pues si es presidente de brigada solo tendrá acceso a visualizar los estudiante de su brigada en una columna a la izquierda y a su derecha visualizara a los estudiantes que ya poseen la actividad, en caso de ser presidente de facultad solamente tendrá acceso a los estudiantes de su facultad y en caso de ser presidente a nivel de universidad tendrá total acceso.

3.4.4 IMPLEMENTACIÓN DE LAS PRUEBAS

Como se describió en el epígrafe 3.3.2 unas de las técnicas de caja blanca es la del camino básico, que se le aplica a un fragmento de código de la aplicación. En este caso se seleccionó la funcionalidad que asigna una actividad a uno o varios estudiantes en el sistema.

Para esta funcionalidad, se identificaron los bloques de ejecución y se enumeraron, como se muestra en el Anexo#3. Se obtuvieron 9 bloques identificados por nodos, así como el camino básico mostrado en la figura, donde se observa a simple vista los nodos predicados 2,3 y 5, que son aquellos que se derivan en más de un camino.

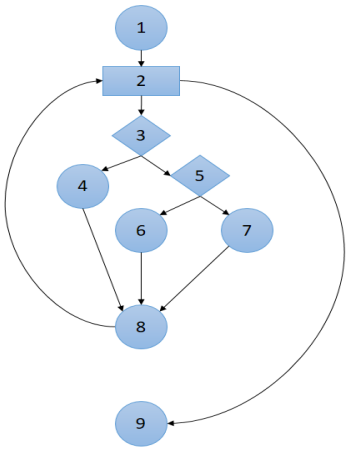


Imagen 6 Grafo de flujo del método asignar participación.

Una vez obtenido el camino básico se procede a determinar la complejidad ciclomática con una de las tres formas existentes, se utilizó V (G) = P + 1, para la cual se obtuvo tres predicados, por tanto:

V (G) = 3+1, quedando V (G) = 4. De la misma manera se puede comprobar que las otras estrategias de complejidad arriban al mismo resultado.

El resultado obtenido mediante la fórmula anterior representa los posibles caminos por los que transitará el flujo, así como la cantidad mínima de casos de pruebas que se deben realizar para el procedimiento escogido. A continuación se representa los caminos básicos que se identificaron en el flujo:

Camino1	1,2,9
Camino2	1,2,3,4,8,2,9
Camino3	1,2,3,5,7,8,2,9
Camino4	1,2,3,5,6,8,2,9

Variables para el caso de prueba Insertar nueva actividad

No.	Nombre del campo	Clasificación	Valor nulo	Descripción
1	nombre	Campo de texto	No	Pueden aparecer letras y número.
2	fecha	Date	No	Se seleccionara la fecha en que se desea que sea creada la actividad. Utilizando el formato definido d/m/y

3	fechaact	Date	No	Se selecciona la fecha de activación de la actividad para la que el formato definido es d/m/y.
4	fechadesact	Date	No	Se selecciona la fecha de desactivación de la actividad para la que el formato definido es d/m/y.
5	nivel	Lista despegable	No	Se selecciona el nivel de la actividad.
6	esfera	Lista despegable	No	Se selecciona la esfera de la actividad.
7	descripcion	Campo de texto	No	Pueden aparecer letras y numero.
8	logo	img	No	Se selecciona la imagen.
9	esevento	bool	No	Se marca si será o no un evento.

Caso de prueba1: Insertar nueva actividad

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Insertar nueva actividad	A partir de la tabla de actividades genérica que está definida, se crea la actividad la cual contempla una serie de datos para ubicarlas en tiempo y espacio.	Se inserta una nueva actividad en el listado de actividades. Muestra el listado de actividades general.	Presiona en el botón del panel izquierdo, Actividades, y luego el símbolo de (+) Nueva. Aparece una nueva vista donde solicita llenar los datos de la actividad nueva. Presiona el botón guardar en la esquina inferior izquierda.
EC 1.2 Insertar nueva actividad quedando campos vacío.	Mediante este escenario se crea una nueva actividad con campos vacíos.	El sistema muestra un mensaje de error donde dice que debe llenar los campos vacíos.	Presiona en el botón del panel izquierdo, Actividades, y luego el símbolo de (+) Nueva. Aparece una nueva vista donde solicita llenar los datos de la actividad nueva. Presiona el botón guardar en la esquina inferior izquierda. El sistema muestra el mensaje de notificación donde debe llenar los campos vacíos y vuelve a la vista de crear actividad.

Resultado de las pruebas funcionales

Se realizaron cuatro iteraciones de las pruebas funcionales utilizando los diseños de casos de prueba generados, detectándose varias no conformidades las cuales fueron clasificadas en errores de aplicación y errores ortográficos. Durante la primera iteración se encontraron 16 no conformidades, 10 referidas a las funcionalidades y resultados de las operaciones y seis problemas con la ortografía en los mensajes de la aplicación. Durante la segunda se detectaron ocho no conformidades referentes a la aplicación y dos problemas de ortografía. Durante la tercera iteración se encontraron 5 no conformidades referidas a las funcionalidades, siendo resueltas todas para la cuarta iteración

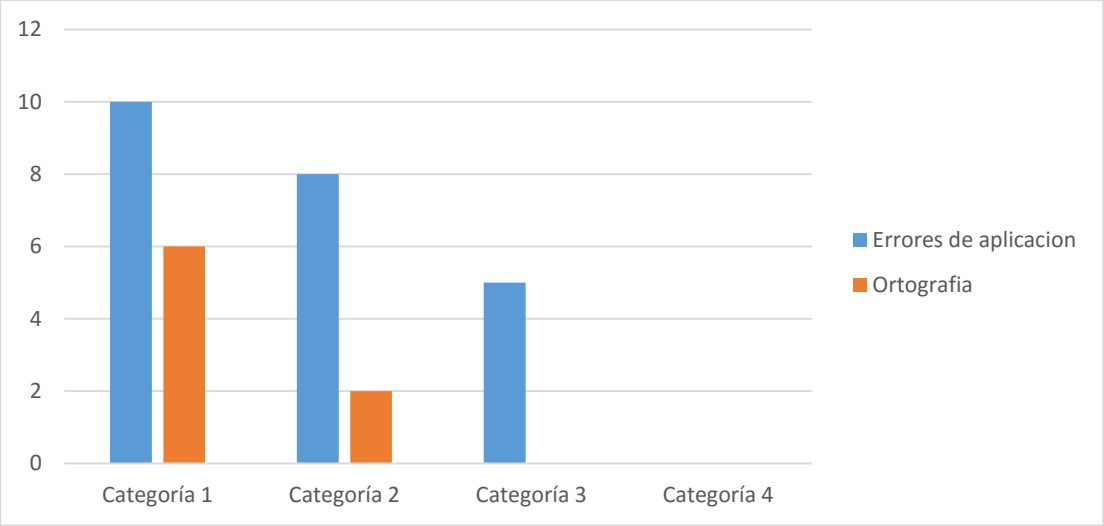


Imagen 7 Pruebas de Caja Negra.

Pruebas de integración

Se realizaron pruebas de integración a cada aplicación con la cual se relaciona dataFEU v2.0 para verificar el correcto funcionamiento de estas en el ecosistema. Se decidió hacerlas incremental y ascendente, para luego ir aumentando la cantidad de aplicaciones en pruebas, comenzando por la solución propuesta y Eventos. Se muestran seguidamente dos ejemplos de las pruebas ejecutadas, encontrándose el resto en el expediente de proyecto.

Funcionalidad	Condiciones de ejecución	Escenario de prueba	Resultado previsto	Resultado real
Realizar la inserción de un evento en la aplicación Eventos del ecosistema MAYA.	Insertar los parámetros solicitados por la herramienta seleccionada.	Envío de parámetros desde la aplicación web Eventos hasta la aplicación web	Integridad e igualdad de los datos en la aplicación dataFEU v2.0 al	Se revisa que el estudiante que participó en el evento tenga en su perfil la actividad

		dataFEU v2.0 utilizando servicios de la arquitectura rest.	comprobar que el estudiante asociado al evento tenga en su perfil el evento en cuestión.	referenciada tomada de la aplicación Eventos.
Realizar la inserción de una sanción en la aplicación GERES del ecosistema MAYA.	Insertar los parámetros solicitados por la herramienta seleccionada.	Envío de parámetros desde la aplicación web GERES hasta la aplicación web dataFEU v2.0 utilizando servicios de la arquitectura rest.	Integridad e igualdad de los datos en la aplicación dataFEU al comprobar que el estudiante asociado a la sanción tenga en su perfil la sanción en cuestión.	Se revisa que el estudiante sancionado tenga en su perfil la sanción referenciada tomada de la aplicación GERES.

Durante el desarrollo de las funcionalidades se realizaron pruebas unitarias al código para ir comprobando el funcionamiento del software siguiendo las propuestas por la metodología XP de no publicar módulos hasta que no hayan pasado todas las pruebas unitarias (26). Estas pruebas fueron realizadas por los propios desarrolladores, apoyándose en la compilación paso a paso que brinda el IDE empleado. Estas no se planificaron ni se registraron sus resultados, fueron haciéndose a medida que la solución se desarrollaba.

3.5 IMPLANTACIÓN DEL SISTEMA

Concluido el desarrollo del software, fue implantado en todas las facultades de la UCI en el curso 2015-2016. El ambiente de la aplicación cumple con estándares de diseño avanzados, con el fin de hacerlo familiar e intuitivo para los usuarios que interactúan con la misma y teniendo en cuenta los requisitos no funcionales, ya que un porciento importante de los usuarios finales no son de la rama de la informática.

La aplicación cuenta con una sección de noticias relevantes sobre lo que sucede en la universidad. En la parte derecha se cuenta con un grupo de accesos directos a las principales funcionalidades (Imagen 8).

Imagen 8 Página de inicio.

Toda la información generada por las diferentes áreas de la vida universitaria es concentrada en un expediente digital (imagen 9), donde se muestra el accionar de cada estudiante, recopilando evidencias que le servirán para el proceso final de integralidad, caracterización y de ubicación laboral en 5to año de la carrera. La información que se muestra se encuentra en un único formato, lo que permite un procesamiento eficiente.

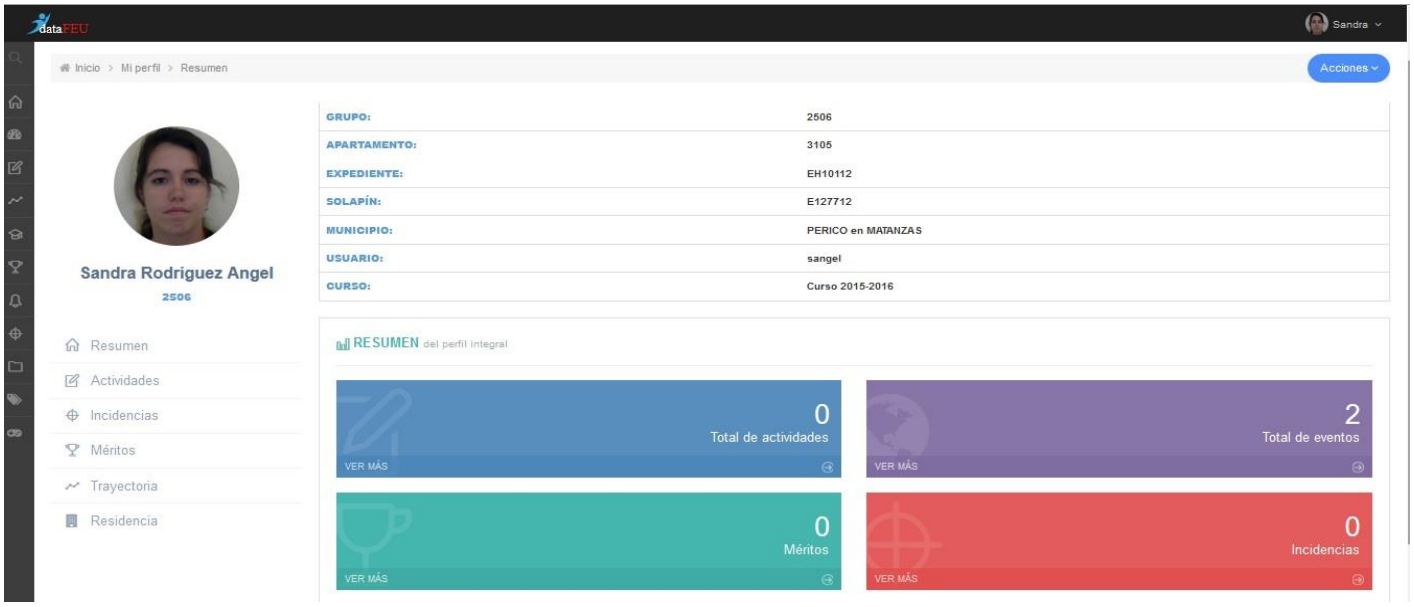


Imagen 9 Vista del perfil del estudiante.

Las opciones de la aplicación son seleccionables en el menú izquierdo, donde las funcionalidades están distribuidas de forma lógica (Imagen 10).

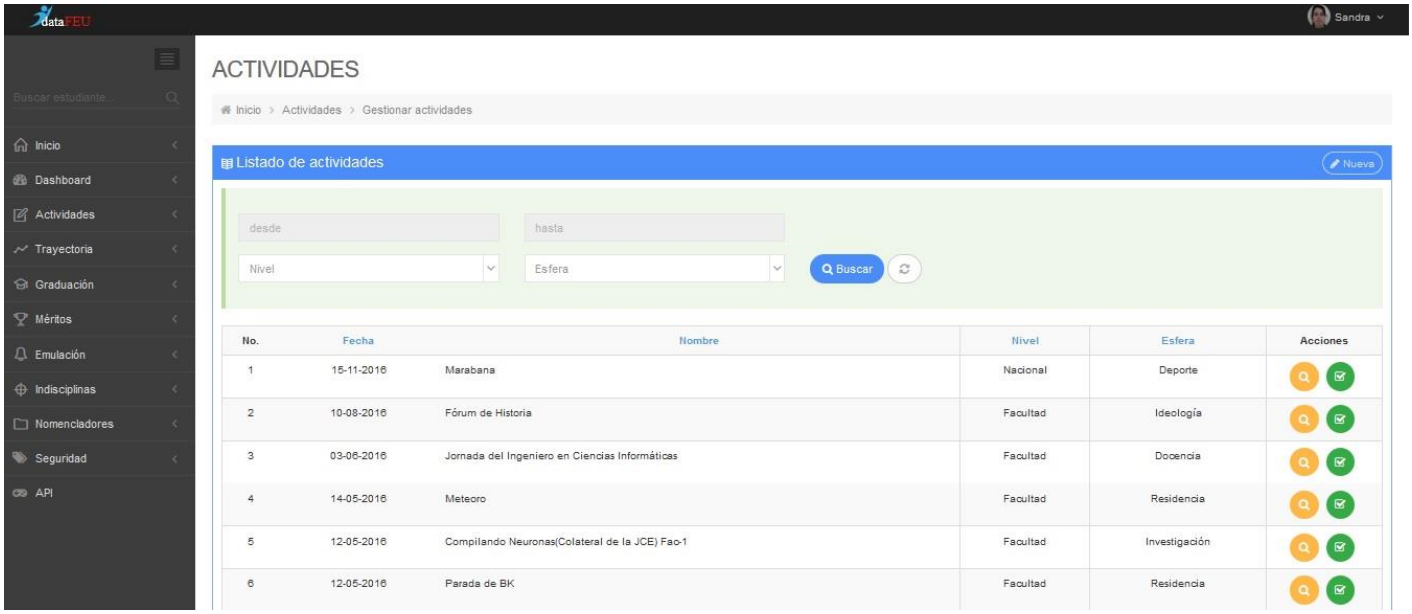


Imagen 10 Vista de asignación de actividades.

3.6 CONLCUSIONES PARCIALES

El modelo conceptual permitió mostrar las entidades importantes que se encuentran en el ámbito del problema y sus relaciones. La aplicación de los diferentes tipos de pruebas posibilitó desplegar una solución libre de errores, así como garantizar la integración de dataFEU v2.0 al ecosistema MAYA.

CONCLUSIONES GENERALES

- A partir del análisis realizado se demostró que a arquitectura de dataFEU v1.0 no permite la integración hacia el ecosistema MAYA.
- El estudio del estado del arte realizado demostró que no se conoce a nivel nacional un sistema para la gestión de los procesos de la Federación Estudiantil Universitaria que solventen estas nuevas situaciones.
- El análisis de las herramientas y tecnologías seleccionadas permitió definir el marco tecnológico en ajuste a las políticas de migración a software libre por la cual aboga el país.
- Se demuestra que el desarrollo de la aplicación permitió la integración con las demás aplicaciones del proyecto MAYA.
- La aplicación de los diferentes tipos de pruebas posibilitó desplegar una solución libre de errores.

RECOMENDACIONES

- Se propone desplegar el sistema en las universidades del país como piloto para contribuir a mejorar los procesos de gestión de la FEU.
- Desarrollar para futuras versiones del sistema el módulo escalafón de integralidad.

BILBIOGRAFÍA REFERENCIADA

1. **Guevara, Yurisander.** Juventud Rebelde. *Juventud Rebelde.* [En línea] Febrero 18, 2015. [Citado el: Febrero 17, 2016.] 23. http://www.juventudrebelde.cu/suplementos/informatica/2015-02-18/las-bases-estrategicas-de-la-informatizacion-cubana/.

2. **Informáticas, Universidad de las Ciencias.** [En línea] UCI, 2014. [Citado el: febrero 3, 2016.] www.uci.cu/?q=mision.

3. **González Cardoso, Victor Gabriel.** *Sistema para la gestión de los procesos de la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas.* La Habana, Cuba : s.n., 2014.

4. —. *"Documento visión del Proyecto MAYA".* Universidad de las Ciencias Informaticas : s.n., 2016.

5. **Eguiluz, Javier.** Symfony. [En línea] 2011. [Citado el: enero 22, 2016.] www.symfony.es/documentación/actualización.

6. **Basso Mesa, Anileidy y Gómez Almaguer, Dayaima.** *Sistema automatizado para el proceso de caracterización de los estudiantes.* Ciudad de la Habana, Cuba : s.n., 2009.

7. **Colectivo de Autores.** *ABC de la FEU, reglamentos y estatutos.* La Habana : Federico Engels, 2013.

8. grupodoccf. DocCF, Software de Gestión Escolar. *Módulo Básico. Gestión de Docentes y Alumnos.* [En línea] [Citado el: febrero 17, 2016.] www.grupocfdeveloper.com/modulo_basico_gestion_de_docentes_y_alumnos.htm.

9. **González Fernández, Yisel y Benítez Campo, Yanitza.** *Sistema de apoyo al ingreso, seguimiento y control del movimiento de alumnos ayudantes en la facultad 1.* La Habana, Cuba : s.n., 2010.

10. **Varela, Rafael.** SAS Académico. *SAS Académico.* [En línea] 2015. [Citado el: febrero 17, 2016.] http://www.rafaelvarela.com/software-academico-sas-notas-boletines.html.

11. **Reyes Redondo, Yoe.** *istema para la gestión de la información asociada a la residencia estudiantil. 11na Peña Tencnológica, Universidad de las Ciencias Informáticas. Memoria de eventos.* La Habana, Cuba : s.n., 2016.

12. **Medina Pichs, Alejandro, Sardañas Rodríguez, María Teresa.** *Sistema para la gestión de los eventos estudiantiles. 11na Peña Tencnológica, Universidad de las Ciencias Informáticas. Memoria de eventos.* La Habana. Cuba : s.n., 2016.

13. **Isasi, Raydel Cabrera.** *EDUCA: Sistema para la gestion de la información de las Estrategias Educativas.* Universidad de las Ciencias Informáticas : s.n., 2016.

14. **Fundation, Mozilla.** Firefoxmanía. *Firefoxmanía.* [En línea] [Citado el: febrero 02, 2016.] http://firefoxmania.uci.cu/download/?p=faq&a=fx..

15. **Hewitt, Joe.** DrDobb's. *DrDobb's.* [En línea] enero 10, 2007. [Citado el: febrero 21, 2016.] http://www.drdobbs.com/tools/ajax-debugging-with-firebug/196802787?pgno=1.

16. **Pressman, Roger S.** Ingeniería de Software, un enfoque práctico. Quinta edición. 2002.

17. **Andrearrs.** Hipertextual. *Hipertextual.* [En línea] mayo 12, 2014. [Citado el: enero 24, 2016.] http://hipertextual.com/archivo/2014/05/git-sistema-control-versiones/.

18. **PgAdmin.** *PgAdmin.* [En línea] 2014. [Citado el: diciembre 14, 2015.] http://www.pgadmin.org.

19. NetBeans. NetBeans. *NetBeans.* [En línea] 2014. [Citado el: febrero 21, 2016.] https://netbeans.org/index_es.html.

20. PostgreSQL. Postgre. *Postgre.* [En línea] 2014. http://www.postgresql.org.es/sobre_postgresql.

21. Symfony2. Symfony. *Symfony.* [En línea] 2014. http://symfony.es/que-es-symfony.

22. Team, Doctrine Project. La web del programador. *La web del programador.* [En línea] noviembre 03, 2011. http://www.lawebdelprogramador.com/cursos/PHP/7033-Doctrine_2_ORM_Documentation.html.

23. Symfony. Libros web. *Libros web.* [En línea] 2010. http://librosweb.es/symfony/capitulo_8/por_que_utilizar_un_orm_y_una_capa_de_abstraccion.html.

24. UBUNTU. Ubuntu. *Ubuntu.* [En línea] [Citado el: febrero 16, 2016.] https://help.ubuntu.com/6.10<apache/HTTPD_-_Servidor_web_Apache2/index.html.

25. Kabir, Mohammed J. *La Biblia del Servidor Apache 2.* Madrid : Anaya Multimedia, 2003. pág. 845. ISBN: 8441514682..

26. Bootstrap. Página Oficial de Boostrap. *Bootstrap.* [En línea] [Citado el: febrero 17, 2016.] http://getbootstrap.com.

27. Nacional, RENA. Red Escolar. RENA. *RENA.* [En línea] Ministerio del Poder Popular para Educacion Universitaria,Ciencia y Tecnologia de Venezuela, 2008. [Citado el: octubre 26, 2016.] www.rena.edu.ve/cuartaEtapa/Informatica/Tema13.html.

28. Network, Mozilla Depeloper. MDN Mozilla Developer Network. *MDN Mozilla Developer Network.* [En línea] [Citado el: febrero 2014, 16.] https://developer.mozilla.org/es/docs/User:Marti1125.

29. Torralba, Miguel A. López. Mialto Tutoriales de desarrollo web para programar desde cero. *Mialto Tutoriales de desarrollo web para programar desde cero.* [En línea] enero 20, 2015. [Citado el: mayo 18, 2016.] http://mialtoweb.es/tecnologias-usadas-en-entorno-cliente-y-servidor/.

30. Fundation, Mozilla Developer. Mozilla. *Mozilla.* [En línea] ttps://developer.mozilla.org/es/docs/Web/HTML.

31. Alexis Goldstein, Louis Lazaris, Estelle Weyl. *Introducción a CSS.* 2011.

32. Rumbaugh, James, Jacobson, Ivar y Booch, Grady. *El Lenguaje Unificado de Modelado.* 2000.

33. *Programación Extrema: "Metodología para desarrollo ágil de aplicaciones".* Universitaria, Dirección de Comunicacion. 486, Xalapa, Veracruz, México : Universidad Veracruzana, 2012.

34. Fowler, Martin. martinfowler.com. *martinfowler.com.* [En línea] mayo 2004. [Citado el: enero 26, 2016.] http://www.martinfowler.com/articles/designDead.html.

35. Ramírez, Ing. Danay Pérez. *Metodologías ágiles. ¿Cómo desarrollo utilizando XP?* s.l. : cujae, 2008.

36. PATÓN, D. Metodologías de Desarrollo de Software. [En línea] [Citado el: marzo 3, 2012.] http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema04.pdf.

37. Española, Real Academia. Diccionario de la lengua española. *Diccionario de la lengua española.* [En línea] abril 13, 2016. [Citado el: mayo 17, 2016.] http://dle.rae.es.

38. LARMAN, Craig. *UML y Patrones.* *Introducción al análisis y diseño orientado a objetos.* 1999. 970-1 7-0261-1.

39. IEEE. Institute of Electrical and Electronics Engineers. *Institute of Electrical and Electronics Engineers.* [En línea] Introducción a la Ingeniería de Requisitos. Ingeniería de software., 2010. www.ieee.org.

40. Buenas Tareas. [En línea] abril 20, 2014. www.buenastareas.com/materias/herramientas-para-pulir-un-manuscrito.

41. Almirón, Cristóbal González. Adictos al Trabajo. [En línea] abril 12, 2014. www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=RM#_Toc260742341.

42. Sommerville, Ian. ***Ingeniería del Software.*** Madrid (España) : Pearson Educación, 2005. ISSN 84-7829-074-5.

43. C., Sergio Alberto. [En línea] Instituto Tecnológico de Sondra, Septiembre 14, 2015. [Citado el: marzo 2, 2016.] www.iswugaps2extremeprogramming.wordpress.com/2015/09/14/ventajas y desventajas/.

44. Guglielmetti, Marcos Gérman. Master Magazine. *Definición de Arquitectura de Software.* [En línea] [Citado el: diciembre 27, 2015.] www.mastermagazine.info/termino/3916.php.

45. Bascón Pantoja, Ernesto. *El patrón de diseño modelo -vista - controlador (MVC) y su implementación el Java Swing. .* 2004. 4 Vol. II..

46. Eguiluz, Javier. *Desarrollo web ágil con Symfony2.* 2012.

47. Labs, Sensio. *Official Symfony Best Practices.* 2014.

48. Hernán Visconti, Marcello y Astudillo. Patrones de diseño. . *Fundamentos de Ingenieria de Software.* 2004.

49. Teleformación, Dirección de. Entorno Virtual de Aprendizaje. Modelo entidad relación (MER).Extensiones del MER. [En línea] marzo 24, 2014. http://eva.uci.cu/file.php/180/2._Clases/Tema_1/Materiales_basicos/2.Modelo_Entidad_Relacion_y_extensiones.pdf..

50. VENEmedia. Conceptos y Definiciones. *Conceptos y Definiciones.* [En línea] VENEmedia, abril 18, 2014. [Citado el: mayo 17, 2016.] http://conceptodefinicion.de/http/.

51. MASSÉ, M. *REST API Design Rulebook.* s.l. : O'Reilly Media, Inc., 2012. 978-1 -449-31050-9.

52. Lazo, M. J. R. y Reyes, M. S. *Modulo Control de Acceso del Proyecto Intranet del Centro Rector de Universidad para Todos.* Universidad de las Ciencias Informaticas : s.n., 2007.

53. Sánchez, Javier Zapata. [En línea] enero 21, 2013. [Citado el: mayo 18, 2016.] https://pruebasdelsoftware.wordpress.com/.

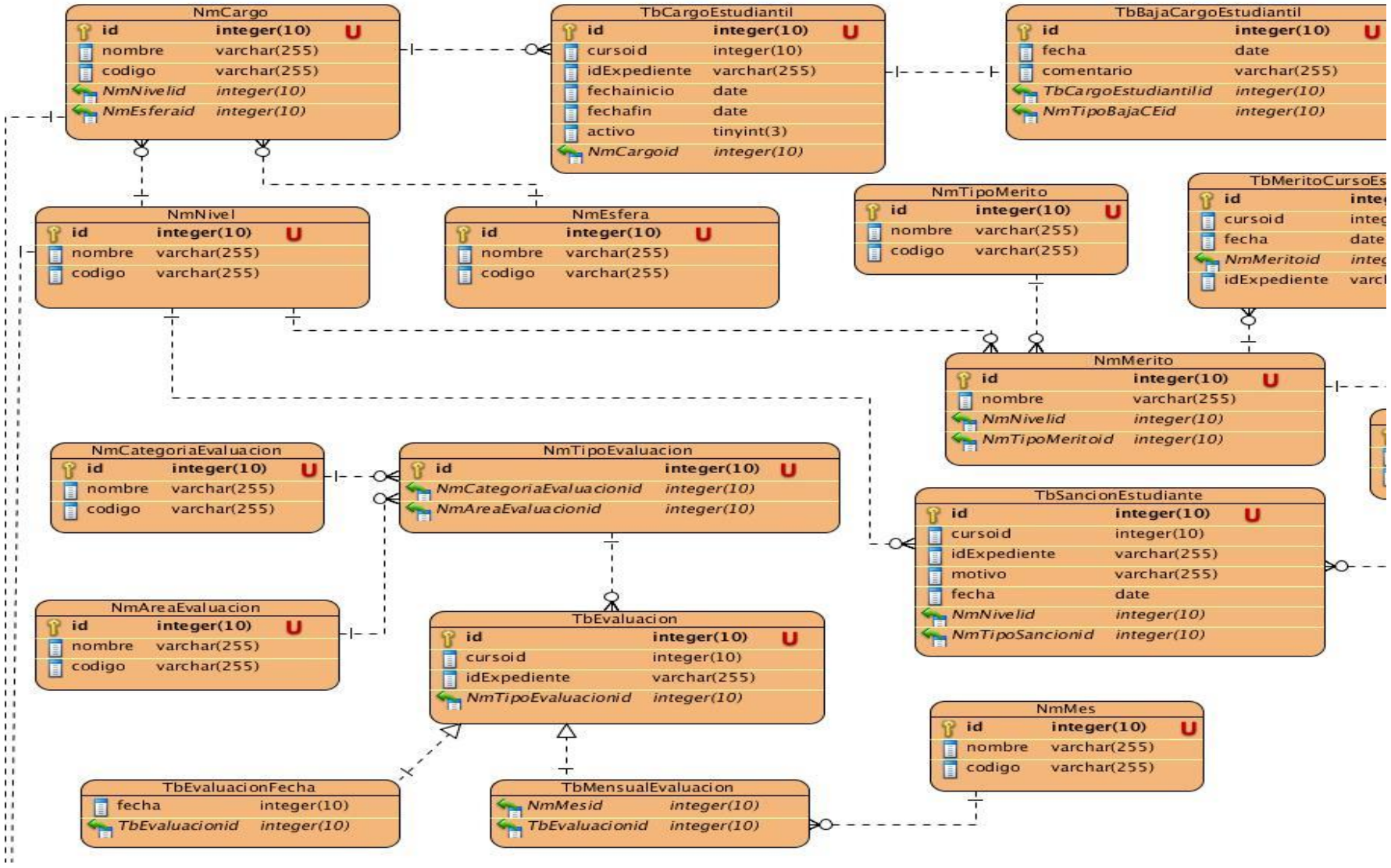
54. Gutiérrez, Ángel Mendoza. Departamento de Informática, Universidad de Valladolid, Campus de Segovia. [En línea] octubre 2015. [Citado el: mayo 20, 2016.] http://docplayer.es/6401012-Departamento-de-informatica-universidad-de-valladolid-campus-de-segovia-tema-7-validacion.html.

55. García Rodríguez, Yanet y González González, Yoandrys. *Modelado de una aplicación web para la gestión de las tareas de funcionamiento de la FEU en la UCI.* La Habana, Cuba : s.n., 2007.

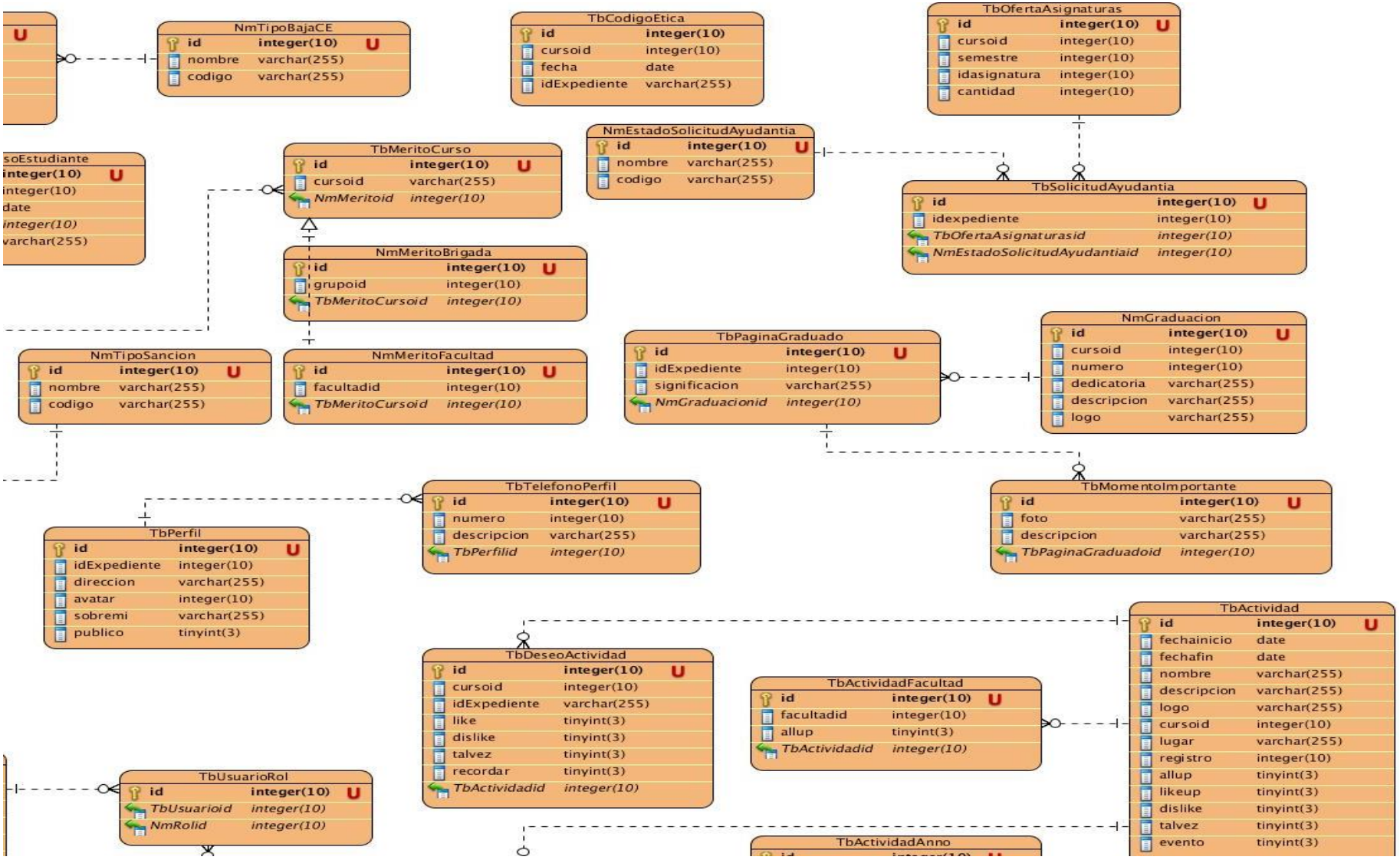
ANEXOS

ANEXO 1: Modelo de base de datos

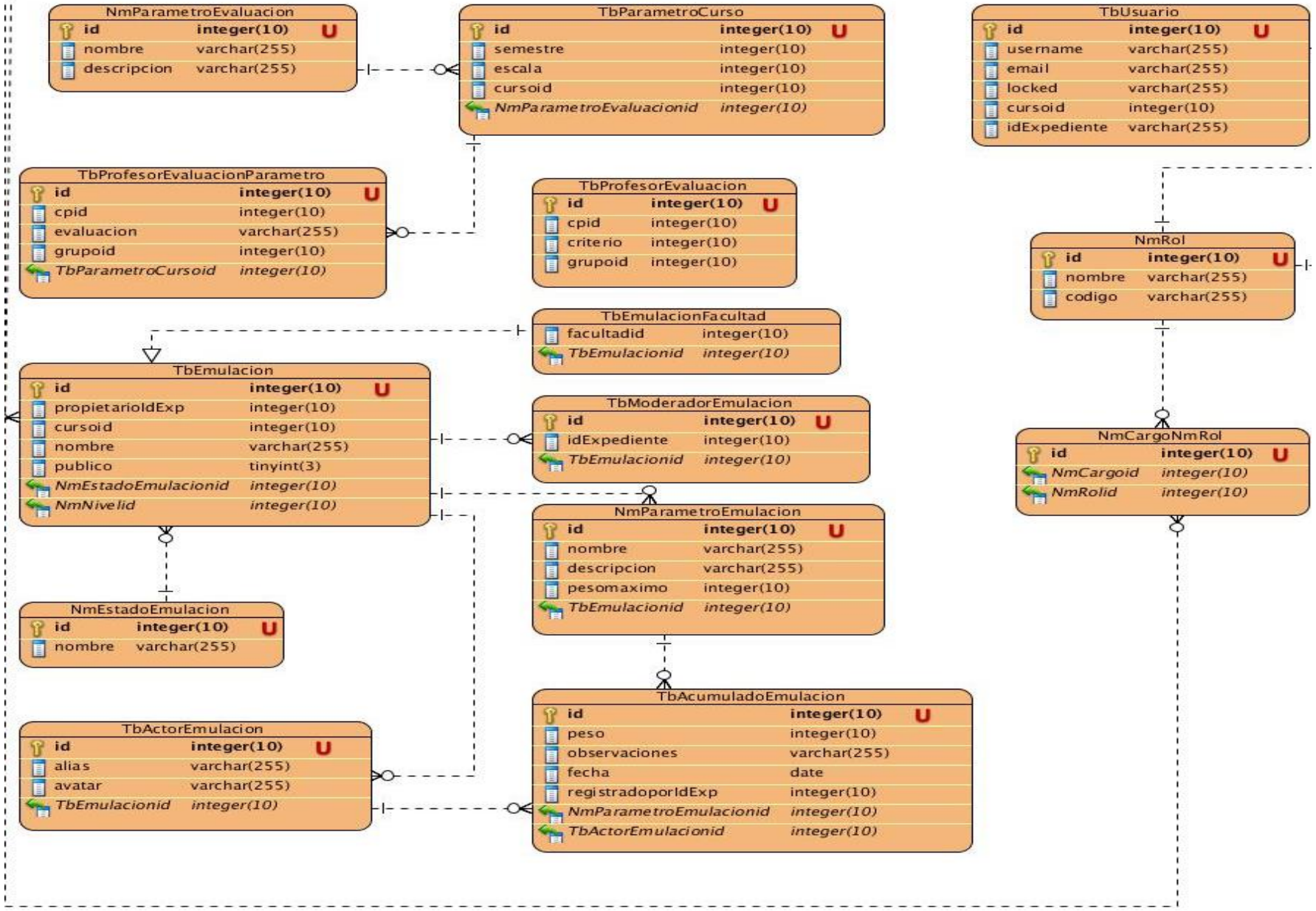
Vista superior izquierda (1/4):



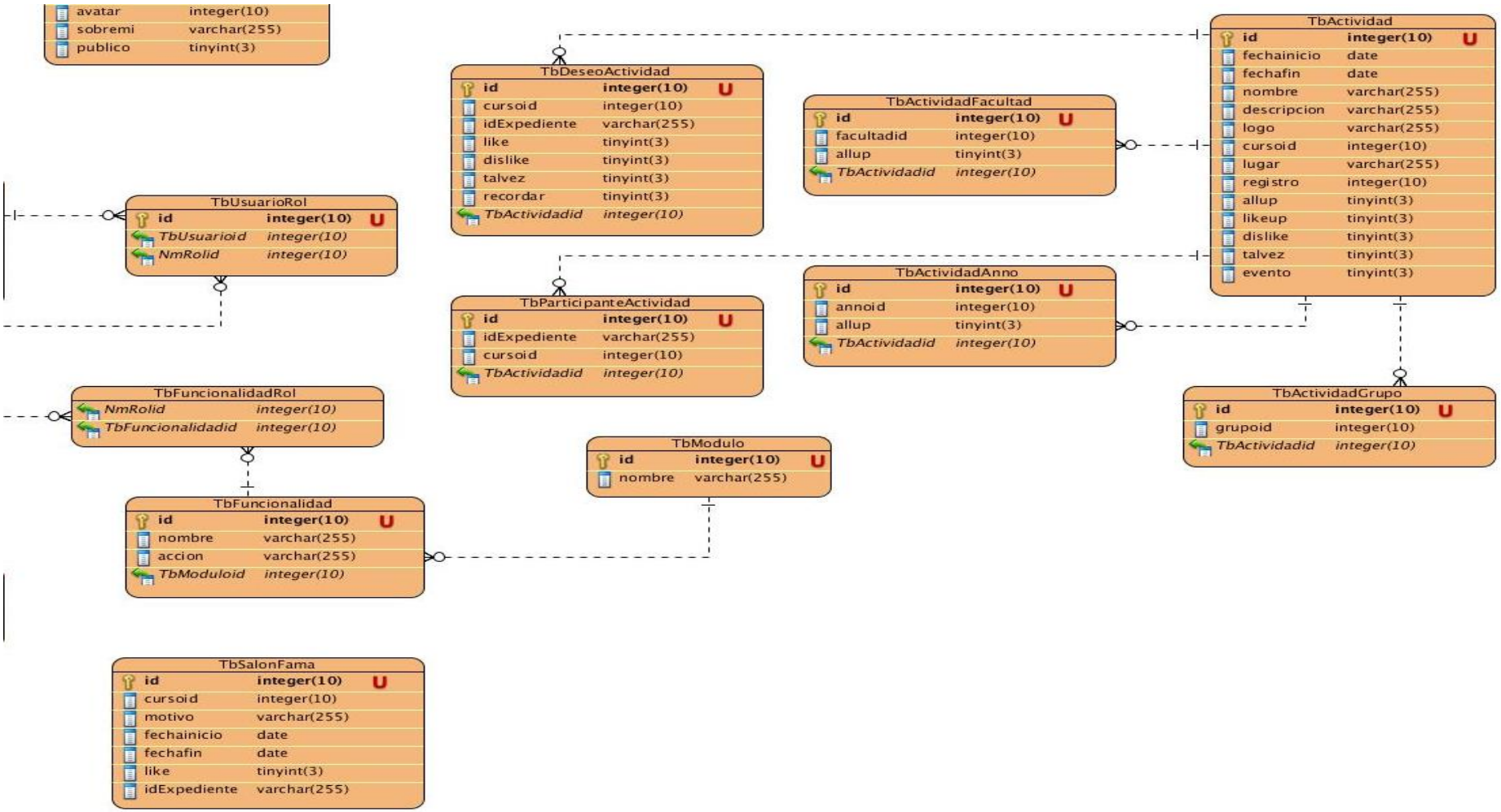
Vista superior derecha (2/4):



Vista inferior izquierda (3/4):



Vista inferior derecha (4/4):



ANEXO 2: Criterios de complejidad

La complejidad de los requisitos se utiliza para estimar el esfuerzo de implementación de este y planificar en qué iteración se implementará. Un requisito puede tener complejidad alta, media o baja, determinándose a partir de los criterios que se describen a continuación:

- Complejidad por número de transacciones: una transacción es un intercambio de datos a nivel de procesos elementales entre las fronteras del sistema, que fuerzan a la lógica de la aplicación a realizar un procesamiento para entrar y/o mostrar información. La complejidad viene representada en el esfuerzo de implementación de un número elevado de procedimientos reflejados en una transacción.

  Alta: El requisito contiene más de 8 transacciones.

  Media: El requisito contiene de 5 a 8 transacciones.

  Baja: El requisito contiene de 1 a 4 transacciones.

- Complejidad por entidades candidatas: las entidades candidatas son las futuras clases de la aplicación que modelarán los elementos persistentes de la base de datos. Una clase envuelve atributos, procedimientos y está relacionada a clases de control, por lo que un monto elevado de entidades o de atributos de la entidad refleja una complejidad en la implementación de las operaciones de un Requisito que las contiene.

  Alta: más de 5 entidades.

  Media: de 3 a 5 entidades.

  Baja: de 1 a 2 entidades.

- Complejidad por interfaces de comunicación con actores: las interfaces de comunicación muestran a un actor del sistema la forma de intercambiar información ya sea para la entrada, como para la salida. Un actor puede estar instanciado en una persona o un sistema externo que interactúe con el sistema. La complejidad viene reflejada por el esfuerzo de la implementación de dichas interfaces unida a las restricciones y funcionalidades de su uso.

  Alta:

  Una persona que interactúa con el sistema mediante una interfaz gráfica donde se evidencie una de las situaciones siguientes:
  La interfaz posee elementos de animación en tercera dimensión 3D y se vale de ellos para la gestión de la misma.

La interfaz utiliza gráficos en segunda dimensión 2D para la muestra de información y se vale de ellos para la gestión de la misma.
La interfaz está formada por más de 15 componentes tradicionales de captura y muestra de información.

Media:

La interfaz contiene desde 8 hasta 15 componentes tradicionales de captura y muestra de información.

Baja:

La interfaz contiene desde 1 hasta 7 componentes tradicionales de captura y muestra de información.
Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API, DLL)
Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto (Servicios Web, XML, Correo electrónico).

- Complejidad por número de requisitos no funcionales: los requisitos no funcionales son características que imponen restricciones al sistema, por lo que condicionan la implementación de los requisitos al agregarle especificidades a las funcionalidades del mismo.

  Alta: más de 6 requisitos no funcionales asociados.

  Media: entre 4 y 6 requisitos no funcionales asociados.

  Baja: menos de 4 requisitos no funcionales asociados.

- Complejidad por tipo de tecnología: la utilización de los avances tecnológicos de software o hardware que necesiten de una asimilación de su funcionamiento o comunicación condicionan el esfuerzo de la implementación de Requisitos que los empleen.

  Alta: la realización del requisito utiliza elementos, plataformas, componentes tecnológicos avanzados de conocimiento externo al equipo de programación que necesiten de capacitación.

  Media: la realización del requisito utiliza elementos tecnológicos avanzados donde el conocimiento reside en el equipo de implementación, producto de la selección y preparación inicial en los acuerdos tecnológicos del contrato.

  Baja: no se emplean nuevas técnicas, plataformas, componentes ni conceptos de programación.

- Complejidad por reutilización: el uso de patrones, componentes predefinidos del lenguaje o plataformas que generen partes de las funcionalidades del sistema aminoran el esfuerzo en la construcción del software.

  Alta: no se reutilizan elementos, el requisito se codifica por completo por un programador.

  Media: se reutilizan elementos donde es necesario ajustar la codificación antes de la generación de la implementación del caso de uso.

  Baja: se utilizan elementos que implementan el requisito y sólo es necesario realizar pequeños ajustes.

ANEXO 3: Método Asignar participación a una actividad.

```
public function createParticipacionActividadesAction(Request $request, $idActividad)
{
    $em = $this->getDoctrine()->getManager();
   $entity = $em->getRepository('AppBundle:TbActividad')->find($idActividad);
   $idExpediente = $request->request->get('idExpediente');

    $form = $request->request->get('form');
    $participacion = $form['tipo'];
    $resultado = $form['resultado'];

    $colateral = false;
    $organizador = false;
    $participante = false;

    foreach ($participacion as $tipo){
        if ($tipo == 'col')
            $colateral = true;
        elseif ($tipo == 'org')

            $organizador = true;
        elseif ($tipo == 'part')
            $participante = true;
    }
    $resultado = $em->getRepository('AppBundle:NmResultado')->find($resultado);
    $persona = $this->get('session')->get('credencialSeguridad');

    $obj = new TbParticipanteActividad();
    $obj->setIdExpediente($idExpediente);
    $obj->setActividad($entity);
    $obj->setResultado($resultado);
    $obj->setColateral($colateral);
    $obj->setOrganizador($organizador);
    $obj->setParticipante($participante);
    $obj->setIdFacultad($persona['idFacultad']);
    $obj->setIdGrupo($persona['idArea']);


    $em->persist($obj);

    $em->flush();

    $this->get('session')->getFlashBag()->add(
        'success',
        'Ha insertado una nueva actividad al estudiante seleccionado satisfactoriamente.'
    );

    return $this->redirectToRoute('participacion_actividades_new', array('id' => $idActividad));
}
```