# Universidad de las Ciencias Informáticas

## Facultad de Ciencias y Tecnologías Computacionales


# Sistema para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos

**Trabajo de diploma para optar por el título de**
**Ingeniero en Ciencias Informáticas**


**Autor:**

Marlen González Cruz

**Tutores:**

MSc. Yordanys Piñeiro Gómez

Ing. Eric Rodríguez Ochoa



**La Habana, junio de 2017**

**"Año 59 de la Revolución"**

# Declaración de autoría

Por este medio declaro que yo, Marlen González Cruz, con carné de identidad 90121140274, soy la autora principal del trabajo titulado “Sistema para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos”, y autorizo a la Universidad de las Ciencias Informáticas (UCI) y al Centro de Innovación y Calidad de la Educación (CICE), a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmo la presente declaración jurada de autoría en La Habana a los ___ días del mes de _____ del año 2017.

_____

Marlen González Cruz

Firma del Autor

_____

MSc. Yordanys Piñeiro Gómez

Firma del Tutor

_____

Ing. Eric Rodríguez Ochoa

Firma del Tutor

# Datos de contacto

**Tutora: MSc. Yordanys Piñeiro Gómez**

Graduada de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en julio de 2007.

Categoría docente: Asistente

Profesor de las asignaturas Ingeniería de Software I y II, Gestión de Software e Introducción a las Ciencias Informáticas.

Centro de Trabajo: Universidad de las Ciencias Informáticas (UCI). Centro de Innovación y Calidad de la Educación (CICE).

Cargo: Profesora. Vicerrectoría de Formación.

Dirección: Carretera a San Antonio, Km 2 ½, Reparto Torrens, La Lisa, La Habana. Universidad de las Ciencias Informáticas (UCI), Edificio: 119, Apto: 107.

Teléfono Oficina: +53 – 7 – 837 – 2522.

Correo electrónico: ypineirog@uci.cu

**Tutor: Ing. Eric Rodríguez Ochoa**

Graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en julio de 2014.

Profesor de la asignatura Introducción a la Programación.

Centro de Trabajo: Universidad de las Ciencias Informáticas (UCI). Centro de Innovación y Calidad de la Educación (CICE).

Cargo: Especialista A. Vicerrectoría de Formación.

Dirección: Carretera a San Antonio, Km 2 ½, Reparto Torrens, La Lisa, La Habana. Universidad de las Ciencias Informáticas (UCI), Edificio: 70, Apto: 102.

Teléfono Oficina: +53 – 7 – 837 – 8126.

Correo electrónico: erochoa@uci.cu

**Resumen**

La evaluación y acreditación en instituciones de Educación Superior constituye un proceso de aprendizaje continuo y una mejora constante de la calidad de los sistemas educativos universitarios. Particularmente, la autoevaluación permite identificar las fortalezas y debilidades que posee la institución para consolidar su trabajo y de esta forma elevar la calidad de sus procesos y las distintas mejoras que pueden aplicarse en la institución. Actualmente en la Universidad de las Ciencias Informáticas, los procesos que se llevan a cabo para la gestión de información, específicamente de la variable infraestructura y gestión de los recursos, se tornan lentos y engorrosos, debido a que los mecanismos que se encuentran establecidos no posibilitan que se desarrolle de otra manera. No existe una constante actualización de la información, ya que esta se encuentra dispersa por todo el campus universitario. Además, dada la cantidad de datos que se almacenan no se pueden generar de forma rápida reportes que se necesiten en un momento determinado. Por tal razón, la presente investigación tiene como objetivo desarrollar un sistema de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos que favorezca los grados de centralidad, disponibilidad e integridad de la información asociada a esta variable en la Universidad de las Ciencias Informáticas. Para desarrollar este sistema se eligió el Proceso Unificado Abierto como metodología de desarrollo de software, Symfony2 como *framework* de desarrollo y PostgreSQL como gestor de base de datos.

**Palabras clave:** evaluación institucional, gestión de información, gestión de los recursos, infraestructura.

# Abstract

***Abstract***

*The evaluation and accreditation in institutions of Higher Education constitutes a process of continuous learning and a constant improvement of the quality of university education systems. In particular, the self-assessment allows identifying the strengths and weaknesses that the institution has to consolidate its work and in this way raise the quality of its processes and the different improvements that can be applied in the institution. Currently in the University of Information Sciences, the processes that are carried out for information management, specifically the variable infrastructure and management of resources, become slow and cumbersome, because the mechanisms that are established do not allow that is developed in another way. There is no constant updating of the information, since it is dispersed throughout the university campus. In addition, given the amount of data that is stored can not quickly generate reports that are needed at any given time. For this reason, the present research aims to develop an information management system for the evaluation and institutional accreditation of the variable infrastructure and management of resources that favors the degrees of centrality, availability and integrity of the information associated with this variable in The University of Computer Science. To develop this system, the Open Unified Process was chosen as a methodology for software development, Symfony2 as a development framework and PostgreSQL as a database manager.*

***Keywords****: institutional evaluation, information management, resource management, infrastructure*

# Índice

# Índice

# Índice de tablas y figuras

**Índice de Tablas**



**Índice de Figuras**

## Introducción

Uno de los grandes desafíos de la educación, según (Giorgetti, y otros, 2014), es sin dudas el mejoramiento de la calidad de los sistemas educativos. La evaluación de la calidad de la educación universitaria se ha presentado como uno de los principales temas en la agenda de las reformas educativas a nivel mundial. La evaluación y especialmente la acreditación aparecieron como herramientas adecuadas para regular el sistema de educación universitaria, desde la perspectiva de la calidad de los servicios educativos que se ofrecen.

Autores como (Escribano Hervis, y otros, 2013) reconocen la amplia experiencia que a nivel internacional se manifiesta en la implementación de proyectos de evaluación institucional, sobre todo en países de Europa y América Latina, con metodologías y criterios muy diversos, pero con un objetivo en común: la mejora continua de la calidad. Esta tendencia consiste en desarrollar una conciencia para mejorar la calidad de la gestión universitaria y ofrecer la posibilidad de impulsar la movilidad de estudiantes, profesores, investigadores y currículos de país a país, en un ámbito más extenso.

En Cuba, el Ministerio de Educación Superior (MES) desde su creación en 1976, ha prestado especial atención al control del trabajo que han desempeñado las instituciones universitarias, como vía fundamental para lograr el mejoramiento continuo de la calidad. Un gran impulsor de esta mejora continua fue nuestro Comandante en Jefe, el cual expresó*: "Yo no diría que otros países de América Latina o del Tercer Mundo estén por delante de nosotros en la calidad de la educación. Creo, sinceramente, que estamos muy por delante de los demás, incluso en eso que estamos discutiendo y con lo cual no estamos satisfechos, que es la calidad de la educación. No podríamos estar satisfechos en eso porque, realmente, podemos hacer mucho más, y, en realidad, necesitamos mucho más: lo necesita el país, lo necesita la Revolución, lo necesita el socialismo, lo necesitan nuestras aspiraciones de alcanzar un día una sociedad superior, incluso, una sociedad comunista. Y esto se ha convertido en el centro de nuestras discusiones."* (Castro Ruz, 1987)

De esta manera el MES establece a partir del año 2002 el Sistema Universitario de Programas de Acreditación (SUPRA). El SUPRA está dirigido a promover, estimular y certificar la calidad de instituciones y programas. Para ello aplica desde el año 2003 la evaluación institucional como forma de control para determinar la calidad del trabajo en los centros y la gestión en todos sus procesos, en correspondencia con la misión y función social encargada por el estado y el gobierno. Para (Rubio Oca, 2007) la acreditación se

identifica cada vez más por las instituciones, como uno de los medios más adecuados para lograr reconocimiento social y prestigio.

Este Ministerio crea la Junta de Acreditación Nacional (JAN), en la actualidad con más de 15 años de creada, la cual es la encargada del proceso de evaluación externa para reconocer la calidad de los diferentes programas académicos que se desarrollan en las universidades. Para la evaluación institucional el Sistema de Evaluación y Acreditación de Instituciones de Educación Superior (SEA-IES) define seis variables: Contexto institucional, Gestión de los recursos humanos, Formación del profesional de pregrado, Interacción social, Infraestructura y gestión de los recursos e Impacto.

Según (Silva Correa, 2017), hasta abril del presente año se habían acreditado en el país 3 instituciones de nivel superior con la categoría de excelencia: la Universidad de La Habana (UH), la Universidad Central Marta Abreu de Las Villas (UCLV) y más recientemente la Universidad Tecnológica José Antonio Echeverría (CUJAE). La Universidad de las Ciencias Informáticas (UCI), a pesar de ser una de las universidades más jóvenes del país, con apenas 13 años de creada, cuenta ya, con la categoría superior de carrera certificada, proceso desarrollado durante el año 2015 en la Facultad 6, hoy Facultad de Ciencias y Tecnologías Computacionales. Cuenta además con la maestría en Gestión de Proyectos Informáticos evaluada de excelencia, se trabaja con empeño en la acreditación de la maestría en Calidad de Software y el doctorado en Ciencias Informáticas, así como en el mejoramiento de la calidad de todos los procesos que se desarrollan en la institución.

A pesar de estos importantes logros, la UCI tiene como premisa, en el presente año, realizar junto a la JAN un proceso de evaluación y acreditación institucional. Hoy, se lleva a cabo un proceso de autoevaluación, en el cual se organizan varios equipos de trabajo que responden directamente a la recopilación y gestión de información asociada a cada una de las variables mencionadas con anterioridad. Es interés de la presente investigación, la variable infraestructura y gestión de los recursos, para la cual se hace necesario registrar y compilar una gran cantidad de información que está actualmente muy dispersa en toda la universidad.

Específicamente para la variable en cuestión, se designa un equipo de trabajo, que registra la información necesaria, gran parte de ella se compila a través de un fichero Excel, proporcionado con anterioridad por la JAN, y otra se evalúa de manera personal en cada una de las áreas necesarias. Luego se elaboran dos documentos Word, donde se recogen: las principales fortalezas y debilidades existentes, así como el plan

de mejoras asociado a estas debilidades. Este fichero Excel, se envía a través del correo electrónico o se entrega de manera personal a los implicados en cada una las áreas de la universidad. Se evidencia de esta manera la gran complejidad, dispersión y diversidad de la información que se manipula.

Como consecuencia, todos los procesos que se deben seguir para la gestión de esta información se tornan lentos y engorrosos, debido a que los mecanismos que se encuentran establecidos no posibilitan que se desarrolle de otra manera. No existe una constante actualización de la información por la dispersión de misma. Dada la cantidad de datos que se almacena, no se pueden generar de forma rápida reportes que se necesiten en un momento determinado. Además de que ocurre una mayor probabilidad de errores en los informes finales, porque depende de la habilidad que tenga una persona para realizar el procesamiento que se necesita para obtener los resultados finales.

Actualmente, el proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos, presenta las siguientes deficiencias:

- La integridad de la información está comprometida por posibles errores humanos en su almacenamiento y transmisión; así como en el procesamiento manual de esta.
- La disponibilidad de la información necesaria, está afectada debido a que los procesos de búsqueda son manuales y utilizan datos en diferentes formatos.
- La información que existe se encuentra dispersa y en diferentes formatos. No se dispone de un medio capaz de almacenar toda la información que se genera en el proceso.
- No es posible generar informes de manera dinámica y con fácil actualización.

A partir de la situación problemática anteriormente expuesta, se plantea como **problema de la investigación**:

¿Cómo favorecer los grados de centralidad, disponibilidad e integridad de la información en el proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos?

En correspondencia con el problema planteado, el **objeto de estudio** lo constituye el proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos.

# Introducción

Para la realización de la investigación, se delimita como **campo de acción** los sistemas informáticos de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos.

Para dar solución al problema planteado se define como **objetivo general** desarrollar un sistema de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos que favorezca los grados de centralidad, disponibilidad e integridad de la información asociada a esta variable en la Universidad de las Ciencias Informáticas.

Con el propósito de dar cumplimiento al problema planteado se formulan las siguientes **preguntas de la investigación:**

1. ¿Qué fundamentos teóricos sustentan el proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos?
2. ¿Qué características tiene el proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos en la Universidad de las Ciencias Informáticas?
3. ¿Cómo organizar el proceso de desarrollo del sistema de gestión para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos?
4. ¿El sistema de gestión desarrollado garantiza la centralidad, disponibilidad e integridad de la información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos en la Universidad de las Ciencias Informáticas?

Para darle solución al objetivo general se definen las siguientes **tareas de investigación:**

1. Determinación de los fundamentos teóricos del proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos.
2. Caracterización del estado actual del proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos.
3. Implementación del sistema de gestión para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos.
4. Validación del sistema de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos a través de las pruebas que se diseñen.

Con el objetivo de facilitar el desarrollo de la investigación, se utilizaron los siguientes **métodos de investigación.** Como parte de los **métodos teóricos** utilizados se encuentran:

- **Método histórico – lógico:** Se utilizó para determinar antecedentes, tendencias y regularidades del objeto de estudio y el campo de acción. Específicamente para estudiar la forma en que han evolucionado las tecnologías para el desarrollo de sistemas informáticos que promueven los procesos de evaluación y acreditación de la Educación Superior.
- **Método análisis – síntesis:** Se empleó para determinar las generalidades y especificidades en el objeto de estudio y el campo de acción. Específicamente para la determinación de los fundamentos que sustentan el proceso de gestión de información para la evaluación y acreditación de la variable infraestructura y gestión de los recursos, así como en la elaboración de la propuesta de solución.
- **Modelación:** Se utilizó con el objetivo de obtener los artefactos necesarios para estudiar y transformar el proceso de análisis de datos de la concepción del sistema para la evaluación y acreditación de la variable infraestructura y gestión de los recursos.

Se utilizaron como **métodos empíricos:**

- **Análisis documental:** Se utilizó con el objetivo de obtener información mediante la recolección y selección de documentos relacionados con el tema que se estudia.
- **Entrevista:** Se aplicó para obtener información sobre los procesos de evaluación y acreditación de la variable infraestructura y gestión de los recursos, así como para la determinación de las posibles funcionalidades y otras características que debía tener el sistema a construir. Específicamente se utiliza la entrevista no estructurada.

**Posibles resultados:**

1. Documentación ingenieril asociada a los flujos de trabajo: Gestión del proyecto, Requisitos, Arquitectura, Implementación y Pruebas a partir de la metodología seleccionada.
2. Sistema de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos (SEAIVI) en la Universidad de las Ciencias Informáticas.

La investigación se encuentra estructurada de la siguiente forma:

**Introducción:** Se realiza la presentación de la investigación, a través de la inclusión de los antecedentes y situación problemática, actualidad, importancia del problema planteado. Por último, se presenta el diseño teórico-metodológico de la investigación.

En el **capítulo 1** se establecen los referentes teórico-metodológicos concernientes al proceso de gestión de información para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos. De igual forma se analizan los diferentes sistemas informáticos asociados al objeto que se investiga para determinar la necesidad de una nueva solución. Por último, se fundamenta la selección de la metodología de desarrollo que guía la investigación, así como de las herramientas y tecnologías que se utilizan para el desarrollo de la aplicación.

En el **capítulo 2** se detallan las características del sistema a partir de la especificación de los requisitos funcionales y no funcionales. Se muestra la estructura general del sistema a partir de diversos modelos y descripciones asociadas a estos. Además se describe la solución a partir de la definición del patrón arquitectónico a utilizar. Se evidencia la conformación del modelo físico de la Base de Datos y el modelo de despliegue.

En el **capítulo 3** se presentan los principales artefactos que se generan durante la etapa de implementación. También se definen los estándares de codificación a utilizar, así como el tratamiento de errores. Por último se especifican las pruebas que se realizan al sistema para la comprobación del correcto funcionamiento del mismo.

Por último, se presentan las **conclusiones**, las **recomendaciones**, la **bibliografía** y el conjunto de **anexos** para una mejor comprensión de lo expuesto en la investigación.

# Capítulo 1. Proceso de evaluación y acreditación de instituciones de Educación Superior

En este capítulo se describen los fundamentos teóricos relacionados con el proceso de evaluación y acreditación de instituciones de Educación Superior y se enmarca específicamente en la variable infraestructura y gestión de los recursos. El mismo consta de cuatro epígrafes. En el primero se caracteriza de manera sintetizada el proceso de evaluación y acreditación de instituciones de Educación Superior, así como el estado actual del proceso de gestión de la variable infraestructura y gestión de los recursos en la Universidad de las Ciencias Informáticas. En el segundo se formaliza un análisis del proceso de gestión de información, haciendo referencia a la integridad, centralidad y disponibilidad de la información. En el tercer capítulo se hace un estudio de las soluciones existentes que contribuyen en la investigación. Se culmina el capítulo con la caracterización de la metodología de desarrollo de software escogida y la especificación de las herramientas y tecnologías para el desarrollo la aplicación.

## 1.1 Caracterización del proceso de evaluación y acreditación de instituciones de Educación Superior

La transición entre el siglo XX y el actual ha venido acompañada de profundos cambios en los sistemas de Educación Superior a nivel global. Cada vez se hace más patente la necesidad de disponer de herramientas que permitan conocer la calidad de las instituciones de Educación Superior mediante metodologías basadas en la evaluación.

Según (JAN, 2014), para realizar este importante proceso, el país cuenta con el Sistema de Evaluación y Acreditación de Instituciones de Educación Superior. Sistema aprobado por el MES, que cuenta con dos modalidades: Evaluación Interna y Evaluación Externa. Para la primera modalidad la propia institución es la encargada de realizar su autoevaluación; mientras que para la segunda, se designa a la JAN como una entidad especializada e independiente, brindándole así al proceso un carácter externo, sistémico e integrado. El proceso de acreditación institucional tiene dos fases fundamentales: la evaluación y la acreditación.

Para (Pérez Pino, y otros, 2015) la evaluación interna o autoevaluación, como también se le conoce, hay que prestarle especial atención por la importancia que tiene durante el proceso. Con esta se persigue informar, rendir cuentas, comprobar y mejorar la eficiencia, la eficacia y la calidad del objeto o proceso que

se evalúa. Se caracteriza por ser participativa, transparente, reflexiva y ética. Esta autoevaluación da lugar a un informe sobre el funcionamiento de los procesos de la entidad, los recursos y los resultados, además de un plan de mejoras, como producto principal, y la elevación de la cultura de la calidad.

Académicos como (Lazo, 2000) definen la evaluación externa como: *"…el proceso que se realiza por personas o agentes externos a la institución que se evalúa y que no poseen vínculo o están implicados en alguna de las actividades habituales derivadas del quehacer institucional o desarrollo del programa".* Esta evaluación se basa en el informe de autoevaluación, en el correspondiente plan de mejoras; así como en las evidencias que permiten constatar los resultados del informe. Es un proceso integral, orientado a la determinación del estado, el funcionamiento y la proyección de la institución en correspondencia con lo predeterminado, a la obtención de nuevos conocimientos sobre el objeto que se evalúa para emitir juicios que contribuyen a fortalecer el proceso y a corregir las debilidades.

En el Sistema de Evaluación y Acreditación de Instituciones de Educación Superior (JAN, 2014), se define la acreditación como: *"… el proceso mediante el cual se reconoce (o certifica) la calidad de una entidad, sobre la base de la evaluación realizada respecto al cumplimiento de los estándares y criterios de calidad establecidos previamente por el organismo acreditador. Se basa en un conjunto de principios identificados como buenas prácticas en la comunidad internacional vinculada a este tema, aunque la diversidad de modelos de acreditación es extensa".*

Para llevar a cabo la evaluación y acreditación en instituciones de Educación Superior la JAN define, a través del SEA-IES, seis variables que guían todo el proceso:

1. Contexto institucional
2. Gestión de los recursos humanos
3. Formación del profesional de pregrado
4. Interacción social
5. Infraestructura y gestión de los recursos
6. Impacto

La presente investigación se enmarca específicamente en la variable número cinco: Infraestructura y gestión de los recursos.

### *1.1.1 Caracterización del estado actual del proceso de gestión de la variable infraestructura y gestión de los recursos en la Universidad de las Ciencias Informáticas*

La variable infraestructura y gestión de los recursos constituye un apoyo en el desarrollo del proceso de evaluación y acreditación institucional. Según (JAN, 2014) la infraestructura es: "*toda la base material de una universidad, edificaciones, equipos e instalaciones de todo tipo que posibilitan el quehacer del centro*". Define por otro lado que, la gestión de los recursos: "*está determinada por el conjunto de acciones que se planifican, organizan, ejecutan y controlan con el fin de emplear de manera eficiente y eficaz los recursos materiales y financieros que garantizan el desarrollo de todos los procesos universitarios*".

Actualmente, la JAN define un total de cinco indicadores para guiar el proceso de evaluación de la variable infraestructura y gestión de los recursos: aseguramiento de las actividades sustantivas, recursos informáticos, aseguramiento a la residencia estudiantil, aseguramiento a la vitalidad de la institución y recursos y desempeño de la actividad económica-financiera. Estos indicadores recogen un conjunto de criterios, que son evaluados de dos formas: a través de un fichero Excel y otras vías de obtención de información como entrevistas, encuestas y observaciones en cada una de las áreas involucradas en el proceso.

Los indicadores de evaluación especificados en el Excel proporcionado por la JAN, como apoyo al proceso de evaluación de la variable en cuestión, son:

- Estado de las aulas
- Condiciones y actualización de los laboratorios
- Materiales e insumos para el desarrollo de los procesos sustantivos
- Visibilidad, acceso y prestación de servicios en la red nacional y otras redes
- Estado técnico, comodidad y ambientación de la residencia estudiantil
- Calidad de la alimentación
- Servicios sociales, deporte y recreación
- Suministro energético
- Suministro de agua
- Transportación

- Recursos humanos para la gestión económica - financiera
- Recursos técnicos materiales para el desarrollo de la gestión económico - financiera
- Resultados de las actividades de control a la gestión económico - financiera

Estos se agrupan en más de 19 tablas como aulas, laboratorios, equipos y edificios de beca, alimentación, extensión, transporte, contabilidad, entre otras. En cada una de ellas se debe registrar una serie de parámetros a los cuales se les otorga una evaluación (B, R, M) y otros datos cuantitativos y cualitativos, que dependen del tipo de área que se esté evaluando.

Para desarrollar todo el proceso de recopilación y manejo de la información de la variable en cuestión, en la UCI existe un equipo multidisciplinario, encargado de compilar la información que se precisa a través del fichero Excel. Para ello cada integrante tiene la responsabilidad de entrevistarse con el área que se le haya asignado y solicitar los datos que se requieren. Estos pueden ser entregados en diferentes formatos, como excel, word, pdf o tener algún software donde se tenga automatizada parte de la información que se necesita. En un esfuerzo conjunto se completan las tablas que le haya correspondido al responsable de esta área, quien a su vez envía este resultado parcial al jefe del equipo de la variable. Este último recoge toda la información de cada una de las áreas y conforma un Excel final que incluye todos los datos.

Teniendo en cuenta estos datos y la aplicación de otras técnicas, se realiza un informe general de la variable con las diferentes fortalezas y debilidades específicas. De igual forma se elabora un plan de mejoras asociado a las debilidades detectadas, que cuenta con una o varias acciones que se planifican para contrarrestar cada debilidad, el responsable de ejecutar cada una de ellas y la fecha de cumplimiento de las mismas. De manera general las fortalezas y debilidades detectadas se refinan a medida que se realizan diferentes rondas de reuniones con todos los involucrados, hasta delimitar las que son más representativas de la variable. Estas últimas son las que se entregan en el informe final en el cual se integran las correspondientes a las demás variables para la redacción del informe de la UCI. Toda esta información se envía a través del correo electrónico o se entrega de manera personal a cada uno de los implicados en el proceso de evaluación.

Durante todo este proceso se pudo identificar que:

- Existe una alta dispersión de la información pues cada área gestiona sus datos a través de mecanismos propios, en diferentes momentos que tienen planificados, lo cual provoca que a nivel

central no sea posible una constante actualización. Además, los formatos y estilos en que se almacenan los datos son diversos, incluso a lo interno de cada área, para reflejar distintos tipos de información. No se cuenta con plantillas o estándares similares para la recopilación de la información a nivel general, de ahí que el proceso se torne más engorroso.

- La disponibilidad se ve afectada pues no existe un medio al cual se pueda acceder en cualquier momento que se necesite para revisar la información de un área. Esto mayormente está determinado por el modo en que está almacenada la información. Dada la gran cantidad de datos que se gestionan no se pueden generar de forma rápida reportes particulares o generales que se requieran en la institución.
- La integridad se ve afectada ya que toda la compilación y procesamiento de la información depende de diferentes actores que intervienen en el proceso. Estos, obtienen los datos de diferentes áreas, en diferentes formatos y tienen la responsabilidad de agruparlos en un solo formato. Cuando se realiza este tipo de operación de forma manual y se realizan cálculos, existen altas probabilidades de que hayan errores en los resultados que influyen en los informes finales.

## 1.2 Proceso de gestión de información. Integridad, centralidad y disponibilidad de la información

A partir de la caracterización del proceso de evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos, se evidencia que la información que se genera durante todo este proceso, es manipulada por un conjunto de personas, las cuales pueden modificarla o eliminarla, sin importar el área a la que pertenecen. Se pudo determinar también, que la información se encuentra dispersa por toda la institución y que es complejo tener acceso a todo el contenido de manera rápida y desde un mismo sitio.

Investigadores como (Woodman, 1985), afirman que la gestión de información juega un papel fundamental, ya que se encarga de todo lo relacionado con la obtención de la información adecuada, en la forma correcta, para la persona indicada, en el momento oportuno y en el lugar apropiado. De esta forma, el sistema de procesamiento o gestión de información usado actualmente para desarrollar el proceso de evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos no es favorable.

Luego de este análisis, la autora de la investigación considera necesario abordar y definir el proceso de gestión de información, así como las variables: integridad, centralidad y disponibilidad de la información.

Autores como (Capote, y otros, 2003) expresan que: ..." *la información para que pueda utilizarse y genere ventajas competitivas debe tener tres características básicas: completa, confiable y oportuna*". Por otro lado (Soto, y otros, 2006) definen la gestión de información como: "*el conjunto de actividades realizadas con el fin de controlar, almacenar y, posteriormente, recuperar adecuadamente la información producida, recibida o retenida por cualquier organización en el desarrollo de sus actividades*". Mientras que (Bartle, 2011) define gestión de información como: "*el proceso de analizar y utilizar la información que se ha recabado y registrado para permitir a los administradores (de todos los niveles) tomar decisiones documentadas*".

En este contexto, los objetivos principales de la gestión de información según (Pérez-Montoro , y otros, 2010) son:

- Maximizar el valor y los beneficios derivados del uso de la información.
- Minimizar el costo de adquisición, procesamiento y uso de la información.
- Determinar responsabilidades para el uso efectivo, eficiente y económico de información.
- Asegurar un suministro continuo de la información.

Una vez analizados todos los aspectos mencionados con anterioridad y luego de un estudio del tema en varias bibliografías, la autora de la actual investigación considera necesario tener en cuenta los criterios de (Capote, y otros, 2003), (Pérez-Montoro , y otros, 2010) y (Soto, y otros, 2006) sobre la gestión de información definiéndola como un conjunto de actividades.

Como bien asegura (Capote, y otros, 2003) con anterioridad, la información debe ser completa. Según (DRAE), esta palabra es sinónimo de "íntegro", que se refiere a: "que no carece de ninguna de sus partes". Desde el punto de vista informático (McGraw-Hill, 2001) considera que la integridad representa: *"completitud y corrección de los datos almacenados en una computadora, especialmente después de que ésta ha sido manipulada de alguna forma"*. A su vez, (IEEE Std 610.12, 1990) define la integridad como: *"el grado en que un sistema o componente impide el acceso no autorizado a, o la modificación de los programas informáticos o los datos".*

Para el marco de la investigación se propone que se tenga en cuenta la integridad de la información como el grado de completitud, precisión y conformidad al valor esperado de la información después que esta ha sido almacenada de alguna forma y es requerida para su uso.

Otros de los pilares de la gestión de información es la disponibilidad. Para definir lo que representa, la propia autora coincide con lo planteado en (IEEE Std 610.12, 1990), donde es definida como: "*el grado en que un sistema o componente está operativo y accesible cuando se requiere para su uso. A menudo se expresa como una probabilidad.».*

Por último, para comprender el concepto de centralidad de los datos en un sistema de gestión de información, es necesario analizar lo planteado por (Colectivo, 2012) cuando define la centralización como: "*la acción y efecto de centralizar, o sea, reunir varias cosas en un centro común o a hacer que distintas cosas dependan de un poder central*". Además de tener en cuenta que (Urrutia Arriba, 2011) considere, que la centralización de los sistemas de información tiene la cualidad de permitir un control más sencillo, ya que es la mejor forma de captar, manipular y usar la información cuando es necesario que un gran número de usuarios puedan acceder a ella.

## 1.3 Análisis de soluciones existentes que contribuyen en la investigación

Como plantea (Rubio Oca, 2007), la adopción de sistemas de evaluación y acreditación de la Educación Superior, es un hecho generalizado, siendo múltiples las instituciones de este tipo que los tienen incorporados en su quehacer. A pesar de ello, no se conoce, aún, que estos sistemas se encuentren automatizados en su totalidad. Con el objetivo de fundamentar lo anteriormente expuesto, se realiza un análisis a nivel nacional e internacional, identificándose soluciones que de una forma u otra tributan a la investigación.

En el contexto **internacional** se evidencia que:

A partir de una revisión bibliográfica de la información relacionada con el proceso de evaluación y acreditación institucional a nivel global, y particularmente de la variable infraestructura y gestión de los recursos; se llegó a la conclusión que no existe ningún sistema informático a escala internacional que gestione de manera íntegra toda la información que se precisa para desarrollar dicho proceso en el contexto de la universidad cubana.

Sin embargo, recientemente fue lanzado al mercado un software privativo de origen chileno dirigido y comercializado al sector universitario de ese país, para apoyar a las universidades en el proceso de Acreditación de Universidades. A continuación se describe el mismo:

# Capítulo 1. Proceso de evaluación y acreditación

**ISOTools University**, es un software que está orientado a automatizar el proceso de Acreditación de Universidades chilenas, a través de la realización de autoevaluaciones, el establecimiento de prioridades, la puesta en marcha de acciones específicas y la evaluación de los resultados alcanzados. Entre sus principales peculiaridades se destacan que: en su modalidad de contratación Cloud, ISOTools permite llevar a cabo sus gestiones desde cualquier lugar con conexión a internet. Es ajeno al tipo de dispositivo que utilice el usuario (dispositivos móviles, ordenadores o tablets). Por lo que se considera un sistema multidispositivo. Además que a través de las copias de seguridad, garantiza la integridad, seguridad y confidencialidad de los datos.

Es importante enfatizar que ISOTools a pesar de ser un sistema que automatiza procesos relacionados con la evaluación y acreditación de instituciones de Educación Superior, no contribuye de manera directa a la actual investigación. Ya que al ser un software privativo desarrollado a la medida para el contexto de las universidades chilenas, no se tiene acceso a este de forma gratuita.

Mientras que en el ámbito **nacional** se obtuvo que:

En la UCI, existe un número importante de sistemas informáticos los cuales se encargan de informatizar diferentes procesos de cada una de las áreas del campus universitario. Muchas de estas soluciones no están desplegadas hoy en día y otras no recogen toda la información que se necesita. No obstante a continuación se analizan algunos de estos sistemas los cuales de una forma u otra contribuyen en la investigación.

**Sistema de Gestión de Beca**, es un sistema desarrollado en la UCI, que brinda, a través de una aplicación web, información correspondiente a las actividades y procesos realizados en el área de la residencia estudiantil, tales como: el estado constructivo y mobiliario de los apartamentos y edificios. Además ofrece soporte a las personas involucradas en la toma de decisiones de corte evaluativo u organizativo. Este sistema cuenta con un conjunto de funcionalidades importantes, pero particularmente, dos de estas son de interés para esta investigación: controlar defectaciones y controlar medios. La primera tramita el estado de los diferentes apartamentos y la segunda se encarga de registrar los medios físicos pertenecientes a cada uno de estos apartamentos y darle seguimiento a los mismos.

**Sistema para Auditoría y Control de Activos Fijos Tangibles (AFT),** es una solución web desarrollada por la UCI, utilizando el *framework* Symfony. Es un sistema de apoyo para las auditorías realizadas con frecuencia a los disímiles Activos Fijos Tangibles (AFT) existentes en la institución. El mismo, se sustenta

en la adquisición de datos a través de un cliente desktop que permite obtener los medios físicos de un local mediante etiquetas en forma de código de barras. Genera un conjunto de reportes en formato PDF que contribuyen a elevar el control de los AFT y su proceso de identificación, así como un reporte general de los códigos de barras, en un documento Word, con la información de los AFT para ser adherido a estos y evitar errores en la escritura manual de los mismos. Es de provecho para la presente investigación, estudiar cómo se efectúa la generación del reporte general a través de un documento Word.

**Sistema de Gestión de Libros para el área de Aseguramiento Técnico Material (ATM**), es una aplicación web, desarrollada con el CMS Drupal, que gestiona toda la información referente a los libros distribuidos en la UCI. En este sistema el administrador del área de Aseguramiento Técnico Material lleva un control del estado de los disímiles libros que existen actualmente, así como el tipo de distribución sigue (normada o no). Esta propuesta es de utilidad en la presente investigación para estudiar el mecanismo de desagregación o distribución de los recursos materiales no normados sobre qué base o criterio se entrega y si las decisiones corresponden a acuerdos colegiados o a decisiones administrativas.

**Sistema de Entrega Digital (SED)**, es una aplicación web que permite gestionar de forma organizada la cantidad de medios de cómputo de los laboratorios de la UCI, así como las incidencias ocurridas en el mismo. Este proceso hace uso de la firma digital, método criptográfico mediante el cual se garantiza la validez jurídica de los documentos generados. Es considerado como un sistema de control de inventarios, siendo muy útil en el proceso de traspaso de responsabilidad material de los medios del laboratorio. Es de utilidad para la autora de la actual investigación, cómo se controla la cantidad de cómputos en los diferentes laboratorios del campus universitario, así como el estado general de estos locales.

**Conclusiones de los sistemas estudiados**

El estudio de los sistemas anteriores permitió identificar un conjunto de funcionalidades básicas, así como componentes esenciales que debe poseer el sistema a desarrollar. Luego del análisis realizado se concluye que estas aplicaciones constituyen una base para el diseño e implementación de la propuesta, ya que no es posible la adopción de ninguno de estos sistemas porque no presentan de manera integrada todas las funcionalidades que se requieren para satisfacer las necesidades del problema en cuestión. Además que cada uno responde a un negocio específico y algunos fueron desarrollados con tecnologías, que en la actualidad son obsoletas.

Asimismo, se pudo corroborar con algunos fundadores de la Junta de Acreditación Nacional que fueron entrevistados (Ver Anexo 1), que particularmente en Cuba, en ninguna de las instituciones de Educación Superior existe algún sistema que automatice toda la información necesaria.

## 1.4 Tendencias y tecnologías actuales

En este apartado se definen las tecnologías que permiten organizar y preparar el proceso de desarrollo del sistema que se desea construir. Se selecciona la metodología de desarrollo, el lenguaje de modelado, las herramientas CASE, los lenguajes de programación, el servidor web y el sistema gestor de base de datos a utilizar. Se justifica dicha selección y se realiza una descripción de los mismos para el conocimiento de sus características, funcionalidades y ventajas.

### *1.4.1 Metodología de desarrollo de software. Proceso Unificado Abierto (OpenUP).*

Para (Brooks, 2011), las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software. En ellas se va indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, demostrando además qué personas deben participar en el desarrollo de las actividades y qué rol deben de tener, entre otros elementos.

Estudiosos como (Abrahamsson, y otros, 2002), alegan que existe una gran cantidad de metodologías de desarrollo. Teniendo en cuenta la filosofía de desarrollo, aquellas que ponen un mayor énfasis en la planificación y control del proyecto, en la especificación precisa de requisitos y el modelado; reciben el apelativo de metodologías tradicionales. Mientras que las que permiten la incorporación del cliente como un miembro más del equipo de desarrollo, permitiendo además dar respuestas rápidas a los continuos cambios que puedan producirse, son llamadas metodologías de desarrollo ágil.

En función del estudio realizado y las características del equipo de desarrollo y de la solución propuesta en cuestión se determina el empleo de una metodología ágil, ya que el tiempo de desarrollo es limitado, el proyecto que se pretende desarrollar es relativamente pequeño, así como el equipo responsable del desarrollo del mismo. Se necesita solo la documentación imprescindible para dar soporte al sistema y a su proceso de construcción. Para realizar una selección más acertada del enfoque de la metodología a utilizar, se aplicó el método propuesto por Boehm y Turner (Ver Anexo 2).

Una vez determinado que el enfoque de la metodología es ágil, se procede a la selección de la misma, optando en este caso por OpenUp. Fundamentada principalmente en sus características distintivas: apropiada para proyectos pequeños y de bajos recursos, dirigida por casos de uso, centrada en la arquitectura, además permite disminuir las probabilidades de fracaso e incrementar las probabilidades de éxito. Detecta errores tempranos a través de un ciclo iterativo y evita la elaboración de documentación, diagramas e iteraciones innecesarias requeridas por otras metodologías. Posee un enfoque centrado al cliente y con iteraciones cortas ajustándose plenamente a las necesidades para el desarrollo de la propuesta de solución.

### *1.4.2 Tecnologías y lenguajes empleados en programación*

**Lenguaje de modelado. UML v2.0**

En las ciencias informáticas un lenguaje de modelado es un recurso utilizado para especificar, visualizar, construir y detallar un artefacto generado durante el desarrollo de un software (Duggan, y otros, 2012). El lenguaje más estandarizado actualmente es Lenguaje de modelado unificado (UML por sus siglas en inglés). (Fowler, 2007) certifica que UML 2.0 prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan; posibilitando así visualizar, especificar y documentar los artefactos o toda información que se obtiene o modifica durante un proceso de desarrollo de software.

Este lenguaje de modelado es asumido en la presente investigación para modelar y diseñar el ciclo completo de desarrollo de software de la solución propuesta. Específicamente los diagramas de casos de uso, clases del diseño, vistas lógicas, despliegue y componentes.

**Lenguajes de programación**

(Meyer, 2009) denomina lenguaje de programación a: "*la representación diseñada para expresar un proceso o algoritmo que pueden ser interpretados por máquinas y ejecutarse mediante bloques de instrucción. Están compuestos por símbolos y reglas que definen la estructura de cada lenguaje*".

**Lenguaje de programación del lado del servidor: PHP v7.0.5**

PHP, acrónimo de (Hypertext Pre Processor) constituye un lenguaje de programación interpretado, de propósito general y ampliamente difundido en el mundo. Diseñado originalmente para la creación de páginas

web dinámicas, principalmente en interpretación del lado del servidor, aunque puede ser incrustado dentro de código HTML.

Lenguaje concebido para incrementar el dinamismo de las páginas web, aprovechando los recursos de las redes informáticas y los escasos requisitos de hardware que solicita el lenguaje para que sus aplicaciones funcionen correctamente. Además, permite aplicar técnicas de programación orientada a objetos. Es un lenguaje completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en bases de datos. Actualmente soporta un gran número de bases de datos como: MySQL, PostgreSQL, Oracle, MS SQL Server, SybasemSQL, Informix, entre otras. De ahí su amplia utilización para el desarrollo de aplicaciones web. Se caracteriza por ser multiplataforma, se ejecuta del lado del servidor, con licencia GPL o de software libre, presenta una sintaxis cómoda orientado a objetos completamente, presenta un gran número de extensiones o bibliotecas que amplían sus funcionalidades (Bakken, y otros, 2005). Es un lenguaje fácil de asimilar y utilizar lo que permite reducir el tiempo empleado en estas funciones. Posee una comunidad de desarrolladores amplia y activa de manera que la mayor parte de los contratiempos en la implementación pueden ser solucionados con rapidez. Existe un número elevado de marcos de trabajo que toman como base este lenguaje entre los que destaca Symfony.

**Lenguaje de programación del lado del cliente: JavaScript v1.10.2**

JavaScript es uno de los recursos que han surgido para incorporar dinamismo y capacidades al lenguaje HTML. Actualmente es la tecnología más extendida en el enriquecimiento de páginas web del lado del cliente. JavaScript no es un lenguaje orientado a objetos es más bien un lenguaje script u orientado a documento, como pueden ser los lenguajes de macros que tienen muchos procesadores de texto y plantillas de cálculo. Entre las acciones típicas que se pueden realizar en JavaScript se tienen dos vertientes. Por un lado, los efectos especiales sobre páginas web y por el otro, permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que se pueden crear páginas interactivas. Es un lenguaje interpretado con la filosofía orientada a objetos (Egulliz, 2009).

**Lenguaje de estilos: Cascading Style Sheets v3 (CCS3)**

CSS es un lenguaje de hojas de estilos en cascada creados para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML. CSS 3 está dividido en varios documentos llamados módulos. Cada módulo cuenta con nuevas capacidades, sin afectar la compatibilidad de la versión estable anterior. Al hacer referencia a módulos, se pueden nombrar más de cincuenta, sin embargo, cuatro de ellos han sido publicados como recomendaciones formales, y se componen de lo siguiente:

- *Media Queries* (publicado en 2012)
- *Namespaces* (publicado en 2011)
- *Selectors Level* 3 (publicado en 2011)
- Color (publicado en 2011).

Estas hojas de estilos permiten crear diseños atractivos y personalizados de acuerdo a los requisitos no funcionales de apariencia o interfaz externa (Egulliz, 2007).

**Lenguaje de marcado: Hyper Text Markup Language v5 (HTML5)**

HTML, es un lenguaje de marcado usado para estructurar y presentar el contenido para la web. Con el uso de HTML 5, se puede reducir la dependencia de los *plugins* que deben estar instalados para poder visualizar una determinada web y se amplía el desarrollo de aplicaciones que pueden ser usadas en una multiplicidad de dispositivos. Permite que los usuarios accedan a sitios web sin estar conectados a internet. Cuenta con nuevas etiquetas que ayudan a los desarrolladores a mejorar la estructura de las páginas como: <*header*> para las cabeceras, <*article*> para definir un artículo, un comentario de usuario o una publicación independiente dentro del sitio, < section> define todo tipo de secciones dentro de un documento, <*footer*> para el pie de página. Este es soportado por las versiones actuales de los navegadores Mozilla Firefox, Chrome, Chromiun, Safari e Internet Explorer. (W3C, 2016)

### *1.4.3 Marcos de trabajo*

Un *framework* o marco de trabajo es un software que se puede personalizar e intercambiar para favorecer en tiempo el desarrollo de una aplicación. Se puede considerar como otra aplicación incompleta y configurable a la que se le añaden las últimas piezas para construir una aplicación final. Ayudan a desarrollar aplicaciones con mayor rapidez, pues poseen una estructura definida y una organización para el desarrollo y mantenimiento del software desarrollado (Potencier, et al., 2008).

**Symfony v2.8.17**

Symfony2 es un *framework* de desarrollo PHP caracterizado por su madurez, estabilidad y por el grado de documentación con la que cuenta. Explota al máximo todas las nuevas características de PHP 5.3 y por eso es uno de los de mejor rendimiento. Su arquitectura interna está completamente desacoplada, lo que permite reemplazar o eliminar fácilmente aquellas partes que no encajen en los proyectos. (Egulliz, 2013)

Entre las características de este *framework* se pueden destacar:

- Fácil de instalar y configurar en la mayoría de las plataformas.
- Independiente del Sistema Gestor de Bases de Datos (SGBD). Su capa de abstracción y el uso de Doctrine2, permiten cambiar con facilidad el SGBD en cualquier fase del proyecto.
- Utiliza programación orientada a objetos.
- Sigue la mayoría de las mejores prácticas y patrones de diseño para la web.
- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
- Posee una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo.

**Bootstrap v3.3.7**

Bootstrap es un *framework* o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.

**Características de Bootstrap:**

- Compatible con la mayoría de los navegadores web.
- Se integra perfectamente con las principales librerías JavaScript, por ejemplo, JQuery.
- Ofrece un diseño sólido a través del uso de estándares como CSS3/HTML5.
- Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como tabletas y móviles a distintas escalas y resoluciones.

La gran popularidad alcanzada por este *framework* permite encontrar una amplia gama de componentes visuales que agilizan el tiempo de desarrollo de las interfaces de usuario. (W3C, 2016)

**JQuery v1.11.1**

JQuery es una biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. JQuery es la biblioteca de JavaScript más utilizada. Ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con

las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio (Potencier, et al., 2008).

**Características de Jquery:**

- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- Soporta extensiones.
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes.
- Compatible con los navegadores Mozilla Firefox, Internet Explorer, Safari, Opera y Google Chrome.

### *1.4.4 Herramientas a utilizar*

**Herramienta de modelado. Visual Paradigm for UML v8.0**

Visual Paradigm es una herramienta de Ingeniería de Software Asistida por Computadora (*CASE*, por sus siglas en inglés) para el modelado con UML, BPMN y otros lenguajes de modelado. Soporta el ciclo de vida completo del desarrollo de software, permitiendo a los ingenieros diseñar, integrar y modelar visualmente los distintos artefactos que tributan a la documentación. Se integra fácilmente con varios entornos de desarrollo y permite la generación semiautomática de código a partir de los diagramas construidos y también se puede llevar a cabo la ingeniería inversa para refinar los modelos. Es una herramienta eficiente, fácil de usar y con soporte multiplataforma que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Tiene licencia dual, gratuita y comercial. (Visual Paradigm Group, 2011)

**Entorno de Desarrollo Integrado (IDE). PhpStorm v2016.3.2**

Según (Gadja, 2007) "*PHPStorm es un IDE diseñado específicamente para desarrollar aplicaciones de internet escritas en PHP. Las principales características se dividen en las siguientes categorías*".

- Sincronización de ficheros.
- Depuración.
- Edición y refactorización de código.

El propio autor aborda también que "*las aplicaciones modernas, consisten en muchos archivos, librerías, módulos y clases almacenados en diferentes carpetas. PHPStorm, ayuda a la navegación entre ellas. Los archivos abiertos se encuentran distribuidos en pestañas y es posible navegar entre ellos con el mouse y los accesos del teclado*".

**Servidor Web. Apache v2.4**

Apache constituye una tecnología gratuita de código abierto. Es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos. Permite la creación de sitios web dinámicos mediante el uso de lenguajes de scripting como PHP, JavaScript, Python, entre otros. Se ejecuta en varios sistemas operativos. Posee una arquitectura modular que admite ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de módulos, para el desarrollo de módulos específicos. (Apache Software Foundation., 2000)

**Sistema Gestor de Base de Datos. PostgreSQL v9.5.0**

Los sistemas de gestión de bases de datos (en inglés Database Management System, abreviado DBMS) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. El propósito general de los sistemas de gestión de bases de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante para una organización. SEAIVI utiliza como motor de base de datos PostgreSQL 9.5.0

PostgreSQL es un servidor de base de datos relacional libre. Es una alternativa a otros sistemas de bases de datos de código abierto (como MySQL, Firebird y MaxDB), así como sistemas propietarios como Oracle o DB2. Presenta como características principales claves ajenas también denominadas Llaves Ajenas o Llaves Foráneas, disparadores, vistas, implementación del estándar SQL92/SQL99, integridad transaccional, acceso concurrente, capacidad de albergar programas en el servidor en varios lenguajes, herencia de tablas y tipos de datos (PostgreSQL, 2015).

## Conclusiones parciales

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones:

- El proceso de evaluación y acreditación institucional, no cuenta con una gestión de información precisa. Se evidencia la carencia de centralidad, integridad y disponibilidad de la información que se precisa durante el desarrollo de dicho proceso.
- La investigación realizada sobre las diferentes soluciones existentes permitió concluir que ninguna se ajusta a las necesidades expuestas en la investigación, ya que las funcionalidades que poseen se ajustan a un negocio específico, no generalizables a otros entornos, desarrolladas con tecnologías obsoletas, no configuradas para su reutilización y no cumplen con las tendencias y estándares del desarrollo web actuales.
- La aplicación del método o modelo propuesto por Barry Boehm y Richard Turner ayudó a la selección del enfoque la metodología de desarrollo ágil.
- Las herramientas y tecnologías escogidas para el desarrollo de la aplicación, fueron determinadas por las características de ésta, entre las que se encuentran el *framework* Symfony2, el gestor de base de datos PostgreSQL y Visual Paradigm como herramienta CASE para la generación de artefactos.

# Capítulo 2. Requisitos y arquitectura del sistema

En el presente capítulo se exponen las principales características y cualidades de la propuesta de solución, mediante la identificación de los requisitos funcionales y no funcionales con los que debe cumplir. Se incluye la definición y descripción de los actores del sistema y las relaciones entre ellos, además de la estructuración del sistema, así como los diagramas de casos de uso del sistema y las descripciones textuales de estos casos de uso. Se describen los conceptos relacionados con las definiciones patrones de diseño de software empleados, definiendo de esta forma la vista lógica y extendida de la aplicación. Especificando finalmente los diagramas de clases del diseño propuestos, los patrones de diseño a utilizar y modelo de datos.

## 2.1 Requisitos de Software

Una etapa inicial y muy importante dentro del proceso de la Ingeniería de Software, es la Ingeniería de Requisitos. Se realiza el proceso de descubrir, analizar, escribir y verificar los servicios y restricciones, del sistema de software que se desea producir; este proceso se realiza mediante la obtención, el análisis, la especificación, la validación y la administración de los requisitos del software. Los requisitos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar (Jacobson, y otros, 1999).

### *2.1.1 Especificación de Requisitos Funcionales*

Los requisitos funcionales (RF) son declaraciones de los servicios que deben proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (Sommerville, 2005). A partir de las entrevistas realizadas y la aplicación de otras técnicas de recopilación de información se definieron los siguientes requisitos funcionales:

**Tabla 1** Especificación de Requisitos Funcionales

| | | |
|---|---|---|
| **Paquete Facultad** | | |
| **RF1.**Adicionar aula | **RF2.**Mostrar datos de aula | **RF3.**Editar datos de aula |
| **RF4.**Eliminar aula | **RF5.**Buscar información del aula | **RF6.**Generar reporte del estado de las aulas |
| **RF7.**Adicionar laboratorio | **RF8.**Mostrar datos de laboratorio | **RF9.**Editar datos de laboratorio |
| **RF10.**Eliminar laboratorio | **RF11.**Buscar información del laboratorio | **RF12.**Generar reporte del estado de los laboratorio |
| **RF13.**Adicionar laboratorio de computación | **RF14.**Mostrar datos de laboratorio de computación | **RF15.**Editar datos de laboratorio de computación |
| **RF16.**Eliminar laboratorio de computación | **RF17.**Buscar laboratorio de computación | **RF18.**Generar reporte del estado de los laboratorios de computación |
| **Paquete Economía** | | |
| **RF19.**Adicionar datos de Aseguramiento Técnico Material | **RF20.**Mostrar datos de Aseguramiento Técnico Material | **RF21.**Editar datos de Aseguramiento Técnico Material |
| **RF22.**Eliminar datos de Aseguramiento Técnico Material | **RF23.**Buscar Aseguramiento Técnico Material | **RF24.**Generar reporte del estado del Aseguramiento Técnico Material |
| **RF25.**Adicionar datos del área de Contabilidad | **RF26.**Mostrar datos del área de Contabilidad | **RF27.**Editar datos del área de Contabilidad |
| **RF28.**Eliminar datos del área de Contabilidad | **RF29.**Buscar datos del área de Contabilidad | **RF30.**Generar reporte del área de Contabilidad |

| | | |
|---|---|---|
| **RF31.**Adicionar información sobre los recursos técnicos materiales para el desarrollo de la gestión económico - financiera | **RF32.**Mostrar información sobre los recursos técnicos materiales para el desarrollo de la gestión económico - financiera | **RF33.**Editar información sobre los recursos técnicos materiales para el desarrollo de la gestión económico - financiera |
| **RF34.**Eliminar información sobre los recursos técnicos materiales para el desarrollo de la gestión económico - financiera | **RF35.**Buscar información sobre los recursos técnicos materiales para el desarrollo de la gestión económico - financiera | **RF36.**Generar reporte sobre los recursos técnicos materiales para el desarrollo de la gestión económico - financiera |
| **RF37.**Adicionar equipo de beca | **RF38.**Mostrar datos de equipo de beca | **RF39.**Editar datos de equipo de beca |
| **Paquete RRHH** | | |
| **RF40.**Adicionar información sobre los recursos humanos para la gestión económica - financiera | **RF41.**Mostrar información sobre los recursos humanos para la gestión económica - financiera | **RF42.**Editar información sobre los recursos humanos para la gestión económica - financiera |
| **RF43.**Eliminar información sobre los recursos humanos para la gestión económica - financiera | **RF44.**Buscar información sobre los recursos humanos para la gestión económica - financiera | **RF45.**Generar reporte sobre la información de los recursos humanos para la gestión económica - financiera |
| **Paquete Extensión** | | |
| **RF46.**Eliminar equipo de beca | **RF47.**Buscar equipo de beca | **RF48.**Generar reporte sobre los equipos de beca |
| **RF49.**Adicionar edificio de beca | **RF50.**Mostrar datos de edificio de beca | **RF51.**Editar datos de edificio de beca |
| **RF52.**Eliminar edificio de beca | **RF53.**Buscar edificio de beca | **RF54.**Generar reporte del estado de los edificio de beca |
| **RF55.**Adicionar información del área de Extensión Universitaria | **RF56.**Mostrar datos del área de Extensión Universitaria | **RF57.**Editar datos del área de Extensión Universitaria |

| | | |
|---|---|---|
| **RF58.**Eliminar datos del área de Extensión Universitaria | **RF59.**Buscar información del área de Extensión Universitaria | **RF60.**Generar reporte sobre la información del área de Extensión Universitaria |
| **RF61.**Adicionar instalación deportiva / social | **RF62.**Mostrar datos de instalación deportiva / social | **RF63.**Editar datos de instalación deportiva / social |
| **RF64.**Eliminar instalación deportiva / social | **RF65.**Generar reporte del estado de las instalaciones deportivas / sociales | |
| **Paquete Ciudad** | | |
| **RF66.**Adicionar información del área de Mantenimiento | **RF67.**Mostrar información del área de Mantenimiento | **RF68.**Editar información del área de Mantenimiento |
| **RF69.**Eliminar información del área de Mantenimiento | **RF70.**Buscar información del área de Mantenimiento | **RF71.**Adicionar información del área de Inversiones |
| **RF72.**Mostrar información del área de Inversiones | **RF73.**Editar información del área de Inversiones | **RF74.**Eliminar información del área de Inversiones |
| **RF75.**Buscar información del área de Inversiones | **RF76.**Generar reporte del área de Mantenimiento-Inversiones | **RF77.**Adicionar información del área de Servicios Generales |
| **RF78.**Mostrar información del área de Servicios Generales | **RF79.**Editar información del área de Servicios Generales | **RF80.**Eliminar información del área de Servicios Generales |
| **RF81.**Buscar información del área de Servicios Generales | **RF82.**Generar reporte sobre la información del área de Servicios Generales | **RF83.**Adicionar auto al parque de vehículos |
| **RF84.**Mostrar datos del auto del parque de vehículos | **RF85.**Editar datos del auto del parque de vehículos | **RF86.**Eliminar auto del parque de vehículos |
| **RF87.**Buscar datos del auto del parque de vehículos | **RF88.**Generar informe del parque de vehículos | **RF89.**Adicionar información del área de Transporte |

| | | |
|---|---|---|
| **RF90.**Mostrar información del área de Transporte | **RF91.**Editar información del área de Transporte | **RF92.**Eliminar información del área de Transporte |
| **RF93.**Buscar información del área de Transporte | **RF94.**Generar reporte sobre la información del área de Transporte | **RF95.**Adicionar información del área de Alimentación |
| **RF96.**Mostrar datos del área de Alimentación | **RF97.**Editar datos del área de Alimentación | **RF98.**Eliminar datos del área de Alimentación |
| **RF99.**Buscar información del área de Alimentación | **RF100.**Generar reporte sobre la información del área de Alimentación | **RF101.**Adicionar información sobre cocina |
| **RF102.**Mostrar información sobre de cocina | **RF103.**Editar información sobre de cocina | **RF104.**Eliminar información sobre la cocina |
| **RF105.**Buscar datos de cocina | **RF106.**Adicionar comedor | **RF107.**Mostrar datos del comedor |
| **RF108.**Editar datos del comedor | **RF109.**Eliminar comedor | **RF110.**Buscar datos de comedor |
| **RF111.**Generar reporte del estado de las cocinas-comedores | | |
| **Paquete Supervisión** | | |
| **RF112.**Adicionar datos del área de Auditorías y Control | **RF113.**Mostrar datos del área de Auditorías y Control | **RF114.**Editar datos del área de Auditorías y Control |
| **RF115.**Eliminar datos del área de Auditorías y Control | **RF116.**Buscar datos del área de Auditorías y Control | **RF117.**Generar reporte del área de Auditorías y Control |
| **Paquete Tecnología** | | |
| **RF118.**Adicionar datos del área de Informatización | **RF119.**Mostrar datos del área de Informatización | **RF120.**Editar datos del área de Informatización |

| | | |
|---|---|---|
| **RF121.**Eliminar datos del área de Informatización | **RF122.**Buscar datos del área de Informatización | **RF123.**Generar reporte del área de Informatización |
| **Paquete Administración** | | |
| **RF124.**Adicionar fortaleza | **RF125.**Mostrar datos de fortaleza | **RF126.**Editar datos de fortaleza |
| **RF127.**Eliminar fortaleza | **RF128.**Adicionar debilidad | **RF129.**Mostrar datos de debilidad |
| **RF130.**Editar datos de debilidad | **RF131.**Eliminar debilidad | **RF132.**Generar reporte de la variable infraestructura y gestión de los recursos en un fichero Excel |
| **RF133.**Generar el plan de mejoras asociado a la variable infraestructura y gestión de los recursos en un documento Word | **RF134.**Generar el informe final de la variable infraestructura y gestión de los recursos en un documento Word | **RF135.**Autenticar usuario |
| **RF136.**Registrar usuario | **RF137.**Editar datos de usuario | **RF138.**Eliminar usuario |
| **RF139.**Buscar usuario | **RF140.**Adicionar rol | **RF141.**Mostrar datos del rol |
| **RF142.**Editar datos del rol | **RF143.**Eliminar rol | **RF144.**Buscar rol |
| **RF145.**Adicionar Centro de Educación Superior | **RF146.**Editar datos de Centro de Educación Superior | **RF147.**Eliminar Centro de Educación Superior |
| **RF148.**Buscar Centro de Educación Superior | **RF149.**Adicionar identificador de cocina | **RF150.**Editar identificador de cocina |
| **RF151.**Eliminar identificador de cocina | **RF152.**Buscar identificador de cocina | **RF153.**Adicionar identificador de comedor |
| **RF154.**Editar identificador de comedor | **RF155.**Eliminar identificador de comedor | **RF156.**Buscar identificador de comedor |
| **RF157.**Adicionar marca de vehículo | **RF158.**Editar marca de vehículo | **RF159.**Eliminar marca de vehículo |

| | | |
|---|---|---|
| **RF160.**Buscar marca de vehículo | **RF161.**Adicionar modelo de vehículo | **RF162.**Editar modelo de automóvil |
| **RF163.**Eliminar modelo de automóvil | **RF164.**Buscar modelo de automóvil | **RF165.**Adicionar color de chapa de vehículo |
| **RF166.**Editar color de chapa de vehículo | **RF167.**Eliminar color de chapa de vehículo | **RF168.**Buscar color de chapa de vehículo |
| **RF169.**Adicionar provincia | **RF170.**Editar provincia | **RF171.**Eliminar provincia |
| **RF172.**Buscar provincia | **RF173.**Adicionar tipo de vehículo | **RF174.**Editar tipo de vehículo |
| **RF175.**Eliminar tipo de vehículo | **RF176.**Buscar tipo de vehículo | **RF177.**Adicionar ubicación de locales |
| **RF178.**Editar ubicación de locales | **RF179.**Eliminar ubicación de locales | **RF180.**Buscar ubicación de locales |
| **RF181.**Adicionar denominación de aula | **RF182.**Editar denominación de aula | **RF183.**Eliminar denominación de aula |
| **RF184.**Buscar denominación de aula | **RF185.**Adicionar denominación de laboratorio | **RF186.**Editar denominación de laboratorio |
| **RF187.**Eliminar denominación de laboratorio | **RF188.**Buscar denominación de laboratorio | **RF189.**Adicionar denominación de edificio |
| **RF190.**Editar denominación de edificio | **RF191.**Eliminar denominación de edificio | **RF192.**Buscar denominación de edificio |
| **RF193.**Adicionar denominación de instalaciones deportivas / sociales | **RF194.**Editar denominación de instalaciones deportivas / sociales | **RF195.**Eliminar denominación de instalaciones deportivas / sociales |
| **RF196.**Buscar denominación de instalaciones deportivas / sociales | **RF197.**Adicionar auditor | **RF198.**Editar datos del auditor |

| | | |
|---|---|---|
| **RF199.**Eliminar auditor | **RF200.**Buscar auditor | **RF201.**Adicionar entidades auditoras |
| **RF202.**Editar entidades auditoras | **RF203.**Eliminar entidades auditoras | **RF204.**Buscar entidades auditoras |
| **RF205.**Adicionar denominación de centro | **RF206.**Editar denominación de centro | **RF207.**Eliminar denominación de centro |
| **RF208.**Buscar denominación de centro | | |

### *2.1.2 Especificación de Requisitos no Funcionales*

**RNF1. Apariencia o interfaz externa:** La interfaz externa del producto es de fácil navegación por el usuario, sencilla y legible. Su funcionamiento es intuitivo y requiere de información mínima. Se garantiza la uniformidad de las interfaces de usuario, teniendo en cuenta que las operaciones comparables siempre se activen de la misma forma. Los conceptos y términos que se plasman en las interfaces del sistema son obtenidos de la experiencia de las personas que más utilizarán este producto. Contiene mecanismos para recuperarse ante errores. De igual forma, al cometer errores, la interfaz proporciona características de ayuda. Se incluyen elementos visuales para la selección de información siempre que sea posible, para minimizar los posibles errores.

**RNF2. Usabilidad:** El sistema puede ser utilizado por cualquier usuario que posea conocimientos informáticos básicos. El software tiene una curva de aprendizaje baja, que permite al usuario familiarizarse rápidamente con los elementos del sistema y operarlo de forma correcta en poco tiempo de uso. Cumple con la regla de los tres clics para llegar a cada una de las funcionalidades del sistema.

**RNF3. Requisitos de software:**

**Para las estaciones de trabajo:**

Las estaciones de trabajo que usarán los usuarios del sistema para acceder a la aplicación y manipular la misma deberán poseer los siguientes requerimientos de software:

- Navegador Web Internet Explorer v10.0 o superior o Mozilla Firefox v15.0 o superior o Chrome v6.0 o superior.
- Sistemas Operativos Windows 7 Service Pack 1 o GNU/Linux distribución Ubuntu 14.04.

**Para los servidores:**

**Servidor de Aplicación:**

- Servidor Web Apache en su versión 2.0 o superior
- Sistema Operativo: GNU/ Linux Distribución Ubuntu 14.04 o Windows Server 2003

**Servidor de Base de Datos**

- PostgreSQL en su versión 9.1 o superior
- Sistema Operativo: GNU/ Linux Distribución Ubuntu 14.04

**RNF4. Requisitos de hardware:**

**Para las estaciones de trabajo:**

- Tipo de procesador: Intel Pentium IV o superior
- Memoria RAM: 1024 Mb o superior
- Velocidad del procesador: 1.6 GHz o superior
- Tarjeta de red

**Para los servidores:**

**Servidor de Aplicación**

- Tipo de procesador: Intel Dual Core o superior
- Velocidad del procesador: 3.00 GHz
- Memoria RAM: 2 Gb o superior
- Disco duro: 80Gb o superior

**Servidor de Base de Datos**

- Tipo de procesador: Intel Core2Duo o superior
- Memoria RAM: 2Gb o superior
- Velocidad del procesador: 3.00 GHz
- Disco duro: 160Gb o superior
- Tarjeta de red

**RNF5. Rendimiento:** El tiempo de respuesta ante diferentes peticiones de los usuarios será inferior a los tres segundos.

**RNF6. Requisitos de seguridad:** El sistema establecerá una política de acceso basada en roles, usuarios y permisos de los usuarios para las operaciones.

## 2.2 Modelación del Sistema

Según (Jacobson, y otros, 1999), *"…un caso de uso especifica una secuencia de acciones, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto. Los diagramas de casos de uso especifican las funcionalidades y el comportamiento de un sistema mediante la interacción con los usuarios y/u otros sistemas, muestra la relación entre los actores y los casos de uso".* De acuerdo con lo antes descrito, se realiza a continuación una descripción de los actores que intervienen en el sistema y se conforma el modelo de casos de uso del sistema.

### *2.2.1 Descripción de los actores que interactúan con el sistema*

Un actor es un usuario del sistema. Esto incluye usuarios humanos y otros sistemas computacionales. El conjunto de casos de uso al que un actor tiene acceso define un rol en el sistema y el alcance de su acción. En la Tabla 2 se describen los actores que interactúan con las funcionalidades en la aplicación.

**Tabla 2** Descripción de los actores del sistema

| Actor del sistema | Descripción |
|---|---|
| Administrador | Persona encargada de gestionar los usuarios y nomencladores del sistema. Tiene acceso a todas las funcionalidades del mismo. |
| Directivo | Persona que interactúa con el sistema. Puede visualizar la información correspondiente a las diferentes áreas. Además tiene permisos para exportar esta información a un fichero Excel, así como el informe correspondiente a la variable infraestructura y gestión de los recursos y el Plan de Mejoras, ambos como documentos Word. |
| Evaluador Extensión | Persona encargada de gestionar la información de las áreas de residencia, extensión y deporte. Puede generar un fichero Excel con los diferentes reportes de los edificios y equipos de beca; así como los datos correspondientes a la Extensión Universitaria e instalaciones deportivas y sociales de la Universidad. |

| | |
|---|---|
| Evaluador Economía | Persona que gestiona la información necesaria en la evaluación de las áreas de Aseguramiento Técnico Material (ATM) y Contabilidad. Cuenta con los permisos para generar los reportes correspondientes a ATM, contabilidad y los recursos para la gestión económica-financiera. |
| Evaluador Ciudad | Persona que maneja gran cantidad de datos dentro del sistema. La misma, gestiona las áreas de Alimentos, Mantenimiento e Inversiones, Servicios Generales y Transporte. Tiene asignado los permisos necesarios para poder exportar un fichero Excel con la información asociadas a cada una de las áreas mencionadas con anterioridad: Cocina-Comedor, Mantenimiento-Inversiones, Servicios Generales, Parque de Vehículos y Transporte. |
| Evaluador Tecnología | Persona responsable de gestionar parte de la información correspondiente al proceso de informatización. Puede generar los reportes correspondientes a los laboratorios de computación y al proceso de informatización, ambos en ficheros Excel. |
| Evaluador Facultad | Persona responsable de gestionar los datos referentes al estado de las aulas, laboratorios y laboratorios de computación. Puede generar los reportes de cada una de los locales mencionadas anteriormente a través de un fichero Excel. |
| Evaluador RRHH | Persona perteneciente a la Dirección de Desarrollo del Capital Humano, que tiene la tarea de gestionar los datos de los recursos humanos para la gestión económica – financiera. Puede generar esta información en un fichero Excel. |
| Evaluador Supervisión | Persona responsable de gestionar los resultados de las actividades de control a la gestión económico – financiera de los últimos 5 años. Puede generar el reporte correspondiente a las auditorías en un fichero Excel. |

### *2.2.2 Estructuración del sistema*

Los requisitos funcionales se agruparon en 64 casos de uso del sistema auxiliándose de los patrones de casos de uso (múltiples actores: roles comunes y CRUD completo), conformando así el Modelo de Casos de Uso del Sistema, en el cual intervienen 10 actores. Para una mejor organización de los artefactos fue necesario estructurar el Diagrama de Casos de Uso del Sistema (DCUS) general en varios paquetes:

Administración, Ciudad, Economía, Extensión, Facultad, RRHH, Supervisión, Tecnología y Directivo. Los cuales agrupan un conjunto de funcionalidades, las cuales se pueden ver detalladamente en la Tabla 1, quedando finalmente de la siguiente forma:

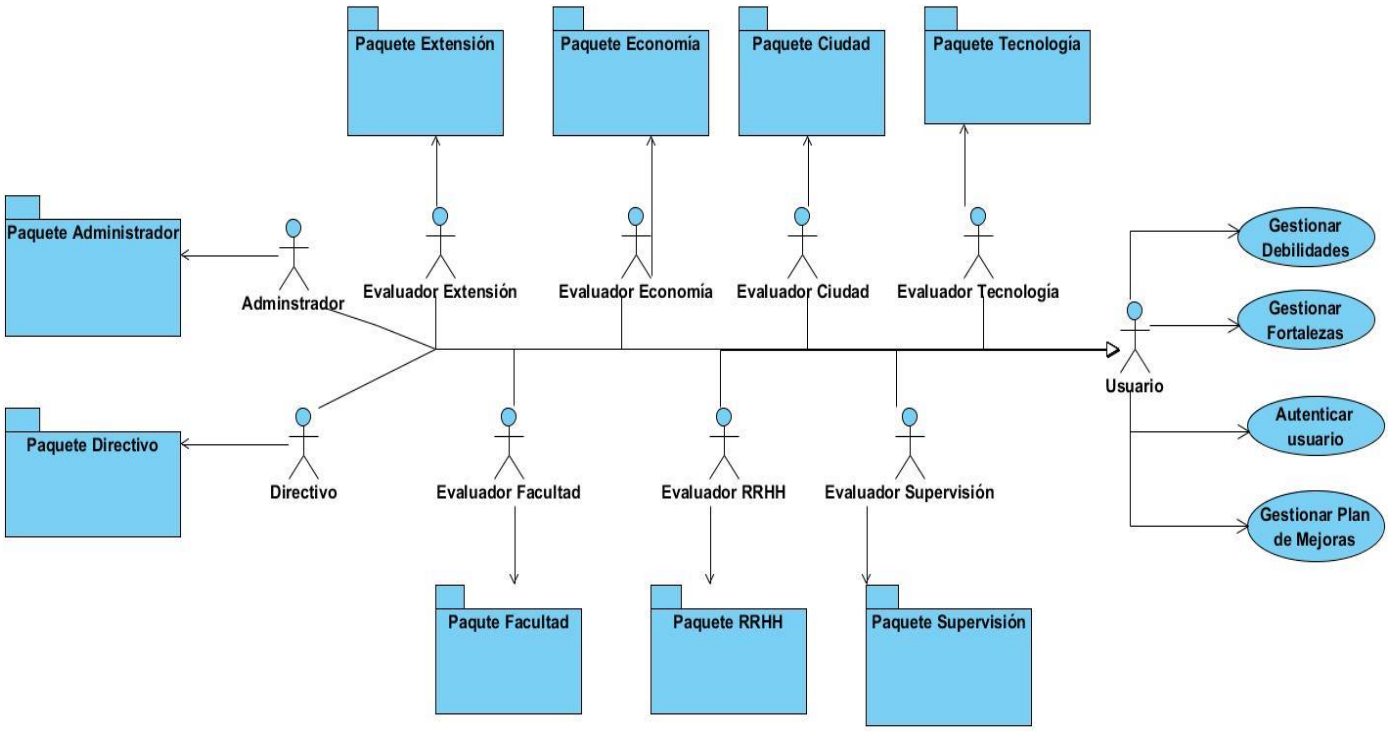



**Figura 1** Estructura del sistema a través de paquetes.

### *2.2.3 Diagrama de Casos de Uso del Sistema*

Los diagramas de casos de uso documentan el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto, los casos de uso determinan los requisitos funcionales del sistema. Su ventaja principal es la facilidad para interpretarlos, lo que hace que sean especialmente útiles en la comunicación con el cliente (Vignaga, y otros, 2014). En la Figura 2 se muestra el DCUS correspondiente al paquete Facultad, en el cual se encuentra el Caso de Uso del Sistema (CUS) Gestionar Laboratorio de Computación que se describe en el próximo epígrafe. El resto de los DCUS se encuentran en el Expediente de Proyecto.

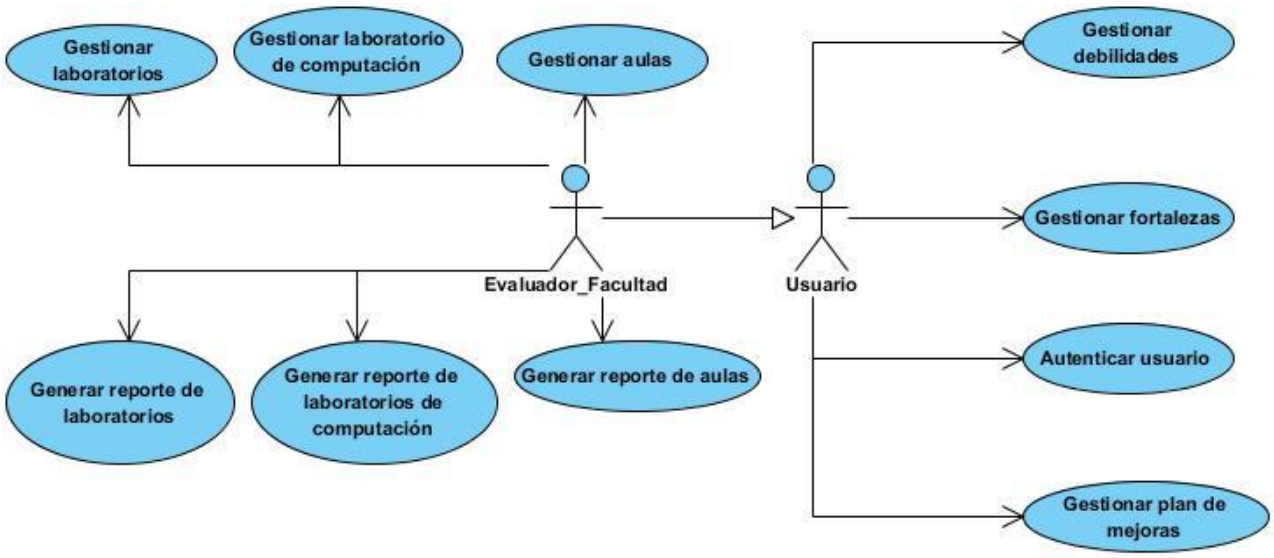



**Figura 2** Diagrama de Casos de Uso del Sistema. Paquete Facultad.

### *2.2.4 Descripción textual de los Casos de Uso del Sistema*

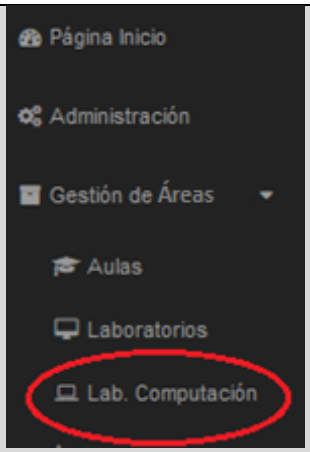
Debido a que las descripciones textuales son muy extensas, en la Tabla 3 solo se muestra la correspondiente al caso de uso Gestionar Laboratorio de Computación. El resto de las descripciones textuales se encuentran en el Expediente de Proyecto.

**Tabla 3** Descripción textual del caso de uso Gestionar Laboratorio de Computación

| | |
|---|---|
| **Caso de Uso** | Gestionar Laboratorio de Computación |
| **Actores** | Evaluador Facultad |
| **Propósito** | El objetivo fundamental de este caso de uso es gestionar toda la información que se precisa sobre el estado de los laboratorios de computación. |

| | |
|---|---|
| **Resumen** | El caso de uso inicia cuando el usuario desea realizar algunas de las siguientes operaciones: Insertar, Modificar, Eliminar, Mostrar o Buscar un Laboratorio de Computación. El mismo termina cuando han sido completadas las acciones seleccionadas. |
| **Complejidad** | Alta |
| **Referencias** | RF13, RF14, RF15, RF16, RF17 |
| **Precondiciones** | El actor tiene que estar autenticado. |
| **Flujo Normal de Eventos** | |
| **Acciones del Actor** | **Respuesta del Sistema** |
| **1.** El caso de uso se inicia cuando se selecciona la opción *Gestión de Áreas* en el lateral derecho de Página Principal. (Ver Interfaz 1.1) | **2.** Se despliegan las diferentes opciones a gestionar. (Ver Interfaz 1.1) |
| **3.** Selecciona la opción *Lab. Computación*. | **4.** Muestra una interfaz *Laboratorio de Computación* (Ver Interfaz 1.2) que contiene:<br>- El listado de los laboratorios de computación evaluados hasta el momento.<br>- Opciones de “Mostrar”, “Editar” y “Eliminar” por cada uno de los laboratorios añadidos.<br>- Los laboratorios insertados.<br>- La evaluación (Bien, Regular o Mal) del *Estado Constructivo*, de la *Iluminación*, de la *Capacidad de Red*, de la *Disposición y espacio entre PC*, del *Mobiliario*, de las *Facilidades Eléctricas*, del *Control y Seguridad*, y por último, la *Ventilación y Climatización* del local.<br>- Botón *“*Adicionar*”.* |

| | |
|---|---|
| | - Botón *"*Reporte*".*<br>- Un campo para realizar búsqueda, acompañado del texto *Buscar*.<br>- Una lista desplegable para seleccionar la cantidad de elementos a mostrar en la página. Pueden mostrarse entre 10 y 100 elementos.<br>- Las opciones *Página Anterior* y *Página Siguiente.*<br>• Si el actor desea adicionar un nuevo laboratorio de computación, ir a la sección "Adicionar laboratorio de computación".<br>• Si el actor desea eliminar un laboratorio de computación ir a la sección "Eliminar laboratorio de computación".<br>• Si el actor desea modificar un laboratorio de computación, ir a la sección "Modificar laboratorio de computación".<br>• Si el actor desea mostrar los datos de un laboratorio de computación, ir a la sección "Mostrar datos de laboratorio de computación".<br>• Si el actor desea buscar un laboratorio de computación, ir a la sección "Buscar laboratorio de computación". |
| **Prototipo de interfaz** | |
| **Interfaz 1.1** | |




**Interfaz 1.2**


**Sección "Adicionar laboratorio de computación"**

| **Acciones del Actor** | **Respuesta del Sistema** |
|---|---|
| **1.** Da clic en la opción *Adicionar* (Ver interfaz 1.3) | **2.** Muestra la interfaz *Laboratorio de computación Adicionar Nuevo*: (Ver interfaz 1.4). Esta contiene los siguientes campos (todos los campos son listas desplegables): *Laboratorio*, *Estado Constructivo, Iluminación, Mobiliario, Facilidades Eléctricas, Control y* |

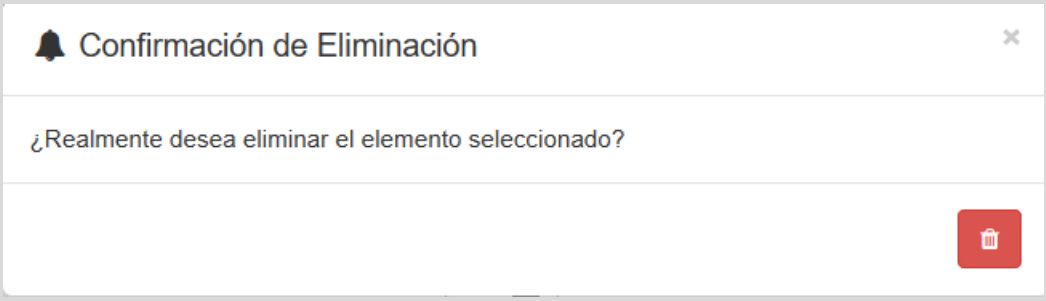
| | |
|---|---|
| | *Seguridad, Disposición Espacio de PC, Ventilación y Climatización*, y, por último, *Capacidad de Red*.<br>Botones: “Aceptar” y “Cancelar”. |
| **3.** Selecciona los datos que se solicitan, siendo todos los campos obligatorios. Luego da clic en el botón “Aceptar”. | **4.** El sistema muestra la interfaz *Laboratorio de computación Información* (Ver interfaz 1.9, que pertenece a la sección Mostrar laboratorio de computación) con los datos del último Laboratorio de Computación añadido. |
| **5.** Da clic en el botón *Listado* (Ver interfaz 1.9). | **6.** El caso de uso termina cuando se ha adicionado un nuevo Laboratorio de Computación y el sistema muestra la interfaz *Laboratorio de Computación Listado de existentes* actualizada. |
| **Flujo Alternativo** | |
| **3.1** No selecciona alguno de los datos y da clic en el botón “Aceptar”. | **3.2** Indica que el actor debe seleccionar un elemento de la lista para que se pueda ejecutar la acción. (Ver interfaz 1.5). |
| **3.1** Se selecciona en la lista *Laboratorio* algún laboratorio ya añadido y da clic en el botón “Aceptar”. | **3.2** Indica al actor que este valor ya se ha utilizado. (Ver interfaz 1.6). |
| **3.1** Selecciona los datos que se solicitan y da clic en el botón “Cancelar”. | **3.2** Cierra la interfaz *Adicionar Laboratorio de Computación* sin realizar cambios y muestra la interfaz *Laboratorio de Computación Listado de existentes.* |
| **Prototipo de interfaz**<br>**Interfaz 1.3** | |


**Interfaz 1.4**

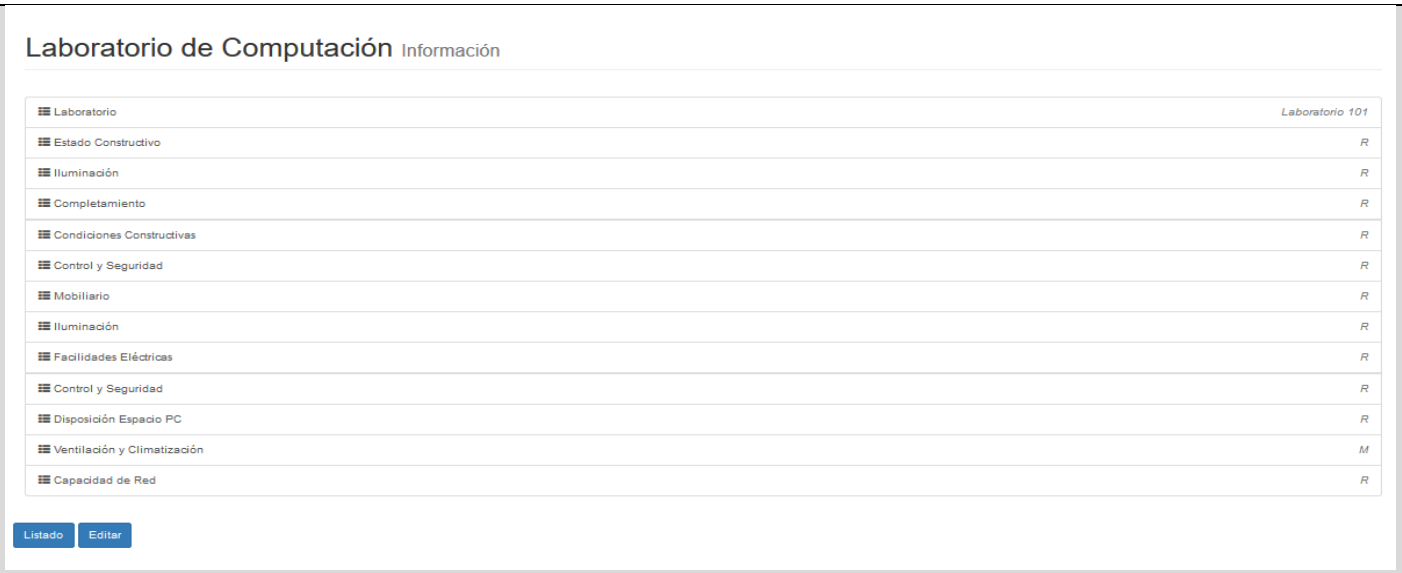

**Interfaz 1.5**


**Interfaz 1.6**


| **Sección "Eliminar laboratorio de computación"** | |
|---|---|
| **Acciones del Actor** | **Respuesta del Sistema** |
| **1.** Selecciona el laboratorio que desea eliminar del listado que se muestra en la interfaz *Laboratorio de Computación Listado de existentes* y da clic en la opción "Eliminar". | **2.** Muestra una ventana al actor donde le indica que debe confirmar si está seguro que desea eliminar el laboratorio seleccionado. (Ver interfaz 1.7) |
| **3.** Da clic en el ícono "Eliminar". | **4.** El caso de uso termina cuando elimina el laboratorio seleccionado por el actor y muestra la interfaz *Laboratorio de Computación Listado de existentes* actualizada. |
| | |

| Flujos Alternos | |
|---|---|
| **3.1** Da clic en la opción cerrar "*x*". | **3.2** Cierra la ventana sin realizar cambios y muestra la interfaz *Laboratorio de Computación Listado de existentes*. |
| **Prototipo de interfaz** | |
| **Interfaz 1.7**   | |
| **Sección "Modificar laboratorio de computación"** | |
| **Acciones del Actor** | **Respuesta del Sistema** |
| **1.** Selecciona el laboratorio al cual desea modificar sus datos a partir del listado que se muestra en la interfaz *Laboratorio de Computación Listado de existentes* y da clic en la opción "Editar". | **2.** Muestra la interfaz *Laboratorio de Computación Modificar datos* con la información del laboratorio seleccionado, (Ver interfaz 1.8). Contiene dos botones: "Aceptar" y "Cancelar". |
| **3.** Selecciona los nuevos datos y da clic en el botón "Aceptar". | **4.** El sistema muestra la interfaz *Laboratorio de computación Información* (Ver interfaz 1.9, que pertenece a la sección Mostrar laboratorio de computación) con los datos del último Laboratorio de Computación modificado. |
| **5.** Da clic en el botón *Listado* (Ver interfaz 1.9). | **6.** El caso de uso termina cuando actualiza los datos del laboratorio y muestra la interfaz *Laboratorio de Computación Listado de existentes* actualizada. (Ver interfaz 1.3) |

| Flujos Alternos | |
|---|---|
| **3.1** No modifica ningún dato y da clic en el botón “Aceptar”. | **3.2** Muestra la *interfaz Laboratorio de Computación Información*. (Ver interfaz 1.9) |
| **3.1** Introduce los nuevos datos y da clic en el botón “Cancelar”. | **3.2** No realiza cambios, se cierra la ventana y se muestra la *Información del Laboratorio de Computación Información*. (Ver interfaz 1.9) |
| **Prototipo de interfaz** | |
| **Interfaz 1.8** | |
|  | |
| **Sección “Mostrar laboratorio de computación”** | |
| **Acciones del Actor** | **Respuesta del Sistema** |
| **1.** Selecciona el laboratorio del cual desea visualizar sus datos a partir del listado que se muestra en la interfaz *Laboratorio de Computación Listado de existentes* y da clic en la opción “Mostrar”. | **2.** Muestra la interfaz *Laboratorio de Computación Información* que contiene los datos del laboratorio seleccionado. (Ver interfaz 1.9). |
| **Prototipo de interfaz**<br>**Interfaz 1.9** | |




**Sección "Buscar laboratorio de computación"**

| Acciones del Actor | Respuesta del Sistema |
|---|---|
| **1.** Introduce la información que desea buscar en el campo correspondiente a la búsqueda. | **2.** Muestra en la interfaz *Laboratorio de Computación Listado de existentes* el resultado de la búsqueda. (Ver interfaz 1.10). |

**Prototipo de interfaz**

**Interfaz 1.10**

## 2.3 Arquitectura de Software

Cuando se inicia la construcción de un sistema de software, luego de conocer cuáles son las necesidades básicas del usuario final, el ambiente donde será desarrollado el mismo, las condiciones para ello, las tecnologías y recursos necesarios para lograr un ambiente de trabajo adecuado, entonces se establecen los elementos fundamentales de la línea base para la arquitectura del sistema.

Según (IEEE Std 610.12, 1990) (Institute of Electrical and Electronics Engineers): '*La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución*".

La arquitectura brinda una visión global del sistema. Esto permite entenderlo, organizar su desarrollo, plantear la reutilización del software y hacerlo evolucionar. Se puede ver que la noción clave de la arquitectura es la organización y está relacionada con aspectos de rendimiento, usabilidad, reutilización, limitaciones económicas y tecnológicas. Esta se ocupa de componentes y no de procedimientos; de las interacciones entre esos componentes y no de las interfaces; de las restricciones a ejercer sobre los componentes y las interacciones y no de los algoritmos, los procedimientos y los tipos. (Almerira, y otros, 2007)

En el presente epígrafe se realiza una descripción de la arquitectura definida y los patrones de diseño que se utilizan. Se muestran los diagramas propuestos por la metodología en este flujo de trabajo, que permiten la compresión del diseño propuesto.

### *2.3.1 Patrón arquitectónico Modelo-Vista-Controlador*

Los patrones arquitectónicos definen la estructura general del software, indican las relaciones entre los subsistemas y los componentes del software y definen las reglas para especificar las relaciones entre los elementos (clases, paquetes, subsistemas) de la arquitectura (Pressman, 2008).

El principio más importante de la arquitectura Modelo-Vista-Controlador (MVC) es la separación del código en tres capas, dependiendo de su naturaleza. La lógica relacionada con los datos se incluye en el modelo, el código de la presentación en la vista y la lógica de la aplicación en el controlador. La programación se puede simplificar si se utilizan otros patrones de diseño (Potencier, et al., 2008).

**Modelo:** Es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en contexto del sistema proveen de información al usuario o a la aplicación misma. En

Symfony, el acceso y la modificación de los datos se realizan mediante objetos. Doctrine2 es el motor que se encarga de esta generación automática para construir sus clases, crea la estructura y genera el código de las mismas. Las clases y archivos relacionados con el modelo se guardan en el directorio src/*NombreBundle*/Entity/.

**Vista:** Es la representación del modelo en forma gráfica, disponible para la interacción con el usuario. En el caso de una aplicación Web, la "Vista" es una página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones. La vista en Symfony2 está formada por plantillas twig (es un motor de plantillas para el lenguaje de programación PHP) que se almacenan en el directorio src/*NombreBundle*/Resources /Views/.

**Controlador:** Es la capa encargada de manejar y responder las solicitudes del usuario, procesa la información necesaria y modifica el Modelo en caso de ser necesario. En Symfony2 todas las peticiones se realizan a través del controlador frontal app.php, el cual a través del enrutamiento delega las responsabilidades en las acciones que se implementan en las clases Controller de cada bundle.

### *2.3.2 Vista lógica*

Representa los elementos de diseño más importantes para la arquitectura del sistema. Describe las clases más significativas, su organización en paquetes y subsistemas, así como las realizaciones de casos de uso más transcendentales.

En la Figura 3 se representa la vista lógica de la aplicación a construir, teniendo en cuenta los principales elementos organizativos de Symfony y en la Figura 4 se representa el Funcionamiento de Symfony teniendo en cuenta el patrón arquitectónico MVC a través del caso de uso Gestionar Laboratorio de Computación.

**Modelo, Vista** y **Controlador:** representan paquetes lógicos en correspondencia con la arquitectura definida.

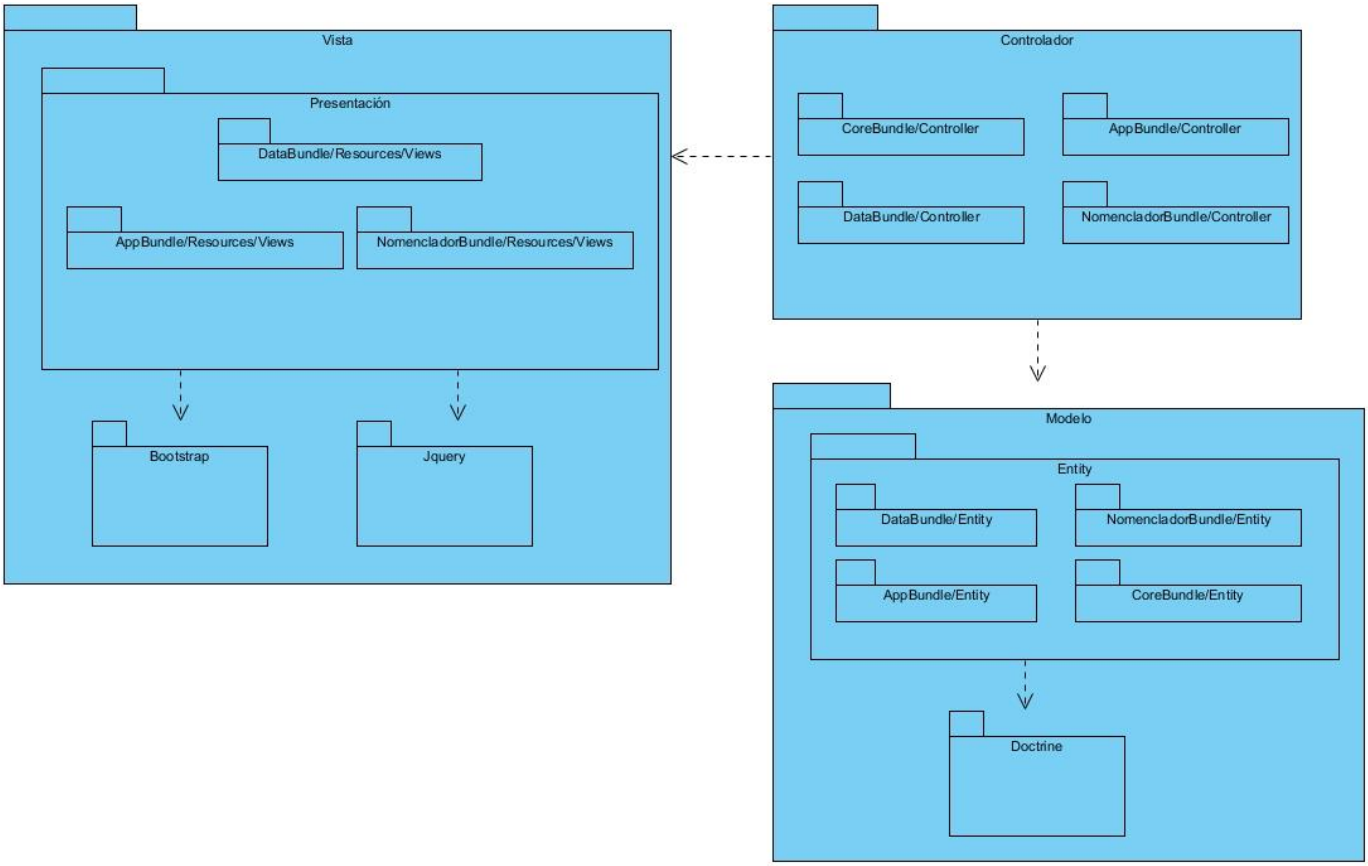



**Figura 3** Vista lógica de la aplicación

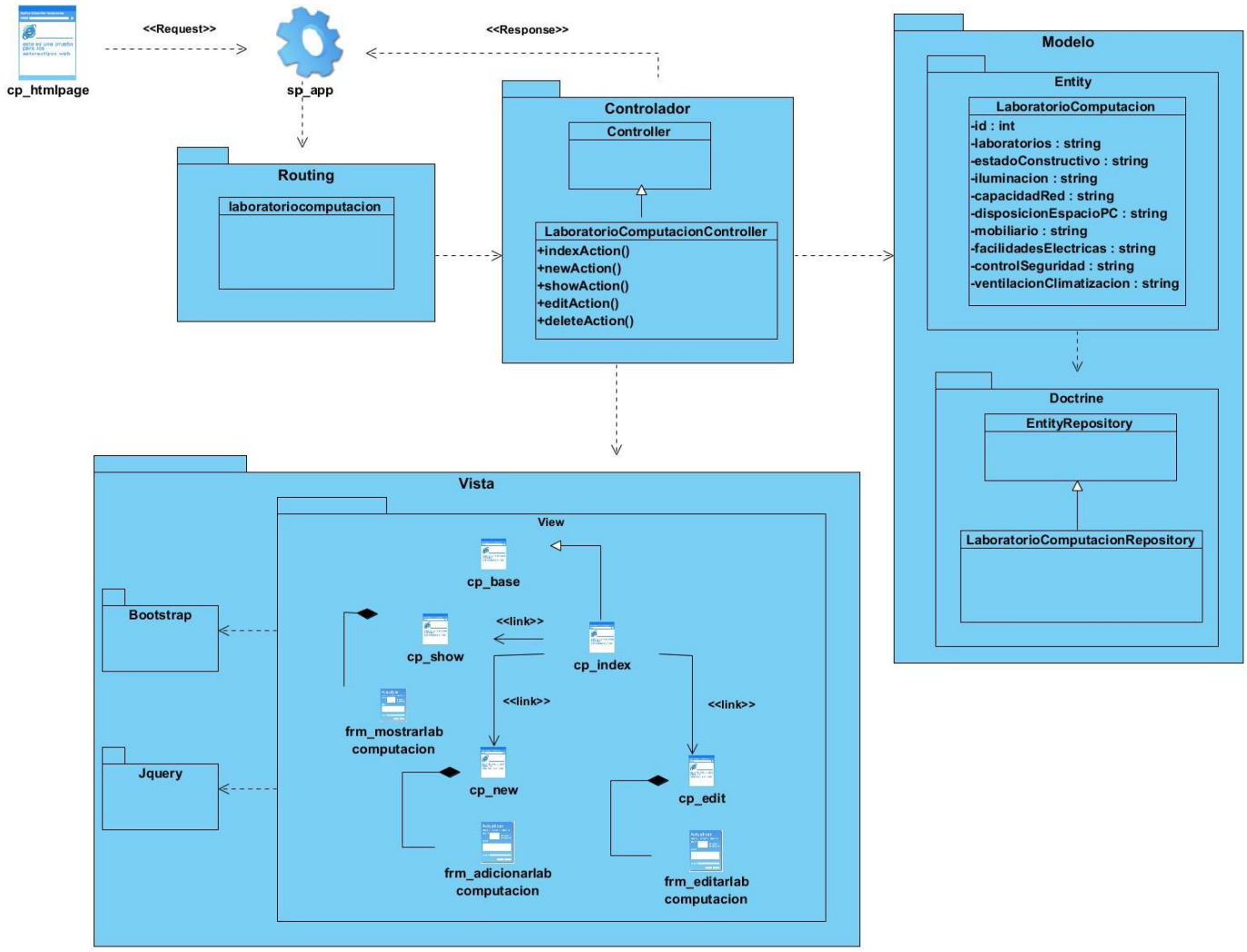



**Figura 4** Funcionamiento de Symfony-MVC. Caso de Uso Gestionar Laboratorio de Computación.

### *2.3.3 Diagrama* de clases del diseño

Los diagramas de clases del diseño (DCD) muestran el diseño estático a través de las clases, subsistemas y relaciones que participan. Las clases del diseño representan una abstracción de una o varias clases en la implementación del sistema. El lenguaje utilizado para especificar una clase del diseño es el mismo que el lenguaje de programación utilizado, los métodos tienen correspondencia directa con el debido método de la implementación de clases.

En la Figura 5 se muestra el Diagrama de Clases del Diseño del Caso de Uso Gestionar Laboratorio de Computación. Los restantes DCD se muestran en el Expediente de Proyecto.

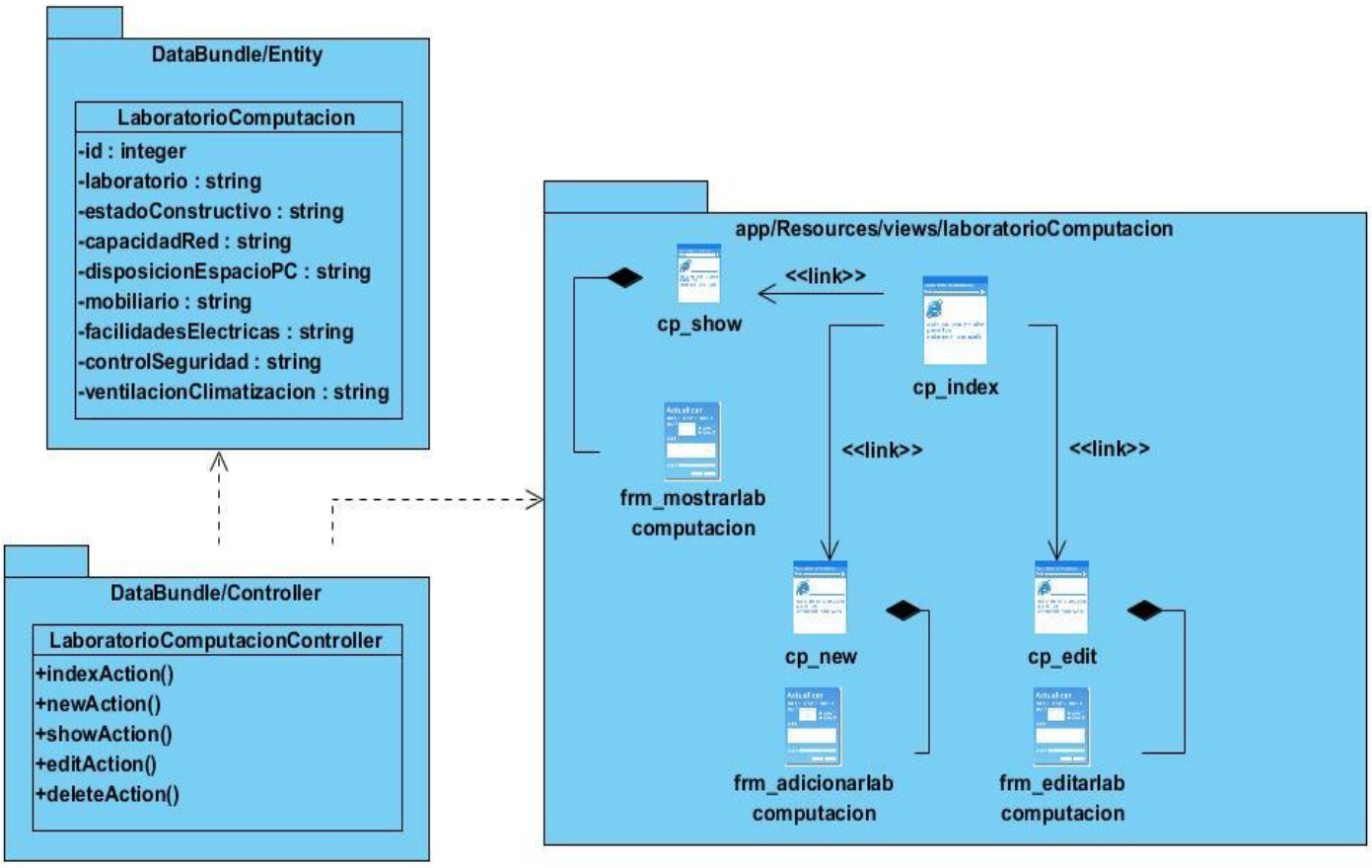



**Figura 5** Diagrama de Clases del Diseño del Caso de Uso Gestionar Laboratorio de Computación

### 2.3.4 *Patrones* del Diseño

Los patrones de diseño representan soluciones a problemas que surgen cuando se desarrolla un software en un contexto particular. Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular (Larman, 1999).

- **Patrones GRASP**

Los Patrones Generales de Software para Asignar Responsabilidades (GRASP por sus siglas en inglés), son parejas de problema y solución con un nombre, que codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades. Describen los principios

fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Los patrones GRASP utilizados en la investigación son:

**Alta Cohesión:** la cohesión es una medida de cuanto están relacionadas y enfocadas las responsabilidades de una clase. El uso del *framework* Symfony provee mediante una alta cohesión la asignación de responsabilidades y el empaquetamiento de archivos. Se evidencia en las clases controladoras debido a que Symfony implementa una clase controladora por cada entidad, lo que garantiza que se trabaje sobre una sola área de aplicación.

**Experto:** permite asignar una responsabilidad al experto en información, a la clase que cuenta con la información necesaria para cumplir la responsabilidad. Symfony2 incluye la librería ORM Doctrine como interfaz de comunicación con las clases del modelo, lo que permite encapsular toda la lógica de los datos y generar clases para manipular la información de las entidades de la base de datos. Este patrón se evidencia en las entidades que son las que conforman el modelo y se encuentran en la carpeta *Entity* de cada *Bundle*. Son expertas en la información que manejan, por ejemplo: la entidad *LaboratorioComputacion* tiene toda la información asociada a un laboratorio de computación.

**Bajo Acoplamiento:** el propósito de este patrón es que exista entre las clases la menor dependencia posible, para en caso de producirse una modificación en alguna, no afecte las restantes. Este patrón se evidencia durante la creación de objetos de las entidades desde los controladores, que heredan únicamente de la clase C*ontroller* que provee el *framework*. El propio patrón arquitectónico abstrae la vista y el controlador del modelo, aportando una baja dependencia entre las clases.

**Creador:** tiene en cuenta para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto solo pueda ser creada por el objeto que contiene la información necesaria. El uso de este patrón permite crear las dependencias mínimas necesarias entre las clases, lo cual favorece al mantenimiento del sistema. Se evidencia en las clases *Controller* porque contienen la información necesaria para crear instancias de una entidad.

**Controlador:** todas las peticiones desde la web son manipuladas por un solo controlador frontal, que es el punto de entrada único para la aplicación dentro del entorno correspondiente. Este patrón plantea que se debe asignar la responsabilidad a una clase del manejo de los eventos de un sistema. Sirve de intermediario entre una clase interfaz y la clase que contiene el algoritmo de la funcionalidad. El controlador es el encargado de procesar las diferentes peticiones hechas por el cliente. Este patrón se evidencia en las clases controladoras, pues son las encargadas de manejar las peticiones de los usuarios, obtener la información

que necesitan, para lo cual utilizan *Doctrine* para conectarse con la Base de Datos y enviar a la vista los datos para que sea observable por el usuario.

- **Patrones GoF**

Los patrones GoF (Gang of Four, en español: Pandilla de los Cuatro) se clasifican en tres (3) categorías basadas en su propósito: creacionales, estructurales y de comportamiento. En esta investigación se utiliza el patrón decorador.

**Decorador:** añade responsabilidades a un objeto de una forma dinámica y transparente, brinda una alternativa flexible a la herencia. En Symfony2 las vistas se encuentran en la carpeta *Resources/Views* de cada *bundle*. Se recomienda usar herencia en tres niveles donde en la base se tiene todo el código HTML que es dinámico para cada una de las páginas de la aplicación. En el segundo nivel se tienen los *layout frontend* y *backend* que contienen los elementos visuales que se repiten, como el menú lateral, el *footer*. En el tercer nivel se encuentran los elementos visuales que cambian según la información que se desea mostrar.

## 2.4 Modelo de datos

Las bases de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Es una colección de información organizada de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que necesite. Una base de datos correctamente diseñada permite obtener acceso a la información exacta y actualizada (Hernandez, 2010).

El modelo de datos constituye la representación de un fenómeno de la realidad objetiva a través de los objetos, sus propiedades y las relaciones que se establecen entre ellos. El mismo está compuesto por tres piezas fundamentales: el objeto de datos, los atributos y las relaciones entre las que se conectan. En la Figura 6 se representa un fragmento del modelo de datos, el modelo completo se puede consultar en el Anexo 4.

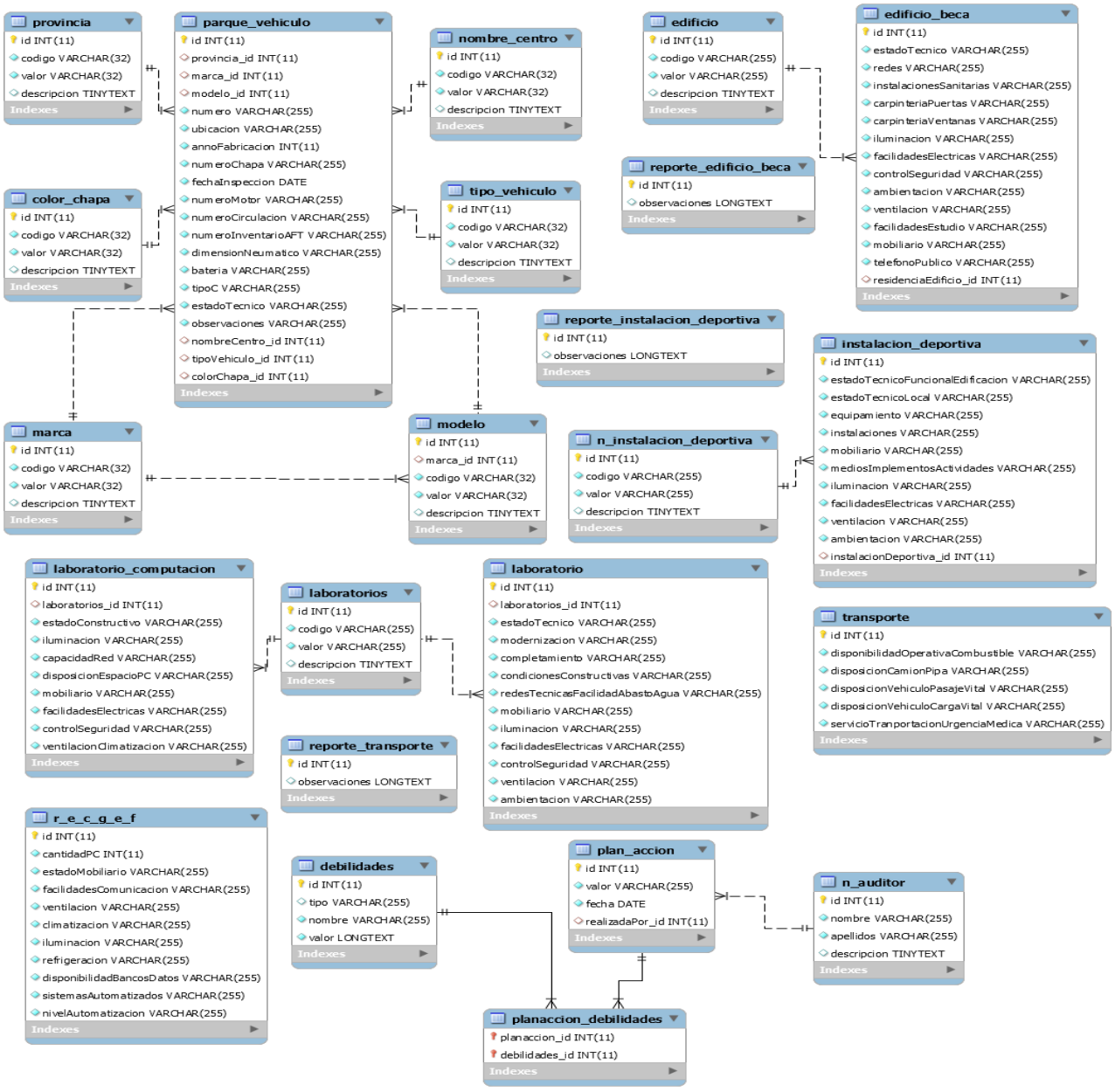



**Figura 6** Fragmento del modelo de datos del SEAIVI.

## 2.5 Modelo de despliegue

El modelo de despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos (Adaptive Ltd, 2007). En la Figura 7 se muestra el Diagrama de Despliegue del sistema.


**Figura 7** Diagrama de Despliegue

### Descripción de los Nodos

- <<executionEnvironment>> Servidor de Base de Datos: En este nodo se almacenan los datos del sistema.
- <<executionEnvironment>> Servidor de Aplicación Web SEAIVI: Este nodo es el encargado de tener instalado el sistema al que tendrán acceso los usuarios desde las estaciones de trabajo.
- <<executionEnvironment>> PC Cliente: Este nodo representa la estación de trabajo que permite al usuario mediante el protocolo HTTPS y el puerto 443 acceder a la aplicación.

## Conclusiones parciales

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones:

- Se determinaron las diferentes funcionalidades que el sistema debe cumplir mediante la especificación de requisitos. Se identificaron 208 requisitos funcionales que fueron agrupados en 64 casos de uso del sistema.
- Se realizó la descripción textual de cada caso de uso, lo cual permitió tener un mejor entendimiento de las funcionalidades que el sistema debe tener, el comportamiento del mismo y las secuencias de actividades que debe realizar el usuario y la respuesta del sistema para lograr su objetivo.
- La definición del patrón arquitectónico Modelo – Vista – Controlador y los patrones de diseño GRASP y GoF empleados, aseguró una adecuada estructuración del sistema, haciendo más fácil la integración de los componentes del mismo, su actualización, corrección y mejora. De igual forma garantizó una mayor simplicidad en el desarrollo, así como la escalabilidad.
- Con la realización de la vista lógica de la aplicación, así como de los diagramas de clases del diseño de cada uno de los casos de uso del sistema, se obtuvo una visión más exacta del sistema en términos de implementación, siendo de gran ayuda para el desarrollador.
- La realización del modelo de despliegue brinda una distribución completa del acople de los distintos componentes por lo que está compuesto el sistema.

# Capítulo 3. Implementación y Pruebas

Para (Jacobson, y otros, 1999) la implementación de software es el centro dentro de la fase de construcción. Los resultados fundamentales de este flujo, están dirigidos a la obtención de los subsistemas de implementación, sus dependencias, interfaces y contenidos, así como los componentes y las dependencias entre ellos. La implementación da lugar a la etapa de pruebas de software, la cual tiene como objetivos: planificar las pruebas necesarias en cada iteración, diseñar e implementar dichas pruebas a través de casos de pruebas y realizar las diferentes pruebas.

En el presente capítulo se muestra el diagrama de componentes, los estándares de codificación y el tratamiento que se le da a los diferentes errores del sistema, como parte del proceso de implementación. Por otro lado se plasma una descripción de los flujos de trabajo de implementación y pruebas realizadas al sistema. Se muestra un diagrama de componentes donde se describen los elementos físicos del sistema y sus relaciones. Finalmente se especifican las pruebas realizadas y los resultados alcanzados.

## 3.1 Diagrama de componentes

El diagrama de componentes es uno de los principales artefactos que se generar durante la implementación. Entre sus principales objetivos comprende: mostrar la dependencia lógica entre los distintos componentes del software y representar las relaciones entre los elementos que forman el código del sistema que se implementa. Estos esquemas describen unidades físicas del sistema y las relaciones existentes dentro del mismo. Cada unidad puede representar archivos simples, paquetes y librerías. (Microsoft, 2014). En la Figura 8 se muestra el Diagrama de Componentes del Caso de Uso Gestionar Laboratorio de Computación.

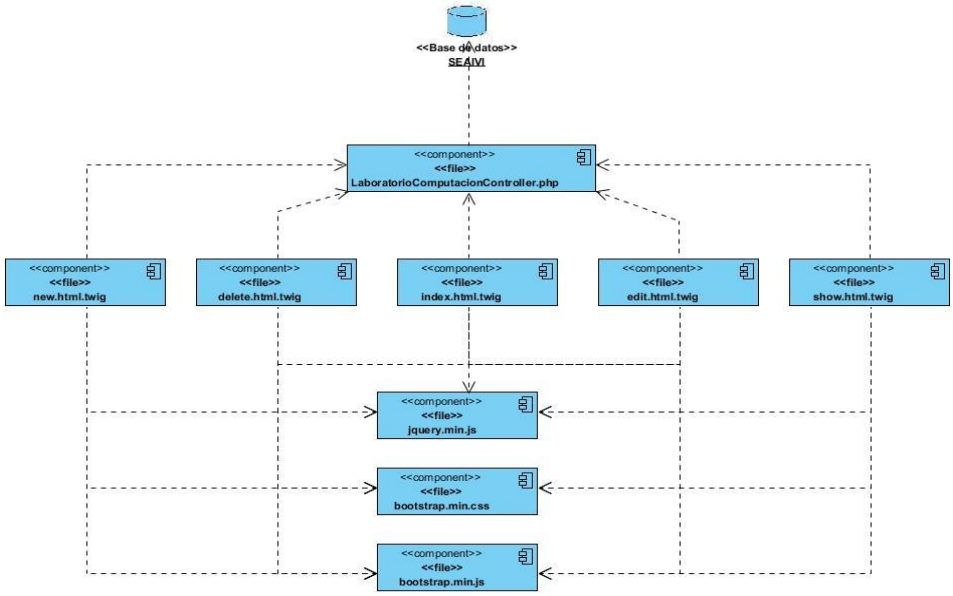



**Figura 8** Diagrama de Componentes del Caso de Uso Gestionar Laboratorio de Computación

## 3.2 Estándares de codificación

Un estándar de codificación comprende todos los aspectos de la generación de código. Un código fuente completo debe quedar como si un único programador hubiera escrito todo el código de una sola vez. Al inicio de un desarrollo de software, es necesario establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Para un programador comprender bien un sistema de software, influye directamente la legibilidad del código fuente (Larman, 1999).

En la solución propuesta se tienen en cuenta los siguientes estándares de codificación limitándose al uso de los estándares utilizados por Symfony2 (Figura 9), los cuales están, según (Potencier, et al., 2008), definidos en los documentos PSR-0, PSR-1, PSR-2.

1. Los archivos solo deben utilizar una codificación UTF-8.
2. Los nombres de las variables, métodos y clases deben expresar el propósito de dicho elemento.
3. Los nombres de clases deben ser escritas utilizando la técnica *StudlyCaps*.
4. Se escribe únicamente una declaración por línea.
5. El número de caracteres por línea deben ser de 80 columnas, aunque también está aceptado que sean hasta 120.

Se utilizó el estilo *lowerCamelCase* para la creación de los métodos, atributos y variables y el estilo *UpperCamelCase* para las clases.

- Declarar las propiedades de clase antes que los métodos.
- Utilizar mayúsculas intercaladas sin guiones bajos, en nombres de variable, función, método o argumentos.
- El nombre de las variables siempre comienza con el carácter especial „$", sin espacio y escrito en minúsculas. En caso de ser un nombre compuesto por más de una palabra, cada una debe escribirse en minúscula, sin espacio y sin guiones.
- Declarar los nombres de las tablas en la base de datos utilizando como separador el guion bajo, en caso de que el nombre sea compuesto.

LaboratorioComputacionController.php

```php
<?php
class LaboratorioComputacionController extends Controller
{
    /*
        Genera un documento excel con la información de la entidad
    */
    public function generarReporteAction(Request $request)
    {
        $em = $this->getDoctrine()->getManager();

        $defaultData = array('message' => 'Reporte de Laboratorio de Computación');
        $form = $this->createFormBuilder($defaultData)
            ->add('ces', TextType::class, array(
                'constraints' => array(
                    new NotBlank(),
                ),
                'attr'   => array(
                    'class' => 'form-control',
                )
            ))
            ->add('fecha',DateType::class,array(
                'format' => 'dd/MM/yyyy',
                'attr'   => array(
                    'class' => 'form-control',
                )
```

**Figura 9** Utilización de los estándares de codificación

## 3.3 Tratamiento de errores

### *3.3.1 Tratamiento de errores del lado del cliente*

La librería JQuery se utiliza para la generación de las vistas permite la validación mediante JavaScript de los recursos que se manipulan. De esta manera se pudieron garantizar las siguientes validaciones:

- Campos en blanco o formato incorrecto: Para cada campo obligatorio se utilizaron validaciones que le indican al usuario cuando el campo fue dejado vacío. Para los campos con formato específico se definieron máscaras para capturar solamente los formatos permitidos (fecha, email). (Ver Figura 10)


**Figura 10** Campos en blanco

- Listas y auto completamiento: La mayoría de las opciones se capturan mediante listas desplegables para evitar la introducción de datos erróneos. En algunos casos, el sistema permite introducir valores y la aplicación se encarga de filtrar la lista y valida siempre que los caracteres introducidos se encuentren dentro del conjunto de datos a seleccionar. (Ver Figura 11)

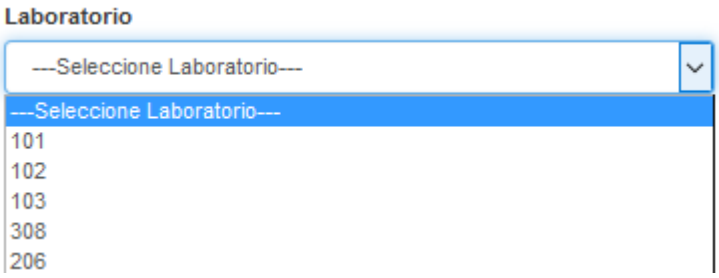



**Figura 11** Auto completamiento

### *3.3.2 Tratamiento de errores del lado del servidor*

En las clases controladoras se realiza para toda acción las validaciones siguientes:

- Peticiones seguras ejecutadas por el método *Post*: Todas las peticiones que salvan información en la base de datos se ejecutan por el método *Post*, lo que evita el acceso indebido a funciones desde la barra de direcciones del navegador.
- Comprobación de la integridad de datos: Para las acciones de inserción y actualización se valida que no se entren valores iguales a otros registrados en la base de datos, que existan los identificadores y que se procesen las operaciones mediante transiciones.
- Captura de errores de respuesta: Los errores generados por el modelo son enmascarados para transmitir al usuario mensajes que le permitan identificar la causa del error.

- Almacenamiento en secciones de valores de usuario: Los valores fijos de los usuarios conectados se almacenan en variables de sesión, teniéndose control de cada ejecución del usuario y validándose que se tengan los permisos para la acción que se solicita.

## 3.4 Pruebas

Las pruebas de software se han convertido en un factor determinante para lograr el éxito de sistemas, de ahí su importancia durante el ciclo de vida del software. Están constituidas por una serie de actividades que se realizan con el propósito de encontrar los posibles fallos de implementación, calidad o usabilidad. Dichas actividades podrán ser implementadas en cualquier momento del proceso de desarrollo, pero deben ser seleccionadas y utilizadas de la manera más eficiente según el contexto del proyecto. Se decide aplicar dos pruebas fundamentales al SEAIVI: pruebas de funcionalidad, rendimiento y fiabilidad.

**Pruebas de función**

Se realizaron pruebas de función, en la dimensión de calidad a nivel de sistema, a través del método de caja negra, usando la técnica de partición de equivalencia. La prueba de caja negra se refiere a las pruebas que se llevan a cabo en la interfaz del software. Esta examina algunos aspectos fundamentales de un sistema con poca preocupación por la estructura lógica interna del software. (Pressman, 2008)

Según lo definido por (Pressman, 2008), con las pruebas de caja negra se trata de demostrar que las funcionalidades del software sean operativas, que las entradas se manejen de forma correcta y que se produzca el resultado esperado.

También conocidas como pruebas de comportamiento, se centran principalmente en los requisitos funcionales detectados con anterioridad. Se especializa en la comprobación de interfaces y cuenta con el apoyo de casos de prueba de cada funcionalidad. Su objetivo principal es detectar errores en algunas de las cinco categorías siguientes:

1. Errores de funciones incorrectas o ausentes.
2. Errores de interfaz.
3. Errores en estructura de datos o en accesos de datos externas.
4. Errores de rendimiento.
5. Errores de inicialización y de terminación.

La técnica de partición equivalencia divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. Para ello, se realizó un caso de prueba por cada caso de uso del sistema. En las Tablas 5, 6, 7, 8, 9 ,10 y 11 que se encuentran en el Anexo 3, se puede observar el caso de prueba correspondiente al caso de uso Gestionar Laboratorio de Computación. El resto de los casos de pruebas se pueden consultar en el Expediente de Proyecto.

**Resultados de las pruebas de función realizadas**

Durante el transcurso de las pruebas al sistema se detectaron un conjunto de No Conformidades (NC), clasificadas según su importancia en significativas y no significativas. Entiéndase por significativa aquellas NC que puedan afectar el funcionamiento del sistema. Mientras, que las no significativas se enfocan en el diseño u otro aspecto que no afecte el funcionamiento de la propuesta de solución como validaciones y errores de ortografía. A continuación, se muestra el resumen de las NC detectadas.

Como se muestra en la Figura 12, en la primera iteración realizada se identificaron un total de 29 NC, de ellas, 4 significativas y 25 no significativas. Luego de realizada las correcciones a las NC detectadas se procedió a una segunda iteración. Obteniéndose como resultado un total de 18 NC, 3 de ellas clasificadas como significativas y 15 como no significativas. Finalmente, para una tercera iteración no fueron identificadas ninguna de ellas, lo que confirmó el buen funcionamiento del sistema.

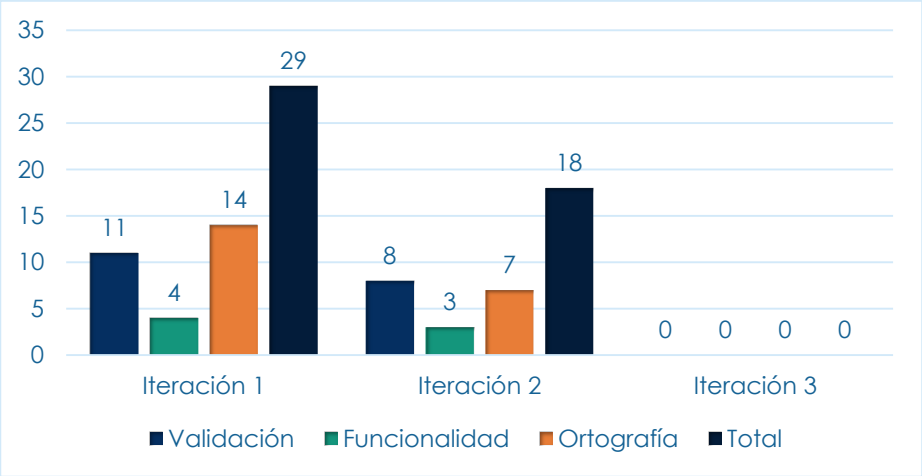



**Figura 12** Gráfica de No Conformidades

**Pruebas de carga y estrés**

Las pruebas de carga son diseñadas para determinar y validar la respuesta de la aplicación cuando es sometida a una carga de usuarios y/o transacciones que se espera en el ambiente de producción. Entre las dimensiones de calidad de esta prueba se encuentra la de rendimiento. Estas se esbozan para asegurar que el sistema pueda procesar una carga esperada. Esto normalmente implica planificar un grupo de pruebas en la que la carga se va incrementado regularmente hasta que el rendimiento del sistema se hace inaceptable. Se ocupan tanto de demostrar que el sistema satisface los requerimientos como de descubrir defectos y problemas en el sistema (Sommerville, 2005).

Por su parte, las pruebas de estrés son creadas para encontrar el volumen de datos o de tiempo en que la aplicación comienza a fallar o es incapaz de responder a las diferentes peticiones. La fiabilidad, es una de las dimensiones de calidad de este tipo de pruebas. Para (Zibert van Gricken, y otros, 2005), la comprobación de la confiabilidad o fiabilidad consiste en probar una aplicación para descubrir y eliminar errores antes de que se despliegue el sistema.

Para este tipo de pruebas se utilizó la herramienta JMeter y usó una computadora de oficina con un procesador Intel Core i3-2120 con una frecuencia de 3.3 GHz y una memoria RAM DDR3 de 2GB.

. A continuación, se muestran las variables analizadas:

- Usuarios: Cantidad de usuarios.
- Muestra: Cantidad de peticiones realizadas para cada URL.
- Media: Tiempo promedio en milisegundos en el que se obtienen los resultados.
- Mediana: Tiempo en milisegundos en el que se obtuvo el resultado que ocupa la posición central.
- Línea 90 %: Máximo tiempo utilizado por el 90 % de la muestra.
- Min: Tiempo mínimo que demora un hilo en acceder a una página.
- Max: Tiempo máximo que demora un hilo en acceder a una página.
- %Error: Porciento de error de las páginas que no se llegaron a cargar de manera satisfactoria.
- Rend: Rendimiento. Se mide en cantidad de solicitudes por segundo.
- Kb/Seg: El rendimiento se mide en cantidad de kilobytes por segundo.

**Resultados de las pruebas de carga y estrés realizadas**

**Tabla 4** Pruebas de carga y estrés realizadas al SEAIVI

| Usuarios | Muestra | Media | Mediana | Línea 90 % | Min | Max | %Error | Rend. | Kb/s |
|---|---|---|---|---|---|---|---|---|---|
| 200 | 1900 | 1130 | 1320 | 1276 | 2 | 1288 | 0 | 83.4 | 740.7 |
| 400 | 40900 | 2243 | 2609 | 2046 | 3 | 423547 | 0.17 | 74.2 | 662.1 |
| 800 | 52200 | 2950 | 5125 | 7946 | 4 | 500956 | 0.32 | 66.1 | 548.9 |

Las pruebas de carga realizadas muestran que el sistema es capaz de responder a 1900 peticiones de 200 usuarios conectados simultáneamente en un tiempo promedio de 1130 milisegundos (1.13 segundos aproximadamente) con 0 % de error.

Se realizaron 40900 peticiones iniciadas por 400 usuarios y en este caso el sistema respondió en 2243 milisegundos (2.24 segundos aproximadamente) como tiempo promedio. No fue capaz de responder correctamente el 0.17 % de las peticiones realizadas.

Por último, se realizaron pruebas de estrés, con 51200 peticiones iniciadas por 800 usuarios y en este caso el sistema respondió en 2950 milisegundos (2.95 segundos aproximadamente) como tiempo promedio. No fue capaz de responder correctamente el 0.32 % de las peticiones realizadas.

**Valoración de la contribución lograda**

A partir de la caracterización del proceso de evaluación y acreditación de instituciones de Educación Superior realizado; la autora de la presente investigación concluye que: la utilización del Sistema para la evaluación y acreditación institucional de la variable infraestructura y gestión de los recursos, permite valorar como positiva la contribución a los grados centralidad, disponibilidad, e integridad de la información que se gestiona a través del sistema mencionado con anterioridad.

Se arriba a dicha conclusión, teniendo en cuenta que:

- El resultado de las pruebas de función realizadas, evidencia los grados de completitud, corrección y precisión de los datos, lo cual representa una contribución al grado de integridad de la información en el SEAIVI.

- Con las pruebas de rendimiento aplicadas se evidenció la alta capacidad de respuesta de la aplicación ante las diversas peticiones de acceso a la información de manera simultánea. Teniendo en cuenta que la disponibilidad de la información depende de la capacidad de respuesta de su propia fuente, se infiere un alto grado de disponibilidad a la información.
- También se garantizó a través de la utilización del *framework* Symfony con el ORM Doctrine la seguridad de la aplicación. Esto es posible a partir de la gestión de usuarios implementada, la definición de roles y permisos y la autenticación en el sistema. Doctrine, que es el ORM que utiliza Symfony para la conexión con la base de datos, se encarga de prevenir las inyecciones SQL. De igual forma se definen las rutas a las que tendrá acceso cada usuario. Si existe una persona que desea quebrantar la seguridad, introduciendo directamente en el navegador una ruta a la cual no debe acceder, el sistema muestra de manera automática un formulario para que el usuario se autentique.

La aplicación desarrollada cumple con los principios anteriormente mencionados, directamente asociados a las variables centralidad, disponibilidad e integridad, las cuales estaban limitadas sustancialmente.

## Conclusiones parciales

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones:

- Con la elaboración del diagrama de componentes como uno de los principales artefactos generados durante la implementación, se pudo mostrar la dependencia lógica entre los distintos componentes del software y representar las relaciones entre los elementos que forman el código del sistema implementado.
- Los estándares de codificación y los estilos de programación utilizados fueron descritos para hacer más fácil el entendimiento del código por el desarrollador, así como un mejor mantenimiento a la aplicación en el futuro.
- Se detalló el tratamiento de errores de la aplicación donde se evidencia la capacidad del sistema de responder ante situaciones no esperadas.
- Se verificó a través de las pruebas realizadas que los requisitos funcionales del sistema y las funciones de este son operativas, cumpliendo de esta manera con los objetivos trazados.

## Conclusiones

- Se contribuyó en el desarrollo del proceso de gestión de información para la evaluación y acreditación de la variable infraestructura y gestión de los recursos, erradicando posibles errores humanos que puedan presentarse en el procesamiento de la información requerida.
- Se construyó un sistema con un enfoque de aplicación web siguiendo la metodología OpenUP, utilizando el *framework* Symfony en su versión 2.8.17 y el SGBD PostgreSQL 9.5.0.
- A partir de la aplicación de técnicas de obtención de información se definieron los requisitos funcionales y no funcionales del sistema. El lenguaje UML permitió modelar el sistema a desarrollar, así como la información solicitada por el cliente para facilitar el futuro mantenimiento y evolución de la aplicación.
- Para comprobar la calidad y correcto funcionamiento del sistema, se diseñaron y ejecutaron casos de prueba, los que arrojaron resultados satisfactorios, lo cual demuestra el cumplimiento de los requisitos funcionales establecidos en la fase inicial del proceso de desarrollo del software.
- Se cumplió el objetivo general de la investigación al desarrollar el sistema para la evaluación y acreditación de la variable infraestructura y gestión de los recursos, como resultado de la investigación realizada. Este garantizará la generación oportuna de reportes que facilitan el trabajo que se realiza a diario en la universidad.

## Recomendaciones

Con el fin de lograr un sistema de gestión de información con mayor alcance, la autora recomienda:

- Informatizar el resto de las variables asociadas al Sistema de Evaluación y Acreditación de Instituciones de Educación Superior e incluirlos como módulos en el sistema desarrollado.
- Informatizar los criterios de evaluación que no se encuentran contemplados en el Excel que establece la Junta de Acreditación Nacional y son objeto de análisis para este propósito.

## Bibliografía

**Abrahamsson, Pekka, y otros. 2002.** *Agile software development methods- Review and analysis.* Universidad de Oulu. Finlandia : ESPOO, 2002.

**Águila, V. 2002.** *El concepto calidad en la educación universitaria: clave para el logro de la competitividad institucional.* La Habana : Revista Iberoamericana de Educación, 2002. 1681-5653.

**Almerira, Sandra y Pérez, Vanina Cavenago. 2007.** *Arquitectura de Software: Estilos y Patrones.* 2007.

**Apache Software Foundation. 2000.** *Apache Tomcat 6.0 User Guide.* [En línea] 2000. [Citado el: 23 de 9 de 2016.]

**Bakken, S y AULBACH, A. 2005.** *Manual de PHP.* 2005.

**Bartle. 2011.** *Información para la gestión.* 2011.

**Boehm, Barry y Turner, Richard . 2002.** *Balancing Agility and Discipline.* Boston : Addison-Wesley, 2002.

**Brooks, Frederick Phillips. 2011.** Prácticas de Software. *Prácticas de Software.* [En línea] 2011. [Citado el: 11 de febrero de 2013.] www.practicasdesoftware.com.ar.

**Calidad, Política. 2015.** *Política de Calidad UCI.* La Habana : s.n., 2015.

**Capote, B y Otros. 2003.** *La gestión de información como herramienta fundamental en el desarrollo de los centros toxicológicos.* La Habana : s.n., 2003.

**Castillo, Alejandro Cantón. 2011.** *Manual de HTML5 en español.* 2011.

**Castro Ruz, Fidel. 1987.** Discurso pronunciado en la clausura del XI Seminario Nacional de Educación Media, Teatro “Karl Marx”. 1987.

**Colectivo. 2012.** *Definición.de.* [En línea] 2012. [Citado el: 03 de 12 de 2016.] http://definicion.de/centralizacion/.

**DRAE.** DRAE. *Diccionario de la Real Academia Española.* [En línea] [Citado el: 15 de Octubre de 2016.] http://www.rae.es/.

**Duggan, Evan y Reichgelt, Han. 2012.** *Measuring Information Systems Delivery Quality.* Hershey : Idea Group Inc, 2012. ISBN 1-59140-859-8.

**Dunglas, Kévin. 2013.** *Persistence in PHP with the Doctrine ORM.* s.l. : Packt Publishing Ltd, 2013.

**Egulliz, Javier. 2013.** *Desarrollo web ágil con Symfony 2.3.* 2013.

**—. 2007.** *Introducción a CSS.* 2007.

**—. 2009.** *Introducción a JavaScript.* 2009.

# Bibliografía


**Escribano Hervis, Elmys y García Naranjo, Marcos Antonio. 2013.** *La evaluación institucional: Un proceso de gestión de la calidad en la Universidad de Ciencias Pedagógicas "Juan Marinello Vidaurreta" de Matanzas.* Matanzas : s.n., 2013.

**Fowler, Martin. 2007.** *UML destilled, A brief guide to standard object modeling languaje.* Tercera : Pearson Education, 2007. ISBN: 978-8131-715-65-9.

**Fowler, Martin, Rice, David y Foemmel, Matthew. 2002.** *Patterns of Enterprise Application Arquitecture.* s.l. : Addison Wesley, 2002. ISBN : 0-321-12742-0 .

**Gadja, Wlodzimierz. 2007.** *Instant PhpStorm Starter.* Birmingham : Packt Publishing Ltd, 2007.

**Giorgetti, Carlos , Romero, Lucila y Vera, Marcela . 2014.** *Estudio de los modelos de evaluación de la calidad existentes para la conceptualización de un modelo adecuado para Instituciones de Educación Superior que implementan Educación a Distancia en Argentina.* Argentina : Congreso Iberoamericano de Ciencia, Tecnología, Innovación y Educación, 2014. ISBN: 978-84-7666-210-6 – Artículo 1466.

**Gonzalez Castilla, Ilianet. 2010.** *La biblia de Ingeniería.* La Habana : s.n., 2010. Vol. VIII.

**Hernandez, Aleida. 2010.** Institución Tecnológico de Colima. *Departamento de Sistemas y Computación.Tutorial de Fundamentos y Base de Datos.* [En línea] 2010. [Citado el: 22 de Abril de 2014.] http://labredes.itcolima.edu.mx/fundamentosbd/sd_u2_1.htm.

**IEEE Std 610.12. 1990.** *IEEE Standard Glossary of Software Engineering Terminology.* 1990.

**Jacobson, Ivar, Booch, G. y Rumbaugh, J. 2000.** *El proceso unificado de desarrollo de software.* Madrid : Pearson Educación, 1999. ISBN: 84-7829-036-2.

**JAN, Junta Acreditación Nacional. 2014.** *Sistema de Evaluación y Acreditación de Instituciones de Educación Superior (SEA-IES).* 2014.

**Larman, Craig. 1999.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : PRENTICE HALL, 1999. ISBN 0-13-748880-7.

**Lazo, J. 2000.** *Evaluación Académica.* s.l. : Universidad Mayor Real y Pontificia San Francisco Xavier de Chuquisaca, 2000.

**McGraw-Hill. 2001.** *Diccionario de informática e internet de Microsoft.* España : s.n., 2001.

**Meyer, B. 2009.** *Construcción de Sofware Orientado a Objetos.* La Habana : Félix Varela, 2009.

**Microsoft. 2014.** [En línea] 2014. [Citado el: 22 de Marzo de 2015.]

**Pérez Pino, María Teresa y Ciudad Ricardo, Febe Ángel. 2015.** *Indicadores para la evaluación de la calidad de la formación del ingeniero en Ciencias Informáticas.* La Habana : Revista Cubana de Ciencias Informáticas, 2015. 2227-1899.

**Pérez-Montoro , Mario y Golkhosravi, Mehrad. 2010.** *Gestión de la información.* León : Universidad de León, 2010.

**PostgreSQL. 2015.** *PostgreSQL documentation.* 2015.

# Bibliografía


**Potencier, Fabier y François, Zaninotto. 2008.** *Symfony, la guía definitiva.* 2008.

**Pressman, Roger S. 2008.** *Ingeniería de Software: un enfoque práctico.* s.l. : Mc Graw Hill, 2008. ISBN: 970-10-5473-3.

**Rubio Oca, Julio. 2007.** *La evaluación y acreditación de la educación superior en México: un largo camino aún por recorrer.* [Documento PDF] Mexico : Universidad Autónoma Metropolitana Unidad Xochimilco, 2007. ISSN: 0188-168X.

**Sommerville, I. 2005.** *Ingeniería del Software.* 2005.

**Soto, María Aurora Balbón y Barrios, Norma M. Fernández. 2006.** [En línea] 2006. http://bvs.sld.cu/revistas/aci/vol14_2_06/aci04206.htm#cargo.

**Tinoco Gómez, Oscar, Rosales Lopez, Pedro Pablo y Salas Bacalla, Julio. 2010.** *Criterios de selección de metodologías de desarrollo de software.* 2010.

**Urrutia Arriba, Amaia. 2011.** Universidad del País Vasco. *¿Centralizar o descentralizar los sistemas de información en la empresa?* [En línea] 2011. [Citado el: 03 de 12 de 2016.] http://grupo.us.es/grehcco/ambitos03-04/03urrutia.pdf.

**Vignaga, Andrés y Perovich, Daniel . 2014.** *Enfoque Metodológico para el Desarrollo Basado en Componentes.* 2014.

**Visual Paradigm Group. 2011.** *Reasons to Choose Visual Paradigm.* 2011.

**—. 2013.** *Visual Paradigm for UML.* 2013.

**W3C. 2016.** W3C. [En línea] 4 de octubre de 2016. https://www.w3.org/TR/css-2015/.

**Wesley, Addison. 2004.** *Software development for small team: A RUP - centric approach.* s.l. : Addison Wesley object technology series, 2004. ISBN: 978-0321-199-50-8.

**Woodman, L. 1985.** *Information management in large organizations.* London : ASLIB, 1985.

**Zibert van Gricken, Carolina and Boucchechter, Israel. 2005.** Pruebas de Confiabilidad. *carolina.terna.net.* [Online] Mayo 30, 2005. [Cited: Marzo 20, 2017.] http://carolina.terna.net/ingsw3/datos/Pruebas_de_Confiabilidad.pdf.