

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 1



Sistema de Reportes y apoyo a la toma de decisiones para la
Dirección de Alimentación de la Universidad de las Ciencias
Informáticas

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

Autor(es):

Geidy González Espinosa
Yoandi Alfonso Posada

Tutor(es):

Ing. Luis Ernesto Acosta
MSc. Osiris Perez Moya

La Habana, Cuba
Abril 2013

| DECLARACIÓN DE AUTORÍA

Declaración de autoría

Declaramos ser autores del presente trabajo de diploma y autorizamos al Centro de Informatización Universitaria (CENIA) de la Universidad de las Ciencias Informáticas (UCI) los derechos patrimoniales de la misma.

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

Geidy González Espinosa

Firma del Autor

Ing. Luis Ernesto Acosta

Firma del Tutor

Yoandi Alfonso Posada

Firma del Autor

MSc. Osiris Perez Moya

Firma del Tutor

Datos de Contacto

Tutor: Ing. Luis Ernesto Acosta.

- Ingeniero en Ciencias Informáticas.

Correo electrónico: leortiz@uci.cu

Tutor: MsC. Osiris Perez Moya.

- Máster en Gestión de Proyectos Informáticos.

Dedicatoria

A nuestros queridos padres, por todo su amor, apoyo y aliento. Para darles un motivo más por el cual sentirse orgullosos de sus hijos. Que en sus recuerdos quede este momento, en el cual se hacen realidad también, sus sueños.

Agradecimientos

Yoandi Alfonso Posada

Ante todo quiero agradecer a Dios por permitir que se cumpla todo lo que he querido. Agradezco a mis padres por apoyarme a lo largo de estos 5 años de universidad, por brindarme su amor y confianza, por quererme tanto y complacerme en todos mis caprichos.

A Osiel por brindarme su ayuda incondicionalmente...él sabe.

A Geidy mi compañera de tesis por haber trabajado conjuntamente para lograr realizar la tesis y por los momentos de risas que pasamos en el lab.

A mis tutores por su colaboración.

A mis amigos de todos estos años que han estado conmigo en las buenas y en las malas, compartiendo momentos, espero que no olviden todas estas vivencias: Leandro, Van brakle, Macdemis, Leydis, Evelyn.

A Nathaly por pasar noches desvelada junto a mi...

A todas mis amistades que de una forma u otra hicieron que mi estancia en la universidad fuese para mi inolvidable Sarahí, Lisandra, Anabel, Osiel, El Dimi(Ramniel), Maria Isabel, Marbelis, Islen, Jose Antonio (jose), Alejandro, Yordanka, Mayabeque(yasmani).

A todos los profesores que durante estos años me formaron y me ayudaron.

A mi hermano Yuniel por apoyarme y siempre preocuparse.

A mis tíos, mi tío Mandy por decirme siempre que yo sería el primer ingeniero de la familia, a mi tía Magda por siempre brindarme amor y ser mi segunda mamá, a mi tía Nery, Zoraida y mi tía Onery por siempre estar preocupadas por mis asuntos escolares y apoyarme.

A mis primos, especialmente a Liany y Rachel por ser unas primas incondicionales y a todos mis primos por estar siempre a mi lado.

A mis abuelas que tanto las quiero por darme ese amor que tanto necesito siempre.

A toda mi familia por ser tan unida y darme esa fuerza para seguir adelante cada día.

A todas las personas que pasaron a formar parte de mi vida en estos años.

A todos los que contribuyeron en la realización este trabajo de diploma.

Geidy González Espinosa

Quiero agradecerles de manera muy especial a mis lindos gorditos, mi mamá y mi papá, a mis abuelitos Melba y Gonzalo y a mi tío Alexander, a estas personas le agradezco lo que soy hoy, gracias por sus consejos, gracias por el sacrificio, el amor, la dedicación, por estar siempre conmigo en los buenos y malos momentos, gracias por enseñarme que nunca debemos rendirnos que todo es posible mientras se tenga ganas de triunfar y vivir, son la mejor familia del mundo y ni siquiera con palabras puedo agradecerles todo lo que han hecho por mí.

Le agradezco a Osiel por todas la noches de desvelo. Por toda su ayuda incondicional por ser mi amigo, como bien él mismo dice "mi tutor", por darme siempre fuerzas para seguir adelante, jamás olvidaré todo lo que has hecho por mí.

A Reisel por ser el mejor amigo del mundo, gracias por ser tan especial e incondicional conmigo, siempre te recordaré como la mejor persona que he conocido.

A mis compañeras de apartamento en especial a Rosalina por todos los momentos divertidos que pasamos juntas, por su compañía y amor fraternal, por todos sus consejos que aunque no los siga, los valoro, es otra de las personas que nunca olvidaré.

A mis tutores por su colaboración.

Resumen

La generación de reportes es una tarea fundamental en los sistemas que gestionan información, todo sistema que trabaje con gran volumen de datos necesita obtener reportes para poder llegar a conclusiones o a estimaciones.

La Universidad de las Ciencias Informáticas cuenta con un sistema de reportes que se encarga de mostrar toda la información relacionada con los servicios de alimentación, este actualmente no posee todas las funcionalidades necesarias para satisfacer las necesidades de los distintos niveles de la Dirección de Alimentación, es por esta razón que el objetivo fundamental de esta investigación se basa en desarrollar un sistema de reportes, con tecnologías que contribuyan al desarrollo de la soberanía tecnológica, para mostrar la información relacionada con los procesos que gestiona la Dirección de Alimentación de la Universidad de las Ciencias Informáticas y apoye el proceso de toma de decisiones.

En la investigación se realiza el estudio de los lenguajes, herramientas y tecnologías necesarias para el desarrollo de la solución definidas por el Centro de Informatización Universitaria (CENIA). Se definen los principales conceptos, requisitos funcionales y no funcionales identificados a partir de las necesidades del cliente. Además, se ejecutan las pruebas para el aseguramiento de la calidad del producto. Concluida la investigación se obtiene como resultado final un sistema que brinda información en forma de reportes y que le permite a la Dirección de Alimentación realizar la planificación de los alimentos para decisiones futuras.

Palabras clave: reporte, alimentación, generación de reportes.

Índice

INTRODUCCIÓN	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA	5
1.1 INTRODUCCIÓN	5
1.2 CONCEPTOS ASOCIADOS AL OBJETO DE ESTUDIO	5
1.4 RESUMEN DEL ANÁLISIS DE LOS SISTEMAS HOMÓLOGOS ESTUDIADOS	7
1.5 LENGUAJES, TECNOLOGÍAS, HERRAMIENTAS Y PROCESO DE DESARROLLO DE SOFTWARE	8
1.5.1 Proceso de desarrollo con enfoque ágil basado en el nivel 2 de CMMI	8
1.5.2 Lenguaje de programación	11
1.5.3 Lenguaje de modelado UML.....	12
1.5.4 Entorno de desarrollo integrado NetBeans IDE	13
1.5.5 Herramienta para el diseño de la aplicación	13
1.5.8 Servidor web.....	16
1.5.9 Sistema de gestión de base de datos (SGBD)	17
1.6 CONCLUSIONES DEL CAPÍTULO	18
CAPÍTULO 2: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS	19
2.1 INTRODUCCIÓN DEL CAPÍTULO	19
2.2 MODELO DE DOMINIO.....	19
2.3 PROPUESTA DE SOLUCIÓN	21
2.4 REQUISITOS FUNCIONALES	21
2.4.2 REQUISITOS NO FUNCIONALES	22
2.5 DESCRIPCIÓN DE LAS ESPECIFICACIONES DE REQUISITO	23
2.7 DESCRIPCIÓN DE LA ARQUITECTURA	29
2.7.1 Arquitectura Cliente- Servidor.....	29
2.7.2 Patrón Arquitectónico	30
2.8 PATRONES DE DISEÑO	32
2.8.1 Patrones de diseño GRASP	33
2.8.2 Patrones de diseño GoF	35
2.9 DESCRIPCIÓN DEL DIAGRAMA DE DESPLIEGUE	36
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA DE LA SOLUCIÓN PROPUESTA	38
3.1 INTRODUCCIÓN	38
3.2 ESTÁNDARES DE CODIFICACIÓN.....	38
3.2.1 Indentación, llaves de apertura y cierre, y tamaño de las líneas	38
3.2.2 Convención de nomenclatura	38
3.3 ESTRATEGIAS DE PRUEBA	41
3.3.1 Estrategia de prueba a seguir para la validación del Sistema de Reportes	41
Métodos de pruebas basados en caja negra.....	42
3.5 RESULTADO DE LAS PRUEBAS REALIZADAS	45

| ÍNDICE DE CONTENIDOS

3.6 CONCLUSIONES DEL CAPÍTULO	46
CONCLUSIONES GENERALES.....	47
RECOMENDACIONES.....	48
GLOSARIO DE TÉRMINOS.....	49
REFERENCIAS BIBLIOGRÁFICAS	50
ANEXOS.....	52

Índice de tablas

Tabla 1: requisitos funcionales del sistema.	22
Tabla 2: requisitos no funcionales del sistema.	22
Tabla 3: especificación de requisito mostrar cantidad de clientes que han recibido servicios en el día de hoy.	24
Tabla 4: especificación del requisito mostrar cantidad de clientes que han accedido en cada evento del día por comedores.	26
Tabla 5: estrategia de prueba.	44
Tabla 6: descripción del caso de prueba para el requisito autenticar usuario.	44
Tabla 7: descripción del caso de prueba para el requisito total de grupos por puertas.	45
Tabla 8: descripción del caso de prueba para el requisito cantidad de clientes que no han recibido asignación de servicios.	45
Tabla 9: resultados de las pruebas.	46
Tabla 10: especificación de requisito: mostrar cantidad de clientes que accedieron por complejo.	55
Tabla 11: especificación de requisito: mostrar cantidad de clientes que no han recibido asignación de servicios.	56
Tabla 12: especificación de requisito: mostrar total de grupos por puerta.	58
Tabla 13: especificación de requisito: mostrar listado de los clientes pertenecientes al complejo 1, o 2 o 3.	59
Tabla 14: especificación de requisito: mostrar cantidad de clientes que recibieron servicios en un rango de fecha.	61
Tabla 15: especificación de requisito: autenticar usuario.	62
Tabla 16: especificación de requisito: mostrar cantidad de accedidos por nivel.	64
Tabla 17: especificación de requisito: mostrar cantidad de clientes por cada categoría en un evento.	66
Tabla 18: descripción del caso de prueba para el requisito mostrar cantidad de clientes que accedieron en cada evento del día por comedores.	68

Tabla 19: descripción del caso de prueba para el requisito mostrar cantidad de clientes que accedieron por complejo.....	69
Tabla 20: descripción del caso de prueba para el requisito mostrar cantidad de clientes que han recibido servicio en el día de hoy.....	69
Tabla 21: descripción del caso de prueba para el requisito mostrar cantidad de clientes que recibieron servicio en un rango de fecha.	70
Tabla 22: descripción del caso de prueba para el requisito mostrar cantidad de accedidos por nivel.	71
Tabla 23: descripción del caso de prueba para el requisito mostrar cantidad de clientes por cada categoría en un evento.	72
Tabla 24: descripción del caso de prueba para el requisito mostrar listado de los clientes pertenecientes al complejo 1 o 2 o 3.	73

Índice de figuras

Figura 1: modelo de dominio.	20
Figura 2: arquitectura cliente servidor.....	30
Figura 3: patrón Modelo Vista Controlador (MVC).....	31
Figura 4: diagrama de despliegue.	36
Figura 5: ejemplo de indentación, llaves de apertura y cierre, y tamaño de las líneas.	38
Figura 6: ejemplo de convención de nomenclaturas en variables.	39
Figura 7: ejemplo de convención de nomenclaturas en clases.....	39
Figura 8: ejemplo de convención de nomenclaturas en funciones.	40
Figura 9: ejemplo de estructuras de control.....	41
Figura 10: método de prueba caja negra.	42

Introducción

Las tecnologías de la información facilitan el desarrollo de las actividades diarias de la sociedad, expanden las vías de comunicación entre los seres humanos y brindan formas de almacenamiento de datos. El avance de las tecnologías de la información y las comunicaciones (TIC) contribuyó con la evolución del desarrollo científico técnico del ser humano, el cual posteriormente inventó las computadoras y con ellas surge la necesidad de informatizar procesos.

Hoy día las instituciones requieren ser informatizadas debido a la cantidad de información que ellas procesan para su desempeño. Generalmente, estas instituciones necesitan un mecanismo que les permita proporcionar información útil de manera rápida. Para ello existen herramientas que permiten a los usuarios obtener con facilidad datos de archivos o bases de datos en forma de reportes. Estos son objetos que entregan información en un formato particular y que permiten realizar ciertas operaciones como: imprimirlos, enviarlos por correo electrónico y guardarlos a un archivo, a partir de los datos almacenados en una base de datos.

La generación de reportes es una tarea necesaria en los sistemas que gestionan información. Generalmente los sistemas de gestión cuentan con un conjunto limitado de reportes, y se ven limitados desde el punto de vista de la inserción de nuevos reportes asociados a cambios a los procesos informatizados.

La Universidad de las Ciencias Informáticas (UCI) es una de las instituciones que debido al gran número de personas que concurren en sus comedores se hace necesario disponer de un sistema que gestione los servicios de alimentación. El Centro de Informatización Universitaria (CENIA) perteneciente a la Facultad 1 es el encargado de desarrollar las soluciones para la informatización de la UCI. Uno de estos procesos informatizados es el de alimentación, para ello existen sistemas encargados de gestionar y publicar la información relacionada con la planificación y servicios de la alimentación, la aplicación que se encuentra en los comedores llamada Control de Acceso a Comedores o Cajero, la cual permite registrar el acceso de cada comensal, el Sistema de Asignación en el cual los funcionarios de la Dirección de Alimentación

| INTRODUCCIÓN

se encargan de realizar la distribución de los grupos por cada complejo comedor y el Sistema de Reportes para la Dirección de Alimentación.

El sistema de Control de Acceso a Comedores exporta un fichero xml que más tarde es cargado en el sistema Control de Acceso a Comedores instalado en cada computadora ubicada en las puertas de los comedores, la contabilidad de personas que accedieron al sistema se sabe mediante fotos que cada cierto intervalo de tiempo se le hacen al sistema Control de Acceso debido a que el Sistema de Reportes de la Dirección de Alimentación no se encuentra funcionando. Estos tres sistemas trabajan sobre la misma base de datos.

Actualmente el Sistema de Reportes no posee todas las funcionalidades necesarias para satisfacer las necesidades de los distintos niveles de la Dirección de Alimentación, cuenta con una serie de reportes que no funcionan por lo que les dificulta tener un control estadístico de los accesos debido a que no muestra datos verídicos, la ausencia o el mal funcionamiento de estos reportes hace que la Dirección de Alimentación planifique los recursos de alimentación guiándose por datos que no son reales lo que trae consigo que quizás la cantidad de comida planificada para cada evento no sea la correcta, es decir, puede que planifiquen más, o menos de lo que en realidad se necesita, además no existe la manera de exportar estos reportes para que más tarde a partir de los datos mostrados la Dirección de Alimentación pueda hacer una correcta planificación de los recursos para cada evento.

A raíz de la problemática anteriormente planteada se formula el siguiente **problema de investigación**: ¿cómo mejorar las funcionalidades que permiten la obtención de la información generada en las actividades asociadas a los servicios de alimentación de la UCI?

Para enmarcar los límites de la investigación se define como **objeto de estudio**: la gestión de reportes, proponiendo como **objetivo general**: desarrollar un sistema de reportes, con tecnologías que contribuyan al desarrollo de la soberanía tecnológica, para mostrar la información relacionada con los procesos que gestiona la Dirección de Alimentos de la Universidad de las Ciencias Informáticas y apoye el proceso de toma de decisiones.

Para dar cumplimiento al **objetivo general** se plantean los siguientes **objetivos específicos**:

| INTRODUCCIÓN

- Estudiar y analizar bibliografías de todos los conceptos y términos relacionados con la gestión de reportes.
- Valorar las herramientas, tecnologías, lenguajes y metodología a utilizar para el desarrollo del sistema de reportes.
- Diseñar la propuesta de solución.
- Desarrollar un sistema de reportes.
- Validar las funcionalidades del sistema.

Durante la investigación se utilizan métodos de investigación teóricos y empíricos, dentro de los **métodos teóricos** se encuentran:

- **Analítico - Sintético:** durante la investigación se analizan documentos, páginas de Internet y otros artículos de donde se extraen los elementos más importantes relacionados con sistemas de reportes. Además se realizan resúmenes y valoraciones de las características más relevantes de estos sistemas.
- **Histórico – Lógico:** durante la investigación se estudió la evolución de la calidad del sistema. Se realizó un estudio crítico de los trabajos anteriores, para la utilización de los mismos como punto de referencia y comprobación de los resultados alcanzados.
- **Sistémico:** es utilizado a la hora de dar cada una de las soluciones de los elementos que se van a desarrollar para lograr que funcione como un sistema.

Como **métodos empíricos** para esta investigación se utilizan:

- **Taller de participación grupal:** se pone en práctica para poder adquirir información sobre la aplicación y su utilización en la UCI, así como de los requisitos funcionales.
- **La observación:** para analizar como las principales herramientas de personalización de sistemas ya existentes desarrollan este proceso, obteniendo conocimientos que serían usados en la implementación.

Se plantea como **justificación de la investigación** que: el desarrollo de un sistema de reportes asociados a los procesos de la dirección de alimentación de la UCI apoyará el proceso de toma de decisiones y facilitará la visualización de la información gestionada.

El presente documento se ha estructurado en capítulos de la siguiente forma:

Capítulo I. Fundamentación teórica del sistema de reportes para la Dirección de Alimentación de la Universidad de las Ciencias Informáticas: en este capítulo se presentan los distintos sistemas de reportes existentes en el ámbito internacional, nacional y dentro de los nacionales los que se han desarrollado en la UCI. Se realiza además un análisis de las herramientas y tecnologías a utilizar para el desarrollo del sistema de reportes.

Capítulo II. Propuesta de solución del sistema de reportes para la Dirección de Alimentación de la Universidad de las Ciencias Informáticas: en este capítulo se definen los requisitos de *software*, los artefactos necesarios para la implementación de la solución propuesta. Además se exponen las principales características del sistema, su diseño y arquitectura.

Capítulo III. Validación del sistema de reportes para la Dirección de Alimentación de la Universidad de las Ciencias Informáticas: se describen las pruebas realizadas a la propuesta de solución con el objetivo de verificar que cumple con todas las funcionalidades requeridas.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En el presente capítulo se exponen elementos teóricos que respaldan la investigación, guiado con el estudio y análisis de los sistemas homólogos existentes además de un estudio donde se describen aspectos importantes sobre los lenguajes y tecnologías definidas para darle solución al problema planteado.

1.2 Conceptos asociados al objeto de estudio

Reporte: un reporte es una noticia o informe que brinda información con algún propósito. En el ámbito de la informática es un informe que organiza y exhibe la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. (1)

A través de los reportes se refleja el comportamiento de los diferentes componentes que integran cualquier sistema de información de una empresa o institución, permitiendo el reconocimiento, control y monitoreo de las diversas problemáticas existentes en la misma, para de esa forma apoyar en la correcta toma de decisiones.

Sistema de generación de reportes: en todo sistema de información es fundamental contar con una herramienta adicional cuyo objetivo sea la de generar reportes. Los sistemas generadores de reportes tienen en las bases de datos su principal fuente de alimentación, ya que a partir de la información almacenada en la misma se realizan consultas para obtener información en forma de reporte. A diferencia de las consultas tradicionales, con los generadores de reportes se puede definir el diseño y la forma en que la información será visualizada, es por ello que se compone por un diseñador de reportes y por un motor de generación de reportes, donde el primero se encarga de brindar las herramientas para diseñar la apariencia del informe y el segundo accede a la fuente de datos, obtiene los necesarios y los introduce en el diseño de la plantilla del reporte con la que luego se puede realizar ciertas operaciones como: imprimir, enviar por correo electrónico o guardar a un archivo. (2)

1.3 Sistemas informáticos homólogos

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

A continuación se muestra un resumen de los sistemas informáticos compuestos por módulos de reportes para la gestión de información, los cuales permiten que la información mostrada se exporte en plantillas con un formato particular.

1.3.1 Ámbito internacional

CENACAD: el Sistema de Censo Académico en Línea para la Automatización de la Evaluación a los Docentes, fue realizado por parte del personal de la Escuela Superior Politécnica del Litoral (ESPOL), Guayaquil, Ecuador. Nace de la necesidad de brindar una solución informática sobre las evaluaciones docentes en la Escuela Superior Politécnica del Litoral (ESPOL) (3). Cuenta con un módulo de reportes el cual permite a los usuarios la obtención de diferentes tipos de reportes, a continuación se muestran algunos de ellos:

- Listado de mejores docentes en base a los mejores promedios obtenidos en la ESPOL por el período de un año.
- Promedio de la(s) facultad(es) o instituto(s).
- Promedios obtenidos en cada área que conforman cada una de las encuestas realizadas a cada materia. Estos promedios deberán ser general, de toda la facultad o unidad. Además son presentadas de manera gráfica.

1.3.2 Ámbito nacional

Info@tletas: es la aplicación web oficial del Instituto Nacional de Deportes, Educación Física y Recreación (INDER). Surge como colofón de varias aplicaciones de escritorio confeccionadas según la solicitud de la dirección de alto rendimiento del INDER. Se encarga principalmente de la administración de los expedientes técnico-acumulativos de atletas y entrenadores nacionales e internacionales. La aplicación cuenta con 12 módulos, entre los que se incluye el Módulo de Reportes encargado de generar reportes estadísticos de atletas y entrenadores visualizados de forma tabular y gráfica de barras (4).

Los reportes se comportan según el tipo de usuario que ha iniciado su sesión en el sistema:

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

- Supervisor Especial y Supervisor Nacional: visualizan los reportes a nivel nacional.
- Supervisor Provincial: visualizan los reportes a nivel provincial.
- Administrador, Metodólogo y Técnico: visualizan los reportes a nivel de centro.

Sistema de Gestión Universitaria: maneja la información referente a los procesos fundamentales en las áreas de la Universidad de las Ciencias Informáticas. Este cuenta con una serie de subsistemas que engloban diferentes tipos de áreas, uno de ellos es el Sistema de Gestión Académica de pregrado que se encarga de la gestión de los procesos académicos. Como tecnologías básicas para el desarrollo de este subsistema se tuvo las técnicas empíricas basadas en programación, utilizando PHP, jquery y código SQL. Este subsistema muestra reportes de fácil entendimiento acerca del resumen de evaluaciones, notas de las asignaturas y semestre del estudiante, bonificaciones, exámenes de premio y promedio. El usuario es capaz de obtener la información mediante el filtrado de los diferentes componentes y esta información puede ser exportada en formato pdf.

1.4 Resumen del análisis de los sistemas homólogos estudiados

El estudio de los diferentes sistemas informáticos existentes a nivel nacional e internacional permitió observar y analizar los diferentes formatos para la presentación de las salidas de información. Se pudo observar que estos módulos de reportes tienen características y funcionalidades similares, ya que fueron creados con el mismo propósito de ayudar a los usuarios a obtener reportes de información almacenada en una base de datos además de poder exportarlos a formatos para ser imprimidos. Las diferencias radican en que cada uno responde a ciertos requerimientos del sistema de información para el cual se implementó. Conocer las funcionalidades que realizan los sistemas estudiados ayudó al mejor entendimiento del objeto de estudio, pero se determinó que ninguno de estos sistemas está acorde a lo que se quiere diseñar según las necesidades del cliente.

1.5 Lenguajes, tecnologías, herramientas y proceso de desarrollo de software

El sistema de reportes para la dirección de alimentación de la universidad de las ciencias informáticas fue desarrollado utilizando los lenguajes, tecnologías y herramientas establecidas por CENIA y guiado por el proceso de desarrollo con enfoque ágil basado en el nivel 2 de CMMI (*Capability Maturity Model Integration*, en español Integración de Modelos de Madurez de Capacidades). A continuación se realiza una descripción de los mismos:

1.5.1 Proceso de desarrollo con enfoque ágil basado en el nivel 2 de CMMI

Las metodologías ágiles de desarrollo están especialmente indicadas en proyectos con requisitos poco definidos o cambiantes. Estas metodologías se aplican bien en equipos pequeños que resuelven problemas concretos, lo que no está reñido con su aplicación en el desarrollo de grandes sistemas, ya que una correcta modularización de los mismos es fundamental para su exitosa implantación. Dividir el trabajo en módulos abordables minimiza los fallos y el coste. Las metodologías ágiles presentan diversas ventajas, entre las que se destacan:

- Capacidad de respuesta a cambios de requisitos a lo largo del desarrollo.
- Entrega continua y en plazos breves de software funcional.
- Trabajo conjunto entre el cliente y el equipo de desarrollo.
- Importancia de la simplicidad, eliminando el trabajo innecesario.
- Atención continua a la excelencia técnica y al buen diseño.
- Mejora continua de los procesos y el equipo de desarrollo.

Dentro de las metodologías ágiles se encuentran la Scrum y Programación Extrema (XP), a continuación se dará una breve panorámica sobre las características de cada una de estas metodologías (5).

1.5.1.1 Scrum

Scrum es una metodología ágil enfocada a la gestión de proyectos. Sus principales características se pueden resumir en dos: el desarrollo de sprint o iteraciones y reuniones a lo largo del desarrollo. Las iteraciones en Scrum tienen una duración máxima de 30 días y el resultado de cada una de ellas define un incremento del producto a desarrollar. La evolución del proyecto por la metodología se define a través de reuniones diarias donde el trabajo del día

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

anterior es revisado por el equipo, previendo además la labor a realizar el día siguiente. Dentro de las prácticas definidas por la metodología Scrum se encuentran (5):

- Planificación de la iteración o *sprint*.
- Revisión de la iteración o *sprint*.
- Reunión diaria.
- Pila del producto.
- Incremento.
- Propietario del producto.
- Auto-organización.
- XP (*Programación Extrema*).

1.5.1.2 Programación Extrema (XP)

Es una metodología ágil de desarrollo de *software* que posee cuatro tareas fundamentales: planificación, diseño, desarrollo y pruebas. Esta metodología está basada en la simplicidad durante el desarrollo, la comunicación entre las partes implicadas (clientes y desarrolladores) y la retroalimentación para poder reutilizar el código desarrollado. En su concepción establece entregas frecuentes con posibilidad de refactorización continua, permitiendo mejorar el diseño cada vez que se añade una funcionalidad. Para su implementación XP establece un conjunto de prácticas que deben ser empleadas en los proyectos de desarrollo, las mismas se mencionan a continuación (5):

- El juego de la planificación
- Entregas pequeñas
- Metáfora
- Diseño simple
- Pruebas
- Refactorización (*Refactoring*)
- Programación en parejas
- Propiedad colectiva del código
- Integración continua
- 40 horas por semana

- Cliente in-situ
- Estándares de programación

1.5.1.3 CMMI

CMMI es un modelo de madurez de mejora de los procesos para el desarrollo de productos y de servicios. Consiste en las mejores prácticas que tratan las actividades de desarrollo y de mantenimiento que cubren el ciclo de vida del producto, desde la concepción a la entrega y el mantenimiento, específica que deben hacer los proyectos para entregar productos de calidad pero no dice como llevarlo a cabo (5).

Este modelo mide la madurez del desarrollo del software mediante 5 niveles:

- Nivel 1 (Inicial): el proceso es impredecible, es reactivo y pobremente controlado.
- Nivel 2 (Administrado): el proceso es reactivo y se caracteriza por su aplicación a proyectos.
- Nivel 3 (Definido): el proceso es proactivo y se ve a nivel de la organización.
- Nivel 4 (Administrado Cuantitativamente): el proceso es medido y controlado.
- Nivel 5 (Optimizado): el proceso se enfoca en la mejora continua.

La Universidad de las Ciencias Informáticas está llevando a cabo desde el año 2008 un proceso de mejora encaminado a alcanzar el nivel 2 del modelo CMMI adoptando como guía en el desarrollo del software la integración de las metodologías ágiles Scrum y (XP). Este proceso persigue como objetivos dar garantía a las cuatro demandas principales de la industria en la que se ha generado: valor, reducción del tiempo de desarrollo, agilidad y fiabilidad.

De acuerdo al conjunto de prácticas propuestas por el modelo CMMI y la simplificación del empleo de los métodos ágiles durante el desarrollo de software, autores como *Lebsanft* plantean que ambos son inaplicables en negocios turbulentos. Sin embargo en el 2008, se hace un artículo "*CMMI or Agile: Why not embrace both*" donde se expone la compatibilidad entre este modelo y el enfoque ágil, siempre y cuando se utilicen correctamente dando lugar a la posibilidad de aplicación de este modelo conjuntamente con metodologías de desarrollo de *software*.

1.5.2 Lenguaje de programación

1.5.2.1 PHP

PHP en su versión 5.3 significa *Hypertext Pre-processor*, es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es un lenguaje de propósito general ampliamente usado y que está diseñado especialmente para desarrollo web y puede ser embebido dentro de código HTML. Se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. El código fuente escrito en PHP es invisible al navegador y al cliente, esto hace que la programación sea segura y confiable (6).

Entre sus principales ventajas se puede mencionar:

- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con PostgreSQL.
- Posee una amplia documentación en su página oficial, entre la cual sobresale que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Permite las técnicas de Programación Orientada a Objeto (POO).
- Posee una biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.

1.5.2.2 JavaScript

JavaScript en su versión 1.2 es un lenguaje de programación interpretado, se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. A pesar de su nombre, no guarda ninguna relación directa con el lenguaje de programación Java, tienen semánticas y propósitos diferentes.

Los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del *Document Object Model* (DOM).

Existen al menos tres formas para incluir código JavaScript en las páginas web:

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

- Incluir el código en la misma página web: el código JavaScript se encierra entre etiquetas `<script>` y se incluye en cualquier zona del documento. Aunque es correcto incluir el bloque de código dentro de la cabecera del documento.
- Definir el código en un archivo externo: las instrucciones JavaScript se incluyen en un archivo de extensión `*.js` que luego es enlazado a través de etiquetas dentro del código de la página. Es muy útil cuando se trata de reutilizar código para las demás páginas web dentro de un mismo proyecto.
- Incluir el código dentro de elementos de la página web: consiste en incluir trozos de código JavaScript dentro del código XHTML de la página. Se utiliza, en ocasiones, para definir eventos, puesto que complica el mantenimiento del código.

JavaScript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, con funciones y estructuras de datos complejas. Además, pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente (7).

1.5.2.3 CSS

Las Hojas de Estilo en Cascada (*Cascading Style Sheets*, CSS) en su versión 2.0 es un lenguaje creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas. Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo, también llamados documentos semánticos. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes (8).

1.5.3 Lenguaje de modelado UML

El Lenguaje Unificado de Modelado (*Unified Modeling Language*, UML) en su versión 2.0 se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software. Es el lenguaje de modelado más conocido y utilizado en la actualidad. Se

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. UML incluye conceptos semánticos, notación, y principios generales. Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo, pero no especifica en sí mismo qué metodología o proceso usar (9).

1.5.4 Entorno de desarrollo integrado NetBeans IDE

El entorno de desarrollo integrado NetBeans en su versión 7.3 está disponible para Windows, MacOS, Linux y Solaris. Consiste en un IDE de código abierto y una plataforma de aplicación, las cuales pueden ser usadas como una estructura de soporte general para compilar cualquier tipo de aplicación.

Es una herramienta para que los programadores puedan escribir, depurar y ejecutar programas. Está escrito en Java, es un producto libre y gratuito sin restricciones de uso (10).

Principales características:

- Propone un esqueleto para organizar el código fuente, el editor conjuntamente integra los lenguajes como HTML, JavaScript y CSS. Además posee un sistema para examinar todos los directorios de cada proyecto, haciendo reconocimiento y carga de clases, métodos y objetos, para acelerar la programación.
- El editor de PHP, es mucho más ágil y a la vez robusto, contiene más ayuda en línea, reconocimiento de sintaxis y todo lo que provee la última versión de PHP.

1.5.5 Herramienta para el diseño de la aplicación

1.5.5.1 Visual Paradigm

Visual Paradigm en su versión 8.0 es una herramienta de diseño que hace uso del UML, soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Brinda además, la posibilidad de crear, modificar y diseñar estos diagramas con rapidez y calidad, lo que ayuda a aumentar la eficiencia del sistema de análisis y diseño de manera significativa (11).

Características principales:

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

- Tiene disponibilidad en múltiples plataformas y en múltiples versiones. Esta característica es muy importante pues Visual Paradigm está disponible para varios sistemas operativos como Windows, Linux, Unix.
- Proporciona una plataforma de modelado colaborativo para el trabajo en equipo, los miembros pueden ver y editar el mismo proyecto, o el mismo esquema, incluso de forma simultánea.
- Todos los cambios se almacenan en el servidor de Visual Paradigm en función de revisión.
- Brinda la posibilidad de generar código a partir de los diagramas, para plataformas como .Net, Java y PHP, así como obtener diagramas a partir de código.

1.5.5.2 EvolusPencil

EvolusPencil en su versión 1.3.4 es una herramienta gratuita y de código abierto que posibilita la creación de diagramas y prototipos de interfaz gráfica de usuario. EvolusPencil es la evolución de Pencil, es una extensión de Firefox que se utiliza para el diseño de los prototipos de las interfaces de usuario, se caracteriza por:

- Brindar un conjunto de componentes como: entradas de texto, íconos y botones.
- Edición en pantalla de los elementos de texto.
- Permitir exportar imágenes al formato png, html o pdf.
- Permitir operaciones estándar de dibujo: alineado, escalado, rotación.
- Ser multi-plataforma.
- Posibilitar, a través de las propiedades de los componentes, cambiar el estilo al diseño (12).

1.5.6 Herramienta para la generación de reportes

1.5.6.1 IReport

IReport en su versión 5.0.4 es una herramienta visual que sirve para generar ficheros XML (plantillas de informes) utilizando la herramienta de generación de informes JasperReport. Escrito en Java IReport provee a los usuarios de JasperReport una interfaz visual para construir reportes. También permite que los usuarios corrijan visualmente informes complejos con cartas,

imágenes y subinformes. Está además integrado con *JFreeChart*, una de la bibliotecas gráficas *OpenSource* más difundida para Java (13).

Las siguientes son algunas de sus características

- 100% escrito en Java además *OpenSours* y gratuito.
- Maneja el 98% de las etiquetas de JasperReports.
- Permite diseñar con sus propias herramientas: rectángulos, líneas, elipses, campos de texto (*textfields*), cartas, *subreports* (subreportes).
- Tiene asistentes para generar los subreportes.
- Tiene asistentes para las plantillas.
- Facilidad de instalación.

1.5.6.2 JasperReports

JasperReports en su versión 4.7 es una poderosa librería Java de código abierto, que combinada con herramientas para el diseño facilita y agiliza la generación, la pre-visualización y la impresión de los reportes. Está licenciada bajo Licencia Pública General Reducida (*Lesser General Public License*, LGPL) y permite su uso en aplicaciones de código abierto y código cerrado. Puede reutilizarse tanto en aplicaciones cliente y cliente/servidor, como en aplicaciones web.

Es un módulo que dispone de un depósito de archivos que usa un sistema de carpetas, una aplicación web que muestra todos los informes que están en el depósito y un visor de dichos informes (14).

Características de JasperReports:

- JasperReports es capaz de generar reportes profesionales que incluyen imágenes y gráficas.
- Posee un diseñador de reportes flexible, que permite mostrar los datos de disímiles maneras y acepta datos de una gran variedad de fuentes.
- Los reportes que se generan mediante su uso pueden ser exportados a varios formatos como PDF, Excel, entre otros.

1.5.7 Marco de trabajo Symfony

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

Symfony en su versión 2.2 es un *framework* de aplicaciones web escrito en PHP que sigue el modelo-vista-controlador (MVC). Symfony tiene como objetivo acelerar la creación y mantenimiento de aplicaciones web y sustituir repetitivas tareas de codificación.

Symfony2.2.1 es la versión más reciente de Symfony, el popular *framework* para desarrollar aplicaciones PHP. Symfony2.2.1 ha sido ideado para exprimir al límite todas las nuevas características de PHP 5.3 y por eso es uno de los *frameworks* PHP con mejor rendimiento (15).

Características:

- Fácil de instalar y configurar en la mayoría de plataformas (y garantizado para trabajar en el estándar de *Windows*).
- Base de datos del motor independiente.
- Fácil de usar, en la mayoría de los casos, pero todavía lo suficientemente flexible como para adaptarse a los casos complejos.
- Partiendo de la premisa de convención sobre configuración el desarrollador tiene que configurar sólo lo no convencional.
- Compatible con la mayoría de las prácticas *Web* y patrones de diseño.

1.5.8 Servidor web

Es un programa que se ejecuta continuamente en un computador, manteniéndose a la espera peticiones de ejecución que le hará un cliente o un usuario de Internet. El servidor *web* se encarga de contestar a estas peticiones de forma adecuada, entregando como resultado una página *web* o información de todo tipo, de acuerdo a los comandos solicitados (16).

1.5.8.1 Servidor web Apache

Apache en su versión 2.2 es un servidor web altamente configurable, robustez y estabilidad. Es un sistema de código abierto para plataformas *Windows*, *Unix*, *Macintosh* y otras que implementa el protocolo *HTTP* (17).

Características principales:

- Es una tecnología gratuita de código fuente abierta.
- Es personalizable, la arquitectura modular de Apache permite construir un servidor hecho a la medida y posibilita la implementación de los últimos y nuevos protocolos.

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

- Es un servidor altamente configurable de diseño modular. Permite aumentar fácilmente su capacidad e instalar cualquier módulo para cumplir una función específica.

Principales mejoras de Apache 2.0:

- Tiene la infraestructura necesaria para servir distintos protocolos.
- Rapidez y estabilidad en sistemas que no son tipo Unix, tales como BeOS, OS/2 y Windows.
- Nueva interfaz de programación (API) para los módulos, muchos de los problemas de ordenación y prioridad de módulos de la versión 1.3 desaparecieron.
- Los módulos de Apache pueden escribirse para que se comporten como filtros que actúan sobre el flujo de contenidos tal y como salen del servidor, o tal y como son recibidos por el servidor.

1.5.9 Sistema de gestión de base de datos (SGBD)

Un SGBD se define como el conjunto de programas dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Tiene el objetivo de suministrarle al usuario las herramientas que le permitan manipular, en términos abstractos, los datos; o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado, garantizando la seguridad e integridad de la información (18).

1.5.9.1 PostgreSQL

PostgreSQL 8.4.1 es el sistema de gestión de base de datos relacional orientada a objeto, publicado bajo la licencia BSD. Es el SGBD de código abierto más potente del mercado.

Principales ventajas:

- Estabilidad, flexibilidad y alto rendimiento.
- Administración efectiva de seguridad.
- Ágil navegación y administración de base de datos.
- Excelentes herramientas visuales y de texto para elaboración de consultas.
- Se puede extender su funcionalidad.

| CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

- Permite crear o migrar aplicaciones desde Access, Visual Basic, Visual Fox Pro, y Visual C/C.
- Varias Interfaces de Programación: ODBC, JDBC, C/C++, SQL Embebido, Perl, Python, PHP.
- Conexión vía reenvío de puerto local a través del túnel SSH.
- Impresionantes opciones de exportación e importación de datos (19).

1.5.9.2 Administrador de base de datos PgAdmin

PgAdmin 1.14.0 es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia *Open Source*. Está escrita en C++ usando la librería gráfica multiplataforma *wxWidgets*, lo que permite que se pueda usar en Linux, FreeBSD, Solaris, Mac OS X y Windows. Es capaz de gestionar versiones a partir de la PostgreSQL 7.3 ejecutándose en cualquier plataforma (18) .

1.6 Conclusiones del capítulo

Con el estudio de las herramientas y sistemas generadores de reportes existentes se arribó a la conclusión de cuál es la más indicada para darle solución a la problemática planteada además de aportar una visión más amplia del objeto de estudio de la investigación.

Por último, se realizó un estudio de los lenguajes de programación, herramientas y tecnologías utilizadas por el CENIA, lo que permitió adquirir y fomentar los conocimientos necesarios para el desarrollo de la propuesta de solución.

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

Capítulo 2: Propuesta de solución del sistema de reportes para la dirección de alimentación de la Universidad de las Ciencias Informáticas

2.1 Introducción del capítulo

En el presente capítulo se realiza el análisis del sistema a desarrollar, una descripción de los principales conceptos significativos para el problema a desarrollar quedando plasmados mediante un modelo de dominio los procesos que intervienen en la generación de reportes de la Dirección de Alimentación de la UCI. Se especifican las técnicas de obtención de requisitos, identificando así los requisitos funcionales y no funcionales. Se realiza la descripción de la arquitectura y patrones empleados.

Debido a que no se define claramente un proceso de negocio, se hace necesario realizar una conceptualización o modelo de dominio donde se explican claramente los principales conceptos definidos.

2.2 Modelo de dominio

El Modelo de dominio o Modelo conceptual es una representación visual de los principales conceptos u objetos del mundo real, significativos para un problema o área de interés. Este es de gran ayuda para desarrolladores y usuarios, ya que de esta forma utilizan un vocabulario común y pueden entender el contexto en que se enmarca el sistema (20).

El sistema a desarrollar es un sistema de reportes cuya función principal es mostrar en forma de reportes consultas que se realizan a la base de datos, lo que implica que no es un sistema tradicional que necesite gestionar algún tipo de proceso por lo que no se evidencia con claridad los procesos ni los actores que intervienen, además de que las funcionalidades no presentan una variabilidad marcada debido a que todas son reportes, no se tiene claramente definido que usuario se beneficia de algún reporte en particular lo que dificulta establecer alguna relación de asociación entre los usuarios y las funcionalidades del sistema, es por esto que se propone un modelo de dominio donde se explicarán los conceptos con los que se relacionan los usuarios y el sistema.

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

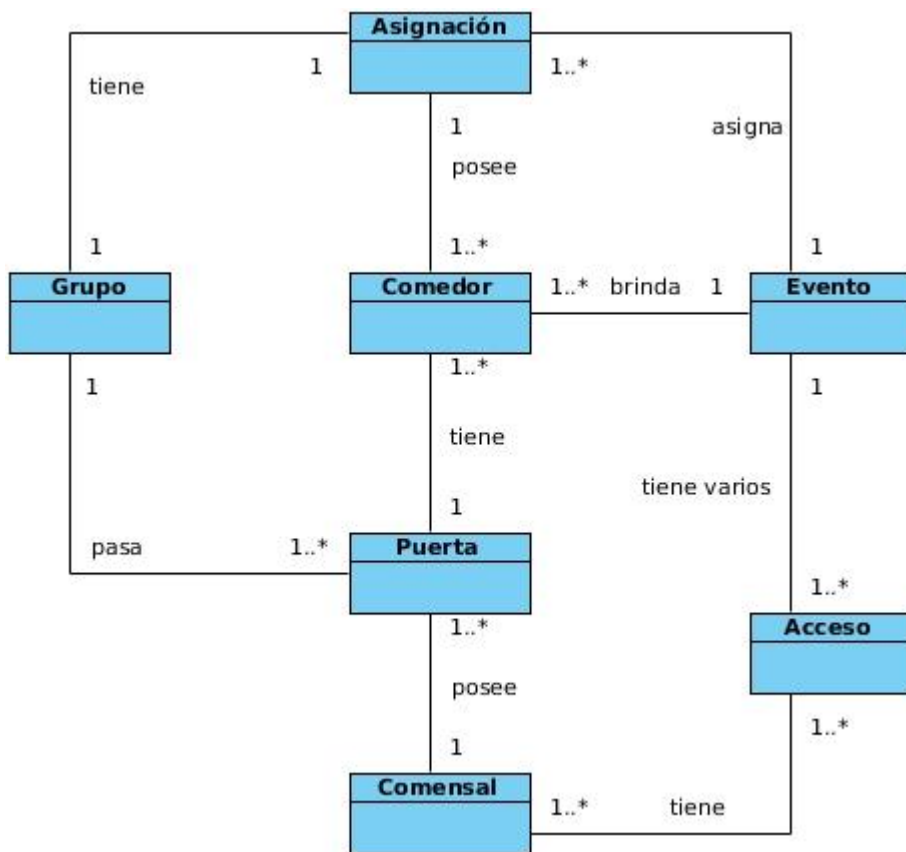


Figura 1: modelo de dominio.

Definición de las clases del modelo de dominio

- **Comedor:** se refiere a cada uno de los comedores pertenecientes a cada complejo comedor.
- **Puerta:** se refiere a cada puerta de cada uno de los comedores de cada complejo.
- **Asignación:** se refiere a la asignación que se realiza de los grupos y eventos a cada comedor.
- **Comensal:** se refiere a las personas que están autorizadas a pasar por cada puerta del comedor.
- **Evento:** este concepto está asociado al desayuno, almuerzo y comida.
- **Grupo:** se refiere a un conjunto de personas que están asignadas a una puerta.
- **Acceso:** se refiere a los permisos que tiene un comensal de acceder a un evento.

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

2.3 Propuesta de solución

Se propone una aplicación web que permitirá a la Dirección de Alimentación de la UCI consultar toda la información necesaria para realizar la planificación de los recursos de alimentación, estas consultas se mostrarán a través de reportes, la propuesta de solución le permitirá al usuario conocer la cantidad de personas a las que se le ha ofrecido servicios, además de poder saber la distribución realizada por cada complejo comedor entre otros reportes importantes para la Dirección de Alimentación, se propone además como parte de la propuesta de solución un análisis profundo a cerca de los métodos o vías más factibles para lograr que este departamento logre hacer una correcta planificación de los recursos de alimentación y de esta forma apoyar el proceso de toma de decisiones.

2.4 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe realizar, es decir, define qué es lo que el sistema debe hacer, así como las funciones que el sistema será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas, para producir salidas (21).

2.4.1 Técnicas de obtención de requisitos

Las técnicas de obtención de requerimientos son aquellas que permiten comprender el dominio del sistema, buscar y recolectar información para definir sus límites y restricciones, e identificar a las personas interesadas en el sistema. El resultado brindará obtener una colección y clasificación de los requerimientos del sistema, mediante la participación de los clientes y usuarios (22).

Las técnicas aplicadas en esta investigación son:

- **Entrevistas:** Se llevó a cabo una conversación con el cliente con el objetivo de entender el dominio del problema y sus necesidades. Esta se basó en un formato de preguntas y respuestas, buscando obtener criterios sobre el sistema actual con vista a mejorarlo.
- **Observación del flujo actual de los procesos:** Esta técnica ayuda a la obtención de información de los procesos actuales y de las principales características del sistema que antecede a esta investigación. La aplicación de esta técnica aportó grandes beneficios

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRRECCIÓN DE ALIMENTACIÓN DE LA UCI

para la elaboración de los requisitos funcionales que conformarán la propuesta de solución.

- **Investigación de sistemas existentes:** Se analizó el sistema que ya existe para valorar las funcionalidades que tiene que pueden seguir formando parte de la propuesta de solución y así llegar a conformar nuevas funcionalidades que pueden ser importantes.

A continuación se muestra la tabla de los requisitos funcionales obtenidos, de donde se derivarán las funcionalidades que contendrá el sistema propuesto.

Tabla 1: requisitos funcionales del sistema.

Requisitos funcionales		Complejidad
RF_1	Autenticar usuario en el sistema.	Media
RF_2	Mostrar listado de los clientes que no han recibido asignación de servicio.	Alta
RF_3	Mostrar total de grupos por puertas.	Media
RF_4	Mostrar listado de los clientes pertenecientes al complejo 1 o 2 o 3.	Alta
RF_5	Mostrar cantidad de clientes que han recibido servicio en el día de hoy.	Media
RF_6	Mostrar cantidad de clientes que accedieron por cada categoría en un evento.	Media
RF_7	Mostrar cantidad de clientes que accedieron en cada evento del día por comedores.	Alta
RF_8	Mostrar cantidad de clientes que accedieron en cada evento del día por complejo.	Alta
RF_9	Mostrar cantidad de clientes que accedieron por nivel.	Baja
RF_10	Mostrar cantidad de clientes que recibieron servicios en un rango de fecha.	Media

2.4.2 Requisitos no funcionales: los requisitos no funcionales son propiedades o cualidades que el producto debe tener. El sistema a desarrollar cuenta con los siguientes requisitos no funcionales (23):

Tabla 2: requisitos no funcionales del sistema.

Requisitos no funcionales	
Usabilidad	El sistema debe presentar una interfaz que permita una fácil interacción con el usuario y llegar

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

	<p>de manera rápida y efectiva a la información buscada. Debe además, ser una interfaz agradable que posibilite a los usuarios sin experiencia una rápida adaptación.</p> <p>Especificación de la terminología utilizada: el módulo debe adaptarse al lenguaje y términos utilizados por los usuarios en la rama abordada con vista a una mayor comprensión por parte del cliente de la herramienta de trabajo.</p>
Seguridad	<p>La seguridad de la base de datos está a nivel de roles, con el fin de mantener la integridad de los datos en función del acceso de cada uno de ellos, brindando además la protección de la información. La seguridad del sistema se garantiza mediante roles, mediante usuarios y contraseña definidos en el sistema.</p>
Restricciones del diseño	<p>Servidor de base de datos con Postgres 8.4.</p> <p>Servidor de aplicaciones Web: Apache 2.0 o superior.</p> <p>Navegador Web: Mozilla Firefox 2.3 o superior.</p>
Restricciones de implementación	<p>El sistema debe ser desarrollado con:</p> <ul style="list-style-type: none"> • Entorno de desarrollo: NetBeansIDE 7.3. • Lenguaje de programación: PHP 5.3. • Marco de trabajo base de desarrollo: Symfony2
Interfaz	<p>Interfaz web: la interfaz debe ser sencilla con colores suaves a la vista y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo.</p>
Software	<p>PC cliente multiplataforma.</p> <p>PC servidor de base de datos: el servidor debe contar con el sistema operativo Linux y el SGDB PostgreSQL 8.4.</p>

2.5 Descripción de las especificaciones de requisito

Las especificaciones de requisitos son la técnica utilizada en el proceso de desarrollo de software con enfoque ágil para un nivel 2 de CMMI para especificar los requisitos del *software*. Las mismas son descritas por los clientes como las tareas que el sistema debe hacer. A continuación se muestran algunas descripciones de los requisitos con prioridad alta.

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

Tabla 3: especificación de requisito mostrar cantidad de clientes que han recibido servicios en el día de hoy.

N °	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF5.]	Mostrar cantidad de clientes que han recibido servicios en el día de hoy.	Una vez que el usuario de clic sobre la funcionalidad se mostrará el evento, la cantidad de servicios ofertados y en la parte inferior izquierda el total de todos los servicios ofertados. En la esquina superior izquierda aparecerá la opción de exportar a pdf para poder imprimir el reporte.	Media	
Prototipo				
Prototipo Mostrar cantidad de clientes que han recibido servicios en el día de hoy.				

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRRECIÓN DE ALIMENTACIÓN DE LA UCI

Banner														
Inicio	Reportes	Bienvenido: oldiaz Salir												
Cantidad de clientes que han recibido servicios en el día de hoy.														
Versión pdf														
<table border="1"> <thead> <tr> <th>Eventos</th> <th>Cantidad de accesos</th> </tr> </thead> <tbody> <tr> <td>Desayuno</td> <td>150</td> </tr> <tr> <td>Almuerzo</td> <td>400</td> </tr> <tr> <td>Doble almuerzo</td> <td>600</td> </tr> <tr> <td>Comida</td> <td>580</td> </tr> <tr> <td>Total</td> <td>1730</td> </tr> </tbody> </table>			Eventos	Cantidad de accesos	Desayuno	150	Almuerzo	400	Doble almuerzo	600	Comida	580	Total	1730
Eventos	Cantidad de accesos													
Desayuno	150													
Almuerzo	400													
Doble almuerzo	600													
Comida	580													
Total	1730													
Campos	Tipos de Datos	Reglas o Restricciones												
Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos. 													

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

Tabla 4: especificación del requisito mostrar cantidad de clientes que han accedido en cada evento del día por comedores.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF7]	Mostrar cantidad de clientes que accedieron por comedores.	Primeramente se muestra una lista despegable donde el usuario podrá elegir el complejo, el comedor, la puerta y la fecha, luego el usuario al dar clic sobre el botón solicitar se le mostrarán los eventos y la cantidad de accedidos por cada evento, mostrándose en la parte inferior izquierda la cantidad total de accesos. Seguido de la opción solicitar se encontrará la opción exportar a pdf para poder imprimir el reporte.	Alta	
Prototipo				
Prototipo Mostrar cantidad de clientes que accedieron en cada evento del día por comedores.				

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRRECCIÓN DE ALIMENTACIÓN DE LA UCI

Banner												
Inicio	Reportes	Bienvenido: oldiaz Salir										
Cantidad de clientes que accedieron en cada evento del día por comedores.												
<input type="text" value="Fecha"/>												
<input type="text" value="Complejo"/> <input type="text" value="Comedor"/> <input type="text" value="Puerta"/>												
Solicitar Versión pdf												
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Eventos</th> <th style="text-align: right;">Cantidad de accesos</th> </tr> </thead> <tbody> <tr> <td style="text-align: left;">Desayuno</td> <td style="text-align: right;">150</td> </tr> <tr> <td style="text-align: left;">Almuerzo</td> <td style="text-align: right;">400</td> </tr> <tr> <td style="text-align: left;">Doble almuerzo</td> <td style="text-align: right;">600</td> </tr> <tr> <td style="text-align: left;">Total</td> <td style="text-align: right;">1730</td> </tr> </tbody> </table>			Eventos	Cantidad de accesos	Desayuno	150	Almuerzo	400	Doble almuerzo	600	Total	1730
Eventos	Cantidad de accesos											
Desayuno	150											
Almuerzo	400											
Doble almuerzo	600											
Total	1730											
Campos	Tipos de Datos	Reglas o Restricciones										
Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos. 											

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

2.6 Validación de requisitos

La validación es la actividad de la ingeniería de requisitos que permite demostrar que los requerimientos definidos en el sistema son los que realmente el cliente necesita y quiere. La validación verifica que los requisitos estén completos y sean consistentes y además que definan lo que el usuario espera del producto final (22).

Existen diversas técnicas para la validación de requisitos entre los más comunes se encuentran la revisión de requisitos, la realización de prototipos de interfaz y la generación de casos de prueba. Para realizar la validación de requisitos de la presente investigación se utilizaron la realización de prototipos y la realización de los casos de prueba.

Revisión de requisitos: consiste en una o varias reuniones en las que participa el analista, estas reuniones son planificadas con el objetivo de detectar fallas o inconsistencias en el documento de especificación de requisitos. Este método de validación de requisitos permite encontrar una gran cantidad de errores por lo que permite minimizar el tiempo de prueba. Como resultado final de las reuniones se obtiene un documento que contiene un listado de los errores encontrados.

Prototipos: no es más que una versión del producto final, tienen como objetivo comprobar la corrección y completitud de las especificaciones de requisito. Se realiza un esbozo plasmando la idea de lo que pudiera obtenerse como resultado final una vez ya implementados los requisitos identificados, estos prototipos propuestos por el analista pueden ser cambiados a medida que se vaya desarrollando el sistema con el objetivo de lograr una interfaz más amigable para el cliente.

Resultados de la revisión de requisitos

Durante el proceso de revisión de requerimientos se llevaron a cabo verificaciones en las especificaciones de requisito en las cuales se midieron los siguientes aspectos:

- **Verificaciones de validez:** se verifica que los requisitos cumplan con las necesidades del cliente.
- **Verificaciones de consistencia:** se analiza cada requisito verificando que no existan restricciones o descripciones contradictorias de las funciones del sistema.

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

- **Verificaciones de completitud:** el documento de requerimientos debe incluir requerimientos que definan todas las funciones y restricciones propuestas por el cliente con el fin de satisfacer sus necesidades.

Al concluir este proceso se determinaron inconsistencias como:

- Falta de funcionalidades necesarias en los prototipos de interfaz.
- Falta de ortografía en las especificaciones de requisito.
- Falta de concordancia entre el identificador del requisito en la especificación y la registrada en el documento de Evaluación de Requisitos.

2.7 Descripción de la arquitectura

Para seleccionar la arquitectura a utilizar por la aplicación se tuvo en cuenta que la misma sea la apropiada para las aplicaciones web, así como que sea adaptable a las tecnologías proporcionadas por la plataforma PHP, se consideró que el estilo arquitectónico que cumple con las condiciones planteadas es el cliente servidor haciendo uso del patrón modelo vista controlador. También se trató de lograr que todo el desarrollo esté soportado sobre software libre, para que la herramienta no herede ningún tipo de limitación en lo relativo a la soberanía tecnológica.

2.7.1 Arquitectura Cliente- Servidor

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, que le da respuesta. En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema (22).

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

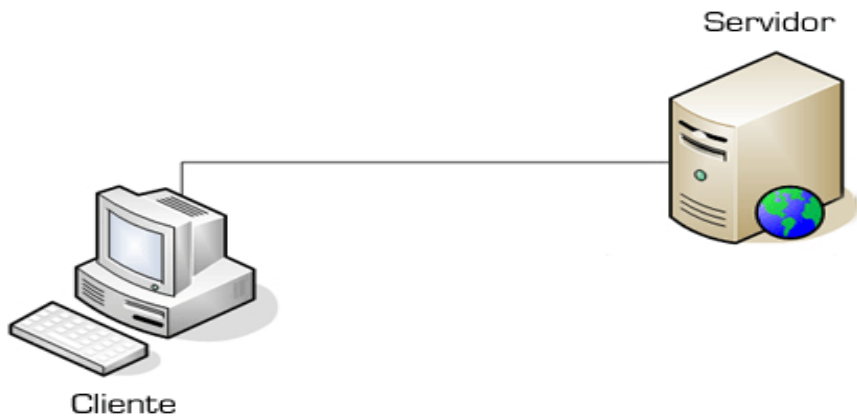


Figura 2: arquitectura cliente servidor.

Las características del cliente (remitente de una solicitud) son:

- Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación.
- Espera y recibe las respuestas del servidor.
- Por lo general, puede conectarse a varios servidores a la vez.
- Normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.

Las características del servidor (receptor de la solicitud) son:

- Al iniciarse esperan a que lleguen las solicitudes de los clientes, desempeñan entonces un papel pasivo en la comunicación.
- Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.
- Por lo general, aceptan conexiones desde un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).
- No es frecuente que interactúen directamente con los usuarios finales.

2.7.2 Patrón Arquitectónico

Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de *software*, que consta de subsistemas, sus responsabilidades e interrelaciones. No es una arquitectura como tal, sino un concepto que captura elementos esenciales de una arquitectura de *software*. El patrón de desarrollo Modelo-Vista-Controlador (MVC), separa los

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRRECCIÓN DE ALIMENTACIÓN DE LA UCI

datos de una aplicación, la interfaz de usuario y la lógica de negocio en tres capas conceptuales (23).

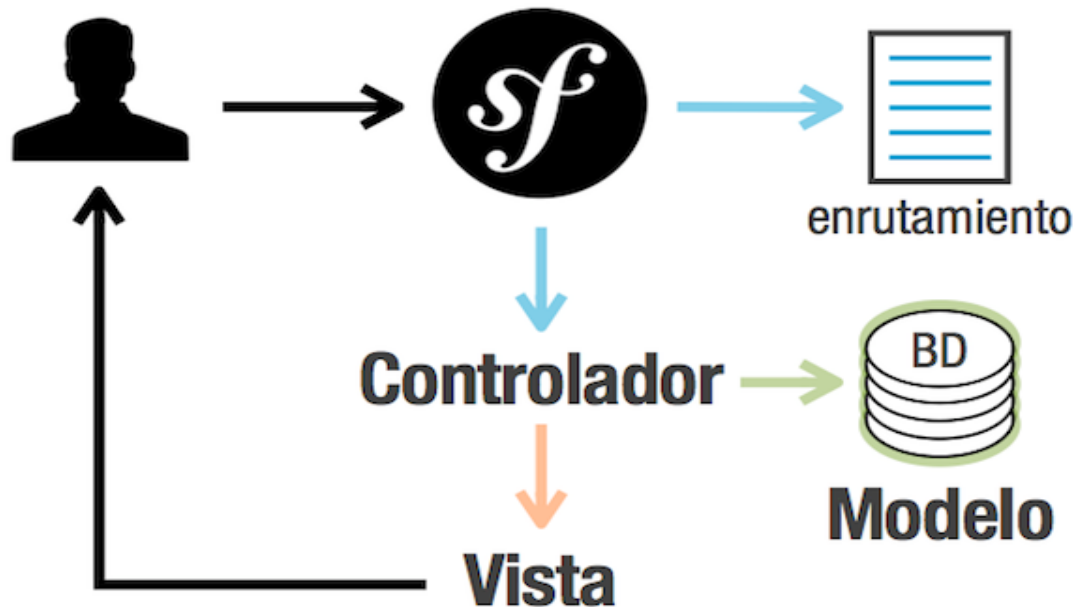


Figura 3: patrón Modelo Vista Controlador (MVC).

- **Modelo:** representa la información con la que trabaja la aplicación, o sea, su lógica de negocio. Gestiona los datos solicitados por el controlador. Esto se evidencia cuando se necesita mostrar la fuente de datos.
- **Vista:** convierte el modelo en una página web que facilita al usuario interactuar con ella. Contiene los elementos asociados a la presentación de los datos y la información visual de los elementos que conformarán el reporte, encontrándose el código asociado a las interfaces, que se encargan de visualizar la información que el controlador devuelve.
- **Controlador:** es el encargado de procesar las interacciones del usuario y ejecuta los cambios adecuados en el modelo o en la vista. La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista), lo que permite un mantenimiento más sencillo de las aplicaciones. El controlador es el encargado de aislar al modelo y a la vista de los detalles del protocolo usado para las peticiones (HTTP, consola de comandos,

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

email). El modelo se encarga de la abstracción de la lógica referida a los datos, lo que permite que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos que la aplicación utiliza. El controlador se encarga de la transferencia de información entre la vista y el modelo. Esto se aprecia cuando se envía desde la vista la configuración del reporte al controlador, este lo captura y se lo manda al modelo para que lo almacene.

El primer paso en el ciclo de vida empieza cuando el usuario hace una solicitud al controlador con información sobre lo que él desea realizar. Entonces el controlador decide a quién debe delegar la tarea y es aquí donde el modelo empieza su trabajo. En esta etapa, el modelo se encarga de realizar operaciones sobre la información que maneja para cumplir con lo que le solicita el controlador. Luego de terminada su labor, le regresa al controlador la información resultante de sus operaciones, el cual a su vez redirige a la vista la cual se encarga de transformar los datos en información visualmente entendible para el usuario.

En el Sistema de Reportes el patrón Modelo Vista Controlador (MVC) se aplica de la siguiente manera:

1. El cliente se conecta con el sistema y este consulta la tabla de enrutamiento, que es donde se encuentran todas las rutas de los controladores que tiene el sistema.
2. A través de esta ruta se conecta con el controlador.
3. El controlador va al modelo y obtiene los datos que este a su vez se conecta con la base de datos para obtener la información.
4. Esta información es mostrada al usuario mediante la vista.

2.8 Patrones de diseño

Un patrón es un esquema o micro-arquitectura que supone una solución a problemas, cada patrón es adecuado para ser adaptado a un cierto tipo de problema. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias (24).

2.8.1 Patrones de diseño GRASP

Los patrones GRASP representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones. GRASP es el acrónimo para *General Responsibility Assignment Software Patterns* (Patrones Generales de Software para Asignar Responsabilidades) (24).

2.8.1.2 Patrones básicos de asignación de responsabilidades

Experto: se encarga de asignar una responsabilidad al experto en información, o sea, aquella clase que cuenta con la información necesaria para cumplir la responsabilidad.

Indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtendrá un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento).

Beneficios: se mantiene el encapsulamiento, los objetos utilizan su propia información para llevar a cabo sus tareas. Se distribuye el comportamiento entre las clases que contienen la información requerida. Son más fáciles de entender y mantener.

En Symfony2 este patrón es utilizado en la inclusión de Doctrine para mapear la base de datos. Se utiliza para crear una capa de abstracción en el modelo, encapsular toda la lógica de los datos y generar las clases en todas las funcionalidades comunes de las entidades, las clases de abstracción de datos poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla que representan.

Ejemplo: la aplicación de este patrón se evidencia en la clase Usuario la cual tiene la información necesaria para gestionar los usuarios contenidos en la base de datos a través de su repositorio correspondiente.

Bajo acoplamiento: es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. En Symfony2, la clase *Default_Controller* hereda solamente de *Controller*, que es una clase estable en cuanto a su implementación, garantizando un bajo acoplamiento entre las clases.

Ejemplo: la utilización de este patrón se evidencia en la poca dependencia entre las clases del sistema, de tal forma que si ocurrieran cambios en alguna de las clases no se afectan otras. Solo existe la dependencia entre los controladores *Default_Controller* de los *bundles Alimentacion_Bundle* y *Ldap_Bundle* para la realización de la autenticación ya que esta cuenta de dos partes, autenticación por el protocolo *Ldap* y la existencia del usuario en la lista de usuarios permitidos.

Alta cohesión: la Alta Cohesión es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño. Se da una alta cohesión funcional cuando los elementos de un componente (clase, por ejemplo) colaboran para producir algún comportamiento bien definido.

Symfony2 permite asignar responsabilidades a una clase con una alta cohesión pues plantea que debe existir una acción para cada una de las plantillas que producen la vista. Además la clase *Controller* colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder las propiedades, proporcionando que el software sea flexible frente a grandes cambios.

Ejemplo: esto se evidencia en la clase *Usuario* perteneciente al modelo la cual cuenta con la información estrechamente relacionada con los usuarios del sistema.

Controlador: es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto para aumentar la reutilización de código y a la vez tener un mayor control.

Ejemplo: el empleo de este patrón se evidencia cuando el formulario de autenticación envía los datos a la clase *DefaultController* y esta se encarga de enviarlos a la clase *Usuario* para realizar el procesamiento de autenticación a partir de la base de datos.

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

Creador: este patrón es el responsable de asignarle a la clase B la responsabilidad de crear una instancia de clase A. B es un creador de los objetos A. Este patrón es utilizado en las clases controladoras donde se encuentran las acciones definidas para el sistema y se ejecutan en cada una de ellas. En dichas acciones se crean los objetos de las clases que representan las entidades, lo que evidencia que las clases controladoras son creadoras de dichas entidades. En *Symfony* hay clases que aplican este patrón, un ejemplo de esto es en las clases “*action*” cuando se desea usar una instancia de una clase del modelo.

La nueva instancia deberá ser creada por la clase que:

- Tiene la información necesaria para realizar la creación del objeto.
- Usa directamente las instancias creadas del objeto.
- Almacena o maneja varias instancias de la clase.
- Contiene o agrega la clase.

Ejemplo: el uso de este patrón está evidenciado en la clase *DefaultController* específicamente en la función *loginHomeAction()* cuando se crea la instancia de la clase *UsernamePasswordToken* a partir de los datos de usuario contenidos en ella.

2.8.2 Patrones de diseño GoF

Los patrones *GoF* (Gang of Four, en español Pandilla de los Cuatro, formada por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides) se clasifican en 3 categorías basadas en su propósito: creacionales, estructurales y de comportamiento (26).

- **Fábrica Abstracta:** también conocido como, permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí haciendo transparente el tipo de familia concreta que se esté usando. Cuando el *framework* necesita por ejemplo crear un nuevo objeto para una petición, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea. Como la definición por defecto de la factoría para las peticiones es *getRequest()*, *Symfony2* crea un objeto de esta clase para tratar con las peticiones.
- **Decorator:** también conocido como Decorador añade funcionalidad a una clase de una forma dinámica y transparente. En *Symfony2* los archivos *base.html.twig* y *layout.html.twig*

II | CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

almacenan el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de las plantillas se integra en estos archivos, o si se mira desde el otro punto de vista, el archivo base y el *layout* decoran el contenido de las plantillas.

2.9 Descripción del diagrama de despliegue

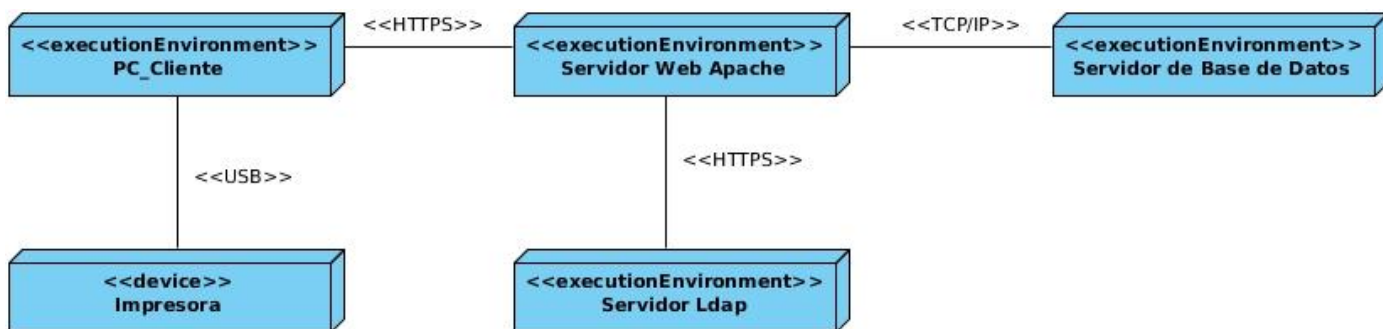


Figura 4: diagrama de despliegue.

El diagrama de despliegue representado muestra la siguiente distribución:

- **PC cliente:** en este nodo es donde procesan todas las interfaces de usuario que han sido previamente implementadas. Se encuentra alojado el navegador web (Mozilla Firefox, Internet Explorer, Opera, entre otros) mediante el cual el usuario accederá a las interfaces del sistema utilizando el protocolo HTTPS.
- **Servidor web:** ordenador en que se encuentra el servidor web Apache, este será el lugar en que se gestione todo el contenido de la aplicación. El mismo establecerá comunicación con los ordenadores clientes mediante protocolo HTTPS y con el servidor de base de datos por medio del protocolo TCP/IP.
- **Servidor de base de datos:** es donde se encuentra toda la información estructurada. Ordenador en que se encuentra el gestor de base de datos PostgreSQL capaz de mantener persistente la información generada y a utilizar.

| CAPÍTULO II: PROPUESTA DE SOLUCIÓN DEL SISTEMA DE REPORTES PARA LA DIRECCIÓN DE ALIMENTACIÓN DE LA UCI

2.10 Conclusiones

Este capítulo posibilitó una mejor comprensión de las características del sistema debido a que se explican una serie de conceptos y términos asociados al sistema mediante un modelo de dominio permitiendo así identificar los requerimientos de la propuesta de solución, los cuales se describen mediante artefactos que plantean las metas y condiciones que el sistema debe cumplir obteniéndose así el análisis y diseño del sistema de reportes, con el diseño de los prototipos de interfaz se dio una versión inicial de como pudiera quedar diseñado el sistema. Con la culminación de este capítulo quedaron establecidos los cimientos para pasar al próximo paso, que sería, la implementación del sistema.

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

Capítulo 3: Implementación y prueba de la solución propuesta

3.1 Introducción

Una vez concluido el proceso de análisis y diseño del sistema de alimentación están creadas todas las condiciones para generar el código fuente, obtener una solución funcional y validar su funcionamiento. En este capítulo se desarrolla la solución propuesta siguiendo un estándar de codificación y guiadas por las especificaciones de requisitos descritas en el capítulo anterior y por último se describen las pruebas de *software* aplicadas y sus resultados.

3.2 Estándares de codificación

3.2.1 Indentación, llaves de apertura y cierre, y tamaño de las líneas

Usar una indentación sin tabulaciones, con un equivalente a 4 espacios. El uso de las llaves “{}” es seguido del nombre del método. La longitud de las líneas de código es aproximadamente de 75-80 caracteres. Para mantener la legibilidad del código.

Ejemplo:

```
public function loginHomeAction(){
    return $this->
    render('AlimentacionBundle:Default:inicio_reportes.html.twig');
}
```

Figura 5: ejemplo de indentación, llaves de apertura y cierre, y tamaño de las líneas.

3.2.2 Convención de nomenclatura

Variables: se rigen por la nomenclatura *camelCase*. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula.

Ejemplo:

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

```
//Esta es una clase entidad que gestiona la tabla usuario en la base de datos
namespace CENIA\AlimentacionBundle\Entity;

use Doctrine\ORM\Mapping as ORM;

/**
 * CENIA\AlimentacionBundle\Entity\Usuario
 *
 * @ORM\Table()
 * @ORM\Entity(repositoryClass="CENIA\AlimentacionBundle\Entity\UsuarioRepository")
 */
class Usuario
{
    /**
     * @var integer $id
     *
     * @ORM\Column(name="id", type="integer")
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="AUTO")
     */
    private $id;
```

Figura 6: ejemplo de convención de nomenclaturas en variables.

Clases: siempre comienzan con mayúscula, en caso de nombre compuesto la primera letra con mayúscula.

Ejemplo:

```
//Esta es una clase entidad que gestiona la tabla usuario en la base de datos
namespace CENIA\AlimentacionBundle\Entity;

use Doctrine\ORM\Mapping as ORM;

/**
 * CENIA\AlimentacionBundle\Entity\Usuario
 *
 * @ORM\Table()
 * @ORM\Entity(repositoryClass="CENIA\AlimentacionBundle\Entity\UsuarioRepository")
 */
class Usuario
{
    /**
     * @var integer $id
     *
     * @ORM\Column(name="id", type="integer")
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="AUTO")
     */
    private $id;
```

Figura 7: ejemplo de convención de nomenclaturas en clases.

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

Funciones: se rigen por la nomenclatura *camelCase*. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula terminando con la palabra *Action*. Los parámetros son separados por espacio luego de la coma que los separa.

Ejemplo:

```
public function loginHomeAction(){
    return $this->
        render('AlimentacionBundle:Default:inicio_reportes.html.twig');
}
```

Figura 8: ejemplo de convención de nomenclaturas en funciones.

Ficheros: todo siempre en minúscula y en caso de nombres compuestos se usa el carácter Subrayado”_”.

Vistas: intuitivo y relacionado con el formulario y/o vista que representa.

Modelos: con el mismo nombre de la clase que representa.

Librerías: con el mismo nombre de la clase que representa.

Controladoras: con el mismo nombre de la clase que representa, terminando con la palabra *Controller*.

Estructuras de control

Se incluye *if*, *for*, *foreach*, *while*, *switch*, entre las estructuras de control y los paréntesis debe de existir un espacio. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales. Esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos.

Ejemplos:

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

```
if (defined('INTL_ICU_VERSION')) {  
    $version = INTL_ICU_VERSION;  
} else {  
    $reflector = new \ReflectionExtension('intl');  
  
    ob_start();  
    $reflector->info();  
    $output = strip_tags(ob_get_clean());  
  
    preg_match('/^ICU version +(?:=> )?(.*)$/m', $output, $matches);  
    $version = $matches[1];  
}
```

Figura 9: ejemplo de estructuras de control.

3.3 Estrategias de prueba

Las pruebas del software son un elemento crítico para la garantía de calidad del producto y representa una revisión final de las especificaciones, del diseño y de la codificación (21).

“Grady Booch define un caso de prueba como: un conjunto de entradas de prueba, condiciones de ejecución y resultados esperados, desarrollados para un objetivo concreto, tal como probar un camino concreto a través de un caso de uso o comprobar que se cumple un requisito determinado” (25).

Las pruebas de software son un conjunto de actividades que se pueden planificar por adelantado y llevar a cabo sistemáticamente. Por esta razón, se debe definir en el proceso de la ingeniería del software una plantilla para las pruebas del software: un conjunto de pasos en los que se puede situar los métodos específicos de diseño de casos de prueba. Las pruebas constituyen el último bastión desde el que se puede evaluar la calidad y de forma más pragmática descubrir los errores. La prueba del software es un elemento de un tema más amplio que, a menudo, es conocido como verificación y validación (21).

3.3.1 Estrategia de prueba a seguir para la validación del Sistema de Reportes

Existen cuatro niveles o estrategias de pruebas según lo define Pressman, el nivel de unidad, el nivel de integración, el nivel de validación y el nivel de sistema. En el Sistema de Reportes para

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

la Dirección de Alimentación se define como estrategia o nivel de prueba a usar el nivel de unidad y el nivel de integración.

Nivel de unidad

La prueba de unidad centra el proceso de verificación en la menor unidad del diseño de software: el componente software o módulo. La prueba se centra en cada módulo individualmente asegurando que funcionen adecuadamente como una unidad.

Nivel de integración

La prueba de integración se centra en probar los componentes del sistema combinado. Se centra en descubrir errores de las especificaciones de las clases. (21).

Para ambos niveles se aplicarán pruebas de caja negra guiadas por la técnica de prueba partición de equivalencia.

Métodos de pruebas basados en caja negra.

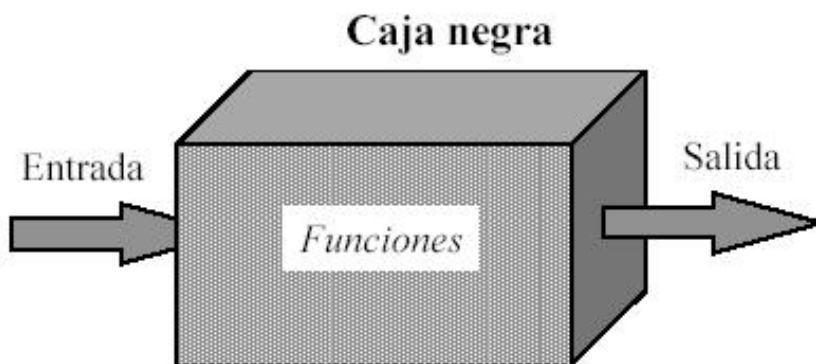


Figura 10: método de prueba caja negra.

- Técnica de la partición de equivalencia

La partición equivalente es una técnica de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores, que de otro modo requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número de

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

casos de prueba que hay que desarrollar. El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos o inválidos para condiciones de entrada. Las condiciones de entrada son un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica.

Las clases de equivalencia se pueden definir a través de las siguientes condiciones (21).

- Si una condición de entrada especifica un rango, se define una clase de equivalencia válida y dos inválidas.
- Si una condición de entrada requiere un valor específico, se define una clase de equivalencia válida y dos inválidas.
- Si una condición de entrada especifica un miembro de un conjunto, se define una clase de equivalencia válida y una inválida.
- Si una condición de entrada es lógica, se define una clase válida y una inválida.

Particiones de equivalencias

- Identificar las clases de equivalencia válidas e inválidas para cada condición de entrada.
- Escribir un nuevo caso de prueba para cubrir tantas clases de equivalencia válidas como sea posible.
- Escribir un nuevo caso de prueba que cubra una y solo una clase de equivalencia inválida.

Especificaciones de clases válidas e inválidas

Para un Rango → 3 clases de equivalencia: por debajo, en el rango y por encima del rango.

Para un valor concreto → 3 clases de equivalencia: por debajo del valor, el valor en concreto y por encima del valor.

Para un valor dentro de un conjunto → 2 clases de equivalencia: en el conjunto o fuera de él.

Para un booleano → 2 clases equivalentes: verdadero o falso.

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

Tabla 5: estrategia de prueba.

NIVEL	OBJETIVO	MÉTODO DE PRUEBA	TÉCNICA DE PRUEBA
Unidad	Probar que todas las funcionalidades del sistema funcionen correctamente.	Caja Negra	Partición de equivalencia.
Integración	Detectar errores de interfaces y relaciones entre componentes, Verificar el correcto funcionamiento de los componentes del sistema integrados como un todo.	Caja Negra	Partición de equivalencia.

Para comprobar la calidad del producto se hace una descripción de casos de pruebas para cada requisito funcional utilizando el método de caja negra con el objetivo de demostrar que el sistema de reportes cumple con los requisitos funcionales definidos. En la realización de estos casos de pruebas la técnica a emplear será partición de equivalencia, pues la misma permite analizar los valores válidos y no válidos para las condiciones de entrada existentes en el software (21).

3.4 Casos de prueba

A continuación se describen los casos de pruebas realizados para los requisitos funcionales: autenticar usuario en el sistema, total de grupos por puertas.

Tabla 6: descripción del caso de prueba para el requisito autenticar usuario.

RF_ Autenticar usuario en el sistema		
Entrada	Condición de ejecución	Respuesta del sistema
El usuario se registra en el sistema llenando los siguientes campos: Usuario: "ggonzaleze" Contraseña:*****	Los campos que se introducen deben ser correctos.	Permite la entrada al sistema.
Usuario: "ggonzaleze" Contraseña: "***** "	Debe ser un usuario permitido.	No permite la entrada al sistema.
Usuario: "ggonzaleze" Contraseña: " "	No se pueden dejar campos vacíos.	No permite la entrada al sistema.

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

Tabla 7: descripción del caso de prueba para el requisito total de grupos por puertas.

RF_ Total de grupos por puertas		
Entrada	Condición de ejecución	Respuesta del sistema
El usuario una vez autenticado escoge del menú Reportes la funcionalidad: total de grupos por puertas y llena los siguientes campos Complejo: complejo 1 Luego de la entrada de datos da clic en la opción solicitar.	El usuario tiene que estar autenticado en el sistema.	El sistema debe mostrar una tabla con el nombre del grupo, el id del grupo, el nombre de la puerta y el id de la puerta, mostrando también el total de grupos por puerta además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.
El usuario una vez autenticado escoge del menú Reportes la funcionalidad: total de grupos por puertas y llena los siguientes campos Complejo: "" Luego de la entrada de datos da clic en la opción solicitar.	No puede haber campos vacíos.	El sistema muestra un mensaje indicando que no puede haber campos vacíos.

Tabla 8: descripción del caso de prueba para el requisito cantidad de clientes que no han recibido asignación de servicios.

RF_ Cantidad de clientes que no han recibido asignación de servicios.		
Entrada	Condición de ejecución	Respuesta del sistema
El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes que no han recibido asignación de servicios.	El usuario tiene que estar autenticado en el sistema.	El sistema debe mostrar una tabla con el nombre de la persona, su número de solapín y la causa de exclusión además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.
El usuario no da clic en la funcionalidad cantidad de clientes que no han recibido asignación de servicios.		No se muestra ningún dato.

3.5 Resultado de las pruebas realizadas

La realización de las pruebas unitarias, pruebas de integración y los casos de prueba de caja negra estos últimos utilizando la técnica de partición de equivalencia permitió comprobar las funcionalidades del módulo y detectar una serie de no conformidades, las cuales fueron erradicadas a medida que se fueron encontrando en cada una de las iteraciones que se hicieron.

| CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN PROPUESTA

Estas pruebas realizadas permitieron comprobar que el Sistema de Reportes cumple con todas las especificaciones que se trazaron.

Tabla 9: resultados de las pruebas.

Iteración	NC Detectadas	Asociado a	NC Resueltas
1	12	Errores de interfaz y validación	12
2	9	Errores ortográficos y de validación.	9
3	0	-	0

3.6 Conclusiones del capítulo

En este capítulo se diseñaron los reportes en el iReport dando como resultado final un Sistema de Reportes para la Dirección de Alimentación de la Universidad de las Ciencias Informáticas. La realización de pruebas unitarias, de integración y los casos de prueba de caja negra a la solución permitió corregir errores detectados logrando la correcta implementación de los requisitos funcionales.

Conclusiones generales

Al terminar la investigación se llegaron a las siguientes conclusiones:

- El análisis de los elementos teóricos y técnicos que sustentaron la investigación permitió conocer y aprender sobre las herramientas, tecnologías y metodología a utilizar para el desarrollo del sistema.
- A partir de las funcionalidades descritas se realizó el análisis y diseño del sistema de reportes, proporcionando el punto de partida para las actividades de implementación.
- Se desarrolló el Sistema de Reportes permitiendo que la Dirección de Alimentación realice una correcta planificación de los recursos de alimentación.
- Las pruebas ejecutadas a la solución detectaron errores que al corregirlos, se garantizó el cumplimiento de los requisitos funcionales.

Recomendaciones

Al concluir la investigación se proponen como recomendaciones:

- Poder integrar en algún momento el Sistema de Reportes al Sistema de Asignación que utiliza la Dirección de Alimentación.
- Aplicarle una limpieza de datos a la base de datos utilizada por el Sistema de Reportes con el objetivo de aplicarle como técnica de descubrimiento de conocimiento en bases de datos la minería de datos, para poder clasificar, agrupar o analizar tendencias de algún reporte.

Glosario de términos

Document Object Model: es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos.

Framework: plataformas o herramientas del mundo de la informática que le proveen a los programadores un grupo de facilidades en el ámbito para la cual han sido creadas.

HTTP HyperText Transfer Protocol: es el protocolo de transferencia de hipertexto, es el método más común de intercambio de información en la World Wide Web, el método mediante el cual se transfieren las páginas web a un ordenador.

Open Source: es el término con el que se conoce al software distribuido y desarrollado libremente. El código abierto tiene un punto de vista más orientado a los beneficios prácticos de poder acceder al código.

Referencias bibliográficas

1. Definicion.De. [En línea] 2008. <http://definicion.de/reporte/>.
2. **Jenny Infante Frómeta, Yasmany Hernández Hernández, Yasmany Molina Díaz.** *Sistema de Gestión de Reportes Dinámicos*. La Habana : s.n., 2009.
3. CENACAD. *CENACAD*. [En línea] CISE, 2006. [Citado el: 6 de diciembre de 2012.] <http://www.cenacad.espol.edu.ec/>.
4. info@tltas. *info@tltas*. [En línea] Inder, 2008. [Citado el: 6 de diciembre de 2012.] <http://www.inder.cu/FichaAtleta/home/default.aspx>.
5. **Castro, Ing. Eileén Llano.** *PROPUESTA PARA LA INTEGRACIÓN DE PRÁCTICAS DE LAS METODOLOGÍAS XP Y SCRUM CON EL PROCESO DE ADMINISTRACIÓN DE REQUISITOS DEL NIVEL 2 DE CMMI*. 2011.
6. **Marley, Jimi.** ¿Por qué elegir PHP? [En línea] 2012. http://www.programacion.com/articulo/por_que_elegir_php_143.
7. **Pérez, Javier Eguíluz.** ¿Qué es JavaScript? [En línea] 2012. <http://www.librosweb.es/javascript/index.html>.
8. —. CSS avanzado. [En línea] 2012. <http://www.librosweb.es/css/capitulo1.html>.
9. **Schmuller, Joseph.** UML en 24 Horas. [En línea] 2012. <http://www.scribd.com/doc/38392190/UML-en-24-Horas>.
10. **Oracle Corporation and/or its affiliates.** NetBeans. [En línea] 2012. http://netbeans.org/community/releases/68/index_es.html.
11. **Free Download Manager.ORG.** Visual Paradigm para UML. [En línea] 2012. http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5.
12. **Pencil Project working group.** Pencil proyect. [En línea] 2012. <http://pencil.evolus.vn/en-US/Home.aspx>.
13. Diseñador Ireport. *Diseñador Ireport*. [En línea] JasperSoft, 2000. [Citado el: 14 de enero de 2013.] <http://community.jaspersoft.com/project/ireport-designer>.
14. **Izquierdo, Susana.** JASPER REPORTS. [En línea] 2012. <http://www.abartiateam.com/jasperreports>.
15. **Eguiluz, Javier.** *Desarrollo web ágil con Symfony2*. La Habana : s.n., 2012.

| REFERENCIA BIBLIOGRAFICA

16. slideShare. *slideShare*. [En línea] 2010. [Citado el: 14 de 2 de 2013.] <http://www.slideshare.net/josegregoriob/servidor-web-8451426>.
17. **Ciberaula**. Una Introducción a APACHE. [En línea] 2012. http://linux.ciberaula.com/articulo/linux_apache_intro.
18. Guia_ubuntu. *Guia_ubuntu*. [En línea] 1 de 10 de 2012. <http://www.guia-ubuntu.com/?title=PostgreSQL>.
19. postgresql. *postgresql*. [En línea] 2008. [Citado el: 14 de 2 de 2013.] http://www.postgresql.org.es/sobre_postgresql.
20. **SYNERGIX.Tecnología y synergix**. Modelo de dominio. [En línea] 2012. <http://synergix.wordpress.com>.
21. **Pressman, Roger S.** *Ingeniería del software. Un enfoque práctico*. La Habana : Félix Varela, 2005.
22. Scribd. *Scribd*. [En línea] 2003. [Citado el: 14 de 2 de 2013.] <http://es.scribd.com/doc/43836633/Diseno-de-Software-en-Arquitectura-Cliente-Servidor>.
23. Patron_Modelo_Vista_Controlador. *Patron_Modelo_Vista_Controlador*. [En línea] 2003. [Citado el: 14 de 2 de 2013.] <http://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador/>.
24. **LARMAN, CRAIG**. *UML Y PATRONES. Introducción al análisis y diseño orientado a objetos*. . PRENTICE HAL : 17, 1999. 970-17-0261-1.
25. **BOOCH, GRADY**. *El Proceso Unificado de Desarrollo de Software*. s.l. : Felix Varela, 2004.

Bibliografía

1. Definicion.De. [En línea] 2008. <http://definicion.de/reporte/>.
2. **Jenny Infante Frómata, Yasmany Hernández Hernández, Yasmany Molina Díaz.** *Sistema de Gestión de Reportes Dinámicos*. La Habana : s.n., 2009.
3. CENACAD. *CENACAD*. [En línea] CISE, 2006. [Citado el: 6 de diciembre de 2012.] <http://www.cenacad.espol.edu.ec/>.
4. info@tltas. *info@tltas*. [En línea] Inder, 2008. [Citado el: 6 de diciembre de 2012.] <http://www.inder.cu/FichaAtleta/home/default.aspx>.
5. **Castro, Ing. Eileén Llano.** *PROPUESTA PARA LA INTEGRACIÓN DE PRÁCTICAS DE LAS METODOLOGÍAS XP Y SCRUM CON EL PROCESO DE ADMINISTRACIÓN DE REQUISITOS DEL NIVEL 2 DE CMMI*. 2011.
6. **Marley, Jimi.** ¿Por qué elegir PHP? [En línea] 2012. http://www.programacion.com/articulo/por_que_elegir_php_143.
7. **Pérez, Javier Eguíluz.** ¿Qué es JavaScript? [En línea] 2012. <http://www.librosweb.es/javascript/index.html>.
8. —. CSS avanzado. [En línea] 2012. <http://www.librosweb.es/css/capitulo1.html>.
9. **Schmuller, Joseph.** UML en 24 Horas. [En línea] 2012. <http://www.scribd.com/doc/38392190/UML-en-24-Horas>.
10. **Oracle Corporation and/or its affiliates.** NetBeans. [En línea] 2012. http://netbeans.org/community/releases/68/index_es.html.
11. **Free Download Manager.ORG.** Visual Paradigm para UML. [En línea] 2012. http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5.
12. **Pencil Project working group.** Pencil project. [En línea] 2012. <http://pencil.evolus.vn/en-US/Home.aspx>.
13. Diseñador Ireport. *Diseñador Ireport*. [En línea] JasperSoft, 2000. [Citado el: 14 de enero de 2013.] <http://community.jaspersoft.com/project/ireport-designer>.
14. **Izquierdo, Susana.** JASPER REPORTS. [En línea] 2012. <http://www.abartiateam.com/jasperreports>.
15. **Eguiluz, Javier.** *Desarrollo web ágil con Symfony2*. La Habana : s.n., 2012.

16. slideShare. *slideShare*. [En línea] 2010. [Citado el: 14 de 2 de 2013.] <http://www.slideshare.net/josegregoriob/servidor-web-8451426>.
17. **Ciberaula**. Una Introducción a APACHE. [En línea] 2012. http://linux.ciberaula.com/articulo/linux_apache_intro.
18. Guia_ubuntu. *Guia_ubuntu*. [En línea] 1 de 10 de 2012. <http://www.guia-ubuntu.com/?title=PostgreSQL>.
19. postgresql. *postgresql*. [En línea] 2008. [Citado el: 14 de 2 de 2013.] http://www.postgresql.org.es/sobre_postgresql.
20. **SYNERGIX.Tecnología y synergix**. Modelo de dominio. [En línea] 2012. <http://synergix.wordpress.com>.
21. **Pressman, Roger S.** *Ingeniería del software. Un enfoque práctico*. La Habana : Félix Varela, 2005.
22. Scribd. *Scribd*. [En línea] 2003. [Citado el: 14 de 2 de 2013.] <http://es.scribd.com/doc/43836633/Diseno-de-Software-en-Arquitectura-Cliente-Servidor>.
23. Patron_Modelo_Vista_Controlador. *Patron_Modelo_Vista_Controlador*. [En línea] 2003. [Citado el: 14 de 2 de 2013.] <http://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador/>.
24. **LARMAN, CRAIG**. *UML Y PATRONES. Introducción al análisis y diseño orientado a objetos*. . PRENTICE HAL : 17, 1999. 970-17-0261-1.
25. **BOOCH, GRADY**. *El Proceso Unificado de Desarrollo de Software*. s.l. : Felix Varela, 2004.
26. **Jacobson, Ivar**. *El proceso unificado de desarrollo de software*. La Habana: Félix Varela, 2004. ISBN 0-201-57169-2.
27. **BOOCH, GRADY**. *El Proceso Unificado de Desarrollo de Software*. La Habana, Félix Varela, 2004. [Consultado el: 19 de Enero de 2013].
28. **Pressman Roger, S.** *Ingeniería del Software. Un enfoque práctico*. Quinta edición. La Habana: McGraw-Hill, 2002.
29. ISSI CAMY, LÁZARO. *JavaScript*. Madrid: ANAYA MULTIMEDIA, 2002. ISBN: 84-415-1384-8

30. **SOMMERVILLE, IAN.** Ingeniería del software. Séptima Edición. España, Madrid. Pearson Educación, SA., 2005. 119 pp. ISBN: 978 0 321 31379 9.

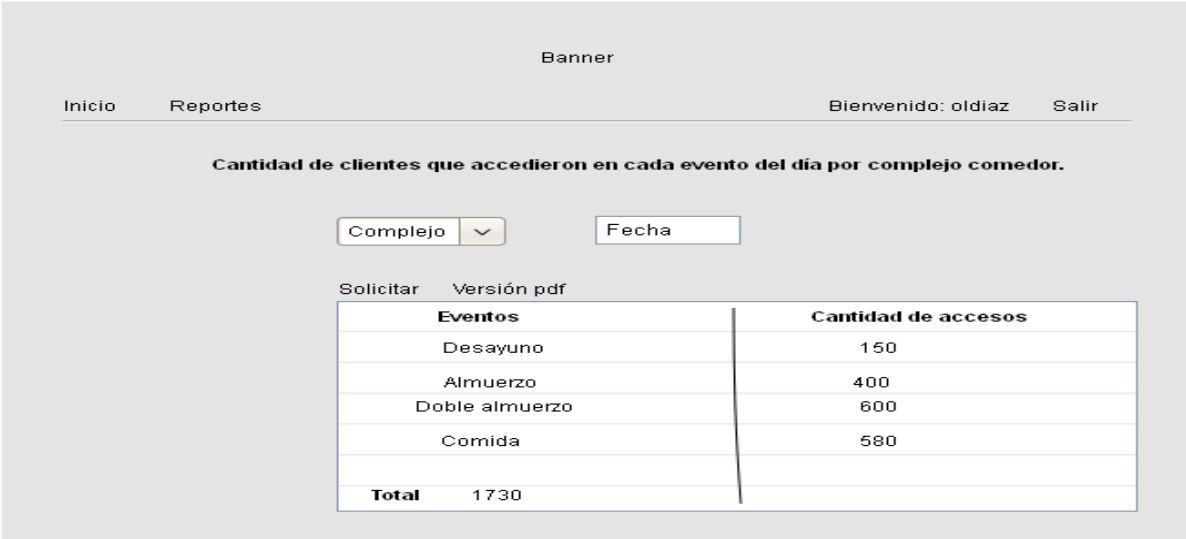
31. Definición de reporte - Qué es, Significado y Concepto [En Línea] 2012
<http://definicion.de/reporte/>

Anexos

Anexo A

Especificaciones de requisito

Tabla 10: especificación de requisito: mostrar cantidad de clientes que accedieron por complejo.

	Nombre	Descripción	Complejidad	Prioridad para cliente												
[RF8.]	Mostrar cantidad de clientes que accedieron por complejo.	Primeramente se muestra una lista despegable donde el usuario podrá elegir el complejo y la fecha, luego el usuario al dar clic sobre el botón solicitar se le mostrarán los eventos y la cantidad de accedidos por cada evento, mostrándose en la parte inferior izquierda la cantidad total de accesos. Seguido de la opción solicitar se encontrará la opción exportar a pdf para poder imprimir el reporte.	Alta													
Prototipo																
<p style="text-align: center;">Prototipo Mostrar cantidad de clientes que accedieron por complejo.</p>  <p style="text-align: center;">Banner</p> <p style="text-align: center;">Inicio Reportes Bienvenido: oldiaz Salir</p> <p style="text-align: center;">Cantidad de clientes que accedieron en cada evento del día por complejo comedor.</p> <p style="text-align: center;">Complejo <input type="text"/> Fecha <input type="text"/></p> <p style="text-align: center;">Solicitar Versión pdf</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Eventos</th> <th>Cantidad de accesos</th> </tr> </thead> <tbody> <tr> <td>Desayuno</td> <td>150</td> </tr> <tr> <td>Almuerzo</td> <td>400</td> </tr> <tr> <td>Doble almuerzo</td> <td>600</td> </tr> <tr> <td>Comida</td> <td>580</td> </tr> <tr> <td>Total</td> <td>1730</td> </tr> </tbody> </table>					Eventos	Cantidad de accesos	Desayuno	150	Almuerzo	400	Doble almuerzo	600	Comida	580	Total	1730
Eventos	Cantidad de accesos															
Desayuno	150															
Almuerzo	400															
Doble almuerzo	600															
Comida	580															
Total	1730															

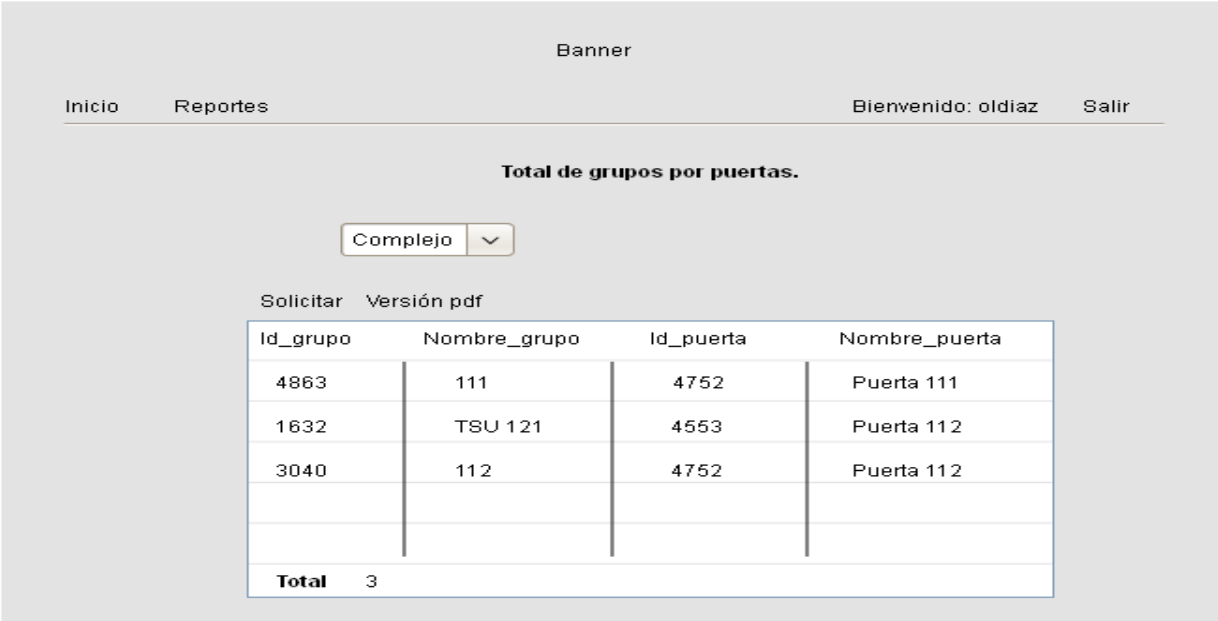
	Campos	Tipos de Datos	Reglas o Restricciones
	Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos. 	

Tabla 11: especificación de requisito: mostrar cantidad de clientes que no han recibido asignación de servicios.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF2.]	Mostrar cantidad de clientes que no han recibido asignación de servicios.	Una vez que el usuario de clic sobre la funcionalidad se mostrará el nombre de la persona, el solapín y la causa de exclusión. En la esquina superior izquierda aparecerá la opción de exportar a pdf para imprimir el reporte.	Alta	
Prototipo				
Prototipo Mostrar cantidad de clientes que no han recibido asignación de servicios.				

Banner																				
Inicio	Reportes	Bienvenido: oldiaz Salir																		
Cientes que no han recibido asignación de servicios.																				
Versión pdf																				
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 33%;">Nombre_Persona</th> <th style="width: 33%;">Solapín</th> <th style="width: 33%;">Causa de exclusión</th> </tr> </thead> <tbody> <tr> <td>Yaima</td> <td>EH04848</td> <td>Defectuoso</td> </tr> <tr> <td>Yoandis</td> <td>EH04850</td> <td>Defectuoso</td> </tr> <tr> <td>Geidy</td> <td>EH04550</td> <td>Defectuoso</td> </tr> <tr> <td> </td> <td> </td> <td> </td> </tr> <tr> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table>			Nombre_Persona	Solapín	Causa de exclusión	Yaima	EH04848	Defectuoso	Yoandis	EH04850	Defectuoso	Geidy	EH04550	Defectuoso						
Nombre_Persona	Solapín	Causa de exclusión																		
Yaima	EH04848	Defectuoso																		
Yoandis	EH04850	Defectuoso																		
Geidy	EH04550	Defectuoso																		
Campos	Tipos de Datos	Reglas o Restricciones																		
Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos. 																			

Tabla 12: especificación de requisito: mostrar total de grupos por puerta.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente																				
[RF3]	Mostrar total de grupos por puerta.	Una vez que el usuario de clic sobre la funcionalidad se mostrará una lista despegable donde el usuario debe elegir el complejo, luego al dar clic en la opción solicitar se mostrará el id del grupo, el nombre del grupo, el id de la puerta y el nombre de la puerta. En segundo a solicitar estará la opción de exportar a pdf para poder imprimir el reporte.	Media																					
Prototipo																								
<p style="text-align: center;">Prototipo total de grupos por puerta.</p>  <p>The screenshot shows a web interface with a banner at the top, navigation links for 'Inicio' and 'Reportes', and a user greeting 'Bienvenido: oldiaz' with a 'Salir' link. The main content area displays the title 'Total de grupos por puertas.' followed by a dropdown menu set to 'Complejo'. Below this are two links: 'Solicitar' and 'Versión pdf'. A table with four columns (Id_grupo, Nombre_grupo, Id_puerta, Nombre_puerta) contains three rows of data. At the bottom of the table, a 'Total' row shows a count of 3.</p> <table border="1" data-bbox="532 1522 1334 1795"> <thead> <tr> <th>Id_grupo</th> <th>Nombre_grupo</th> <th>Id_puerta</th> <th>Nombre_puerta</th> </tr> </thead> <tbody> <tr> <td>4863</td> <td>111</td> <td>4752</td> <td>Puerta 111</td> </tr> <tr> <td>1632</td> <td>TSU 121</td> <td>4553</td> <td>Puerta 112</td> </tr> <tr> <td>3040</td> <td>112</td> <td>4752</td> <td>Puerta 112</td> </tr> <tr> <td>Total</td> <td colspan="3">3</td> </tr> </tbody> </table>					Id_grupo	Nombre_grupo	Id_puerta	Nombre_puerta	4863	111	4752	Puerta 111	1632	TSU 121	4553	Puerta 112	3040	112	4752	Puerta 112	Total	3		
Id_grupo	Nombre_grupo	Id_puerta	Nombre_puerta																					
4863	111	4752	Puerta 111																					
1632	TSU 121	4553	Puerta 112																					
3040	112	4752	Puerta 112																					
Total	3																							

	Campos	Tipos de Datos	Reglas o Restricciones
	Observaciones	1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos.	

Tabla 13: especificación de requisito: mostrar listado de los clientes pertenecientes al complejo 1, o 2 o 3.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF4.]	Mostrar listado de los clientes pertenecientes al complejo 1 o 2 o 3.	Una vez que el usuario de clic sobre la funcionalidad se mostrará una lista desplegable donde el usuario podrá escoger el complejo, luego al dar clic en la opción solicitar se mostrarán el nombre de la persona y el tipo de persona. Seguido de la opción de solicitar aparecerá la opción de exportar a pdf para poder imprimir el reporte.	Alta	
Prototipo				
Prototipo Mostrar listado de los clientes pertenecientes al complejo 1 o 2 o 3.				

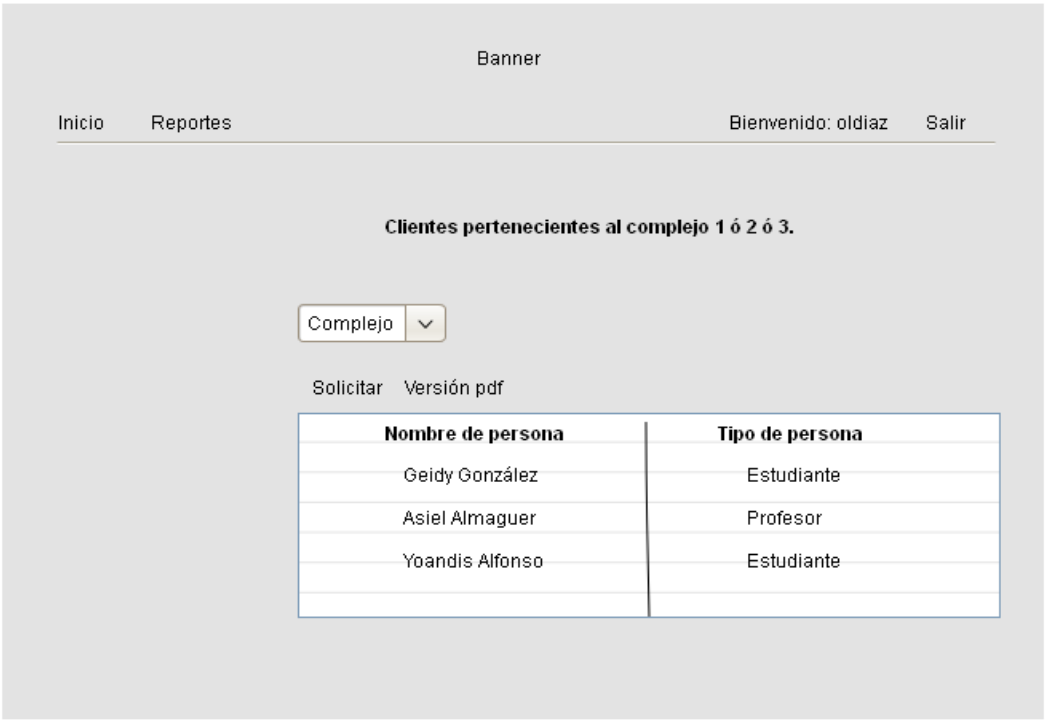
		
Campos	Tipos de Datos	Reglas o Restricciones
Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos. 	

Tabla 14: especificación de requisito: mostrar cantidad de clientes que recibieron servicios en un rango de fecha.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF10.]	Mostrar cantidad de clientes que recibieron servicios en un rango de fecha.	Primeramente se muestra una lista despegable donde el usuario podrá elegir el la fecha, luego el usuario al dar clic sobre el botón solicitar se le mostrarán los eventos y la cantidad de accedidos por cada evento, mostrándose en la parte inferior izquierda la cantidad total de accesos. Seguido de la opción solicitar se encontrará la opción exportar a pdf para poder imprimir el reporte.	Media.	
Prototipo				
Prototipo Mostrar cantidad de clientes que recibieron servicios en un rango de fecha.				

	Banner												
	Inicio Reportes	Bienvenido: oldiaz Salir											
	Cantidad de clientes que recibieron servicios en un rango de tiempo.												
	Fecha_Inicial	Fecha_Final											
	Solicitar Versión pdf												
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Eventos</th> <th style="text-align: left;">Servicios ofertados</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">Desayuno</td> <td style="text-align: center;">150</td> </tr> <tr> <td style="text-align: center;">Almuerzo</td> <td style="text-align: center;">400</td> </tr> <tr> <td style="text-align: center;">Doble almuerzo</td> <td style="text-align: center;">600</td> </tr> <tr> <td style="text-align: center;">Total</td> <td style="text-align: center;">1730</td> </tr> </tbody> </table>			Eventos	Servicios ofertados	Desayuno	150	Almuerzo	400	Doble almuerzo	600	Total	1730
Eventos	Servicios ofertados												
Desayuno	150												
Almuerzo	400												
Doble almuerzo	600												
Total	1730												
	Campos	Tipos de Datos	Reglas o Restricciones										
	Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos. 											

Tabla 15: especificación de requisito: autenticar usuario.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF1.]	Autenticar usuario en el sistema.	El usuario entra su usuario y su contraseña si los datos entrados son correctos se le mostrará la página inicial del sistema.	Media.	

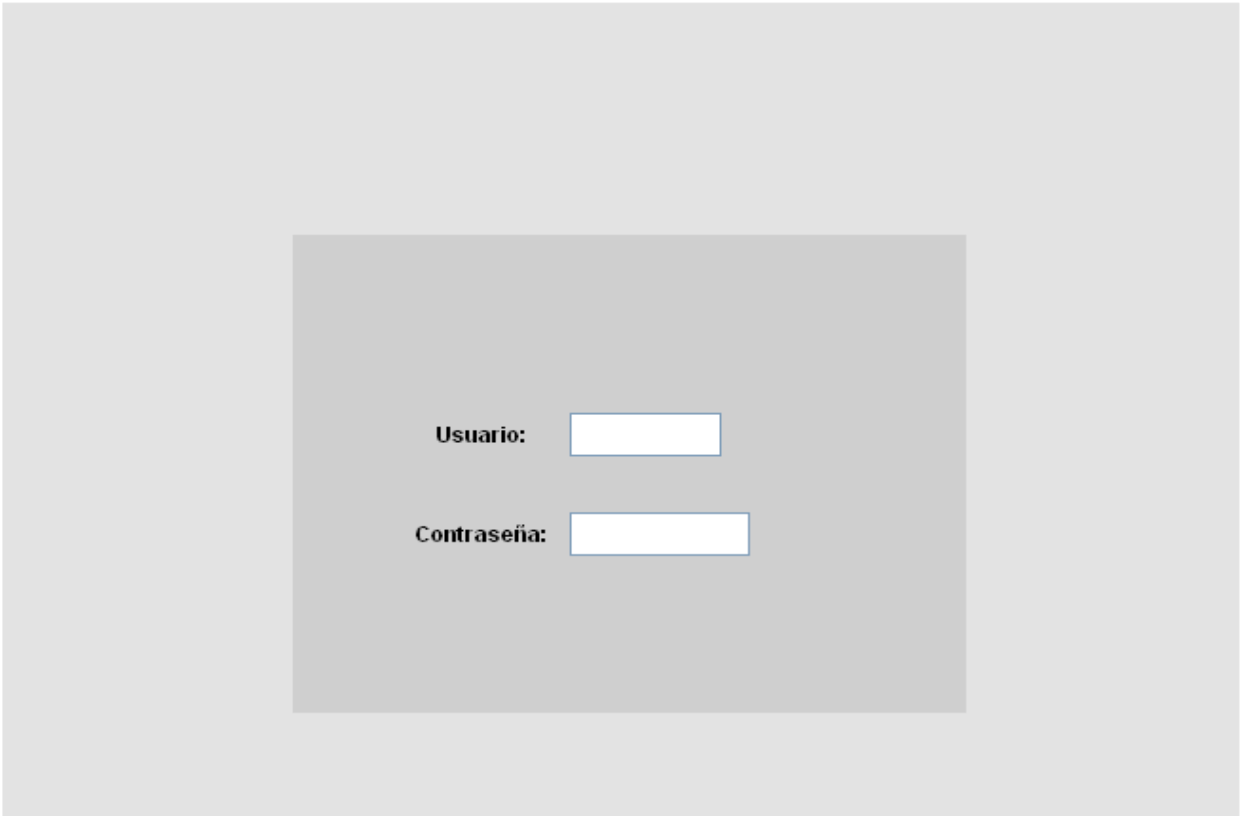
Prototipo		
<p>Prototipo Autenticar usuario en el sistema.</p> 		
Campos	Tipos de Datos	Reglas o Restricciones
Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario entre mal la contraseña no podrá entrar al sistema. 	

Tabla 16: especificación de requisito: mostrar cantidad de accedidos por nivel.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF9.]	Mostrar cantidad de accedidos por nivel.	Primeramente se muestra una lista despegable donde el usuario podrá elegir el evento, complejo, el comedor, la puerta y la fecha inicial y final, luego el usuario al dar clic sobre el botón solicitar se le mostrarán los eventos y la fecha inicial y final, mostrándose en la parte inferior izquierda la cantidad total de accesos. Seguido de la opción solicitar se encontrará la opción exportar a pdf para poder imprimir el reporte.	Baja.	
Prototipo				
Prototipo Mostrar cantidad de accedidos por nivel.				

<div style="text-align: center;">Banner</div> <hr/> <p>Inicio Reportes Bienvenido: oldiaz Salir</p> <p style="text-align: center;">Cantidad de clientes que accedieron por nivel.</p> <div style="display: flex; justify-content: space-around; margin-bottom: 10px;"> <input type="text" value="Fecha_Inicial"/> <input type="text" value="Fecha_Final"/> <input type="text" value="Evento"/> </div> <div style="display: flex; justify-content: space-around;"> <input type="text" value="Complejo"/> <input type="text" value="Comedor"/> <input type="text" value="Puerta"/> </div> <p style="text-align: center;">Solicitar Versión pdf</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Eventos</th> <th style="width: 35%;">Fecha_Inicial</th> <th style="width: 35%;">Fecha_Final</th> </tr> </thead> <tbody> <tr><td>Desayuno</td><td></td><td></td></tr> <tr><td>Almuerzo</td><td></td><td></td></tr> <tr><td>Comida</td><td></td><td></td></tr> <tr><td>Total</td><td></td><td></td></tr> </tbody> </table>	Eventos	Fecha_Inicial	Fecha_Final	Desayuno			Almuerzo			Comida			Total		
Eventos	Fecha_Inicial	Fecha_Final													
Desayuno															
Almuerzo															
Comida															
Total															
Campos	Tipos de Datos	Reglas o Restricciones													
Observaciones	<ol style="list-style-type: none"> En caso de que el usuario no elija ninguna opción se muestra el listado vacío. Interactúa con esta acción el administrador y los usuarios permitidos. 														

Tabla 17: especificación de requisito: mostrar cantidad de clientes por cada categoría en un evento.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
[RF6.]	Mostrar cantidad de clientes por cada categoría en un evento.	Una vez que el usuario de clic sobre la funcionalidad se mostrará una lista desplegable donde el usuario selecciona el complejo, el comedor y la puerta además de especificar una fecha, luego el usuario da clic en la opción solicitar y se mostrarán las categorías por eventos, la cantidad de accesos y en la parte inferior izquierda el total de estos accesos. Seguido de la opción solicitar estará la opción exportar a pdf para poder imprimir el reporte.	Media	
Prototipo				
Prototipo Mostrar cantidad de clientes que accedieron por cada categoría en un evento.				

<div style="text-align: center;">Banner</div> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Inicio Reportes Bienvenido: oldiaz Salir </div> <hr style="border: 0.5px solid #ccc; margin: 10px 0;"/> <p style="text-align: center; font-weight: bold; margin-bottom: 10px;">Clientes que accedieron por cada categoría en un evento.</p> <div style="margin-bottom: 10px;"> <input style="width: 80px; border: 1px solid #ccc;" type="text" value="Fecha"/> </div> <div style="display: flex; justify-content: space-around; margin-bottom: 10px;"> <div style="border: 1px solid #ccc; padding: 2px 5px; display: flex; align-items: center;"> Complejo ▼ </div> <div style="border: 1px solid #ccc; padding: 2px 5px; display: flex; align-items: center;"> Comedor ▼ </div> <div style="border: 1px solid #ccc; padding: 2px 5px; display: flex; align-items: center;"> Puerta ▼ </div> </div> <div style="margin-bottom: 5px;"> Solicitar Versión pdf </div> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 70%;">Categoría por eventos</th> <th style="width: 30%;">Cantidad de accesos</th> </tr> </thead> <tbody> <tr> <td>Regular</td> <td>0</td> </tr> <tr> <td>No regular venta</td> <td>0</td> </tr> <tr> <td>No regular doble</td> <td></td> </tr> <tr> <td> </td> <td> </td> </tr> <tr> <td>Total</td> <td>0</td> </tr> </tbody> </table>	Categoría por eventos	Cantidad de accesos	Regular	0	No regular venta	0	No regular doble				Total	0
Categoría por eventos	Cantidad de accesos											
Regular	0											
No regular venta	0											
No regular doble												
Total	0											
Campos	Tipos de Datos	Reglas o Restricciones										
Observaciones	<ol style="list-style-type: none"> 1. En caso de que el usuario no elija ninguna opción se muestra el listado vacío. 2. Interactúa con esta acción el administrador y los usuarios permitidos. 											

Anexo B

Casos de prueba

Tabla 18: descripción del caso de prueba para el requisito mostrar cantidad de clientes que accedieron en cada evento del día por comedores.

RF_ Mostrar cantidad de clientes que accedieron en cada evento del día por comedores.		
Entrada	Condición de ejecución	Respuesta del sistema
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes que accedieron en cada evento del día por comedores y llena los siguientes campos</p> <p>Complejo: complejo1 Comedor: comedor 1 Puerta: puerta 112 Fecha: 4/04/2013</p> <p>Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>El usuario tiene que estar autenticado en el sistema.</p>	<p>El sistema debe mostrar una tabla con los eventos, la cantidad de accedidos, en la parte inferior izquierda la cantidad total de accesos y además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.</p>
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes que accedieron en cada evento del día por comedores y llena los siguientes campos</p> <p>Complejo: " " Comedor: comedor 1 Puerta: puerta 112 Fecha: 4/04/2013</p> <p>Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>No puede haber campos vacíos.</p>	<p>El sistema muestra un mensaje indicando que no puede haber campos vacíos.</p>

Tabla 19: descripción del caso de prueba para el requisito mostrar cantidad de clientes que accedieron por complejo.

RF_ Mostrar cantidad de clientes que accedieron por complejo.		
Entrada	Condición de ejecución	Respuesta del sistema
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes que accedieron por complejo y llena los siguientes campos</p> <p>Complejo: complejo1</p> <p>Fecha: 4/04/2013</p> <p>Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>El usuario tiene que estar autenticado en el sistema.</p>	<p>El sistema debe mostrar una tabla con los eventos y la cantidad de accedidos por cada evento, en la parte inferior izquierda la cantidad total de accesos y además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.</p>
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes que accedieron por complejo y llena los siguientes campos</p> <p>Complejo: " "</p> <p>Fecha: 4/04/2013</p> <p>Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>No puede haber campos vacíos.</p>	<p>El sistema muestra un mensaje indicando que no puede haber campos vacíos.</p>

Tabla 20: descripción del caso de prueba para el requisito mostrar cantidad de clientes que han recibido servicio en el día de hoy.

RF_ Mostar cantidad de clientes que han recibido servicio en el día de hoy.		
Entrada	Condición de ejecución	Respuesta del sistema
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de</p>	<p>El usuario tiene que estar autenticado en el sistema.</p>	<p>El sistema debe mostrar una tabla con el evento, la cantidad de servicios ofertados, en la parte</p>

clientes que han recibido servicio en el día de hoy.		inferior izquierda la cantidad total de accesos y además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.
El usuario no da clic en la funcionalidad: cantidad de clientes que han recibido servicio en el día de hoy.		No se muestra ningún dato.

Tabla 21: descripción del caso de prueba para el requisito mostrar cantidad de clientes que recibieron servicio en un rango de fecha.

RF_ Mostrar cantidad de clientes que recibieron servicio en un rango de tiempo.		
Entrada	Condición de ejecución	Respuesta del sistema
El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes que recibieron servicio en un rango de fecha y llena el siguiente campo: Fecha: 4/04/2013 Luego de la entrada de datos da clic en la opción solicitar.	El usuario tiene que estar autenticado en el sistema.	El sistema debe mostrar una tabla los eventos, la cantidad de accesos en la parte inferior izquierda la cantidad total de accesos y además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.
El usuario una vez autenticado escoge del menú Reportes la funcionalidad cantidad de clientes que recibieron servicio en un rango de fecha y llena los siguientes campos Fecha: "" Luego de la entrada de datos da clic en la opción solicitar.	No puede haber campos vacíos.	El sistema muestra un mensaje indicando que no puede haber campos vacíos.

Tabla 22: descripción del caso de prueba para el requisito mostrar cantidad de accedidos por nivel.

RF_ Mostrar cantidad de accedidos por nivel.		
Entrada	Condición de ejecución	Respuesta del sistema
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de accedidos por nivel y llena los siguientes campos</p> <p>Complejo: complejo1</p> <p>Comedor: comedor 1</p> <p>Puerta: puerta 112</p> <p>Evento: desayuno</p> <p>Fecha Inicial: 4/04/2013</p> <p>Fecha Final: 5/04/2013</p> <p>Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>El usuario tiene que estar autenticado en el sistema.</p>	<p>El sistema debe mostrar una tabla con los eventos, la fecha inicial y final, en la parte inferior izquierda la cantidad total de accesos y además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.</p>
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad cantidad de accedidos por nivel y llena los siguientes campos</p> <p>Complejo: " "</p> <p>Comedor: comedor 1</p> <p>Puerta: puerta 112</p> <p>Evento: desayuno</p> <p>Fecha Inicial: 4/04/2013</p> <p>Fecha Final: 5/04/2013</p> <p>Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>No puede haber campos vacíos.</p>	<p>El sistema muestra un mensaje indicando que no puede haber campos vacíos.</p>

Tabla 23: descripción del caso de prueba para el requisito mostrar cantidad de clientes por cada categoría en un evento.

RF_ Mostrar cantidad de clientes por cada categoría en un evento.		
Entrada	Condición de ejecución	Respuesta del sistema
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes por cada categoría en un evento y llena los siguientes campos</p> <p>Complejo: complejo1 Comedor: comedor 1 Puerta: puerta 112 Fecha: 4/04/2013 Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>El usuario tiene que estar autenticado en el sistema.</p>	<p>El sistema debe mostrar una tabla con las categorías por eventos, la cantidad de accesos, en la parte inferior izquierda la cantidad total de accesos y además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.</p>
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes por cada categoría en un evento y llena los siguientes campos</p> <p>Complejo: " " Comedor: comedor 1 Puerta: puerta 112 Fecha: 4/04/2013 Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>No puede haber campos vacíos.</p>	<p>El sistema muestra un mensaje indicando que no puede haber campos vacíos.</p>

Tabla 24: descripción del caso de prueba para el requisito mostrar listado de los clientes pertenecientes al complejo 1 o 2 o 3.

RF_ Mostrar listado de los clientes pertenecientes al complejo 1 o 2 o 3.		
Entrada	Condición de ejecución	Respuesta del sistema
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes pertenecientes al complejo 1 o 2 o 3 y llena el siguiente campo Complejo: complejo 1 Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>El usuario tiene que estar autenticado en el sistema.</p>	<p>El sistema debe mostrar una tabla con el nombre de la persona y el tipo de persona y además que cuando el usuario escoja la opción versión pdf se deben mostrar todos estos datos en un pdf.</p>
<p>El usuario una vez autenticado escoge del menú Reportes la funcionalidad: cantidad de clientes pertenecientes al complejo 1 o 2 o 3 y llena el siguiente campo Complejo: " " Luego de la entrada de datos da clic en la opción solicitar.</p>	<p>No puede haber campos vacíos.</p>	<p>El sistema muestra un mensaje indicando que no puede haber campos vacíos.</p>