Universidad de las Ciencias Informáticas

Facultad 3

Personalización del módulo de Ventas para el Sistema de Administración de Relaciones con el Cliente en Odoo

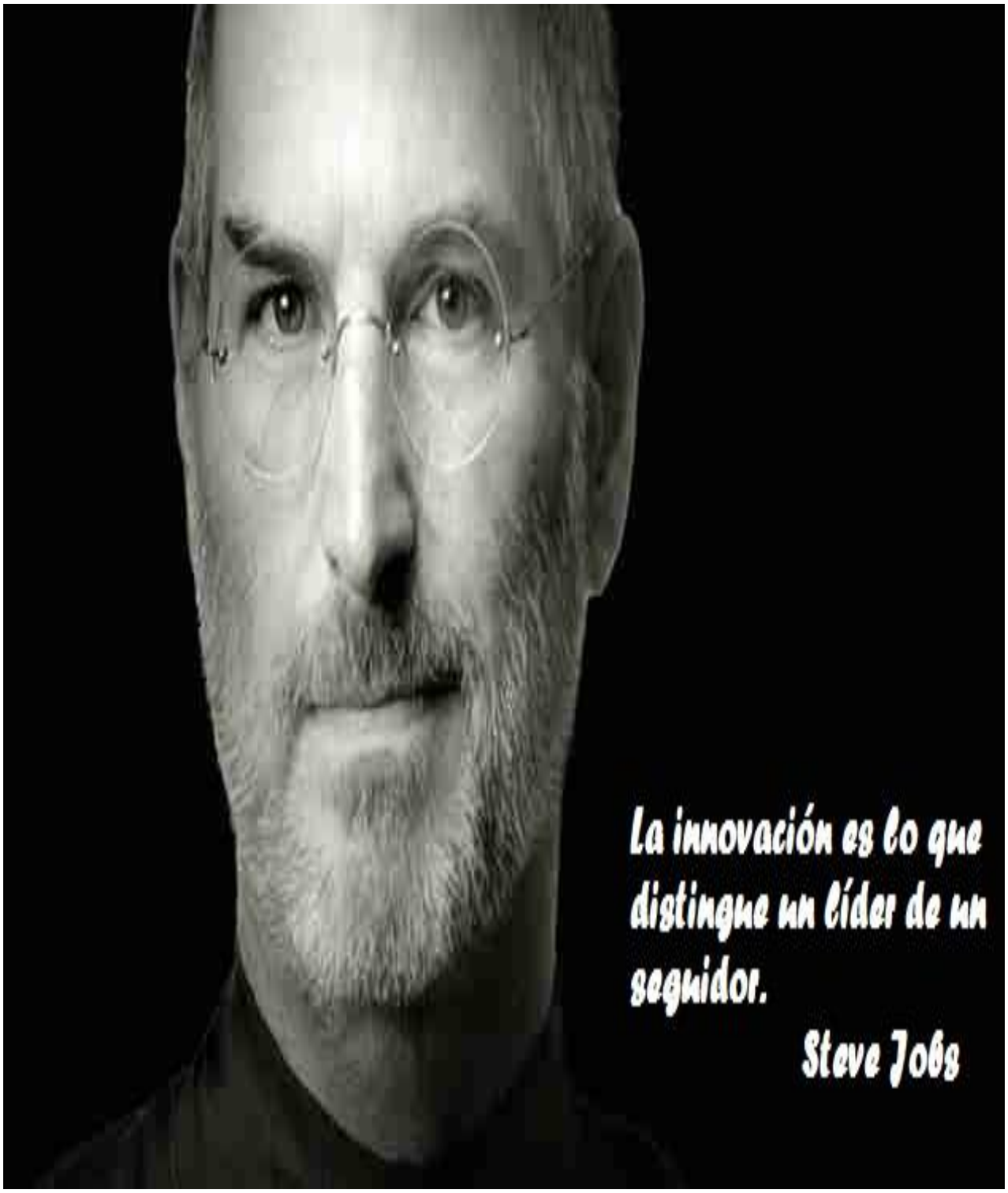
Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Oscar Miguel Santana Chirino

Tutor(es): Ing. Yoenry Vanega Hechavarría

Ing. Ernesto Mató Roque

La Habana, 2018


La innovación es lo que distingue un líder de un seguidor.
Steve Jobs

DECLARACIÓN DE AUTORÍA

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

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2018.

Autor:

_____

Oscar Miguel Santana Chirino

Tutores:

_____

Ing. Yoenry Vanega Hechavarría

_____

Ing. Ernesto Mató Roque

Agradecimientos

Agradezco a toda mi familia, los que tengo cerca y los que están lejos en especial a mis padres por todo su apoyo y sacrificio durante estos cinco años de carrera los quiero mucho. A mi novia por haberme apoyado en las buenas y en las malas te amo. A mis tutores por su amistad y por su tiempo dedicado en este trabajo. A mis compañeros de año y de apartamento Leodan, Julio Luis y Bárbaro por todo el apoyo brindado a lo largo de estos 5 años.

A Lucio por sus consejos, sus correcciones …..

A Osvaldo por sonsacarme a jugar y no estudiar hasta el día antes de las pruebas.

A Raykof, el Dacho, el Moreno por alegrar mis tardes en las canchas de futbol.

A toda mi familia por su apoyo incondicional siempre. En especial a mi mamá mujer increíble y sobreprotectora gracias por darme la vida.

A mi papá que decir de mi papá hombre sin igual, amigo y mejor maestro en esta vida apoyándome hasta en mis mayores locuras…

A mi tía que siempre me ha apoyado en todo, complaciéndome y mimándome.

A mi segunda mamá Erla la cual estuvo siempre al tanto de toda mi carrera.

RESUMEN

El surgimiento de nuevas formas y técnicas para el trabajo posibilitan obtener software que ayuden a viabilizar los procesos empresariales donde juega un papel fundamental los Sistemas de Administración de Relaciones con el Cliente. Con el desarrollo de las Tecnologías de la Información y las Comunicaciones se ha logrado una mayor gestión de la información. Actualmente, en la Universidad de las Ciencias Informáticas, el procedimiento usado para el seguimiento a las tareas de la gestión de ventas es parcialmente manual. La mayoría de la información se almacena en formato duro (papel) y la información de los clientes se encuentra dispersa en diferentes tipos de documentos digitales y agendas personales. Lo que dificulta identificar las necesidades de los mismos para la creación de oportunidades y ofertas. Debido a los problemas que presenta esta dirección se plantea como objetivo la personalización del módulo de Ventas de Odoo permitiendo la gestión de los procesos de ventas en la Administración de Relaciones con el Cliente de la Dirección de Transferencia de Tecnología en la UCI. Para guiar el proceso de desarrollo del sistema se utilizó la metodología de software Variación de AUP para la UCI y en la implementación se emplearon tecnologías y herramientas definidas por el proyecto al que pertenece esta investigación. El módulo elaborado fue validado a través de las pruebas definidas por la metodología y se aplicó la técnica de valoración de expertos utilizando el método Delphy para validar la investigación.

Palabras claves: administración, clientes, ventas.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La toma de decisiones empresariales o administrativas a nivel mundial no ha quedado exenta a grandes cambios en el proceso evolutivo tales como la globalización e internacionalización de los mercados, la creciente incertidumbre en el entorno, el aumento de la competencia, entre otros que motivan la ocurrencia de variaciones en las concepciones y modos de actuación de las empresas que desean proseguir en el mundo cambiante de hoy (Quiñones Montoro, 2016).

Durante la mayor parte del siglo XX, la práctica de la Administración de Ventas fue una combinación de experiencias personales e intuición no contando con técnicas para este proceder. Los gerentes de ventas realizaban pocas investigaciones y recurrían poco a la teoría de la administración para entender mejor los motivos y las conductas de sus propios vendedores y clientes. Como resultado, prácticamente no existía ningún apoyo para los gerentes de ventas en la práctica. Durante la década de 1970, los académicos e investigadores en ventas se dieron a la tarea de emprender estudios empíricos y elaborar modelos teóricos para comprender mejor las motivaciones y conductas de sus propios vendedores (HALL, 2014).

Con el desarrollo de las Tecnologías de la Información y las Comunicaciones (TICs) se ha logrado una mayor gestión de la información, jugando un papel fundamental en el desarrollo del sector empresarial. De tal manera surgieron modelos de gestión para las organizaciones, basados principalmente en la orientación al cliente, enfocándose en la optimización de los procesos. Por esta razón, surgen las estrategias de negocio enfocadas a seleccionar y gestionar los clientes.

Cuba también forma parte de este avance tecnológico surgiendo así la Universidad de las Ciencias Informáticas (UCI), estructurada por seis facultades docentes. En dicha estructura se encuentran los centros de desarrollo, estos tienen como misión fundamental el desarrollo de aplicaciones y servicios informáticos para colaborar en la solución de necesidades y problemas existentes en la sociedad, permitiendo la gestión de los diversos procesos antes mencionados (Lopez, 2008).

Estas aplicaciones y servicios informáticos son comercializados por la Dirección de Transferencia de Tecnología de la UCI. Esta dirección teniendo en cuenta la gama de procesos que en él ocurren, tiene la necesidad de gestionar y controlar la información que se maneja en el proceso que se desarrolla en torno a la gestión de ventas. Dicha dirección consta de varios departamentos uno de ellos es el Departamento Comercial, en cual existen un conjunto de limitaciones que no permiten una correcta gestión del proceso de ventas, como, por ejemplo:

- La información de los clientes se encuentra dispersa en diferentes tipos de documentos digitales y agendas personales lo que dificulta identificar las necesidades de los mismos para la creación de oportunidades y ofertas.

- El procedimiento usado actualmente para el seguimiento a las tareas de la gestión de ventas es parcialmente manual y la mayoría de la información se almacena en formato duro (papel). Desde esta perspectiva se identifican las siguientes deficiencias:
  - Pérdida de información relevante por deterioro del papel con el paso del tiempo.
  - Insuficiente seguridad de la información en los documentos pudiendo existir alteración por parte de personal no autorizado.
  - Incrementa el consumo de recursos materiales.
  - Se hace difícil la consulta de la información histórica del proceso de venta.
- La herramienta que apoya la gestión de venta para la Administración de Relaciones con el Cliente, aunque ayuda de forma significativa en el proceso no cumple con todas las necesidades, generando las siguientes limitaciones:
  - Incumplimiento en la fecha de entrega de los contratos.
  - Limita el seguimiento a las actividades con los clientes.
  - Limita el seguimiento de las facturas impagadas.
  - Limita la creación de reportes, que es la mayor necesidad que tiene hoy el colectivo de especialistas en el Departamento Comercial de la Dirección de Transferencia de Tecnología.

El Centro de Informatización de Entidades (CEIGE) en aras de desarrollar un ERP (Planificación de Recursos Empresariales) por las características de esta herramienta, adoptó a Odoo como base para la implementación. Este sistema permite la integración de la mayoría de las operaciones que se realizan en una empresa. Sin embargo, el módulo provisto por defecto para gestionar las ventas no cumple en su totalidad con las regulaciones cubanas.

Partiendo de lo anterior se plantea como **problema a resolver:** ¿Cómo contribuir a la gestión de los procesos de Venta en la Administración de Relaciones con el Cliente en la UCI?

Para ello la investigación centra su **objeto de estudio** en la Informatización del proceso de Ventas y el **campo de acción** se enmarca en la personalización del módulo de Ventas que tiene Odoo para la Dirección de Transferencia de Tecnología en la UCI.

Así mismo, para darle respuesta al problema se plantea el siguiente **objetivo general:** personalizar el módulo CRM de Odoo para contribuir la gestión de los procesos de Venta en la Dirección de Transferencia de Tecnología en la UCI.

Teniendo como referencia el problema a resolver y el objetivo general se plantea la siguiente **idea a defender**:

La personalización del módulo de Ventas permitirá la gestión de los procesos de ventas en la administración de relaciones con el cliente de la Dirección de Transferencia de Tecnología en la UCI.

Por ello se establecen los siguientes **objetivos específicos:**

- Elaborar el marco teórico de la investigación a partir de los principales referentes acerca de los sistemas CRM para identificar las limitaciones de Odoo y las nuevas funcionalidades a incorporar.
- Realizar el análisis y diseño del proceso de Ventas para la Dirección de Transferencia de Tecnología para sentar las bases de la implementación del sistema.
- Implementar la personalización del módulo de Ventas para la Administración de Relaciones con el Cliente en Odoo.
- Validar la solución propuesta mediante la aplicación de métricas y pruebas de software.
- Validar la investigación mediante criterio de expertos (método Delphy).

Con el fin de resolver y dar cumplimiento a los objetivos específicos, se hace necesaria la utilización de los siguientes métodos científicos:

Métodos Teóricos:

Histórico-Lógico: Analizan la trayectoria completa del fenómeno, su condicionamiento a los diferentes períodos de la historia, revela las etapas principales de su desenvolvimiento, las conexiones históricas fundamentales, además de poner de manifiesto la lógica interna de su desarrollo (González, 2002). Se empleó con el objetivo de conocer toda la evolución histórica sobre el contenido del sistema de gestión de ventas para la administración de relaciones con el cliente en la Dirección de Transferencia de Tecnología y aquellos sucesos significativos en los procesos de administración, que puedan ser de gran aporte a la investigación.

Modelación: La modelación es el método mediante el cual se crean abstracciones con el objetivo de explicar la realidad. El modelo como sustituto del objeto de investigación es semejante a él, existiendo una correspondencia objetiva entre el modelo y el objeto, siendo el investigador quien elabora dicho modelo. El modelo es el eslabón entre el sujeto y el objeto intermedio (González, 2002). Se utilizó para crear todos los modelos y diagramas que permitieron un mejor entendimiento de los procesos del sistema, destacando: el modelo conceptual, la especificación de requisitos, los diagramas de clases del diseño y el modelo de datos.

Hipotético – deductivo: A partir de la hipótesis y siguiendo reglas lógicas de deducción se llega a nuevos conocimientos y predicciones, las que posteriormente son sometidas a verificaciones

empíricas. (González, 2002) Su aplicación se evidenció cuando se realizó un profundo estudio del sistema encargado de la administración de relaciones con el cliente utilizado en la Dirección de Transferencia de Tecnología en la UCI, en el cual se analizó el estado actual de este sistema, recolectando así mucha información de su desenvolvimiento, lo que trajo consigo formular la posibilidad de que un nuevo sistema tecnológico apoyaría con mayor eficiencia los procesos de ventas.

Método Empírico:

Entrevista: Comunicación verbal entre dos o más personas con el objetivo de obtener información acerca de determinadas cuestiones (González, 2002). Su aplicación permitió realizar entrevistas individuales con especialistas de la Dirección de Transferencia de Tecnología en la UCI, obteniendo con esto una valiosa información del negocio y del esclarecimiento de los requisitos.

El presente documento está estructurado en los siguientes capítulos:

Capítulo 1. Fundamentación teórica: constituye la fundamentación teórica de la investigación. Se tratan los principales conceptos para una correcta comprensión del tema. Se realizará una breve conceptualización del diseño y de los patrones a utilizar. Se describen las tecnologías, lenguajes y herramientas necesarias para el desarrollo del sistema, así como la metodología que guiará el proceso de desarrollo de software.

Capítulo 2. Análisis y diseño de la propuesta de solución: Se obtiene los productos de trabajo resultantes del desarrollo de las actividades ejecutadas en las disciplinas de modelado del negocio, requisitos, análisis y diseño que se definen en la metodología.

Capítulo 3. Implementación y validación de la propuesta de solución: En este capítulo se abordan aspectos relacionados con la implementación del sistema, se describe la nomenclatura usada en la implementación de la solución, la estrategia de pruebas definida a partir de la metodología de desarrollo, con el objetivo de garantizar la validación de la solución propuesta. Se analizan los niveles de pruebas ejecutados y los resultados alcanzados.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se realiza el estudio de los principales CRM que existen en el mundo, sus características y funcionalidades en el área de ventas, comparándolas con las funcionalidades del módulo de Ventas de Odoo. Se enuncian los principales conceptos relacionados con la gestión de ventas y se define la metodología, tecnologías y herramientas que serán empleadas para la construcción de la solución.

1.1 Marco conceptual

Planificación de Recursos Empresariales (ERP)

Los sistemas de Planificación de Recursos Empresariales (en inglés ERP, Enterprise Resource Planning) son sistemas de gestión de información que automatizan muchas de las prácticas de negocio asociadas con los aspectos operativos o productivos de una empresa, básicamente es una arquitectura de software para empresas que facilita e integra la información entre las funciones de manufactura, logística, finanzas y recursos humanos de una empresa. (excelia, 2018)

Administración de Relaciones con el Cliente (CRM)

El CRM consiste en una estrategia de la organización en la cual centra sus esfuerzos en el conocimiento de sus clientes, detectando sus necesidades, aumentando su grado de satisfacción, incrementando su fidelidad a la empresa e incrementando la rentabilidad o beneficios del cliente a la empresa, mediante el análisis de las informaciones extraídas por los clientes desde los diferentes canales o medios de comunicación. (Benaque, 2007)

Un CRM es una herramienta de negocios que, a partir de soluciones tecnológicas, se pueden mejorar las relaciones con los clientes. Para esto el sistema brinda un grupo de datos acerca de los clientes que son usados para realizar un estudio previo del mercado para conocer los gustos de los clientes, adaptando los productos a sus necesidades y lograr que queden totalmente satisfechos. (excelia, 2018)

Se adopta como concepto de CRM a seguir a lo largo de todo el proceso el planteado por la dirección de ventas de Exelia porque es el que más se adapta a la Dirección de Transferencia de Tecnología en la UCI.

Odoo

Odoo (conocido anteriormente como OpenERP) es un ERP integrado de código abierto producido por OpenERP S.A. El mismo también implementa un módulo de CRM. Su fabricante define su producto como el ERP de código abierto más sencillo y destacado del momento, el mismo cumple

las 4 libertades del software libre, basado en estándares abiertos y desarrollado con plataformas libres (Odoo, 2017). Además, Odoo ha ido evolucionando a un ritmo muy rápido. En apenas 4 años ha pasado de la versión 7 (Open ERP) a la versión Odoo 10. Esta última versión combina avanzadas funciones con una interfaz muy amigable. Entre las principales características de Odoo 10 se encuentra su escalabilidad, modularidad y flexibilidad. Posee una importante comunidad de desarrolladores, con presencia en varios países de América y Europa, así como Asia y Australia, que están constantemente ampliando y mejorando el proyecto (amplia documentación, foros, sistema de control de versiones (cvs), listas de correo, desarrollo comunitario y traducciones en una plataforma de desarrollo colaborativo de software (Launchpad)). La arquitectura que utiliza Odoo es cliente-servidor donde el servidor se ejecuta independientemente del cliente y maneja la lógica de negocio y comunica con la aplicación de base de datos. El desarrollo de módulos se realiza editando archivos Python y XML. No hay un editor oficial, aunque los tutoriales destacan Eclipse o PyCharm + PyDev. Odoo 10 también incluye un sistema de reportes con integración con OpenOffice.org, lo que permite personalizar los informes. También hay motores de reportes alternativos utilizando Webkit o Jaspersoft (Odoo, 2017).

Acciones de ventas:

- Gestionar iniciativa de venta
- Gestión de pedido de venta
- Gestión de cliente
- Convertir iniciativa en oportunidad (Odoo, 2017).

Ventas:

La American Marketing Asociation, define la venta como "el proceso personal o impersonal por el que el vendedor comprueba, activa y satisface las necesidades del comprador para el mutuo y continuo beneficio de ambos (del vendedor y el comprador)" (Rendón, 2007).

El concepto de venta supone que es preciso estimular a los consumidores para que compren. Para ello, las empresas que ponen en práctica este concepto, utilizan todo un arsenal de herramientas de venta y promoción para estimular más compras (Thompson, 2012).

En general, las ventas para una empresa u organización es vender lo que produce. Para ello, dirige sus esfuerzos hacia todas las actividades que le permitan estimular a sus clientes para que tomen una decisión favorable como inscribirse o comprar.

Clientes

En toda estrategia empresarial, el cliente es una figura que siempre está presente, es lógico, es quien demanda los productos y servicios que las empresas ofrecen, por lo que consiguen consolidarse en el mercado para obtener rendimientos e ingresos para posicionarse y sobrevivir (Zaragoza, 2012).

1.2 Análisis de los principales sistemas

Un sistema de CRM resulta clave para desarrollar tácticas de marketing y ventas que impacten y es una de las piezas más importantes que sustentan la estrategia comercial de una compañía. Con el uso de los CRM las empresas mejoran las relaciones con sus clientes, facilitan la toma de decisiones, optimizan el proceso de venta, por lo que crean un servicio al cliente más eficiente lo que hace posible un aumento del grado de fidelidad con la empresa (excelia, 2018).

Se relacionan algunos de los CRM más utilizados en el mundo empresarial. Con el objetivo de encontrar las mejores funcionalidades asociadas al proceso de ventas cumpliendo con las necesidades de la Dirección de Transferencia de Tecnología y las regulaciones cubanas vigentes para dicho proceso.

Salesforce

Salesforce es el software de CRM y ecosistema de nubes empresariales líder en el mundo. Eso significa que está todo en línea, sin software, sin hardware. No hay costos de configuración, ni mantenimiento, sus empleados pueden trabajar desde cualquier dispositivo con conexión a Internet (teléfono inteligente, tableta o computadora portátil) y con 3 actualizaciones gratuitas cada año es ideal tanto para pequeñas empresas como para grandes empresas. La plataforma de éxito del cliente de Salesforce puede ayudar a que la empresa crezca y se convierta en una organización más racionalizada, eficaz y eficiente en ventas, servicios, marketing (DUGAN, 2016).

Acciones de ventas:

- Gestión de oportunidades.
- Gestión de contratos.
- Gestión de clientes o perfiles de usuario.
- Gestión de clientes potenciales.
- Gestión de pronósticos de ventas.
- Gestión de ventas (DUGAN, 2016).

AllProWebTools

AllProWebTools es una suite de gestión de relaciones con los clientes (CRM) basada en la nube para pequeñas empresas que ofrece aplicaciones de comercio electrónico, automatización de marketing, gestión de fuerza de ventas y servicio y soporte al cliente. Proporciona un cronograma de flujo de trabajo con actualizaciones en tiempo real sobre las operaciones comerciales, desde la actividad del cliente, nuevos pedidos o carritos abandonados, hasta el progreso de los empleados en las tareas (AllProWebTools, 2016).

Acciones de ventas:

- Gestión iniciativa de venta.
- Gestión de pedidos de venta.
- Gestión de solicitudes de ventas.
- Gestión de descuentos de ventas.
- Gestión de clientes.
- Gestión de clientes potenciales (AllProWebTools, 2016).

NetSuite

Oracle NetSuite CRM es la única solución en la nube que ofrece una vista completa en tiempo real de sus clientes. NetSuite CRM proporciona un flujo continuo de información a lo largo de todo el ciclo de vida del cliente, desde el liderazgo hasta la oportunidad, el pedido de ventas, el cumplimiento, la renovación, la venta adicional, la venta cruzada y el soporte. Además de ofrecer capacidades CRM tradicionales como gestión de servicio al cliente y automatización de marketing, NetSuite CRM ofrece cotizaciones, gestión de pedidos, comisiones, previsión de ventas y capacidades integradas de comercio electrónico (Netsuite, 2013).

Acciones de ventas:

- Gestión de cuentas.
- Gestión de pedido.
- Gestión de oferta.
- Gestión de usuarios.
- Gestión de casos.
- Gestión de ventas (Netsuite, 2013).

Microsoft Dynamics

Microsoft Dynamics CRM permite definir una estrategia de negocio centrada en anticipar, conocer, satisfacer las necesidades y los deseos de sus clientes, incrementando la efectividad de los empleados de ventas y servicios. Microsoft Dynamics CRM permite que pequeñas y medianas empresas, y áreas específicas de grandes organizaciones, incrementen el éxito de sus ventas, entreguen un mejor servicio al cliente y tomen mejores decisiones, en base al manejo de información de valor (Microsoft Dynamics , 2015).

Acciones de ventas:

- Gestiona tipo de órdenes.
- Gestiona tipo de operación.

- Gestiona los pedidos de venta.
- Gestiona las solicitudes de ventas.
- Gestiona los descuentos de ventas.
- Gestiona las liquidaciones de comisiones.
- Genera reportes personalizados para cada etapa (Microsoft Dynamics , 2015).

SugarCRM

SugarCRM es distinto a las demás soluciones que existían en el mercado antes de su aparición. La gran diferencia es que se trata de la primera aplicación de "código abierto" que consiguió posicionarse como líder de ese segmento. Esto implica que en principio cualquiera puede descargar la versión de código abierto y empezar a utilizarla, ya que no existen costos por licencia de software. Estas características han hecho que muchísimas empresas lo hayan adoptado en todo el planeta (SugarCRM, 2016).

Acciones de ventas:

- Gestión de contratos.
- Gestión de clientes.
- Gestión de clientes potenciales.
- Gestión de pronósticos de ventas.
- Gestión de oportunidades (SugarCRM, 2016).

1.2.1 Valoración de los sistemas estudiados en cuanto a funcionalidades.

Tabla 1: Resultado de los sistemas estudiados.

CRM	Funcionalidades
SalesforceCRM	Ventas: gestiona las oportunidades, clientes, clientes potenciales, pronósticos de ventas y ventas.
AllProWebTools	Ventas: gestiona iniciativas de ventas, pedidos de ventas, solicitudes de ventas, descuentos de ventas, clientes y clientes potenciales.
NetSuiteCRM	Ventas: gestiona cuentas, pedidos, ofertas, usuarios y casos.
Microsoft Dynamics	Ventas: gestiona ordenes, operaciones, pedidos de venta, descuentos de ventas, solicitudes de ventas y genera reportes personalizados por etapa.
SugarCRM	Ventas: gestiona las oportunidades, clientes, clientes potenciales, pronósticos de ventas y contratos.

Odoo CRM	Ventas: gestiona iniciativa de venta, pedido de venta, cliente y convierte iniciativas en oportunidades.

A partir del estudio realizado a los sistemas CRM, se observó que, a pesar de brindar ventajas en el área de Ventas, no cumplen con todas las necesidades de las entidades del país. Muchos de estos sistemas además de brindar soporte desde el contacto inicial hasta la resolución final y poseer grandes funcionalidades para el manejo de las relaciones con los clientes, tienen un costo inicial muy alto, por lo que el país de utilizar alguno de ellos incurriría en gastos excesivos de licencia y mantenimiento. De todos los sistemas estudiados se tomaron las funcionalidades más importantes y de mayor demanda en el mercado como: gestionar oportunidades, gestionar contratos a partir de ofertas y la generación de reportes personalizados. Estas funcionalidades se aplicaron al Odoo ya que brinda flexibilidad y simplicidad cuando se realizan modificaciones y adaptaciones. También brinda la posibilidad de integración con otras herramientas del negocio; integra un CRM, sistema de gestión de la cadena de suministro (SCM), Sistema de planificación de recursos empresariales (ERP) y muchas más soluciones verticales, utiliza un flujo de trabajo flexible y dinámico, pudiéndosele agregar funciones, campos y módulos e integrarlos a los ya existentes, además de soportar plataformas como Linux, Mac OS y Windows. La propuesta es desarrollar el módulo de ventas del CRM tomando como plataforma Odoo 10, adicionándole características y funcionalidades que se necesitan para alcanzar el producto deseado a partir de los sistemas estudiados anteriormente.

1.3 Metodología de desarrollo de software

Proceso Unificado Ágil variación UCI

La UCI propone para el desarrollo de sus proyectos de software la metodología AUP-UCI, en el presente trabajo se hará uso de esta en su versión 1.2.

De las tres fases que propone AUP-UCI, (Inicio, Ejecución y Cierre) se desarrollará la fase Inicio: en esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto. En la fase de Ejecución se elaborarán las actividades requeridas para desarrollar el modelo de negocio, la obtención requisitos, el análisis y diseño, además de la implementación, las pruebas internas, pruebas de liberación y pruebas de aceptación. (Sanchéz, 2015)

En la Figura 1 se muestran las fases de la metodología AUP-UCI:

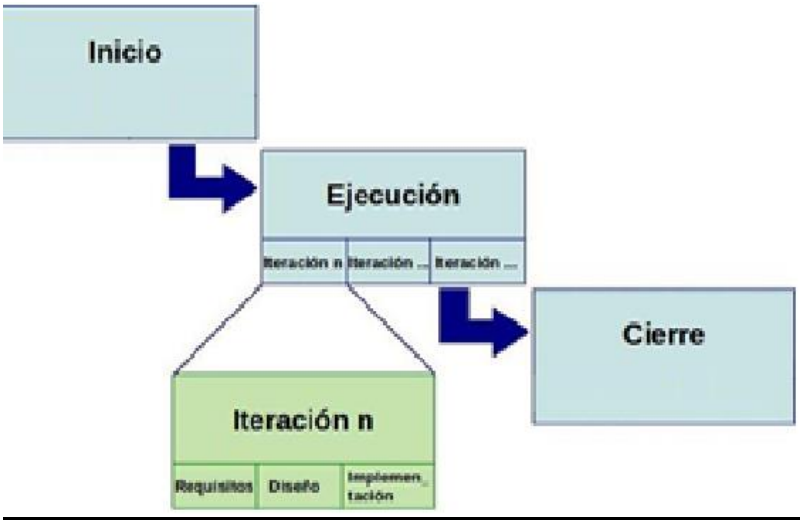


Figura 1: Fases de desarrollo de software (Sánchez, 2015).

Escenarios para la disciplina de requisitos:

La metodología propone cuatro escenarios para las disciplinas de requisitos. El presente trabajo estará regido por el escenario 3, su selección está basada en que dentro de sus características cumple, que se está desarrollando un módulo, el cual será integrado en un sistema más complejo, proporcionando objetividad, solidez, y su continuidad.

Escenario No 3:

Figura 2: Escenario No 3.

Al ser identificada la metodología a utilizar, así como las fases y disciplinas que rigen la misma, se hace necesario describir las herramientas y las principales tecnologías a utilizar en el proceso de desarrollo del software.

1.4 Lenguaje y herramienta de modelado

Lenguaje Unificado de Modelado v2.4: (Unified Modeling Language, UML v2.4) prescribe un conjunto de notaciones y diagramas estándares para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan; posibilitando así visualizar, especificar y documentar los artefactos o toda información que se obtiene o modifica

durante un proceso de desarrollo de software, además de poder utilizarse para modelar distintos tipos de sistemas de software, hardware y organizaciones del mundo real. (Larman, 2003)

Herramienta de modelado: para el modelado de los procesos de esta investigación se hace uso del Visual Paradigm v8.0. Es una herramienta para desarrollo de aplicaciones utilizando el Lenguaje Unificado de Modelado (*UML*) ideal para ingenieros de software, analistas de sistemas y arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. Ofrece un entorno de creación de diagramas para UML. El diseño es centrado en casos de usos y enfocado al negocio. Para la etapa de diseño del sistema se seleccionó Visual Paradigm por ser una herramienta de modelado multiplataforma que no se inclina por ninguna metodología específica, además de ser un estándar ampliamente utilizado en las empresas para el modelado de software (Visual Paradigm, 2017).

1.5 Ambiente de desarrollo

Se define el marco de trabajo que facilitara la creación de las nuevas funcionalidades a desarrollar.

Marco de trabajo (Framework): contiene numerosos archivos y directorios cuyo propósito es facilitar la creación de aplicaciones incorporando diferentes funcionalidades ya desarrolladas y probadas, esto para un determinado lenguaje de programación. (Orichijuan, 2014)

OpenObject v1.0

Marco de trabajo de código abierto, inteligente, profesional y rápido en el desarrollo de aplicaciones en Python. Está basado en la arquitectura modelo-vista-controlador, además de poseer Inteligencia de Negocios, Mapeador Relacional de Objetos(ORM), casos de pruebas, motores de flujos de trabajo, grabador de módulos, envases de módulos, entre otros. OpenObject ofrece, en un solo paquete el componente básico para la construcción de una aplicación de negocios: multilenguaje, servicios web, campos traducibles, ingeniería de reportes, PostgreSQL, Python como lenguaje de programación y licencia pública general de Affero (GNU AGPL v3) (Odoo, 2016). Se utilizó como framework OpenObject debido a que es el que utiliza Odoo.

Lenguajes de programación: Python v2.7.6

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, funcional. Es un lenguaje interpretado, legible, utiliza tipado dinámico, es multiplataforma y usa conteo de referencias para la administración de memoria (Python, 2016). Para el desarrollo de la aplicación se utilizó Python 2.7.6 debido a que es el lenguaje de programación que utiliza Odoo y que es un lenguaje bastante poderoso donde se consiguen buenos resultados con pocas líneas de código, permitiendo

generar programas multiplataforma que pueden ser ejecutados (con sus obvias dependencias) en cualquier sistema en que pueda instalarse un intérprete Python.

Lenguaje de Marcas Extensible XML 1.2: por sus siglas en inglés de Xtensible Markup Language (Lenguaje de marcas extensible), es un lenguaje de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible. Proviene del lenguaje SGML (por sus siglas en inglés de *Standard Generalized Markup Language*) y permite definir la gramática de lenguajes específicos (de la misma manera que HTML sigla en inglés de HyperText Markup Language o (lenguaje de marcas de hipertexto) es a su vez un lenguaje definido por SGML) para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones deben comunicarse entre sí o integrar información (XML, 2016). Para el desarrollo dela aplicación se utilizó XML 1.2 debido a que es el lenguaje de marcas que utiliza Odoo.

Herramientas de desarrollo propuestas para la confección de la solución:

Entorno de Desarrollo Integrado (IDE) **PyCharm 5.0.1**

PyCharm es un IDE o entorno de desarrollo integrado multiplataforma utilizado para desarrollar en el lenguaje de programación Python. Proporciona análisis de código, depuración gráfica, integración con VCS / DVCS y soporte para el desarrollo web con Django, entre otras tolerancias. Entre las características fundamentales que posee el PyCharm se encuentran el autocompletado, resaltador de sintaxis, herramientas de análisis y refactorización. Posee un depurador avanzado, además de la integración con lenguajes de plantillas como Mako, Jinja2, Django. Soporta entornos virtuales e intérpretes de Python 2.x, 3.x, PyPy, Iron Python y Jython. Posee también compatibilidad con SQLAlchemy (ORM), Google App Engine, Cython (Python, 2016).

Postgresql v9.3

Es un sistema de gestión de bases de datos relacional orientado a objetos, el cual incluye características como herencia, restricciones, tipos de datos, reglas e integridad transaccional. Tiene soporte total para transacciones, disparadores, vistas, procedimientos almacenados, almacenamiento de objetos de gran tamaño. Se destaca en ejecutar consultas complejas, consultas sobre vistas, sub-consultas y uniones. Permite la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Utiliza el modelo cliente servidor y es un manejador de base de datos de código abierto liberado bajo la licencia. Postgresql está diseñado para administrar grandes volúmenes de datos (PostgreSQL, 2016). Se utilizó como sistema gestor de base de datos debido a que es el que define Odoo.

Servidor web

Un servidor web o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor, realizando conexiones bidireccionales o unidireccionales y síncronas o asíncronas con el cliente y generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente. (Hostname, 2016)

Servidor Apache 2.4.9 es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual. El código fuente es totalmente abierto. Su arquitectura modular, construida sobre un pequeño núcleo, se adapta a las necesidades específicas de cada usuario (foundation, 2016). Se utilizó como servidor de aplicaciones Apache 2.2.9 porque es el que define Odoo.

Gestor de Base de Datos

PgAdmin 1.18.1

Es una aplicación gráfica para gestionar y administrar las bases de datos PostgreSQL. PgAdmin se diseña para responder a las necesidades de la mayoría de los usuarios, desde escribir simples consultas SQL hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y hace simple la administración. Está disponible en más de una docena de lenguajes y para varios sistemas operativos, incluyendo Microsoft Windows, Linux, Mac OSX y Solaris (PgAdmin, 2016).

1.6 Patrones de diseño

El patrón es una pareja de problema/solución con un nombre y que es aplicable a otros contextos, con una sugerencia sobre la manera de usarlo en situaciones nuevas. (Larman, 1999)

Existen dos clasificaciones en las que se agrupan, los patrones GRASP (General Responsibility Asignment Software Patterns) en español Patrones Generales de Software para Asignar Responsabilidades, los cuales describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones; los patrones GOF (Gang Of Four) que solucionan problemas de creación de instancias. Los mismos se pueden clasificar según su propósito en:

- **Creacionales:** Corresponden a patrones de diseño de software que solucionan problemas de creación de instancias. Nos ayudan a encapsular y abstraer dicha creación.
- **Estructurales:** Son los patrones de diseño software que solucionan problemas de composición (agregación) de clases y objetos.
- **Comportamiento:** Se definen como patrones de diseño software que ofrecen soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan.

- **Interacción:** Los patrones de interacción buscan la reutilización de interfaces eficaces y un manejo óptimo de los recursos de las páginas web, haciendo más eficaz el consumo de tiempo en el diseño del sitio web y permitiendo a los programadores novatos adquirir más experiencia. (Alpízar, y otros, 2017)

1.7 Técnicas de validación

Para garantizar la calidad del análisis y el diseño se proponen un conjunto de métricas que permitirán medir la calidad de la especificación de los requisitos y del diseño propuesto en la solución, así como evaluar la calidad de los atributos internos del sistema.

1.7.1 Validación de Requisitos

La validación de requisitos trata de mostrar que estos son realmente definen el sistema que el cliente desea. (Sommerville, 2011)

Con el objetivo de verificar si los requisitos del software obtenidos definen el sistema que el cliente desea, se llevará a cabo un proceso de validación, para el cual se emplearán las siguientes técnicas.

- **Revisión técnica formal**: En este punto los requisitos son analizados sistemáticamente por un equipo de revisores. Con el objetivo de verificar, comprender y controlar los cambios en los requisitos del sistema. (Sommerville, 2011)
- **Prototipado del software:** En este enfoque de validación, se muestra un modelo ejecutable del sistema a los usuarios finales y a los clientes, así estos pueden experimentar con este modelo para ver si cumple con sus necesidades reales. (Sommerville, 2011)

1.7.2 Validación del diseño

Para la validación del diseño se usarán las métricas basadas en clases: Tamaño Operacional de Clases (TOC) y Relaciones entre Clases (RC). El resultado de aplicar ambas métricas permitirá la validación del diseño propuesto en la investigación.

Tamaño Operacional de Clases (TOC): consiste en un cálculo de los atributos u operaciones totales que se realizan en las clases, esta métrica mide el grado de responsabilidad, complejidad y reutilización que poseen las clases.

- **Responsabilidad:** Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
- **Complejidad de implementación:** Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
- **Reutilización:** Un aumento del TOC implica una disminución del grado de reutilización de la clase.

Relaciones entre Clases (RC): está dada por el número de relaciones de uso de una clase con otras. La aplicación de dicha métrica permite evaluar atributos como el Acoplamiento, la Complejidad de mantenimiento, la Reutilización y la Cantidad de pruebas.

- **Acoplamiento:** Un aumento del RC implica un aumento del Acoplamiento de la clase.
- **Complejidad de mantenimiento:** Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
- **Reutilización:** Un aumento del RC implica una disminución en el grado de reutilización de la clase.

  Cantidad de pruebas: Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar una clase.

1.8 Pruebas de Calidad de Software

Las pruebas son un conjunto de actividades que se planean con anticipación y se realizan de manera sistemática con el objetivo de detectar errores en el software. (Pressman, 2010)

Hay dos maneras de probar cualquier producto construido: uno si se conoce la función específica para la que se diseñó el producto, se aplican pruebas que demuestren que cada función es plenamente operacional, mientras que se buscan los errores de cada función; dos si se conoce el funcionamiento interno, se aplican pruebas para asegurarse que “todas las piezas encajan”, es decir que las operaciones internas se realizan de acuerdo con las especificaciones, y se han probado todos los componentes internos de manera adecuada. Al primer enfoque de prueba se le denomina prueba de caja negra, y al segundo prueba de caja blanca. (Pressman, 2010)

1.8.1 Pruebas Unitarias

En programación, una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. (debugmodeon, 2017)

Para darle cumplimento a lo anteriormente planteado se decide aplicar las pruebas de caja blanca.

Pruebas de caja blanca

La prueba de caja blanca del software se basa en un examen cercano al detalle procedimental. Se prueban las rutas lógicas del software y la colaboración entre componentes, al proporcionar casos de pruebas que ejerciten conjuntos específicos de condiciones, bucles o ambos. Al emplear los métodos de prueba de caja blanca el ingeniero del software podrá derivar casos de prueba que:. (Pressman, 2010)

- garanticen que todas las rutas independientes dentro del módulo se han ejercido al menos una vez.
- ejerciten los lados verdadero y falso de todas las decisiones lógicas.
- ejecuten todos los bucles en sus límites y dentro de sus límites operacionales.
- ejerciten estructuras de datos internos para asegurar su validez.

Técnica de prueba: Ruta básica

La técnica del camino básico es empleada en las pruebas de caja blanca, la misma tiene como objetivo comprobar que cada camino se ejecute independiente de un componente o programa, obteniéndose una medida de la complejidad lógica del diseño. Para comprobar que cada sentencia de código se ejecuta al menos una vez, se realizan pruebas al código de las funcionalidades más complejas desde el punto de vista de la programación. (Pressman, 2010)

Para obtener los casos de prueba a partir la técnica seleccionada se debe construir el grafo de flujo correspondiente al código de la funcionalidad. Luego se determina la complejidad ciclomática V (G) del grafo resultante, la cual es un indicador del número de caminos independientes que existen en un grafo, es decir, es cualquier camino dentro del código que introduce por lo menos un nuevo conjunto de sentencias de proceso o una nueva condición. (Pressman, 2010)

Esta métrica se calcula sobre un grafo y se puede realizar mediante tres formas distintas:

- V (G) = E – N + 2, donde E es el número de aristas, y N el número de nodos de la gráfica de flujo.
- V(G) = R, donde R es el número de regiones que favorece a estimar el valor de la complejidad ciclomática.
- V (G) = P + 1, donde P es el número de nodos predicados incluidos en el grafo.

1.8.2 Método de caja negra

Las pruebas de caja negra, también denominadas, pruebas de comportamiento, se concentran en los requisitos funcionales del software. Permiten derivar conjuntos de condiciones de entrada que ejercitarán por completo todos los requisitos funcionales del programa.

La prueba de caja negra intenta encontrar errores de las siguientes categorías (Pressman, 2010):

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

Técnica de partición de equivalencia.

La partición equivalente es un método 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. (Pressman, 2010) Los casos de pruebas que se diseñan para este tipo de técnica se basan en una evaluación de las clases de equivalencia para una condición de entrada.

Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada. (Pressman, 2010)

Las clases de equivalencia se definen de acuerdo a las siguientes directrices (Pressman, 2010):

- Si una condición de entrada especifica un rango, se define una clase de equivalencia válida y dos no válidas.
- Si una condición de entrada requiere un valor específico, se define una clase de equivalencia válida y una no válida.
- Si una condición de entrada especifica un miembro de un conjunto, se define una clase de equivalencia válida y una no válida.
- Si una condición de entrada es lógica, se define una clase de equivalencia válida y una no válida

Al aplicar estas directrices para la derivación de clases de equivalencia, se desarrollarán y se ejecutarán los casos de prueba para cada objeto de los datos del dominio de entrada.

1.9 Validación mediante criterios de expertos: Método Delphy.

El Delphy es una metodología estructurada para recolectar sistemáticamente juicios de expertos sobre un problema, procesar la información y a través de recursos estadísticos, construir un acuerdo general de grupo (García Valdés, y otros, 2013).

Este método es considerado como uno de los métodos subjetivos de pronosticación más confiables. El procesamiento estadístico y matemático de la información es la característica más importante del método que lo diferencia del resto de los métodos de pronosticación de base subjetiva, ya que la decisión final que toma el investigador es un criterio fuertemente avalado por la experiencia y conocimiento del colectivo consultado, así como por indicadores objetivos (Win2PDF, 2015).

Para la aplicación de este método se deben tener en cuenta dos aspectos fundamentales, el primero la selección de los expertos y el segundo la elaboración y análisis del cuestionario a aplicar. Atendiendo a estos aspectos (García Valdés, y otros, 2013) plantea las siguientes fases:

I. Preparatoria

- Selección de expertos.
- Preparación del instrumento.
- Decisión de la vía de consulta.

II. **Consulta**

- Rondas de consulta.
- Procesamiento estadístico.
- Retroalimentación.

III. **Consenso**

- Construcción de consenso.
- Reporte de resultados.

Por su importancia se realizará énfasis en la fase de consulta, específicamente en el procesamiento estadístico, esta fase incluye el tratamiento matemático y estadístico del cuestionario aplicado. Se propone por (Win2PDF, 2015) el uso del siguiente procedimiento:

1. Determinación de la media aritmética por pregunta:

$$\overline{Cj} = \frac{\sum_{i=1}^{mj} Cij}{mj}$$

Donde:

m: cantidad de expertos.

n: cantidad de preguntas.

mj: cantidad de expertos que evalúan la pregunta j.

Cij: evaluación en puntos de la pregunta j realizada por el experto i.

2.Grado de concordancia de los expertos por pregunta:

$$\sigma_j^2 = \frac{\sum_{i=1}^{mj}(Cij - \overline{Cj})^2}{mj - 1}$$

A partir de la fórmula anterior para determinar la varianza, se determina el coeficiente de variación (Vj), este coeficiente es una medida del grado de concordancia de los expertos por cada pregunta, donde mientras mayor sea el valor de Vj menor será el grado de concordancia de los expertos.

$$Vj = \frac{\sqrt{\sigma_j^2}}{Cj}$$

Conclusiones del capítulo

El estudio realizado a los diferentes sistemas proporcionó un conjunto de funcionalidades a tener en cuenta en el desarrollo del módulo para la gestión de las ventas en la Dirección de Transferencia de Tecnología. Este módulo se desarrollará con tecnologías y herramientas definidas por el proyecto al que pertenece esta investigación. Se utilizará la metodología Variación de AUP para la UCI para

guiar el proceso de desarrollo. Además, se estudiaron las métricas de validación del diseño y los diferentes métodos de pruebas que serán aplicadas en la validación de la investigación y la solución.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

Introducción

En este capítulo se describen los artefactos de trabajo propuestos por AUP-UCI para el desarrollo de la solución propuesta en las fases de: requisitos, análisis y diseño. En la etapa de requisitos se identifican y describen los requisitos funcionales, no funcionales. En el análisis y el diseño, se detallan los principales artefactos de dicha disciplina, destacando el modelo de conceptual, los diagramas de clases, así como el modelo de datos. Igualmente, se especifica la utilización de los patrones de arquitectura y de diseño para la concepción de la solución.

2.1 Disciplina Modelado de negocio

El Modelado del Negocio es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito (Sánchez, 2015).

2.1.1 Descripción de los procesos de negocio

La descripción de los procesos del negocio consiste en describir la realidad de manera que esta pueda ser entendida y de ser necesario modificada con el fin de incorporarle mejoras.

En la Dirección de Transferencia de Tecnología se reciben las ofertas técnicas una vez que se tienen, se les realiza un análisis, asignándoles los hitos de pago y especificaciones necesarias para convertirlas en ofertas técnicas comerciales. Dichas ofertas son enviadas al director del centro para su aprobación. Al ser aprobadas son enviadas a los clientes, los cuales pueden interesarse en dicha propuesta notificando a los comerciales, los cuales comenzarían un proceso de contratación con los abogados marcando las pautas y términos legales necesarios para el cumplimiento de la venta. La cual culminaría luego de la firma del contrato, una de las partes involucradas (cliente) recibiría el proyecto o servicio contratado y la otra facturaría el monto pactado en el contrato.

En la figura 3 que se presenta a continuación, se describen los pasos del negocio enunciados anteriormente en el proceso de venta:

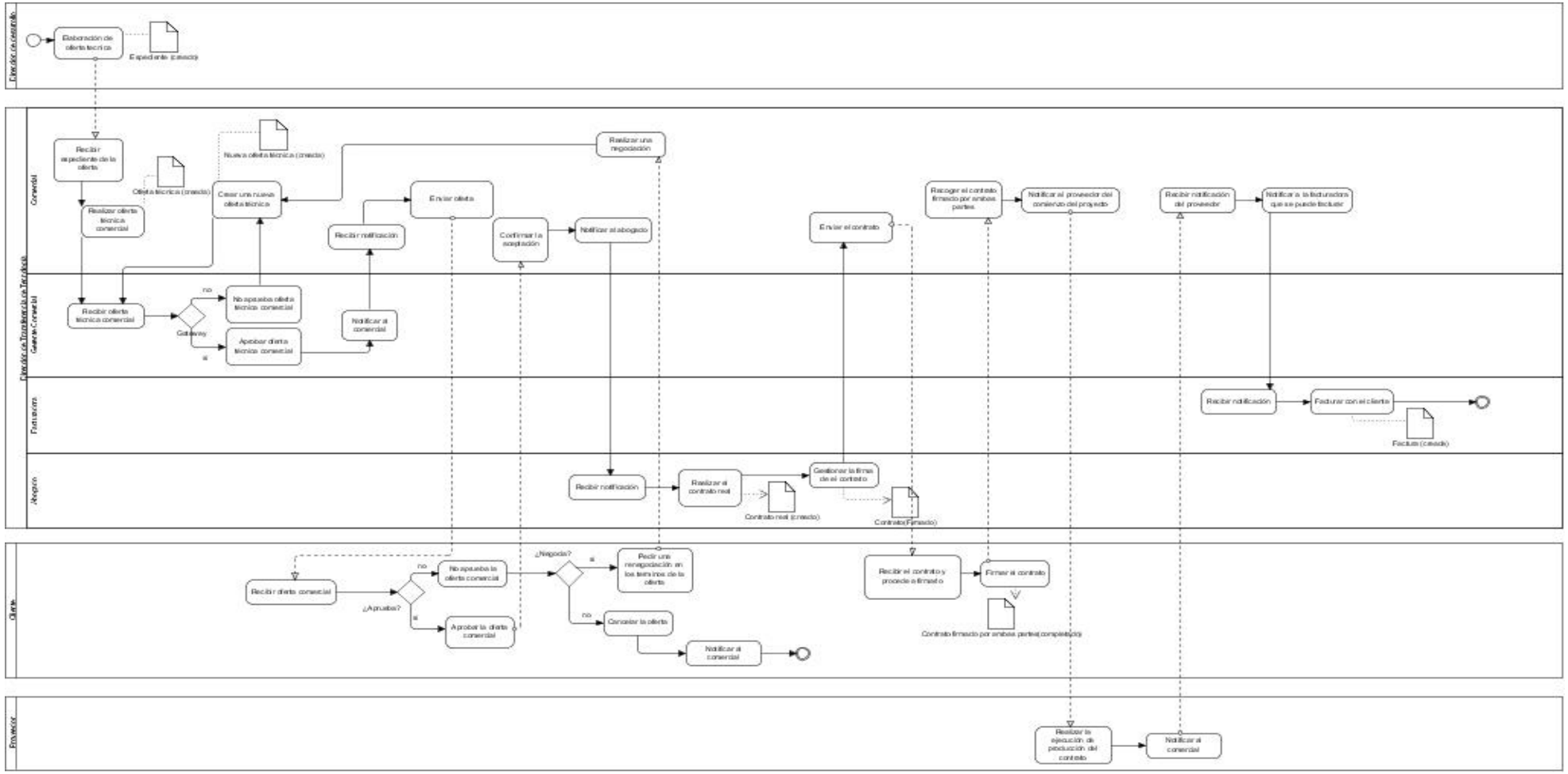


Figura 3: Diagrama de procesos de negocio de Gestión de Ventas

2.1.2 Modelo conceptual

Un modelo conceptual representa la estructura y dinámica de la organización, en este caso se representarán los principales conceptos, los tipos más importantes de objetos en el contexto del negocio, los cuales representan lo que existe y los eventos que suceden en el entorno de trabajo del sistema. (Pressman, 2010)

En la Figura 4 se muestra el modelo conceptual del sistema.

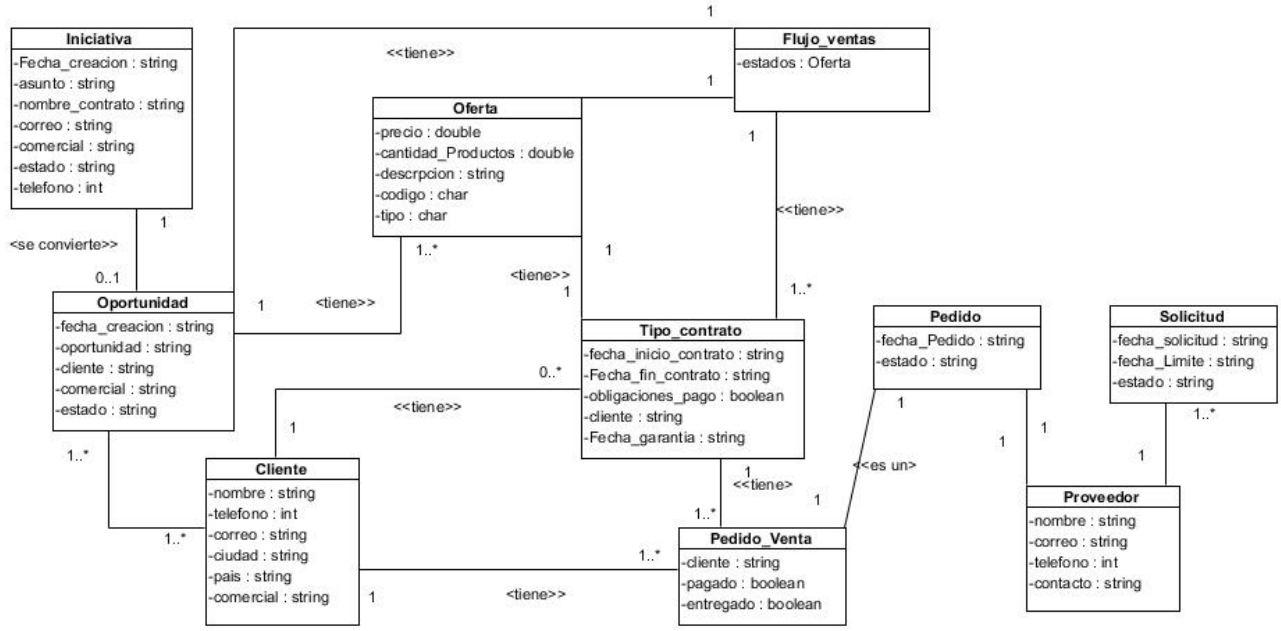


Figura 4: Modelo conceptual del sistema.

Fuente: Elaboración propia.

La información sobre el Diccionario de datos y el Modelo conceptual anteriormente abordado se encuentra en el documento entregable "*Modelo conceptual. odt"*

2.2 Disciplina de Requisitos

El esfuerzo principal en la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto (Sánchez, 2015).

Para la definición de los requisitos funcionales del sistema se hace necesario recopilar información detallada de las funcionalidades que se quieren implementar para que el sistema cumpla las necesidades del cliente.

2.2.1 Técnicas de identificación de requisitos

El propósito general de la captura de requisitos es obtener una descripción correcta de lo que debe hacer el sistema y delimitar su alcance. Los requisitos juegan un papel importante durante el ciclo de vida de un proyecto, ya que establece las características que debe poseer el sistema, así como las condiciones que debe cumplir. Los requisitos se clasifican en funcionales y no funcionales (Pressman, 2002), las técnicas utilizadas son:

Entrevista

La entrevista es de gran utilidad para obtener información cualitativa como opiniones, o descripciones subjetivas de actividades. Se entrevistó al especialista superior del Departamento Comercial de la Dirección de Transferencia de Tecnología en la UCI para un mejor entendimiento de los procesos del negocio (ver Anexo 1).

Tormenta de ideas

Es una herramienta de trabajo grupal que facilita el surgimiento de nuevas ideas sobre un tema o problema determinado. Se utilizó la tormenta de ideas para la captura de requisitos, la agrupación de ellos en procesos de negocio y su esclarecimiento.

A continuación, se definen los requisitos funcionales identificados luego de aplicar las técnicas anteriores.

2.2.2 Requisitos funcionales

A continuación, en la Tabla 4, se nombran los requisitos funcionales del proceso de Ventas del Sistema de Administración de Relaciones con el Cliente en Odoo.

Tabla 2: Agrupamiento de Requisitos Funcionales.

Requisitos	Descripción
Agrupación de requisito: Gestionar Cliente	
RF1.Crear cliente.	Se adiciona los datos de un cliente.
RF2.Buscar cliente existente.	Se buscan los clientes.
RF3.Listar clientes.	Se listan los clientes.
RF4.Eliminar cliente.	Se elimina un cliente.
RF5.Modificar datos de clientes.	Se modifica los datos de un cliente.
RF6.Ver detalles de clientes	Se muestra el detalle del cliente.
RF7.Imprimir datos de cliente.	Se imprime los datos de los clientes.

Agrupación de requisito: Gestionar Contrato	
RF8.Crear contrato.	Se adiciona los datos de un contrato.
RF9.Buscar contrato.	Se buscan los contratos.
RF10.Listar contrato.	Se listan los contratos.
RF11.Eliminar contrato.	Se elimina un contrato.
RF12.Modificar contrato.	Se modifica los datos de un contrato.
RF13.Ver detalles del contrato.	Se muestran el detalle del contrato.
RF14.Imprimir datos del contrato.	Se imprime los datos de los contratos.
Agrupación de requisito: Gestionar Oferta Técnica Comercial	
RF15.Crear oferta técnica comercial.	Se adiciona los datos de una oferta.
RF16.Listar oferta técnica comercial.	Se buscan las ofertas.
RF17.Eliminar oferta técnica comercial.	Se listan las oferta.
RF18.Modificar oferta técnica comercial.	Se elimina una oferta.
RF19.Buscar oferta técnica comercial.	Se modifica los datos de una oferta.
RF20.Ver detalles de la oferta técnica comercial.	Se muestran el detalle de la oferta.
RF21.Imprimir datos de la oferta técnica comercial.	Se imprime los datos de las ofertas.
Agrupación de requisito: Gestionar Proveedor	
RF22.Crear proveedores.	Se adiciona los datos de un proveedor.
RF23.Buscar proveedores.	Se buscan los proveedores.
RF24.Listar proveedores.	Se listan los proveedor.
RF25.Eliminar proveedor.	Se elimina un proveedor.
RF26.Modificar datos de proveedor.	Se modifica los datos de un proveedor.
RF27.Ver detalles de proveedor.	Se muestran el detalle de un proveedor.
RF28.Buscar proveedor.	Se imprime los datos de un proveedor.
Agrupación de requisito: Gestionar Oportunidad	
RF29.Crear oportunidad.	Se adiciona los datos de una oportunidad.
RF30.Modificar oportunidad.	Se modifica una oportunidad.
RF31.Buscar oportunidad existente .	Se buscan las oportunidades.
RF32.Listar oportunidad.	Se listan las oportunidad.
RF33.Ver detalles de la oportunidad.	Se muestran el detalle de una oportunidad.

RF34.Eliminar oportunidad.	Se elimina una oportunidad.
Agrupación de requisito: Gestionar Iniciativa	
RF35.Crear iniciativa	Se adiciona los datos de una iniciativa.
RF36.Modificar iniciativa	Se modifica una iniciativa.
RF37.Buscar iniciativa existente	Se buscan las iniciativas.
RF38.Listar iniciativa	Se listan las iniciativas.
RF39.Ver detalles de la iniciativa.	Se muestran el detalle de una iniciativa.
RF40.Eliminar iniciativa	Se elimina una iniciativa.
Agrupación de requisito: Pedido de venta	
RF41.Crear pedido de venta	Se adiciona los datos de un pedido.
RF42.Listar pedido de venta	Se listan los pedidos.
RF43.Eliminar pedido de venta	Se elimina un pedido.
RF44.Modificar pedido de venta	Se modifica un pedido.
RF45.Buscar pedido de venta	Se buscan los pedidos.
RF46.Ver detalles del pedido de ventas.	Se muestran el detalle de un pedido.

La información sobre las agrupaciones de requisitos de software anteriormente abordada se encuentra en el documento entregable "*Especificación de requisitos de software. odt"*

2.2.3 Requisitos no funcionales

Los requisitos no funcionales son restricciones que afectan a los servicios o funciones del sistema, tales como restricciones de tiempo, sobre el proceso de desarrollo, estándares (Pressman, 2002). Estos requisitos especifican o restringen las propiedades emergentes del sistema, pueden especificar el rendimiento, la protección, la disponibilidad y otras propiedades del sistema (Sommerville, 2005).

El incumplimiento de un requerimiento no funcional puede significar que el sistema entero sea inutilizable ya que estos son más críticos en ocasiones que los requisitos funcionales debido a que los usuarios normalmente pueden encontrar formas de trabajar alrededor de una función del sistema que realmente no cumple sus necesidades, pero no ante la falla de un requerimiento no funcional que sea crítico para el funcionamiento del sistema (Sommerville, 2005).

Se identificaron 18 requisitos no funcionales los cuales para un mejor entendimiento se agruparon por las siguientes clasificaciones:

Requisitos de Software

- Un servidor de base de datos PostgreSQL en su versión 9.1 o superior.
- Computadoras clientes con un navegador web (Firefox 40.0 o superior).

Requisitos de Hardware

- En el cliente se requiere una computadora con 256 MB de RAM como mínimo, procesador Intel® a 1 GHz de velocidad de procesamiento o superior, tarjeta de red Ethernet.
- Todas las máquinas implicadas en la funcionalidad de la aplicación deben estar conectadas a la red.
- El servidor requiere mínimo 1024 MB de RAM, procesador a 3 GHz de velocidad y tarjeta de red Ethernet.

Requisitos de Seguridad

- El sistema podrá ser utilizado solamente por usuarios autenticados en el mismo.
- La autenticación será la primera acción en el sistema, proporcionándole los privilegios concebidos al usuario, de forma tal que la información sensible sea vista y manejada por la persona adecuada.
- El administrador es el único que tiene el control total del sistema.
- Los cambios en el sistema sólo pueden ser realizados por los usuarios con los permisos requeridos para esta acción.
- La información solo podrá ser vista por los usuarios con el nivel de acceso requerido para ello; permitiendo que las funcionalidades del sistema se muestren de acuerdo al rol del usuario que esté activo.

Interfaz

- Tiene que poseer una interfaz amigable y de fácil entendimiento para el cliente.
- El sistema deberá ser independiente de la plataforma.
- Los colores serán estándares en toda la aplicación.

Usabilidad

- El sistema deberá poseer una interfaz web sencilla, lo más atractiva y clara posible para el usuario, además de poder ser usado por cualquier persona con conocimientos básicos en el manejo de la computadora y el entorno Web en sentido general.
- El software tendrá en su portada inicial las políticas de uso concebidas para hacer uso de él por cualquier usuario.

Portabilidad

- El sistema estará desarrollado con tecnologías multiplataforma permitiendo que el servidor sea instalado tanto en Windows, Mac OS como en Linux, llevando a cabo esta operación sin necesidad de efectuar cambios significativos.

Disponibilidad

- El sistema podrá ser usado en cualquier momento por todos los usuarios autorizados.
- Para poder hacer uso del sistema deberá existir conexión.

La información sobre los requisitos no funcionales de software anteriormente abordada se encuentra en el documento entregable Especificación de requisitos de software. Odt

2.3 Técnicas utilizadas para la validación de requisitos

Dando cumplimiento a las técnicas de validación propuestas en el Capítulo 1 para validar los requisitos y el diseño propuesto a continuación se muestran los resultados obtenidos en cada una de las técnicas aplicadas.

Revisión técnica formal: se realizaron revisiones a cada uno de los requisitos por parte del equipo de desarrollo conjuntamente con el cliente. En una primera revisión se generaron un total de 4 no conformidades, las cuales fueron corregidas satisfactoriamente y en tiempo. Con el cliente se desarrollaron 2 revisiones. En la primera, el mismo quedó satisfecho con el trabajo efectuado por el equipo de desarrollo, sin embargo, se modificaron detalles a 6 de los requisitos funcionales para su perfeccionamiento y se añadieron 12 requisitos. En el segundo encuentro se aprobaron todos los requisitos, generándose de este encuentro el Acta de aceptación por parte del cliente (ver Anexo 2).

Construcción de prototipos: mediante los prototipos se le mostró al cliente un modelo estático del sistema que le permitió tener una visión preliminar del funcionamiento, se comprobó si satisfacía sus necesidades, los mismos fueron aprobados por el cliente con un alto nivel de satisfacción. Se utilizó Visual Paradigm v8.0 para su construcción. Los mismos se encuentran en la carpeta de proyecto en los documentos entregables “Descripción de la agrupación de requisitos Gestionar Cliente.odt”

2.4. Descripción de requisitos por procesos

La descripción de los requisitos es una versión completa del comportamiento del sistema que se va a desarrollar.

Gestionar Contratos

En el requisito se gestionan todos los contratos que tiene la entidad con el objetivo de conocer la situación de los posibles clientes para crear oportunidades de negocio en el mercado.

Descripción del requisito Adicionar contrato

Tabla 3: Descripción de requisito Adicionar contrato.

Precondiciones		El usuario se ha autenticado en el sistema y tiene permiso para realizar esta acción.
Flujo de eventos		
Flujo básico Adicionar		
1	Se registran los datos: 1. Fecha inicio del contrato 2. Fecha de garantía 3. Fecha fin del contrato 4. Cliente 5. Obligaciones de pago 6. Precio total 7. Hitos de pago	
2	El sistema valida (ver validación 1) los datos introducidos.	
3	Si los datos son correctos el sistema los registra.	
4	El sistema confirma el registro de los datos.	
5	Concluye el requisito.	
Pos-condiciones		
1	Se ha registrado un nuevo contrato.	
Flujos alternativos		
Flujo alternativo 3.a Información errónea		
1	El sistema señala los datos erróneos y permite corregirlos.	
2	El usuario corrige los datos.	
3	Volver al paso 2 del flujo básico.	
Pos-condiciones		
	N/A	
Flujo alternativo 3.b Información incompleta		
1	El sistema señala los datos vacíos y permite corregirlos.	
2	El usuario corrige los datos.	
3	Volver al paso 2 del flujo básico.	
-condiciones		
1	N/A	
Flujo alternativo *.a El usuario cancela la acción		
1	Concluye el requisito.	
-condiciones		
1	No se registran los datos	
Validaciones		
2	Se validan los datos según lo establecido en el Modelo conceptual .Descripción de la agrupación de requisitos Gestionar Contrato.doc.	
Conceptos	**Contrato**	Visibles en la interfaz: Se registran los datos: 1. Fecha inicio del contrato 2. Fecha de garantía 3. Fecha fin del contrato 4. Cliente 5. Obligaciones de pago 6. Precio total 7. Hitos de pago Hitos de pago en MN Hitos de pago en CUC

		Hitos de pago en USD
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Prototipo elemental de interfaz gráfica de usuario

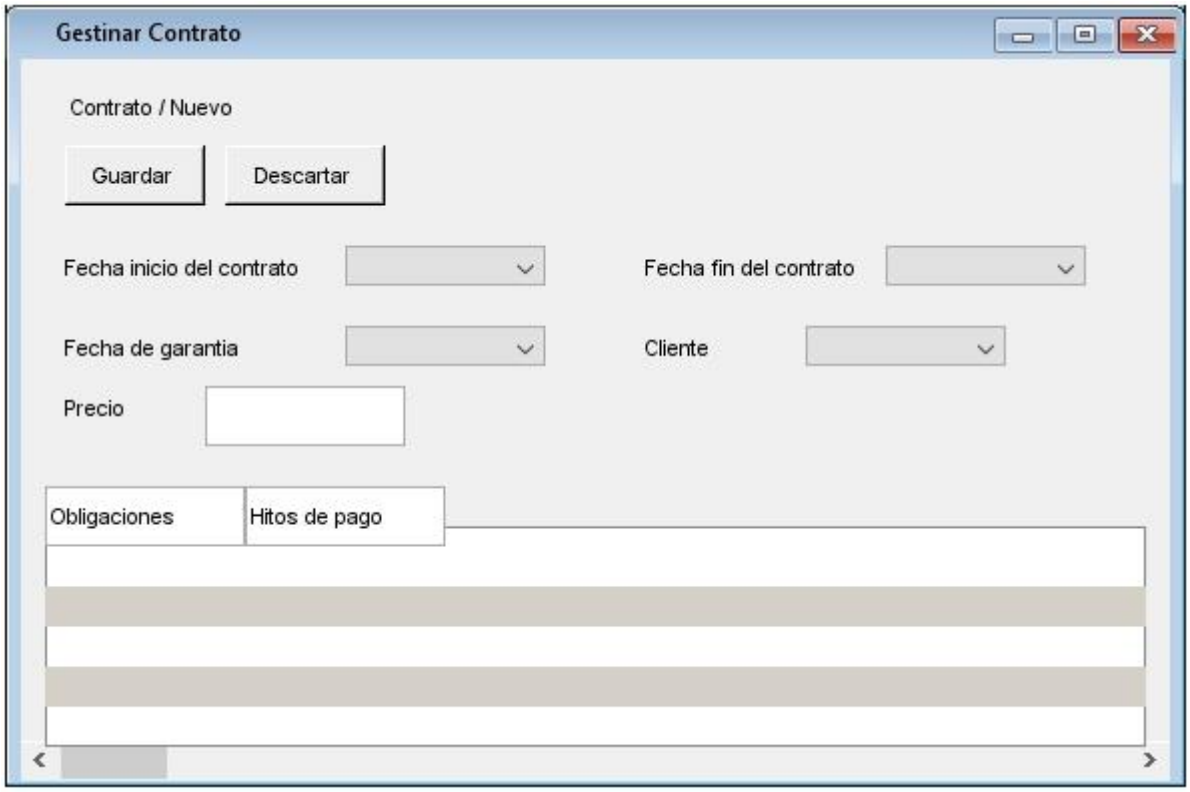


Figura 5: Adicionar contrato.

La información correspondiente a la descripción del requisito anteriormente abordado se encuentra en el documento entregable *"Descripción de la agrupación de requisitos Gestionar Contrato.odt"*. Los requisitos funcionales del módulo de Ventas se encuentran descritos en los documentos entregables:

- *"Descripción de la agrupación de requisitos Gestionar Cliente.odt"*
- *"Descripción de la agrupación de requisitos Gestionar Oportunidad.odt"*
- *"Descripción de la agrupación de requisitos Gestionar Oferta Técnica Comercial.odt"*
- *"Descripción de la agrupación de requisitos Gestionar Pedido de venta.odt"*

- *"Descripción de la agrupación de requisitos Gestionar Proveedor.odt"*
- *"Descripción de la agrupación de requisitos Gestionar Iniciativa.odt"*

2.5 Disciplina Diseño

En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura). Además, en esta disciplina se modela el sistema y su forma (incluida su arquitectura) para que soporte todos los requisitos, incluyendo los requisitos no funcionales. Los modelos desarrollados son más formales y específicos que el de análisis (Sánchez, 2015).

Diseño de la propuesta de solución

Durante el desarrollo de esta disciplina se modela el sistema, teniendo en cuenta la arquitectura, para que soporte todos los requisitos tanto funcionales como no funcionales. De este modo se propicia el desarrollo de una arquitectura sólida para el sistema y la adaptación del diseño para que sirva de base a la etapa de implementación.

2.5.1 Diseño arquitectónico

Los patrones arquitectónicos, o patrones de arquitectura, son patrones de software que ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Dan una descripción de los elementos y el tipo de relación que tienen junto con un conjunto de restricciones sobre cómo pueden ser usados. Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones (Parra, 2006).

El modelo-vista-controlador (MVC) es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario (Parra, 2006). La siguiente figura muestra cómo se ve la aplicación de este patrón en un fragmento del diagrama de clases del diseño:

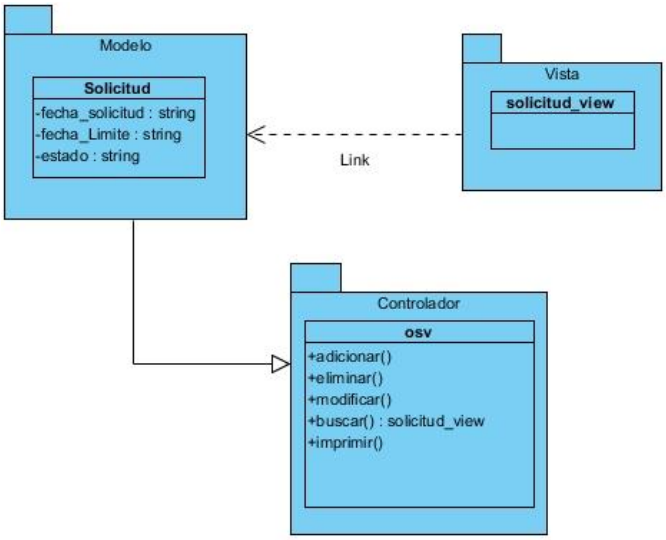


Figura 6: Modelo-vista-controlador.

Modelo: Los modelos se encuentran dentro de la carpeta models, divididos en tres componentes: configuración, ventas, e informes. El mapeo de la base de datos es gestionado automáticamente por Odoo, y el mecanismo responsable por esto es el modelo objeto relacional, (ORM –object relational model).

Vista: Las vistas en Odoo manejan la presentación visual de los datos representados por el Modelo a través de archivos xml. Se encuentran almacenadas dentro de la carpeta views, divididos en cuatro carpetas: configuración, ventas, compras e informes.

Controlador: En Odoo el controlador se manifiesta a través del modelo controller.py, presente en el archivo base_import, el cual es el encargado de hacer peticiones al modelo cuando se hace alguna solicitud de la información por parte del cliente.

Odoo está basado en un patrón clásico del diseño web conocido como arquitectura MVC. El sistema a desarrollar respeta dicha arquitectura, siendo el controlador el encargado de gestionar el flujo de la petición realizada a través de la clase base_import, dando al modelo la opción del negocio que el usuario solicita (ejemplo contratos.py), y devuelve como respuesta una representación del modelo a través de la vista (ejemplo contrato.xml), guardando o accediendo a datos mediante un mapeo a la base de datos. La siguiente imagen ilustra la cooperación entre estos tres objetos dentro de la solución propuesta.

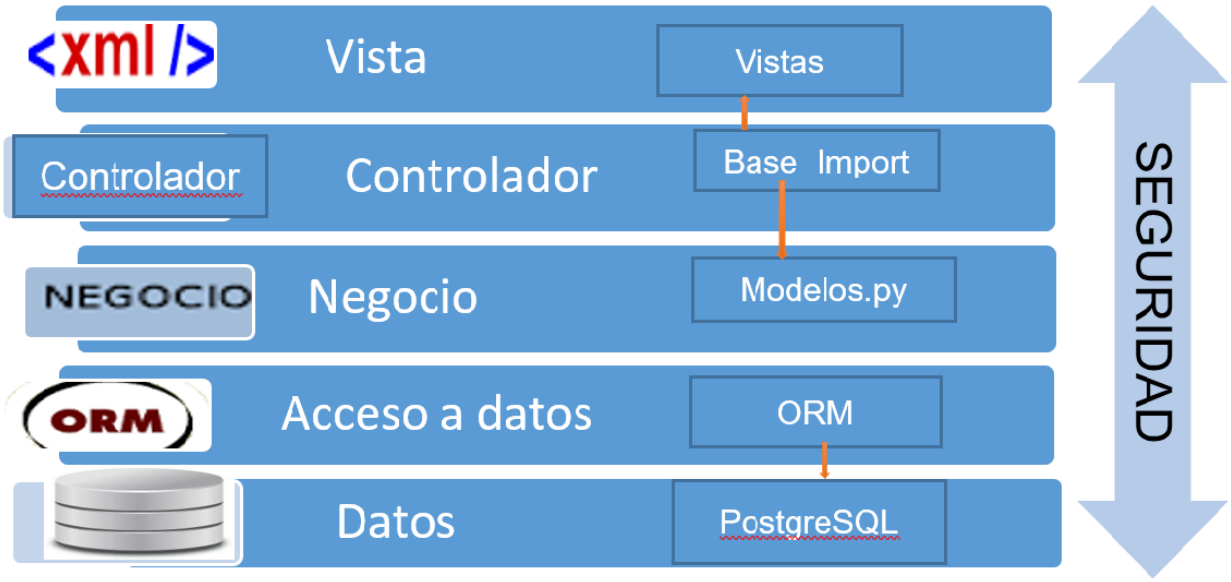


Figura 7: Descripción ampliada del MVC.

2.5.2 Patrones del diseño

Patrones GRASP

Los patrones utilizados para la asignación de responsabilidades son (Larman, 2003):

Experto: Asigna una responsabilidad a la clase que tiene la información necesaria para realizar la responsabilidad (Larman, 2003).

Creador: Asignar a la clase B la responsabilidad de crear una instancia de clase A si se cumple uno o más de los casos siguientes (Larman, 2003):

- B agrega objetos de A.
- B contiene objetos de A.
- B registra instancias de objetos de A.
- B utiliza más estrechamente objetos de A.

Bajo acoplamiento: El acoplamiento es una medida de la fuerza con que un elemento está conectado a, tiene conocimiento de, confía en, otros elementos. Un elemento con bajo (o débil) acoplamiento no depende de demasiados otros elementos. Estos elementos pueden ser clases, subsistemas y sistemas (Larman, 2003).

Alta cohesión: la cohesión (o de manera más específica, la cohesión funcional) es una medida de la fuerza con la que se relacionan y del grado de focalización de las responsabilidades de un

elemento. Un elemento con responsabilidades altamente relacionadas, y que no hace una gran cantidad de trabajo, tiene alta cohesión. Estos elementos pueden ser clases y subsistemas (Larman, 2003)

En la Figura 8 se muestra como la clase contrato tiene una alta cohesión debido a su alto grado de focalización en las responsabilidades de un elemento.

Controlador: Asignar la responsabilidad de controlar el flujo de eventos del sistema (Larman, 2003). En la Figura 8 se muestra como la clase import_controller es la encargada de controlar los eventos del sistema, controlando tanto las vistas, como los modelos.

El siguiente diagrama de clases muestra la presencia de los patrones Controlador, Alta cohesión y Bajo acoplamiento:

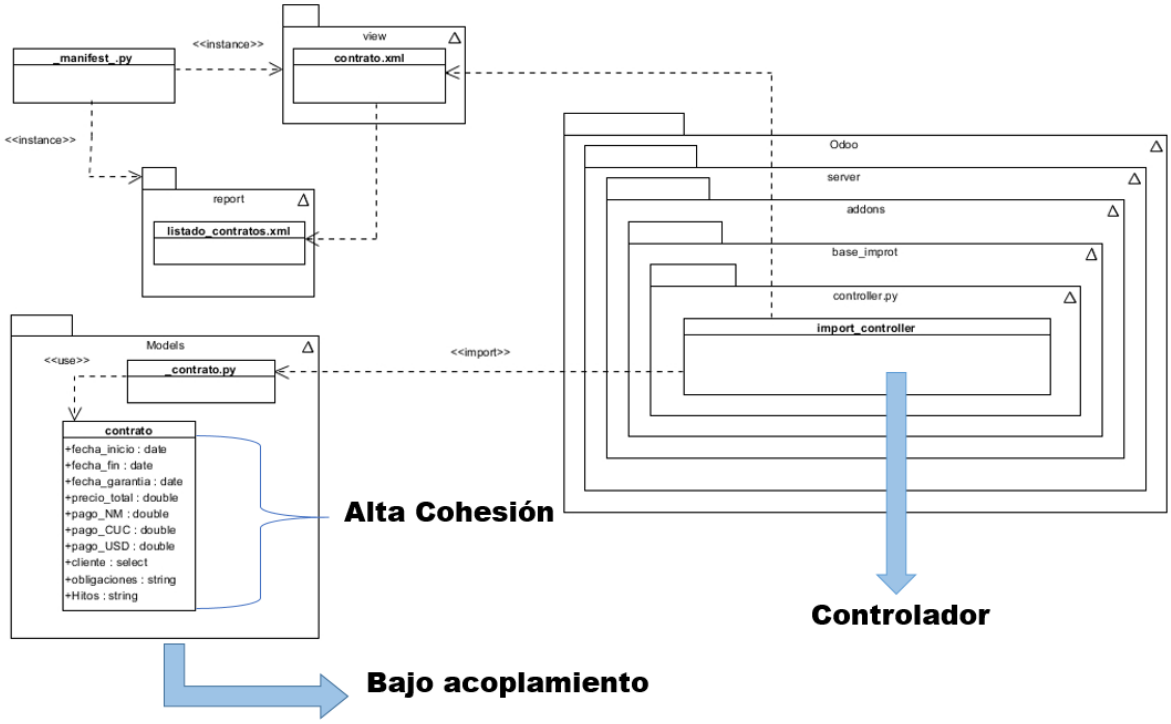


Figura 8: Diagrama de clases del Diseño Gestionar Contrato.

El uso del patrón experto se evidencia en el desarrollo del sistema en las entidades, siendo cada una de ellas experta en gestionar la información que les compete, por tanto, las encargadas de suministrarle toda información necesaria a las clases controladoras.

A continuación, se muestra la clase **Contrato** la cual es experta en información referente a un contrato.

Figura 9: Diagrama de clases. Clase contrato

2.6 Modelo de datos

Un diagrama o modelo entidad-relación (DER) es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información, así como sus interrelaciones y propiedades. El propósito fundamental es representar objetos de datos y sus relaciones. (Pressman, 2010). La base de datos se encuentra en tercera forma normal pues sus atributos dependen directamente de la clave primaria, o sea no se relacionan a través de otros atributos, eliminando de esta manera la dependencia transitiva. Con el objetivo de garantizar la persistencia de los datos se modeló y normalizó el DER del sistema para la gestión del proceso Venta. De estos se muestra seguidamente una porción, que demuestra de forma general como se garantiza la persistencia de datos.

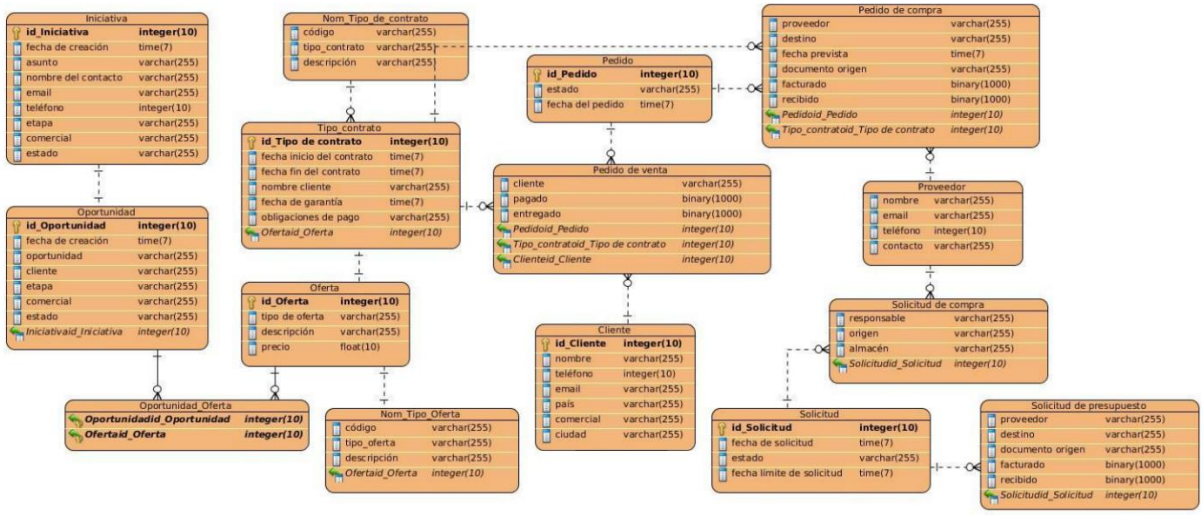


Figura 10: Modelo entidad-relación.

Descripción de las tablas de datos

iniciativa: contiene la información de la iniciativa registrada permitiendo conocer la fecha en que fue creada, en qué estado se encuentra y quien es el comercial responsable de la misma.

oportunidad: contiene la información de las oportunidades registradas permitiendo conocer fecha de creación, cliente al cual se le asocia dicha oportunidad y el comercial responsable del proceso.

oferta: contiene la información referente a las ofertas técnicas comerciales con que cuenta la dirección, permitiendo conocer sus hitos de pagos, precio total y darle un seguimiento a través de su estado.

contrato: permite conocer la cantidad de contratos reales existentes, su fecha de creación, así como su fecha fin, dándole un seguimiento a través de los estados por los que transita en todo su proceso.

proveedor: contiene la información referente de todos los proveedores con que cuenta el centro.

cliente: contiene la información de todos los clientes

pedido_venta: permite a los directivos conocer y controlar los pedidos a través de los estados y su fecha de creación.

Conclusiones del capítulo

Se elaboraron los productos de trabajo relacionados con las disciplinas de Modelado de negocio, Requisitos y Análisis y diseño de la metodología, sirviendo estos como guía para la implementación del sistema. El estudio de los sistemas y las entrevistas realizadas permitieron definir 46 requisitos funcionales y 18 requisitos no funcionales. La definición del modelo de datos relacional permitió un

mejor entendimiento y organización de la información. La aplicación de los patrones arquitectónicos y del diseño permitieron crear una estructura común y conocida para varios programadores, lo que facilitará un mejor mantenimiento y reutilización del sistema.

CAPÍTULO 3: IMPLEMENTACIÓN Y VALIDACIÓN DE LA PROPUESTA DE SOLUCIÓN

Introducción

En este capítulo se abordan los elementos de la fase de implementación y prueba propuestos por la metodología. Se definen los estándares de codificación utilizados, así como las pruebas realizadas: caja blanca, caja negra, y de aceptación para la validación de la propuesta de solución.

3.1 Implementación

En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema (Sánchez, 2015).

En esta fase a partir de los resultados obtenidos del análisis y diseño se construye el sistema. El objetivo de la misma, es realizar las actividades necesarias para poner a disposición de los usuarios finales, el sistema desarrollado.

Seguidamente se describen los componentes en los cuales está estructurado el sistema desarrollado.

3.2 Diagrama de componentes

El diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean estos ficheros de código fuente, binarios o ejecutables. Los elementos de modelado que lo conforman son los componentes y paquetes que muestran la estructura del sistema en términos de implementación a un alto nivel. (Pressman, 2002). El diagrama de componentes proporciona una visión física de la implementación del software. Muestra la organización de los componentes y sus relaciones.

La Figura 11 muestra el diagrama de componentes del sistema, la organización y dependencia existente entre los componentes Vistas, Controlador, Base de Datos y Models.

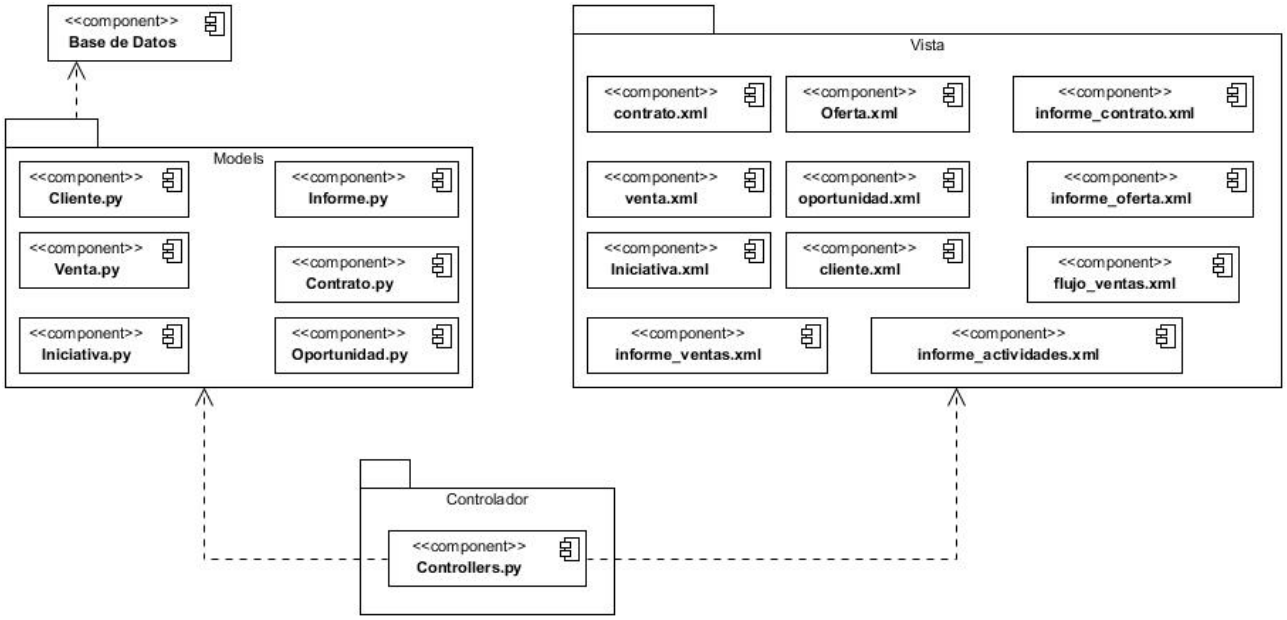


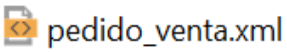
Figura 11: Diagrama de componentes. Elaboración propia.

3.3 Estándares de codificación

Los estándares de código son una forma de "normalizar" la programación de forma tal que, al trabajar en un proyecto, cualquier persona involucrada en el mismo tenga acceso y comprenda el código (LinkedIn Corporation, 2017). A continuación, se presentan diferentes ejemplos de estilos de codificación definidos para la implementación de la aplicación:

Nombre de los archivos

- vista: módulo_nombre



Clases de un modelo

- Si la clase es nueva: class módulo_nombre

```
class pedido_venta(models.Model):
```

Vistas

- vista Form:id = ¨módulo_modelo_form ¨

```
<record model="ir.ui.view" id="pedido_venta_form">
```

- vista Tree: id = ¨módulo_modelo_tree ¨

```
<record model="ir.ui.view" id="pedido_venta_tree">
```

Menú

- id = ¨menú_módulo_nombre¨

```
<menuitem id="menu_pedido_venta"
```

3.4 Resultado de la implementación

A continuación, se muestra ejemplificado un fragmento de código de la funcionalidad: ***Gestionar contrato***, correspondiente al módulo de Ventas, específicamente la funcionalidad permite adicionar un contrato de acuerdo con los parámetros definidos.

```
class contrato(models.Model):
    _name = 'contrato'

    name=fields.Date("Fecha inicio del contrato",required=True)
    fecha_fin = fields.Date("Fecha fin del contrato", required=True)
    fecha_garan = fields.Date("Fecha de garantía")
    oblig = fields.Text("Obligaciones de pago")
    cliente= fields.Many2one('res.partner', 'Cliente',required=True)
    hitos=fields.Many2many('hitos_contrato', string="Hitos de Contrato")
    costo=fields.Float("Precio en CUC", store=True)
    codigo=fields.Char("Código", required=True, size=4)

    _sql_constraints=[('codigo_uniq','unique(codigo)','El el código del contrato ya existe en la base de datos. ')
                                        #   validaciones
    @api.multi
    @api.constrains('name', 'fecha_fin')
    def validacion(self):
        if self.name:
            if self.fecha_fin:
                if self.fecha_fin <= self.name:
                    raise exceptions.ValidationError('La fecha final debe ser posterior a la fecha de inicio.')

    @api.multi
    @api.constrains('name', 'fecha_garan','fecha_fin')
    def validacion1(self):
        if self.name:
            if self.fecha_garan:
                if self.fecha_fin:
                    if self.fecha_garan <= self.name or self.fecha_fin <= self.fecha_garan:
                        raise exceptions.ValidationError('La fecha de garantía debe estar en el periodo entre fech
```

Figura 12: Fragmento de código de la funcionalidad: Gestionar contrato

Después de haber realizado la implementación del módulo, a continuación se muestra la interfaz gráfica del requisito Adicionar contrato, donde se muestran los datos a introducir para registrar en el sistema un nuevo contrato. En la parte izquierda se muestra el menú con las diferentes funcionalidades del sistema.

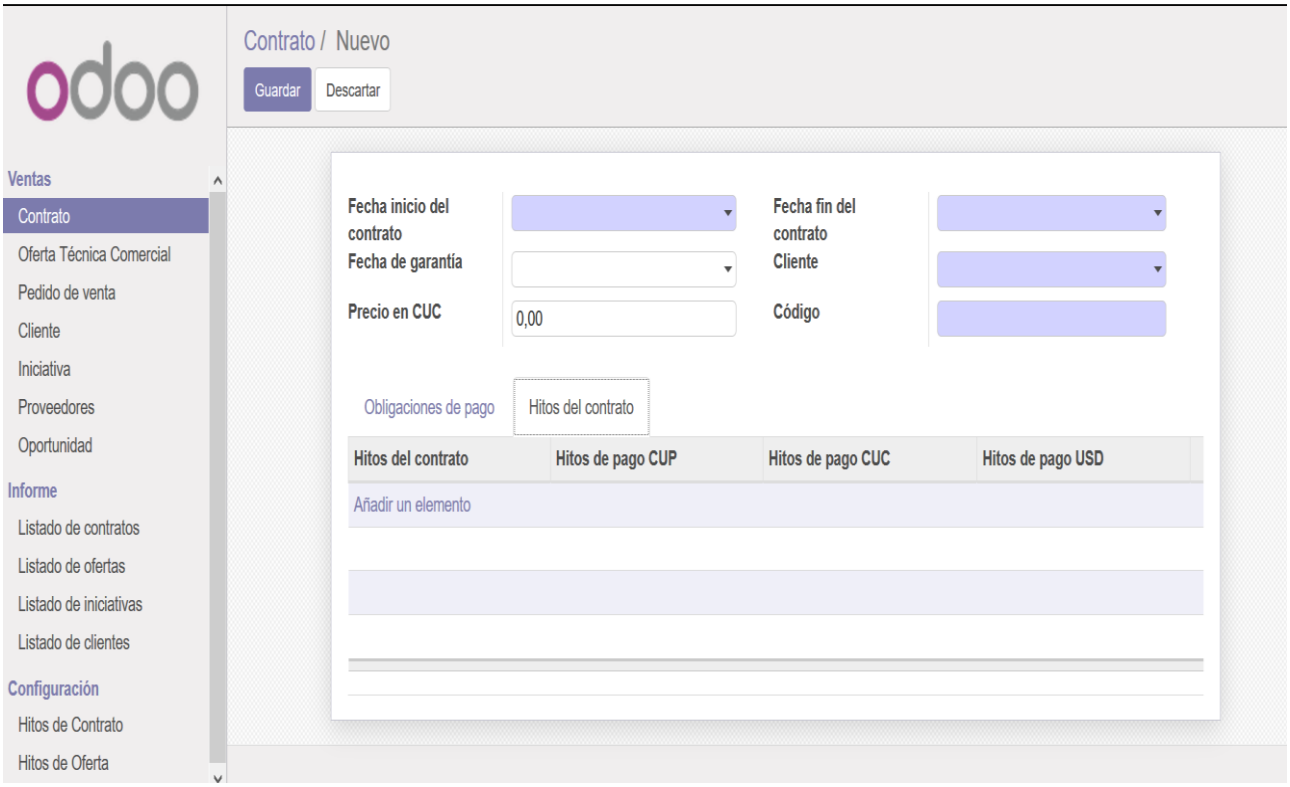


Figura 13: Interfaz gráfica del requisito Adicionar contrato.

La aplicación de métricas para validar tanto el diseño como la propuesta de solución es fundamental para elevar la calidad de la misma y cumplir con las aspiraciones del cliente. A continuación, se definen las métricas utilizadas para evaluar el diseño propuesto.

3.5 Métricas para la validación del diseño

Las métricas de diseño permiten medir de forma cuantitativa la calidad de los atributos internos del software. A nivel de componentes se centran en las características internas de los componentes del software con medidas que pueden ayudar al desarrollador a juzgar la calidad de un diseño a nivel de componente. Para la validación del diseño se aplicaron las métricas Tamaño operacional de clase (TOC) y Relaciones entre clases (RC) debido al conjunto de atributos de calidad de diseño que ambos miden. A continuación, se describe cómo son aplicados en la propuesta de solución.

3.5.1 Métrica de Relaciones entre Clases (RC)

Esta métrica se basa en la cantidad de relaciones de uso que presenta una clase determinada con las demás clases. Utiliza los atributos Acoplamiento, Complejidad de Mantenimiento, Reutilización y Cantidad de Pruebas para medir la calidad del diseño de la clase. (Hitz, y otros, 1996)

Los umbrales para evaluar el diseño según esta métrica se basan en el promedio de relaciones por clases y el caso del atributo Acoplamiento, utiliza la propia cantidad de relaciones, estos valores

fueron los aplicados en el diseño del sistema y los mismos son reflejados en la siguiente tabla: (Hitz, y otros, 1996)

Resultados obtenidos al aplicar la métrica RC

A partir de la evaluación de cada una de las clases según los atributos anteriormente mencionados se obtuvo el siguiente resultado:

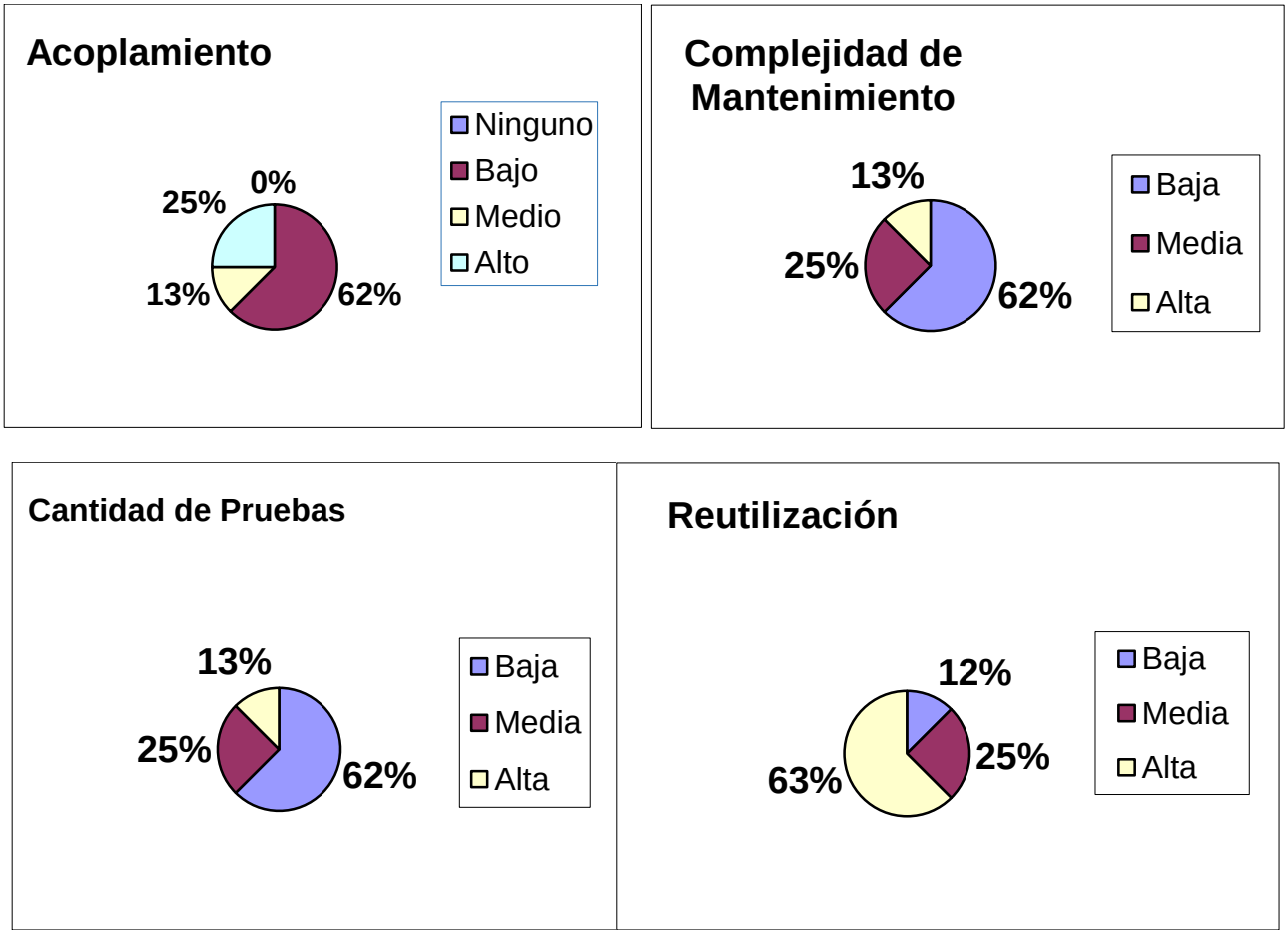


Figura 14: Resultado de la métrica RC

Aplicando la métrica Relaciones entre clases a estas 7 clases se obtuvo un total de 14 dependencias obteniéndose un promedio aproximado de 1.75 relaciones por clase. También se llega a la conclusión de que el 62% de las clases tienen un bajo acoplamiento, baja complejidad de mantenimiento y baja cantidad de pruebas. Por último, se tiene que existe un 63% de clases con reutilización alta. Como se puede observar en la figura el acoplamiento posee un nivel bajo por lo que existe poca dependencia entre las clases trayendo como consecuencia una alta probabilidad de reutilización. También se puede apreciar que existe un bajo nivel de complejidad de

mantenimiento por lo que a la hora de optimizar métodos y demás operaciones no es necesario realizar una gran cantidad de pruebas.

3.5.2 Métrica Tamaño operacional de las clases (TOC)

Las métricas basadas en el tamaño operacional de las clases se basan en contar la cantidad de atributos y la cantidad de operaciones que tiene una clase individual y el promedio que presenta el sistema en su totalidad. Una vez analizado el indicador tamaño de clase, si el valor resultante tiende al crecimiento, es probable que la clase esté saturada de responsabilidades; en consecuencia, el nivel de reutilización sería mínimo y la implementación altamente compleja.

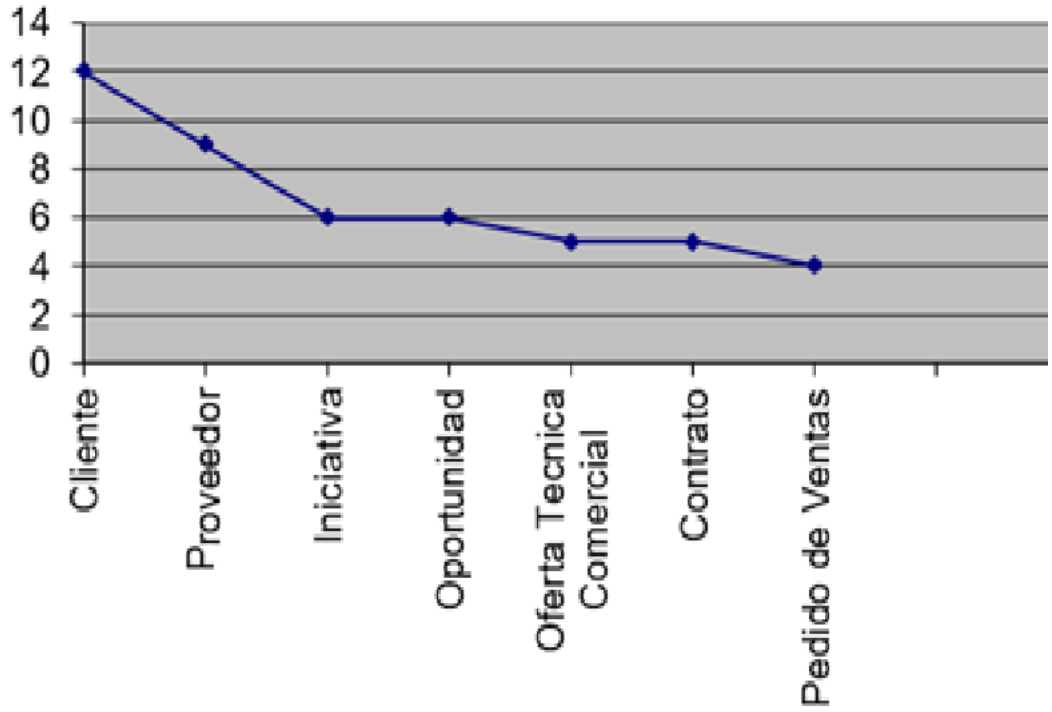
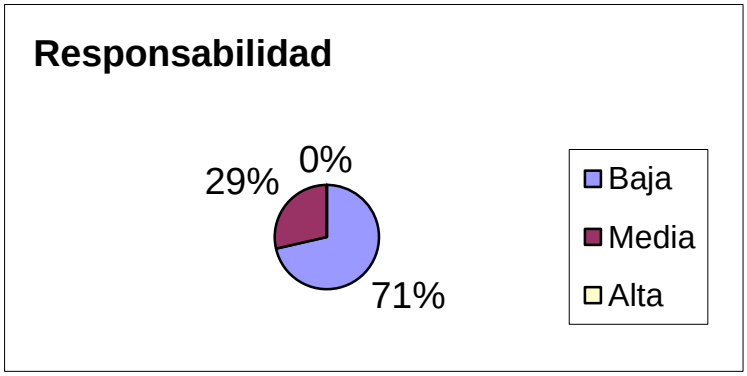


Figura 15: Cantidad de procedimientos por clase.



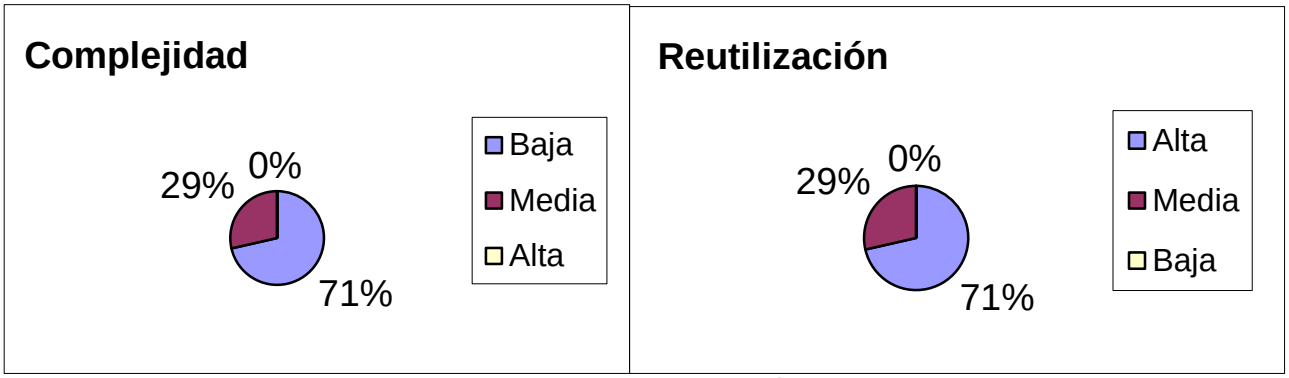


Figura 16: Resultado de la métrica TOC

El resultado de aplicar el método a estas 7 clases fue de un 29% medio y un 71% alto de responsabilidad, comportándose de manera para la complejidad y la reutilización tiene un 29% de media, 71% de baja. Analizando los resultados se puede concluir que más del 65% de las clases poseen un bajo nivel de complejidad por lo que se puede realizar una alta explotación de la reutilización.

3.6 Disciplina Pruebas internas

En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posibles componentes de prueba ejecutables para automatizar las pruebas. (Sánchez, 2015)

Pruebas de software

Las pruebas de software son un elemento crítico para la garantía de la calidad de software y una revisión final de las especificaciones del diseño y de la codificación. El objetivo fundamental de estas pruebas es medir el grado en que el software cumple con los requisitos definidos.
(Pressman, 2010).

3.6.1 Pruebas caja blanca

Para comprobar que cada sentencia de código se ejecuta al menos una vez, se realizaron pruebas al código de las funcionalidades más complejas desde el punto de vista de la programación en cada uno de los modelos de la aplicación siguiendo la técnica propuesta en el Capítulo1.

La técnica a aplicar es la de la ruta básica, la cual permite al diseñador de caso de prueba obtener una medida de complejidad lógica de un diseño de procedimiento o usar esta como guía para la definición de un conjunto básico de caminos independientes. Para obtener dicho conjunto se construye el grafo de flujo asociado a una función y se calcula su complejidad, para este caso se

tomó como ejemplo el método calcPrecio(self), encargado de calcular el precio total del contrato a partir de los hitos de pagos asociados al mismo. Primeramente, se enumeran las sentencias de código.

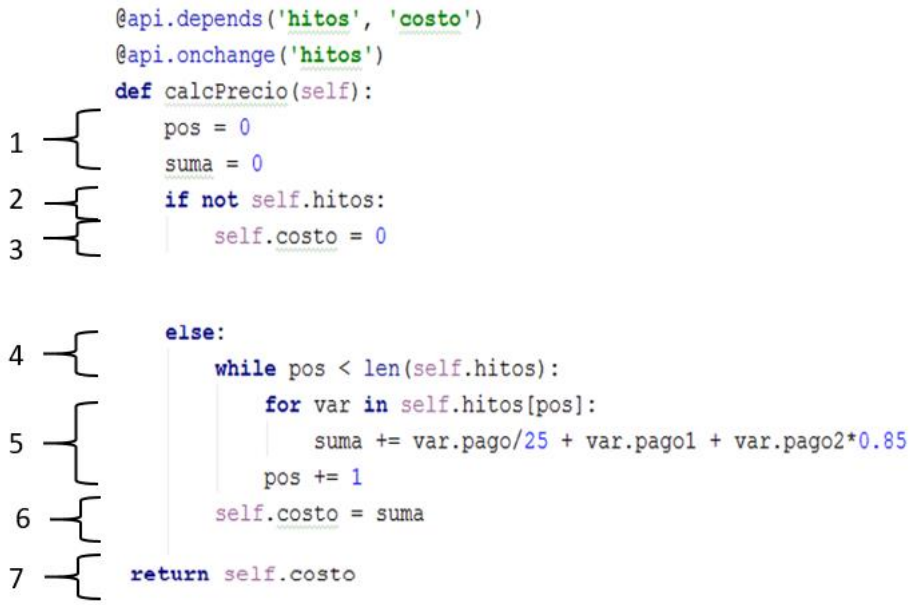


Figura 17: Método utilizado para aplicarle la prueba de caja blanca.

Grafo de flujo correspondiente al código del método.

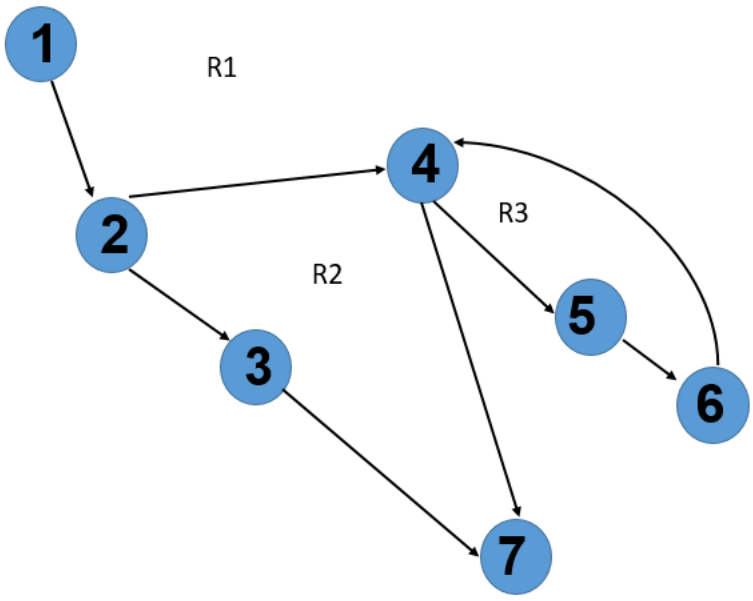


Figura 18: Descripción de la aplicación de la Técnica: Ruta Básica**.**

La complejidad ciclomática será calculada mediante la siguiente fórmula:

- V (G) = E - N + 2, donde E es el número de aristas y N el número de nodos de la gráfica de flujo

Realizando los cálculos correspondientes se obtiene el siguiente resultado:

V (G) = 8-7+2

V (G) =3

- V(G) = R = 3
- V (G) = P + 1
  V (G)=2+1=3

Por lo que el conjunto de caminos básicos sería:

Camino 1: 1-2-3-7

Camino 2: 1-2-4-7

Camino 3: 1-2-4-5-6-4-7

Por tanto, existen 3 rutas independientes en el método analizado, por lo que serán necesarios igual número de casos de pruebas para verificar que se prueben todos los posibles datos de entradas en el método.

Tabla 4: Descripción del caso de prueba aplicado al camino básico 3.

Descripción: Se verificará que el sistema permita añadir un hito de pago de contrato		
Condición de Ejecución	**Entrada**	**Resultados esperados**
Crear un nuevo hito de pago al contrato.	Se llenan los hitos de pago	El sistema debe calcular el total del costo del contrato
Resultado: Satisfactorio.		

Luego se ejecutan los casos de prueba para comparar los resultados obtenidos con los esperados, una vez comprobado que estos coinciden, se puede asegurar que todas las sentencias del método se han ejecutado al menos una vez.

3.6.2 Método caja negra

Para la realización de estas pruebas se diseñaron casos de pruebas basados en requisitos, a partir de la técnica de partición de equivalencia definida en el Capítulo 1. Las pruebas de calidad fueron realizadas por el grupo de calidad del centro CEIGE perteneciente a la Facultad 3. A continuación se muestra el flujo central del caso de prueba del requisito Crear Cliente:

Descripción del caso de prueba Crear Cliente:

El requisito funcional permite crear (adicionar) un nuevo cliente en el sistema, a partir de los atributos que lo componen.

Condiciones de ejecución:

- El usuario autenticado en el sistema debe tener los permisos para ejecutar esta acción.
- El usuario debe seleccionar la opción Gestionar Cliente.

Tabla 5: Diseño de caso de prueba de Caja Negra

Escenario	**Descripción**	**Nombre**	**Teléfono**	**Dirección**	**Nombre de la empresa**	**Respuesta del Sistema**	**Flujo Central**
EC 1.1 Adicionar cliente introduciendo	Se registra el cliente introduciendo datos de forma correcta	V Oscar	V 45283287	V Ave.8 #15	V Datec	Se activa otras opciones para trabajar en el cliente.	1.Selecciona en el Menú Izquierdo la opción

datos válidos.							"Gestionar Cliente " 2.Se introducen los datos del cliente correctamente. 3. Se presiona el botón Guardar.
EC 1.2 Adicionar cliente introduciendo datos inválidos.	No se registra el cliente introduciendo datos de forma incorrecta	Oscar123	Bhdd345	varchar		Se muestra un mensaje diciendo que los datos introducidos solo deben contener letras en el caso del nombre y números en el caso de teléfono.	1.Selecciona en el Menú Izquierdo la opción "Gestionar Cliente " 2.Se introducen los datos del cliente correctamente.

							3. Se presiona el botón Guardar.
EC 1.3 Adicionar cliente dejando campos vacíos.	Se dejan campos vacíos al adicionar un cliente.	Vacío	Vacío			Se muestra los campos en rojo.	1.Selecciona en el Menú Izquierdo la opción "Gestionar Cliente " 2.Se introducen los datos del cliente correctamente. 3. Se presiona el botón Guardar.
EC 1.4 Descartar.	Se descarta la operación de adicionar un cliente.						Se introducen o no los datos del cliente.

							Se presiona el botón Descartar.

Tabla 6: Descripción de variables del requisito Adicionar Cliente.

Nombre del campo	Clasificación	Valor nulo	Descripción
Nombre	Campo de texto	No	Solo letra
Teléfono	Campo de texto	Si	Solo números
Email	Campo de texto	Si	Dirección de correo electrónico, ejemplo(usuario@gmail.com)
Sector empresarial	Campo de selección	Si	Selección
Sitio web	Campo de texto	Si	Dirección de sitio web, ejemplo (http://www.etecsa.com)
Nombre de la empresa	Campo de texto	Si	Solo letra
Dirección	Campo de texto	Si	Dirección
Provincia	Campo de selección	Si	Selección
Código postal	Campo de texto	Si	Solo números
País	Campo de selección	Si	Selección
Observaciones	Campo de texto	Si	Admite letras, números y caracteres

En los documentos entregables correspondientes al expediente de proyecto, se pueden encontrar los restantes diseños de casos de pruebas de caja negra aplicados a la solución.

La técnica de partición de equivalencia fue aplicada por los especialistas del grupo de calidad en el centro CEIGE.

Resultado de la prueba

En la primera iteración se probaron un total de 7 casos de pruebas de los cuales 4 arrojaron resultados satisfactorios para un 58% de efectividad, identificándose un total de 32 no conformidades tales como errores ortográficos y validaciones incorrectas las cuales se corrigieron. Para la segunda iteración se detectaron 9 no conformidades las cuales se corrigieron. Para la tercera iteración no se detectaron no conformidades resultando satisfactorias las pruebas en un 100% quedando de esta forma validado el sistema, emitiendo para ello un Acta de Liberación (Anexo 3). A continuación, se muestra el resultado de las pruebas funcionales para cada iteración.

Tabla 7: No conformidades detectadas por iteración.

No conformidades	Aplicación	Ortografía
Primera iteración	23	9
Segunda iteración	7	2
Tercera iteración	-	-

3.7 Pruebas de aceptación

Las pruebas de aceptación se realizan para que el cliente certifique que el sistema es válido para él. La planificación detallada de estas pruebas debe haberse realizado en etapas tempranas del desarrollo del proyecto, con el objetivo de utilizar los resultados como indicador de su validez: si se ejecutan las pruebas documentadas a satisfacción del cliente, el producto se considera correcto, por tanto, adecuado para su puesta en producción. (Ramos, 2007).

El cliente, luego de haber revisado los productos de trabajo entregados, determina que está de acuerdo con el producto final mostrado, el cual fue desarrollado bajo los requisitos previamente definidos. La aceptación del producto arrojó un total de 2 solicitudes de cambios en 2 Iteraciones, las cuales fueron resueltas. Estas solicitudes fueron solucionadas para el correcto funcionamiento del sistema. Una vez comprobada la implementación de los requisitos, el cliente emitió el Acta de aceptación con el cliente, la cual puede ser consultada en el Anexo 6.

3.8 Resultados de la validación mediante criterios de expertos: Método Delphy

A partir del análisis de la población, objeto de aplicación de la encuesta, se decidió aplicar un muestreo a conveniencia. Consiste en una técnica de muestreo no probabilístico donde los sujetos

son seleccionados, dada la conveniente accesibilidad y proximidad de los sujetos para el investigador. Además, se consideró que las personas a encuestar no son expertos en el tema en cuestión, pero sí tienen un alto conocimiento de los procesos relacionados con la gestión de ventas. En correspondencia con estos elementos se diseñó un cuestionario orientado a obtener información sobre la valoración de los expertos. Se propone un cuestionario de un total de 8 preguntas de respuesta cerrada, facilitándose de esta forma el procesamiento y la cuantificación. La vía utilizada para la aplicación de la encuesta fue el cuestionario impreso.

Las respuestas a las preguntas están representadas en una escala de Likert. Es una escala psicométrica comúnmente utilizada en cuestionarios y se considera la escala de uso más amplio en encuestas para la investigación, principalmente en ciencias sociales. En cada pregunta del cuestionario se responde con el nivel de acuerdo o desacuerdo con una declaración, elemento o pregunta (ver anexo 4).

Para el procesamiento de la información obtenida en las encuestas se hizo corresponder un valor numérico a cada posible resultado de la escala utilizada. Los valores utilizados se muestran en la siguiente tabla:

Tabla 8: Valores utilizados en la encuesta

Respuesta	Valor numérico
mucho	5
bastante	4
poco	3
muy poco	2
nada	1

Procedimiento para determinar el grado de consenso entre los expertos:

A partir de la tabla 8 se puede afirmar que existe un alto grado de concordancia en el criterio de los expertos en todas las preguntas realizadas. En todos los casos el coeficiente de variación muestra valores por debajo de 0.12 lo que sustenta la afirmación anterior. De este modo se puede concluir que, de acuerdo al consenso en el juicio de los expertos, la solución propuesta contribuye a elevar la gestión de los procesos de ventas de la Dirección de Transferencia de Tecnología en la UCI. Destacan las preguntas 2 y 4 con un coeficiente de variación por debajo de 0.10.

Tabla 9*:* Resultado de la aplicación del método Delphy.

Expertos	Preguntas							
	1	2	3	4	5	6	7	8
1	4	4	4	4	5	5	4	4
2	4	4	4	4	4	5	4	5
3	5	3	4	4	4	5	4	4
4	4	4	4	4	4	5	4	3
5	4	4	5	4	5	5	4	4
6	4	4	5	4	4	4	3	4
7	4	4	4	4	4	4	4	4
8	4	4	4	4	4	4	4	4
9	4	4	4	4	4	4	4	4
10	5	4	4	4	4	4	3	4
Cj media	4.20	3.9	4.2	4	4.2	4.5	3.8	4
Varianza	0.18	0.10	0.18	0.00	0.18	0.28	0.18	0.22
Coef. de Variación	0.10	0.08	0.10	0.00	0.10	0.12	0.11	0.12

Fuente: Elaboración propia

El análisis de la tabla anterior permitió concluir lo siguiente por cada uno de las preguntas:

Tabla 10: Evaluación de las preguntas del cuestionario utilizado en la aplicación del método DELPHY.

Preguntas	Valoración de los expertos
1. La solución propuesta contribuirá a garantizar la integridad de los procesos de ventas.	Bastante
2. La solución propuesta contribuye a mejorar el cumplimiento de las fechas de entrega de los contratos.	Bastante

3. La solución propuesta contribuye a mejorar la creación de ofertas técnicas comerciales enfocadas a las necesidades de los clientes.	Bastante
4. La solución propuesta contribuye a mejorar la planificación entorno a los procesos de ventas.	Bastante
5. La solución propuesta contribuye a mejorar el seguimiento de los procesos de ventas.	Bastante
6. La solución propuesta contribuye a la centralización de los datos de los clientes.	Mucho
7. La solución propuesta contribuye a mejorar la consulta de la información histórica del proceso de venta.	Bastante
8. La solución propuesta contribuye a mejorar la creación de reportes en el proceso de venta.	Bastante

Conclusiones del capítulo

El empleo de métricas de software validó el diseño propuesto en el capítulo anterior y demostró que el componente cuenta con un diseño robusto. El uso de estándares de codificación aseguró una homogeneidad en la nomenclatura de las clases y métodos, que facilita su comprensión a otros programadores. la práctica de pruebas de caja blanca y caja negra comprobó el cumplimiento de los requisitos funcionales y la correcta ejecución del código.

CONCLUSIONES GENERALES

La realización de este trabajo responde a la necesidad de buscar una solución al problema planteado. Concluida la investigación se llegó a las siguientes conclusiones:

- Con el estudio de los diferentes sistemas CRM, se identificaron las principales limitaciones de Odoo y a partir de ellas se definieron las funcionalidades a implementar para cumplir con los objetivos planteados en la investigación.
- A partir de la metodología utilizada se generaron artefactos que permitieron guiar el desarrollo del sistema, utilizándose métricas para la validación de los mismos, obteniéndose resultados satisfactorios tanto para los requisitos como para el diseño.
- En la fase de implementación, se obtuvo un sistema basado en la personalización del módulo de Ventas para la Administración de Relaciones con el Cliente en Odoo, acorde con las necesidades de la Dirección de Transferencia de Tecnología en la UCI.
- Las pruebas de software ejecutadas permitieron verificar el correcto funcionamiento de la aplicación, validándose que cumple con las especificaciones y requisitos definidos por el cliente.
- La validación de la investigación mediante el uso del método de criterio de experto arrojó que la propuesta de solución contribuye de manera satisfactoria a la gestión de los procesos de ventas para la Dirección de Transferencia de Tecnología en la UCI.

RECOMENDACIONES

Se recomienda para futuras versiones continuar mejorando las funcionalidades del sistema, así como realizar la integración con el módulo de Mercadotecnia para obtener un sistema CRM más completo.

BIBLIOGRAFÍA

AllProWebTools. 2016. AllProWebTools.com. [En línea] 2016. [Citado el: 21 de 3 de 2018.] www.allprowebtools.com/es/products/what-is-AllProWebTools.

Alpízar, Jean Carlo Murillo, Oconitrillo Rodriguez, Carolina y Esquivel Bolaños, Leylin. 2017. *Patrones de diseño.* Costa Rica : s.n., 2017.

Benaque, José Luis. 2007. *Otros conceptos y herramientas de contabilidad y finanzas.* 2007.

debugmodeon. 2017. debugmodeon.com. [En línea] 2017. [Citado el: 15 de 6 de 2018.] http://es.debugmodeon.com/articulo/frameworks-php-conceptos-basicos.

DUGAN, DEB. 2016. Salesforce. *Salesforce.* [En línea] 2016. [Citado el: 10 de enero de 2018.] www.salesforce.com/es/products/what-is-salesforce.

excelia. 2018. https://www.excelia.com. *https://www.excelia.com.* [En línea] Herramienta de Gestión Empresarial – Customer Relation Management, 2018. [Citado el: 10 de 1 de 2018.] https://www.excelia.com/que-es-un-crm-herramienta-de-gestion-empresarial/.

García Valdés, M y Suérez Marín, M. 2013. *El método Delphi para la consulta a expertos en la investigación científica.* 2013.

González, Rolando Alfredo Hernández León y Sayda Coello. 2002. *El paradigma cuantitaivo de la investigación científica.* Ciudad de la Habana : Editorial Universitaria, 2002.

HALL, RICHRD. 2014. *Planeación y Organización de Empresas.* México : Editorial Prentice-Hall Hispanoamérica, 2014.

Hitz, Martin y Montazeri, Behzad. 1996. *Chidamber and Kemerer´s metrics suite: a measurement theory perspective.* 1996. págs. 267-271. Vol. 22.

Hostname. 2016. https://www.hostname.cl. [En línea] 2016. [Citado el: 12 de marzo de 2018.] https://www.hostname/ Blog / Servidores.

Larman. 1999. *UML y Patrones, Introducción al análisis y diseño orientado a objetos.* 1999.

Larman, Craig. 2003. *UML y patrones, Introducción al análisis y diseño orientado a objetos.* s.l. : 2da edicion, 2003.

LinkedIn Corporation. 2017. *Revisiones de código y estándares de codificación.* [En línea] 2017. [Citado el: 10 de mayo de 2018.] https://es.slideshare.net/PiXeL16/estandares-de-codigo-emanuel.

Lopez, T. 2008. *Selección de contenidos del Proyecto Estratégico de .* 2008.

Microsoft Dynamics . 2015. Dynamics 365. *Dynamics 365.* [En línea] 2015. [Citado el: 10 de enero de 2018.] https://www.microsoft.com/en-us/dynamics365/what-is-crm.

Netsuite. 2013. Oraclenetsuite. *netsuite.* [En línea] 26 de Abril de 2013. [Citado el: 10 de enero de 2018.] http://www.netsuite.com/portal/resource/articles/crm/what-is-crm.shtml.

Odoo. 2017. www.odoo.com. *www.odoo.com.* [En línea] 2017. [Citado el: 14 de 2 de 2018.] www.odoo.com/document.

Orichijuan. 2014. *¿Qué son los framework de desarrollo?* 2014.

Parra, José David. 2006. *Guía de patrones, prácticas y arquitectura.* 2006.

PgAdmin. 2016. www.pgadmin.org. *www.pgadmin.org.* [En línea] 2016. [Citado el: 12 de marzo de 2018.] www.pgadmin.org..

Pressman, Roger. 2010. *Ingeniería del software.* 2010.

—. 2002. *Ingeniería del software. Un enfoque práctico.* 2002. Quinta Edición.

Pressman, Roger S. 2010. *Ingeniería del Software. Un Enfoque Práctico.* 2010.

Quiñones Montoro, Mariela. 2016. *Globalización de las relaciones laborales en el sector financiero.* Madrid : s.n., 2016.

Rendón, Carlos Augusto. 2007. *La extrategia de ventas.* 2007.

Sanchéz, Tamara. 2015. *Metodología de desarrollo para la UCI.* 2015.

Sánchez, Tamara Rodríguez. 2015. Metodología de desarrollo para la Actividad productiva de la UCI. *https://excriba.prod.uci.cu/page/context/shared/document-details?nodeRef=workspace://SpacesStore.* [En línea] 2015. [Citado el: 7 de mayo de 2016.]

—. 2015. Metodología de desarrollo para la Actividad productiva de la UCI. [En línea] 2015. [Citado el: 26 de 10 de 2017.] https://excriba.prod.uci.cu/page/context/shared/document-details?nodeRef=workspace://SpacesStore.

Sommerville, Ian. 2005. *Ingenieria de Software.* 2005.

—. 2011. *Ingeniería del software.* novena. 2011.

SugarCRM. 2016. SugarCRM. [En línea] 2016. [Citado el: 10 de enero de 2018.] www.sugarcrm.com/resources/crm-guides.

Thompson, Iván. 2012. *Concepto de Venta.* 2012.

Visual Paradigm. 2017. Visual Paradigm.com. [En línea] 2017. [Citado el: 21 de 6 de 2018.] www.visual-paradigm.com/what-is-visual-paradigm.

Win2PDF. 2015. *Utilización del método Delphi en la pronosticación: una experiencia inicial.* 2015.

Zaragoza, Jesús. 2012. www.deconceptos.com. *De conceptos.* [En línea] 5 de marzo de 2012. [Citado el: 10 de enero de 2018.]

ANEXOS

Anexo 1.

Entrevista realizada al Jefe del Departamento Comercial de la Dirección de Transferencia de Tecnología en la UCI.

Pregunta 1: ¿Cuáles son los principales conceptos asociados a la administración de relaciones con el cliente en nuestro país?

Pregunta 2: ¿Cuáles son los principales documentos por los que se rigen las ventas en nuestra universidad y en el país?

Pregunta 3: ¿Cómo se gestionan los contratos y ofertas actualmente en el centro?

Pregunta 4: ¿Cuál es la diferencia entre una Oferta Técnica y una Oferta Técnica Comercial?

Pregunta 5: ¿Qué significan los hitos de pagos?

Pregunta 6: ¿Qué significa que un contrato este en estado de negociación?

Anexo 2.

Acta de aceptación por parte del cliente.

UCI Universidad de las Ciencias Informáticas

Acta de aceptación

ACTA DE ACEPTACIÓN

En cumplimiento del Trabajo de diploma titulado Personalización del módulo de Ventas para el Sistema de Administración de Relaciones con el Cliente en Odoo y en función de la ejecución del proyecto del Centro de Informatización de Entidades (CEIGE), se hace entrega de los productos que se relacionan a continuación:

- Descripción de requisitos por proceso
- Descripción de Proceso de Negocio
- Modelo Conceptual

La Parte Cliente, luego de haber revisado los productos de trabajo determina que sí se aceptan.

Comentarios

Se acepta las Especificación de Requisitos de Software con la condición de que se modifiquen determinadas cosas.

Entrega	Recibo
Nombre y apellidos: Dra. M. Sintura Chirino	Nombre y apellidos:
Cargo:	Cargo:

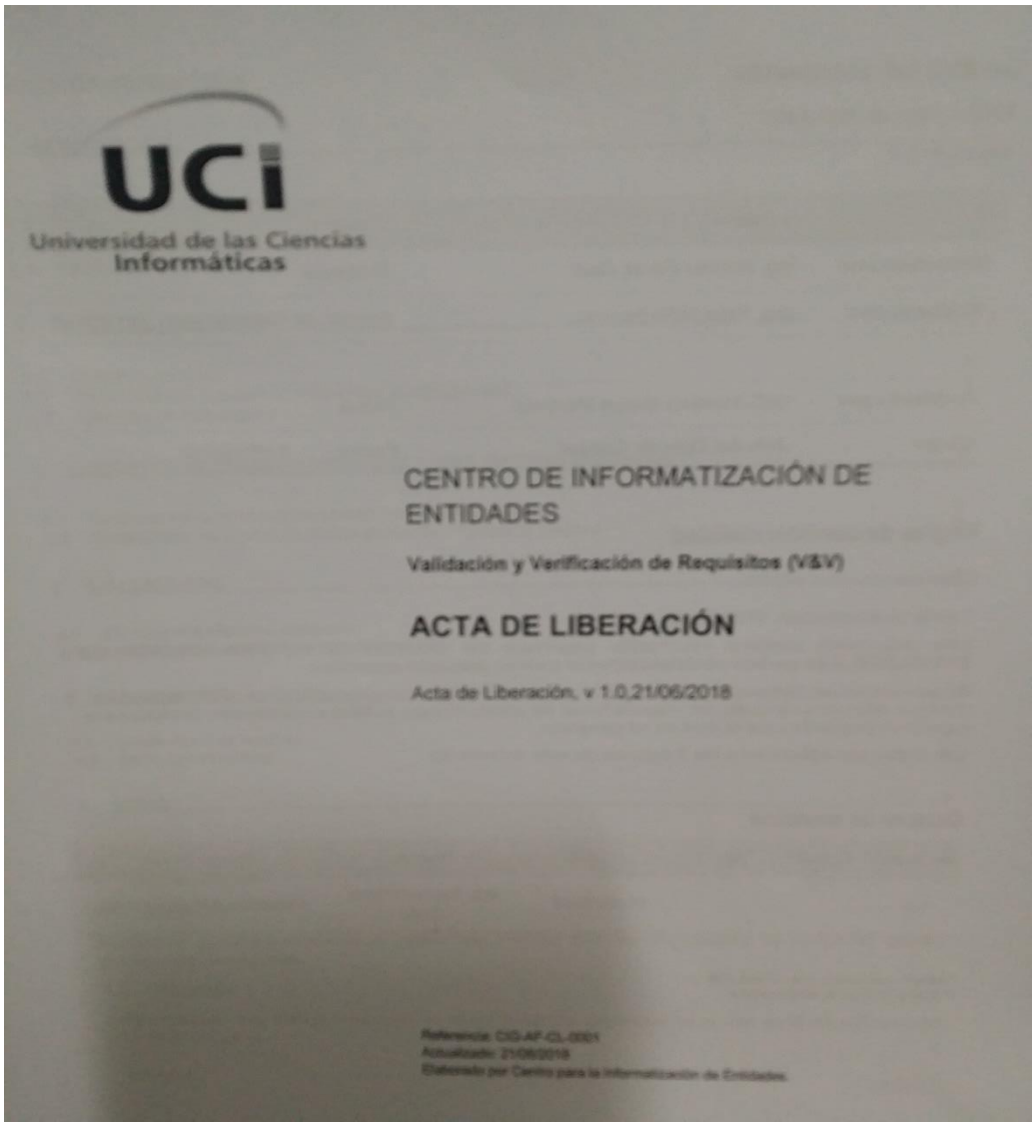
Representante Parte Suministradora

Nombre y Apellidos: Dra. M. Sintura Chirino

Cargo: Estudiante

Firma:

Anexo 3. Acta de liberación



Anexo 4. Encuesta para aplicar a expertos

La siguiente encuesta ha sido diseñada con el objetivo de recoger su juicio acerca del impacto de la solución de la ¨Personalización del módulo de Ventas para el Sistema de Administración de Relaciones con el Cliente en Odoo¨ sobre la gestión de los procesos relacionados con las ventas en la Dirección de Transferencia de Tecnología en la UCI. Después de tener conocimiento de las funcionalidades del sistema, le pedimos que responda cada pregunta con el mayor nivel de objetividad posible. Usted debe marcar solo un ítem por pregunta. ¡Gracias por su contribución!

Preguntas:

1. En qué medida considera usted que la solución propuesta contribuirá a garantizar la integridad de los procesos de ventas en la Dirección de Transferencia de Tecnología en la UCI.

Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___

2. Considera usted que la solución propuesta contribuye a mejorar el cumplimiento de las fechas de entrega de los contratos en el proceso de ventas en la Dirección de Transferencia de Tecnología en la UCI.

Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___

3. Considera usted que la solución propuesta contribuye a mejorar la creación de ofertas técnicas comerciales enfocadas a las necesidades de los clientes en el proceso de ventas en la Dirección de Transferencia de Tecnología en la UCI.

Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___

4. En qué medida cree que la solución propuesta contribuya a mejorar la planificación entorno a los procesos de ventas en la Dirección de Transferencia de Tecnología en la UCI.

Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___

5. Considera usted que la solución propuesta contribuye a mejorar el seguimiento de los procesos de ventas en la Dirección de Transferencia de Tecnología en la UCI:

Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___

6. En qué medida cree que la solución propuesta contribuye a la centralización de los datos de los clientes en el proceso de ventas en la Dirección de Transferencia de Tecnología en la UCI:

Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___

7. Considera usted que la solución propuesta contribuye a mejorar la consulta de la información histórica del proceso de venta en la Dirección de Transferencia de Tecnología en la UCI:
   Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___
8. Considera usted que la solución propuesta contribuye a mejorar la creación de reportes en el proceso de venta en la Dirección de Transferencia de Tecnología en la UCI:
   Mucho ___ Bastante ___ Poco ___ Muy poco ___ Nada ___