Universidad de las Ciencias Informáticas

Facultad 3

Personalización del módulo de Contabilidad de Odoo 8.0

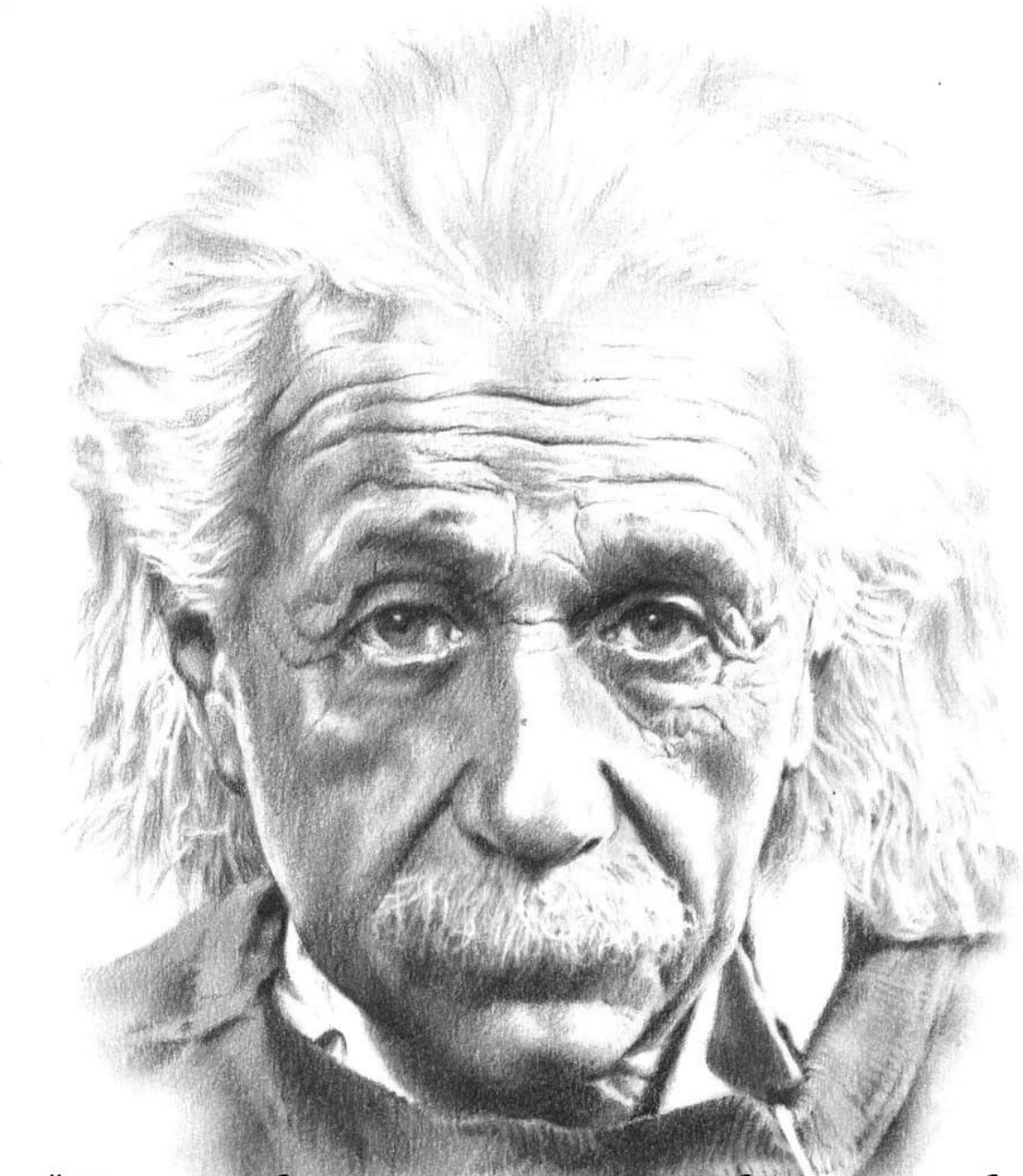
Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Juan Carlos Kelly Naranjo

Tutor(es): Ing. Annia Verdecia Boza

Ing. Yunior Feria Chapman

La Habana, 2016



"Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad".

Albert Einstein

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 _____.

Autor:

_____

Juan Carlos Kelly Naranjo

Tutores:

_____ _____

Yunior Feria Chapman Annia Boza Verdecia

Agradecimientos

A mis compañeros de aula, a mis hermanos de toda la vida. A todos los que de una forma u otra me han acompañado a lo largo de todos estos años. A los que me ayudaron, a los que no, a todos en general, les agradezco por toda su compañía.

Dedicatoria

A mi abuelita que no está presente. Uno de sus sueños era verme graduado, y con esto se lo estoy cumpliendo, donde quiera que estés, éste es tu regalo. A mis padres, en especial a mi mamá, mujer incomparable y muy especial. Durante todos estos años ha hecho lo imposible por entregarme una vida mejor, gracias por tu amor, tu preocupación, te quiero mucho. A mi papá, que, aunque no pueda estar aquí conmigo presente, esto va para ti también. A mis hermanos, los quiero mucho, gracias por estar ahí siempre.

RESUMEN

La Universidad de las Ciencias Informáticas está estructurada por varias facultades. Entre estas se encuentra la facultad 3, la cual cuenta con CEIGE (Centro de Informatización de Entidades), en donde se encuentra actualmente el proyecto Cedrux, el cual desarrollan sistemas para informatizar los procesos de gestión de las entidades presupuestadas y empresariales. La creación de Cedrux data del año 2008, cuando se constituyó su base tecnológica, a partir de un estudio de los principales Marcos de Trabajo (MT) implementados en PHP. A pesar de su calidad, éstos MT implementaban parcialmente los aspectos básicos que distinguen este tipo de soluciones. Por esta razón se decidió desarrollar una base tecnológica que reutilizara y extendiera las mejores prácticas aplicadas en los MT estudiados, a la cual se le denominó Sauxe. Con el tiempo, fueron identificados problemas en la base tecnológica de Sauxe, que influían negativamente en el proceso de desarrollo del software (Entidades(CEIGE), Centro de Informatización de, 2015). Debido a los problemas que presenta esta base tecnológica, se ha decidido utilizar el ERP Odoo 8.0 con el objetivo de desarrollar un sistema que satisfaga las normas contables cubanas a través de una localización cubana.

Para la construcción del sistema, se utilizó la metodología AUP en su versión UCI, lo que permitió guiar el proceso de desarrollo del software. La definición de herramientas y tecnologías contribuyeron a la implementación de la solución. Con este trabajo se contribuye a mejorar los procesos contables cubanos.

Palabras Clave: base tecnológica**,** normas cubanas, contabilidad, localización.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Los sistemas de planificación de recursos empresariales (ERP, por sus siglas en inglés, enterprise resource planning) son sistemas de información gerenciales que integran y manejan muchos de los negocios asociados con las operaciones de producción y de los aspectos de distribución de una compañía en la producción de bienes o servicios. La planificación de recursos empresariales puede intervenir en ventas, entregas, pagos, producción, administración de inventarios, calidad de administración y la administración de recursos humanos. El propósito fundamental de un ERP es otorgar apoyo a los clientes del negocio, tiempos rápidos de respuesta a sus problemas, así como un eficiente manejo de información que permita la toma oportuna de decisiones y disminución de los costos totales de operación (Services, 2016).

En el mundo existen grandes ERP capaces de adaptarse a las necesidades de cualquier empresa como son: Epicor, SAP, OpenERP, Oracle, Mycrosoft Dynamics, OpenBravo, entre otros. Dichos ERP tienen una amplia experiencia en diferentes tipos de empresas y muchos clientes alrededor del mundo utilizan sus soluciones. Algunos son sistemas propietarios, otros sistemas libres, desarrollados sobre tecnologías de última generación proporcionando la competitividad y su actualización mediante el avance de las TIC (Tecnologías de la Información y la Comunicación) (Services, 2016). Debido a la peculiaridad de los procesos contables financieros de nuestro país con respecto al mundo, estos sistemas ERP no se adaptan a las normas cubanas de contabilidad.

En Cuba se utilizan sistemas contables que a pesar de no ser ERP han soportado con mayor humildad la gestión de los procesos de las empresas, entre estos se destacan Versat Sarasola, Rodas XXI, E-Test, Siscont5, Cóndor, Cedrux, entre otros. Algunos de estos sistemas son soluciones de escritorio, que no cumplen con los principios de independencia tecnológica del país o la tecnología en que fueron desarrolladas se encuentra obsoleta (Entidades(CEIGE), Centro de Informatización de, 2015).

De los sistemas contables financieros antes mencionados se quiere destacar el Sistema Integral de Gestión Cedrux, paquete de soluciones integrales de gestión para las entidades presupuestadas y empresariales con funcionalidades generales de los procesos y las peculiaridades de la economía cubana. El sistema cuenta con módulos para la gestión contable, finanzas, facturación, inventario, capital humano y activos fijos.

Los módulos de contabilidad, finanzas y capital humano se encuentran certificados según las exigencias de la resolución 340/2015 del Ministerio de Finanzas y Precios. A pesar de estas fortalezas que provee la certificación del negocio y funcionalidades de dichos módulos de Cedrux,

la parte tecnológica presenta serias limitaciones que han llevado al equipo de proyecto de Cedrux a estudiar el ERP libre Odoo v8.0 como alternativa para obtener una localización cubana.

En el año 2015, el equipo de proyecto del sistema Cedrux realizó un estudio de la base tecnológica que lo sustentaba, donde quedaron identificados los siguientes problemas (Entidades(CEIGE), Centro de Informatización de, 2015) :

- Desactualización completa de la base tecnológica (ExtJs, PHP, Zend, Doctrine, PostgreSQL).
- Desactualización de los componentes desarrollados en la librería UCID. En esta librería se implementaron algunos componentes de ExtJs con comportamiento y atributos personalizados para satisfacer algunas necesidades propias del negocio de Cedrux y para facilitar la implementación de la capa de presentación.
- La versión que se usa de PHP presenta problemas con la carga en memoria de objetos muy grandes. Por el negocio que maneja Cedrux es frecuente que se necesiten cargar volúmenes de datos considerables.
- Las validaciones, excepciones y los servicios están registrados de forma global por lo que se hace muy difícil trabajar con esos ficheros de configuración.
- El mecanismo para el registro de las trazas afecta en gran medida el rendimiento de Cedrux.
- En la implementación del mecanismo para el manejo de las transacciones no se tuvo en cuenta el manejo de transacciones anidadas. Este es uno de los problemas de mayor criticidad puesto que se introducen en BD inconsistencia en los datos.
- El mecanismo de comunicación entre componentes es deficiente y muy rígido.
- La definición de los componentes no siempre contó con el visto bueno de un especialista lo que provocó que en ocasiones se abusara de este aspecto y el nivel de dependencias internas es considerable. Este elemento afecta directamente la modularidad de Cedrux.
- En ocasiones los conceptos globales no responden con los valores esperados. Se han identificado en varios escenarios que este problema está dado por el mal uso de este aspecto.
- El mecanismo para la internacionalización de Cedrux no tiene en cuenta los datos que están en la base de datos. A partir de ficheros json de idiomas maneja el idioma a nivel de interfaz. Para el correcto funcionamiento de este aspecto es necesario que se manejen correctamente los ficheros json. Esto puede ser en ocasiones engorroso para los programadores.

Por todo lo anterior planteado se definió un estudio de factibilidad de varias soluciones basándose en los criterios de funcionamiento de la tecnología, necesidad de capacitación, esfuerzo que implica el cambio, tiempo, escalabilidad, modularidad, licenciamiento, entre otros.

Durante el desarrollo del estudio se fueron descartando las soluciones con mayor inconveniente, hasta inclinarse por la utilización del ERP Odoo 8.0, como alternativa para obtener una localización cubana (Entidades(CEIGE), Centro de Informatización de, 2015).

A partir de la problemática anterior, se define el siguiente **problema:** ¿Cómo adaptar los procesos contables cubanos del sistema integral Cedrux al ERP Odoo 8.0? Para ello se plantea como **objeto de estudio**: Personalización de sistemas ERP. El **campo de acción** se centra en: Personalización de módulos del sistema ERP Odoo 8.0, con el **objetivo general** de: Personalizar el módulo de Contabilidad de Odoo 8.0 en función de las normas contables cubanas.

Del objetivo general de la investigación se trazan los siguientes **objetivos específicos:**

1. Elaborar el marco teórico de la investigación relativo a la personalización de módulos de Odoo 8.0.
2. Diseñar el módulo de Contabilidad de Odoo 8.0 en función de las normas contables cubanas.
3. Implementar el módulo de Contabilidad de Odoo 8.0 en función de las normas contables cubanas.
4. Validar la propuesta de solución mediante la aplicación de técnicas, métricas y pruebas.

La idea a defender es: Con la personalización del módulo de Contabilidad de Odoo 8.0 se logrará el desarrollo de una localización basado en las normas contables cubanas.

Métodos de investigación

- **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. Expresa de forma teórica la esencia del objeto, permitiendo unir el estudio de la estructura del objeto de investigación con su concepción histórica (González, 2002). Su aplicación se evidenció en el estudio de Cedrux a través del informe de CEIGE.
- **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). Su aplicación se evidenció en la elaboración de diagramas, mapas conceptuales y prototipos durante el desarrollo del sistema.

- **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 de Gestión Integral Cedrux, en el cual se analizó el estado actual de este sistema, recolectando así mucha información de su desenvolvimiento, lo que trajo consigo formular la hipótesis de que un nuevo sistema tecnológico apoyaría con mayor eficiencia los procesos contables en nuestro país.
- **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 la especialista del proyecto Cedrux, obteniendo con esto valiosa información del negocio contable de Cedrux y del esclarecimiento de los requisitos.
- **Observación:** La observación científica es la percepción planificada dirigida a un fin y relativamente prolongada de un hecho o fenómeno. Es el instrumento universal del científico, se realiza de forma consciente y orientada a un objetivo determinado (González, 2002). Se utilizó la observación al apreciar sistemáticamente el negocio contable de Cedrux.

Estructuración del trabajo

El trabajo consta de tres capítulos que cubren la fundamentación teórica, diseño del sistema, implementación y prueba, además de las conclusiones, recomendaciones, referencias bibliográficas, glosario de términos y anexos.

Capítulo1. Fundamentación teórica: En este capítulo se realiza una investigación del módulo de Contabilidad de Odoo y las normas cubanas de contabilidad. Se realiza un estudio sobre cómo personalizar el módulo de Contabilidad de Odoo. Se definen además la metodología de desarrollo, herramientas y lenguajes a utilizar.

Capítulo2. Análisis y diseño de la propuesta de solución: En este capítulo se identifican los requisitos funcionales y no funcionales de la aplicación. Se confeccionan los diagramas de clase de diseño, los diagramas de casos de uso del sistema, diagrama de entidad-relación, el modelo conceptual del sistema, así como los patrones arquitectónicos, los patrones de diseño y la validación de diseño.

Capítulo3. Implementación y validación de la propuesta de solución: En este capítulo se elabora el diagrama de componentes, se mencionan los estándares de codificación, así como se describen las pruebas a realizar, con el objetivo de comprobar las funcionalidades del sistema.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se realiza el estudio del módulo de Contabilidad de Odoo, sus características y una descripción de las principales normas por la que se rigen los sistemas contables de nuestro país. Se enuncian los principales conceptos relacionados con la gestión contable y se define la metodología, tecnologías y herramientas que serán empleadas para la construcción de la solución.

1.1 Marco teórico. Conceptos y definiciones

Las normas contables constituyen bases o reglas establecidas con carácter obligatorio en diferentes países con el objetivo de regir y direccionar la contabilidad financiera. En Cuba existen un conjunto de resoluciones y documentos que condicionan la validez del proceso contable. Los sistemas contables en Cuba se rigen por la Resolución 340/2015 del Ministerio de Finanzas y Precios, la cual aborda el procedimiento para dictaminar sobre el grado de adaptación a las normas contables cubanas de los sistemas contables-financieros soportados sobre las tecnologías de la información. Según esta resolución, los procesos fundamentales que deben conformar el módulo de Contabilidad son (Precios, 2015):

Apertura de saldos en el Mayor

La apertura de saldos en el Mayor, cuya fuente ha de ser un Balance de Comprobación, debe estar condicionada a que no se haya efectuado el cierre de la misma. Otros saldos que se incorporen o actualicen tienen que ser absolutamente a través del Comprobante de Operaciones (Precios, 2015).

Cierre de apertura

El cierre de la apertura de saldos del Mayor tiene que estar condicionado a que esté cuadrado el balance que dichos saldos generan (Precios, 2015).

Operaciones

El proceso de Operaciones debe contener al menos las siguientes opciones (Precios, 2015):

- Pantalla de captación de las partidas que conforman un comprobante, así como de las correcciones inherentes.
- Cuadre automático del comprobante en su captación.
- Posibilidad de salir de la pantalla de captación independientemente a que se haya concluido el pase de todas las partidas del comprobante.
- Borre de un comprobante mientras no haya sido traspasado al Mayor.
- Imposibilidad de borrar las cuentas o subcuentas con movimientos y con saldos durante el ejercicio económico.
- Traspaso del comprobante al Mayor a través de opción.

- Información sobre los comprobantes captados cuya situación es:
    a) Traspasados al Mayor.
    b) No traspasados al Mayor.
    c) No cuadrados.
  - Visualización de un comprobante de los períodos contables de los tres ejercicios fiscales anteriores, como mínimo.
  - Posteo (traspaso al Mayor), bajo el cumplimiento de las siguientes premisas:
    a) Proponer el número consecutivo de manera automática.
    b) Que los comprobantes estén cuadrados.
    c) Que la fecha de los comprobantes esté en correspondencia con el período contable vigente.
    d) Que previo a iniciarse la operación de posteo y como parte integrante de la misma el sistema emita en un registro impreso o visual del comprobante o lote de comprobantes a postear.
    e) Inclusión en un Fichero Histórico de las operaciones posteadas.

Informes

En el proceso de Informes deben ser emitidos básicamente los siguientes reportes (Precios, 2015):

- Edición de comprobantes con el detalle de la operación para cualquier período.
- Consultas al Mayor, donde se muestre de cada cuenta solicitada, el saldo inicial, el saldo de cada mes desde enero del año en curso hasta el mes vigente y el saldo hasta la fecha.
- Balance de Comprobación a nivel de cuentas y subcuentas.
- Listado del Fichero Histórico de comprobantes para cualquier período.

Cierres mensual y anual

Los procesos de cierres mensual y anual deben estar condicionados a los siguientes eventos (Precios, 2015):

Cierre mensual

- El nuevo período contable debe ser el inmediato siguiente al período contable vigente.
- Se efectuará solo cuando hayan sido cerrados los demás módulos.
- No deben quedar comprobantes pendientes de posteo.
- Aviso para salva de información.

Cierre anual

- Haberse realizado el cierre del último mes del período contable.
- Que se haya efectuado el cierre contable de las cuentas nominales.
- Que se hayan emitido el resto de los Estados Financieros establecidos.

- Aviso para salva de información.

Existen, además, otras resoluciones por las que se rige la contabilidad en nuestro país (Precios, 2015):

Resolución No. 1173/2015. Norma Específica de Contabilidad No. 5 “Proformas de Estados Financieros para la actividad empresarial, unidades presupuestadas de tratamiento especial y el sector cooperativo agropecuario y no agropecuario” (Precios, 2015).

Los Estados Financieros integran otro de los elementos primordiales, que sintetizan la ejecución de la contabilidad y se integran por las proformas establecidas en esta norma y el resto de los elementos contenidos en la Norma Cubana de Contabilidad No. 1 “Presentación de Estados Financieros”. Las proformas son (Precios, 2015):

- Proforma EFE 5920 - 03 Estado de Situación.
- Proforma EFE 5921 - 03 Estado de Rendimiento Financiero.
- Proforma EFE 5922 - 03 Estado Rendimiento Financiero - Actividad de Seguro y Reaseguro.
- Indicaciones Metodológicas de la Proforma EFE 5922 - 03 Estado de Rendimiento Financiero - Actividad de Seguro y Reaseguro.
- Proforma EFE 5924 - 03 Estado de Gastos por Elementos.
- Proforma EFE 5925 - 03 Estado de Inversiones.
- Proforma EFE 5926 - 01 Estado de Valor Agregado Bruto.

Resolución No. 1100/2015. Nomenclador de cuentas para la actividad empresarial, unidades presupuestadas de tratamiento especial y el sector cooperativo agropecuario y no agropecuario (Precios, 2015).

Nomenclador de cuentas: El nomenclador de cuentas debe contar con un nombre, un código contable, una naturaleza (acreedora, deudora o mixta), además de agruparse en grupos y subgrupos contables.

Resolución No. 14/2007. Registros, Submayores, Mayores y Comprobante de operaciones del Manual de Normas de Control Interno (Precios, 2015).

A continuación, se expresan las características de los informes contables referidos anteriormente:

Mayor

Su objetivo es resumir las operaciones que afectan las cuentas control, en las entidades que registran sus operaciones en forma manual, mecanizada o computarizada. Este Registro se puede mostrar en pantalla solamente u obtener su impresión. Debe contener los siguientes datos obligatorios:

- Nombre y código de la entidad.
- Nombre y código de la cuenta control, según el Nomenclador de Cuentas establecido.
- Fecha que corresponde al Comprobante de Operaciones que se registra.
- Clave y número que corresponde al Comprobante de Operaciones que se registra.
- Breve explicación de la operación.
- Importe que se debita (Columna DEBE).
- Importe que se acredita (Columna HABER).
- Diferencia entre los totales de las columnas de débito y crédito (Columna SALDO).
- Número consecutivo del folio que conforma el Libro habilitado.

Comprobante de operaciones

Su objetivo es contabilizar las operaciones anotadas en los diferentes Submayores específicos que se habiliten en los diferentes módulos, para su posterior pase a las cuentas control del Mayor y subcuentas de los Submayores, afectadas. Además, se utiliza para contabilizar aquellas operaciones que no se recogen en ningún Submayor específico y que se generan en el módulo de Contabilidad General. Debe contener los siguientes datos obligatorios:

- Nombre y código de la entidad.
- Código de la cuenta según el Nomenclador establecido.
- Código de la subcuenta según el Nomenclador establecido.
- Espacio habilitado para los análisis cuando proceda, dentro de las cuentas o subcuentas establecidas. (Cuando no son emitidos por los diferentes módulos).
- Nombre de las cuentas y subcuentas. (Análisis cuando no son emitidos por los diferentes módulos).
- Importe de los débitos o créditos efectuados a las subcuentas y/o análisis que corresponden.
- Importe que se debita a las cuentas control (DEBE).
- Importe que se acredita a las cuentas control (HABER).
- Se detalla claramente y con precisión el origen de las operaciones que se asienten.
- Número consecutivo asignado al comprobante.

Submayor

Su objetivo es analizar las subcuentas y análisis de las cuentas que lo requieran excepto aquellas que son analizadas en Submayores específicos en el resto de los módulos del sistema, con la finalidad de obtener todos y cada uno de sus saldos, con vista al cuadre mensual con las cuentas control del Mayor. Este registro se puede mostrar en pantalla solamente u obtener su impresión. Debe contener los siguientes datos obligatorios:

- Nombre y código de la entidad.

- Nombre y código de la cuenta, según el Nomenclador de Cuentas establecido.
- Nombre y código de la subcuenta, según el Nomenclador de Cuentas establecido.
- Nombre y código del análisis cuando proceda, que corresponde a la cuenta y/o subcuenta, establecido en el Nomenclador de Cuentas.
- Código del deudor o acreedor, en el caso de los modelos habilitados para el análisis de las cuentas de control correspondientes.
- Fecha que corresponde al Comprobante de Operaciones que se contabiliza.
- Clave y número que corresponde al Comprobante de Operaciones que se registra.
- Breve explicación de la operación.
- Importe que se debita (Columna DEBE).
- Importe que se acredita (Columna HABER).
- Diferencia entre los totales de las columnas de débito y crédito. (Columna SALDO)

A continuación, se explican algunos de los conceptos que se utilizan en la contabilidad:

Grupos económicos

También denominados grupos y subgrupos contables, son el mecanismo mediante el cual se define un conjunto de grupos y subgrupos dentro de los cuales se organizarán los Contenidos económicos (Precios, 2015).

Contenido económico

Es el mecanismo mediante el cual se define un rango de códigos en el que se incluyen las cuentas definidas en su concepto contable. Esto permite identificar, clasificar y registrar un elemento o hecho económico realizado por una empresa (Precios, 2015).

Cuentas contables

Las cuentas no son otra cosa que medios contables con instrumentos de operación, mediante los cuales podemos subdividir el activo, el pasivo y el capital y agruparlos de acuerdo a ciertas características de afinidad, las cuales nos permiten graficar todos los aumentos y disminuciones que ocurren en los diversos elementos de la ecuación (Precios, 2015).

Asiento contable

Un asiento es una anotación en el libro de contabilidad que refleja los movimientos económicos de una persona o institución. Se realiza cada vez que la empresa contabiliza una entrada contable relacionada con la actividad que realiza (Precios, 2015).

1.2 Módulo de Contabilidad de Odoo

El módulo de Contabilidad de Odoo abarca la contabilidad general, las cuentas pendientes de cobro, las cuentas pendientes de pago, la reconciliación bancaria, el control de costes y el control

presupuestario. El módulo de Contabilidad de Odoo ofrece la mejor manera de colaborar con los contables, clientes y proveedores. Guarda las transacciones de manera automática, y controla fácilmente toda la actividad contable desde una misma aplicación. La interfaz de usuario del software de contabilidad permite aumentar la productividad de los empleados. Da acceso además a los contables a los últimos datos del negocio de manera que todos se mantengan siempre actualizados con los últimos cambios e información. Odoo importa los estados de las cuentas bancarias, envía al banco las órdenes y términos de pago para los proveedores y acreedores. Crea también facturas de aspecto profesional, y recibe los pagos online. Es capaz de facturar automáticamente en función de órdenes de venta, órdenes de entrega, contratos, o según el tiempo y material. Odoo integra las operaciones contables**,** las horas trabajadas**,** proyectos**,** facturas**,** gastos**,** entre otros**.** Con Odoo no se necesita guardar manualmente las transacciones, todos los datos se archivarán automáticamente en función de las preferencias del usuario (Odoo, 2016).

Odoo no solo permite llevar la contabilidad, sino que existen otras áreas, como finanzas, cobros y pagos, que, enlazadas con todo el resto de módulos de aplicación, interactúan con otras áreas que optimizan la gestión global de la empresa.

1.3 Adaptación de las normas contables en Odoo

Nomenclador de cuentas

El nomenclador de cuentas en el módulo de Contabilidad de Odoo se confecciona por niveles. Los datos que registran son: cuenta, padre, código contable, tipo interno y tipo de cuenta. Se debe crear una cuenta raíz que agrupe al nomenclador de cuentas de la entidad. Subordinada a la raíz, se crean los grupos y subgrupos contables. Asociado a los subgrupos, se definen las cuentas y subcuentas hasta definir el último nivel deseado de cada cuenta contable. Las cuentas creadas desde la raíz hasta el penúltimo nivel se registran de tipo interno ¨Vista¨ y el último nivel de la cuenta se registra de tipo interno ¨Regular¨.

Cuentas Padre: Son cuentas contables, lo que definidas en un nivel superior.

Tipo interno: Posee siete tipos**,** los cuales pueden ser**:**

Vista: Permite la agrupación de cuentas que agrupan a otras cuentas.

- **Regular**: Cuenta real, poseedora de dinero.
- **A cobrar**: Cuentas a cobrar por la venta de mercancías o la prestación de servicios.
- **A pagar**: Cuentas a pagar por la compra de bienes o servicios.
- **Liquidez**: Cuentas a cobrar por algún préstamo de un servicio brindado por la empresa.
- **Consolidación**: Son cuentas para la consolidación multi-empresa.
- **Cierre**: Se utiliza para cuentas de depreciación.

En la Figura 1 se muestra la interfaz para adicionar una cuenta contable en el módulo de Contabilidad de Odoo:

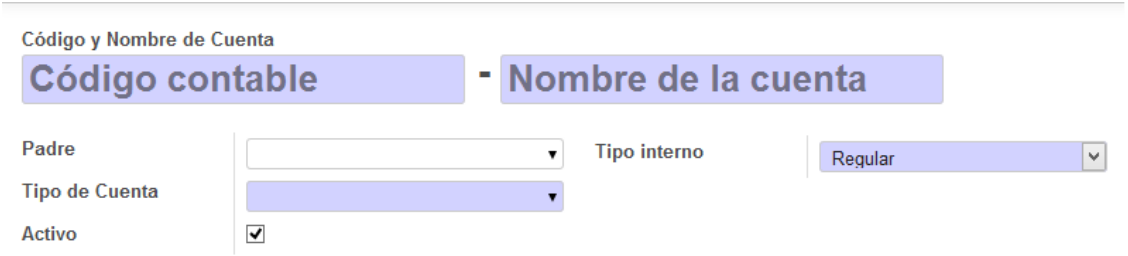


Figura 1. Interfaz para la gestión de las cuentas contables (Odoo, 2016).

Mayor

Referido a este criterio el módulo de Contabilidad de Odoo genera este informe diferente a la Resolución No. 14/2007 Registros, Submayores, Mayores y Comprobante de operaciones del Manual de Normas de Control Interno, la cual expresa que el informe mayor debe contener:

- Nombre y código de la entidad (No está en el módulo de Contabilidad de Odoo).
- Nombre y código de la cuenta según el nomenclador de cuentas (No está en el módulo de Contabilidad de Odoo).
- Fecha que corresponde al comprobante de operaciones que se registra (Si está en el módulo de Contabilidad de Odoo).
- Clave y número que corresponde al Comprobante de Operaciones que se registra (No está en el módulo de Contabilidad de Odoo).
- Breve explicación de la operación (No está en el módulo de Contabilidad de Odoo).
- Importe que se debita. (Columna DEBE) (Si está en el módulo de Contabilidad de Odoo).
- Importe que se acredita. (Columna HABER) (Si está en el módulo de Contabilidad de Odoo).
- Diferencia entre los totales de las columnas de débito y crédito. (Columna SALDO) (Si está en el módulo de Contabilidad de Odoo).
- Número consecutivo del folio que conforma el Libro habilitado (No está en el módulo de Contabilidad de Odoo).

  En la Figura 2 se muestra el reporte generado en el módulo de Contabilidad de Odoo del Libro Mayor:

2016-05-12 22:33 Your Company /

Your CompanyLibro mayor

Árbol de cuentas
Your Company

Ejercicio fiscal
2016

Diarios
VENTA, GASTO, AVENT, ACOMP, Vario, OPEJ, BNC1, BNC2, 12345, 98

Mostrar cuenta
Con movimientos

Filtrado por:
No filtrado

Ordenado por:
Date

Movimientos destino
Todos los asientos publicados

Fecha	LIBRO	Partner	Ref.	Movimiento	Etiqueta del asiento	Contrapartida	Debe	Crédito	Progreso	Moneda
002 nueva34							0,00 €	0,00 €	0,00 €	

Figura 2. Reporte generado para el Libro Mayor (Odoo, 2016).

Comprobante de operaciones

En el módulo de Contabilidad de Odoo no existe un comprobante de operaciones como tal con ese nombre, sin embargo, existen los asientos contables, compuestos por apuntes contables y asientos diarios, los cuales son los encargados de registrar los movimientos realizados a las cuentas contables.

Estos asientos contables no cumplen con la Resolución No. 14/2007. Registros, Mayores Submayores y Comprobante de operaciones del Manual de Normas de Control Interno.

A continuación, se enuncian los campos válidos para este tipo de documento:

- Nombre y código de la entidad (No está en el módulo de Contabilidad de Odoo)
- Código de la cuenta según el Nomenclador establecido (Si está en el módulo de Contabilidad de Odoo).
- Código de la subcuenta según el Nomenclador establecido (No está en el módulo de Contabilidad de Odoo).
- Espacio habilitado para los análisis cuando proceda, dentro de las cuentas o subcuentas establecidas. (Cuando no son emitidos por los diferentes Módulos) (No está en el módulo de Contabilidad de Odoo).
- Importe de los débitos o créditos efectuados a las subcuentas y/o análisis que corresponden (No está en el módulo de Contabilidad de Odoo).
- Importe que se debita a las cuentas control. (DEBE) (Si está en el módulo de Contabilidad de Odoo).
- Importe que se acredita a las cuentas de control (HABER) (Si está en el módulo de Contabilidad de Odoo).

- Se detalla claramente y con precisión el origen de las operaciones que se asienten (No está en el módulo de Contabilidad de Odoo).
- Número consecutivo asignado al comprobante (No está en el módulo de Contabilidad de Odoo).

En la Figura 3 se muestra cómo se adiciona un asiento contable en el módulo de Contabilidad de Odoo:

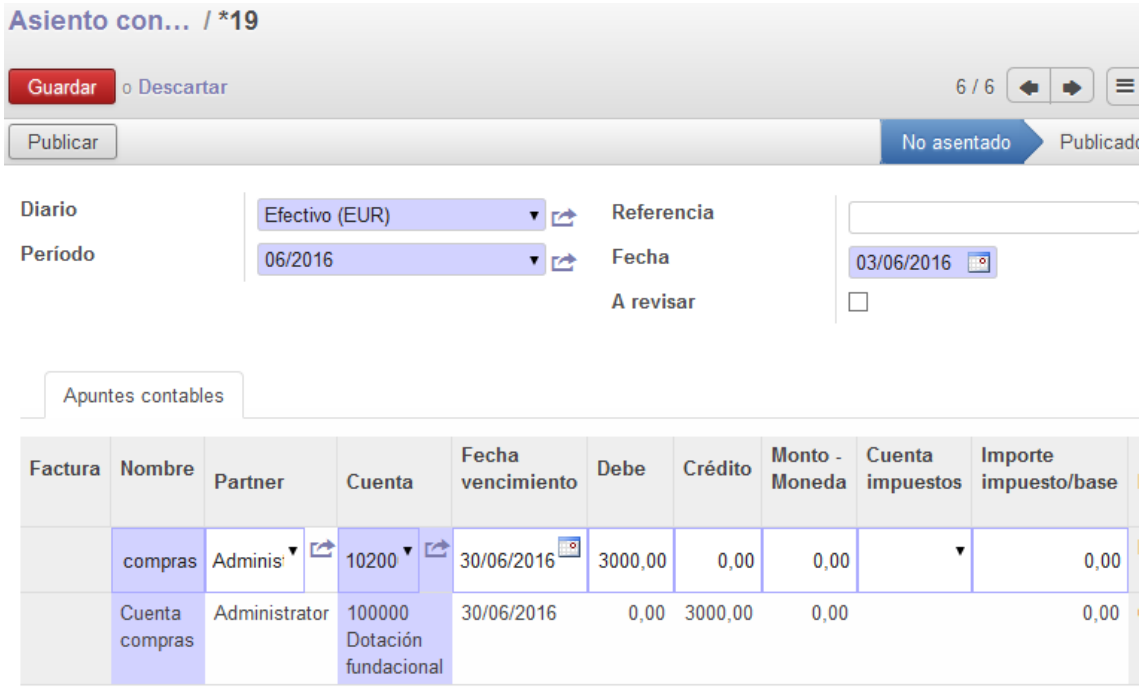


Figura 3. Interfaz para la creación o gestión de los asientos contables (Odoo, 2016).

Cierre de período

El cierre de período en el módulo de Contabilidad de Odoo se realiza igual al contemplado en la Resolución 340/2015.

A continuación, se enuncian los eventos válidos del cierre de período:

- El nuevo período contable debe ser el inmediato siguiente al período contable vigente (Si está en el módulo de Contabilidad Odoo).
- Se efectuará solo cuando hayan sido cerrados los demás módulos (En Odoo el cierre de período se realiza de forma centralizada, o sea, el módulo de Contabilidad es el encargado de realizar el cierre contable).
- No deben quedar comprobantes pendientes de posteo (Si está en el módulo de Contabilidad Odoo).
- Aviso para salva de información (Si está en el módulo de Contabilidad Odoo).

En la Figura 4 se muestra cómo cerrar un período en el módulo de Contabilidad de Odoo:

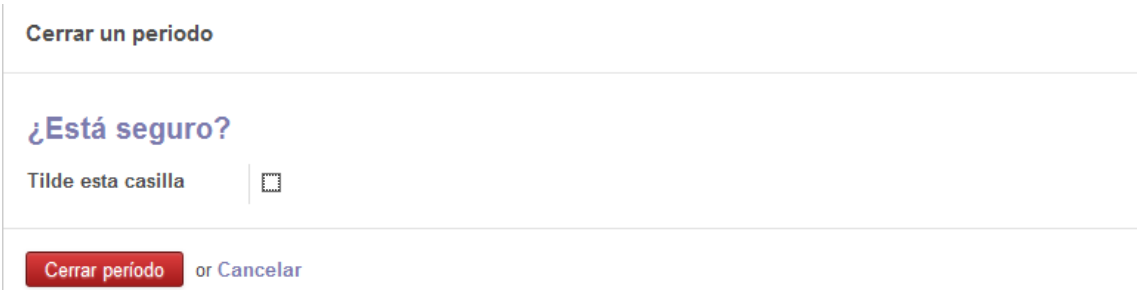


Figura 4. Interfaz para el cierre de un período (Odoo, 2016).

Cierre de un año fiscal

El cierre de año fiscal en el módulo de Contabilidad de Odoo se realiza de manera diferente al contemplado en la Resolución 340/2014.

A continuación, se enuncian los eventos válidos para el cierre anual:

- Haberse realizado el cierre del último mes del período contable (No está en el módulo de Contabilidad de Odoo).
- Que se haya efectuado el cierre contable de las cuentas nominales (Si está en el módulo de Contabilidad de Odoo).
- Que se hayan emitido el resto de los Estados Financieros establecidos (No está en el módulo de Contabilidad de Odoo).
- Aviso para salva de información (No está en el módulo de Contabilidad de Odoo)

En la Figura 5 se muestra cómo cerrar un año fiscal en el módulo de Contabilidad de Odoo:

Figura 5. Interfaz para el cierre de un año fiscal (Odoo, 2016).

Resultados de la adaptación de las normas contables cubanas

A partir del estudio realizado al módulo de Contabilidad de Odoo y a las normas contables cubanas se llegó a la conclusión de la presencia en Odoo de funcionalidades contempladas en las resoluciones que se nombran de manera diferente, es decir existe contrariedad en la terminología utilizada. El asiento contable (Comprobante de operaciones en la resolución) constituye un ejemplo del cambio de términos. La ausencia de diferentes funcionalidades en Odoo como el cálculo del submayor y la creación del contenido económico son otro ejemplo a mencionar. Por último, se evidencia en Odoo, la inclusión de funcionalidades plasmadas en las normas contables cubanas,

pero que su cumplimiento se expresa de manera diferente. Un ejemplo es la elaboración del informe del mayor, la creación de una cuenta contable y la realización del cierre anual.

1.4 Personalización de sistemas ERP

La personalización es el desarrollo de sistemas, aplicaciones, programas, sitios web, páginas web, entre otros, de acuerdo con las especificaciones del cliente, es decir, el software se construye y forma a las necesidades únicas del cliente contratante (a gusto del cliente). O sea, no es más que adaptar algo a las características, al gusto o a las necesidades de una persona, un cliente, una empresa.

La personalización puede ser total o parcial:

- El software puede ser 100% personalizado, es decir, totalmente desarrollado específicamente para el cliente en cuestión.
- El software se puede adaptar a las necesidades del cliente, a partir de un software básico existente.

Personalización de módulos en Odoo

Existen dos maneras de llevar a cabo la personalización en Odoo: mediante programación de los archivos de código Python y XML o a través de la misma interfaz web (Yeray Fernández, 2014).

Programación de los archivos mediante código Python y XML

Este método consiste en la alteración del código fuente de los archivos correspondientes al módulo o a los módulos que se quieran cambiar. Para efectuar cambios en el sistema basta con realizar las modificaciones pertinentes en el código del módulo para cargar después la actualización vía web. Se eliminan, añaden y/o modifican los elementos existentes en el archivo correspondiente del módulo y se realizan los cambios pertinentes en los archivos XML para que quede reflejado en las vistas. Estas alteraciones se realizan manualmente, como, por ejemplo, con la herramienta IDE PyCharm. El software reconoce los archivos alterados y aparecen en el apartado “Actualizar lista de módulos”, que está en la sección “Módulos” del menú “Configuración”. Para cargar la actualización hay que hacer clic en “Actualizar”. A través del código se pueden añadir nuevas variables, nuevos menús, nuevos campos, entre otros elementos. Todos esos cambios deben ser consistentes y ser declarados por igual en los distintos archivos para que el programa los reconozca y se puedan implementar en la solución final.

Vista - List

```
<record id="view_tree_todo_task" model="ir.ui.view">
  <field name="name">To-do Task Tree</field>
  <field name="model">todo.task</field>
  <field name="arch" type="xml">
    <tree colors="gray:is_done==True">
      <field name="name"/>
      <field name="is_done"/>
    </tree>
  </field>
</record>
```

Figura 6. Creación de una vista de tipo lista (árbol) a través del código (Negocio, 2016).

View - Action

- Agregamos action:

```
<!-- Action to open To-do Task list -->
<act_window id="action_todo_task" name="To-
do Task" res_model="todo.task"
view_mode="tree,form" />
```

Figura 7. Creación de una acción de tipo act_window id a través del código) (Negocio, 2016).

Entre las ventajas de realizar el cambio mediante la programación se encuentran las siguientes:

- Las posibilidades de modificación son enormes. Se pueden añadir los elementos que se consideren necesarios, declarar su tipo y función de una sola vez.
- Se pueden trasformar los campos bases, que de otra manera permanecen inmutables.
- Se da la posibilidad de aplicar numerosos cambios al programa simultáneamente.
- Ofrece un alto grado de versatilidad.

A través de la interfaz

El proceso consiste básicamente en definir nuevas variables o campos y coordinarlo con las vistas para que se puedan utilizar de manera funcional. De la misma manera se pueden modificar los menús y las acciones ligadas a estos.

Para comenzar a trabajar con este método en la modificación del módulo, es necesario activar el modo desarrollador. Esta opción se encuentra al hacer clic en “Acerca de Odoo”. Tras activar este modo desarrollador, en cada pantalla de Odoo aparecerá un menú desplegable en la esquina superior, a la izquierda del nombre de la sección. Lo más importante que se puede llevar a cabo desde él es la edición de las vistas, donde se puede añadir y quitar campos, introducir nuevos tipos de vistas, entre otras modificaciones.

Las siguientes figuras muestran la secuencia de pasos para activar el modo desarrollador en el módulo contable de Odoo:

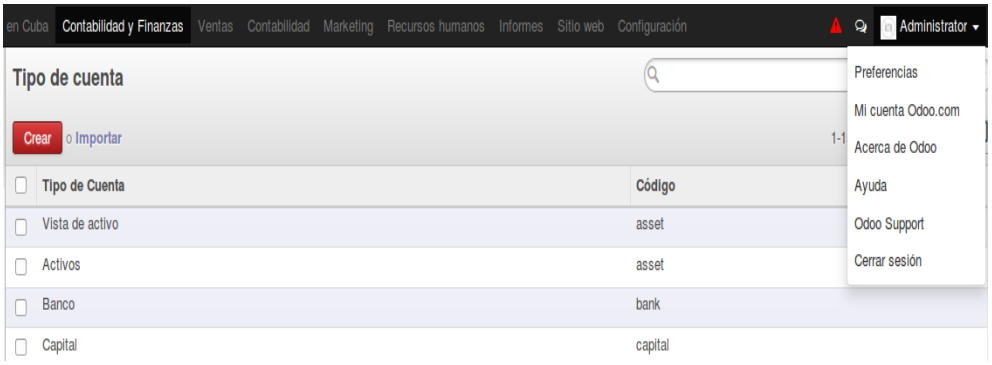


Figura 8. Interfaz para la activación del modo desarrollador paso 1 (Odoo, 2016).

Figura 9. Interfaz para la activación del modo desarrollador paso 2 (Odoo, 2016).

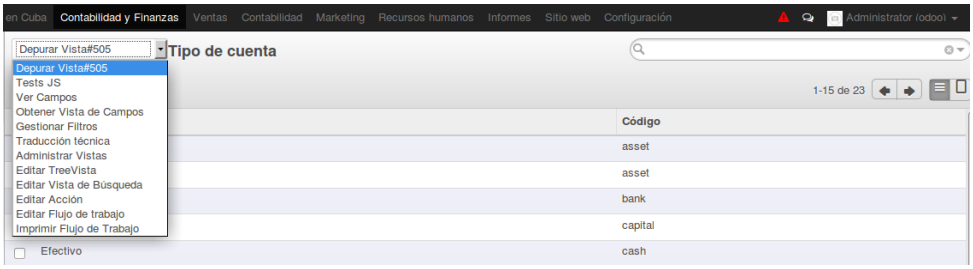


Figura 10. Interfaz para la activación del modo desarrollador paso 3 (Odoo, 2016).

Las ventajas que presenta esta forma de modificación son:

- Es un método intuitivo y rápido.
- No es necesario salir del ERP
- Se puede ir observando el resultado mientras se van realizando los cambios.
- Los campos se ven de manera compacta, de forma que en una misma ventana se pueden observar sus propiedades y modificarlas, asignar grupos, añadir menús.

Por el contrario, presenta los siguientes inconvenientes:

- Hay que coordinar la coherencia con las vistas para evitar errores y fallos.
- No se pueden cambiar los campos base. En este sentido, la personalización es menos profunda.

1.5 Estudio de Herramientas y Tecnologías

1.5.1 Metodología de desarrollo de software Variación de AUP para la UCI

De las tres fases que propone AUP-UCI, se desarrollará la fase Inicio y Ejecución en el desarrollo del presente trabajo. En el Inicio se llevaron a cabo las actividades relacionadas con la planeación del desarrollo de la tesis donde se definió el cronograma. En la fase de ejecución se ejecutaron las actividades requeridas para desarrollar el modelo de negocio, la obtención requisitos, el análisis y diseño, además de la implementación y las pruebas internas (Sánchez, 2015).

En la Figura 11 se muestran las fases de la metodología AUP-UCI:

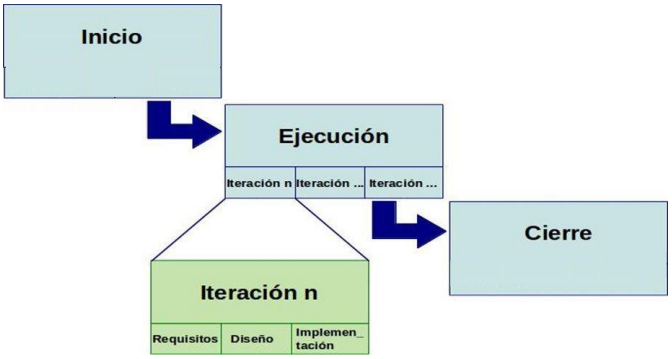


Figura 11. Fases de desarrollo de software (Sánchez, 2015).

A partir de que el Modelado de negocio propone tres variantes a utilizar en los proyectos (Casos de uso del negocio(CUN), Diagramas de proceso del negocio(DPN) o Modelo conceptual(MC)) y existen tres formas de encapsular los requisitos (CUS, HU, DRP), surgen cuatro escenarios para modelar el sistema en los proyectos, de los cuales, se escogió el segundo debido a que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma se modela exclusivamente los conceptos fundamentales del negocio (Sánchez, 2015).

En la Figura 12 se muestra el escenario No 2: Proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS.

Figura 12. Escenario 2. Elaboración propia.

1.5.2 Herramienta para el modelado

Visual Paradigm 8.0

Es una herramienta para desarrollo de aplicaciones utilizando 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 (Paradigm, 2016).

1.5.3 Lenguaje de modelado UML

El Lenguaje Unificado de Modelado (UML) 2.4 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).

1.5.4 ERP

Odoo 8.0

Odoo (conocido anteriormente como OpenERP) es un sistema de ERP integrado de código abierto producido por OpenERP s.a. Actualmente han cambiado el nombre de OpenERP por el de Odoo. El fabricante define su producto como una alternativa de código abierto a SAP ERP y Microsoft Dynamics, así 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. Entre las principales características de Odoo 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, cvs, listas de correo, desarrollo comunitario y traducciones en Launchpad.) (Entidades(CEIGE), Centro de Informatización de, 2015). 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 descantan Eclipse o PyCharm + PyDev. Odoo 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 (Entidades(CEIGE), Centro de Informatización de, 2015).

1.5.5 Lenguaje de programación

Python 2.7.6

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 dela aplicación se utilizó Python debido a que es el lenguaje de programación que utiliza Odoo.

1.5.6 Lenguaje de Marcas Extensible

XML 1.2

XML, 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 y permite definir la gramática de lenguajes específicos (de la misma manera que HTML 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 debido a que es el lenguaje de marcas que utiliza Odoo.

1.5.7 Marco de trabajo

OpenObject 1.0

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 GNU AGPL v3 (Odoo, 2016). Se utilizó como framework OpenObject debido a que es el que utiliza Odoo.

1.5.8 Herramientas para administrar el 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).

Postgresql 9.3

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 joins. 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 BSD8. 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 usa Odoo.

1.5.9 Servidor de aplicaciones

Apache 2.2.9

El servidor HTTP Apache 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. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que alguien quería que tuviese la connotación de algo que es firme y enérgico, pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de internet. Además, Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA (foundation, 2016). Se utilizó como servidor de aplicaciones Apache porque es el que define Odoo.

1.5.10 Entorno de desarrollo integrado (IDE)

PyCharm 4.5

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 bondades. 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). Se utilizó el Pycharm como IDE porque es el IDE para la programación en Python.

1.6 Conclusiones parciales

Según lo expuesto en el capítulo, se arribaron a las siguientes conclusiones:

- El estudio de las normas cubanas contables permitió entender la gestión de los procesos contables en Cuba.
- El estudio del módulo de Contabilidad de Odoo demostró la necesidad de realizar una personalización debido al incumplimiento de las normas contables cubanas.
- La selección de las herramientas, tecnologías y metodología a utilizar permitió establecer el entorno de desarrollo en el que se implementará la propuesta de solución.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN

Introducción

Conocer el funcionamiento del negocio y sus reglas es imprescindible en la construcción de un software efectivo. En este capítulo se define lo que debe hacer la aplicación a través de la caracterización del negocio, se confeccionan los artefactos de ingeniería del software que permitirán desarrollar el sistema de forma más fácil. Se realiza el modelo conceptual, se hace un estudio del negocio permitiendo identificar los requisitos funcionales y no funcionales, así como los diagramas de clases de diseño, diagramas para alcanzar una mayor comprensión del funcionamiento del sistema.

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 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, 2002).

En la Figura 13 se muestra el modelo conceptual del sistema, y en la Tabla 1 se encuentra la descripción del modelo conceptual.

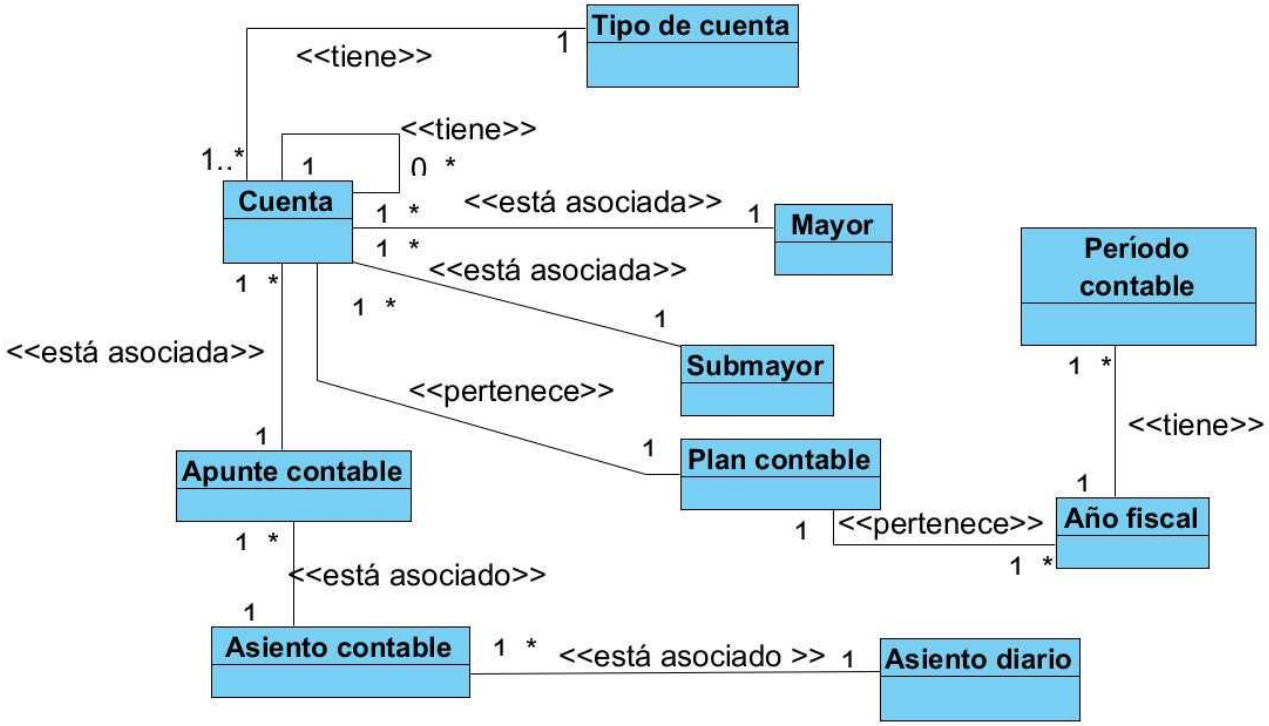


Figura 13. Modelo conceptual del sistema. Elaboración propia.

Nombre	Descripción	¿Puede ser nulo?	¿Es único?	Clases válidas	Clases inválidas
Tipo de cuenta	Clasificación de una cuenta contable	No	Sí	Cadena de caracteres	Campos nulos, caracteres extraños y espacios
Cuenta	Unidad contable que resume un hecho económico en la empresa	No	Sí	Cadena de caracteres	Campos nulos, caracteres extraños y espacios
Apunte contable	Anotaciones que se realizan en el diario de la empresa	No	Sí	Cadena de caracteres	Campos nulos, caracteres extraños y espacios
Asiento contable	Conjunto de apuntes contables que se registran en	No	Sí	Cadena de caracteres	Campos nulos, caracteres extraños y espacios

	el diario de la empresa				
Mayor	Libro donde se recogen las operaciones realizadas a las cuentas de la empresa	No	SÍ	Conjunto de cuentas	Campos nulos, caracteres extraños y espacios
Submayor	Libro donde se recogen las operaciones realizadas a las subcuentas de la empresa	No	Sí	Conjunto de subcuentas	Campos nulos, caracteres extraños y espacios
Plan contable	Documento que recoge todas las cuentas contables de la empresa	No	No	Conjunto de cuentas	Campos nulos, caracteres extraños y espacios
Asiento diario	Libro donde se recogen diariamente los hechos económicos de una empresa	No	Sí	Conjunto de cuentas	Campos nulos, caracteres extraños y espacios
Año Fiscal	Duración del ejercicios fiscal de la empresa	No	Sí	Fecha	Campos nulos, caracteres extraños y espacios
Período contable	Espacio de tiempo en el que se realiza el registro contable de la empresa	No	Sí	Fecha	Campos nulos, caracteres extraños y espacios

Tabla 1. Descripción del modelo conceptual.

2.2 Disciplina 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).

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).

Entrevista

La entrevista es de gran utilidad para obtener información cualitativa como opiniones, o descripciones subjetivas de actividades. Se entrevistó a la analista del proyecto Cedrux para un mejor entendimiento de los procesos contables.

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 casos de uso y su esclarecimiento.

2.2.2 Requisitos funcionales

A continuación en la Tabla 2, se listan los requisitos funcionales definidos para darle solución al problema, agrupados en tres paquetes:

No	Requisitos Funcionales	Prioridad
Configuración		
RF1	Adicionar tipo de cuenta	Alta
RF2	Modificar tipo de cuenta	Alta
RF3	Eliminar tipo de cuenta	Alta
RF4	Listar tipo de cuenta	Media
RF5	Mostrar vista formulario de tipo de cuenta	Baja

RF6	Buscar tipo de cuenta	Media
RF7	Adicionar cuenta contable	Alta
RF8	Modificar cuenta contable	Alta
RF9	Eliminar cuenta contable	Alta
RF10	Listar cuenta contable	Baja
RF11	Mostrar vista formulario de cuenta contable	Media
RF12	Buscar cuenta contable	Media
RF13	Adicionar año fiscal	Alta
RF14	Modificar año fiscal	Alta
RF15	Eliminar año fiscal	Alta
RF16	Listar año fiscal	Media
RF17	Mostrar vista formulario de año fiscal	Baja
RF18	Buscar año fiscal	Media
RF19	Adicionar período al año fiscal	Alta
RF20	Modificar período al año fiscal	Alta
RF21	Eliminar período del año fiscal	Alta
RF22	Listar período del año fiscal	Media
RF23	Mostrar vista formulario de período del año fiscal	Baja
RF24	Buscar período del año fiscal	Media
RF25	Cerrar período	Alta
Contabilidad		
RF26	Adicionar asientos diarios	Alta
RF27	Modificar asientos diarios	Alta
RF28	Eliminar asiento diario	Alta

RF29	Listar asiento diario	Media
RF30	Mostrar vista formulario de diario	Baja
RF31	Buscar asiento diario	Media
RF32	Mostrar plan de cuentas	Media
RF33	Adicionar asiento contable	Alta
RF34	Modificar asiento contable	Alta
RF35	Eliminar asiento contable	Alta
RF36	Listar asiento contable	Media
RF37	Mostrar vista formulario de asiento contable	Baja
RF38	Buscar asiento contable	Media
RF39	Adicionar apunte contable	Alta
RF40	Modificar apunte contable	Alta
RF41	Eliminar apunte contable	Alta
RF42	Listar apunte contable	Media
RF43	Mostrar vista formulario de apunte contable	Baja
RF44	Buscar apunte contable	Media
RF45	Cerrar ejercicio fiscal	Alta
Informes		
RF46	Calcular mayor	Alta
RF47	Calcular submayor	Alta

Tabla 2. Requisitos funcionales.

A partir de la identificación de requisitos se llegó al siguiente resultado:

Requisitos con prioridad baja	7
Requisitos con prioridad media	15
Requisitos con prioridad alta	25

Tabla 3. Prioridad de requisitos.

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). A continuación en la Tabla 4, se enuncian los requisitos no funcionales identificados.

No	**Requisitos no funcionales**
Confiabilidad	
RNF1	La información manejada en el sistema debe de estar protegida de acceso no autorizado.
Usabilidad	
RNF2	Las funcionalidades y diseño del sistema deben ser intuitivas de forma tal que los usuarios puedan interactuar con el sistema sin necesidad de poseer conocimientos básicos de informática. El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final.
Rendimiento	
RNF3	Los tiempos de respuesta del sistema deben ser rápidos, no mayores de 5 segundos para cualquier operación realizada por el usuario.
Soporte	
RNF4	Se debe brindar al usuario la facilidad de instalación.
Mantenimiento	
RNF4	El sistema debe ser fácil de mantener después de desarrollado.

Software	
RNF5	Odoo 8.0 Apache 2.2.9 en adelante PostgreSQL 8.3 en adelante Python 2.5 en adelante Mozilla Firefox 4.5 en adelante
Hardware	
RNF6	512MB RAM mínimo Sistemas operativos compatibles: Linux , Windows

Tabla 4. Requisitos no funcionales.

2.2.4 Diagrama de caso de uso del sistema

Los diagramas de casos de uso del sistema documentan el comportamiento de un sistema desde el punto de vista del usuario. Es una representación gráfica de parte o el total de los actores y casos de uso del sistema, incluyendo sus interacciones, es decir, describe lo que hace un sistema desde el punto de vista de un observador externo. Estos diagramas muestran los distintos requisitos funcionales que se esperan de una aplicación o sistema y cómo se relaciona con su entorno (usuarios u otras aplicaciones). (Diagramas de Caso de Uso. Tello, Jesús Cáceres. Ma, 2013).

Un caso de uso es una descripción narrativa de un proceso de dominio. En la Figura 14. Diagrama de caso de uso del se muestra el diagrama de caso de uso del sistema, en el cual el contador es el actor principal, el cual es el que inicia todos los casos de uso del sistema, a excepción de Gestionar apuntes contables, el cual lo inicia el caso de uso del sistema Gestionar asientos contables:

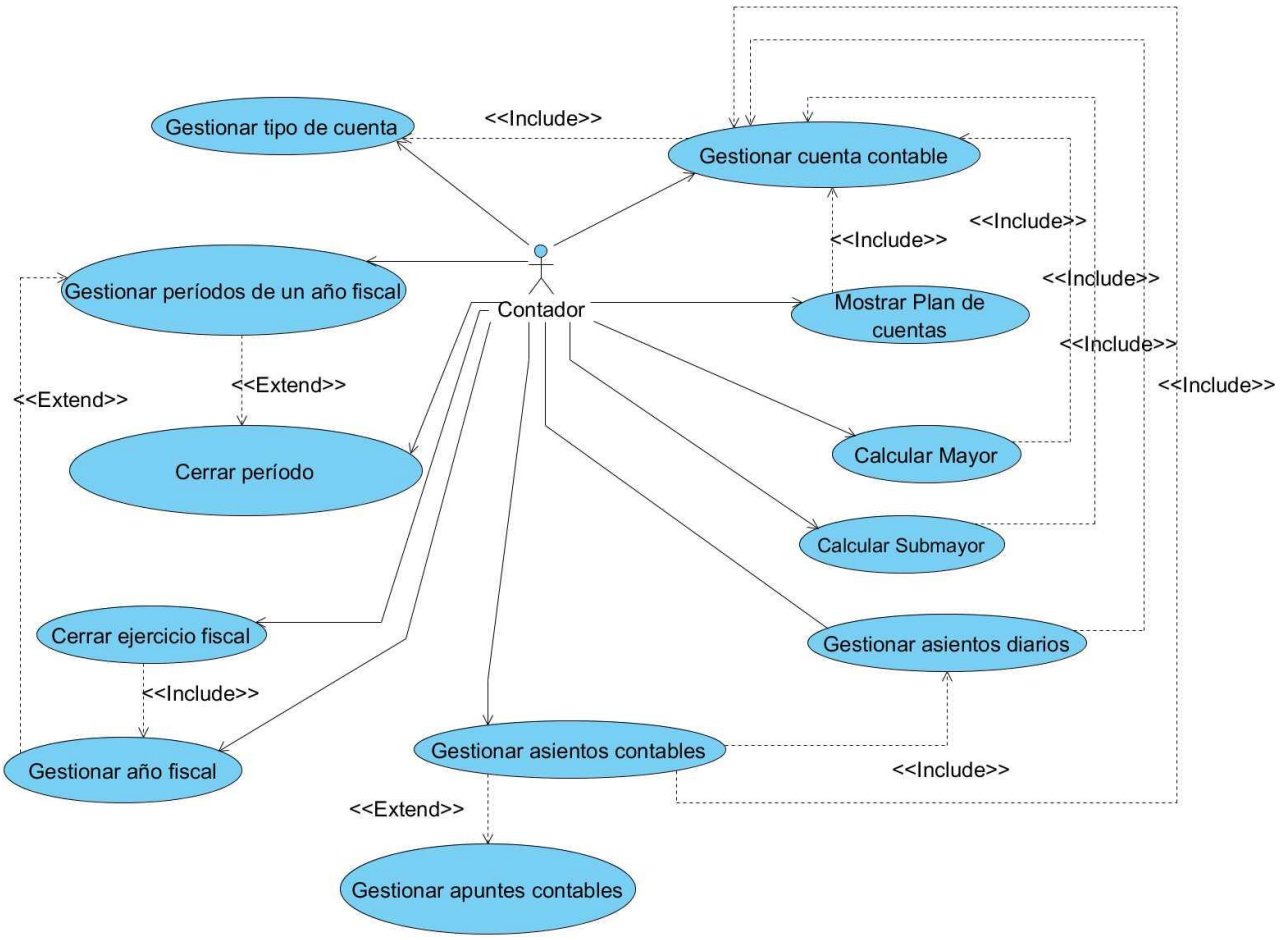


Figura 14. Diagrama de caso de uso del sistema. Elaboración propia.

La Figura 15 muestra la especificación del caso de uso de Calcular Mayor, ver en anexo las otras especificaciones de casos de uso:

Objetivo	Mostrar el Libro Mayor	
Actores	Contador	
Resumen	Mostrar el informe del Libro Mayor	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	Deben haberse configurado los ejercicios fiscales, períodos contables, cuentas contables y monedas	
Pos condiciones	Se mostró el informe del Libro Mayor	
Flujo de eventos		
Calcular mayor		
	Actor	**Sistema**
1	Selecciona la opción Libro Mayor	1.1 El sistema muestra una interfaz con los campos a llenar para calcular el mayor
2	Llena los campos correspondientes y selecciona la opción imprimir	2.1 El sistema muestra el informe relacionado con el cálculo del mayor
3		3.1 Termina el caso de uso
Flujos alternos		
1 No se introdujeron campos obligatorios		
	Actor	**Sistema**
1.	No introduce el período contable	1.1 Muestra un cartel afirmando la presencia del campo inválido Período
2.	La fecha de inicio es mayor que la fecha fin	2.1 Muestra un cartel afirmando la presencia de rangos inválidos
3	El período contable no coincide con las fechas	3.1 Muestra un cartel afirmando la presencia de rangos inválidos
4	No introduce las cuentas contables	4.1 Muestra un cartel afirmando la presencia del campo inválido Cuentas

Figura 15. Especificación del caso de uso Calcular Mayor. Elaboración propia.

2.2.5 Técnicas de validación de requisitos

Revisión de requisitos

Las revisiones de requisitos consisten en una o varias reuniones planificadas, donde se intenta confirmar que los requisitos poseen los atributos de calidad deseados. Se utilizó esta técnica al comprobar con la analista del proyecto Cedrux la relación de los requisitos de Cedrux y Odoo y la definición de los requisitos a personalizar.

Prototipado

El prototipado consiste en la creación de una maqueta o versión del producto final. Los objetivos de los prototipos varían en función de la disciplina. En el caso de la actividad de requisitos, los prototipos se utilizan, fundamentalmente, para comprobar la corrección y completitud de la

especificación de requisitos. La Figura 16 muestra el prototipado de caso de uso Calcular Mayor, en el cual se definieron las funcionalidades presentes en la Resolución No. 14/2007. Registros, Submayores, Mayores y Comprobante de operaciones del Manual de Normas de Control Interno.

A continuación, se muestra un ejemplo de prototipado para el requisito Calcular Mayor, ver en anexo los otros prototipos:

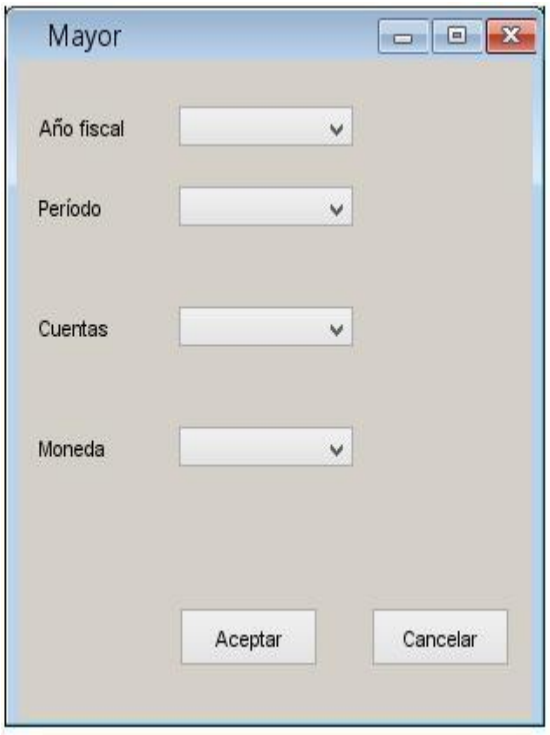


Figura 16. Prototipo del caso de uso Calcular Mayor. Elaboración propia.

2.3 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).

2.3.1 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).

Modelo: Objetos Python cuyos datos son almacenados en una base de datos PostgreSQL. Los modelos se encuentran dentro de la carpeta models, divididos en tres componentes: configuración, contabilidad 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 tras carpetas: configuración, contabilidad 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.

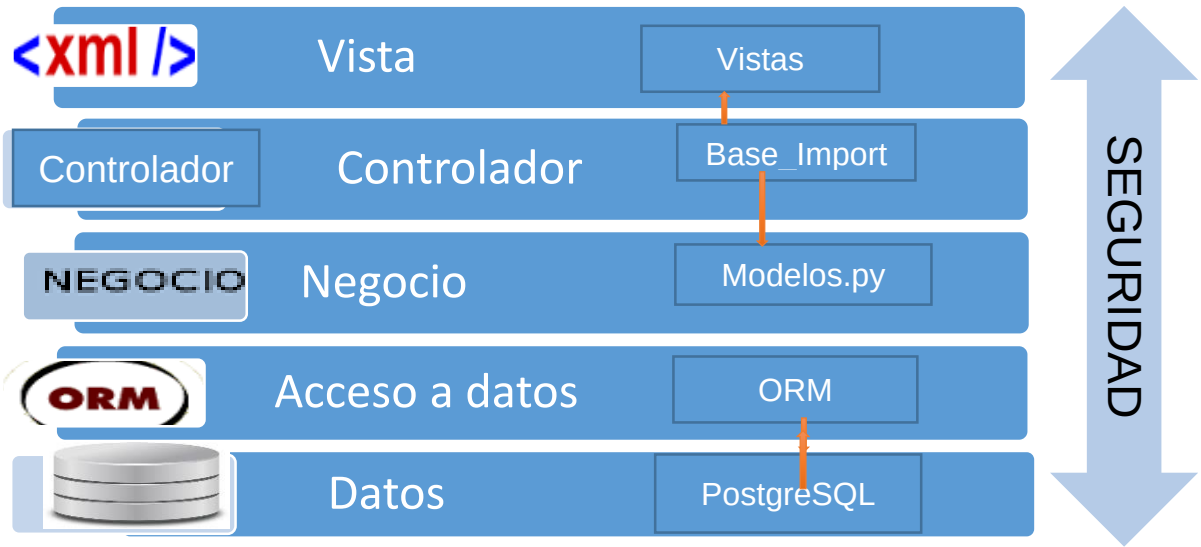


Figura 17. MVC. Elaboración propia.

Seguridad: La seguridad consiste en asegurar la confidencialidad, integridad y disponibilidad de un sistema. En Odoo la seguridad se extiende tanto a las vistas, como a los modelos, como al controlador. La misma se logra a través de los permisos que se le pueden otorgar a los usuarios en el sistema. La Figura 18 muestra la interfaz de usuario sobre la creación del grupo Acceso a cuentas. En ella se evidencia como solamente el usuario mostrado tiene acceso sobre las cuentas contables del sistema.

A continuación, se explican las pestañas mostradas en la Figura 18:

Usuarios: Personas que conforman el grupo creado.

Heredado: Los usuarios asociados a este grupo automáticamente se asocian al grupo ya creado.

Menús: Permiten seleccionar los menús a los que va a tener acceso el grupo creado.

Vistas: Permiten seleccionar las vistas a los que va a tener acceso el grupo creado.

Permiso de acceso: Permite otorgarle al grupo diferentes tipos de permisos, dentro de los que se encuentra lectura, escritura, creación y eliminación.

Grupo compartición: Grupo creado para establecer derechos de acceso para compartir información con algunos usuarios.

Portal: Si está marcado este grupo tiene acceso desde el portal de la aplicación.

Reglas: Definición de reglas que permiten acceso de lectura, escritura, creación y eliminación.

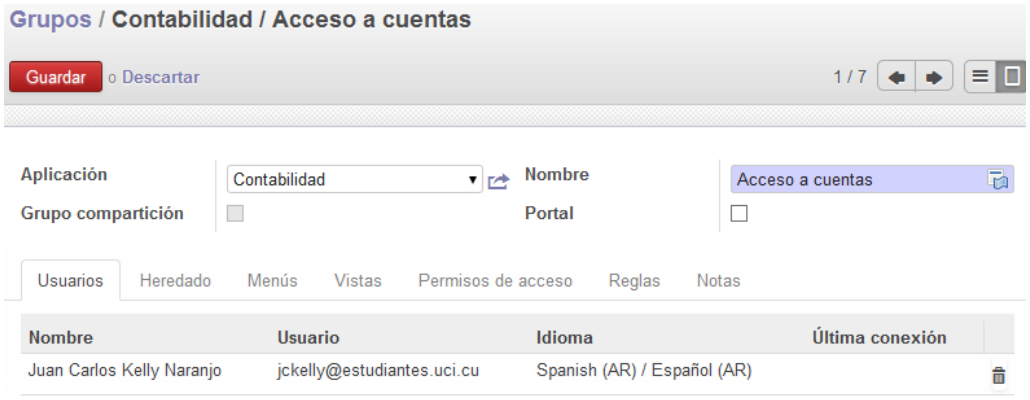


Figura 18. Seguridad en el módulo de Contabilidad de Odoo (Odoo, 2016).

2.3.2 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 DER fue propuesto originalmente por Peter Chen pare el diseño de sistemas de base de datos relacionales. Se identifica un conjunto de componentes primarios para el DER: objeto de datos, atributos, relaciones y varios indicadores de tipo. El propósito fundamental es representar objetos de datos y sus relaciones (Pressman, 2002). La Figura 19 muestra el modelo de datos del sistema, el cual está compuesto por 10 entidades, las cuales poseen atributos y relaciones entre sí, siendo la entidad cuenta la más importante debido a su alta responsabilidad en el sistema.

Figura 19.Modelo de datos. Elaboración propia.

2.3.3 Modelo del diseño

El Diagrama de Clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y relaciones entre ellos (Pressman, 2002).

A continuación, se muestra el diagrama de clases del diseño del caso de uso Calcular Mayor, el cual está compuesto por el modelo Informe.py, que contiene las clases libro_mayor_base y herencia_libro mayor. La vista y el controlador están presentes en la carpeta view y base_import respectivamente. Ver en anexo los otros diagramas de clases.

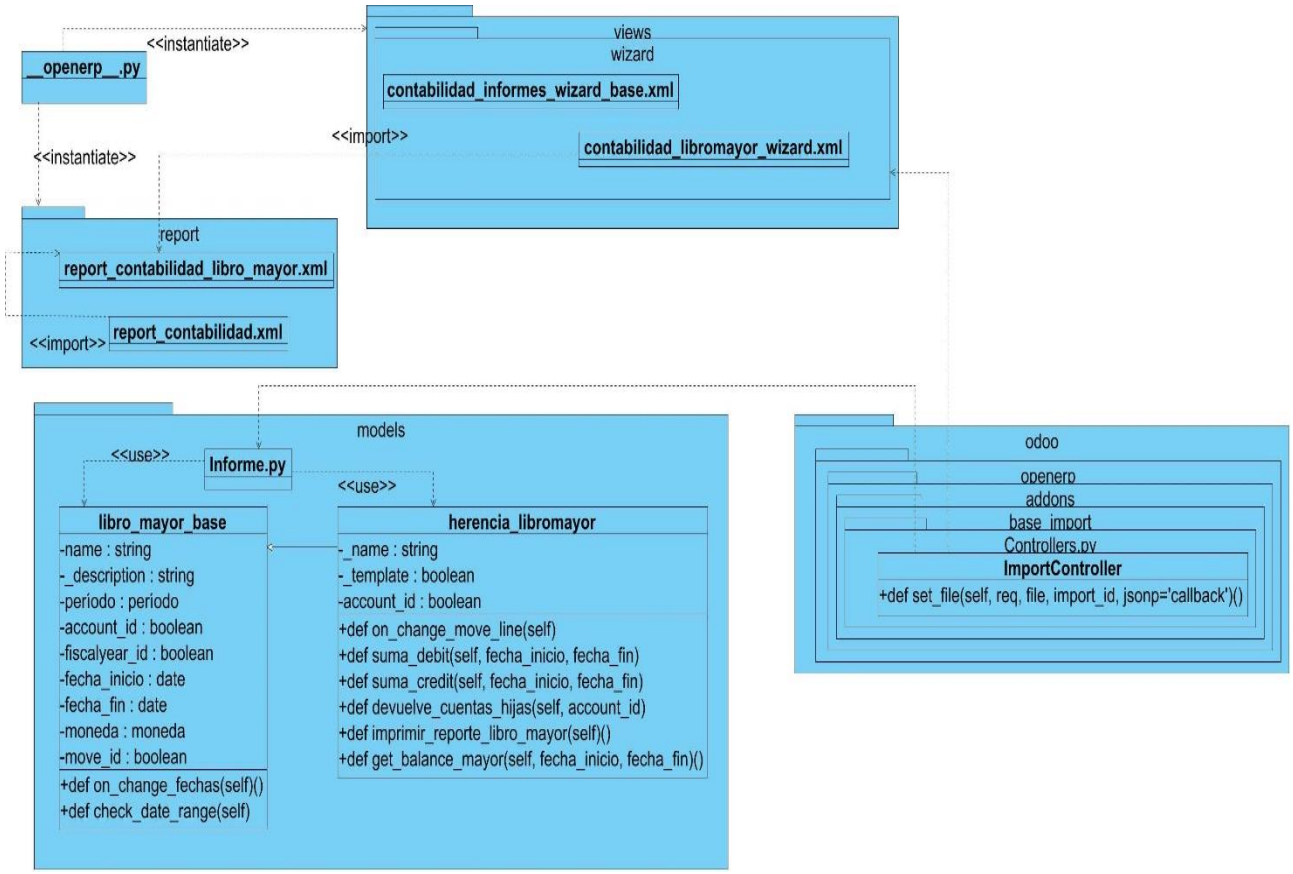


Figura 20. Diagrama de clases del diseño del caso de uso del sistema Calcular Mayor. Elaboración propia.

2.3.3 Patrones del diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. Un patrón de diseño identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades, además que ayuda a construir clases y a estructurar sistemas de clases. (Larman, 2003)

Patrones GRASP

Los patrones GRASP (Patrones Generales de Software para la Asignación de Responsabilidades) describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades, expresados como patrones. El nombre se eligió para sugerir la importancia de captar estos principios para diseñar con éxito el software orientado a objetos. Es necesario elegir cuidadosamente las clases adecuadas y decidir cómo estas deben interactuar, por esta razón resulta de vital importancia el uso de 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). En la figura 22 se muestra como la clase account_fiscalyear es la encargada de crear un año fiscal, ya que es la que posee la información necesaria para esto.

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.

En la Figura 22 se muestra como la clase account_fiscalyear es la encargada tanto de crear instancias de períodos mensuales, como de períodos trimestrales.

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, sistemas (Larman, 2003). En la Figura 21 se aprecia el bajo acoplamiento existente en el sistema, ya que el diseño muestra como las clases del componente models no depende de ningún otro componente.

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 21 se muestra como el modelo Informe es el que posee la responsabilidad de crear los informes de contabilidad (libro_mayor_base y herencia_libro_mayor). La clase

herencia_libro_mayor es la responsable tanto de imprimir un libro mayor, como de sumar los débitos y los créditos.

Controlador: Asignar la responsabilidad de controlar el flujo de eventos del sistema (Larman, 2003). En la Figura 21 se muestra como la clase ImportController 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:

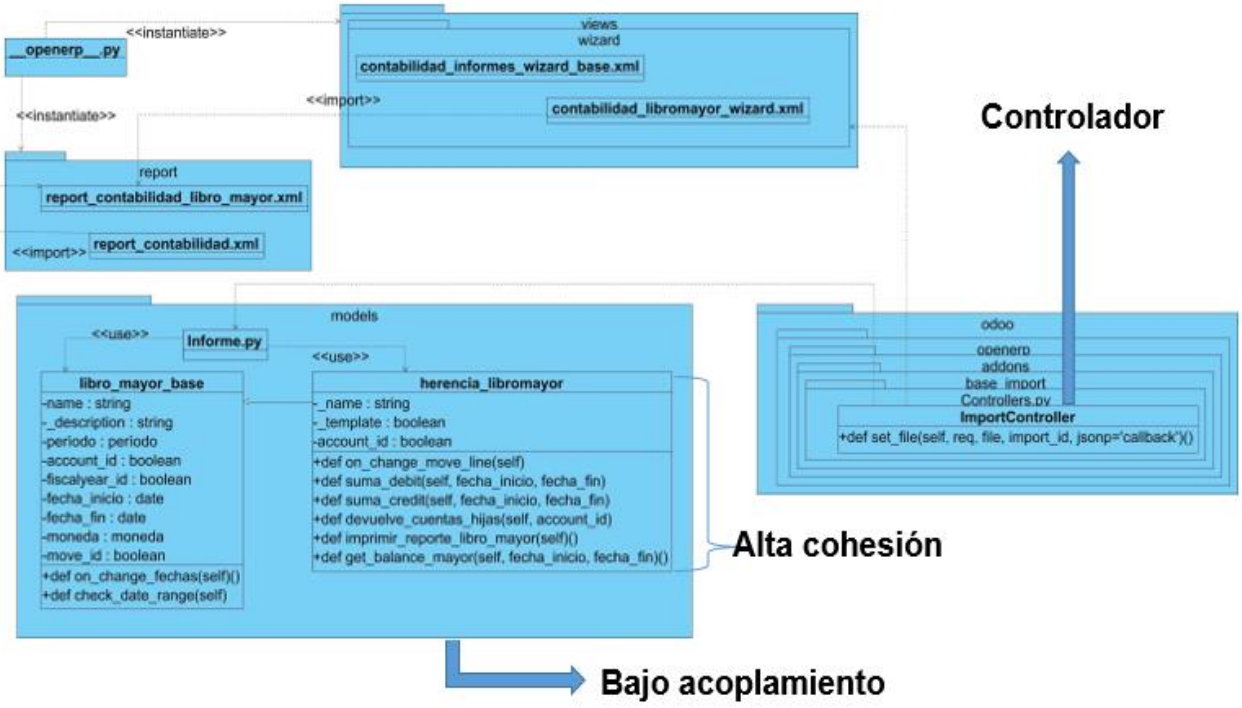


Figura 21. Patrones GRASP.Elaboración propia.

El siguiente diagrama de clases muestra la presencia de los patrones experto y creador:

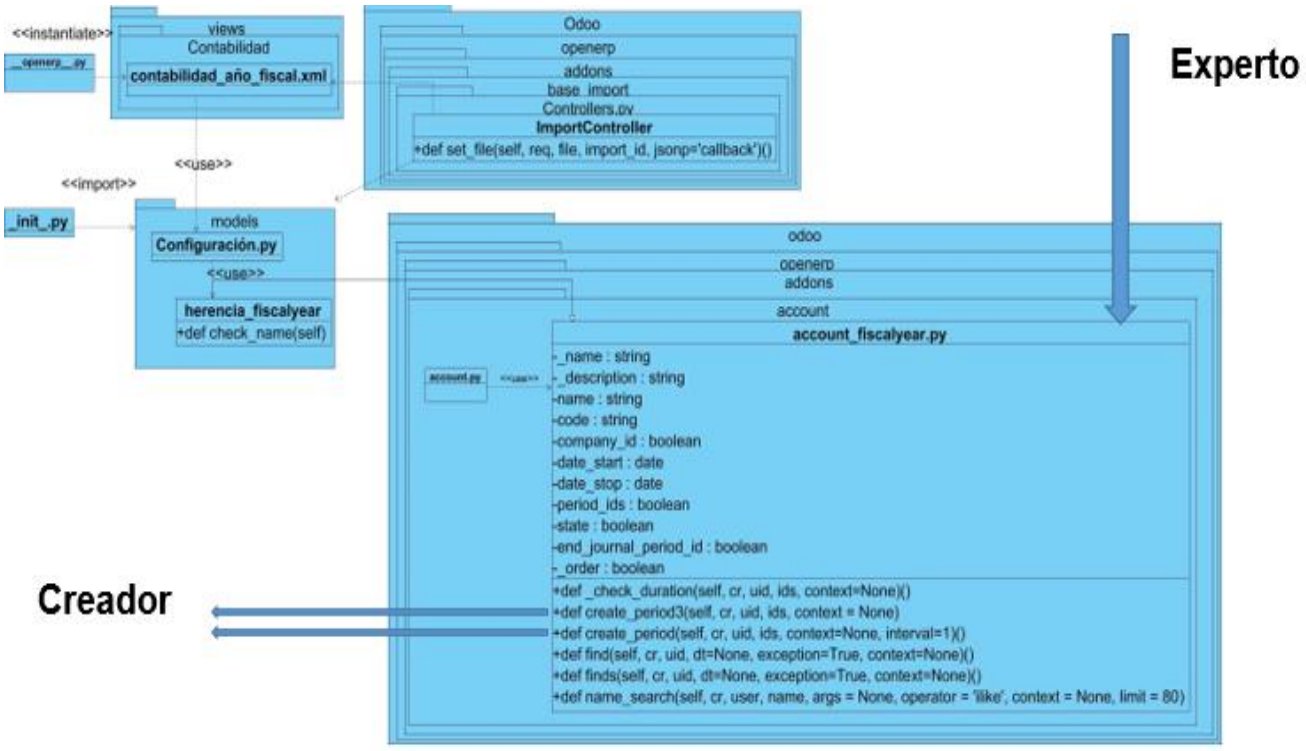


Figura 22. Patrones GRASP.Elaboración propia.

Patrones GOF aplicados:

Decorador: Añade responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización cuando se trata de añadir funcionalidades (Larman, 2003).

En la Figura 23, se muestra la clase libro_mayor_base, en donde se evidencia el patrón decorador a través de los métodos señalados.

El método def_on_change_fechas decora el método onchange, o sea implementa todos los métodos de esta clase y además adiciona la responsabilidad de dar valores a la fecha de inicio y fin en dependencia del período escogido.

El método def_check_date_range decora el método constrains, o sea implementa todos los métodos de esta clase y además adiciona la responsabilidad de validar que el rango de fechas esté dentro del período escogido.

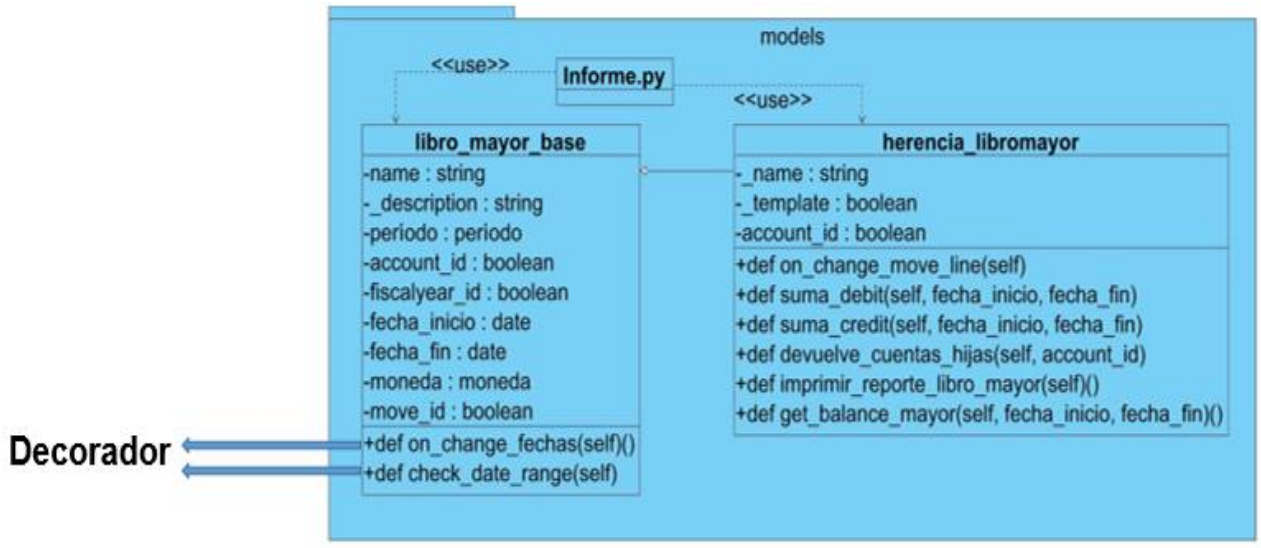


Figura 23. Patrones GOF.

2.3.5 Validación del diseño

Para la validación del diseño se utilizaron las métricas Profundidad de árbol de herencia, Número de métodos heredados y Acoplamiento entre objetos. A continuación, se describe cómo son aplicados en la propuesta de solución.

Conjunto de métricas CK

Chidamber y Kemerer establecen 6 métricas basadas en clases para medir cinco atributos básicos en el diseño orientado a objetos: acoplamiento, complejidad de una clase, reutilización, cohesión y herencia.

En el presente trabajo se utilizará la métrica relacionada con la herencia, debido a la alta utilización de herencia para desarrollar la personalización del módulo de contabilidad de Odoo. A continuación, se explica la métrica Profundidad del árbol de herencia (Depth of Inheritance Tree -DIT) propuesta por Chidamber y Kemerer en 1994.

Profundidad del árbol de herencia (Depth of Inheritancetree-DIT)

Es la distancia desde una clase, a la clase raíz del árbol de herencia. Si la clase se encuentra en situación de herencia múltiple, el DIT será la longitud máxima hasta la raíz (Kemerer, 1994). Chidamber y Kemerer proponen esta métrica como medida de la complejidad de una clase, complejidad del diseño y el potencial de rehúso. Esto se debe a que las clases que estén más profundas en el árbol de herencia heredarán más métodos. Por lo tanto, mientras mayor sea el valor de DIT las clases serán más complejas de desarrollar y de mantener, la jerarquía será más profunda haciendo el diseño más trabajoso y la probabilidad de detección de fallos será mayor. Por otra parte,

indica que se pueden reutilizar muchos métodos. Es deseado obtener un número bajo de DIT (Kemerer, 1994). Presman menciona como límite para los niveles de herencia el número 6.
En la Figura 24 se muestran los niveles de herencia de todas las clases del sistema:

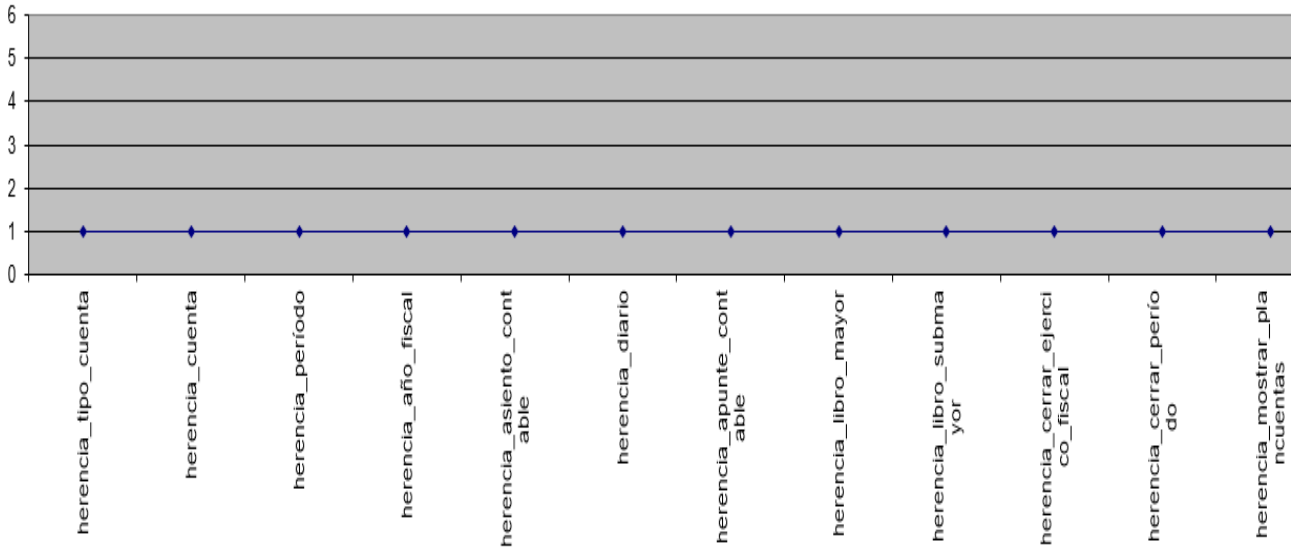


Figura 24. Niveles de herencia de las clases.

Como resultado, se obtuvo la presencia en todas las clases de un nivel 1 de profundidad de herencia, por lo que se puede afirmar que la complejidad del diseño es baja y es fácil predecir el comportamiento de las clases, lo que trae como resultado la detección oportuna de fallos en las clases, ya que no es necesario recurrir a niveles de herencia profundos. Esto trae consigo también poca dependencia entre las clases heredadas, traducido en la existencia de un bajo acoplamiento en la herencia.

Acoplamiento entre objetos (Coupling Between Objects –CBO-)

Acoplamiento es el uso de métodos o atributos definidos en una clase que son usados por otra. Las clases tienen que interactuar entre ellas para formar sistemas, y esta interacción puede indicar su complejidad (Mark Lorenz, 1994).

CBO

CBO de una clase es el número de clases a las cuales una clase está ligada. Existe dependencia entre dos clases cuando una clase usa métodos o variables de la otra clase. Las clases relacionadas por herencia no se tienen en cuenta (Mark Lorenz, 1994).

La Figura 25 muestra la cantidad de relaciones de las clases del sistema:

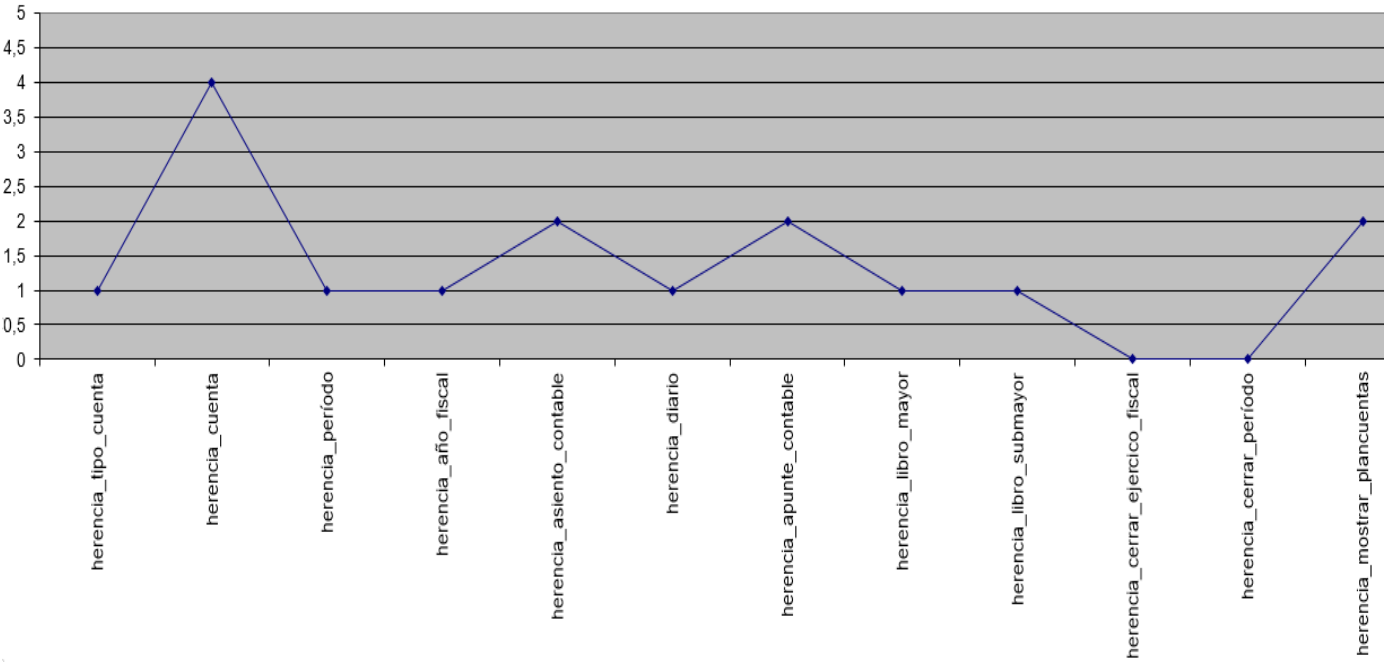


Figura 25. Cantidad de relaciones entre clases.

A continuación, se expresa las relaciones de las clases evidenciadas en la gráfica anterior:

Clases	Clases con las que se relaciona	Cantidad
herencia_tipo_cuenta	herencia_cuenta	1
herencia_cuenta	herencia_tipo_cuenta, herencia_mostrar_plancuenta herencia_apunte_contable herencia_asiento_contable	4
herencia_período	herencia_año_fiscal	1
herencia_año_fiscal	herencia_período	1
herencia_asiento_contable	herencia_diario, herencia_apunte_contable	2
herencia_diario	herencia_asiento_contable	1
herencia_apunte_contable	herencia_cuenta herencia_asiento_contable	2
herencia_libro_mayor	herencia_cuenta	1

herencia_libro_submayor	herencia_cuenta	1
herencia_cerrar_ejercico_fiscal	herencia_año_fiscal	1
herencia_cerrar_período	herencia_período	1
herencia_mostar_plancuenta	herencia_año_fiscal herencia_cuenta	2

Tabla 5. Cantidad de clases relacionadas.

A partir del resultado anterior, queda evidenciado la presencia de pocas relaciones entre clases, lo cual es un indicador de poca dependencia entre las clases, bajo acoplamiento, alta cohesión y buena reutilización.

Conjunto de métricas LK

Lorenz y Kidd separan las métricas basadas en clases en tres grupos:

- Métricas de tamaño de la clase.
- Métricas de herencia.
- Métricas de las características internas de las clases.

Estas métricas están enfocadas a las características internas del diseño orientado a objeto y de esta manera, contribuyen a asegurar la mantenibilidad de los productos de software (Mark Lorenz, 1994).

En el presente trabajo se utilizará la métrica relacionada con la herencia, debido a la alta utilización de herencia para desarrollar la personalización del módulo de contabilidad de Odoo. A continuación, se explica la métrica Número de Métodos Heredados (Number of inherited methods-NMI).

Número de métodos heredados (NM)

Es el número de métodos que hereda una subclase. Es deseado un número alto de NMI ya que indica la fuerza de la subclase en la especialización. También mide la calidad del uso de la herencia (Mark Lorenz, 1994).

La Figura 26 muestra el número de métodos heredados por cada clase:

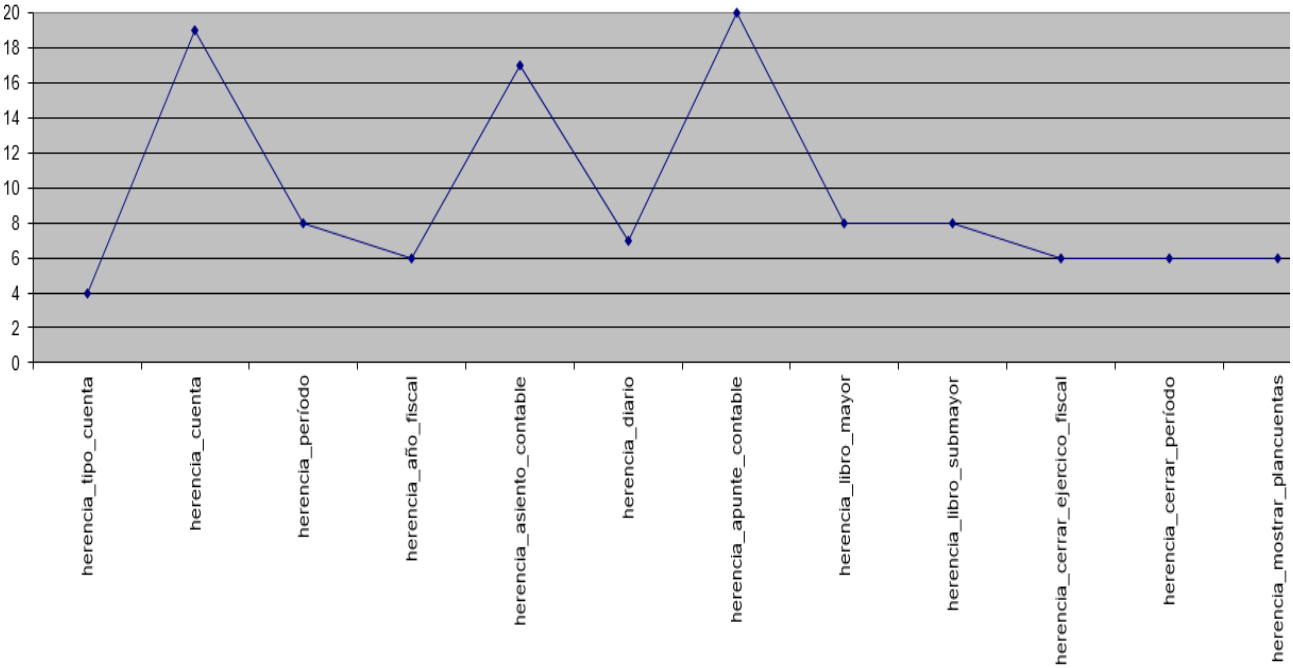


Figura 26. Número de métodos heredados por cada clase.

Clases	Número de métodos heredados
herencia_tipo_cuenta	4
herencia_cuenta	19
herencia_período	8
herencia_año_fiscal	6
herencia_asiento_contable	17
herencia_diario	7
herencia_apunte_contable	20
herencia_libro_mayor	8
herencia_libro_submayor	8
herencia_cerrar_ejercico_fiscal	6
herencia_cerrar_período	6
herencia_mostar_plancuenta	6

Tabla 6. Número de métodos heredados.

Como se puede observar, el número de métodos heredados por cada clase es elevado, por lo que se puede afirmar el uso de una buena herencia y de una fuerte especialización en las clases.

2.3.6 Conclusiones parciales

A partir de lo expuesto en el capítulo se llegó a la siguiente conclusión:

- Mediante la creación del modelo conceptual se seleccionaron los conceptos principales del negocio y se definió las relaciones entre ellos.
- La elaboración de especificaciones de casos de uso del sistema permitió el diseño de los prototipos de las funcionalidades del módulo de Contabilidad de Odoo.
- El uso de patrones de diseño contribuyó a asignar correctamente las responsabilidades a las clases.
- Para la validación del diseño se utilizaron las métricas Profundidad del árbol de herencia**,** Número de métodos heredados y Acoplamiento entre objetos, lo que permitió determinar la presencia de una buena herencia y de poca dependencia entre las clases.

CAPÍTULO 3: IMPLEMENTACIÓN 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 Disciplina Implementación

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

3.1.1 Estándares de codificación

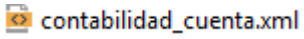
Los estándares de código son una forma de "normalizar" la programación de forma tal que, al trabajar en un proyecto, cualquier persona involucrada en el mismo tenga acceso y comprenda el código (Microsoft, 2016). A continuación, se presentan diferentes ejemplos de estilos de codificación definidos para la implementación de la aplicación:

Nombre de los archivos

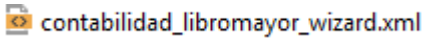
- reportes: report_módulo_nombre

```
<template id="report_contabilidad_libromayor">
```

- vista: módulo_nombre



- wizard: módulo_nombre_wizard



Clases de un modelo

- Si la clase es heredada: class herencia_nombre

```
class herencia_asientoscontables(models.Model)
```

- Si la clase es nueva: class módulo_nombre

```
class libro_mayor_base(models.Model)
```

Vistas

- vista Form: id = ¨módulo_modelo_form ¨

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

- vista Tree: id = ¨módulo_modelo_tree ¨

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

Menú

id = ¨menú_módulo_nombre¨

```
<menuitem id="menu_contabilidad_cuenta"/>
```

3.1.2 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). La Figura 29 muestra el diagrama de componentes del sistema, la organización y dependencia existente entre los componentes Vistas, Controlador, Base de Datos y Models, este último operando sobre la Base de Datos para la gestión de la información. El paquete Vistas está compuesto por 12 componentes agrupados en configuración, contabilidad e informes. El paquete Models está compuesto por 3 componentes que representan los modelos existentes en la arquitectura y el paquete Controlador representa la clase controladora del sistema.

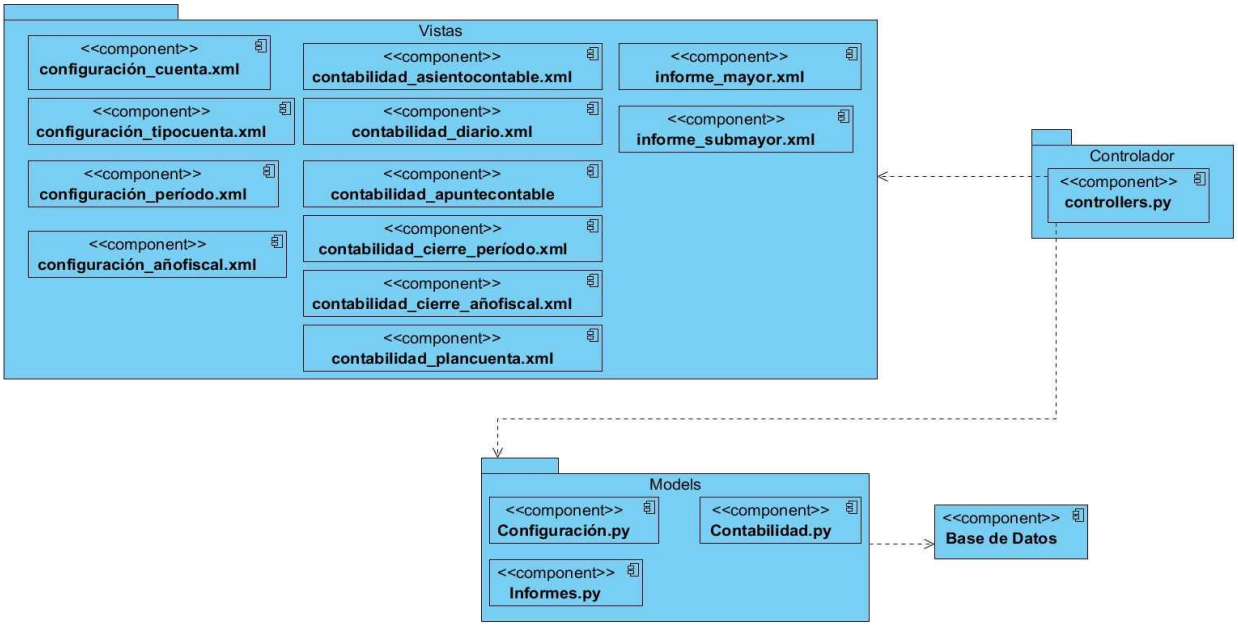


Figura 27. Diagrama de componentes. Elaboración propia.

3.2 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.

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 requerimientos definidos (Pressman, 2002).

3.2.1 Pruebas de caja blanca

La prueba de caja blanca, denominada a veces prueba de caja de cristal es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba. Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada módulo (Pressman, 2002).

Técnica del camino básico

Es una técnica de prueba de caja blanca propuesta inicialmente por Tom McCabe. El método del camino básico permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución (Pressman, 2002).

Complejidad ciclomática

La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Cuando se usa en el contexto del método de prueba del camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y nos da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez (Pressman, 2002).

La Figura 28 muestra el código de la funcionalidad get_balance_mayor (obtener balance de mayor) de la clase herencia _libro mayor, la cual constituye uno de los requisitos de mayor importancia para la construcción del sistema, debido a que calcula el balance total de todas las operaciones

realizadas sobre las cuentas en una determinada fecha, además constituye el requisito de mayor complejidad a implementar.

```
def get_balance_mayor(self, fecha_inicio, fecha_fin):
1   lista_cuentas_hijas = self.env['account.account'].search([('id', 'child_of', [self.account_id.id])])
    lista_cuentas_id = self.env['account.move.line'].search([('account_id', 'in', [cuenta.id for cuenta in lista_cuentas_hijas]),
2                       ('date_maturity', '>', fecha_inicio),('date_maturity', '<', fecha_fin),('currency_id', '=', self.moneda.id)
    list_vacia = []
3   if len(lista_cuentas_id)==0:
        list_vacia.insert(0,0)
        list_vacia.insert(1,0)
4       list_vacia.insert(2,0)
        list_vacia.insert(3,0)
9       return list_vacia
    else:
        aux = 0.00
5       lista_objetos = []
        lista_aux = []
        debit = 0.00
6       credit = 0.00
        total = 0.00
7       for datos in lista_cuentas_id:
            debit = debit + datos.debit
            credit = credit + datos.credit
            total = debit - credit
            cuentas_total = {
8               'name': datos.name,
                'partner': datos.partner_id.name,
                'date_maturity': datos.date_maturity,
                'debit': datos.debit,
                'credit': datos.credit,
                'total': aux + (datos.debit - datos.credit)}
            lista_objetos.append(cuentas_total)
            aux = aux + (datos.debit - datos.credit)

        lista_aux.insert(0, lista_objetos)
        lista_aux.insert(1, debit)
        lista_aux.insert(2, credit)
        lista_aux.insert(3, total)
9   return lista_aux
```

Figura 28. Método get_balance_mayor.

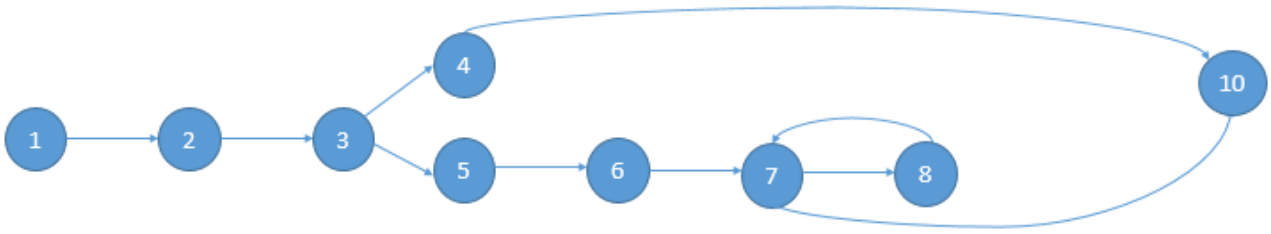


Figura 29. Técnica del camino básico. Elaboración propia.

V (G) = A – N + 2 V (G) = Complejidad ciclomática

= 11 – 10 +2
= 3
V (G) = P +1
= 2+1
= 3
V (G) = Regiones
3 = 3

A = Cantidad de aristas del grafo
N = Cantidad de nodos del grafo
P= Nodos predicados. Nodos que tienen más de una arista de salida

Regiones: Áreas delimitadas por nodos y aristas del grafo

Cantidad de caminos mínimos = 3

Camino 1: 1, 2, 3, 4,9

Camino 2: 1, 2, 3, 5, 6, 7, 8, 7,9

Camino 3: 1, 2, 3, 5, 6, 7, 9

Caso de prueba para el camino 2

Descripción	A partir de seleccionar datos, se comprueba si estos datos son válidos para adicionarlos posteriormente
Condición de ejecución	Si selecciona los campos período ,cuenta contable y fecha
Entrada	Período, cuenta contable , fecha de inicio y fecha fin
Resultado	Devuelve el balance del libro Mayor

Tabla 7. Diseño de caso de prueba para el camino 3.

Luego de haberle aplicado el método del camino básico a la funcionalidad anterior se comprobó que cada sentencia es ejecutada al menos una vez. El valor calculado como complejidad ciclomática definió el número de caminos independientes del conjunto básico, lo que facilitó el límite superior para el número de pruebas que se deben realizar.

3.2.2 Prueba de caja negra

Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software. O sea, la prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa (Pressman, 2002).

A continuación, se presenta la especificación de requisito del caso de uso Gestionar cuenta:

Objetivo	El contador debe ser capaz de realizar las acciones relacionadas con una cuenta contable.	
Actores	Contador	
Resumen	El caso de uso inicia cuando el contador selecciona la opción Cuenta/Cuentas contables	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	Debe haberse insertado un tipo de cuenta	
Pos condiciones	Se realizó alguna acción sobre una cuenta contable	
Flujo de eventos		
Flujo básico <Gestionar cuenta contable>		
	Actor	**Sistema**
1	Realiza alguna acción relacionada con una cuenta contable	
		-Adicionar cuenta contable. Ver Sección 1: Adicionar cuenta contable. -Modificar cuenta contable. Ver Sección 2: Modificar tipo de cuenta -Eliminar cuenta contable. Ver Sección 3: Eliminar cuenta contable 4-Listar cuenta contable: Ver Sección 4: Listar cuenta contable 5-Mostrar vista formularia de cuenta contable: Ver Sección 5: Mostrar vista de cuenta contable.

Sección 1: "Adicionar cuenta contable"		
Flujo básico Adicionar cuenta contable		
	Actor	**Sistema**
1.	Selecciona la opción Cuenta/Cuentas contables contable	1.1 Muestra una interfaz con el listado de todas las cuentas contables
2.	Selecciona la opción crear	2.1 Muestra una interfaz con los campos a llenar
3.	Llena los campos correspondientes y selecciona la opción guardar	3.1 Guarda una nueva cuenta contable
4.		4.1 Termina el caso de uso
Flujos alternos Sección 1		
1.1 No se introdujeron campos obligatorios		
	Actor	**Sistema**
1.	No introduce el código de la cuenta contable	1.1 Muestra un cartel afirmando la presencia del campo inválido Código
2.	No introduce el nombre de la cuenta	2.1 Muestra un cartel afirmando la presencia del campo inválido Nombre
3.	No introduce el tipo de cuenta	3.1 Muestra un cartel afirmando la presencia del campo inválido Tipo de cuenta
Sección 2 : "Modificar cuenta contable"		
Flujo básico Modificar cuenta contable		
	Actor	Sistema
1.	Selecciona la opción Cuenta/Cuentas contables	1.1 Muestra una interfaz con el listado de todas las cuentas contables
2.	Selecciona una cuenta contable	2.1 Muestra una interfaz con los campos de la cuenta contable seleccionada

3.	Selecciona la opción editar	3.1 Muestra una interfaz con los campos editables de la cuenta contable seleccionada
4.	Llena los campos con los nuevos datos y selecciona la opción guardar	4.1 Guarda una nueva cuenta contable
5		5.1 Termina el caso de uso
Flujos alternos. Volver a Flujos alternos Sección 1.1		
Sección 3 : "Eliminar cuenta contable"		
Flujo básico Eliminar Tipo de cuenta		
	Actor	Sistema
1.	Selecciona la opción Cuenta/ Cuentas contables	1.1 Muestra una interfaz con el listado de todas las cuentas contables
2.	Selecciona una cuenta contable	2.1 Muestra la información relacionada con la cuenta contable seleccionada
3.	Selecciona la opción Más	3.1 Muestra una lista con diferentes acciones a realizar sobre una cuenta contable
4.	Selecciona la opción Suprimir	4.1 Muestra un cartel preguntando si en realidad quiere eliminar este registro
5.	Selecciona la opción aceptar	5.1 Elimina la cuenta contable
6.		6.1 Termina el caso de uso

4.	Selecciona la opción Suprimir	4.1 Muestra un cartel preguntando si en realidad quiere eliminar este registro
5.	Selecciona la opción cancelar	5.1 Muestra la información del tipo de cuenta seleccionado
6.		6.1 Termina el caso de uso
Sección 4 : "Listar cuenta contable"		
Flujo básico <Listar cuenta contable>		
	Actor	Sistema
1.	Selecciona la opción Cuenta/ Cuentas contables	1.1 Muestra una interfaz con el listado de todas las cuentas contables
2.	Selecciona una cuenta contable	2.1 Muestra la información relacionada con la cuenta contable seleccionada
3.	Selecciona la opción Vista lista(arriba a la derecha)	3.1 Muestra la información de todas las cuentas contables en forma de Vista de listas
4.		4.1 Termina el caso de uso
Sección 5 : "Mostrar vista formulario de una cuenta contable"		
Flujo básico <Mostrar vista formulario de una cuenta contable>		
	Actor	Sistema
1.	Selecciona la opción Cuenta/ Cuentas contables	1.1 Muestra una interfaz con el listado de todas las cuentas contables
2.	Selecciona la opción Vista formulario(arriba a la derecha)	2.1 Muestra la información de las cuentas contables en forma de Vista de formularios
3.		3.1 Termina el caso de uso
Sección 5 : "Buscar cuenta contable"		
Flujo básico <Buscar tipo de cuenta>		
	Actor	Sistema
1.	Selecciona la opción cuenta contable	1.1 Muestra una interfaz con los códigos, nombres, debe, crédito, tipo interno y moneda de todas las cuentas contables

2.	Escribe el nombre o código de la cuenta contable.	2.1 Muestra la cuenta contable que coincide con el nombre o el código
3.		3.1 Termina el caso de uso

Figura 30. Caso de prueba del caso de uso del sistema Gestionar tipo de cuenta. Elaboración propia.

La técnica de partición de equivalencia fue aplicada por los especialistas del Grupo de Calidad del centro CEIGE.

En la Tabla 5 se muestran la cantidad de no conformidades encontradas en las 2 iteraciones, las cuales fueron resueltas, obteniéndose así el Acta de liberación del producto.

Criterios	Iteración 1	Iteración 2
No Conformidades(NC)	14	0
Significativas	10	0
No significativas	1	0
Recomendaciones	1	0
Ortografía	1	0
Excepciones	1	0
Interfaz	1	0

Tabla 8. Iteración 1 y 2.

3.2.3 Pruebas de aceptación

Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido. En esta prueba se evalúa el grado de calidad del software con relación a todos los aspectos relevantes para que el uso del producto se justifique (Software, 2016). En la Tabla 6 se muestra la cantidad de no conformidades encontradas en las dos iteraciones, las cuales fueron resueltas, obteniéndose así la carta de aceptación del cliente.

Criterio	Iteración 1	Iteración 2
No Conformidades(NC)	10	0

Significativas	8	0
No significativas	0	0
Recomendaciones	5	0
Ortografía	0	0
Excepciones	1	0
Interfaz	1	0

Tabla 9. Iteración 1 y 2. Pruebas de aceptación.

3.3 Validación de la investigación

La validación del objetivo de la investigación se realizó con la técnica de IADOV, ya que permite conocer el grado de satisfacción de los clientes.

3.3.1 Técnica de IADOV

Para conseguir la validación del presente trabajo, se utilizó la Técnica de IADOV, la cual constituye una vía para evaluar el grado de satisfacción de procesos y productos. Está conformada por cinco preguntas, tres cerradas y dos abiertas. Las preguntas cerradas se relacionan a través de lo que se denomina el “Cuadro Lógico de IADOV” (Tabla 10), por otra parte las preguntas abiertas permiten profundizar en la naturaleza de las causas que originan los diferentes niveles de satisfacción (Vergara, 2016). En correspondencia con esto, se seleccionaron un total de 5 clientes del proyecto CEIGE, los cuales se muestran en la tabla 2. A estos se les aplicó la encuesta para determinar el grado de satisfacción con la propuesta del módulo de Contabilidad.

Nombre y apellidos
Yunior Feria Chapman
Annia Boza Verdecia
Ernesto Mató Roque
Erich Mario Gómez
Omar Sablón Mirabal

Tabla 10. Clientes.

El número resultante de la interrelación de las tres preguntas nos indica la posición de cada sujeto en la escala de satisfacción, o sea su satisfacción individual. La escala de satisfacción utilizada es la siguiente:

1. Clara satisfacción
2. Más satisfecho que insatisfecho
3. No definida
4. Más insatisfecho que satisfecho
5. Clara insatisfacción
6. Contradictoria

Esta técnica también permite obtener el índice de satisfacción grupal (ISG), para lo cual se trabaja con los diferentes niveles de satisfacción que se expresan en la escala numérica que oscila entre +1 y - 1 de la siguiente forma:

Escala	Resultado
+1	Máximo de satisfacción
0,5	Más satisfecho que insatisfecho
0	No definido y contradictorio
- 0,5	Más insatisfecho que satisfecho
-1	Máxima insatisfacción

Figura 31. Niveles de satisfacción. Elaboración propia.

La satisfacción grupal se calcula por la siguiente fórmula:

$$ISG = \frac{A\ (+\ 1) + B\ (+\ 0{,}5) + C\ (0) + D\ (-\ 0{,}5) + E\ (-\ 1)}{N}$$

Figura 32. Cálculo del índice de satisfacción grupal. Elaboración propia.

En esta fórmula A, B, C, D, E, representan el número de sujetos con índice individual 1; 2; 3 ó 6; 4; 5 y donde N representa el número total de sujetos del grupo.

El índice grupal arroja valores entre + 1 y - 1. Los valores que se encuentran comprendidos entre - 1 y - 0,5 indican insatisfacción; los comprendidos entre - 0,49 y + 0,49 evidencian contradicción y los que caen entre 0,5 y 1 indican que existe satisfacción.

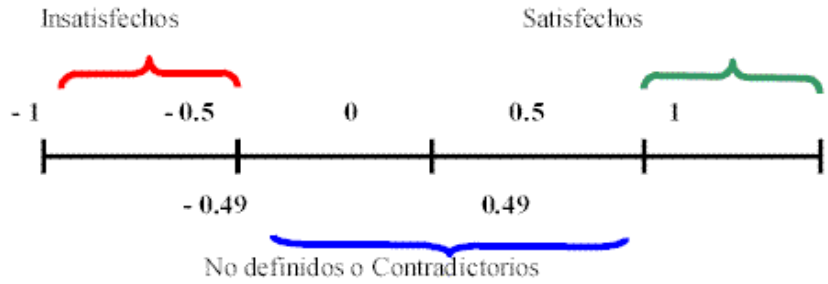


Figura 33. Rango de valores para el cálculo de satisfacción grupal. Elaboración propia.

Cuadro Lógico de IADOV

3-Luego de haber interactuado con el módulo de Contabilidad de Odoo, refleje en qué medida se siente satisfecho con el resultado que obtuvo al gestionar los procesos contables de Cuba en el módulo de Contabilidad de Odoo.	1- ¿Considera usted que es posible continuar gestionando los procesos contables de nuestro país con la base tecnológica con que cuenta actualmente Cedrux?								
	No			**No Sé**			**Sí**		
	2- ¿Cree usted que con la implantación del módulo de Contabilidad de Odoo se gestionarán con mayor eficiencia los procesos contables en Cuba?								
	Sí	**No Sé**	**No**	**Sí**	**No Sé**	**No**	**Sí**	**No Sé**	**No**
Me gusta mucho	**1**	**2**	**6**	**2**	**2**	**6**	**6**	**6**	**6**
No me gusta tanto	**2**	**2**	**3**	**2**	**3**	**3**	**6**	**3**	**6**
Me da lo mismo	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
Me disgusta más de lo que me gusta	**6**	**3**	**6**	**3**	**4**	**4**	**3**	**3**	**4**
No me gusta nada	**6**	**6**	**6**	**6**	**4**	**4**	**6**	**6**	**5**
No Sé qué decir	**2**	**3**	**6**	**3**	**3**	**3**	**6**	**6**	**4**

Tabla 11. Cuadro Lógico de IADOV.

A continuación, se muestran los resultados obtenidos de la aplicación de la encuesta.

Total de encuestados(N)	**5**	**Escala**
Máxima satisfacción	**5**	**A**
Más satisfecho que insatisfecho	**0**	**B**
No definida	**0**	**C**
Más insatisfecho que satisfecho	**0**	**D**
Clara insatisfacción	**0**	**E**
Contradictoria	**0**	**F**

Tabla 12. Resultados de la escala de satisfacción.

Resultados obtenidos

Cálculo de Índice de Satisfacción Grupal

$$ISG = \frac{5(+1)+0(0.5)+0(0)+0(-0.5)+0(-1)}{5}$$

ISG = 1

El valor obtenido del cálculo del ISG fue 1, lo cual indica que, según el rango de valores de satisfacción grupal, el máximo de satisfacción de los encuestados respecto a la personalización del módulo de Contabilidad de Odoo. Las preguntas abiertas contribuyeron al mejoramiento del módulo de Contabilidad para futuras versiones.

Con este resultado queda cumplido el objetivo de la investigación, ya que con todos los clientes encuestados se alcanzó el máximo de satisfacción respecto a la personalización del módulo de Contabilidad de Odoo, por lo que se puede afirmar que con la solución propuesta mejorará la gestión de los procesos contables en Cuba.

3.4 Conclusiones parciales

- La ejecución de las pruebas de caja blanca, realizadas a través de la técnica del camino básico, permitieron obtener una medida lógica del diseño procedimental.
- Las ejecuciones de las pruebas de caja negra permitieron garantizar el correcto funcionamiento del módulo de Contabilidad.
- Las pruebas de aceptación permitieron satisfacer las necesidades del cliente, entregando así un producto libre de errores.
- Con la validación de la solución se determinó que la propuesta de solución contribuye a una mejor gestión de los procesos contables en Cuba.

Conclusiones generales

- El estudio del estado del arte determinó que el módulo de Contabilidad de Odoo no cumple lo establecido en las normas contables cubanas quedando demostrado la necesidad de desarrollar una personalización del mismo.
- El diseño del módulo de contabilidad de Odoo en función de las normas contables cubanas permitió obtener una vista de la estructura del sistema teniendo en cuenta patrones para la asignación de responsabilidades.
- La fase de implementación permitió desarrollar el módulo de Contabilidad de Odoo en función de las normas contables cubanas.
- La fase de pruebas de software permitieron garantizar la verificación y validación del sistema, garantizando así la obtención de un sistema con mayor calidad.

Recomendaciones

Se recomienda:

Para futuras versiones añadirle otras funcionalidades a la personalización del módulo de Contabilidad de Odoo, como los estados financieros y balance de comprobación de saldos, además de la inclusión de la doble moneda.

REFERENCIAS

Yeray Fernández, Alonso. 2014. Personalización de módulos en OpenERP. [book auth.] Yeray Fernández Alonso. *Personalización de módulos en OpenERP.* Universidad de valladolid, Escuela de Ingenierías Industriales, Grado en Ingeniería en Organización Industrial : Valladolid, 2014.
Comunicaciones, Ministerio de. Los ERP en Cuba. *http://www.mincom.gob.cu.* [Online] [Cited: abril 20, 2016.]
Diagramas de Caso de Uso. Tello, Jesús Cáceres. Ma. 2013. Diagramas de Caso de Uso. *http://datateca.unad.edu.co/contenidos/204023/Caceres_J._s.f._._Diagramas_de_Casos_de_uso.pdf.* [Online] Universidad de Alcalá, 2013. [Cited: mayo 1, 2016.]
Entidades(CEIGE), Centro de Informatización de. 2015. *Informe del estudio realizado a Cedrux.* Universidad de las Ciencias Informáticas(UCI) : s.n., 2015.
foundation, The apache software. 2016. www.apache.org. *The apache software foundation.* [Online] 2016. [Cited: abril 15, 2016.]
Gamma, Erich. 1994. *Design Patterns.* 1994.
gesoft. http://www.gesoft.com.br/vista/desarrollo/personalizacion-de-software.jsp. [Online] [Cited: mayo 2, 2016.]
gestiopolis. www.gestiopolis.com. *gestiopolis.* [Online] [Cited: mayo 2, 2016.]
González, Rolando Alfredo Hernández León y Sayda Coello. 2002. *El paradigma cuantitaivo de la investigación científica.* Ciudad de la Habana, Noviembre del 2002 : Editorial Universitaria, 2002. 959-16-0343-6.
Informática, Dirección de Industria. 2016. http://www.mincom.gob.cu. *Las comunicaciones al servivio de la sociedad.* [Online] 2016. [Cited: abril 6, 2016.]
Informáticas, Universidad de las Ciencias. Catalogo de productos y servicios UCI. *http://www.uci.cu/sites/default/files/Cat%C3%A1logo%202015.pdf.* [Online] [Cited: noviembre 17, 2015.]
Kemerer, Shyam Chidamber y Chris. 1994. *A metrics suite for object oriented design.* s.l. : Volumen 20, Número 6, Junio, 1994.
Larman, Craig. 2003. *UML y patrones, Introducción al análisis y diseño orientado a objetos.* s.l. : 2da edicion, 2003.
Mark Lorenz, Hatteras Software Inc., Cary, NC, Jeff Kidd, Raleigh, NC. 1994. *Object oriented software metrics.* 1994.
Microsoft. 2016. https://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx. *Revisiones de código y estándares de codificación.* [Online] 2016. [Cited: mayo 10, 2016.]
Moss, Greg. 2015. *Working with Odoo.* 2015.
Negocio, Lógica de la Aplicación ORM – Apoyo a los Procesos de. 2016. Lógica de la Aplicación ORM – Apoyo a los Procesos de Negocio. *www.github.com.* [Online] 2016. [Cited: mayo 29, 2016.]
Odoo. 2016. *Interfaz para la gestión de las cuentas contables.* s.l. : Odoo, 2016.
—. 2016. www.odoo.com. *Odoo.* [Online] 2016. [Cited: enero 20, 2016.]
openerpweb. http://www.openerpweb.es/la-contabilidad-en-openerp/. *OpenERP.* [Online] [Cited: diciembre 7, 2015.]
Paradigm, Visual. 2016. https://www.visual-paradigm.com. *Visual Paradigm, What is Visual Paradigm?* [Online] 2016. [Cited: mayo 7, 2016.]
Parra, José David. 2006. *Guía de patrones, prácticas y arquitectura.* 2006.

PgAdmin. 2016. www.pgadmin.org. *PgAdmin.* [Online] 2016. [Cited: abril 7, 2016.]
PostgreSQL. 2016. www.postgresql.org. *PostgreSQL.* [Online] 2016. [Cited: marzo 5, 2016.]
Precios, Ministerio de Finanzas y. 2015. *Resolución 1173.* 2015.
—. 2015. *Resolución 340.* La Habana : s.n., 2015.
Pressman, Roger. 2005. *Ingeniería del software.* 2005.
—. 2002. *Ingeniería del software. Un enfoque práctico.* s.l. : Quinta Edición, 2002.
Pycharm. www.jetbrains.com. *Pycharm.* [Online] [Cited: marzo 7, 2016.]
Python. 2016. www.python.org. *Python.* [Online] 2016. [Cited: febrero 3, 2016.]
Roberto Hernadez Sampieri, Carlos Fernández Collado y Pilra Baptista Lucio. 2006. *Metodología de la Investigación.* México : McGraw-Hill, 2006. 970-10-3632-.
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.* [Online] 2015. [Cited: mayo 7, 2016.]
Services, Apser Cloud. 2016. El software ERP : ejemplos, tipos y uso en la empresa. *http://www.apser.es/blog/2015/04/26/el-software-erp-ejemplos-tipos-y-uso-en-la-empresa/.* [Online] 2016. [Cited: abril 20, 2016.]
Software, Pruebas de. 2016. Gestión de calidad y pruebas de software. *http://www.pruebasdesoftware.com/pruebadeaceptacion.htm.* [Online] 2016. [Cited: junio 1, 2016.]
umldiagramadespliegue. umldiagramadespliegue.blogspot.com. *UML 2013.* [Online] [Cited: abril 20, 2016.]
Vergara, Lee Hans Morales Vidal y Juan Pablo Ojeda. 2016. Conocer el nivel de satisfacción e insatisfacción de los alumnos y alumnas de 2do año de enseñanza media por la clase de Educación Física de tres liceos. [Online] Escuela de pedagogía en Educación Física, Los Ángeles, diciembre 2016. [Cited: mayo 17, 2016.]
Weitzenfeld, Alfredo. 2010. *Modelo de dominio.* 2010.
XML. 2016. www.xml.com. *XML.* [Online] 2016. [Cited: febrero 8, 2016.]