UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Trabajo de Diploma para optar por el título

de Ingeniero en Ciencias Informáticas

Módulo de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO

Facultad 4

Autor:

Darian de Jesús Villanueva Rodríguez.

Tutores:

Ing. Julio César Espronceda Pérez.

Ing. Leannys Rodríguez Moreno.

Ing. Adrián Quesada Gómez.

La Habana, junio 2018

Declaración de autoría

Declaro que soy el único autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Autor:

_____

Darian de Jesús Villanueva Rodríguez

Tutores

_____

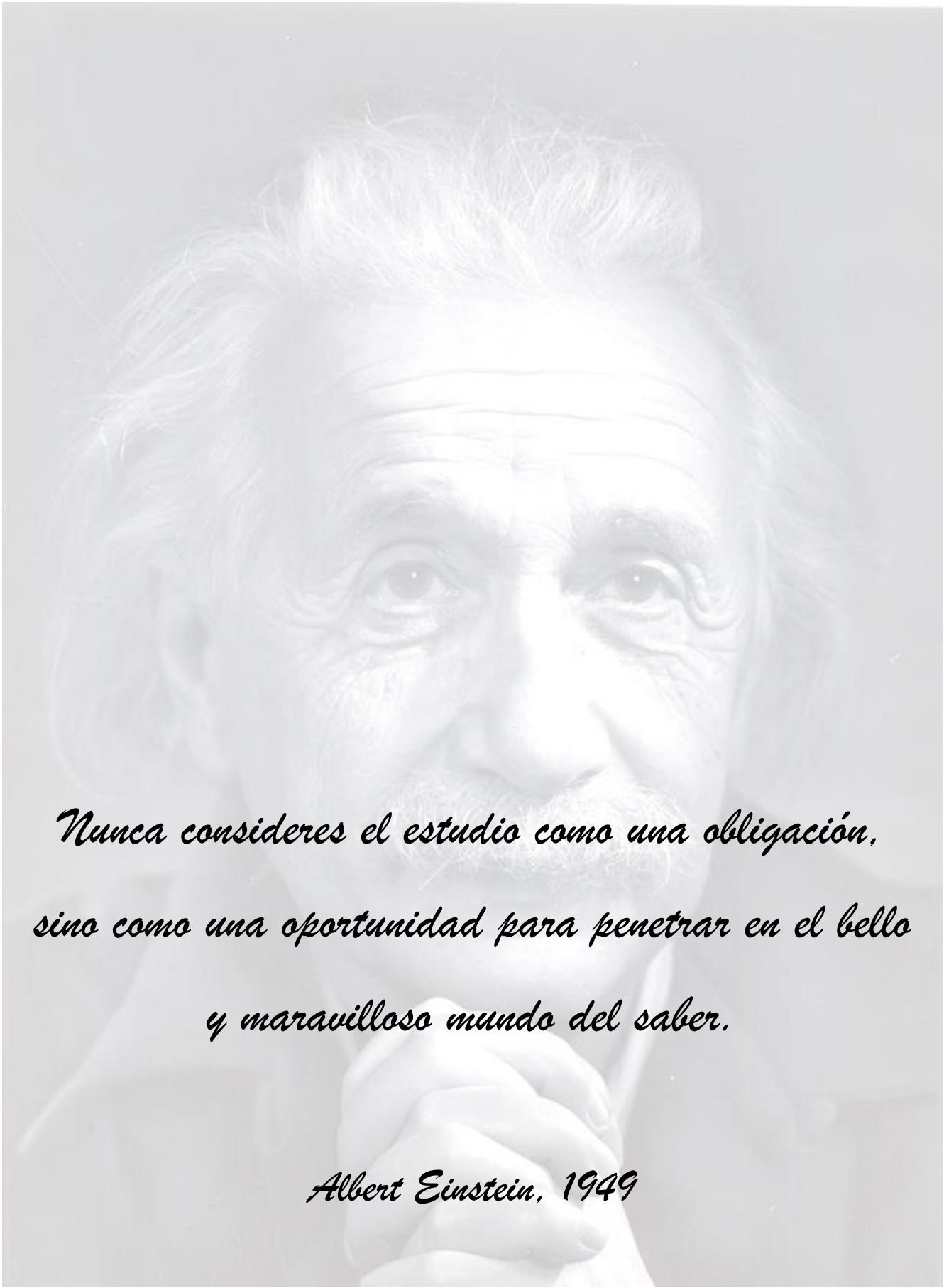
Ing. Julio César Espronceda Pérez.

_____

Ing. Leannys Rodríguez Moreno.

_____

Ing. Adrián Quesada Gómez.



Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber.

Albert Einstein, 1949

Agradecimientos

Agradecerle a mi mamá Lourdes y mi papá Sergio por ser mis ejemplos a seguir como persona y profesional. Agradecerle a toda mi familia por su apoyo incondicional. Agradecerle a mi novia Milly y familia por estar siempre a mi lado para ayudarme y apoyarme en todo este proceso. Agradecerles a todos mis amigos por ser más que amigos casi hermanos. Agradecerles a todos los integrantes de mi apartamento, que gracias a ellos la vida en la beca se hizo más fácil. En general muchas gracias a todos.

Dedicatoria

Le dedico este trabajo a mi ayita linda, a mi madre y a mi padre por ser ellos la razón por la cual hoy en día yo me hago Ingeniero en Ciencias Informáticas.

Resumen

La instalación del Sistema de Gestión Integral del Agente Transitario TRANSCARGO SIGAX desarrollado en el Centro de Tecnologías para la Formación (FORTES) que radica en la Universidad de las Ciencias Informáticas (UCI), es un proceso que actualmente se realiza de forma manual. Esto provoca que exista un considerable gasto en el tiempo que se debe utilizar para el despliegue, además aumenta el riesgo de que se cometan errores en este proceso. Si no se copian todos los archivos necesarios, el sistema no funcionará correctamente por las dependencias que existen entre ellos.

Como objetivo principal se tiene desarrollar una aplicación que posibilite la instalación de forma automática y personalizada para el Sistema de Gestión Integral del Agente Transitario TRANSCARGO SIGAX. Con este propósito se investigaron diferentes sistemas informáticos nacionales y extranjeros que arrojaron funcionalidades deseadas para el instalador a realizar. Se describen las herramientas y tecnologías utilizadas para llegar a la propuesta de solución, en la que se muestran los artefactos generados en cada una de las fases: análisis, diseño, implementación y pruebas lográndose medir la calidad del trabajo desarrollado con el uso de métricas en el diseño y pruebas funcionales a la aplicación obtenida.

Como resultado se obtuvo dos instaladores que permiten la instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO SIGAX de forma automática.

Palabras claves: ERP, instalar, SIGAX, Usabilidad.

Índice de contenido

Introducción

Un agente de carga internacional, transitario o empresa transitaria, es un agente que actúa en nombre de los importadores, exportadores y otras empresas para organizar el transporte de mercancías de forma segura, eficiente y rentable (TRANSCARGO, 2016).En Cuba la empresa transitaria de cargas internacional y nacional, TRANSCARGO S.A., es una entidad con más de 20 años de experiencia en el sector. Cuenta con profesionales especializados, con amplio conocimiento del transporte internacional, que proyectan, coordinan, controlan y dirigen todas las operaciones necesarias para realizar el transporte y la logística internacional de mercancías (TRANSCARGO, 2016).

Según su sitio oficial en internet, la empresa surge el 8 de octubre del año 1993, “como una organización económica estatal adscrita al Ministerio del Transporte. A lo largo de todos estos años TRANSCARGO ha obtenido logros que le han permitido constituirse en una empresa de referencia con un sistema de gestión de calidad avalado nacional e internacionalmente por la Oficina Nacional de Normalización y el Registro de Control de Calidad Lloyds, y tiene implantado desde el año 2003 el Perfeccionamiento Empresarial”. Esta institución, fue la primera transitaria cubana en pertenecer a la Federación Internacional de Agentes Transitarios (FIATA), premiada como Empresa de Excelencia Empresarial desde el año 2000 hasta el 2003. También fue galardonada con el Premio Giraldilla a la Excelencia Empresarial en la edición del año 2009. A escala internacional le prestigia contar con sólidos acuerdos de trabajo con empresas que constituyen altos exponentes en la actividad a nivel global, permitiéndole ofrecer a los clientes, un alcance mundial en la prestación de sus servicios, con un alto estándar (TRANSCARGO, 2016).

En la actualidad TRANSCARGO se encuentra inmersa en una serie de renovaciones dirigidas a la mejora de los servicios que se ofrecen y para ello la automatización juega un papel importante en las proyecciones futuras, por lo que se hace necesario que la informatización llegue a cada uno de los servicios que presta.

La Universidad de las Ciencias Informáticas (UCI) contiene en su estructura organizativa al Centro de Tecnología para la Formación (FORTES), donde se trabaja con Odoo en su versión 10, el cual es un sistema de planificación de recursos empresariales de código abierto completamente adaptable que permite personalizar la gestión de la información a las necesidades específicas de cada empresa, en dicho sistema han sido creados un grupo de módulos que conforman el sistema de gestión integral para el agente transitario de cargas TRANSCARGO.

En la actualidad para la realización del proceso de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO SIGAX se realizan un conjunto de pasos comenzando por la instalación y configuración de la plataforma Odoo. Dicha plataforma se instala y configura correctamente luego de realizar un serie de operaciones, comenzando por la adición del repositorio de

Odoo, la instalación de PostgreSQL y pgAdmin, así como de Odoo y otras librerías necesarias para el proyecto, seguido de la modificación del fichero odoo.conf y la configuración de la carpeta para módulos de terceros que no pertenezcan al core de Odoo, la copia de la librería Wkhtmltopdf para la ruta /usr/bin/ y por último clonar el proyecto del repositorio donde se encuentra. Luego de instalada y configurada la plataforma Odoo se deben tener en cuenta la dirección en la que se va a trabajar para el otorgamiento de los permisos, la eliminación de la base de datos para el establecimiento de una nueva dirección para la obtención de los reportes y la elección del idioma español. Una vez realizadas estas acciones se debe establecer una contraseña y se realiza el levantamiento de la imagen del sistema. Seguidamente, se establecen los permisos de administrador y se instalan los módulos que conforman SIGAX, desinstalando después los módulos que no sean necesarios para este. Luego se debe realizar los cambios en las opciones de importar y exportar del sistema y deshabilitar el idioma inglés.

La ejecución de estos pasos no cuenta con una presentación visual apropiada ya que el proceso de instalación y configuración de Odoo y de SIGAX se realiza manualmente mediante líneas de comandos en consola. Esto trae como consecuencia pérdida de tiempo, disminuyendo la eficiencia con la cual el usuario interactúa con el sistema, provocando insatisfacción con dicho proceso, que es poco atractivo, comprendido y aprendido por el usuario, por lo que disminuye su usabilidad.

Se debe tener en cuenta que una instalación exitosa es una condición necesaria para el correcto funcionamiento de cualquier software, además este proceso debe tratar de que sea lo más sencillo posible para el usuario, pues será su primera interacción con la aplicación. Debido a la cantidad de archivos y la dependencia de otros softwares que presenta el proceso de instalación y configuración de SIGAX, la probabilidad de que ocurra alguna falla durante la instalación es más probable, esto será fatal, aunque la falla sea solo parcial, ya que el objetivo que persigue la instalación, probablemente no pueda ser alcanzado pues la herramienta no funcionará correctamente y se corre el riesgo de que ocurran diversos errores cuando se trabaje con la misma.

Lo anterior expuesto denota la presencia de un problema a resolver, por lo que surge la siguiente **interrogante**: ¿Cómo lograr mayor usabilidad en el proceso de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO?

Por lo que se define como **objetivo general:** Desarrollar un módulo que garantice mayor usabilidad en la instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.

El **objeto de estudio** de este trabajo se enmarca en los procesos de instalación y configuración de módulos para plataformas web, teniendo como **campo de acción** el proceso de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.

Se definen como **tareas de investigación**:

1. Análisis de las fuentes teóricas para conformar la base teórica-metodológica de la investigación.
2. Análisis del funcionamiento y desarrollo del módulo para la instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.
3. Diseño del módulo para la instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.
4. Implementación del módulo para la instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.
5. Validar el módulo para la instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.

A partir de lo antes expuesto se tiene como **idea a defender:** el desarrollo de un módulo para mejorar la usabilidad en el proceso de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.

Posibles resultados:

El desarrollo de un módulo que garantiza una mayor usabilidad, desde la presentación visual, la facilidad de aprendizaje, la eficiencia y la satisfacción, en el proceso de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.

Los **métodos de investigación** que se utilizaron son:

Tabla 1: Métodos de investigación.

Métodos científicos	Descripción	Utilización
Métodos teóricos		
Histórico – lógico	Analizan la trayectoria completa del fenómeno, su condicionamiento a los diferentes períodos de la historia, revela las etapas principales, su desenvolvimiento y las conexiones históricas	Establecer un estudio bibliográfico y una base sólida de fundamentos teóricos que permitan relacionar el análisis documental y estado del arte, para saber con mayor precisión cómo ha ido evolucionando el desarrollo de instaladores, investigando como es

	fundamentales.	tratado el tema en la actualidad, con el objetivo de detectar los elementos a mejorar en la solución que se propone.
Analítico – Sintético	Permiten la división mental del fenómeno en sus múltiples relaciones y componentes, para facilitar su estudio y establecer mentalmente la unión entre las partes previamente analizadas, posibilitando descubrir sus características generales y las relaciones esenciales entre ellas.	Utilizado para profundizar en el tema que se desarrolla. Permite concretar los elementos más importantes relacionados con la instalación y configuración de plataformas y sistemas informáticos, posibilitando el descubrimiento de las características generales que presenta en la actualidad dicho proceso.
Modelación	Es el proceso mediante el cual se crea una representación o modelo para investigar la realidad.	Para el esbozo de los diferentes diagramas y modelos generados durante el desarrollo del instalador que contribuyan al proceso de instalación y de SIGAX, además para crear los modelos definidos en la metodología utilizada.
Métodos empíricos		

Entrevista	Es la recogida de información a través de un proceso de comunicación en el transcurso, del cual el entrevistado responde a cuestiones, previamente diseñadas en función de las dimensiones que se pretendan estudiar, planteadas por el entrevistador.	Se utilizó el proceso de comunicación verbal con el personal encargado del desarrollo del proyecto y con otras personas que contribuyen en la investigación del objeto de estudio con vista a recopilar informaciones del proceso de instalación y configuración de SIGAX.

Para lograr una mejor **organización del trabajo** la presente investigación está compuesta por tres capítulos, quedando estructurados de la siguiente manera:

Capítulo 1: "Fundamentación teórica de la investigación".

Se definen conceptos asociados al proceso de instalación y configuración de la del sistema de gestión integran del agente transitario TRANSCARGO (SIGAX) y los conceptos principales de las herramientas, metodologías, lenguajes y tecnologías que serán utilizadas para el desarrollo del trabajo de diploma.

Capítulo 2: "Propuesta de solución. Análisis y diseño".

Se realiza el análisis y diseño de la propuesta de solución, se describen los requisitos funcionales y no funcionales que posteriormente serán utilizados como punto de partida en la implementación del módulo. Se refleja el diagrama de modelo de dominio y la definición de la arquitectura a utilizar que garantiza que el sistema cumpla con los requerimientos establecidos.

Capítulo 3: "Implementación y validación de la propuesta".

Se implementan los requisitos definidos y se le realizan los diferentes tipos de pruebas que debe superar el módulo para verificar la calidad del trabajo.

Capítulo 1: Fundamentación teórica de la investigación.

En este capítulo se realiza la fundamentación teórica de la investigación, abordando temas de relevancia como los conceptos y términos asociados al dominio del problema. Se realiza un estudio de las diferentes herramientas que son útiles para la implementación del módulo. Se realiza la selección de la metodología a utilizar en el ciclo de vida del software. De igual manera se hace una revisión y elección de lenguajes de programación propuestos para el desarrollo del sistema y se analizan soluciones similares.

La palabra instalar en el diccionario de la Real Academia de la Lengua Española se define como la acción de colocar, arreglar o disponer determinados elementos para que funcionen o cumplan ciertos objetivos (RAE, 2016). Otra de las fuentes consultadas fue el libro *Beyond Software Architecture* del autor Luke Hohmann, resumiendo que un instalador es una herramienta que le permita al usuario final dar uso del software adicionándolo a la lista de programas que presente el sistema operativo en el cual trabaje.

En la investigación se define instalador como una aplicación que permite copiar los archivos necesarios de los programas a instalar en el lugar adecuado. Además, debe brindar la posibilidad de configurarlos y luego inicializarlos para que la aplicación funcione correctamente.

1.1 Vías de instalación

La instalación de un programa se puede realizar de distintas formas, cada una de ellas tiene sus particularidades. En la investigación se estudiaron varias con el objetivo de seleccionar la más factible a utilizar para realizar el despliegue de SIGAX.

Instalación mediante ficheros Bash

Bash es un programa informático cuya función consiste en interpretar órdenes. Está basado en la interfaz entre el usuario y el sistema operativo conocido como Shell de Unix y es compatible con el acrónimo de *Portable Operating System Interface* (POSIX); y se le añade la X para que sepan que se habla de Unix. Fue escrito para el proyecto GNU y es el intérprete de comandos predeterminado en la mayoría de sistemas GNU/Linux, además de Mac OS X Tiger y puede ejecutarse en la mayoría de los sistemas operativos tipo Unix. También se ha llevado a *Microsoft Windows* por el proyecto Cygwin (BASH, 2018). La instalación de programas mediante ficheros bash es utilizando comandos específicos que se encargan de la instalación de un paquete binario, estos comandos varían en dependencia del sistema operativo en que se utilicen.

Instalación mediante ficheros ".exe"

Los ficheros "*.exe"*, son ficheros ejecutables que realizan determinadas operaciones cuando se les activa. Para comenzar la instalación se ejecuta el fichero y se siguen las instrucciones de la aplicación. Es importante saber que para lograr la instalación mediante este tipo de ficheros los ordenadores deben estar equipados con sistema operativo de *Microsoft* (DOS), *Microsoft Windows*, OS/2 y *ReactOS* (Abel, 2010).

Instalación Web

Para realizar la instalación web se necesita copiar la aplicación de instalación en una estación de trabajo y se accede a ella a través de un navegador web. Una vez que se visualice se realizan las configuraciones pertinentes para que funcione el sistema correctamente. Dentro de las configuraciones necesarias se encuentra la conexión de la base de datos y que se puede acceder a la aplicación en otra estación que no sea en el servidor local solamente (Mateu, 2012).

Instalación mediante ficheros ".deb", ".rpm"

Los ficheros con extensión ".deb" son paquetes de aplicaciones preparados para instalarse de una forma sencilla en el sistema. Lo que ocurre generalmente es que se descargan, se ejecutan y luego se da clic en *Instalar.* Los paquetes se descargan dependiendo de la distribución de Linux que se utilice. Si se utiliza una versión basada en "Debian" se utilizan paquetes "deb", en cambio si se utiliza una versión basada en "*Red Hat*" se utilizan los paquetes "rpm" (Ubuntu, 2018).

Con el estudio realizado de las diferentes vías de instalación, se observó que para realizar el proceso de instalación del Sistema de Gestión Integral del Agente Transitario TRANSCARGO (SIGAX), es más factible la vía web para realizar las configuraciones al sistema. Se detectó que la instalación mediante ficheros bash, es viable para realizar la instalación y las configuraciones de la plataforma Odoo.

1.2 Estudio de soluciones similares

Instaladores existentes

- **BitNami:** es un instalador multiplataforma, con licencia GPL de aplicaciones web de software libre. Su objetivo es facilitar la instalación y configuración de gran cantidad de aplicaciones web como, por ejemplo: WordPress, Joomla, Drupal, phpBB, MediaWiki, Alfresco, entre otras. Proporciona instaladores para Linux, Windows y Mac OS y para este último, en algunos casos versiones para PowerPC y para Intel. BitNami crea paquetes, que llama stacks o pilas, que contienen todo lo necesario (programas, scripts, bases de datos, dependencias de librerías resueltas) para la

instalación de la aplicación, con total independencia del software que se tenga instalado y sin interferir en él **(BitNami, 2014)**.

Como resumen se puede decir que BitNami es:

1. Fácil de utilizar: Con solo unos clics de ratón, se puede tener una aplicación de software libre funcionando.
2. Multiplataforma: Existen BitNami Stacks disponibles para Linux, Windows y Mac OS X.
3. Independiente: Los Stacks BitNami no va a interferir con el software ya instalado en el sistema.
4. Funciona de forma nativa o virtual: Permite la instalación de la pila directamente en el sistema, o se puede ejecutar como una máquina virtual.
5. Open Source: Todas las pilas BitNami se pueden descargar libremente y utilizar en los términos de la licencia Apache 2.0.

- **Mantis:** es una aplicación de software libre, multiplataforma que permite gestionar las incidencias de las empresas, sistemas o proyectos. Está implementada en PHP y con soporte para bases de datos MySQL, PostgreSQL y MS SQL. Puede instalarse en cualquier servidor web con PHP y una de las bases de datos comentada. A nivel de cliente, Mantis puede ser accedido desde cualquier plataforma o sistema operativo, tan solo hace falta conexión a la red apropiada y un navegador de Internet. El Mantis, no tiene ninguna restricción al tipo de navegador que debe usarse para trabajar como cliente. Puede ser instalado en sistemas operativos Windows, sistemas operativos Mac OS o sistemas operativos de tipo Unix. Está desarrollado sobre PHP y requiere para su correcto funcionamiento, una base de datos (MySQL, PostgreSQL, MS SQL), un servidor de aplicaciones web (Apache) y un módulo PHP Apache (Mantis, 2014).

- **Cedrux:** este instalador se encarga del proceso de instalación de la aplicación mediante la web. Está desarrollado en su capa de negocio con PHP, siendo esta un lenguaje de programación libre y con altas prestaciones. Cuenta con varias funcionalidades como la instalación inicial del sistema, ejecutando los scripts asociados a la misma y cuenta con un módulo que permite seleccionar los subsistemas que serán instalados. Está desarrollado sobre el marco de trabajo Sauxe que a su vez utiliza Zend Framework para la capa de negocio y en la capa de presentación ExtJS. Como gestor de Base de Datos utiliza PostgreSQL en su versión 8.3 y como servidor web el Apache. Permite al usuario realizar configuraciones necesarias durante el proceso de instalación para que cuando termine la aplicación funcione correctamente **(Díaz, 2011)**.

- **Instalador Bash del sistema Cedrux:** Es el instalador que se desarrolló inicialmente para el sistema Cedrux. Consiste en un fichero Bash o intérprete de comandos que se encarga de realizar las operaciones necesarias para instalar el sistema. Este fichero se ejecuta mediante la consola de Linux haciendo del proceso de instalación algo sencillo, aunque actualmente no cumple con los aspectos o necesidades para la instalación del sistema, porque instala el sistema de forma íntegra y no por subsistemas (Velázquez, 2010).

Valoración sobre las soluciones existentes

Después de realizar el análisis de las aplicaciones existentes y haciendo énfasis en sus ventajas, se detectó que ninguno puede ser utilizado como solución a la problemática planteada. El instalador BitNami instala y configura aplicaciones web como por ejemplo: WordPress, Joomla!, Drupal, phpBB, MediaWiki y Alfresco las cuales no son necesarias para la solución del problema planteado, Mantis es una aplicación multiplataforma que permite gestionar las incidencias de las empresas, sistemas o proyectos por lo que tampoco soluciona dicha problemática, Cedrux cuenta con un instalador bash que interpreta los comandos específicos para la instalación y configuración única del sistema. El estudio de estas aplicaciones demuestra que la instalación mediante ficheros bash es factible para desarrollar las configuraciones de un software, ya que mejora la usabilidad en los procesos de instalación y configuración.

1.3 Usabilidad

La usabilidad se define como la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso. Para el estándar ISO 9241, que trata los requerimientos ergonómicos, la usabilidad es “el grado en el que un producto puede ser utilizado por usuarios específicos para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un determinado contexto de uso”. Los métodos de evaluación de usabilidad pueden ser **heurísticos** o **empíricos**. Los heurísticos (también llamados no empíricos), implican la participación de expertos especialistas en usabilidad. Los empíricos constan de técnicas e instrumentos que requieren de la participación de usuarios. (Mascheroni, C., R., G., & M., 2010)

Atributos de usabilidad

La usabilidad es una cualidad abstracta por lo cual no puede ser medida directamente. Se descompone habitualmente en “atributos”, que pueden ser medidos utilizando técnicas denominadas pruebas de usabilidad. Según el enfoque tradicional, las pruebas de usabilidad se aplican sobre el producto software para garantizar o determinar si el mismo alcanza un nivel aceptable de usabilidad. Algunos de estos atributos de usabilidad son:

1. **Facilidad de Aprendizaje:** Indica qué tan fácil es aprender la funcionalidad básica del sistema, como para ser capaz de realizar correctamente las tareas que desea llevar a cabo cualquier tipo de usuario (Mascheroni, C., R., G., & M., 2010). Las pruebas que se realizan son las de sesiones guiadas, métodos de seguimiento y las de protocolo de pensamiento manifestado ("pensar en voz alta") que implican la participación de usuarios. También puede llevarse a cabo inspecciones por parte de expertos. Lo que se trata de determinar es qué proporción de las funciones del software son evidentes al usuario en un tiempo dado (Mascheroni, C., R., G., & M., 2010). El resultado indicará que tan "fácil de aprender" es el software.
2. **Eficiencia:** La eficiencia se determina por el número de transacciones por unidad de tiempo que el usuario puede realizar usando el sistema. Lo que se busca es la máxima velocidad de realización de tareas del usuario. Cuanto mayor es la usabilidad de un sistema, más rápido opera el usuario al utilizarlo, y el trabajo puede realizarse en un menor tiempo (Mascheroni, C., R., G., & M., 2010). Se llevan a cabo inspecciones con expertos para evaluar las llamadas al sistema operativo y a la aplicación, y el tiempo de respuesta basado en ello. Se puede hacer uso de test exploratorios y sesiones guiadas con usuarios, para determinar qué cantidad de tareas pueden efectuar en un tiempo dado.
3. **Presentación visual apropiada:** El concepto de sistema se materializa al realizar el diseño de la parte visual de la interacción, es decir, la "interfaz gráfica de usuario". Hay una serie de normas provenientes del campo del diseño gráfico sobre cómo escoger los colores, tipos de letra y la disposición de los elementos en una ventana. Esta tarea suele realizarla un diseñador gráfico profesional. Un método perteneciente al prototipado que permite reproducir la interacción con un ínfimo esfuerzo de implementación es el de "Borradores en papel" con representaciones de las ventanas de aplicación. También se puede hacer uso del llamado "seguimiento del ojo" (Mascheroni, C., R., G., & M., 2010), para definir "qué es lo que miran los usuarios" durante el test, determinando qué tan fácil les resulta a los mismos interactuar con la interfaz. Es cierto que la interfaz gráfica es una parte importante del sistema, y un buen diseño de la misma puede hacer que un sistema aumente su nivel de usabilidad, pero un sistema con un diseño de interacción pobre no puede mejorar su nivel de usabilidad tan solo cambiando la interfaz gráfica.
4. **Satisfacción:** Es el atributo más subjetivo. Muestra la impresión subjetiva que el usuario obtiene del sistema. Para ello se utilizan cuestionarios, encuestas y entrevistas, diseñados especialmente para recabar un cierto "grado de satisfacción" en función de aspectos predefinidos**.**

1.4 Sistema de planificación de recursos empresariales de código abierto o ERP

Actualmente podemos observar cómo se está produciendo un crecimiento acelerado a nivel empresarial. Esto hace cada vez más complicado poder llevar un control sobre todos los recursos que forman una empresa desde humanos y financieros, hasta tecnológicos y materiales. Ante este problema en crecimiento, la tecnología de la información nos ofrece una posible solución mediante la implantación de un ERP. Estas son las siglas de "*Enterprise Resource Planning*" o también llamado sistema de planificación de recursos empresariales. Los ERP son sistemas gerenciales que integran e incorporan la mayoría de las prácticas de negocio que tienen lugar en una empresa (Fabián, 2007).

Este tipo de sistemas se caracterizan por estar formados por diferentes partes integradas en una misma aplicación y reciben el nombre de "*packaged software*". Esto quiere decir que estamos hablando de un software estándar; pero en general una empresa que adquiere un software ERP, también contrata las herramientas de desarrollo necesarias para adaptar el software a cada necesidad particular. La función principal de un ERP es organizar y estandarizar procesos y datos internos de la empresa, transformándolos en información útil para ser analizados para la toma de decisiones y de esta forma suministrar un apoyo a los clientes del negocio, para seguir tiempos rápidos de respuestas a sus problemas, así como un eficiente manejo de la información que conllevaría a una disminución en los costes de las operaciones (Fabián, 2007).

Es importante recordar que estos sistemas por si solos no establecen la toma de decisiones, deben ser administrados por personas que serán finalmente las encargadas de establecer las estrategias adecuadas para la empresa.

Características de un ERP

Cabe notar que dentro de las principales características de los ERP se destacan los sistemas integrales, modulares y adaptables.

- **Sistemas integrales:** Estas aplicaciones permiten tener un control de los diferentes procesos que se llevan a cabo en la empresa. Siendo importante destacar que todos sus departamentos se relacionan entre sí.

  Un ejemplo típico de esta estrecha relación lo podemos encontrar al momento de realizar un pedido. Este hecho conlleva la creación de una orden de venda que implicara un proceso de producción, el cual necesita una distribución. Este proceso se verá culminado con el cobro del pedido y el registro en la contabilidad de la compañía.

Si una empresa no utiliza este tipo de aplicaciones integradas necesitara varios programas que realicen un control sobre las operaciones anteriormente mentadas. Este proceso puede conducir a un posible duplicado de la información, lo que puede inducir a error en alguno de los pasos intermedios, así como la redundancia y solapamiento de trabajos con la consecuente pérdida de tiempo y dinero.

Si por el contrario utilizamos un ERP, la información no se duplicará, ya que los datos solo se introducen una vez y el sistema se encarga de realizar los cambios necesarios en cada uno de los apartados, proporcionando además protección de la información (Fabián, 2007).

- **Sistemas modulares:** Los ERP entienden la empresa como un conjunto de departamentos que se encuentran relacionados por la información que comparten, la cual se crea a partir de los propios procesos de la compañía.

  Por otro lado, al ser modular tiene la capacidad de instalar solo las partes en las cuales esté interesada el cliente. Esto se puede considerar una ventaja tanto técnica como económica para el comprador (Fabián, 2007).
- **Sistemas adaptables:** A pesar de ser un software con un diseño estándar, nos permite adaptar el sistema a las necesidades de cada uno de los clientes y a su forma de trabajo, proporcionando gran versatilidad. Esto se consigue mediante la configuración y parametrización de cada uno de los procesos. La parametrización nos permite diseñar y amoldar nuestro sistema a las necesidades de la empresa contratante (Fabián, 2007).
- **Otras características de los sistemas ERP son:**
  1. Utilización de una base de datos centralizada.
  2. Los componentes e ERP interactúan entre ellos para afianzar todas las operaciones que realizan en cada uno de los módulos.
  3. La incorporación de datos en un ERP se efectúa una sola vez, pero estos datos tienen que ser consistentes, completos y comunes.
  4. Las empresas que toman la decisión de implantar un sistema ERP suelen tener que realizar pequeñas modificaciones en algunos de sus procesos para adaptarse a los procesos del sistema ERP. Este proceso de adaptación recibe el nombre de reingeniería de procesos.
  5. Dentro de un sistema ERP podemos encontrar menús modulares en función de los roles de cada usuario, pero la base de la aplicación es un sistema único.
  6. El objetivo principal de este tipo de sistemas es ofrecer aplicaciones especializadas para cada empresa en particular. Este tipo de especificación recibe el nombre de versiones sectoriales, las cuales se adaptan en función de los procesos de negocios del sector en cuestión.

Ventajas que supone la implantación de un ERP

- La utilización de un ERP constituye una importante estrategia de comercio electrónico donde su uso y utilización aporta una mejora en la productividad de la empresa y ofrece una mayor ventaja competitiva en contraste con las diferentes empresas del mismo sector.
- Garantiza un exhaustivo control de datos e información, ya que estos estarán sometidos a filtros constantes de control quedando registrada cualquier operación.
- El tiempo y el coste requerido para la administración de áreas funcionales como atención al cliente, recursos humanos, manejo de inventario o proveedores se optimizan y se benefician gracias al manejo y el uso oportuno de los datos convertidos en cuanto al servicio al cliente y la atención de los mismos.
- Se optimiza la toma de decisiones gerenciales, así como todo lo relacionado con los diferentes procesos empresariales, por tener la capacidad de contar con información fiable, oportuna y veraz.
- Los sistemas ERP cuentan con herramientas flexibles para trabajar en red, lo que beneficia enormemente a las empresas que tienen diferentes departamentos.

Limitaciones que podemos encontrar en un ERP

- Un cambio continuo de personal puede provocar un fracaso durante la explotación del sistema. Las compañías emplean administradores que no están capacitados para el manejo del sistema ERP que tiene la compañía que los contrata.
- Los sistemas ERP son vistos como sistemas rígidos. Es difícil de adaptar al flujo especifico de los trabajadores y al proceso de negocio de algunas empresas, siendo este una de las principales causas de falla, ya que hay diferentes maneras de hacer negocios en función de las empresas y este tipo de sistema es poco flexible para la personalización y elaboración de ciertos reportes necesarios para las compañías.
- Los sistemas de planificación de recursos empresariales pueden resultar difíciles de utilizar. Parte de la información que en ellos se utiliza se organiza en módulos de manera muy compleja, por lo que el sistema resulta poco práctico y con difícil acceso a los datos.
- Otro de los problemas es la reticencia por parte de algunos departamentos a compartir la información interna, ya que puede reducir la eficiencia del software.
- Un gran obstáculo es el tiempo de implantación de un ERP. Este tiempo puede resultar variable ya que depende del tipo de implementación que necesite.

Odoo es un sistema de planificación de recursos empresariales de código abierto anteriormente conocido como OpenERP. El cambio de nombre se debe a que desde su versión 8.0 Odoo ha evolucionado y ha incorporado funcionalidades muy interesantes y útiles (Odoo, Open Source ERP and CRM | Odoo., 2016).

Características de Odoo

- **Sistema de código abierto**: esto significa que cualquier módulo puede ser modificado o diseñado desde cero, adaptándolo a las necesidades de la empresa.
- **Multi-plataforma:** independientemente del sistema operativo que utilice, a través de un navegador web se podrá acceder a su interfaz.
- **Fácil manejo:** no son necesarios grandes conocimientos de informática para poder utilizarlo gracias a su sencilla interfaz.
- **Gran comunidad:** miles de empresas y personas de todo el mundo colaboran cada día en el desarrollo de Odoo.
- **Modular:** con más de dos mil módulos liberados disponibles que pueden combinarse entre sí.
- **Integración con otras aplicaciones:** visualización de pdf, importación/exportación de documentos de *Microsoft Office* u *Open Office*, compatibilidad con *Google Maps*.
- **Flexible:** Puedes contratar únicamente lo que necesites. Lo más habitual es contratar el proceso de implantación a una empresa especializada y dado si carácter modular se podrá instalar solamente aquellas partes que realmente necesites.

Características técnicas de Odoo

La arquitectura del sistema es cliente/servidor aplicable tanto en Linux como en Windows, de esta forma todos los usuarios pueden trabajar sobre el mismo repositorio de datos. Odoo se puede utilizar desde cualquier navegador y dispositivo, ya sea desde una computadora de escritorio, una laptop, Tablet, o móvil y desde cualquier parte del mundo. Dispone de una vista simplificada, pero con todas las funciones disponibles en una vista completa, adaptada específicamente para pantallas de Smartphone (Odoo, Open Source ERP and CRM | Odoo., 2016). Odoo permite total control de stocks, almacenes e inventarios, la gestión de proyectos y tareas, el manejo de ventas y facturas, la contabilidad analítica y control del trabajo, la planificación/ejecución de órdenes de trabajo, el control de procesos productivos para manufactura, el marketing relacionado con los clientes, la gestión de tesorería que son los cobros y los pagos y la gestión del personal de la empresa.

1.5 Metodología, herramientas y tecnologías a utilizar para el desarrollo del componente

El centro FORTES se especializa en desarrollar tecnologías que permitan ofrecer servicios y productos para la implementación de soluciones de formación, aplicando las Tecnologías de la Información y las Comunicaciones a todo tipo de instituciones con modelos de formación y condiciones tecnológicas diferentes. En el módulo a implementar se seleccionan un conjunto de tecnologías y herramientas atendiendo a sus características y a las nuevas políticas de desarrollo del centro.

Tecnologías

- Suite de aplicaciones: Odoo v10.0.
- Gestor de base de datos: PostgreSQL v9.5.
- Lenguaje de programación para el servidor: Python
- Lenguaje para el cliente: XML

Metodología de desarrollo de software

El surgimiento de fundamentos de la metodología tiene inicios a partir de la fuerte necesidad de llevar un determinado proyecto a su meta deseada, estos fundamentos se adaptaron al desarrollo de software. La nueva etapa de adaptación contenía el desarrollo dividido en etapas de manera secuencial que permitía el mejoramiento de la necesidad existente en el campo del software. Entre las principales metodologías tradicionales se encuentra RUP, que centra su atención en llevar una documentación exhaustiva de todo el proyecto y XP como metodología ágil que garantiza la calidad del software desarrollado, haciendo que este supere las expectativas del cliente. La metodología AUP se forma de la recopilación de los elementos positivos de las dos metodologías mencionadas anteriormente, que fue adaptada a las necesidades de sus proyectos para lograr una estandarización en cuanto al desarrollo de cada producto. Para el desarrollo del módulo se escogió esta metodología debido a que AUP-UCI es la propuesta en SIGAX por política del proyecto, además provee una organización detallada con respecto a la información para su posterior utilización en el mantenimiento del sistema, y de esta forma lograr la integración del módulo al sistema en general (Sánchez, 2012).

La metodología AUP aplica técnicas ágiles incluyendo:

1. Desarrollo Dirigido por Pruebas (*test driven development* - TDD en inglés).
2. Modelado ágil.
3. Gestión de cambios ágil.

Fases de AUP:

1. Inicio.
2. Elaboración.

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva.

3. Construcción.
4. Transición.
5. Gestión de configuración.
6. Gestión de proyectos.
7. Entorno.

Variación de AUP para la UCI

Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, entre otras) exigiéndose así que el proceso sea configurable, se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI. Una metodología de desarrollo de software tiene entre sus objetivos aumentar la calidad del software que se produce, de ahí la importancia de aplicar buenas prácticas, la cual constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora. Estas prácticas se centran en el desarrollo de productos y servicios de calidad (Sánchez, 2012).

Características por escenarios

Escenario No 1: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que puedan modelar una serie de interacciones entre los trabajadores del negocio/actores del sistema (usuario), similar a una llamada y respuesta respectivamente, donde la atención se centra en cómo el usuario va a utilizar el sistema. Es necesario que se tenga claro por el proyecto que los CUN muestran como los procesos son llevados a cabo por personas y los activos de la organización.

Escenario No 2: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

Escenario No 3: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio con procesos muy complejos, independientes de las personas que los manejan y

ejecutan, proporcionando objetividad, solidez, y su continuidad. Se debe tener presente que este escenario es muy conveniente si se desea representar una gran cantidad de niveles de detalles y la relaciones entre los procesos identificados.

Escenario No 4: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido. El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos. Se recomienda en proyectos no muy extensos, ya que una HU no debe poseer demasiada información. Todas las disciplinas antes definidas (desde Modelado de negocio hasta Pruebas de Aceptación) se desarrollan en la Fase de Ejecución, de ahí que en la misma se realicen iteraciones y se obtengan resultados incrementales. En una iteración se repite el flujo de trabajo de las disciplinas: requisitos, análisis y diseño, implementación y pruebas internas. De esta forma se brinda un resultado más completo para un producto final de manera creciente. Para llegar a lograr esto, cada requisito debe tener un completo desarrollo en una única iteración (Sánchez, 2012).

Para la realización del módulo se decide utilizar el escenario No 3 debido a que este modela un negocio y utiliza la descripción de requisitos por proceso, este escenario está definido por la metodología del proyecto.

Herramienta CASE

Una herramienta CASE es aquella que permite la creación de los artefactos necesarios para el modelado de la propuesta de solución, siguiendo una metodología en la construcción de un software. En la utilización de la metodología AUP, se hace necesario el uso de las herramientas Case en las fases de Análisis, Diseño e Implementación, donde se generan la mayor cantidad de artefactos, a los cuales, es muy fácil realizarles los cambios que ocurren en el diseño de un software, precisamente por ser modelados con estas herramientas. Permite mayor calidad y rapidez del software.

Lenguaje de modelado

UML

El Lenguaje Unificado de Modelado (UML) es, tal como su nombre lo indica, un lenguaje de modelado y no un método o un proceso. UML está compuesto por una notación muy específica y por las reglas semánticas relacionadas para la construcción de sistemas de software. UML en sí mismo no prescribe ni aconseja cómo usar esta notación en el proceso de desarrollo o cómo parte de una metodología de diseño orientada a objetos (LLC, 2015).

UML soporta un conjunto rico en elementos de notación gráfica. Describe la notación para clases, componentes, nodos, actividades, flujos de trabajo, casos de uso, objetos, estados y cómo modelar la relación entre esos elementos. UML también soporta la idea de extensiones personalizadas a través elementos estereotipados. UML provee beneficios significativos para los ingenieros de software y las organizaciones al ayudarles a construir modelos rigurosos, trazables y sostenibles, que soporten el ciclo de vida de desarrollo de software completo (LLC, 2015). Ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del mismo, aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

Visual Paradigm 8.0 para UML

Visual Paradigm es una Herramienta CASE que soporta todo el ciclo de vida del desarrollo de un *software*: Análisis y Diseño, Construcción, Pruebas y Despliegue. Además, permite elaborar todos los diagramas de clases, casos de uso y diagramas de actividades. Genera código y documentación desde los diagramas y posibilita el diseño de prototipos de interfaz de usuario. Proporciona además diferentes tutoriales que sirven para un mejor entendimiento de la herramienta (Headquarters, 2012).

Permite el diseño de software con el UML 2.0, en general proporciona un entorno unificado de diseño de software para el analista de sistemas y desarrollador de software donde deben analizar, diseñar y mantener aplicaciones de software en una disciplina (Headquarters, 2012). Es una herramienta multi-plataforma y muy fácil de usar. Teniendo en cuenta que está definida por el proyecto como una de las herramientas a usar en el sistema SIGAX y las ventajas que ofrece la misma, se decide su utilización como Herramienta CASE en el presente trabajo.

Lenguajes de desarrollo

Python es un lenguaje de scripting independiente de plataforma y orientado a objeto, preparado para realizar cualquier tipo de programa. Python es interpretado, lo que trae consigo que no necesita compilar los programas para ejecutarlos, por lo que ofrece como ventajas la rapidez de desarrollo e inconvenientes como una menor velocidad (ALEGSA, 2014).

En los últimos años este lenguaje se ha hecho muy popular, algunas de las principales razones son:

- La cantidad de librerías que contiene, tipos de datos y funciones y funciones incorporadas en el propio lenguaje.
- La sencillez y velocidad con que crean los programas.
- La cantidad de plataformas en las que es posible desarrollar, como Unix, Windows y Mac.
- Es gratuito hasta para propósitos empresariales.

Entorno de Desarrollo Integrado (IDE)

Un IDE consiste básicamente en un software que previamente ha sido instalado en la máquina del cliente y cuyo principal objetivo es el desarrollo de otro software, permite mantener proyectos informáticos, los cuales se pueden implementar en diferentes lenguajes de programación, así como realizar una serie de operaciones básicas sobre ellos.

PyCharm

Desarrollado por la compañía *Jetbrains*, está basado en IntelliJ IDEA, el IDE de la misma compañía, pero enfocado hacia *Java* y la base de *Android Studio*. Pycharm tiene cientos de funciones que lo puede ver como una herramienta pesada, pero ayuda con el desarrollo del día a día (JetBrains,2016). Dado a que PyCharm es una herramienta definida por el proyecto y el lenguaje en el que se debe desarrollar la aplicación es Python se llega, a la conclusión de que el entorno de desarrollo más adecuado para el mismo es PyCharm.

Algunas de las características de PyCharm:

- Autocompletado, resaltador de sintaxis, herramienta de análisis y refactorización.
- Integración con *framework* web como: Django, Flask, Pyramid, Web2Py.
- Soporta entornos virtuales e intérpretes de Python 2.x, 3.x, PyPy, Iron Python y Jython.
- Sistemas de control de versiones: Git, CVS, Mercurial.
- *Framework javascripts*: jQuery, Angular JS.
- *Debugger* avanzado de Python y Javascript.

Sistema gestor de base de datos(SGBD)

Los sistemas gestores de bases de datos, también conocidos como sistemas manejadores de bases de datos o DBMS (DataBase Management System), son un conjunto de programas que manejan todo acceso a la base de datos, con el objetivo de servir de interfaz entre ésta, el usuario y las aplicaciones utilizadas.

Gracias a este sistema de software específico el usuario puede gestionar la base de datos (almacenar, modificar y acceder a la información contenida en ésta) mediante el uso de distintas herramientas para su análisis, con las que puede realizar consultas y generar informes (SGDB, 2017).

PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto

más potente del mercado y en sus últimas versiones no tiene nada que envidiarles a otras bases de datos comerciales. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (SGDB, 2017)

A continuación, se muestran algunas de las características más importantes y soportadas por PostgreSQL:

- Completa documentación.
- Licencia BSD.
- Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit.
- Funciones/procedimientos almacenados en numerosos lenguajes de programación, entre otros PL/pgSQL (similar al PL/SQL de oracle), PL/Perl, PL/Python y PL/Tcl.

1.6 Conclusiones del capítulo

En este capítulo se realizó la investigación de la base teórica que fundamenta el desarrollo de la aplicación, a continuación, se detallan los aspectos abordados como resultado del estudio preliminar:

- El estudio de sistemas similares arribó a que estos cumplen algunas de las exigencias, pero no son integrables al sistema, por lo que se demuestra la necesidad de dichos módulos.
- Queda definida la metodología AUP-UCI haciendo uso del escenario 3, por la necesidad de modelar un negocio y se realizará la descripción de cada requisito definido.
- Se utilizaron como lenguajes de programación Python, Odoo v10.0 como suite de aplicaciones y el entorno de desarrollo integrado PyCharm, esto nos permite lograr una integración con el sistema debido a que se utilizan las herramientas definidas por el proyecto.

Capítulo 2: Propuesta de solución. Análisis y diseño.

La obtención y entendimiento de los requisitos es un paso fundamental en el desarrollo de software en la actualidad, de esto depende que el ciclo de vida del proyecto sea estable y continuo, por eso se realiza una detallada descripción de cada uno de los requisitos que se clasifican en funcionales y no funcionales. En el presente capítulo también se proponen los diagramas que describen los procesos del negocio, desarrollando el modelo de dominio. Se utilizará el patrón modelo-vista-controlador con el cual trabaja Odoo. De manera general el capítulo recoge el flujo de análisis y diseño del sistema a desarrollar.

2.1 Propuesta de solución

Como se ha expuesto hasta este punto de la investigación, para dar solución al problema tratado se propone el desarrollo de un módulo que permita realizar el proceso de instalación y configuración de SIGAX de una manera más fácil, diseñada completamente por las tecnologías y herramientas expuestas en el capítulo 1, permitiendo dos instalaciones cómodas, agradable a la vista y de fácil entendimiento.

El módulo debe poder ser utilizado por cualquier persona con los conocimientos mínimos sobre informática, dándole una información del estado de la instalación del sistema durante toda la duración de la misma. Entre las principales acciones que debe permitir el módulo es la instalación y configuración de la plataforma Odoo, instalación de Postgresql y pgAdmin, así como otras librerías necesarias para el proyecto, seguido de la modificación del fichero odoo.conf y la configuración de la carpeta para módulos de terceros que no pertenezcan al core de Odoo, el otorgamiento de los permisos, la eliminación de la base de datos para el establecimiento de una nueva dirección para la obtención de los reportes, la elección del idioma español, el establecimiento de una contraseña, otorgar los permisos de administrador e instalar los módulos que conforman SIGAX, desinstalar los módulos que no sean necesarios para este y eliminar las opciones de importar y exportar del sistema.

2.2 Modelo de dominio

En el modelo de dominio se exponen las clases principales o entidades y sus interacciones en el proceso de desarrollo, el modelo de dominio se crea también para tener una mejor comprensión del módulo a desarrollar. Es una representación visual estática del entorno real donde se desarrollará el sistema. Muestra (a los modeladores) clases conceptuales significativas en un dominio de problema; es el artefacto más importante que se crea durante el análisis orientado a objetos (Larman, 2003).

Conceptos de las clases del modelo de dominio

1. Odoo: plataforma de gestión modular completamente adaptable.
2. Módulos: grupo de funcionalidades que integran la plataforma Odoo.
3. SIGAX: modulo del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.
4. Instalación y configuración: procesos de instalación y configuración de Odoo y SIGAX.
5. Consola: aplicación utilizada para ejecutar comandos y scripts.
6. Código: instrucciones o comandos a bajo nivel para ejecutar en consola.
7. Usuario: persona encargada de la instalación y configuración en consola de Odoo y SIGAX.

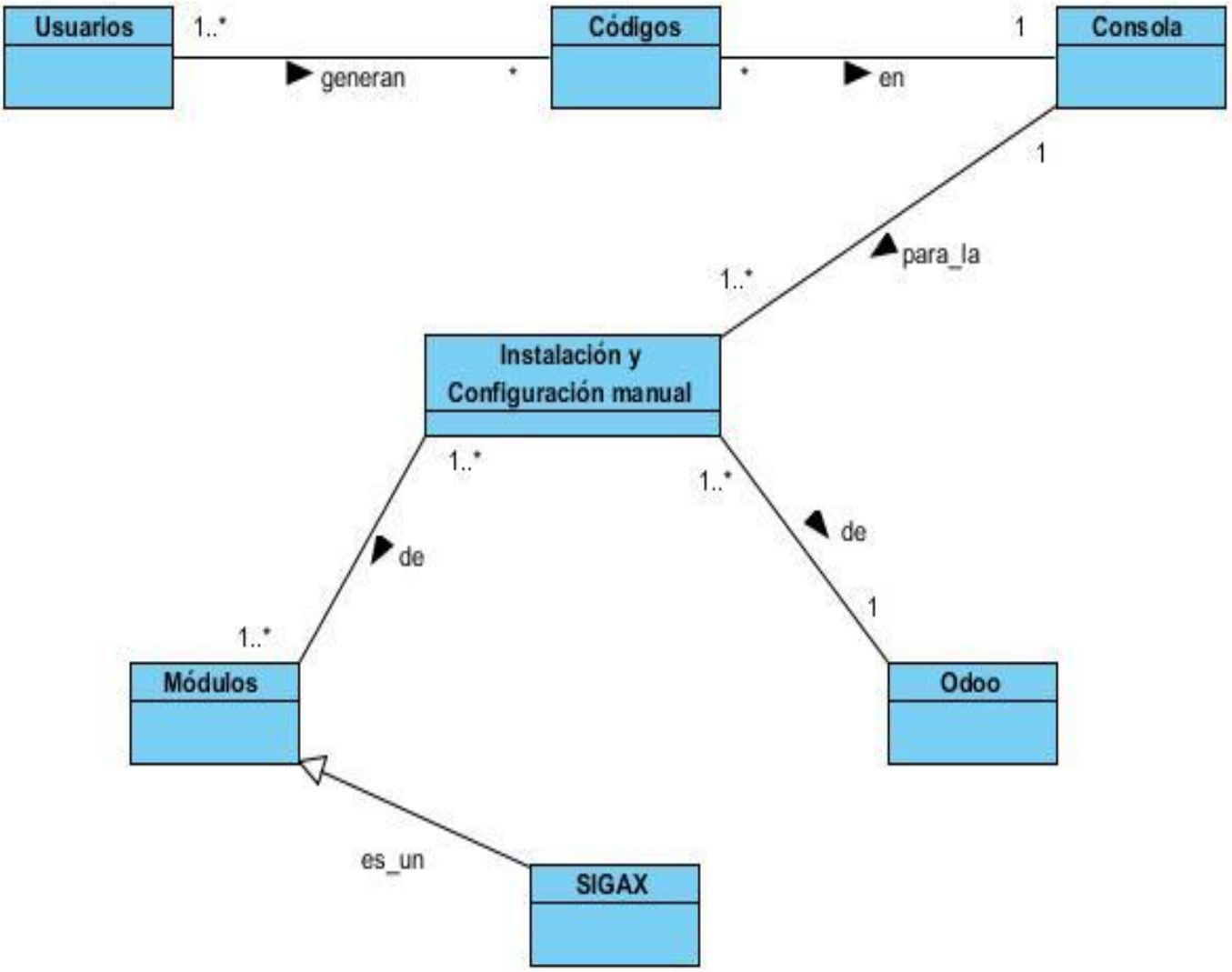


Figura 1: Modelo de dominio, (Fuente: elaboración propia)

2.3 Patrón arquitectónico Modelo Vista Controlador (MVC)

El patrón MVC fue diseñado para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos. Sus características principales están dadas por el hecho de que, el Modelo, las Vistas y los Controladores se tratan como entidades separadas; esto hace que cualquier cambio producido en el Modelo se refleje automáticamente en cada una de las Vistas. Este modelo de arquitectura se puede emplear en sistemas de representación gráfica de datos, donde se presentan partes del diseño con diferente escala de aumento, en ventanas separadas (Martínez, 2018).

Al incorporar el modelo de arquitectura MVC a un diseño, las piezas de un programa se pueden construir por separado y luego unirlas en tiempo de ejecución. Si uno de los componentes, posteriormente, se observa que funciona mal, puede reemplazarse sin que las otras piezas se vean afectadas. Este escenario contrasta con la aproximación monolítica típica de muchos programas de pequeña y mediana complejidad. Todos tienen un *frame* que contiene todos los elementos, un controlador de eventos, varios cálculos y la presentación del resultado (Martínez, 2018).

El MVC separa el desarrollo de las aplicaciones en 3 capas como aquí se muestra (Martínez, 2018):

- El Modelo es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo.
- La Vista es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa preferentemente con el Controlador, pero es posible que trate directamente con el Modelo a través de una referencia al propio Modelo.
- El Controlador es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo. Centra toda la interacción entre la Vista y el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo.

Odoo sigue una arquitectura MVC y recorre todas las capas durante la implementación de la aplicación. El modelo, define la estructura de los datos, la vista describe la interfaz con el usuario y el controlador soporta la lógica de negocio de la aplicación. La capa modelo es definida por objetos Python cuyos datos son almacenados en una base de datos PostgreSQL. El mapeo de la base de datos es gestionado

automáticamente por Odoo, y el mecanismo responsable de esto es el modelo objeto relacional, (ORM - *object relational model*). La capa vista describe la interfaz con el usuario. Las vistas son definidas usando el lenguaje XML, las cuales son usadas por el marco de trabajo (*framework*) del cliente web para generar vistas HTML de datos. Las vistas del cliente web ejecutan acciones de datos persistentes a través de la interacción con el servidor ORM. Estas pueden ser operaciones básicas como escribir o eliminar, pero pueden también invocar métodos definidos en los objetos Python del ORM, ejecutando lógica de negocio más compleja.

El patrón modelo vista controlador en la plataforma Odoo puede ser representado de la siguiente manera (Odoo, 2016):

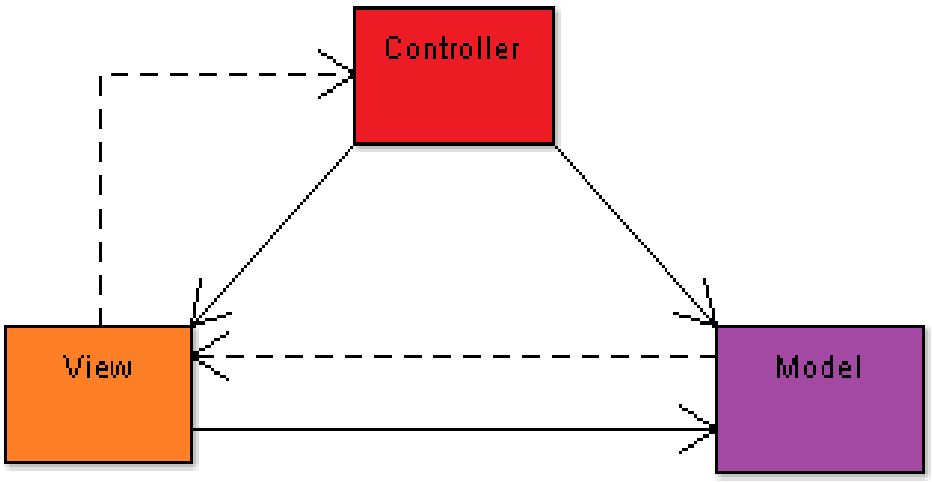


Figura 2: Patrón arquitectónico (Fuente: elaboración propia)

El instalador de SIGAX se desarrolla en la capa del modelo mediante la clase sigax_model_config y en la capa de la vista a través de la clase sigax_view_config.

2.4 Requerimientos de la aplicación

La ingeniería de requerimientos es un proceso trabajoso, extenso, el cual precisa de un análisis intenso del sistema a desarrollar, pues este provee un cambio en el entorno y las relaciones entre las personas implicadas en el mismo, haciéndose necesario la identificación de dichas personas, considerando sus necesidades y asegurando que entiendan todo lo relacionado al proceso (Sommerville, 2005).

Requisitos funcionales del sistema

Los requerimientos funcionales de un sistema describen lo que el sistema debe hacer. Estos requerimientos dependen del tipo de software que se desarrolle, de los posibles usuarios del software y del enfoque general tomado por la organización al redactar requerimientos. Cuando se expresan como

requerimientos del usuario, habitualmente se describen de forma bastante abstracta. Sin embargo, los requerimientos funcionales del sistema describen con detalle la función de este, sus entradas y salidas y excepciones. En resumen los requerimientos funcionales definen la manera en que el sistema debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (Sommerville, 2005).

Técnica para la captura de requisitos

Para realizar la captura de requisitos se utilizaron las siguientes técnicas:

Entrevista:

Las entrevistas son utilizadas para recopilar información de manera verbal, mediante preguntas que realiza el analista a los diferentes especialistas. Los dos tipos básicos son (PRESSMAN, 1988):

- **Preguntas abiertas** que permiten al entrevistado expresar de manera flexible y en consideración de su experiencia responder a lo que se le pregunta.
- **Preguntas cerradas** que limitan las respuestas disponibles de los entrevistados mediante una lista de opciones o alternativas de respuestas.

En este caso se le realizó la entrevista a Celia Indira, analista y líder del proyecto de desarrollo con gran conocimiento sobre el tema tratado y a los demás integrantes del equipo de despliegue. Las preguntas realizadas fueron preguntas abiertas. Para poder observar las preguntas realizadas ver Anexo 1.

Tormenta de ideas:

La tormenta de ideas, también conocida como lluvia de ideas, sirve de guía para elegir un tema o una idea. Esta técnica se aplicó a los miembros del equipo de despliegue donde se acumularon ideas para tener una perspectiva general de las necesidades del sistema mediante talleres donde su objetivo era que los miembros del personal de despliegue opinaran acerca del proceso de instalación.

Listado de requisitos funcionales

RF1: Instalar la plataforma Odoo.

RF1.1: Autenticar como súper usuario (contraseña).

RF1.2: Validar funcionamiento del repositorio.

RF1.3: Descargar llave GPG.

RF1.4: Crear área de trabajo.

RF1.5: Instalar gestor de base de datos.

RF1.6: Configurar gestor de base de datos (usuario, contraseña).

RF1.7: Actualizar configuraciones del gestor de base de dato.

RF1.8: Instalar servicio Odoo.

RF1.9: Configurar el acceso a la base de datos.

RF1.10: Reiniciar servicio Odoo.

RF1.11: Instalar el sistema de control de versiones (GIT).

RF1.12: Descargar del GIT las carpetas de los módulos que conforman SIGAX.

RF2: Instalar SIGAX.

RF2.1: Instalar módulo Agenciamiento Aduanal.

RF2.2: Instalar módulo Arrendamiento de Contenedores.

RF2.3: Instalar módulo Insurance.

RF2.4: Instalar módulo Nomenclator.

RF2.5: Instalar módulo Facturación.

RF2.6: Instalar módulo Jasper Report.

RF2.7: Instalar módulo Correspondent.

RF2.8: Instalar módulo Logística.

RF2.9: Instalar módulo SIGAX-Transporte.

RF2.10: Instalar módulo Commercial.

RF2.11: Instalar módulo Main.

RF2.12: Eliminar la acción importar en el sistema.

RF2.13: Cambiar logo del sistema.

RF2.14: Generar reportes.

Descripción de requisitos por proceso

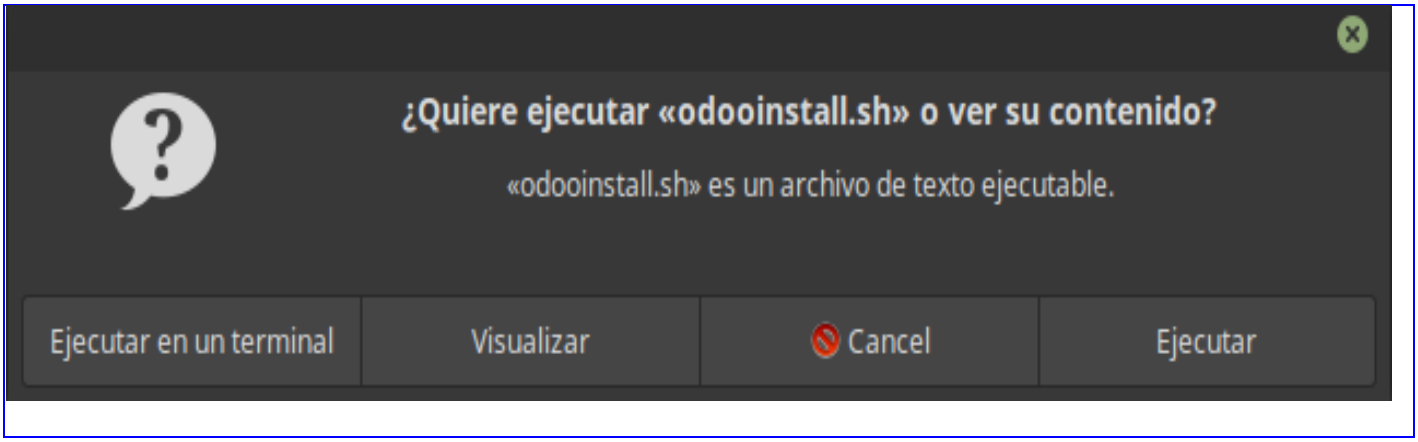
La descripción de los requisitos es muy importante para saber el flujo de vida del proceso. En la siguiente tabla se describe detalladamente cada uno de los elementos a tener en cuenta para el desarrollo del módulo.

Descripción de requisito para instalar la plataforma Odoo.

Tabla 2: Descripción de requisitos

Precondiciones	El usuario debe poseer los permisos de súper usuario para instalar la plataforma Odoo.
Flujo de eventos	
Flujo básico instalar plataforma Odoo.	
1	El usuario inicia la consola de comandos de Linux.
2	El usuario introduce el comando para ejecutar el instalador.
3	El sistema debe de informar si el usuario está ejecutando el instalador como súper usuario o como un usuario común.
4	El usuario ejecuta el instalador como súper usuario.
5	El sistema limpia la consola.
6	El sistema verifica que los repositorio se encuentran disponible.
7	El sistema descarga las llaves necesarias para el proceso de instalación.
8	El sistema crea el área de trabajo.
9	El sistema instala el gestor de bases de datos PostgreSQL.
10	El sistema configura el gestor de bases de datos PostgreSQL.
11	El sistema actualiza las configuraciones en el gestor de bases de datos PostgreSQL.
12	El sistema instala el servicio Odoo.
13	El sistema reinicia el servicio Odoo.
14	El sistema crea la carpeta donde se copiarán los módulos de SIGAX.
15	El sistema cambia los permisos de la carpeta donde se copiarán los módulos de SIGAX.

16	El sistema verifica que el GIT esté instalado en el ordenador en el que se está trabajando.
17	El sistema instala el GIT.
18	El sistema solicita usuario, contraseña y rama.
19	El usuario ingresa su usuario, su contraseña y la rama que desea descargar.
20	El sistema descarga del GIT los módulos que conforman SIGAX.
21	El sistema copia hacia la carpeta SIGAX los módulos descargados del GIT.
22	Concluye así el requisito.
Pos-condiciones	
	Se instaló la plataforma Odoo satisfactoriamente.
Flujos alternativos	
Flujo alternativo 3.a	
1.	Muestra mensaje *"Tiene que ejecutarse con permisos de administrador ".*
2.	El usuario introduce y/o selecciona los datos.
3.	Volver al paso 4 del flujo básico.
Flujo alternativo 6.a	
1.	Muestra mensaje *"Los repositorios necesarios para la instalación no se encuentran disponibles ".*
Flujo alternativo 16.a	
1.	Muestra mensaje *"GIT no encontrado, se procede a la instalación".*
2.	Volver al paso 17 del flujo básico.
Flujo alternativo 16.b	
1.	Muestra mensaje *"GIT encontrado".*
2.	Volver al paso 17 del flujo básico.
Prototipo de interfaz	



Descripción de requisito para instalar SIGAX.

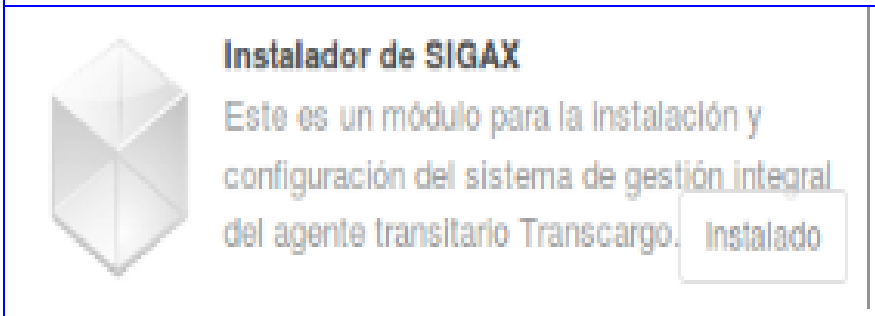
Tabla 3: Descripción de requisitos

Precondiciones	El usuario debe de estar autenticado en la plataforma Odoo.
Flujo de eventos	
Flujo básico instalar plataforma Odoo.	
1	El usuario busca los módulos que pertenecen a SIGAX.
2	El usuario da clic en el botón *" instalar "* del módulo llamado Instalador de SIGAX .
3	El sistema instala los módulos llamados: • Arrendamiento de Contenedores. • Insurance. • Nomenclator. • Jasper Report. • Facturación. • Correspondent. • Logística. • SIGAX-Transporte. • Commercial. • Main. • Agenciamiento Aduanal
4	El sistema elimina la acción de exportar.

5	El sistema elimina la acción importar.
6	El sistema actualiza el logo que viene predefinido por Odoo.
7	El sistema genera los reportes que son de: • Solicitud de embarque. • Contratos próximos a vencer. • Cantidad de servicios por períodos. • Exportaciones. • Servicios a la importación por empresa importadora. • Atención al cliente. • Tipo de rectificación y corresponsal. • Total de BL citados. • Cantidad de contenedores tramitados. • Facturación. • Facturación total. • Contenedores por cliente. • Contenedores por estado. • Contenedores por proveedor. • Contenedores por tamaño y tipo. • Explotación. • Bultos despachados por corresponsal. • Bultos aparecidos en almacén. • Contenedores desagrupados según el tipo de carga. • Bultos por corresponsal en almacén. • Bultos retenidos por Aduana. • Trazabilidad en almacén. • Bultos por estado. • Extracciones en el CADC. • Carga existente en almacén. • Bultos según la cantidad de días de almacenaje. • Bultos despachados por contenedor. • Carga en almacenes por contenedor. • Contenedores arribados al país.

	• Contenedores desagrupados. • Contenedores vacíos. • Bultos desagrupados. • Trazabilidad de contenedores llenos en el CADC. • Contenedores extraídos del Puerto Mariel. • Trazabilidad de contenedores llenos en la TCM. • Trazabilidad de contenedores devueltos. • Contenedores llenos en el CADC. • Contenedores llenos en el país con estadía. • Carga comercial en existencia.
8	Concluye así el requisito.
Pos-condiciones	
	Se instaló SIGAX satisfactoriamente.
Prototipos de interfaz	
	


Instalador de SIGAX
Por Centro de desarrollo FORTES perteneciente a la Universidad de las Ciencias Informaticas
Instalar
Información
Datos técnicos
Sitio web
http://www.uci.cu
Categoría
Transcargo
Resumen
Este es un módulo para la instalación y configuración del sistema de gestión integral del agente transitario Transcargo.
Nombre técnico
sigax_install
Licencia
LGPL versión 3
Última versión
10.0.1.0
Permite la instalación y configuración del sistema de gestión integral del agente transitario Transcargo.


Instalador de SIGAX
Este es un módulo para la instalación y configuración del sistema de gestión integral del agente transitario Transcargo.
Instalado


Instalador de SIGAX
Por Centro de desarrollo FORTES perteneciente a la Universidad de las Ciencias Informaticas
Actualizar
Desinstalar
Permite la instalación y configuración del sistema de gestión integral del agente transitario Transcargo.

Requisitos no funcionales

Los requerimientos no funcionales, como su nombre sugieren, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema (Sommerville, 2005).

Listado de los requisitos no funcionales

Usabilidad

RNF1: El tiempo estimado en el proceso de instalación y configuración de SIGAX está previsto para no superar los 20 minutos.

RNF2: Para el proceso de instalación y configuración de los dos instaladores no se necesita de amplios conocimientos de informática.

Interfaz o apariencia externa

RNF3: El módulo de SIGAX debe de presentar una interfaz con fondo de color blanco, letras en negro que indiquen el nombre del módulo, por el centro que fue creado y una breve descripción. Además, debe de tener un botón de color azul en la parte inferior derecha que diga instalar.

Seguridad

RNF4: El instalador solo podrá ser ejecutado por un usuario administrador del ordenador.

Software

RNF5: Disponer del sistema operativo Ubuntu 16.x.x o superior, Linux Mint 18.2.x o superior.

Hardware

RNF6: El sistema necesitará 1 GB de memoria RAM.

RNF7: 2 GB de espacio libre en la partición del disco duro en la que será instalada la aplicación.

2.5 Conclusiones del capítulo

En el presente capitulo se realizó el análisis y diseño del sistema, en el cual se obtuvieron los siguientes resultados:

- La descripción de la propuesta de solución permitió hacer un levantamiento de los requisitos funcionales acorde al objetivo general para dar respuesta a la problemática planteada.
- Con la identificación de los requerimientos funcionales y no funcionales de la propuesta de solución se logró generar los artefactos propuesto según la metodología seleccionada.
- Los artefactos generados permitieron documentar el proceso para un mejor entendimiento.

Capítulo 3: Implementación y validación de la propuesta.

Una vez realizado el flujo de trabajo correspondiente al análisis y diseño, se tiene una visión más general de cómo se implementarán las funcionalidades del sistema. El propósito de este flujo permitirá describir de manera general la implementación de todas las clases y objetos asociados al sistema. Un aspecto de vital importancia en el desarrollo del software son las pruebas debido a que propician la obtención de buenos resultados. Se ejecutan con el objetivo de revisar que el software tenga el nivel de calidad requerido y permiten detectar errores durante su funcionamiento. Este proceso debe comenzar en la fase de requerimientos y terminar con la finalización de la aplicación. En el proceso de pruebas se definen varios métodos, técnicas y tipos de pruebas, las cuales se abordarán durante el desarrollo del presente capítulo.

3.1 Estándares de codificación

Un estándar de codificación son reglas que se siguen para la escritura del código fuente. De tal manera que otros programadores puedan identificar las variables, las funciones o métodos. Se definen estándares de codificación porque un estilo de programación homogéneo en un proyecto para un lenguaje de programación permite que todos los participantes lo puedan entender en menos tiempo y que cualquier persona que se desempeñe como codificador de dicho lenguaje pueda interpretar de manera eficiente (López, 2016).

Durante la implementación de los módulos se emplearon los estándares de codificación que se definieron en el análisis y diseño. A continuación, se exponen detalladamente, estos estándares de codificación:

Directorios: este estándar permite la organización de los directorios por los que está integrado el módulo los cuales son:

- *data/:* demo y datos XML
- *models/:* modelos de definición
- *views/:* contiene las vistas y plantillas
- *static/:* contiene los elementos web, separados en css / js /, img /, lib /.

Nomenclatura de archivos: permite enunciar las clases del modelo y vistas, separando la lógica de negocio por conjuntos de modelos. Facilitando la comprensión de los elementos correspondientes en cada modelo permitiendo crear para cada conjunto denominado <*main_model*> los siguientes archivos:

- *models/<main_model>.py*
- *models/<inherited_main_model>.py*

- *views/<main_model>_templates.xml*
- *views/<main_model>_views.xml*

Ejemplo:

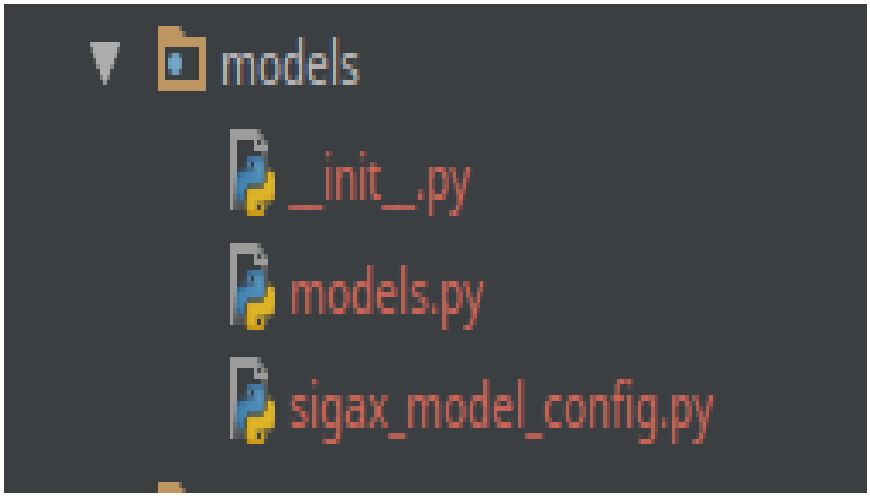


Figura 3: Nomenclador de archivos

Archivos XML

Formato: Se utilizó para la declaración de registros en XML en los cuales el atributo id se escribe antes del modelo y en la declaración de campo el nombre de atributo es el primero.

Ejemplo:

```
<record id="extracciones_cadc_jasper" model="ir.actions.report.xml">
    <field name="string">Reporte de extracciones en el CADC</field>
    <field name="auto">True</field>
    <field name="model">sigax.logistica.reportes.extracciones_cadc</field>
    <field name="jasper_model_id" ref="sigax_logistica.model_sigax_logistica_reportes_extracciones_cadc"/>
    <field name="name">Reporte de extracciones en el CADC</field>
    <field name="report_name">sigax.logistica.template_extracciones_cadc</field>
    <field name="rml">sigax_logistica/report/extracciones en el cadc/extracciones_cadc.jrxml</field>
    <field name="report_rml">sigax_logistica/report/extracciones en el cadc/extracciones_cadc.jrxml</field>
    <field name="menu">True</field>
    <field name="header">False</field>
    <field name="jasper_report">True</field>
    <field name="jasper_output">pdf</field>
</record>

<record id="extracciones_cadc_file" model="ir.actions.report.xml.file">
    <field name="file" type="base64"
           file="sigax_logistica/report/extracciones en el cadc/extracciones_cadc.jrxml"/>
    <field name="filename">extracciones_cadc.jrxml</field>
    <field name="report_id" ref="extracciones_cadc_jasper"></field>
    <field name="default">True</field>
</record>
```

Figura 4: Archivo XML

Import

Se utilizó para importar las librerías tanto internas como externas las cuales se organizan alfabéticamente de arriba hacia abajo y de izquierda a derecha.

Ejemplo:

```
# -*- coding: utf-8 -*-

from odoo import models, fields, api
import os
```

Figura 5: Comando Import

3.2 Diagrama de componentes

Un diagrama de componentes al igual que un sistema de software es dividido en componentes y muestra las dependencias entre ellos. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema (UML, 2018).

En el diagrama de componente de la presente investigación cuenta con 4 paquetes los cuales son el paquete de modelo, de vistas, de data y de interacción con la plataforma de desarrollo Odoo. El paquete de modelo está compuesto por todos los modelos de la propuesta de solución y los paquetes de vista y data por las vistas de cada modelo. El modelo se encarga de guardar todos los datos en el sistema gestor de base de datos y las vistas son realizadas en archivos xml las cuales también son almacenadas en la base de datos. Odoo es el responsable de adquirir la información guardada por el modelo en la base de datos y a su vez de renderizar los archivos xml para convertirlos en vistas.

Este artefacto, generado durante el análisis y diseño, comprende los componentes que se implementaron y la relación entre ellos. A continuación, se presenta el diagrama de componentes del módulo de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO.

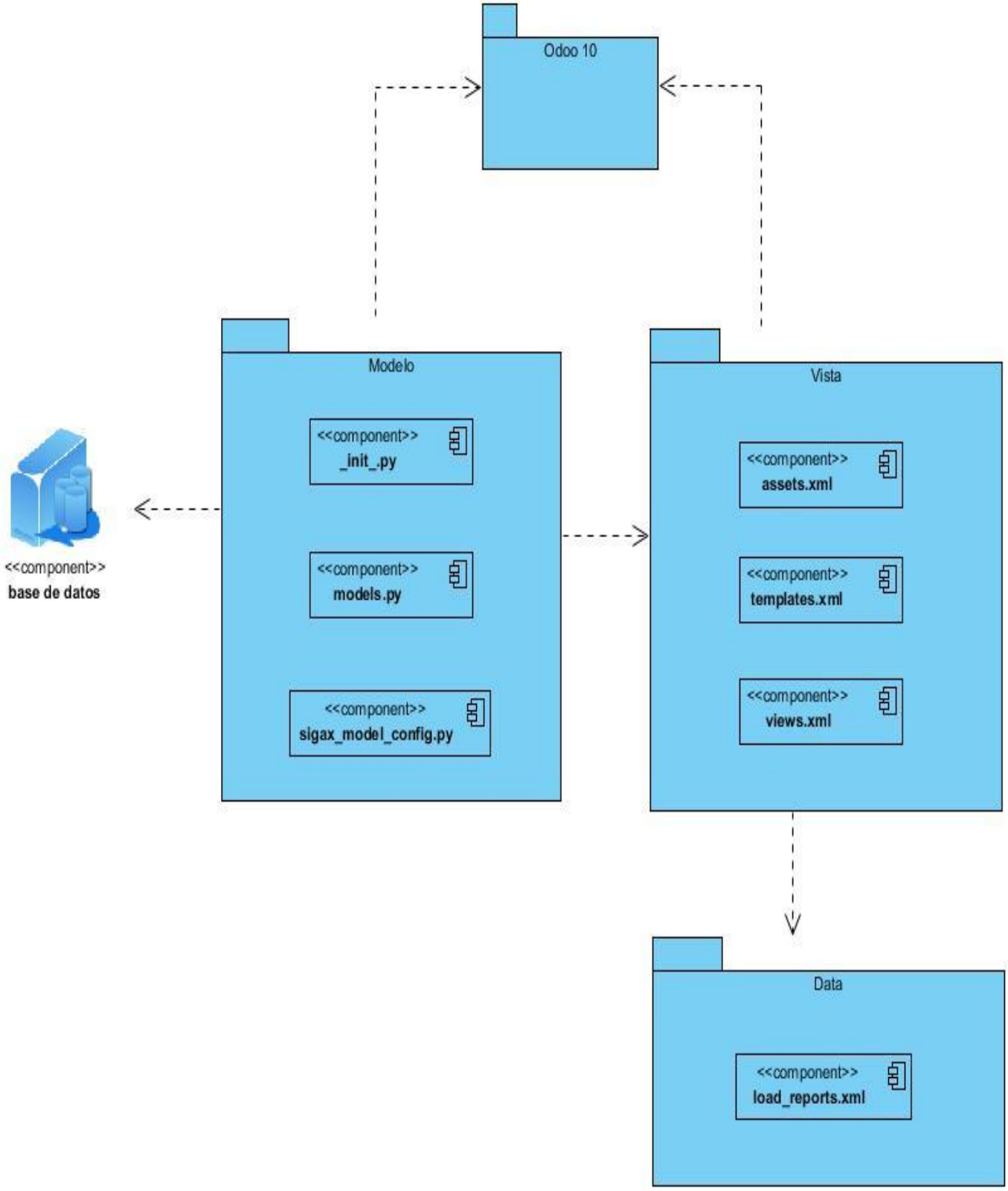


Figura 6: Diagrama de componentes (Fuente: elaboración propia)

3.3 Pruebas de calidad de software

Las pruebas del software, conocidas también como técnicas de evaluación dinámica son un elemento crítico para la garantía de la calidad del sistema, representan una revisión final de las especificaciones del diseño y de la implementación. Su principal objetivo es diseñar pruebas que, sistemáticamente, saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo (PRESSMAN, 1988).

Disciplinas

AUP propone 7 disciplinas (Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno), se decide para el ciclo de vida de los proyectos de la UCI tener 7 disciplinas también, pero a un nivel más atómico que el definido en AUP. En el caso de Prueba se desagrega en 3 disciplinas: Pruebas Internas, de Liberación y Aceptación.

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. Para realizar estas pruebas se utilizó la técnica de Iadov y las pruebas Alfa y Beta.

Prueba Alfa y Beta

Cuando se construye software a medida para un cliente, se lleva a cabo una serie de pruebas de aceptación para permitir que el cliente valide todos los requisitos. La mayoría de los desarrolladores de productos de software llevan a cabo un proceso denominado pruebas alfa y beta para descubrir errores que parezca que sólo el usuario final puede descubrir (Paz, 2016).

- **Prueba alfa:** se lleva a cabo, por un cliente, en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario y registrando los errores y problemas de uso. Las pruebas alfa se llevan a cabo en un entorno controlado.
- **Prueba beta:** se llevan a cabo por los usuarios finales del software en los lugares de trabajo de los clientes. A diferencia de la prueba alfa, el desarrollador no está presente normalmente. Así, la prueba beta es una aplicación en vivo del software en un entorno que no puede ser controlado por el desarrollador. El cliente registra todos los problemas que encuentra durante la prueba beta e informa a intervalos regulares al desarrollador.

A continuación, se muestra un resumen de las no conformidades detectadas en las diferentes iteraciones para este tipo de pruebas:

Figura 7: No conformidades (Fuente: elaboración propia)

Las no conformidades de una iteración se resuelven antes de comenzar las siguientes, por lo que en un tercer momento no se registran dificultades con el uso de la aplicación.

Técnica de Iadov

Iadov es una técnica que evalúa el nivel de satisfacción del usuario, de esta forma se conoce si los componentes realizados cumplen las expectativas de los clientes; está conformada por cinco preguntas: 3 cerradas y 2 abiertas (Kuzmina, 1970).

La técnica de Iadov constituye una vía indirecta para el estudio de la satisfacción, debido a que los criterios que se utilizan se fundamentan en las relaciones que se establecen entre las preguntas cerradas (preguntas 1, 2 y 3) que se intercalan dentro de un cuestionario (ver Anexo 2) y cuya relación el sujeto desconoce. Estas tres preguntas se relacionan a través de lo que se denomina el "Cuadro Lógico de Iadov".

A continuación, se muestra el cuadro lógico de Iadov con las tres preguntas del cuestionario referentes a la satisfacción con la propuesta de solución:

Tabla 4: Cuadro lógico de Iadov

	1. ¿Considera usted que se puede instalar y configurar SIGAX sin una correcta instalación y configuración de la plataforma Odoo?								
	No			No sé			Sí		
	2. ¿Considera que sea necesario el modulo para el proceso de instala y configuración de SIGAX?								
3. ¿ Satisface sus expectativas, como desarrollador, el modulo propuesto?	Sí	No sé	No	Sí	No sé	No	Sí	No sé	No
Me satisfacen mucho	1	2	6	2	2	6	6	6	6
Más satisfecho que Insatisfecho	2	2	3	2	3	3	6	3	6
Me es indiferente	3	3	3	3	3	3	3	3	3
Más insatisfecho que Satisfecho	6	3	6	3	4	4	3	4	4
No me satisfacen	6	6	6	6	4	4	6	4	5
No sé qué decir	2	3	6	3	3	3	6	3	4

El número resultante de la interrelación de las tres preguntas indica la posición de cada encuestado en la escala de satisfacción siguiente:

1. Clara satisfacción

2. Más satisfecho que insatisfecho

3. No definida

4. Más insatisfecho que satisfecho

5. Clara insatisfacción

6. Contradictoria

Para el desarrollo de la técnica se escogió una muestra de 6 encuestados con conocimientos básicos sobre el proceso de instalación y configuración de SIGAX.

Para obtener el índice de satisfacción grupal (ISG) se trabaja con los diferentes niveles de satisfacción que se expresan en una escala numérica que oscila entre +1 y - 1 de la siguiente forma:

Tabla 5: Índice de satisfacción grupal (ISG)

Escala	Nivel de Satisfacción
+1	Máximo de satisfacción
0.5	Más satisfecho que insatisfecho
0	No definido y contradictorio
-0.5	Más insatisfecho que satisfecho
-1	Máxima insatisfacción

Estos valores se emplean para obtener el Índice de Satisfacción Grupal (ISG), como puede observarse en la siguiente ecuación.

$$ISG = \frac{A\,(+1) + B(+0.5) + C(0) + D(-0.5) + E(-1)}{N}$$

Figura 8: Ecuación para obtener el ISG

Donde A, B, C, D y E representan la cantidad de sujetos en cada una de las categorías.

Tabla 6: Nivel de satisfacción

Variables	Nivel de satisfacción	Valores
A	Máximo de satisfacción	3
B	Más satisfecho que insatisfecho	1
C	No definido y contradictorio	1
D	Más insatisfecho que satisfecho	1
E	Máxima insatisfacción	0

El cálculo del ISG arrojó un valor de 0.5, lo que indica satisfacción de usuarios potenciales con respecto al módulo propuesto.

Pruebas de integración

Las pruebas de integración se ocupan de probar las interfaces entre los componentes, las interacciones con distintas partes de un mismo sistema, como el sistema operativo, el sistema de archivos y el hardware, y las interfaces entre varios sistemas (ISTQB, 2010). Las pruebas de integración deben concentrarse exclusivamente en la propia integración. Así, por ejemplo, si están integrando el módulo A con el B, deben concentrarse en probar la comunicación entre ellos, no las funcionalidades de cada módulo individual (ISTQB, 2010).

Las pruebas de integración son una técnica sistemática para construir la arquitectura del software mientras se llevan a cabo pruebas para descubrir errores asociados con la interfaz. El objetivo es tomar los componentes probados de manera individual y construir una estructura de programa que se haya dictado por diseño (Zapata, 2010).

La prueba de integración descendente es un enfoque incremental a la construcción de la arquitectura de software. Los módulos se integran al moverse hacia abajo a través de la jerarquía de control, comenzando con el módulo de control principal (programa principal).

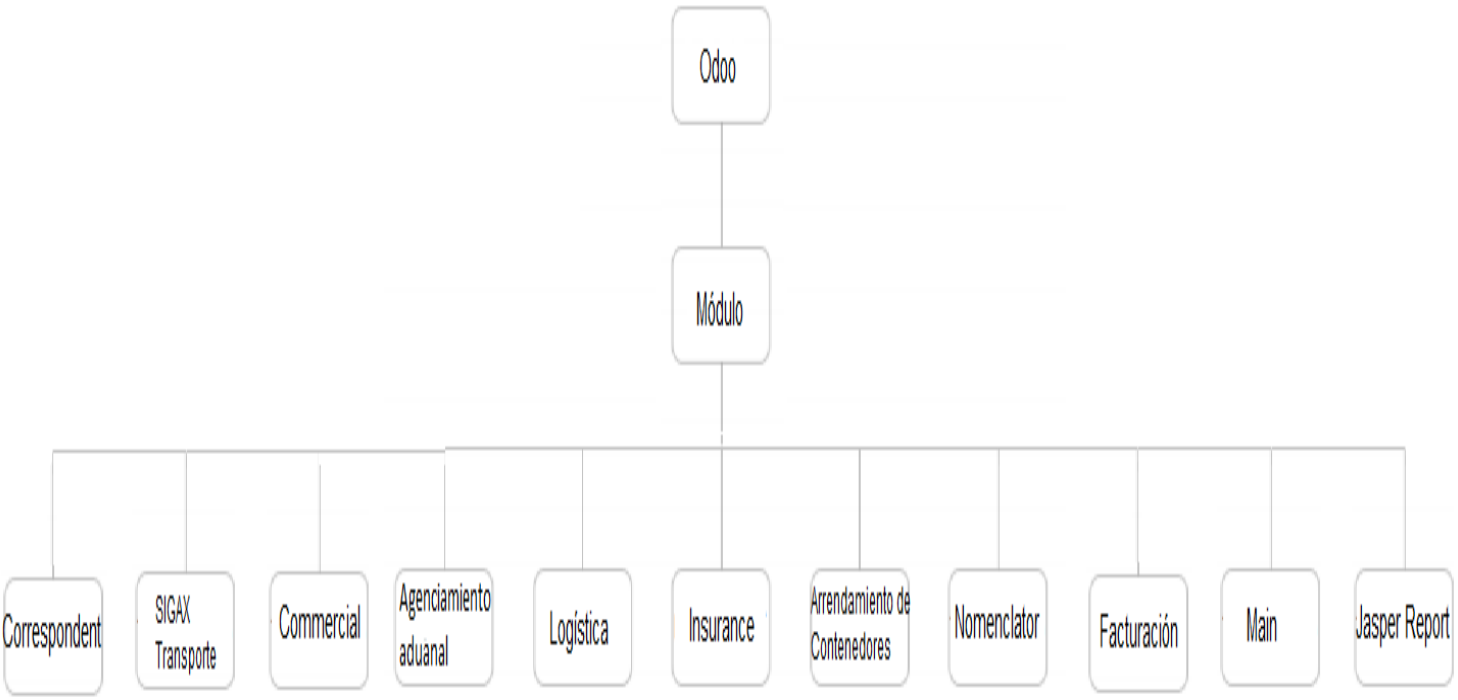


Figura 9: Integración descendente (Fuente: elaboración propia)

La integración primero en anchura incorpora todos los componentes directamente subordinados en cada nivel, y se mueve horizontalmente a través de la estructura. Los módulos Correspondent, SIGAX-Transporte, Commercial, Agenciamiento aduanal, Logística, Insurance, Arrendamiento de Contenedores, Nomenclador, Facturación, Main y Jasper Report se integrarían primero al componente Módulo y este a su vez al siguiente nivel de control Aplicación como muestra la figura 9. Se tiene como resultado que la integración del módulo se logra de una manera adecuada debido a que se trabajó desde una rama del sistema actualizada.

Pruebas a nivel de desarrollador

Este tipo de pruebas son ejecutadas normalmente por el equipo de desarrollo, básicamente consiste en la ejecución de actividades que le permitan verificar al desarrollador que los componentes unitarios están codificados bajo condiciones de robustez, esto es, soportando el ingreso de datos erróneos o inesperados y demostrando así la capacidad de tratar errores de manera controlada. Adicionalmente las pruebas sobre componentes unitarios, suelen denominarse pruebas de módulos o pruebas de clases, siendo la convención definida por el lenguaje de programación la cual influye en el término a utilizar. Es importante que toda la funcionalidad de cada componente unitario sea cubierta, por al menos, dos casos de pruebas, los cuales deben centrarse en probar al menos una funcionalidad positiva y una negativa (Paz, 2016).

Durante el proceso de desarrollo se fueron aplicando este tipo de pruebas garantizando que todas las funcionalidades se probaron al menos una vez.

Pruebas de Usabilidad

En el primer capítulo de la presente investigación se dan a conocer varios atributos con los cuales son medidos la usabilidad. Para dar validez de que dichos atributos están utilizados correctamente se realizaron varias pruebas como son la de juicio de expertos para validar la presentación visual y la facilidad de aprendizaje, técnica de comparación para medir la eficiencia y para medir, en alguna medida, la satisfacción del usuario final se tuvo en cuenta las pruebas descritas en el epígrafe anterior.

Juicio de expertos

El juicio de expertos se define como una opinión informada de personas con trayectoria en el tema, que son reconocidas por otros como expertos cualificados en este, y que pueden dar información, evidencia, juicios y valoraciones. La identificación de las personas que formaran parte del juicio de expertos es una parte critica en este proceso, frente a lo cual Skjong y Wentworht (2000) proponen los siguientes criterios de selección: (a) Experiencia en la realización de juicios y toma de decisiones basada en evidencia o experticia (grados, investigaciones, publicaciones, posición, experiencia y premios entre otras), (b) reputación en la comunidad, (c) disponibilidad y motivación para participar, y (d) imparcialidad y cualidades inherentes como confianza en sí mismo y adaptabilidad. También plantean que los expertos pueden estar relacionados por educación similar, entrenamiento, experiencia (Pérez, 2018).

A continuación, se muestra una tabla con los datos de los expertos seleccionados:

Tabla 7:Relación y roles de los expertos

Nombre	**Apellido**	**Rol**
Celia Indira	Hidalgo Tagle	Analista
Leannys	Rodríguez Moreno	Analista
Adrián	Quesada Gómez	Desarrollador
Daryl	Yturralde López	Desarrollador
Sandy	Guerra Fernández	Administrador de calidad
José Antonio	Falcón de Cárdenas	Desarrollador

Una vez definido los expertos se ubican seis estaciones de trabajo para someter a su evaluación la calidad visual con que se presenta la información en el instalador de SIGAX, así como la facilidad de aprendizaje. Los criterios a tener en cuenta se establecen por los especialistas de acuerdo a sus años de trabajo en el proyecto y la relación con el cliente. Cada uno otorgará un valor entre uno y tres en correspondencia del nivel de presencia del criterio, bajo, medio o alto.

En la siguiente tabla se muestra la evaluación promedio para cada uno de los criterios definidos:

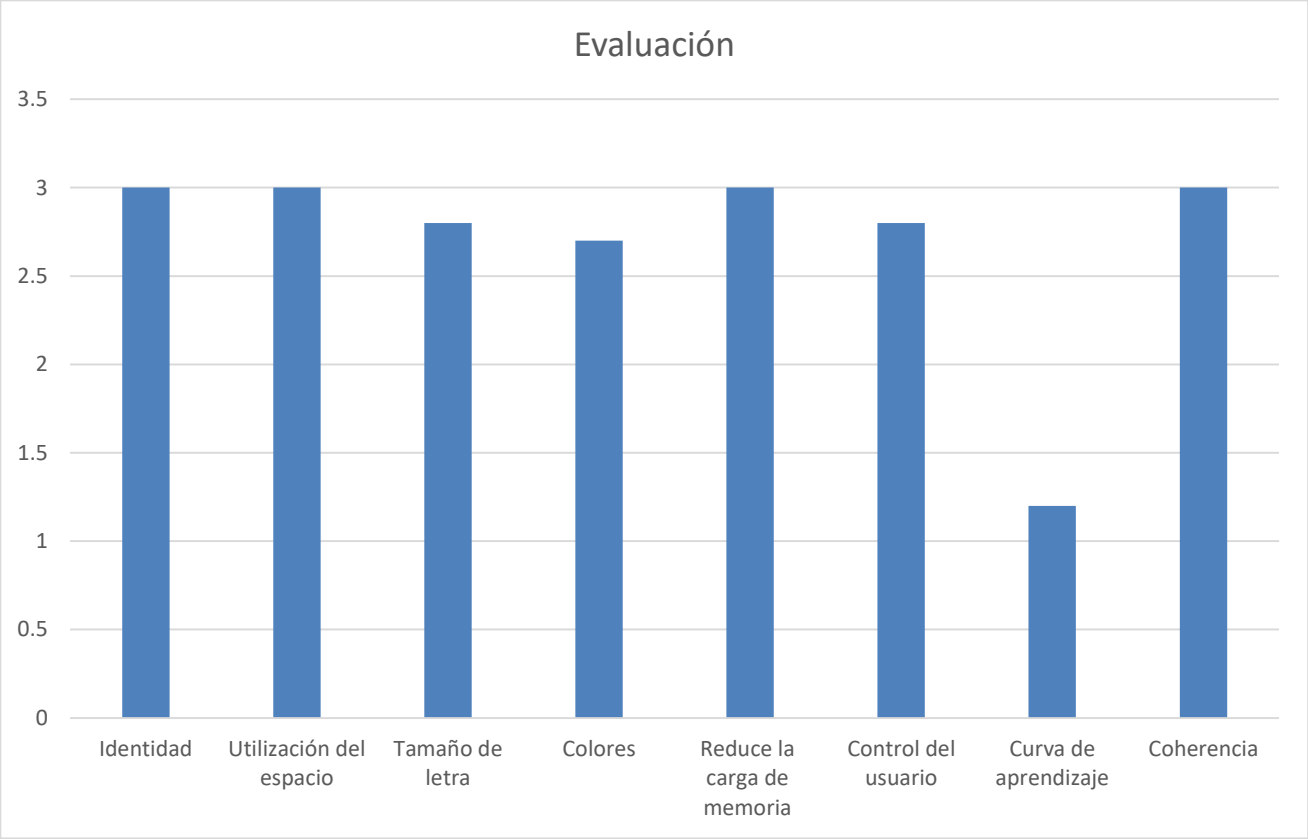


Figura 10: Evaluación de la presentación visual y la facilidad de aprendizaje (Fuente: elaboración propia)

La mayoría de los criterios cuentan con un nivel alto y se considera baja la curva de aprendizaje, evaluando de manera satisfactoria la presentación visual y la facilidad de aprendizaje.

Eficiencia

Para probar la eficiencia, que no es más que la cantidad de transacciones por unidad de tiempo que el usuario puede realizar usando el sistema, se establece una comparación entre un antes y un después del despliegue de la propuesta de solución, que consiste en medir el tiempo en que el usuario se demoraba instalando la aplicación, sin usar el instalador y luego haciendo uso de este.

A continuación, se muestra una tabla donde queda plasmado el tiempo en minuto del proceso de instalación y configuración de SIGAX, manualmente y haciendo uso del instalador propuesto:

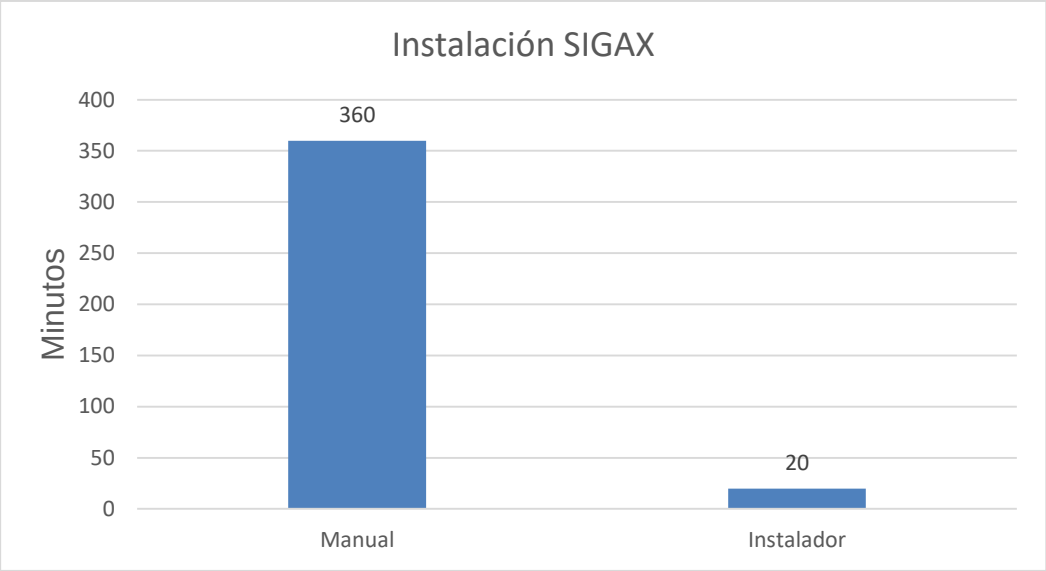


Figura 11: Tiempo para medir la eficiencia (Fuente: elaboración propia)

3.4 Conclusiones del capítulo

En este capítulo se definieron una serie de pautas para la implementación y se realizaron actividades con el objetivo de verificar el correcto funcionamiento del sistema:

- ✓ Para la implementación se tuvieron en cuenta los artefactos generados en el análisis y diseño como son el diagrama de componentes y los estándares de codificación, entre otros.
- ✓ Los métodos de pruebas seleccionados permitieron detectar errores existentes al concluir la etapa de implementación, los cuales fueron corregidos para garantizar la calidad del instalador.
- ✓ Las ejecuciones de los tipos de pruebas definidos permiten comprobar atributos de la usabilidad como la presentación visual, la facilidad de aprendizaje, la eficiencia y la satisfacción.

Conclusiones

Luego de realizado el presente trabajo se arribaron a las siguientes conclusiones:

- El estudio de las principales normas y estándares internacionales de calidad de software permitió definir la usabilidad como una cualidad fundamental para el desarrollo en este tipo de aplicaciones.
- Se desarrolló un módulo que garantizó una mayor usabilidad en la instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO, dotándolo de una interfaz gráfica, disminuyendo considerablemente el tiempo de ejecución del proceso y la curva de aprendizaje, todo lo cual influyó en la satisfacción final del usuario.

Recomendaciones

Para contribuir al enriquecimiento de la presente investigación se realizan las siguientes recomendaciones:

- Se recomienda a la dirección del proyecto utilizar el módulo para el proceso de instalación y configuración del Sistema de Gestión Integral del Agente Transitario TRANSCARGO, con el fin de ayudar a agilizar dicho proceso y disminuir los errores humanos en la configuración del sistema.
- Se recomienda realizar una interfaz gráfica para ofrecer una mayor representación visual del proceso de instalación.

Bibliografía

Abel, P. (2010). Lenguaje Ensamblador y Programación para PC IBM y compatibles.

ALEGSA, L. (s.f.). *Definición de Python (lenguaje de programación)*. Recuperado el 26 de noviembre de 2017, de http://www.alegsa.com

BASH, E. (28 de febrero de 2018). *Bash interiorismo, Seguridad y Distribución*. Obtenido de http://www.bash.cl

BitNami. (12 de 4 de 2014). *The AppStore for Server*. Recuperado el 2 de dicembre de 2017, de http://bitnami.com

Díaz, R. C. (2011). *Desarrollo e Implementación de la versión 2.0 del instalador de Cedrux .* Habana: Universidad de las Ciencias Informáticas.

Fabián, V. M. (2007). *Estudio previo a la implantación de un ERP en una empresa Transitaria.* Barcelona: Universidad autónoma Barcelona.

Headquarters, C. (25 de 5 de 2012). *Desarrollo Web*. Recuperado el 26 de noviembre de 2017, de http://www.desarrolloweb.co

Hernández, A. M. (15 de dic de 2016). Empresa Transcargo.

ISTQB. (2010). *Probador Certificado. Programa de estudio de nivel básico.*

JetBrains. (s.f.). *PyCharm*. Recuperado el 28 de noviembre de 2017, de https://www.jetbrains.com

Kuzmina. (1970). *Metódicas investigativas de la actividad pedagógica.* Leningrado.

Larman, C. (2003). *Modelo de dominio.*

LLC, T. (20 de 2 de 2015). *Sistemas de Información 2*. Recuperado el 25 de noviembre de 2017, de http://sistemasdeinformacion2.wikispace.com

López, L. d. (2016). *Módulo Transporte para el Sistema de Gestión Integral del Agente Transitario.* Habana.

Mantis. (20 de 4 de 2014). *Analisis de aplicaciones : Mantis Bug Tracker : bilib - Centro de Apollo Tecnológico a Emprendedores*. Recuperado el 2 de diciembre de 2017, de http://www.bilib.es

Martínez, C. I. (7 de marzo de 2018). *MVC*. Obtenido de http://http://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r88846.PDF

Mascheroni, C., G., R., P., G., D., & M., E. (1 de 10 de 2010). *Calidad de software e Ingeniería de Usabilidad.* Recuperado el 27 de noviembre de 2017

Mateu. (2012). *Desarrollo Web*. Obtenido de https://cita.campuseuropeodeformacion.com

Odoo. (2016). Obtenido de https://doc.odoo.com/6.0/developer/1_3_oo_architecture/mvc/

Odoo. (1 de 10 de 2016). *Open Source ERP and CRM | Odoo.* Recuperado el 15 de noveimbre de 2017, de http://www.odoo.com

Paz, J. A. (19 de oct de 2016). *Análisis del proceso de pruebas de calidad de software*. Obtenido de http://repository.ucc.edu.co/handle/ucc/962

Pérez, J. E. (3 de mayo de 2018). *VALIDEZ DE CONTENIDO Y JUICIO DE EXPERTOS: UNA APROXIMACIÓN A SU UTILIZACIÓN.* Obtenido de http://www.humanas.unal.edu.co/psicometria/files/7113/8574/5708/Articulo3_Juicio_de_expertos_27-36.pdf

PRESSMAN, R. (1988). *Ingenieria de software. un enfoque práctico.*

RAE. (28 de febrero de 2016). *Diccionario Real Academia de Lengua Española.* Madrid. Obtenido de www.educar.org/diccionari/i.asp

redalyc. (2017). Obtenido de http://www.redalyc.org/pdf/2230/223028547004.pdf

Sánchez, T. R. (2012). *Metodología de Desarrollo para la Actividad productiva de la UCI .* Habana.

scielo. (s.f.). *Evaluación del atributo de Satisfacción*. Obtenido de http://scielo.sld.cu/pdf/rpr/v21n6/rpr07617.pdf

SGDB. (14 de 5 de 2017). *Gestor de Base de Datos*. Recuperado el 29 de noviembre de 2017, de http://blog.powerdata.es

Sommerville, I. (15 de 8 de 2005). *Ingeniería del Software*.

TRANSCARGO. (2016). Obtenido de http://transcargo.transnet.cu

Ubuntu. (28 de febrero de 2018). *Manual de Instalación de aplicaciones en Ubuntu*. Obtenido de http://ayudalinux.wordpress.com

UML. (18 de 2 de 2018). *Component Diagramming Guidelines*. Obtenido de http://www.agilemodeling.com/style/componentDiagram.htm

Velázquez, R. P. (2010). *Instalador Web para el Sistema Integral de Gestión Cedrux.* Habana: Univesidad de las Ciencias Informáticas.

wordpress. (21 de enero de 2013). *Pruebas de software* . Obtenido de http://pruebasdesoftware.wordpress.com

Zapata, M. T. (2010). *Ingeniería del software un Enfoque Práctico* .

Anexos

Anexo 1:

Preguntas de la entrevista

- ¿Qué programas necesita tener instalado una computadora para que el sistema SIGAX pueda funcionar correctamente?
- ¿Qué opinión tienes acerca del proceso de instalación actual?
- ¿Cuál es la vía utilizada para descargar los paquetes necesarios?
- ¿Cuál crees que sería la mejor?
- ¿Cómo crees que debería ser el proceso de instalación del sistema SIGAX?

Anexo 2:

Preguntas para evaluar la aceptación del módulo de instalación y configuración de SIGAX.

1. ¿Considera usted que se puede instalar y configurar SIGAX sin una correcta instalación y configuración de la plataforma Odoo?

   ___ Si

   ___ No

   ___ No se

2. ¿Les resulta atractivo el proceso de instalación y configuración haciendo uso del módulo propuesto?

   ___ Si

   ___ No

   ___ No se

3. ¿Satisface sus expectativas, como desarrollador, el modulo propuesto?

   ___ Me satisfacen mucho
   ___ Más satisfecho que insatisfecho

   ___ Me es indiferente

   ___ Más insatisfecho que satisfecho

   ___ No me satisface

   ___ No sé qué decir

4. ¿Considera con mayor eficiencia el proceso de instalación y configuración de SIGAX haciendo uso del módulo? ¿Por qué?

5. ¿Considera que sea necesario el modulo para el proceso de instalación y configuración de SIGAX? ¿Por qué?