Universidad de las Ciencias Informáticas

Facultad 4

Título: Módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0.

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

Autor:

Rayko Azcue Pérez

Tutor:

Ing. Andris Villalón De la Cruz

La Habana, junio 2018.

"Año 60 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro ser 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 se firma el presente a los ____ días del mes de _____ del año _____.

Rayko Azcue Pérez

_____

Firma del Autor

Andris Villalón De la Cruz

_____

Firma del Tutor

DATOS DE CONTACTO

Autor:

Rayko Azcue Pérez (razcue@estudiantes.uci.cu)

Estudiante asociado al Departamento de Desarrollo de Aplicaciones del Centro de Informática Industrial, perteneciente a la Facultad 4, específicamente al proyecto sobre el Sistema Integral de Perforación de Pozos, ocupa rol de programador.

Tutor:

Ing. Andris Villalón De la Cruz (avillalon@uci.cu)

Graduado de Ingeniero en Ciencias Informática en el año 2013. Se desempeña como Jefe de Proyecto del Sistema Integral de Perforación de Pozos desarrollado en el Departamento Desarrollo de Aplicaciones del Centro de Informática Industrial.

A…

Mis hermanitos Rafe, Omar, Andrés y Josué.

Mi hermanita Thalía.

Mis padres y abuelos.

A…

Mis hermanitos y mi hermanita, debo servir de ejemplo como su hermano mayor, fueron mi mayor fuente de motivación al enfrentarme con el trabajo de diploma.

Mi mamá, la mujer más importante en mi vida, siempre lo serás. El apoyo y el amor de una madre es un remedio infalible para cualquier problema.

Mi papá, por el abanico de valores que me inculcó, por acompañarme en el accidentado camino de la vida junto a la guía de Dios.

Mi papá Omar, aunque no esté presente sé que estaría orgulloso.

Mi abuela, la más tierna y preocupada entre todas las maestras que he tenido, la primera, la que me despertaba cada mañana con el desayuno servido, la que aún lo hace.

Mis abuelos Armando “Checho” y Rafael “Felo”, por enseñarme que un hombre debe tener carácter y actitud positiva frente al trabajo.

Mi tío Oscar, mis tías Laura y Anabel, mis primos Andy y Milton, muchas gracias.

Mi familia, muchas gracias.

Mis otras dos familias, Elizabeth y Cary, Kiki y familia, muchas gracias.

Mis grandes amigos, Raúl, Sergio y Alejandro, por la cantidad de vivencias que hemos compartido, por tantos años.

Liorge, por el tiempo juntos en la universidad, se ha convertido en un hermano para mí.

Los malos tipos, Pablo, Lorenzo, Sergio y Andro, por reírse siempre de mis chistes y alabar mis buenas jugadas en la cancha, ironía, de ser cierto no serían los malos tipos.

Jose Armando “el chiqui”, por su apoyo incondicional y las largas trovas.

A…

Pabel y Yeni, tenerlos ha sido como tener una familia dentro de la universidad.

Osley, Osmany y Brian, muy buenos amigos, muchachos especiales.

Yailyn y Lianet, mi gente del antiguo 5102, mi gente del 5203, muchas gracias.

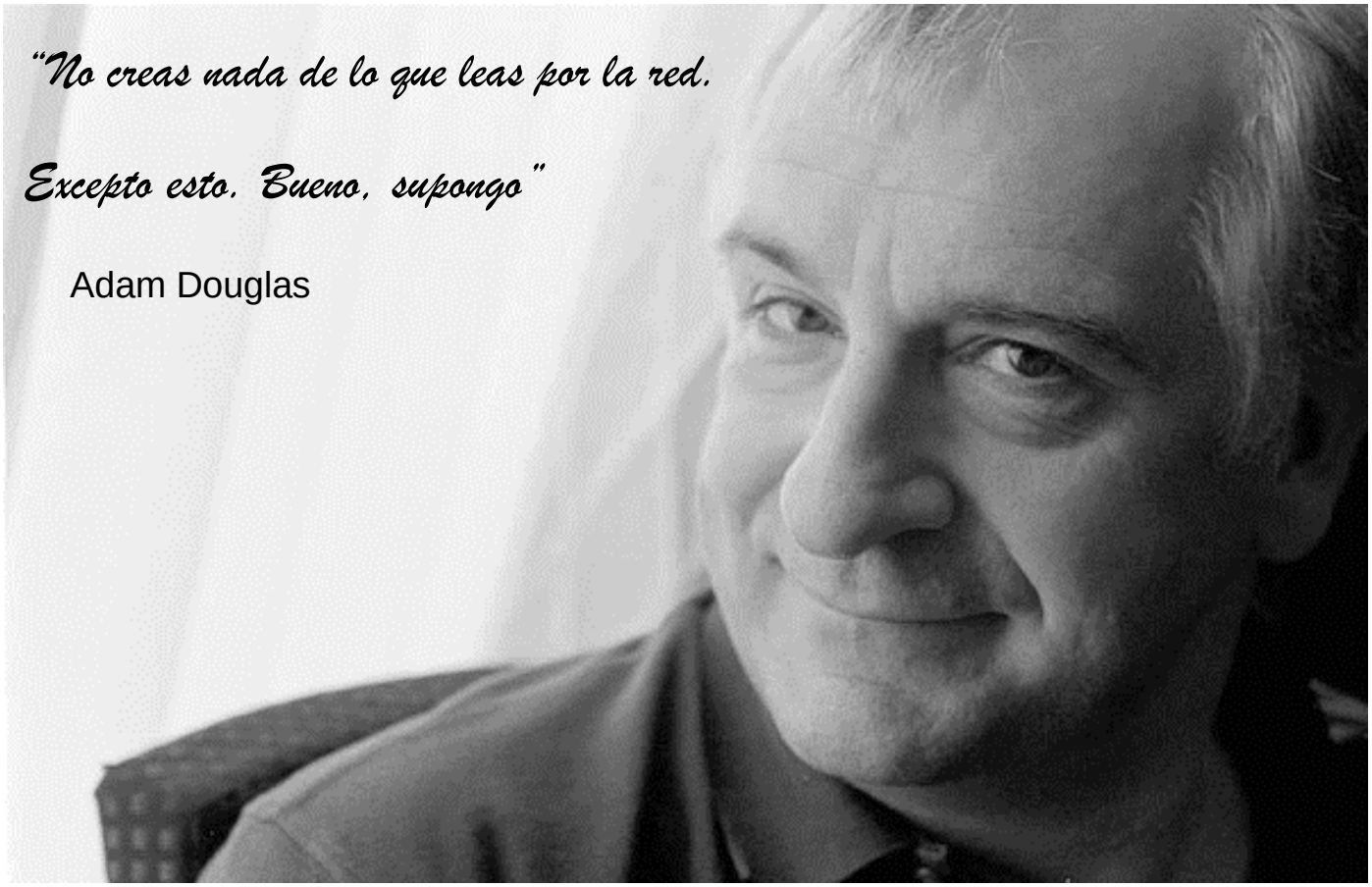
Mi gente del basket, por las largas tardes en la cancha, por ser ustedes, muchas gracias.

Todas las amistades que hice en mi período como estudiante universitario, muchas gracias.

Mi tutor, Andris, por el apoyo y la charla constructiva, por su preocupación, por sus enseñanzas, por ser cómo es, muchas gracias.

Yidian, Lidiexy, Andy, Mayra, Yaself, Olga, Gelson, muchas gracias, a todos los profesores que han contribuido en mi formación como ingeniero y como persona, muchas gracias por sus enseñanzas, por la obra consumada, por su preocupación y su empatía.

Dovier, Ramón, Reinier, Tony, Tomás, Yonny, Alcolea, Bruceta, a todos los viejos amigos y colegas del MPC-TLJ, muchas gracias.


"No creas nada de lo que leas por la red.
Excepto esto. Bueno, supongo"
Adam Douglas

Resumen

El Sistema Integral de Perforación de Pozos ha estado sujeto a cambios desde su concepción, para la nueva versión del mismo fue indispensable la construcción de un módulo para la gestión de las herramientas y tuberías de perforación que se presentará como una mejora considerable en relación con el desplegado, en cuanto a usabilidad del módulo y facilidad del manejo de los inventarios de herramientas y tuberías de perforación dentro del mismo. La presente investigación está enfocada al desarrollo del módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0, para la cual fue utilizada como metodología de desarrollo, la Variación del Proceso Unificado Ágil empleada en la Universidad de las Ciencias Informáticas. El módulo fue construido con tecnologías de software libre y con el objetivo de aumentar la usabilidad y disminuir el tiempo utilizado en la gestión de los inventarios de herramientas y tuberías de perforación, con respecto al módulo que hoy se encuentra en funcionamiento; los objetivos fueron cumplidos satisfactoriamente y el producto fue validado y aceptado por el cliente. El módulo para la gestión de las herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0 desarrollado, permite de forma rápida, intuitiva y eficiente, la gestión de los inventarios de herramientas y tuberías de perforación, por lo que se puede afirmar que el uso del mismo constituye una mejora considerable en cuanto a usabilidad dentro del Sistema.

Palabras Clave:

Gestión, herramientas de perforación, módulo, usabilidad.

Abstract

The Well Drilling Integral System has been subject to changes since its conception, for the new version was essential to build a module for pipes and drilling tools management so that it constitutes a considerable improvement in relation to the deployed, in terms of usability of the module and ease of handling of the tools and pipes inventories within it. This research is focused on the development of the module for pipes and drilling tools management in the Well Drilling Integral System v3.0, for which is used a Variation of the Agile Unified Process used in the University of the Informatics Sciences as the development methodology. The module was built with free software technologies and with the aim of increasing usability and decreasing the time used in the pipes and drilling tools management, with respect to the module that today is in operation; the objectives were met satisfactorily and the product was validated and accepted by the client. The module for pipes and drilling tools management in the Well Drilling Integral System v3.0 developed, allows in a fast, intuitive and efficient way, the management of the tools and pipes inventories, for which can be affirmed that the use of the developed module constitutes a considerable improvement in terms of usability within the System.

Key Words:

Drilling tools, management, module, usability.

INTRODUCCIÓN

Los sistemas de cómputo se han convertido en objetos cotidianos e inseparables para gran parte de la población. Son útiles para la resolución de problemas tanto en esferas productivas como de investigación, debido a que realizan tareas con mayor precisión que un ser humano.

La sociedad cubana se encuentra inmersa en el proceso de informatización, experimentando un aumento del uso las nuevas Tecnologías de la Información y las Comunicaciones (TIC). Como engranajes dentro de este proceso se encuentran los sistemas de gestión, estos permiten optimizar recursos, reducir costos y mejorar la productividad en entornos productivos empresariales.

En una empresa se genera gran cantidad de información referente a los procesos que maneja, la cual, debe ser almacenada y a la vez difundida entre los departamentos, equipos de trabajo y sistemas informáticos que hacen uso de la información generada. El control de procesos de manera automática es un elemento primordial para instituciones u organizaciones, razón por la cual se hace necesaria la utilización de herramientas que contribuyan a garantizar una mayor eficiencia y calidad del proceso productivo.

La sociedad moderna se ha hecho dependiente del petróleo y el gas natural, recursos no renovables que son utilizados como combustible y como materia prima en la industria. Controlar, documentar e informatizar los procesos relacionados con las actividades de perforación y exploración de pozos de petróleo y gas natural son tareas de primer orden para nuestra nación debido, en gran parte, a la necesidad de sustituir importaciones. El gobierno cubano ha propuesto el desarrollo de soluciones nacionales basadas en *software* libre[1], bajo este precepto, se concibe el desarrollo de un sistema que brinde apoyo a la perforación de pozos de petróleo dentro del territorio nacional; un sistema que comprenda la administración de los recursos, unifique los criterios de trabajo y ordene los procedimientos relacionados a la actividad de perforación.

El desarrollo del sistema para la gestión de los procesos referentes a la perforación de pozos de petróleo dentro del territorio nacional, es una tarea asignada al Centro de Informática Industrial (CEDIN) perteneciente a la Universidad de las Ciencias Informáticas (UCI), centro que tiene como objetivo desarrollar productos y servicios informáticos de automatización de procesos con un alto valor agregado y que cumplan

[1] Software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software.

con las necesidades y expectativas de los clientes. Este centro cuenta con varios proyectos encaminados a la informatización de la industria petrolera, siendo uno de estos, el Sistema Integral de Perforación de Pozos (SIPP).

El Sistema Integral de Perforación de Pozos es un sistema de gestión orientado al área de negocios de la perforación de pozos petroleros en Cuba, el mismo brinda cobertura a todo el proceso de perforación de pozos desde el inicio de la perforación hasta su culminación, cubriendo además la etapa de terminación de pozos. Actualmente el Sistema Integral de Perforación de Pozos en su versión 2.0 (SIPP v2.0), cuenta con un módulo para la gestión de herramientas y tuberías de perforación. Aunque el módulo satisface las necesidades básicas del cliente, no brinda una interfaz amigable para la gestión de los inventarios de herramientas y tuberías de perforación, dificultando así el trabajo realizado por parte de los supervisores.

El módulo para la gestión de herramientas y tuberías de perforación de SIPP presenta rigidez en su modelo de datos, aspecto que atenta contra la escalabilidad y dificulta la usabilidad de dicho módulo. Este presenta una arquitectura estática donde los diferentes tipos de herramientas de perforación son presentadas como elementos separados. La interfaz actual es poco intuitiva y el proceso de inventariar las herramientas y tuberías es redundante y trabajoso, razones esenciales por las que se hace necesario mejorar dicha interfaz. Las características que presenta el módulo actual de SIPP para la gestión de herramientas y tuberías de perforación atentan contra la flexibilidad del mismo, dificultando el registro de nuevas herramientas de fondo. Al existir una separación en la gestión de las herramientas se hace mucho más complicado el trabajo con el módulo, provocando que el especialista encargado de registrar dichas herramientas en el sistema emplee más tiempo del necesario.

Como parte del proceso de informatización de la industria cubana, la Unidad Empresarial de Base de Intervención y Perforación de Pozos (UEB-IPP), perteneciente a la Empresa de Producción y Extracción de Petróleo del Centro (EPEP-C), desea continuar con el desarrollo de la versión 3.0 del Sistema Integral de Perforación de Pozos (SIPP v3.0), sumándole nuevas funcionalidades y mejorando las ya existentes; siendo de interés primario la gestión de herramientas y tuberías de perforación debido a la importancia que representa para la empresa contar con la información de todas estas herramientas y tuberías de perforación para su posterior uso, ya que la gestión de dichas herramientas y tuberías sirve como base para la confección de disimiles reportes y procesos. Es necesario que el proceso de gestión de las herramientas y

tuberías de perforación se realice de una manera más eficiente, evitando de esta forma, la pérdida de tiempo y aumentando además la usabilidad del módulo.

A partir de la situación problemática planteada se formula el siguiente **problema de investigación**:

¿Cómo realizar la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0, de manera que permita aumentar la usabilidad y disminuir el tiempo de trabajo?

Se define como **objeto de estudio:** Sistemas de gestión basados en tecnologías web para el apoyo a los procesos de perforación de pozos de petróleo. Como **campo de acción** se concibe: Gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0.

El **objetivo general** de la presente investigación consiste en: Desarrollar un módulo que permita aumentar la usabilidad y disminuir el tiempo de trabajo en la gestión de herramientas y tuberías de perforación en el Sistema Integral de perforación de Pozos v3.0, haciendo uso de tecnologías de *software* libre.

Por lo anteriormente planteado se expone como **idea a defender** que: El desarrollo del módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0, permitirá aumentar la usabilidad y disminuirá el tiempo dedicado al trabajo con el módulo.

Para satisfacer el objetivo general de la presente investigación, se definen las siguientes tareas:

- Estudio de tecnologías web.
- Elaborar el marco teórico conceptual relacionado con los aspectos teóricos que sustentan la investigación.
- Realizar el análisis y diseño del módulo a desarrollar.
- Desarrollar una herramienta informática como soporte al modelo.
- Realizar pruebas funcionales a la herramienta.
- Validar la herramienta propuesta a través de los métodos definidos en la investigación.

Con el objetivo de presentar una solución factible al problema planteado anteriormente es necesario la aplicación de métodos de investigación para entender, verificar, corregir y aplicar los conocimientos adquiridos, logrando así un trabajo investigativo mejor estructurado.

Los Métodos Teóricos permiten estudiar las características del objeto de investigación que no son observables directamente. Entre los Métodos Teóricos a utilizar se encuentra el **Histórico-Lógico**, mediante

el cual se realiza un estudio acerca de los antecedentes y tendencias actuales sobre el desarrollo de componentes para sistemas de gestión basados en la web, así como conceptos y metodologías de desarrollo existentes para llevar a cabo la construcción de estos.

La utilización del método **Analítico-Sintético** aplicado a la bibliografía consultada, facilita el análisis bibliográfico a través de los procesos de búsqueda y síntesis del contenido relacionado al objeto de estudio. Este método permite a su vez definir los requisitos necesarios en el proceso de la gestión de herramientas y tuberías usadas en la perforación de pozos.

Haciendo uso del método de **Modelación** como ayuda para el estudio de nuevas relaciones y cualidades del objeto de estudio mediante diagramas obtenidos por medio de las herramientas de modelado y con el Lenguaje Unificado de Modelado (UML), se logra la obtención de los artefactos generados de todas las etapas por las que transita el desarrollo de *software*: dominio, requerimientos, diseño e implementación.

Entre los Métodos Empíricos o Experimentales se encuentra la **Observación**, mediante la cual se lleva a cabo un registro visual del Sistema Integral de Perforación de Pozos en funcionamiento, permitiendo observar hechos e intentar explicarlos y comprenderlos. Este método posibilita que se creen suposiciones a partir de los datos observados, propiciando la concepción de modelos e ideas asociadas a la investigación.

Tras la culminación de la investigación y como resultado de la misma, se espera la obtención de un módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0.

A continuación, se expone una síntesis de cómo está estructurada la presente investigación.

Capítulo 1: Fundamentación Teórica.

Se definen los conceptos fundamentales para el posterior uso en el desarrollo del trabajo, así como los temas relacionados con el desarrollo de módulos para la gestión de herramientas y tuberías de perforación dentro de sistemas de gestión basados en la web, enfocados a la industria del petróleo y con especificidades relativas a SIPP v3.0. Este capítulo incluye el estudio del estado del arte del tema de la investigación a nivel nacional e internacional, así como las tendencias, técnicas, tecnologías, metodologías y *softwares* utilizados en la actualidad que sirven como apoyo para darle solución al problema planteado**.**

Capítulo 2: Descripción y Análisis.

Se detallan las funcionalidades que debe presentar el sistema, comenzando por la recolección y revisión de los requisitos funcionales y no funcionales, y abarcando las características y restricciones con las que debe cumplir el módulo, con la finalidad de lograr una claridad en la solución que se desea desarrollar. Se realiza el análisis y diseño de la propuesta de solución; presentándose como evidencia del proceso de desarrollo, artefactos como el diagrama de clases del diseño, patrón arquitectónico y de diseños empleados, y el diagrama de entidad-relación como una vista del modelo de datos.

Capítulo 3: Implementación y Prueba.

Se presentan artefactos concernientes a la etapa de implementación, dígase modelo de implementación, diagrama de componentes y modelo de despliegue, describiendo la fase de desarrollo del módulo y a este como componente de SIPP v3.0. Se describe además la estrategia de pruebas diseñadas para comprobar el cumplimiento de los objetivos trazados y el análisis de los resultados obtenidos luego de la aplicación de las mismas.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En el presente capítulo se hace referencia a los principales conceptos asociados al dominio del problema. Se expone el estado del arte sobre el desarrollo de módulos para sistemas de gestión basados en la web y centrados en el campo de acción definido, ubicados en el contexto nacional e internacional, así como las tendencias y soluciones existentes. Como base para la tarea de desarrollo vinculada a la presente investigación, se presentan la metodología de desarrollo, así como las herramientas y tecnologías a utilizar.

1.2. Conceptos asociados al dominio del problema

Para el estudio relacionado al desarrollo de componentes para sistemas de gestión basados en la web, específicamente sobre la gestión de herramientas y tuberías de perforación dentro de SIPP v3.0, es imprescindible el conocimiento de un conjunto de conceptos básicos para la comprensión de la presente investigación. Estos conceptos acerca de herramientas y procesos sirven de cimiento para el desarrollo del módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0.

Es fácil de reconocer un pozo de extracción de petróleo al observar la gran torre de perforación que sobresale del terreno. El término pozo petrolífero alude a cualquier perforación del suelo diseñada con el objetivo de hallar y extraer petróleo o hidrocarburos gaseosos (Silva, 2009).

Según *Marr Varhaug*, editor en jefe de *Schlumberger*[2], el proceso de perforación transcurre de la siguiente manera: el departamento de perforación planifica una trayectoria que maximice la exposición del pozo a las zonas productivas y diseña los arreglos de fondo de pozo para lograr ese recorrido. Los ingenieros preparan un plan detallado para cada etapa del proceso de perforación. Esta *prognosis*[3] de perforación designa una localización en la superficie y la profundidad total del pozo, especifica el tamaño de la barrena, las densidades anticipadas del lodo y los programas de entubación necesarios para alcanzar la profundidad

[2] Líder mundial de tecnologías para la caracterización de yacimientos, perforación, producción y procesamiento para la industria del petróleo y el gas.

[3] Conocimiento anticipado de algún hecho.

total. En lo que respecta a los pozos desviados, la *prognosis* establece la localización del fondo del pozo, la profundidad de deflexión inicial y el acimut[4] para el punto de comienzo de la desviación. La *prognosis* sirve además como base para presupuestar y obtener la autorización para las erogaciones requeridas por la perforación (Varhaug, 2012).

A grandes rasgos se tiene una torre de perforación que contiene todo el equipamiento necesario para bombear el fluido de perforación, bajar y elevar la línea, controlar las presiones bajo tierra, extraer las rocas del fluido, y generar la energía necesaria para la operación.

1.3. Análisis del objeto de estudio

En el presente trabajo de diploma se ha definido como **objeto de estudio**: sistemas de gestión basados en tecnologías web para el apoyo a los procesos de perforación de pozos de petróleo.

1.3.1. Descripción general del objeto de estudio

Es común que una empresa hoy en día maneje un gran flujo de información, posea un extenso abanico de procesos internos y cuente con varios centros distanciados geográficamente, para las mismas es factible usar un sistema que garantice la correcta gestión de la infraestructura empresarial.

Un *software* de gestión basado en la web es un sistema de información que permite que cualquier gestión sea organizada por internet. Este sistema presenta muchas ventajas frente a los demás, pues su fácil acceso y practicidad lo convierten en el mejor vehículo para transportar la información crucial dentro de una empresa (Ponchar, 2013), entre estas ventajas encontramos las siguientes:

- Se puede acceder al sistema en cualquier hora, momento y lugar, sólo basta con tener acceso a la red y un navegador instalado.
- Todos los usuarios que se encuentren autorizados a acceder a determinada información pueden hacerlo, siempre y cuando posean las credenciales que se requieren para demostrar que está autorizado.

[4] La dirección magnética de un levantamiento direccional o del pozo, como la planificada o medida con un levantamiento direccional. El acimut se expresa generalmente en grados con respecto al polo norte geográfico o magnético.

- Se ahorran costos, ya que este sistema no genera mayor inversión, y por el contrario son pocos los gastos que se derivan de su función.
- Toda la información es almacenada en una base central a la que se tiene acceso por medio de internet, así no será necesario instalar un *software* adicional en los puestos de trabajo y esta base de datos puede ser accedida a su vez desde cualquier localización bajo las normas establecidas.
- La seguridad de la información no está sujeta al personal, ya que esta no se guardará en computadores, ni en dispositivos de almacenamiento personales, ahora toda la información estará almacenada en un servidor donde se garantiza la seguridad de la misma.

Es necesario, antes de continuar con la presente investigación, exponer los conceptos referentes a las herramientas de perforación. La sarta de perforación es un conjunto de componentes metálicos armados secuencialmente que conforman el ensamblaje de fondo y la tubería de perforación, esta se encarga de transmitir las fuerzas de empuje y rotación al *tricono*[5], además de determinar la profundidad del pozo.

El BHA es un componente de la sarta de perforación y está integrado por el conjunto de todas las herramientas entre la broca (barrena) y la tubería de perforación. Tiene como funciones proporcionar el peso requerido sobre la broca para maximizar la tasa de penetración, producir hoyos de calibre, evitar la formación de desviaciones y minimizar vibraciones de la sarta de perforación. Está compuesto por la barra de perforación (*Drill Collar*), la tubería pesada (*Heavy Weight*), estabilizadores y accesorios (Fernández, y otros, 2015).

La barra de perforación o vástago de perforación es una barra de acero larga, cuadrada o hexagonal, con un orificio perforado en el centro para proveer un trayecto de fluido. Se utiliza para transmitir el movimiento rotativo desde la mesa rotativa[6] o el buje del vástago[7] a la sarta de perforación, a la vez de permitir bajar o subir la sarta de perforación durante la rotación (Schlumberger, 2018).

[5] Herramienta de perforación usada para triturar la roca.

[6] La sección giratoria o rotativa del piso de perforación que proporciona la potencia para hacer girar la sarta de perforación.

[7] Un adaptador que sirve para conectar la mesa rotativa al vástago de perforación.

La tubería pesada constituye el componente intermedio del ensamblaje de fondo. Es un tubular de espesor de pared gruesa, cuya conexión posee las mismas dimensiones que las de la tubería de perforación para facilitar su manejo, pero es ligeramente más larga. La función más importante de la tubería pesada es servir de zona de transición entre las barras y la tubería de perforación, para minimizar los cambios de rigidez entre los componentes de la sarta, con el objeto de reducir las fallas originadas por la concentración de flexión cíclica en la conexión de la tubería de perforación (NeitamPetrol, 2018).

La barrena de perforación es la herramienta utilizada para triturar o cortar la roca. Todo lo que se encuentra en un equipo de perforación asiste directa o indirectamente a la barrena para la trituración o el corte de la roca. La barrena se encuentra en la parte inferior de la sarta de perforación y debe cambiarse cuando se desgasta excesivamente y deja de avanzar. La mayoría de las barrenas funcionan raspando o triturando la roca, o ambas acciones a la vez, generalmente como parte de un movimiento de rotación. Algunas barrenas, denominadas barrenas de tipo martillo, martillan la roca verticalmente en forma similar a un martillo neumático utilizado en operaciones de construcción (Schlumberger, 2018).

Un motor de desplazamiento positivo es un tipo de motor de fondo de pozo utilizado en el campo petrolero para accionar la barrena de perforación u otras herramientas de fondo de pozo durante las actividades de perforación direccional o perforación de alto rendimiento. A medida que se bombea a través del motor de desplazamiento positivo, el fluido de perforación convierte la potencia hidráulica del fluido en potencia mecánica para hacer rotar la barrena (Schlumberger, 2018).

El martillo de fondo o *jar* (denominación del inglés para vibración o choque) es un dispositivo mecánico utilizado para generar una carga de impacto sobre otro componente de fondo de pozo, especialmente cuando ese componente se encuentra atascado. El principio es similar al de un carpintero que utiliza un martillo. La energía cinética se almacena en el martillo a medida que éste se balancea, y luego es liberada de manera repentina en el clavo y la tabla, cuando el martillo golpea el clavo (Schlumberger, 2018).

1.3.2. Antecedentes y concepción de SIPP

El grupo empresarial Cuba Petróleo (CUPET) se encuentra al mando de todas las actividades relacionadas a la industria del petróleo y el gas en Cuba, las cuales van desde la exploración de yacimientos hasta la refinación del crudo, así como de satisfacer al mercado nacional. CUPET tiene como institución subordinada

a la Empresa de Perforación y Reparaciones Capitales de Pozos de Petróleo y Gas (EMPERCAP), la cual presta servicios a la Unidad Empresarial de Base de Intervención y Perforación de Pozos (UEB-IPP).

La UEB-IPP se encuentra ubicada en el municipio Varadero perteneciente a la provincia de Matanzas, es la encargada de velar sobre las actividades de perforación de todos los pozos terrestres en el territorio cubano, gestionando los contratos a compañías de servicio que trabajan en los pozos, las cuales brindan servicios de Direccionales, de Lodo, de Perforación, de Registro, de Geología, entre otros; a partir de los servicios se generan órdenes de trabajo, facturas o reportes que deberá firmar el supervisor de la UEB-IPP que atiende el pozo para su aprobación.

En la UEB-IPP se confeccionaban una serie de informes, partes y reportes manualmente utilizando un archivo de Excel con las informaciones que se recibían diariamente a las ocho de la mañana por correo electrónico provenientes de los pozos petroleros en perforación y del Centro de Investigaciones del Petróleo (CEINPET), en caso de no existir la conexión se emitían por vía telefónica. Estos documentos generados contenían información de la perforación diaria, donde se incluían datos de la cantidad de combustible, barrenas, herramientas, camisas, cantidad de dinero invertido y productos químicos utilizados en los pozos, así como las operaciones y actividades que se realizarían en el día.

Debido a la imperiosa necesidad de informatizar los procesos referentes a la perforación de pozos de petróleo e hidrocarburos gaseosos dentro de territorio nacional, la Unidad Empresarial de Base de Intervención y Perforación de Pozos plantea la tarea de desarrollar un sistema para la gestión y control de la información generada en todo el proceso de perforación de pozos. Esta tarea es aceptada por el CEDIN en el año 2008, respondiendo a las necesidades de la UEB-IPP.

El SIPP es concebido como una herramienta para gestionar una serie de datos relacionados con la perforación de pozos realizados por la UEB-IPP en conjunto con compañías extranjeras que operan y prestan servicios a la industria del crudo en Cuba, pilar fundamental para la economía del país. Este sistema presenta una arquitectura modular y está basado en el producto *MaximusDrillPro*, solución inicial desplegada para satisfacer las necesidades la UEB-IPP.

1.4. Análisis de las soluciones similares existentes

En el siguiente epígrafe se realiza un estudio de soluciones similares existentes, tanto dentro del ámbito nacional como internacional, escogidas por sus características comunes con SIPP o porque manejan procesos similares a los usados en la gestión de herramientas y tuberías de perforación.

1.4.1. Soluciones en la esfera nacional

Dentro del ámbito nacional, se han construido dos sistemas para la gestión y el control de las actividades de perforación en pozos de petróleo, ambas desarrolladas por el CEDIN a pedido de la UEB-IPP y que son la base para SIPP v3.0.

MaximusDrillPro

Con vista a satisfacer las necesidades de la UEB-IPP, la primera solución informática para la gestión y control de las actividades de perforación en pozos de petróleo fue *MaximusDrillPro*, la cual es una previa aproximación de SIPP. *MaximusDrillPro* provee de una interfaz poco amigable para el usuario; este fue desarrollado con tecnologías elementales para el desarrollo web y prescindiendo de otras que permitieran mejorar el producto, propiciando que el sistema concebido fuera de difícil manejo dado su bajo nivel de funcionalidad y usabilidad. En este sistema todos los datos debían ser ingresados carácter a carácter en plantillas enormes, esto provocaba errores debido al tiempo y capacidad de la conexión. Esta solución hoy en día no es utilizada, ya que, con el desarrollo de SIPP, se volvió obsoleta.

SIPP v2

La versión actual de SIPP fue desplegada en tres pozos a lo largo del territorio nacional. SIPP automatiza los procesos de gestión y control de información, resultado del proceso de perforación en los pozos de petróleo, obteniendo como resultado reportes de estado de la perforación del pozo, dirección del pozo (inclinometría), reportes de estado de las barrenas, bombas, herramientas, costos, lodo, cementación, encamisado, entre otros. Posee un modelo de base de datos robusto, con alto rendimiento y con altos volúmenes de peticiones, que responde al proceso de manera efectiva, posibilitando manejar todos los datos necesarios para el apoyo a la toma de decisiones. En este sistema, se maneja de forma separada las tuberías con las que cuenta CUPET y las herramientas para la perforación que existen en cada pozo, las cuales pertenecen a la compañía contratada para el proceso de perforación, estas se manejan dentro del sistema desde los ámbitos DIPP y Pozo respectivamente.

1.4.2. Soluciones en la esfera internacional

A nivel internacional hay disímiles aplicaciones desarrolladas sobre la gestión y control de las actividades relacionadas con las distintas fases de procesamiento y extracción del crudo, las que en su mayoría se enfocan en brindar una plataforma para la creación de los reportes generados en base a las operaciones realizadas en los pozos en perforación.

Well Logger™

Well Logger™ es un *software* desarrollado por *Porpoise Media*[8] que permite, mediante el uso de una sencilla interfaz, crear registros de perforación del suelo y diagramas de construcción de pozos. Su interfaz de usuario, presenta diseños personalizables, patrones de relleno definibles por el usuario, escalamiento ajustable y vista previa de impresión; esta fue diseñada con el objetivo de que consultores ambientales y geotécnicos creen registros de perforación de suelo elaborados con calidad. La información de entrada incluye litología de la perforación, toma de muestras, construcción del pozo, detalles anexos de la perforación y la información general acerca del proyecto y la perforación. La licencia de este *software* tiene un valor $299, esta provee al comprador de soporte vía correo electrónico y actualizaciones gratuitas por un año (PorpoiseMedia, 2018).

STRATER

Strater es un programa de *Golden Software*[9] el cual facilita la comprensión de los registros y modelos manejados en los pozos, siendo una gran herramienta para la toma de decisiones y promoviendo la productividad en los trabajos de perforación. Este permite compartir de forma segura, reportes en formatos PDF[10] o TIFF[11], así como ingresar diagramas en herramientas de presentación como *Microsoft Word* o *PowerPoint* de la manera más simple. Consta de una interfaz dinámica que cuenta con múltiples vistas de

[8] Grupo de desarrollo de software, activo desde 1996, produce software enfocados en el estudio geológico.

[9] Proveedor de software para graficación.

[10] Formato de archivo que proporciona una imagen electrónica de texto y objetos gráficos, parecida a un documento impreso.

[11] Formato común para intercambiar imágenes de mapa de bits entre aplicaciones.

perforación y donde se puede crear modelos y visualizar los pozos, mapas y secciones transversales, artefactos que brindan gran cantidad de información de la perforación y del subsuelo. Con *Strater* se pueden visualizar gráficamente detalles de la construcción de pozos, contabilidad de impactos, datos de ensayos, profundidad de la perforación, concentración de contaminantes, humedad, entre otros datos. *Golden Software* tiene tres licencias de comercialización de *Strater*, una para estudiantes con un precio de $100, una para universidades con un precio de $229 y otra general con un precio de $449 (GoldenSoftware, 2018).

WELLVIEW

WellView, producto insignia de *Peloton*[12], es un sistema completo de administración de datos de pozos con licencia privativa que permite a las empresas de petróleo y gas administrar sus datos a lo largo del ciclo de vida del pozo. Con sus poderosos esquemas, informes claros y herramientas robustas de análisis, el *software WellView* pone la información en las manos de las personas que la necesitan, en el momento que la necesitan. Posee herramientas para el apoyo del análisis de la *prognosis* geológica, la planificación del pozo, el manejo de reportes, operaciones de perforación y geología, así como intervenciones de pozos. Está integrado dentro del sistema *MasterView*® el cual cuenta con otras herramientas para el manejo de reportes de los BHAs, la programación de plataformas de perforación, gestión de contratos y reportes corporativos, construcción de instalaciones, monitoreo y análisis ingenieril, y otros procesos. (Peloton, 2018)

Resultados de la investigación

Durante la investigación realizada pudo apreciarse que las aplicaciones *Well Logger*™, *Strater* y *WellView* presentan un gran abanico de funcionalidades en común con SIPP. Estas aplicaciones, aunque tienen gran competitividad a nivel mundial y han sido probadas por empresas con experiencia y prestigio en los procesos concernientes a la industria petrolera, cuentan con licencias de *software* comercial, lo que representa un costo adquisitivo para el país, es imprescindible destacar que estas son aplicaciones de escritorio que difieren en los requerimientos para su despliegue en relación con SIPP, el cual es una aplicación web.

Luego del estudio de las soluciones *Well Logger*™, *Strater* y *WellView*, y con el objetivo de desarrollar el módulo para la gestión de las herramientas de fondo para SIPP v3.0, fueron tomadas ideas de estas

[12] *Peloton* es una empresa líder en el desarrollo de sistemas de información de construcción y operaciones de pozos.

aplicaciones en correspondencia con las características que poseen como sistemas para la gestión de las actividades de perforación y extracción de crudo. Las principales características tomadas en consideración se presentan a continuación:

- Los *softwares Well Logger*™, *Strater* y WellView cuentan con interfaces diseñadas para ser simples y funcionales, enfocadas a minimizar el tiempo empleado en la gestión de los procesos manejados desde las mismas.
- Estos sistemas presentan bases de datos relacionadas con la información referente a todos los aspectos del ciclo de vida del pozo, contando con inventarios para las herramientas y tuberías de perforación.

1.5. Definición de las tecnologías de desarrollo

Para lograr el desarrollo del módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0, es necesario contar con conocimientos de las tecnologías a las que se recurrirá.

A excepción de la metodología de desarrollo que fue seleccionada por los buenos resultados obtenidos con su aplicación en proyectos de la Universidad, del lenguaje de modelado que fue seleccionado por su uso difundido, de la herramienta de modelado que fue seleccionada por las facilidades que brinda en cuanto a las funcionalidades para la modelación y del editor de código que fue seleccionado por su comprensible utilización y el enfoque del mismo hacia desarrollo de aplicaciones web, las tecnologías restantes fueron escogidas previamente por el Centro de Informática Industrial para el desarrollo de SIPP v3.0

Escogidas cautelosamente con el fin de lograr la integración con la nueva versión del producto y por las características que presentan, se describen a continuación las herramientas y tecnologías utilizadas.

1.5.1. Metodología de desarrollo

Las metodologías de desarrollo consisten en una colección de documentación formal referente a los procesos, las políticas y los procedimientos que intervienen en el desarrollo del *software*, constituyendo una guía para el mismo. Estas se traducen en enfoques de carácter estructurado y estratégico que permiten el desarrollo de programas, teniendo como base a modelos de sistemas, reglas, sugerencias de diseño y guías. La finalidad de una metodología de desarrollo es garantizar el cumplimiento de los requisitos iniciales (eficacia) y minimizar las pérdidas de tiempo (eficiencia) en el proceso de desarrollo de *software*. En los

últimos años se han desarrollado dos corrientes fundamentales: las metodologías ágiles y las tradicionales. La diferencia fundamental entre ambos tipos de metodologías consiste en que las tradicionales pretenden lograr sus objetivos mediante el orden y la documentación, mientras que las ágiles intentan mejorar la calidad del *software* mediante la comunicación directa e inmediata entre las personas que intervienen en el proceso (OBS Business School, 2016).

Variación AUP-UCI

El Proceso Unificado Ágil (AUP, siglas de *Agile Unified Process*), se presenta como una versión simplificada de RUP (Torrecilla, 2012). AUP describe una aproximación al desarrollo de aplicaciones que combina conceptos propios del proceso unificado tradicional con técnicas ágiles, con el objetivo de mejorar la productividad. Tiene la ventaja de ser un proceso ágil que incluye explícitamente actividades y artefactos a los que la mayoría de desarrolladores están acostumbrados. En AUP se establecen cuatro fases que transcurren de manera consecutiva:

1. Inicialización: el objetivo de esta fase es obtener una comprensión común entre el cliente y el equipo de desarrollo en cuanto al alcance del nuevo sistema y la proposición de un conjunto de arquitecturas para el mismo.
2. Elaboración: el objetivo es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema, así como validar la arquitectura.
3. Construcción: durante esta fase el sistema es desarrollado y probado en el ambiente de desarrollo.
4. Transición: el sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación, para finalmente ser desplegado en los sistemas de producción.

De forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI, la propia Universidad decide hacer una variación de la metodología AUP bajo el nombre AUP-UCI, la cual busca erradicar los problemas detectados en los proyectos de la Universidad. La UCI establece aplicar AUP-UCI en todos los proyectos de desarrollo de *software* que se desarrollan en ella. El SIPP se ajusta a esta metodología, la misma cubre todas las particularidades del proyecto, por lo tanto, la presente investigación se acoge al uso del Proceso Unificado Ágil variación UCI como metodología desarrollo de *software*.

1.5.2. Lenguaje de modelado

El uso del modelado de software es más valorado mientras mayor es la complejidad de la solución informática que se esté desarrollando. Los modelos ayudan al desarrollador a visualizar el sistema a construir, mientras tanto, los modelos que presentan un nivel de abstracción mayor pueden utilizarse para la comunicación con el cliente.

El lenguaje de modelado de objetos es un conjunto estandarizado de símbolos utilizados para modelar un diseño de *software* orientado a objetos. Con la utilización del lenguaje de modelado se puede llegar a obtener diseños sólidos que sirvan como guía a los analistas, desarrolladores y clientes. Logrando unificar la visión y proyección del equipo de desarrollo hacia un producto de calidad (Craig, 2008).

Lenguaje Unificado de Modelado 2.0

UML (Lenguaje de Modelado Unificado, UML por sus siglas en inglés) es el lenguaje de modelado de sistemas de *software* más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema, ofreciendo un estándar para describirlo. Este presenta esquemas para el modelado de base de datos, componentes, procesos del negocio, entre otros (LucidChart, 2018).

1.5.3. Herramienta de modelado de software

Una herramienta CASE (*Computer Aided Software Engineering*, lo que se traduce como Ingeniería de *Software* Asistida por Computadora) supone la automatización del proceso de desarrollo de *software*, permitiendo modelar un sistema previo a su construcción. Contribuye además a mejorar la calidad y la productividad en el desarrollo de sistemas informáticos con el objetivo de permitir la aplicación de metodologías y técnicas de ingeniería de *software* (Krishnamurthy, 2018). Cubre todo el ciclo de vida de un proyecto:

- Concepción y formalización del modelo.
- Construcción de los componentes.
- Transición a los usuarios.
- Certificación de las distintas fases.

Visual Paradigm para UML 8.0

Visual Paradigm para UML es una herramienta que brinda soporte al ciclo de vida del desarrollo de *software*. *Visual Paradigm* entra dentro de esta selección de herramientas y tecnologías como la herramienta CASE a utilizar dentro del desarrollo del módulo descrito en el presente trabajo, ya que es gratuita y factible; esta permite generar todo tipo de diagramas UML, documentación y código desde los propios diagramas, posee múltiples funcionalidades para los procesos y técnicas de la ingeniería de *software*. Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de *software* de forma fiable a través de la utilización de un enfoque orientado a objetos (VisualParadigm, 2018).

1.5.4. Marco de trabajo

Un marco de trabajo o *framework* (por su denominación en el inglés) es una plataforma sobre la cual es posible implementar programas informáticos. Encapsula operaciones complejas en instrucciones sencillas y simplifica el desarrollo mediante la automatización de patrones de diseño utilizados comúnmente. Entre otros usos útiles, un *framework*, proporciona una estructura para el código fuente a nivel de proyecto, brindando un esqueleto para el mismo, por último, facilita la colaboración dentro de un equipo de desarrollo mediante la estandarización (Fabre, 2013).

***Symfony* 2.7**

Symfony es un *framework* de PHP multiplataforma diseñado para desarrolladores web el cual está enfocado a la arquitectura Modelo-Vista-Controlador. Realiza acciones comunes sobre bases de datos, como es el caso de CRUDs (por las siglas de create, read, update y delete, denominaciones del inglés para los vocablos crear, leer, modificar y eliminar), apoyándose en un Mapeo Relacional de Objetos (ORM, por sus siglas en inglés) definido y configurado dentro del proyecto en sí, todo esto abstrayéndose del SGBD utilizado. *Symfony* es distribuido bajo la licencia *MIT*[13] la cual es gratuita y posee una gran comunidad de desarrollo, siendo un *framework* muy difundido y utilizado en la actualidad (Symfony, 2018).

[13] Es una Licencia de software libre permisiva lo que significa que impone muy pocas limitaciones en la reutilización.

Bootstrap 2.3.2

Según lo que afirma Txema Rodríguez[14] en su artículo para el portal de desarrolladores *GenbetaDev*, "*Bootstrap* ofrece una serie de plantillas CSS y ficheros *Javascript* que nos permiten integrar el *framework* de forma sencilla y potente en nuestros proyectos webs. Permite crear interfaces que se adapten a todos los navegadores; se integra perfectamente con las principales bibliotecas de *Javascript*, por ejemplo, *JQuery*; ofrece un diseño sólido usando LESS[15] y estándares como CSS3/HTML5 (Rodríguez, 2018)."

1.5.5. Motor o biblioteca de plantillas

Según Eduard Tomás y Miguel Angel Alvarez[16], "el paradigma de desarrollo del lado del cliente está cambiando y estas bibliotecas son una muestra de ello. Cada vez más en el *front-end*[17] consumimos más lo que se llaman rest-api[18], que ofrecen datos provenientes del *back-end*[19] en formato de texto, generalmente en notación JSON[20]. Pues estas bibliotecas de plantillas lo que ofrecen es pasar rápidamente de esos JSON a pedazos de código HTML que puedes insertar cómodamente en el DOM[21] de la página (Tomás, y otros, 2013).201D

Twig

Twig es una biblioteca y lenguaje de plantilla para el lenguaje de programación PHP que sirve como motor de plantillas para *Symfony* v2. Este provee un sistema encapsulado de trabajo, es flexible y seguro, permite

[14] Líder del equipo de desarrollo para Android en JobAndTalent.com

[15] Lenguaje de Hojas de Estilo que puede ser compilado a CSS

[16] Colaboradores del blog de desarrolladores DesarrolloWeb.com

[17] Se refiere a la programación o tecnologías que corren del lado del navegador web.

[18] Transferencia de Estado Representacional, REST por sus siglas en inglés, es una arquitectura para servicios sobre la web.

[19] Se refiere a la programación o tecnologías funcionando del lado del servidor.

[20] JavaScript Object Notation o Notación de Objetos de JavaScript, es un formato de intercambio de datos ligero.

[21] Modelo de Objetos del Documento, traducción al español de *Document Object Model*, es una interfaz de programación para documentos HTML y XML.

la herencia de plantilla y excepciones, y la separación entre la capa lógica y la de presentación, es un producto de código abierto, a la vez que uno de los motores de plantillas para PHP más veloces y con un mayor conjunto de características (Twig, 2018).

1.5.6. Lenguajes de programación

Un lenguaje de programación es utilizado para describir, con un basamento sintáctico y semántico, un conjunto de símbolos y reglas que son traducidas a instrucciones para ser ejecutadas por una computadora.

PHP 5.6

PHP (siglas de PHP: Hipertext Protocol, el cual es un acrónimo recursivo que se traduce como PHP: Protocolo de Hipertexto) es un lenguaje de programación orientado al desarrollo del lado del servidor para la web con contenido dinámico. Creado en 1994 y tomando su sintaxis en su mayoría de C[22], *Java*[23] y *Perl*[24], es un lenguaje interpretado utilizado en la creación de páginas web dinámicas, embebidas en páginas de Lenguaje de Marcado de Hipertexto (HTML, por sus siglas en inglés) (Van Der Henst, 2018). Otras características de PHP se presentan a continuación:

- Capacidad de conexión con la mayoría de los gestores de base de datos.
- No requiere manejo detallado a bajo nivel, ni es fuertemente tipado[25].
- Su sintaxis es sencilla de aprender.
- Existe una amplia documentación del lenguaje, así como gran cantidad de bibliotecas y *frameworks* para el mismo.

[22] C es un lenguaje de programación de propósito general, conocido como el lenguaje de programación de sistemas.

[23] Java es un lenguaje de programación de propósito general, cuya principal intención es permitir que los desarrolladores escriban el programa una vez y lo ejecuten en cualquier dispositivo.

[24] Perl es un lenguaje de programación de propósito general, es un lenguaje de script lo que significa que no hacen falta ficheros binarios para ejecutar código escrito en el propio lenguaje.

[25] Un lenguaje de programación es fuertemente tipado si no se permiten violaciones de los tipos de datos.

HTML 5

El Lenguaje de Marcado de Hipertextos (HTML, por las siglas de HyperText Markup Language) es el material fundamental para la construcción de páginas web, determinando el contenido de la misma. El término Hipertexto es poco conocido, este hace referencia a los enlaces entre páginas o sitios web, siendo estos vínculos, un aspecto fundamental de la *World Wide Web* (Red Mundial, es reconocible por las siglas www dentro de los navegadores). El término HTML se suele referir a ambas cosas, tanto al tipo de documento como al lenguaje de marca, el cual sirve para colocar etiquetas o marcas en un texto, indicando como debe verse (MDN-Mozilla, 2017).

JavaScript

JavaScript es un lenguaje de script ampliamente difundido en el desarrollo web, interpretado, ligero y ampliamente probado en sistemas y soluciones informáticas. A su vez, es un lenguaje de programación dinámico que soporta la construcción de objetos basado en prototipos, cuya sintaxis es similar a la de *Java* y C++[26]. El poder de *JavaScript* es más visible en el lado *front-end*, agregando mayor interactividad a la web, también es posible utilizar bibliotecas y *frameworks* como es el caso de *jQuery,* una de las múltiples bibliotecas escritas sobre el propio lenguaje (MDN-Mozilla, 2015).

1.5.7. Lenguaje para hojas de estilo

En un inicio, HTML se centraba en el contenido dentro de una página web, definiendo la estructura, la semántica y el aspecto físico de los documentos; sucede que, con la evolución de la *Word Wide Web*, se ha visto cómo el diseño y la presentación cobran un papel protagónico. Las hojas de estilo nacen como un mecanismo que encapsula todas las características de presentación y diseño en documentos, desaconsejando el uso de ciertos elementos relacionados con el aspecto físico dentro de HTML y propiciando que este último se centrara en los aspectos estructural y semántico, que es el principal rol del mismo (W3-ORG, 2018).

[26] C++ es un lenguaje de propósito general basado en C, considerado una superclase del mismo, le aporta disímiles características como la manipulación de objetos.

CSS 3

Hojas de Estilo en Cascada (*Cascading Style Sheets* o CSS, por sus siglas) es el lenguaje utilizado para describir la presentación de disímiles formatos de documentos y posee una especificación estandarizada por parte del W3C[27]. El lenguaje CSS se basa en una serie de reglas que rigen el estilo de los elementos en los documentos estructurados, y que forman la sintaxis de las hojas de estilo, definidas mediante un selector y una declaración (Lapuente, 2013).

1.5.8. Editor de código

Un editor de código fuente posee características diseñadas para simplificar y acelerar la escritura de código fuente de programas informáticos. Permiten la asimilación de compiladores, intérpretes, depuradores, entre otras aplicaciones relevantes en el desarrollo de *software*; normalmente incluyen funcionalidades como resaltado de sintaxis, auto completamiento, eliminación de espacios en blanco, conversión de *tokens*[28] o pareo de llaves, en algunos casos estas y otras funcionalidades pueden ser incluidas como componentes.

Brackets 1.12.0

Brackets es un editor de código multiplataforma, ligero y amigable que cuenta con herramientas visuales y múltiples funcionalidades enfocadas, en su mayoría, al desarrollo web. Es un editor de código abierto escrito en HTML, CSS y *JavaScript*, creado por *Adobe Systems*[29] bajo la licencia MIT y mantenido en la actualidad como un proyecto de *GitHub*[30]; también en *GitHub* es posible encontrar gran cantidad de extensiones para *Brackets*, desarrolladas por la gran comunidad que se ha creado alrededor de este editor (Brackets, 2018).

---

[27] World Wide Web Consortium, consorcio internacional que genera recomendaciones y estándares que aseguran el crecimiento de la World Wide Web a largo plazo.

[28] Cadena de caracteres que tiene un significado coherente en cierto lenguaje de programación.

[29] *Adobe Systems Incorporated* es una empresa de software que Destaca en el mundo del software por sus programas de edición de páginas web, vídeo e imagen digital.

[30] *GitHub* es una plataforma de desarrollo colaborativo que posibilita alo*jar* repositorios de código y brinda herramientas muy útiles para el trabajo en equipo.

Para el desarrollo de la solución descrita en el presente trabajo, es utilizado *Brackets* v.1.12.0 con las siguientes extensiones:

- ***Beautify***, otorga un formato visual para archivos *JavaScript*, HTML y CSS, así como para los respectivos lenguajes.
- ***Brackets Git***, permite la integración de proyectos de *Git*[31] en *Brackets*.
- **PHP *Code Quality Tools***, provee herramientas de análisis de código para PHP.

1.5.9. Mapeo Relacional de Objetos

El mapeo relacional de objetos (ORM, por sus siglas en inglés) permite la traducción de objetos a un formato que pueda ser guardado dentro de una base de datos, sirviendo como una capa intermedia y vinculándose a la base de datos otorgándole persistencia a la información. El mapeo de la información es un proceso laborioso donde se transforma la información proveniente de objetos para el almacenamiento en la base de datos y viceversa, este proceso es automatizado por un ORM y es independiente del gestor de base de datos utilizado, el cual puede ser cambiado si es necesario. El uso de un ORM permite al desarrollador enfocarse en la lógica del negocio, reduciendo la preocupación de la persistencia, beneficio a nivel de programación orientada a objetos que se adquiere al separar las entidades de la persistencia (Block, 2018).

Doctrine 2

Doctrine es un ORM para PHP 5.4 o posterior que brinda transparencia a la persistencia de objetos en PHP, implementando el patrón de mapeo de datos en su núcleo. Algunas de sus principales características son la sencillez de la configuración inicial necesitada para el trabajo con el mismo y el uso de simples anotaciones para especificar la forma en que una entidad (objetos de PHP que contienen variables) es mapeada a una base de datos, características explotada por *Symfony* para la configuración de ciertos componentes y para la definición de sus entidades (Doctrine-Project, 2018).

[31] *Git* es un software de control de versiones, se enfoca en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente.

1.5.10. Sistema de Gestión de Base de Datos

Los sistemas de gestión de bases de datos (SGBD) proporcionan servicios y lenguajes para la creación, configuración y manipulación de una base de datos. Son fiables y ofrecen mecanismos de respaldo para la información contenida en la base de datos, así como un acceso controlado a la misma para su mantenimiento. Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos, disponen de un complejo sistema de permisos a usuarios y grupos de usuarios como mecanismo para asegurar la información, proporcionan una forma eficiente de realizar copias de respaldo de la información almacenada en ellos y cuentan con características que permiten asegurar la integridad, consistencia y control de concurrencia de sus datos (CAVSI, 2018).

PostgreSQL 9.4.1

PostgreSQL es un SGBD altamente escalable y de distribución gratuita, presenta un esquema relacional de objetos, es de código abierto y su desarrollo es dirigido por una comunidad de desarrolladores y organizadores comerciales bajo el nombre Grupo Global de Desarrollo de *PostgreSQL*. Permite la creación de funciones propias, la definición de disparadores, índices, reglas y vistas, entre otras características; además de brindar soporte para extensiones, nuevos índices, datos espaciales y a la minería de datos[32]. Proporciona un soporte al protocolo de comunicación encriptado por SSL[33], así como para los lenguajes PHP, C/C++, *Perl* y *Python*[34] (PostgreSQL-ORG, 2018).

[32] Campo de la estadística y las ciencias de la computación referido al proceso que intenta descubrir patrones en grandes volúmenes de conjuntos de datos.

[33] *Secure Sockets Layer* (SSL) es un protocolo diseñado para permitir que las aplicaciones para transmitir información en ambos sentidos y de manera segura.

[34] Python es un lenguaje de programación interpretado y de prepósito general cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.

1.5.11. Servidor web

Los servidores web, conocidos además como servidores HTTP, son servidores utilizados para la distribución de contenido multimedia en redes internas o en Internet. Un servidor web puede entregar los contenidos simultáneamente a varios ordenadores o navegadores web, donde la cantidad de solicitudes y la velocidad con la que pueden ser procesadas dependen, entre otros aspectos, del hardware, la carga del *host*[35] y la complejidad del contenido, ya que los contenidos dinámicos en la web necesitan más recursos que los contenidos estáticos. Atendiendo a las peticiones hechas, gestiona las páginas web o ejecuta códigos en el servidor para dar respuesta a los clientes, siendo su principal función la de almacenar los archivos de un sitio y emitirlos por la red para poder ser visitado por los usuarios (1&1-DigitalGuide, 2016).

Apache 2

Apache es un servidor de páginas web de código abierto, multiplataforma y modular, desarrollado por la Fundación de Software Apache. Soporta CGI (siglas en inglés de Interfaz de Entrada Común), bases de datos, encriptado por SSL, la creación de *host* virtuales, HTTP, Perl y PHP. Además de estas características, Apache es extensible mediante el uso de módulos que pueden ser cargados al compilar o ejecutar el servidor (Cases, 2018).

1.5.12. Biblioteca de código

Una biblioteca es un conjunto de recursos (algoritmos) prefabricados, que pueden ser utilizados por el programador para realizar determinadas operaciones. Las declaraciones de las funciones utilizadas en estas bibliotecas, junto con algunas macros y constantes predefinidas que facilitan su utilización, se agrupan en ficheros de nombres conocidos que suelen encontrarse en sitios predefinidos. En general, el término biblioteca se utiliza para referirse a un conjunto de módulos agrupados en un solo fichero. Estos ficheros permiten tratar las colecciones de módulos como una sola unidad, y representan una forma muy conveniente para el manejo y desarrollo de aplicaciones (Valerio, 2013).

[35] Host o anfitrión es un término utilizado en informática para referirse a computadoras u otros dispositivos conectados a una red que proveen y utilizan servicios de ella.

jQuery

jQuery es una biblioteca de *JavaScript* con variedad de funcionalidades, versátil, rápida, extensible y que cuenta con una API[36] que funciona en gran multitud de navegadores. Como biblioteca, e incluso el trabajo con su API, es muy sencilla de utilizar, siendo posible obtener excelentes resultados con pocas líneas de código, cuenta con una licencia MIT y es de código abierto. Una de las mayores potencialidades de la biblioteca es un complemento flexible llamado *jQuery* UI, el cual cuenta con un conjunto de funcionalidades como efectos, temas y *widgets*[37], que facilitan el desarrollo de interfaces de usuario altamente interactivas (JQuery, 2018).

1.6. Conclusiones

En el capítulo recién concluido se observó la necesidad pertinente de crear un nuevo módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0, debido a las deficiencias encontradas en la implementación actual y en vista a satisfacer las necesidades de las entidades responsables de las actividades relacionadas con la perforación de pozos en el País. Luego de haberse elaborado el marco teórico se logró un mayor entendimiento de los procesos referentes a las actividades de perforación y a la importancia de los sistemas de gestión, además de una visualización previa de lo que se espera lograr con el desarrollo del módulo.

A raíz del estudio de otros sistemas de apoyo a la gestión de las actividades relacionadas con la industria del petróleo se obtuvo la información necesaria para el desglose de las necesidades de SIPP v3.0 referentes al módulo para la gestión de herramientas y tuberías de perforación. Las herramientas y tecnologías fueron seleccionadas de forma rigurosa, en su mayoría por ser las utilizadas por el equipo al frente del desarrollo de SIPP v3.0, con la finalidad de apoyar el proceso de desarrollo del módulo y la integración del mismo dentro del Sistema.

[36] La interfaz de programación de aplicaciones (API) es un conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro *software* como una capa de abstracción.

[37] Pequeña aplicación o programa, usualmente presentado en ficheros pequeños que son ejecutados por un motor de *widgets*, entre sus objetivos están dar fácil acceso a funciones frecuentemente usadas y proveer de información visual.

CAPÍTULO 2: DESCRIPCIÓN Y ANÁLISIS

2.1. Introducción

En el presente capítulo se describe, desde un punto de vista ingenieril, el desarrollo del módulo para la gestión de herramientas y tuberías de perforación en SIPP v3.0, desde su conceptualización y definición de los requisitos que este debe satisfacer. Se define además el modelo de dominio, las entidades involucradas, así como la arquitectura y los patrones de diseño utilizados. Se presentan las historias de usuarios y los restantes artefactos generados mediante la aplicación de la metodología de desarrollo escogida, describiendo en detalle las características del sistema, partiendo de una buena conceptualización del mismo descrita en el capítulo anterior.

2.2. Modelo de dominio

Un modelo de dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción. Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de *software* o de otro tipo. Similares a los mapas mentales utilizados en el aprendizaje, el modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir (Synergix, 2008).

2.2.1. Descripción del flujo del modelo de dominio

Entre las personas que acceden al Sistema Integral de Perforación de Pozos se encuentran los que poseen el rol de Técnico y otros con el rol de Supervisor, estos ingresan sus credenciales desde la UEB-IPP y el pozo donde trabajan, respectivamente; un Supervisor Jefe cuenta con todos los privilegios de un Supervisor. De ser un técnico, este cuenta con los permisos para gestionar el nomenclador de herramienta/tubería, el nomenclador de herramienta de fondo y el inventario de herramientas/tuberías. En el caso de los supervisores, son los encargados de realizar la gestión del inventario de herramientas de fondo en el pozo donde trabaja.

A continuación, se presenta la descripción gráfica del diagrama de clases del dominio, en este se conciben las relaciones entre las entidades que interactúan en la lógica del negocio, los cuales se detallan con posterioridad.

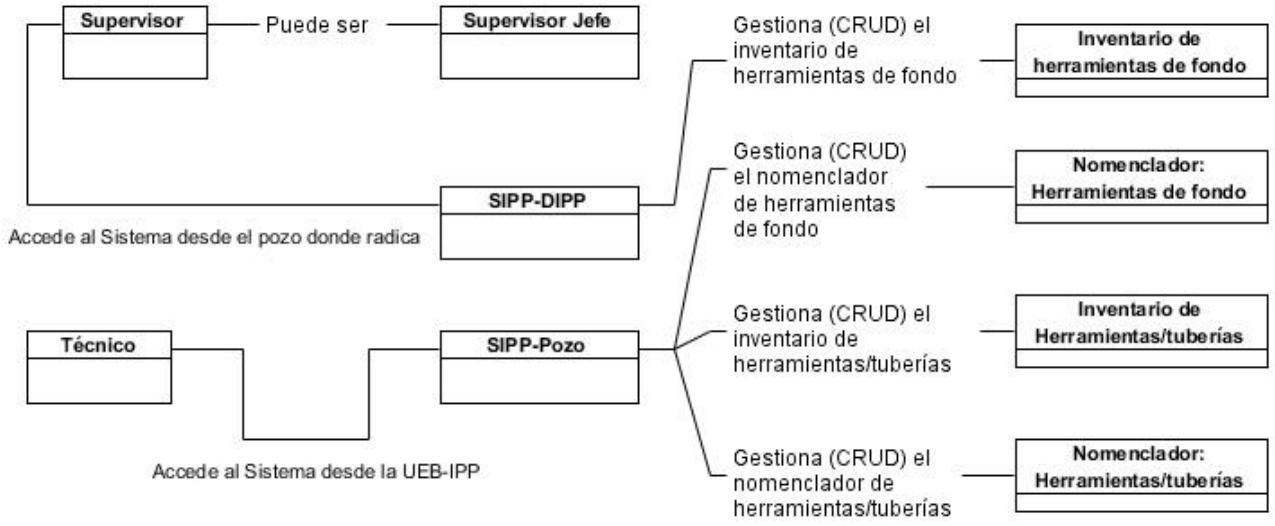


Imagen 1. Modelo de dominio

2.2.2. Descripción de las clases del modelo de dominio

Supervisor: personal del pozo que tiene el rol de Supervisor dentro del Sistema y maneja actividades de gestión y supervisión.

Supervisor Jefe: personal del pozo que tiene el rol de Supervisor Jefe dentro del Sistema, el cual le otorga todos los privilegios de un Supervisor y además puede gestionar los usuarios.

Técnico: personal de la DIPP que tiene entre sus responsabilidades la gestión de los nomencladores de herramientas y tuberías de perforación, además de las tuberías que son manejadas desde la DIPP.

SIPP: Sistema Integral de Perforación de Pozos, este tiene campos y acciones que varían dependiendo si el usuario accede desde la DIPP o desde un Pozo.

POZO: lugar donde se llevan a cabo las actividades de perforación y de extracción de crudo.

DIPP: puesto de mando y control para las actividades de perforación y extracción de crudo.

Inventario de herramientas de fondo: almacena todas las herramientas de fondo.

Inventario de herramientas de fondo: almacena todas las herramientas de tipo tuberías.

Nomenclador: Herramientas de fondo: registro de los tipos de herramientas de fondo.

Nomenclador: Herramientas/tuberías: registro de los tipos de tuberías de perforación.

2.3. Propuesta de solución

Se plantea desarrollar un componente para SIPP v3.0 que permita gestionar de forma rápida y eficiente las herramientas y tuberías de perforación, y que mejore la usabilidad con respecto al implementado actualmente.

El nuevo componente consta de cuatro vistas concernientes al nomenclador para las herramientas de fondo, al nomenclador para las tuberías de perforación, al inventario de herramientas de fondo y al inventario de tuberías de perforación. Desde estas vistas se realiza una gestión completa de la entidad del dominio a la que pertenecen, permitiendo añadir, modificar, listar y eliminar elementos de la misma.

Los nomencladores fueron definidos como vía para definir las clases de herramientas de fondo y tuberías de perforación, estos poseen un campo para establecer tipos o clases de herramientas o tuberías.

En SIPP v2 se gestionaban las herramientas de fondo desde distintas vistas y no se concebía el uso de herramientas de fondo y tuberías de perforación fuera de las clases o tipos definidas en el modelo. Se propone para la solución a desarrollar, la creación de una vista desde la que se gestionen todas las herramientas de fondo, de tal forma que los atributos editables estén en correlación con la clase de herramienta de fondo seleccionada. El componente permitirá definir nuevas clases de herramientas de fondo y tuberías de perforación, definiendo los nombres mediante el uso de los nomencladores de herramientas de fondo y de tuberías de perforación, desde las vistas para la gestión de las herramientas de fondo y de tuberías de perforación, se podrán adicionar herramientas y tuberías de las clases no concebidas inicialmente por el negocio. Las dos características descritas permitirán, luego del desarrollo del módulo, aumentar la usabilidad del sistema y reducir el tiempo empleado en la gestión de los inventarios de herramientas y tuberías de perforación.

Para la implementación del nuevo componente se debe buscar una vía de desechar el tipo de herencia utilizada entre las entidades, ya que esta agrupa toda la información de una jerarquía de clases (padres e hijos) en una única tabla de la base de datos, propiciando que queden huecos relacionados a los atributos no comunes y atentando contra el buen funcionamiento del Sistema debido a que, además, esta tabla tiende a crecer indefinidamente.

2.4. Requisitos

Según la IEEE[38] "los requisitos del proyecto representan una comprensión entre el cliente y el proveedor sobre materias contractuales que pertenecen a la producción de *software* (IEEE, 1998)."

2.4.1. Requisitos Funcionales

Los requisitos funcionales deben definir las acciones fundamentales que deben tener lugar en el *software*, aceptando y procesando las entradas, procesando y generando las salidas (IEEE, 1998). A continuación, se definen mediante una tabla los requisitos funcionales que debe cumplir la solución:

Tabla 1. Descripción de los requisitos funcionales

Nº	Nombre	Descripción	Prioridad	Complejidad
RF1	Crear nomenclador de herramienta de fondo.	El sistema debe permitir crear un nomenclador de herramienta de fondo. Este nomenclador contará con un solo campo que admitirá valores alfanuméricos con una longitud máxima de 50 caracteres.	Alta	Baja
RF2	Modificar nomenclador de herramienta de fondo.	El sistema debe permitir modificar un nomenclador de herramienta de fondo. Este nomenclador contará con un solo campo que admitirá valores alfanuméricos con una longitud máxima de 50 caracteres.	Alta	Baja

[38] *Institute of Electrical and Electronics Engineers* (IEEE), el Instituto de Ingeniería Eléctrica y Electrónica es una asociación mundial de ingenieros dedicada a la estandarización y el desarrollo en áreas técnicas.

RF3	Eliminar nomenclador de herramienta de fondo.	El sistema debe permitir eliminar un nomenclador de herramienta de fondo.	Alta	Baja
RF4	Crear nomenclador de herramienta/ tubería.	El sistema debe permitir crear un nomenclador de herramienta/tubería. Este nomenclador contará con dos campos, Nombre y Nombre corto, que admitirán valores alfanuméricos con una longitud máxima de 250 y 50 caracteres respectivamente.	Alta	Baja
RF5	Modificar nomenclador de herramienta/ tubería.	El sistema debe permitir modificar un nomenclador de herramienta/tubería. Este nomenclador contará con dos campos, Nombre y Nombre corto, que admitirán valores alfanuméricos con una longitud máxima de 250 y 50 caracteres respectivamente.	Alta	Baja
RF6	Eliminar nomenclador de herramienta/ tubería.	El sistema debe permitir eliminar un nomenclador de herramienta/tubería. Si el nomenclador herramienta/tubería se encuentra en uso o fue usado no es posible eliminarlo.	Alta	Baja

RF7	Crear inventario de herramientas de fondo.	El sistema debe permitir crear un inventario de herramientas de fondo. Los campos que se deberán tener en cuenta para crear el inventario son: Herramienta, Fabricante, No. Serie, Modelo, Diámetro exterior, Diámetro interior, Diámetro Máximo, PIN, BOX, Longitud, Descripción, Estado, Cantidad de toberas, IADC, No. Sugerido e Intervalo.	Alta	Alta
RF8	Modificar inventario de herramientas de fondo.	El sistema debe permitir modificar un inventario de herramientas de fondo. Los campos que se deberán tener en cuenta para crear el inventario son: Herramienta, Fabricante, No. Serie, Modelo, Diámetro exterior, Diámetro interior, Diámetro Máximo, PIN, BOX, Longitud, Descripción, Estado, Cantidad de toberas, IADC, No. Sugerido e Intervalo.	Alta	Alta
RF9	Eliminar inventario de herramientas de fondo.	El sistema debe permitir eliminar un inventario de herramientas de fondo. Si la herramienta de fondo que se desea eliminar está en uso o ha sido usada no será posible eliminarla.	Alta	Media

RF10	Crear inventario de herramientas/ tuberías.	El sistema debe permitir crear un inventario de herramientas de tipo tubería. Los campos que se deberán tener en cuenta para crear el inventario son: Herramienta, Diámetro exterior, Diámetro interior, Diámetro Máximo, PIN, BOX, Grado y Descripción.	Alta	Alta
RF11	Modificar inventario de herramientas/ tuberías.	El sistema debe permitir modificar un inventario de herramientas de tipo tubería. Los campos que se deberán tener en cuenta para crear el inventario son: Herramienta, Diámetro exterior, Diámetro interior, Diámetro Máximo, PIN, BOX, Grado y Descripción.	Alta	Alta
RF12	Eliminar inventario de herramientas/ tuberías	El sistema debe permitir eliminar un inventario de herramientas de tipo tuberías. Si la herramienta de tipo tubería que se desea eliminar está en uso o ha sido usada no será posible eliminarla.	Alta	Media

2.4.2. Requisitos no Funcionales

Los requisitos no funcionales son restricciones que afectan a los servicios o funciones del sistema, tales como restricciones de tiempo, sobre el proceso de desarrollo, estándares, etc. (Laguna, 2018). Se puede afirmar que, los requisitos no funcionales, a diferencia de los funcionales, no se basan en las operaciones concernientes a los procesos del negocio, sino que son restricciones que el sistema debe proporcionar.

- **Usabilidad**

RNF1_ Como mecanismo para disminuir el tiempo utilizado en la gestión de los inventarios, el sistema debe permitir las opciones de edición de texto: selección, copiado y pegado de texto.

- **Requisitos de hardware**

RNF2_ Para poder lograr un rendimiento óptimo del sistema, es necesario que las PC clientes posean al menos 256 de RAM.

- **Requisitos de diseño e implementación**

RNF3_ El sistema debe continuar en operación a pesar de la entrada de datos inválidos o fallos en los diferentes componentes que lo conforman, para lo cual debe verificar los campos obligatorios y los patrones definidos por expresiones regulares en los formularios.

RNF4_ En algunos casos el contenido de uno o varios campos de los formularios son válidos si cumplen con unas reglas propias del negocio. El sistema debe hacer uso de estas reglas para validar la integridad de la información ingresada.

- **Seguridad**

RNF5_ La información manejada por el sistema no será pública, de ahí la necesidad de protegerla de usuarios no autorizados, manipulación inadecuada y estados de inconsistencia. Se implementará de forma tal que cada usuario tenga acceso a la información que esté prevista para su rol según las normas de seguridad definidas, los usuarios con rol Supervisor o Técnico son los únicos que tienen acceso a la información sobre las entidades del negocio comprendidas dentro del módulo.

Reglas de usabilidad:

- **Control y libertad del usuario:** La interfaz debe ser diseñada de tal manera que el control de la interacción con el sistema lo tenga el usuario de manera que interactúe directamente con los objetos de la pantalla, haciéndolo más cómodo y no como un módulo más de la aplicación. Esto se consigue cuando el usuario manipula los objetos como si fueran objetos físicos.
- **Consistencia y estándares:** Una buena interfaz contribuye al aumento de la productividad si es consistente en todos los diálogos que desarrolla, basándose en el conocimiento que el usuario ha adquirido con otras aplicaciones y en la aplicación propia. Deberán implementar las mismas reglas de

diseño para mantener la consistencia en toda la interacción. El usuario debe ser capaz de saber en cada momento en qué contexto está trabajando, de donde viene y a donde va.

- **Correspondencia entre el sistema y el mundo real:** El sistema debe hablar el lenguaje de los usuarios, con palabras, frases y conceptos familiares para el usuario, siempre en el contexto de la aplicación. Se debe hacer que la información aparezca en un orden lógico y natural. La aplicación debe interactuar con el usuario de forma que este perciba las palabras y frases cotidianas de la metáfora de la aplicación. La aplicación debe ser lo más parecida posible al objeto del mundo real que representa.
- **Estética y diseño minimalista:** Los diálogos no deben contener información que sea irrelevante para la tarea que está realizando el usuario. Debe ser una interfaz simple, sencilla de aprender y de usar, y con fácil acceso a las funcionalidades que ofrece la aplicación (Alcala, 2007).

2.5. Historias de Usuario

Entre las técnicas ágiles que utiliza AUP-UCI se encuentra el Modelado Ágil, se decide hacer uso de esta técnica para encapsular los requisitos funcionales en **Historias de Usuario**. Una Historia de Usuario describe una funcionalidad que, por si misma, aporta valor al usuario; deben ser breves y con frecuencia se usan tarjetas para la confección de las mismas (Beas, 2011).

Las siguientes tablas hacen alusión a las Historias de Usuario HU10 y HU8, referentes a la creación de una tubería y a la modificación de una herramienta de fondo respectivamente, tareas claves dentro del desarrollo de la solución propuesta. En el portafolios de la solución se encuentran las Historias de Usuario generadas en la presente investigación.

Historia de Usuario HU10: Crear inventario de herramientas/tuberías

Tabla 2. Historia de Usuario HU10: Crear inventario de herramientas/tuberías

Número: 10	**Nombre del requisito:** Crear inventario de herramientas/tuberías.
Programador: Rayko Azcue	**Iteración Asignada:** 1

Prioridad: Alta	**Tiempo Estimado:** *60 horas*
Riesgo en Desarrollo:	**Tiempo Real:** *84 horas*

Descripción:

El sistema debe permitir crear un inventario de herramientas/tuberías.

El sistema debe permitir registrar los datos asociados al inventario de herramientas/tuberías. Siguiendo la ruta en la barra de menú Nomencladores -> Otros -> Herramienta/Tuberías. Se muestra una vista en la cual el usuario debe oprimir la opción “Adicionar”, mostrándose los siguientes campos a registrar:

Tipo de Herramienta: campo seleccionable que toma su valor del nomenclador de herramienta/tubería.

Diámetro Exterior (OD): campo que permite la entrada de valores decimales con una longitud máxima de 15 caracteres.

Diámetro Interior (ID): campo que permite la entrada de valores decimales con una longitud máxima de 15 caracteres.

Diámetro Máximo (OD Max): campo que permite la entrada de valores decimales con una longitud máxima de 15 caracteres.

PIN: campo seleccionable que toma su valor del nomenclador PIN/BOX.

BOX: campo seleccionable que toma su valor del nomenclador PIN/BOX.

Descripción: campo que permite la entrada de texto con una longitud máxima de 255 caracteres.

Grado: campo seleccionable que toma su valor del nomenclador grado.

Si desea volver a la vista principal debe oprimir la opción “Listar”.

Para registrar los datos se debe seleccionar el botón "Guardar y Enviar". Si existe algún campo obligatorio vacío, se muestra un mensaje de error para que se rellenen los

campos, en caso contrario se muestra una ventana emergente mostrando el mensaje de confirmación. Al oprimir el botón “Aceptar” se muestran los datos del inventario de herramientas/tuberías registrado, así como un mensaje de inserción satisfactoria, en caso de oprimir el botón “Cancelar” se aborta el registro. Además se puede acceder a las opciones “Listar”, “Modificar” y “Eliminar”.
Observaciones: El campo Descripción será opcional.
Prototipo elemental de interfaz gráfica de usuario: NO APLICA.

Historia de Usuario HU8: Modificar inventario de herramientas de fondo

Tabla 3. Historia de Usuario HU8: Modificar inventario de herramientas de fondo

Número: 8	**Nombre del requisito:** Modificar inventario de herramientas de fondo.	
Programador: Rayko Azcue		**Iteración Asignada:** 1
Prioridad: Alta		**Tiempo Estimado:** *96 horas*
Riesgo en Desarrollo:		**Tiempo Real:** *108 horas*
Descripción: El sistema debe permitir modificar los datos asociados al inventario de herramientas de fondo. Siguiendo la ruta en la barra de menú Inventarios -> Herramienta de Fondo. Sólo		

es posible desde un pozo. Se muestra una vista con el listado de los inventarios de herramientas de fondo insertados.

Se debe presionar clic en el ícono de modificar ubicado en la parte derecha del inventario de herramientas de fondo a modificar, mostrándose los siguientes campos:

Herramienta: campo seleccionable que toma su valor del nomenclador de herramienta de fondo.

Fabricante: campo seleccionable que toma su valor del nomenclador de fabricante/barrena.

No. Serie: campo que permite la entrada de valores alfanumérico y los caracteres especiales guion (-) y espacio, su longitud máxima es de 50 caracteres.

Modelo: campo que permite la entrada de valores alfanumérico y los caracteres especiales guion (-) y espacio, su longitud máxima es de 50 caracteres.

Diámetro Exterior (OD): campo que permite la entrada de valores decimales con una longitud máxima de 15 caracteres.

Diámetro Interior (ID): campo que permite la entrada de valores decimales con una longitud máxima de 15 caracteres.

Diámetro Máximo (OD Max): campo que permite la entrada de valores decimales con una longitud máxima de 15 caracteres.

PIN: campo seleccionable que toma su valor del nomenclador PIN/BOX.

BOX: campo seleccionable que toma su valor del nomenclador PIN/BOX.

Longitud: campo que permite la entrada de valores decimales con una longitud máxima de 15 caracteres.

Descripción: campo que permite la entrada de texto con una longitud máxima de 255 caracteres.

Estado: campo seleccionable que toma su valor del nomenclador estado.

Cantidad de toberas: campo que permite la entrada de valores enteros con una longitud máxima de 15 caracteres. **IADC**: campo seleccionable que toma su valor del nomenclador IADC. **Intervalo**: campo seleccionable que toma su valor del nomenclador intervalo. **No. Sugerido**: campo que permite la entrada de caracteres alfanuméricos con una longitud máxima de 50 caracteres. Si desea volver a la vista principal debe oprimir la opción “Listar”. Esta vista debe dar la opción de Eliminar. Para guardar los cambios realizados se debe presionar el botón “Modificar”. Si existe algún campo obligatorio vacío, se muestra un mensaje informando que debe rellenarlo, en caso contrario se muestra una ventana emergente mostrando el mensaje de confirmación. Al oprimir el botón “Aceptar” se actualizará automáticamente el inventario de herramientas de fondo en el listado y se muestra un mensaje de modificación satisfactoria, en caso de oprimir el botón “Cancelar” se aborta la modificación.
Observaciones: Si la herramienta seleccionada es de tipo Barrena los campos a habilitar son: Estado, Fabricante, No. Serie, Modelo o Tipo, Intervalo, cantidad de toberas, IADC, OD Max, OD, PIN, Longitud, No. Sugerido y Descripción. En el caso de la barrena el OD Max y el OD tienen el mismo valor. El campo No. Sugerido se habilitará si la herramienta es de tipo barrena y el estado es usado o usado con desgaste. Cualquier otra herramienta que se seleccione, incluidas las que adicione al nomenclador de herramientas de fondo, tendrá habilitado los campos: Estado, Fabricante, No. Serie, Modelo o Tipo, Intervalo, OD Max, OD, ID, PIN, BOX, Longitud y Descripción.

El campo intervalo debe mostrar por defecto el valor del intervalo o sección en que encuentra el sistema. El campo Descripción será opcional.
Prototipo elemental de interfaz gráfica de usuario: NO APLICA.

2.6. Arquitectura

La arquitectura de software es la estructura del sistema que comprende elementos de *software*, las propiedades visibles externamente de esos elementos y las relaciones entre ellos. Una arquitectura de software puede estar basada en elementos sencillos o componentes prefabricados de mayor tamaño, y se especifica de acuerdo con los diferentes tipos de sistemas. Debe ser escogida de manera que minimice los efectos de cambios futuros en el sistema (Weitzenfeld, 2000).

2.6.1. Patrones de arquitectura

Los patrones arquitectónicos se utilizan para expresar una estructura de organización base o esquema para un *software*. Proporcionando un conjunto de sub-sistemas predefinidos, especificando sus responsabilidades, reglas, directrices que determinan la organización, comunicación, interacción y relaciones entre ellos. Los patrones arquitectónicos heredan muchas de las terminologías y conceptos de los patrones de diseño, pero se centran en proporcionar modelos y métodos re-utilizables específicamente para la arquitectura general de los sistemas de información (Gómez, 2013).

2.6.2. Arquitectura Modelo-Vista-Controlador

El patrón arquitectónico Modelo-Vista-Controlador (MVC) especifica que una aplicación consta de un modelo de datos, de información de presentación y de información de control; el patrón requiere que cada uno de estos elementos esté separado en distintos objetos. El modelo contiene únicamente los datos puros de aplicación; no contiene lógica que describe cómo pueden presentarse los datos a un usuario. La vista presenta al usuario los datos del modelo, conoce cómo acceder a los datos del modelo, pero no el significado de estos datos ni lo qué puede hacer el usuario para manipularlos. El controlador escucha los sucesos

desencadenados por la vista (u otro origen externo) y ejecuta la reacción apropiada a estos sucesos, en la mayoría de los casos, la reacción es llamar a un método del modelo, el resultado de esta acción se reflejará automáticamente en la vista (IBM_Knowlegde_Center, 2018).

La clave fundamental detrás de *Symfony* es su sistema basado de *bundles*[39], estos son paquetes que contienen tanto el código de la aplicación como funcionalidades del *framework* para su correcto funcionamiento y que pueden ser distribuidos con facilidad. Dentro de cada *bundle* encontramos un directorio para los controladores, uno para las entidades que forman parte del modelo y uno para los recursos, en este último se almacena las vistas. Las entidades son definidas haciendo uso de Doctrine, permitiendo que estas sean reflejadas dentro del esquema de la base de datos. Las vistas están construidas haciendo uso de Twig, de tal manera que los archivos físicos referentes a estas están constituidos por instrucciones que modifican la plantilla base para la vista del sistema.

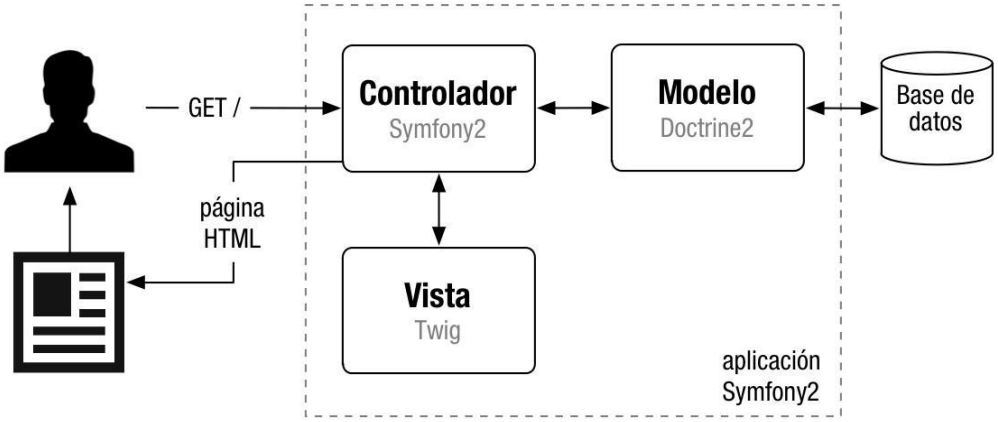


Imagen 2. Patrón arquitectónico Modelo-Vista-Controlador y su relación con Symfony

2.6.3. Patrones de diseño

Los patrones de diseño son formas bien conocidas y probadas de resolver problemas de diseño que son recurrentes en el tiempo. Los patrones de diseño son ampliamente utilizados en las disciplinas creativas y

[39] Conjunto estructurado de archivos dentro de un directorio que implementan una característica única.

técnicas. Los patrones sintetizan la tradición y experiencia profesional de diseñadores de software experimentados que han evaluado y demostrado que la solución proporcionada es una buena solución bajo un determinado contexto. El diseñador o desarrollador que conozca diferentes patrones de diseño podrá reutilizar estas soluciones, pudiendo alcanzar un mejor diseño más rápidamente (Vallejo, y otros, 2012).

Los patrones GRASP (*General Responsibility Assignment Software Patterns* o Patrones Generales de Software para Asignar Responsabilidades) describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable (Alcolea, 2012).

Patrones GRASP utilizados en el desarrollo del módulo para la gestión de herramientas y tuberías de perforación en SIPP v3.0:

- **Experto:** Define que la responsabilidad de realizar una labor recae sobre la clase que tiene los atributos involucrados. Una de las principales características de *Symfony* es el uso de entidades como forma de representación del modelo, una entidad es la abstracción de un objeto del negocio y cumple con el patrón **Experto**, ya que es la encargada del manejo de la información concerniente al objeto del negocio asociado.
- **Creador:** Indica quién debe ser el responsable de la creación de una nueva instancia de alguna clase. Su uso en el desarrollo del módulo brinda la posibilidad de mantenimiento, mayor claridad y reutilización del código. Se ve presente en los controladores que crea *Symfony*, en la función ***createAction***, donde se crea una nueva instancia de la entidad correspondiente, en este ámbito pueden desglosarse en la creación de diferentes instancias de objetos.
- **Controlador:** El patrón controlador es el encargado de mane*jar* eventos en el sistema. Este patrón se pone en evidencia a todo lo largo del desarrollo en *Symfony*, las clases que heredan de ***Controller*** permiten la recepción, manejo y el procesamiento de la información proveniente de las vistas, así como la creación de las mismas, haciendo uso del modelo.
- **Alta cohesión:** La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme (Visconti, y otros, 2018). *Symfony* permite asignar responsabilidades con una alta cohesión, por ejemplo, la clase ***DefaultController***

tiene la responsabilidad de definir las acciones para las plantillas y colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades, otorgándole gran flexibilidad al *software*.

- **Bajo acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases y con que recurre a ellas. Bajo acoplamiento significa que una clase no depende de muchas clases (Visconti, y otros, 2018). En *Symfony* todos los controladores heredan de ***Controller***, utilizando un mecanismo de trabajo alrededor de las funcionalidades que presta la misma, las cuales pueden ser usadas o incluso re-implementadas si es necesario.

Los patrones GoF (*Gang of Four* o Grupo de Cuatro) son soluciones técnicas basadas en Programación Orientada a Objetos (POO). Se clasifican en patrones estructurales, creacionales y de comportamiento (Expósito, 2011).

Patrones GoF utilizados en el desarrollo del módulo para la gestión y manipulación de herramientas y tuberías de perforación en SIPP v3.0:

- **Fachada:** Define una clase con una interfaz común a un grupo de componentes o a un conjunto heterogéneo de interfaces, a esa clase le damos el nombre de **Fachada**. Los elementos heterogéneos pueden ser las clases de un paquete, un conjunto de funciones, un esquema o un subsistema (local o remoto) (Craig, 2008). Las clases *types* utilizadas en *Symfony*, implementan el patrón **Fachada**, estas clases sirven como una interfaz media entre las vistas y las clases controladoras, abstrayendo los mecanismos de manejo para las mismas.
- **Observador:** Define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. Las clases que heredan de ***Controller*** en *Symfony* son las encargadas de escuchar las peticiones que provienen de las vistas, en estas se maneja cada petición según el carácter de la misma y son realizadas las acciones concernientes sobre todos los objetos que interactúan en el contexto.
- **Decorador:** Añade funcionalidad a una clase dinámicamente. En un proyecto de *Symfony*, existe una plantilla global, la cual almacena el código HTML que es común para todas las páginas del portal web. En las plantillas ***Twig*** referentes a cada vista sólo se decora esta plantilla global.

2.7. Modelo de datos

Un modelo de datos es un conjunto de herramientas conceptuales para describir datos, sus relaciones, su significado y sus restricciones de consistencia (Aguilar, 2018).

2.7.1. Diagrama de Entidad-Relación

Un diagrama de Entidad-Relación es la representación de un modelo de datos conceptual basado en objetos, denominado así porque supone relaciones entre entidades. Está compuesto por entidades, atributos, relaciones, cardinalidad y llaves (LucidChart, 2018). A continuación, se presenta el diagrama de Entidad-Relación asociado a la solución que se propone.

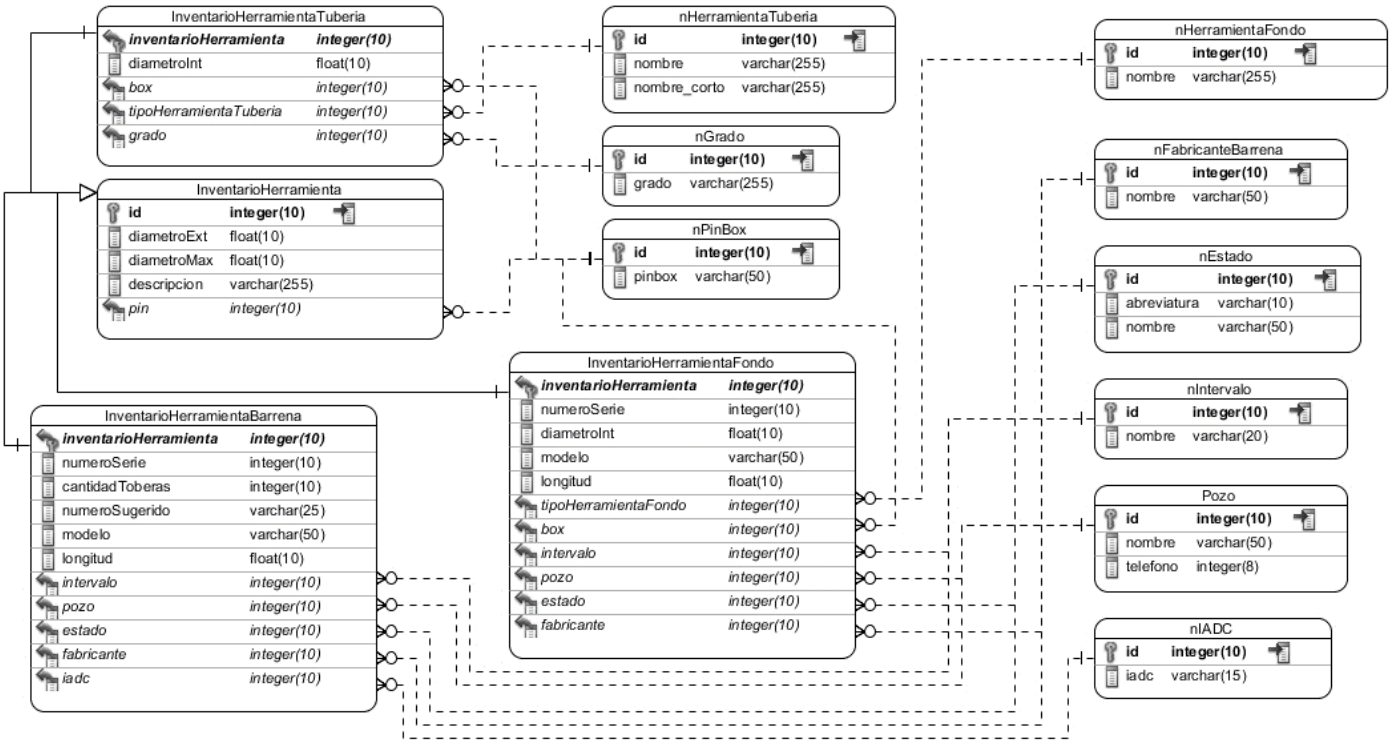


Imagen 3. Diagrama de Entidad-Relación

2.8. Modelo del diseño

El diseño del software es un proceso de muchos pasos pero que se clasifican dentro de uno mismo. En general, la actividad del diseño se refiere al establecimiento de las estructuras de datos, la arquitectura general del *software*, representaciones de interfaz y algoritmos. El proceso de diseño traduce requisitos en

una representación de *software*. El objetivo del diseño es producir un modelo o representación de una entidad que se va a construir posteriormente (Pressman, 1998).

El modelo del diseño es un modelo de objetos que brinda una abstracción de la implementación del sistema, este recoge todos los aspectos relacionados a brindar una solución desde el punto de arquitectura y diseño enfocándose en los requisitos funcionales, no funcionales y otras restricciones concernientes al entorno de dominio. La importancia del diseño recae en la conjunción de la estructura del sistema y la descripción detallada de la solución a los requisitos del mismo, utilizada como entrada para la fase de implementación.

2.8.1. Diagrama de Clases del Diseño

El diagrama de Clases del Diseño es utilizado como una descripción gráfica donde se especifican las clases e interfaces dentro del software desarrollado. Este contiene información sobre las clases, atributos, asociaciones, métodos, constructores y dependencias, aspectos íntimamente relacionados con la implementación. A continuación, se presenta una imagen que muestra el diagrama de Clases del Diseño asociado al módulo para la gestión de herramientas y tuberías de perforación en SIPP v3.0.

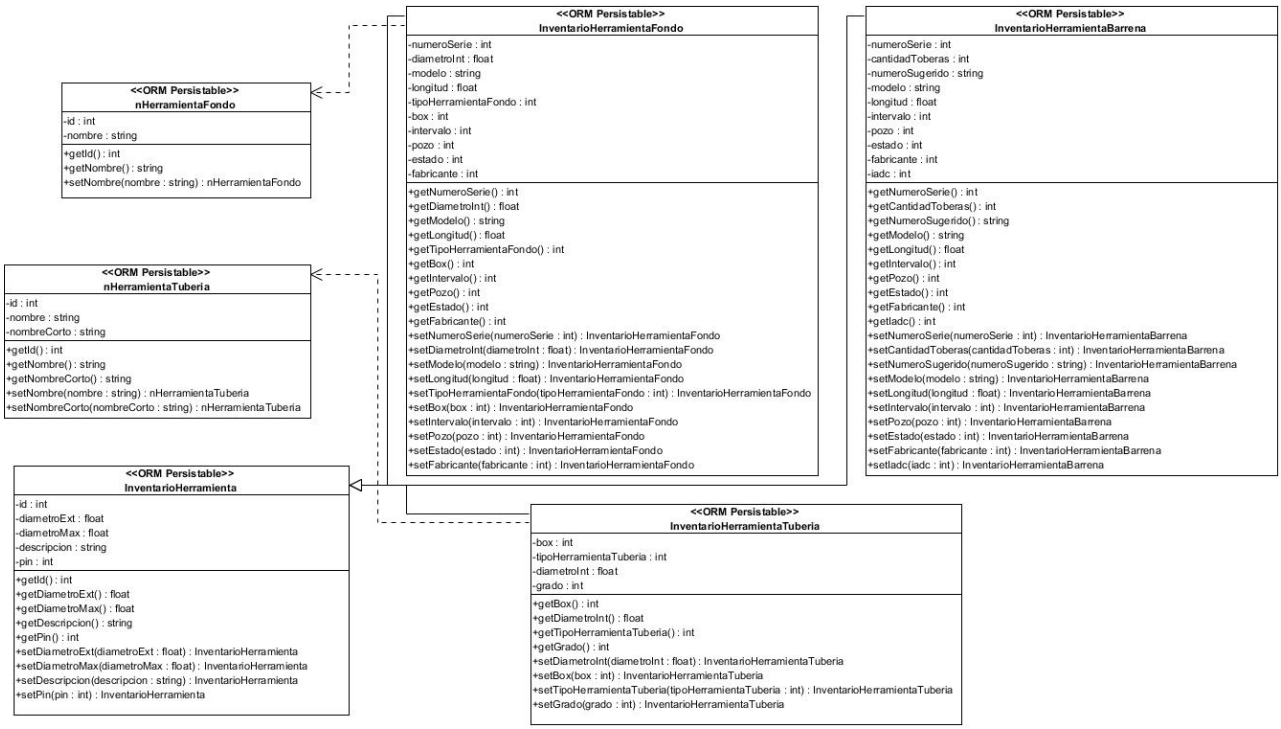


Imagen 4. Diagrama de Clases del Diseño

2.9. Conclusiones

En el presente capítulo fueron tratados temas referentes al análisis y diseño del módulo para la gestión de herramientas y tuberías de perforación en SIPP v3.0, obteniéndose una propuesta de desarrollo del mismo que garantice el cumplimiento de los requisitos del cliente. Se presentó una descripción detallada de los requisitos funcionales y no funcionales, lo cual, en conjunto con el modelo del dominio y las historias de usuario también presentadas, permitió definir de manera clara las necesidades y restricciones que debe cumplir el módulo a desarrollar.

Como parte de la etapa de diseño fueron presentados el patrón arquitectónico y los de diseño a utilizar, elementos que, bien identificados e implementados, proporcionan robustez, escalabilidad y flexibilidad a la solución. La concepción del diagrama de clases del diseño y del diagrama de entidad-relación permitió tener una visión concreta de lo que se desea implementar a nivel de las entidades que interactúan dentro del módulo.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

3.1. Introducción

Tras la culminación de las fases de análisis y diseño, se presenta en este capítulo las particularidades de la implementación del módulo y la validación del cumplimiento de los requisitos del cliente mediante pruebas al software, siguiendo las pautas que propone la metodología de desarrollo escogida. En cuanto a implementación, se describe todo el flujo mediante el modelo de implementación que abre paso al desarrollo de los componentes que se acoplan para lograr una solución fiable, así como la organización de estos dentro del modelo de despliegue. Para garantizar la calidad del software se realizan pruebas funcionales al mismo.

3.2. Modelo de implementación

El Modelo de Implementación está comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a los subsistemas y componentes físicos (Hernández, 2013).

Se presenta a continuación el diagrama de implementación referente al módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0.

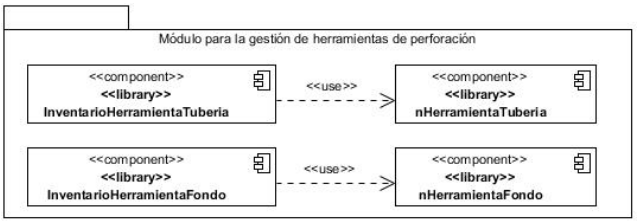


Imagen 5. Diagrama de implementación

3.2.1. Diagrama de componentes

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en un modelo de diseño. Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean estos componentes de código fuente, binarios o ejecutables (Jacobson, 2000).

Según lo planteado, un componente es un objeto físico que representa la abstracción de una o más clases, con sus atributos, métodos y relaciones. Un diagrama de componentes muestra la organización de los componentes dentro del sistema, o de una parte de este, proporcionando una vista arquitectónica del mismo y brindando un punto de apoyo para la implementación. En estos diagramas son utilizados estereotipos para especificar su tipo, como son <<*executable*>> para los programas que pueden ser ejecutados, <<*file*>> para los ficheros de datos o de código fuente, <<*table*>> para las tablas dentro de bases de datos, o <<*library*>> para las bibliotecas estáticas o dinámicas. A continuación, se presenta el diagrama de componentes correspondiente al módulo para la gestión de herramientas y tuberías de perforación en SIPP v3.0.

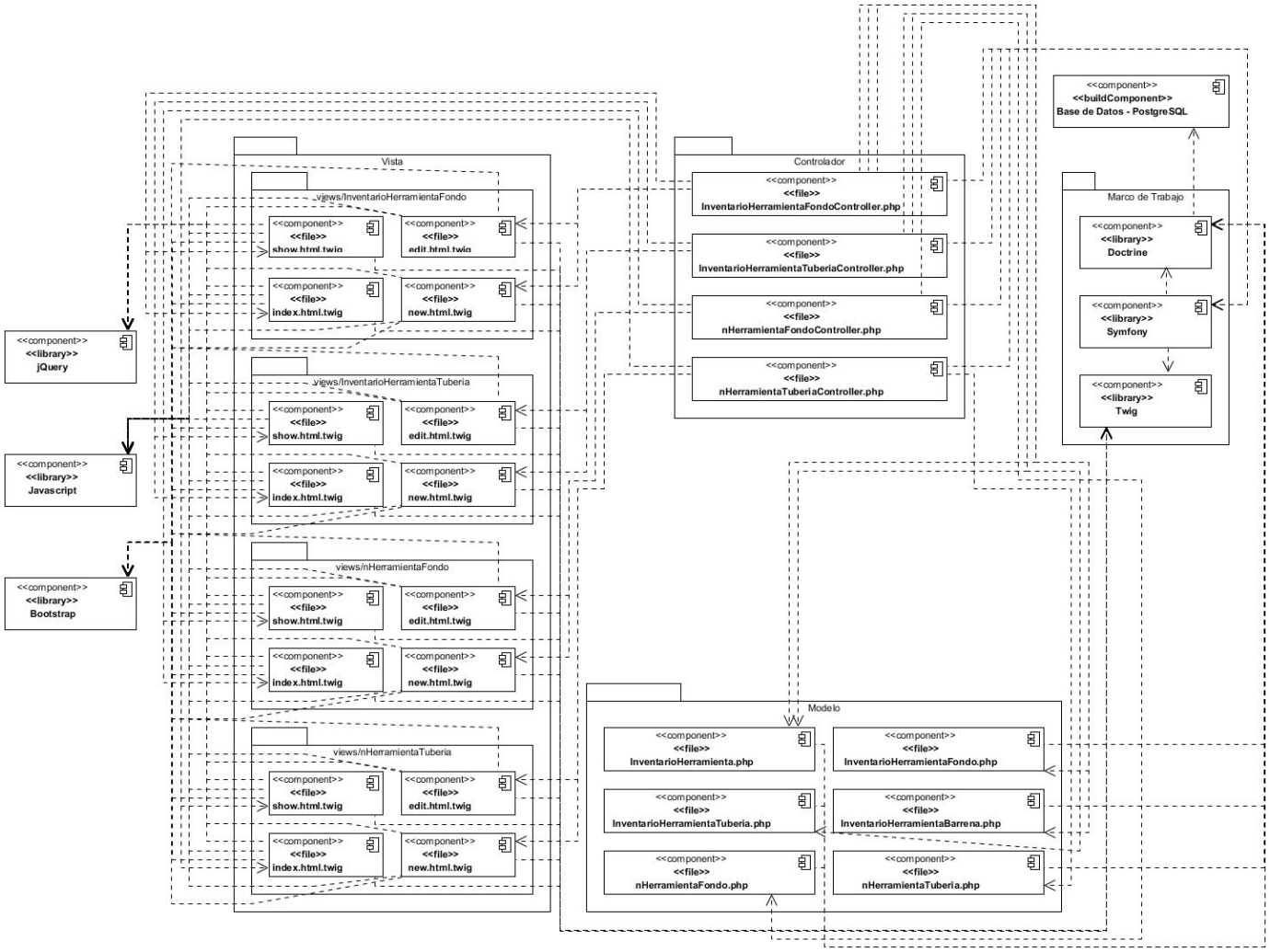


Imagen 6. Diagrama de componentes

3.3. Modelo de despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo, cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo similar. Los nodos poseen relaciones que representan medios de comunicación entre ellos, la funcionalidad de un nodo se define por los componentes que se distribuyen en ese nodo (García, y otros, 2008).

A continuación, se presenta el diagrama de despliegue como una distribución física del módulo y del sistema en ejecución, enmarcados como nodos dentro del mismo.

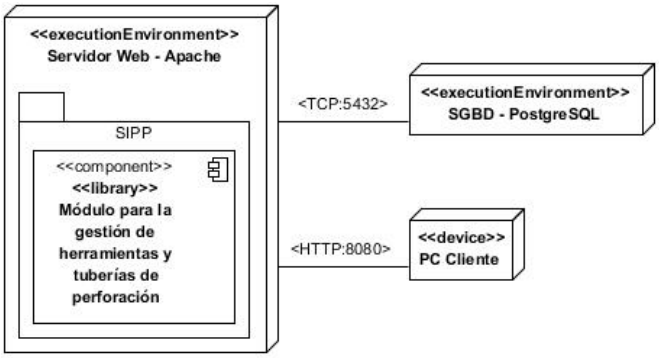


Imagen 7. Diagrama de despliegue

3.4. Pruebas de software

Una prueba es un conjunto de actividades que pueden planearse por adelantado y realizarse de manera sistemática. Por esta razón, durante el proceso de desarrollo de software, debe definirse una plantilla para la prueba del software: un conjunto de pasos que incluyen métodos de prueba y técnicas de diseño de casos de prueba específicos (Tencio, 2015).

Las pruebas de software son un elemento crítico para la garantía de la calidad del software y representan una revisión final de las especificaciones del diseño y de la codificación, en las cuales es observado el comportamiento del sistema o de algún componente del mismo bajo cierto entorno. En esta fase se debe realizar un diseño de pruebas con el objetivo de sacar a relucir errores, los cuales deben ser corregidos antes de entregar el software al cliente; este diseño debe verificar además que los componentes desarrollados se comuniquen de manera eficiente y que hayan sido implementados todos los requisitos correctamente.

3.4.1. Técnicas de prueba

Las pruebas de *software* deben realizarse desde dos puntos de referencia, el primero sería la observación de la lógica del programa que se logra mediante la revisión del código (pruebas de caja blanca) y el segundo sería la verificación de que la propuesta desarrollada se ajuste a las especificaciones y cumpla con todos los requisitos del cliente (pruebas de caja negra). Para validar la eficacia de la solución se deciden realizar pruebas de caja negra, debido a que los sistemas de gestión basados en tecnologías web tienen un fuerte enfoque a la comunicación entre el sistema y el usuario, posibilitando que con pruebas funcionales se detecten errores que resalten en el sistema.

Pruebas de Caja Negra

Las pruebas de caja negra, es una técnica de pruebas de software en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software. En las pruebas de caja negra, nos enfocamos solamente en las entradas y salidas del sistema, sin preocuparnos en tener conocimiento de la estructura interna del programa de *software* (Terrera, 2017).

En estas pruebas se observa el comportamiento del sistema mediante el uso del mismo y desde la revisión contra los casos de uso del sistema, permitiendo así encontrar errores en la interfaz, en las funciones implementadas, en el acceso a datos y en el rendimiento.

3.5. Diseño de casos de pruebas

Un caso de prueba no es más que el artefacto que muestra el nivel de completitud de un caso de uso con las condiciones de entrada y resultados esperados (Ramachandran, 2018). Al módulo desarrollado se le planificaron dos iteraciones de pruebas de caja negra con el objetivo de encontrar errores funcionales que puede presentar y que no fueron detectados en la etapa de implementación, la tabla que se muestra a continuación hace referencia a la segunda iteración de las pruebas aplicadas.

Tabla 4. Caso Prueba HU4: Crear nomenclador de herramienta/tubería

Escenario	Descripción	Nombre	Nombre corto	Respuesta del sistema	Evaluación
EC 1.1 Crear nomenclador de tipo herramientas correctamente	**1:** El usuario sigue la ruta **Nomencladores -> Tipo -> Tipo de herramientas** **3:** Presiona el botón **"Adicionar"** **5:** Introduce los valores correspondientes en cada campo y presiona el botón **"Guardar y enviar".** **7:** Presiona la opción **"Aceptar".**	V Hnta 1	V H2	**2:** El sistema muestra una vista con el listado de herramienta/tubería y el botón **"Adicionar"**. También muestra un campo de acciones a la derecha de cada elemento de la lista con los botones **"Detalles"**, **"Modificar"** y **"Eliminar"** **4:** El sistema muestra una ventana con los campos: Nombre y Nombre Corto. Además de los botones **"Listar"** y **"Guardar y enviar"** **6:** Muestra una nueva ventana con el mensaje de confirmación **8:** Registra el nomenclador con los datos especificados, mostrándolos en el listado	**Satisfactoria**
EC 1.2 Cancelar	**1:** El usuario sigue la ruta **Nomencladores -> Tipo -> Tipo de herramientas** **3:** Presiona el botón **"Adicionar"** **5:** Introduce los valores correspondientes en cada campo y presiona la opción **"Guardar y enviar".** **7:** Presiona la opción **"Cancelar".**	V Hnta 3	V 745	**2:** El sistema muestra una vista con el listado de herramienta/tubería y el botón **"Adicionar"**. También muestra un campo de acciones a la derecha de cada elemento de la lista con los botones **"Detalles"**, **"Modificar"** y **"Eliminar"** **4:** El sistema muestra una ventana con los campos: **Nombre y Nombre Corto.** Además de los botones **"Listar"** y **"Guardar y enviar"** **6:** Muestra una nueva ventana con el mensaje de confirmación **8:** No registra nomenclador con los datos especificados, se muestra el listado	**Satisfactoria**

EC 1.3 Campos vacíos	**1:** El usuario sigue la ruta **Nomencladores -> Tipo -> Tipo de herramientas** **3:** Presiona el botón **"Adicionar"** **5:** Introduce los valores correspondientes en cada campo, dejando campos vacíos y presiona el botón **"Guardar y enviar".**	I (vacío)	I (vacío)	**2:** El sistema muestra una vista con el listado de herramienta/tubería y el botón **"Adicionar"**. También muestra un campo de acciones a la derecha de cada elemento de la lista con los botones **"Detalles"**, **"Modificar"** y **"Eliminar"** **4:** El sistema muestra una ventana con los campos: **Nombre y Nombre Corto.** Además de los botones **"Listar"** y **"Guardar y enviar"** **6:** Detecta la existencia de campos vacíos, señalando él o los campos en cuestión en rojo.	**Satisfactoria**
EC 1.4 Listar	**1:** El usuario sigue la ruta **Nomencladores -> Tipo -> Tipo de herramientas** **3:** Presiona el botón **"Adicionar"** **5:** Presiona el botón **"Listar".**	I N/A	I N/A	**2:** El sistema muestra una vista con el listado de herramienta/tubería y el botón **"Adicionar"**. También muestra un campo de acciones a la derecha de cada elemento de la lista con los botones **"Detalles"**, **"Modificar"** y **"Eliminar"** **4:** El sistema muestra una ventana con los campos: Nombre y Nombre Corto. Además de los botones **"Listar"** y **"Guardar y enviar"** **6:** Regresa al listado de herramienta/tubería	**Satisfactoria**

EC 1.5 Campos con valores incorrectos	**1:** El usuario sigue la ruta **Nomencladores -> Tipo -> Tipo de herramientas** **3:** Presiona el botón **"Adicionar"** **5:** Introduce los valores correspondientes en cada campo, de los cuales algunos son incorrectos y presiona el botón **"Guardar y enviar".**	I unfertty*887	I UF887	**2:** El sistema muestra una vista con el listado de herramienta/tubería y el botón **"Adicionar"**. También muestra un campo de acciones a la derecha de cada elemento de la lista con los botones **"Detalles"**, **"Modificar"** y **"Eliminar"** **4:** El sistema muestra una ventana con los campos: **Nombre y Nombre Corto.** Además de los botones **"Listar"** y **"Guardar y enviar"** **6:** Detecta la existencia de campos vacíos, señalando él o los campos en cuestión en rojo.	**Satisfactoria**

3.5.1. Resultados de las Pruebas de Caja Negra

Durante la fase de prueba se realizaron dos iteraciones al módulo para la gestión de herramientas y tuberías de perforación, estas arrojaron los siguientes resultados:

Iteración 1. Se encontraron un total de cinco no conformidades: una de ortografía referente al vocablo herramienta en la vista para la gestión del inventario de herramientas de fondo; dos de interfaz referentes a las notificaciones de adición y modificación exitosas en la vista para la gestión del inventario de herramientas de fondo, donde los mensajes de confirmación no se mostraban; las dos restantes fueron no conformidades de funcionalidad, primero se detectó que la expresión regular que validaba la entrada de caracteres de los campos 'Número de Serie' y 'Modelo' en la gestión del inventario de herramientas de fondo, permitía la entrada de espacios y del carácter '%', también se detectó que, cuando se adicionaba una herramienta que no fuera barrena el sistema exigía el valor del campo 'Box' y cuando no era barrena exigía el valor del campo 'IADC'. Estos errores fueron corregidos y comprobada la fidelidad de la corrección, dando paso a una nueva iteración de las pruebas.

Iteración 2. No se encontraron no conformidades, quedando el módulo libre de errores y obteniéndose un software que cumple con los requisitos del usuario.

La siguiente gráfica muestra los resultados obtenidos durante las pruebas aplicadas al módulo.

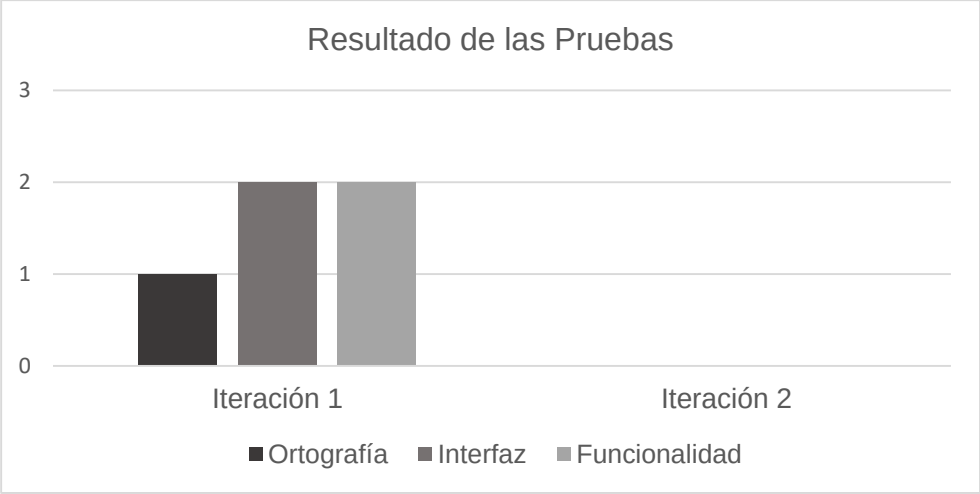


Imagen 8. Resultado de las pruebas de caja negra

3.6. Resultados obtenidos

Como resultado de la presente investigación y de la etapa de desarrollo, se obtuvo un módulo funcional para la gestión de herramientas y tuberías de perforación, el cual fue integrado exitosamente al Sistema Integral de Perforación de Pozos v3.0.

La construcción del módulo siguiendo el modelo definido en la fase de análisis y diseño, permitió simplificar en gran medida el trabajo con el inventario de herramientas de fondo, debido a que antes era necesario acceder a diferentes vistas del sistema para registrar una herramienta específica, mientras que ahora sólo existe un inventario de herramientas de fondo. Con la concepción y posterior implementación de los nomencladores para los tipos de herramientas de fondo y tuberías de perforación, es posible definir nuevas herramientas y tuberías, funcionalidad no existente en SIPP v2, posibilitando que luego de definir una nueva clase de herramienta de fondo o tubería de perforación, se puedan adicionar herramientas o tuberías de este tipo a los inventarios de herramientas y tuberías de perforación.

Desde el punto de vista del nuevo modelo de datos, es más escalable en comparación con el usado actualmente, este presenta menos clases y tablas en la base de datos.

En las vistas para la gestión de los inventarios fue añadido un componente que permite al usuario realizar búsquedas acotando el conjunto de elementos que se presenta en el listado del inventario.

El módulo desarrollado presenta un mayor grado de usabilidad, eficiencia y escalabilidad en relación con el actual, pues facilita en mayor medida la gestión de los inventarios de herramientas y tuberías de perforación, agrupa en una vista el inventario de herramientas de fondo, presenta un buscador para los listados de herramientas y tuberías, además, cuenta con un modelo de datos más estable y robusto. Con el nuevo módulo será posible disminuir en gran medida el tiempo empleado en la gestión de las herramientas y tuberías de perforación.

3.7. Conclusiones

En este capítulo se presentó una descripción de las tareas referentes a las etapas de implementación y pruebas. Como artefacto primario dentro del marco de implementación se muestra el diagrama de componentes, mediante el cual se logra una mayor comprensión de los componentes físicos que conforman el módulo y las dependencias entre los mismos. Se definió como técnica de prueba, las pruebas de caja negra, las cuales mostraron no conformidades y problemas funcionales dentro del sistema de acuerdo a los

requisitos del cliente, elementos que fueron solucionados para la última iteración de las pruebas, comprobándose así que el software no presenta ningún error que atente contra las necesidades del cliente. La culminación del presente capítulo se presenta como un final satisfactorio a la etapa de desarrollo.

CONCLUSIONES GENERALES

Tras la culminación del presente trabajo se consumó el desarrollo del módulo para la gestión de herramientas y tuberías de perforación en Sistema Integral de Perforación de Pozos v3.0, dándole cumplimiento al objetivo general y apoyándose en las tareas de investigación, las cuales fueron satisfactoriamente vencidas. Habiéndose alcanzado la meta planteada, se concluye que:

- Al abordar el problema planteado luego de haberse elaborado el marco teórico, se logró un mayor entendimiento tomando como base la investigación realizada sobre soluciones similares y los conceptos estudiados referentes al objeto de estudio, lo cual, junto a la certera selección de herramientas, tecnologías y metodología de desarrollo, favoreció la concepción de un módulo robusto y la integración del mismo con SIPP v3.0.
- Mediante la descripción de las características del módulo, se logró establecer una base sólida para la implementación del mismo, permitiendo, junto a la realización de las tareas referentes a la metodología y a las buenas prácticas asociadas al diseño de *software*, el desarrollo de un componente funcional y que resuelve las necesidades actuales derivadas del módulo implementado para SIPP v2.
- Las pruebas realizadas al componente permitieron identificar errores dentro del mismo, los cuales fueron corregidos; siendo validado el producto y habiéndose saldado cualquier inconformidad del cliente.
- El módulo concebido y desarrollado tras la investigación realizada, constituye una mejora en usabilidad y contribuye a aumentar la eficiencia de los procesos de negocio referentes a la gestión de los inventarios de herramientas y tuberías de perforación, en relación con el desplegado actualmente.

RECOMENDACIONES

Durante el desarrollo de la presente investigación, brotaron ideas que podrían ser implementadas dentro de futuras versiones, ampliando o mejorando las funcionalidades del módulo para la gestión de herramientas y tuberías de perforación en el Sistema Integral de Perforación de Pozos v3.0, por lo que se recomienda:

- Desarrollar una vista para la gestión y manipulación de la sarta de perforación que se apoye en el nuevo módulo para la gestión de herramientas y tuberías de perforación, haciendo uso eficiente de los inventarios implementados dentro de este.
- Implementar una funcionalidad que permita exportar e importar especificaciones de herramientas de fondo en archivos encriptados, las cuales permitirían a las compañías contratadas en los pozos, poseer la información de las herramientas de su propiedad, las cuales podrían utilizar al firmar un contrato de trabajo en un pozo diferente.

REFERENCIAS BIBLIOGRÁFICAS

1&1-DigitalGuide. 2016. 1&1 - Digital Guide. [En línea] 25 de Octubre de 2016. https://www.1and1.es/digitalguide/servidores/know-how/servidor-web-definicion-historia-y-programas.

Aguilar, Carlos Proal. 2018. Interactive and Cooperative Technologies Lab. [En línea] 2018. http://ict.udlap.mx/people/carlos/is341/bases02.html.

Alcala, Mario Lorenzo. 2007. *Medida de la usabilidad en aplicaciones de escritorio.* 2007.

Alcolea, Walfrido Lora. 2012. *"Desarrollo de un servidor de grabación multiplataforma para el sistema de video vigilancia Suria Visión.".* La Habana, Cuba : UCI, 2012.

ALEGSA. 2011. [En línea] 2011. http://www.alegsa.com.ar/Dic/biblioteca..

Beas, José Manuel. 2011. jmbeas.es. [En línea] 23 de Mayo de 2011. http://jmbeas.es/guias/historias-de-usuario/.

Block, Glenn. 2018. Microsoft - Developer. [En línea] 2018. https://blogs.msdn.microsoft.com/gblock/2006/10/26/ten-advantages-of-an-orm-object-relational-mapper/.

Brackets. 2018. Brackets. [En línea] 2018. http://brackets.io/.

Cases, Eduard Fumàs. 2018. Ibrugor-Blog. [En línea] 2018. http://www.ibrugor.com/blog/apache-http-server-que-es-como-funciona-y-para-que-sirve.

CAVSI. 2018. CAVSI. [En línea] 2018. http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd.

Craig, Larman. 2008. *UML y PATRONES. Introducción al análisis y diseño orientado a objeto.* 2008.

Doctrine-Project. 2018. Doctrine-Project. [En línea] 2018. https://www.doctrine-project.org/projects/doctrine-orm/en/latest/tutorials/getting-started.html#what-is-doctrine.

Expósito, Yuliet Hernández. 2011. *"Desarrollo de un video sensor para la detección de objetos abandonados.".* La Habana, Cuba : UCI, 2011. 5.

Fabre, Joyce Suárez. 2013. Módulo de Seguridad para SIPP v 2.0. *Trabajo de Diploma.* 2013.

Fernández, Adán Andrés y Bazurto, Michelle Joselin. 2015. Análisis Técnico de un Pozo Direccional de Dos Secciones. *Proyecto Integrador.* 2015.

García, Francisco José, Conde, Miguel Ángel y Bravo, Sergio. 2008. Diseño orientado a objetos. *Ingeniería de Software, Tema 6: Diseño orientado a objetos.* s.l., Salamanca, España : Universidad de Salamanca, 16 de Octubre de 2008.

GoldenSoftware. 2018. GoldenSoftware. [En línea] 2018. http://www.goldensoftware.com/products/strater.

Gómez, Mauro. 2013. IngenioDS. [En línea] 16 de Septiembre de 2013. https://ingeniods.wordpress.com/2013/09/16/patrones-arquitectonicos/.

Hernández, Leovigilda. 2013. [En línea] 1 de Junio de 2013. http://ithleovi.blogspot.com/2013/06/unidad-5-modelo-deimplementacion-el.

IBM_Knowlegde_Center. 2018. IBM Knowlegde Center. [En línea] 2018. https://www.ibm.com/support/knowledgecenter/es/SSZLC2_8.0.0/com.ibm.commerce.developer.doc/concepts/csdmvcdespat.htm.

IEEE. 1998. Estándar 830: Especificaciones de los requisitos de software. s.l. : https://www.ctr.unican.es/asignaturas/is1/IEEE830_esp.pdf, 1998.

Jacobson, Ivar, Booch, Grady and Rumbaugh, James. 2000. *El proceso unificado de desarrollo de software.* Madris : Pearson Educación, 2000. 84-78-29-036-2.

JQuery. 2018. JQuery. [En línea] 2018. https://jquery.com/.

Krishnamurthy, Ganesh. 2018. University of Missouri-St. Louis. [En línea] 2018. http://www.umsl.edu/~sauterv/analysis/F08papers/View.html.

Laguna, Miguel A. 2018. Ingeniería de Software. Requisitos. s.l. : Departamento de Informática. Universidad de Valladolit., 2018.

Lapuente, María Jesús Lamarca. 2013. Hipertexto: El Nuevo Concepto de Documento en la Cultura de la Imagen. *Tesis de Doctorado.* 8 de Diciembre de 2013.

LucidChart. 2018. LucidChart. [En línea] 2018. https://www.lucidchart.com/pages/es/qu%C3%A9-es-el-lenguaje-unificado-de-modelado-uml.

LucidChart. 2018. LucidChart. [En línea] 2018. https://www.lucidchart.com/pages/es/qu%C3%A9-es-un-diagrama-entidad-relaci%C3%B3n.

MDN-Mozilla. 2017. MDN-Mozilla. [En línea] 27 de Noviembre de 2017. https://developer.mozilla.org/es/docs/Web/HTML.

MDN-Mozilla. 2015. MDN-Mozilla. [En línea] 26 de Abril de 2015. https://developer.mozilla.org/es/docs/Web/JavaScript/Acerca_de_JavaScript.

NeitamPetrol. 2018. NeitamPetrol. [En línea] 2018. https://www.neitam.com/productos/heavy-weight/.

OBS Business School. 2016. OBS-EDU. [En línea] 2016. https://www.obs-edu.com/int/blog-project-management/metodolog%C3%ADa-agile/que-son-las-metodologias-de-desarrollo-de-software.

Visual Paradigm. 2010. Visual Paradigm. *Visual Paradigm.* [En línea] 2010. http://www.visual-paradigm.com/.

Peloton. 2018. Peloton. *Peloton - WellView.* [En línea] 2018. http://www.peloton.com/products/wellview.

Pimentel, ErnestoAntonio Maña, Javier López, J.M. Troya. 2010. *Un Esquema Eficiente de Protección de Software Basado en Tarjetas Inteligentes.* Málaga : Universitaria, 2010. 29071.

Ponchar. 2013. Ponchar. [En línea] 4 de Mayo de 2013. https://ponchar.com/software-basado-en-web/.

PorpoiseMedia. 2018. PorpoiseMedia. *PorpoiseMedia - Well-Logger.* [En línea] 2018. http://www.porpoisemedia.com/?page_id=49.

PostgreSQL-ORG. 2018. PostgreSQL-ORG. [En línea] 2018. https://www.postgresql.org/about.

Pressman, Roger S. 1998. Ingeniería de software. Un enfoque práctico. 1998.

Ramachandran, Saranya. 2018. Study.com. [En línea] 2018. relaci%C3%B3nhttps://study.com/academy/lesson/what-is-a-test-case-definition-example.html.

Rodríguez, Txema. 2018. GenbetaDev. [En línea] 2018. https://www.genbetadev.com/frameworks/bootstrap.

SafeNet. 2006. SafeNet: The Data Protection Company. *SafeNet: The Data Protection Company.* [En línea] Noviembre de 2006. [Citado el: 6 de Noviembre de 2012.] http://www.safenet-inc.com/software-monetization/sentinel-software-licensing-products/?aldn=true.

Salamanca, Universidad. Departamento de Informatica y Automatica. 2009. Modelo de Despliegue. [En línea] 2009. http://es.scribd.com/doc/49708406/19/Modelo-de-despliegue.

Schlumberger. 2018. Schlumberger.Glosary. [En línea] 2018. http://www.glossary.oilfield.slb.com/es/Terms/.

Sean W. Smith, S.W. 2007. *Building a High-Performance, Programmable Secure Coprocessor.* 2007.

Silva, Angel Da. 2009. LaComunidadPetrolera. [En línea] 11 de Mayo de 2009. www.lacomunidadpetrolera.com/2009/05/pozo-petroleo.html.

2009. SlideShare. [En línea] 28 de Mayo de 2009. http://www.slideshare.net/Danto/patrn-mvc.

SOFPRO, Lab Software Protection. 2011. PC Guard Software. *PC Guard Software.* [En línea] 2011. http://www.sofpro.com/pc.guard.htm.

Symfony. 2018. Symfony. [En línea] 2018. https://symfony.com.

Synergix. 2008. Synergix. [En línea] 10 de Julio de 2008. https://synergix.wordpress.com/2008/07/10/modelo-de-dominio/.

Techerald. 2012. Techerald. *Techerald.* [En línea] 2012. [Citado el: 15 de Noviembre de 2012.] http://techerald.com/.

Tencio, Diego. 2015. Prezi. [En línea] 23 de Noviembre de 2015. prezi.com/9r2q23n4kzpw/administracion-de-calidad.

Terrera, Gustavo. 2017. TestingBaires. [En línea] 26 de Febrero de 2017. https://testingbaires.com/pruebas-caja-negra-enfoque-practico/.

Tomás, Eduard y Alvarez, Miguel Angel. 2013. desarrolloweb. [En línea] 20 de Mayo de 2013. https://desarrolloweb.com/articulos/sistemas-plantillas-javascript.html.

Torrecilla, Pablo. 2012. nosolopau. [En línea] 7 de Junio de 2012. http://nosolopau.com/2012/06/07/mas-sobre-el-proceso-unificado-agil-fases-y-disciplinas/.

Torres, Fernando. 2003. *Integración del PMBOOK al RUP para proyectos de Desarrollo de Software.* 2003.

Tuomas, Aura, D.G. 2005. *Software Licence Management with Smart Cards.* 2005.

Twig. 2018. Twig. [En línea] 2018. https://twig.symfony.com/.

Valencia, Universidad Politecnica de. 2006. *Introducción a RUP.* 2006.

Valerio, Oralis. 2013. Filosofía de uso e importancia de las librerías en C++. *Lenguaje de Programación.* 2013.

Vallejo, David y Martín, Cleto. 2012. *Desarrollo de Videojuegos: Arquitectura del Motor.* Ciudad Real : Bubok, 2012.

Van Der Henst, Christian. 2018. MaestrosDelWeb. [En línea] 2018. http://www.maestrosdelweb.com/phpintro.

Varhaug, Marr. 2012. Un giro a la derecha: Una visión general de las operaciones de perforación. 2012. pág. 3.

Visconti, Marcelo y Astudillo, Hernán. 2018. Fundamentos de Ingeniería de Software. 2018.

VisualParadigm. 2018. VisualParadigm. [En línea] 2018. https://www.visual-paradigm.com.

W3-ORG. 2018. W3-ORG. [En línea] 2018. www.w3.org/TR/1999/REC-html401-19991224/present/styles.html.

Weitzenfeld, Alfredo. 2000. Ingenieria de software orientada a objetos con UML. 2000.