Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Subsistema Seguridad del Sistema de Manejo Integral de la Perforación de Pozos de Petróleo

Autor: David Rodríguez Lagrana

Tutor: David Tavares Cuevas

"Año 53 de la Revolución"

Ciudad de la Habana. 2011

Declaración de Auditoría

Declaro que soy el único autor de este trabajo y autorizo al Centro de Desarrollo de Informática Industrial (CEDIN) de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____ _____

Firma del autor **Firma del tutor**

David Rodríguez Lagrana. **Ing. David Tavares Cuevas.**

Dedicatoria

Agradecimientos

Resumen

En la actualidad la Industria del Petróleo en el mundo, sustenta el modo de vida de los seres humanos. Para que esta sea cada día más rentable y eficiente, se desarrollan sistemas con la intención de solucionar problemas que aún existen y persisten. En nuestro país CUPET es una Unión de Empresas, encargadas de dirigir cada uno de los procesos (Exploración-Producción, Refinación y Comercialización). La Dirección Intervención y Perforación de Pozos (DIPP) es la encargada de dirigir y monitorear el proceso de perforación en tierra en todo el territorio nacional. El Sistema de Manejo Integral de Perforación de Pozos (SIPP) es el proyecto de desarrollo e investigación que está orientado a brindar una solución integral, a las necesidades de la perforación en tierra de CUPET.

El presente trabajo consiste en el desarrollo de un Subsistema de Seguridad para el Sistema de Manejo Integral de la Perforación de Pozos de Petróleo. Este subsistema se enfoca en solucionar los problemas de integridad y confiabilidad de la información que se maneja en el negocio. Se abordan los conceptos asociados al problema y se realiza un estudio del arte, para obtener una visión general del mercado de este tipo de sistemas. Se exponen las herramientas y tecnologías utilizadas para el desarrollo de la solución, haciendo énfasis en las que constituyen elementos críticos de la arquitectura. Se profundiza en las características funcionales y no funcionales de la solución, mostrándose su funcionamiento a través del modelo de diseño e implementación del subsistema. Para validar el resultado se realizaron pruebas de aceptación y pruebas de caja negra, para asegurar el cumplimiento de los requisitos pactados y la calidad del producto respectivamente.

Palabras Claves: Petróleo, seguridad, requisitos y calidad.

Índice de Contenido.

Índice de Figuras.

Índice de Tablas.

Introducción

El uso de las Tecnologías de la Información y las Comunicaciones (TIC) es de vital importancia para el desarrollo de nuestra sociedad. Gracias a su correcta aplicación en sectores medulares de la economía y la sociedad pueden facilitar las comunicaciones, eliminar las barreras de tiempo y espacio, favorecer la cooperación entre distintas entidades, aumentar la producción de bienes y servicios, entre otros. En Cuba desde el triunfo de la Revolución uno de los principales objetivos de nuestro estado socialista, ha sido la superación cultural del pueblo. El uso de las TIC se ha fomentado en la población desde edades tempranas, como estrategia para incrementar el conocimiento y la preparación de nuestra sociedad.

La Unión Cuba-Petróleo (CUPET) es la unión de empresas responsable del desarrollo y mantenimiento de la industria del petróleo en Cuba, en conjunto con otras compañías extranjeras, que operan y prestan servicios a esta importante y vital rama de la economía. Estas empresas con sus distintas entidades distribuidas en todo el país intervienen en los diferentes procesos por los que debe pasar el crudo. La industria del petróleo en nuestro país, como en muchos otros, se divide en tres grandes procesos: Exploración-Producción, Refinación y Comercialización. Las entidades especializadas en Exploración-Producción, como su nombre lo indica, se encargan de la exploración y desarrollo de los campos, la perforación de nuevos pozos, así como la reparación de los que ya se encuentran en producción (Intervención de Pozos) (1)**.**

La automatización de los procesos de la industria del petróleo es crítico para su desarrollo. Los sistemas encargados de su gestión, manejan grandes cantidades de datos los cuales son imprescindibles para la toma de decisiones. Regularmente la información asociada al resultado de análisis sísmicos, prospecciones geológicas, reservas o simplemente resultado de un subproceso determinado, es clasificada como confidencial, por lo cual los sistemas que la procesan y analizan deben garantizar la seguridad sobre los datos y el acceso a los mismos. La seguridad informática es una disciplina que se encarga de proteger la integridad y la privacidad de la información almacenada en un sistema informático; no existe ninguna técnica que permita asegurar la inviolabilidad de un sistema (2). A pesar de que no existen sistemas totalmente seguros, deben valorar la seguridad como elemento fundamental dentro del funcionamiento de los mismos.

En el mundo existen sistemas que gestionan el proceso de perforación a diferentes niveles. Son el caso de software como WellWizard, el cual provee al usuario de información en tiempo real del pozo, dejando un registro histórico de todas las variables monitoreadas, desde el inicio hasta el final de perforación (3). Otros sistemas como Well View el cual permite crear un archivo de pozo corporativo completo desde la petición de perforar un pozo hasta el abandono. Contiene funcionalidades para el

pronóstico geológico, de planificación de la perforación, gestión de las operaciones de perforación, geológicas y de intervención (4). La generalidad de este tipo de aplicaciones comparten características como: son sistemas probados y fiables, desarrolladas para plataforma Windows en entorno de escritorio y su elemento más débil la baja gestión de la seguridad, concentrándose únicamente en la integridad de la información, obviando la confiabilidad.

Para poder comprender el alcance y relevancia de la necesidad de la aplicación de este tipo de software, se hace necesario e indispensable conocer el proceso que se lleva a cabo en nuestro país. La Dirección de Intervención y Perforación de Pozos (DIPP), es la encargada de dirigir y monitorear todos los procesos que se realizan en los pozos en perforación. Esta entidad radica en la Empresa de Producción y Extracción de Petróleo del Centro (EPEPC), perteneciendo a la Dirección de Exploración - Producción de la Unión de CUPET. Aquí se realizan y se aprueban los proyectos de los pozos a perforar, los cuales se elaboran por un equipo multidisciplinario de diferentes ramas (Geología y Perforación), siendo así la responsable de los pozos desde su proyección hasta su terminación.

La DIPP contrata a las Compañías de Servicio (CIAS) que van a trabajar en el pozo, que brindan servicios de Direccionales, Lodo, Perforación (Contratistas), Registro y Geología. De todas estas empresas (compañías) de servicio, nuestro país solo brinda dos servicios: Geología, el cual es brindado por el Centro de Investigaciones de Petróleo (CEINPET); en el caso de Perforación e Intervención la empresa cubana que presta este servicio es EMPERCAP, todas las demás empresas que brindan servicios a la actividad de perforación son extranjeras. En el proceso de contratación se aprueban los presupuestos, los cuales van contenidos dentro del proyecto de pozo. Luego de contratado los servicios comienza el proceso de izaje y montaje del equipo de perforación; concluido este momento se inicia la perforación. Durante la perforación se realizan operaciones que generan información valiosa, confidencial e imprescindible para la toma de decisiones en la DIPP, por ejemplo: Registro del avance diario, actividades diarias, caudal de las bombas del lodo, propiedades de uso de la barrena, descripción de las muestras extraídas (cortes o sartas de perforación), torque y arrastre, propiedades del lodo, aditivos añadidos al lado, uso de combustible, por solo mencionar algunos.

Al no contar con un sistema que gestione y lleve un control de la información, los supervisores de los pozos son los encargados de construir diariamente los reportes de los procesos realizados, integrando todos los reportes entregados por las compañías de servicio; además de también dirigir el proceso de perforación. Todo este trabajo se realiza de forma manual utilizando la herramienta Microsoft Excel 2003. Aunque esta herramienta brinda algunas facilidades esta labor resulta muy engorrosa y trabajosa, haciendo que el costo en tiempo y esfuerzo aumente a medida que avanza el trabajo en el pozo. Luego de concluidas las tareas específicas de la perforación, se inicia el proceso de terminación y ensayo, es donde se ultiman los detalles para iniciar la extracción del crudo. Luego que los pozos

pasan a producir, pasan a manos de una de las empresas de producción y extracción (Centro, Occidente o Majagua), pero esta entidad continua participando, ya que es la encargada de contratar y monitorear los servicios de Intervención de Pozos, que se realizan con el objetivo de reparar o realizar trabajos de control sobre los pozos que ya están en producción. Todos estos datos recopilados permanecen disponibles ya que son muy importantes en caso de averías o reparaciones.

En la Universidad de las Ciencias Informáticas (UCI), se desarrollan sistemas para la automatización de procesos de la Industria del Petróleo. En la facultad 5 se encuentra el Centro de Informática Industrial donde se desarrolla el Sistema de Manejo Integral de Perforación de Pozos (SIPP), el cual responde a las necesidades del Centro de Investigaciones de Petróleo (CEINPET) y la Dirección de Intervención y Perforación de Pozos (DIPP). Este sistema tiene como principal objetivo la gestión de la información consolidada del proceso de perforación en tierra y la del proceso en sí mismo. Uno de los requisitos solicitados por el cliente de SIPP, además de los relacionados con la gestión del proceso, es la necesidad de garantizar la seguridad en el sistema. En las actividades asociadas al proceso de perforación, no existe un control sobre los usuarios que tienen acceso a la manipular información sensible. Para ello el equipo de desarrollo se dio a la tarea de identificar requisitos funcionales y no funcionales para asegurar la seguridad en todos los niveles. Con el análisis y propuesta de solución se identificó la necesidad de un subsistema que fuera capaz de gestionar la seguridad sobre los datos y el control de acceso al sistema garantizando que los trabajadores puedan ser capaces de realizar las acciones asociadas a su responsabilidad en el proceso.

Dada la **situación problemática** anterior, se identifica el **problema a resolver** que será ¿Cómo garantizar la integridad y confidencialidad de la información cumpliendo con los requisitos funcionales y no funcionales de seguridad de SIPP? Luego de identificado el problema se define como **objetivo general** implementar el Subsistema de Seguridad de SIPP. Se identifica como **objeto de estudio** los procesos de gestión y control de las actividades durante la perforación en la DIPP y los pozos en perforación. La presente investigación está enmarcada en la gestión de la seguridad del SIPP, el cual es identificado como el **campo de acción**. Se defiende la idea de que con la implementación del Subsistema de Seguridad de SIPP, se podrá garantizar la integridad y confiabilidad de la información cumpliendo los requisitos pactados.

Este subsistema le proporcionará a la solución SIPP, un enfoque alto hacia la seguridad, distinguiéndolo de los demás software. Aumenta las prestaciones, enfocando la solución a roles, los cuales pueden ser gestionados. Esto permitirá la escalabilidad del sistema en el tiempo de uso, posibilitando que la propia entidad cliente ajuste la gestión de seguridad a sus necesidades. La personalización de la administración de usuarios según el entorno de trabajo (DIPP o Pozo), permite

que la gestión de seguridad se realice sin conexión; sin duda una característica que distingue la solución propuesta.

Para darle cumplimiento al objetivo general se han trazado las siguientes **tareas de investigación:**

- Realizar un estudio al estado del arte de sistemas que se utilicen en la perforación, para obtener una visión del negocio.
- Describir las herramientas y tecnologías que se utilizarán para el desarrollo del Subsistema.
- Implementar el Subsistema.
- Validación de la solución propuesta, para garantizar la calidad del resultado.

Para el desarrollo de esta investigación se hizo necesario utilizar algunos de los métodos científicos tales como los métodos teóricos que tienen su sustento en la concepción materialista-dialéctica y facilitan la recopilación de información necesaria para el desarrollo del Subsistema. Dentro de estos se utilizó, el histórico-lógico que gracias a este fue posible realizar un análisis a nivel nacional e internacional de procesos asociados a seguridad en sistemas Web y poder conocer el estado actual de Subsistemas que puedan servir de ayuda.

Para un mejor entendimiento de este trabajo se decide estructurar el mismo de la siguiente manera:

Capítulo I: Se expone el estado del arte, aspectos generales de SIPP. Al igual se profundiza el objeto de estudio, conceptos asociados al problema y la situación problemática.

Capítulo II: Se describen las tendencias y tecnologías utilizadas para la correcta implementación, además de justificarse las herramientas, lenguajes y metodologías seleccionadas.

Capítulo III: Se enfoca principalmente en la construcción del subsistema, donde se detalla el desarrollo asociado a los requisitos del sistema, se exponen los métodos utilizados para la validación del resultado.

Capítulo 1: Fundamentación Teórica.

1.1 Introducción.

La fundamentación teórica de conceptos así como descripciones de problemas que se ocurren en la actualidad, representan una forma de entendimiento más completa. El presente capítulo estará dividido en varios epígrafes; el primero de ellos se profundiza acerca de conceptos asociados al dominio del problema, en aras de esclarecer el ámbito donde se identifica el objeto de estudio. Posteriormente se expone el estado del arte, realizando un análisis así de las soluciones existentes, profundizando en la aplicación de seguridad a los sistemas para garantizar la integridad, disponibilidad y confidencialidad de los datos que manejan.

1.2 Conceptos asociados al dominio del problema.

El dominio del problema que se presenta es complejo y se requiere de cierta experiencia para conceptualizar los términos que se manejan. En este epígrafe se exponen los conceptos más importantes para comprender el dominio del problema, en torno a la seguridad.

Petróleo: El petróleo es una mezcla heterogénea de compuestos orgánicos, principalmente hidrocarburos insolubles en agua, que se extrae de lechos geológicos continentales o marítimos. También es conocido como petróleo crudo o simplemente crudo. Es un líquido aceitoso, inflamable, cuyo color varia de incoloro a negro (3).

Seguridad: Aspecto fundamental que debe tener cualquier sistema informático, basada principalmente en la protección de activos los cuales pueden ser tangibles o no, tales como un servidor de Base de Datos o la reputación de una empresa. Esta evaluada en tres aspectos fundamentales tales como integridad, disponibilidad, confidencialidad. Estos aspectos a la vez están relacionados y dependen de otros tres elementos principales que encierran todos los distintos controles de que se pueden establecer en un sistema informático (4):

- *Autenticación:* los clientes cualquier aplicación o servicio, deben ser identificados de forma única, sean usuarios finales, otros servicios o computadoras externas.
- *Autorización:* No solo es necesario saber quiénes acceden a nuestros activos, también es necesario establecer que es lo que pueden hacer con ellos. Un nivel de autorización dado determina qué tipo de operaciones o transacciones puede efectuar un usuario dado sobre una información dada.

- *Registro y Auditoria:* Luego de efectuada una operación, es importante que esta sea registrada adecuadamente, es esencial si queremos evitar el repudio de transacciones efectuada por un cliente.

Sistemas de Gestión: Un sistema de gestión es una estructura probada para la gestión y mejora de las políticas, procedimientos y/o procesos de una organización. Estos sistemas son capaces de brindar diferentes tipos de requisitos tales como rentabilidad, competitividad, globalización velocidad de los cambios, capacidad de adaptación, crecimiento, tecnología. La implementación de cualquier sistema de gestión pueden ayudar a mejorar la efectividad operativa, reducir costos, aumentar la satisfacción de clientes y partes interesadas, lograr mejoras continuas, potenciar las innovaciones, entre otras fortalezas (5).

Unión CUPET: Es la empresa que se encarga de toda la actividad petrolera desde la exploración hasta la refinación, así como de satisfacer las necesidades del marcado nacional de hidrocarburos, a partir del incremento de la extracción de crudos nacionales.

Pozo de Petróleo y Gas: Entidad donde se realiza la perforación, la cual agrupa un gran número de entidades y trabajadores, como son las compañías de servicio de lodo, direccionales, contratistas, logísticas, entre otras.

Compañías de Servicio: Son empresas que se especializan en brindar uno o varios servicios indispensables para poder lograr el objetivo de la perforación. Estas empresas brindas servicios de direccional, lodo, registros, sistemas de monitoreo en tiempo real, perforación; esto se explica dado que en la actualidad a los dueños de los campos le es factible contratar estos servicios, que ellos mismo ejecutarlos todos. Estas compañías son muchos más eficientes y sus integrantes son especialistas en la rama en que se desempeñan.

Usuarios: Persona que utiliza o trabaja con algún objeto o que es destinatario de algún servicio público, privado y/o empresarial, también se conoce como actor dentro de un sistema informático. En general un usuario es un conjunto de permisos y de recursos a los cuales se tiene acceso, puede ser tanto una persona, una máquina o un programa. Algunos tipos de usuarios son *usuario administrador, usuario registrado, usuario anónimo*.

Roles: Conjunto de funciones, normas, comportamientos, derechos o responsabilidades que se les son otorgados a los usuarios, con el fin de agrupar y conectar con una serie de permisos por los cuales se van a regir. Existen diferentes tipos de roles para todas las ramas, en el caso específico de la informática podemos mencionar algunos como administrador, directivo, secretaria, supervisor, etc.

Registros de Seguridad: Es un registro oficial de eventos durante un rango de tiempo en particular, se informa sobre el quién, qué, cuándo, dónde y por qué, para controlar las actividades que se realizan sobre un sistema determinado, generalmente se guarda en ficheros de texto, pero también es almacenado en Bases de Datos para cualquier operación que posteriormente se quiera realizar, así como aplicar filtrados de campos, con el objetivo de encontrar una falla o una incidencia de seguridad en momentos determinados (3).

1.3 Objeto de estudio.

1.3.1 Descripción General.

El objeto de estudio lo constituyen los procesos de gestión y control de las actividades durante la perforación en la DIPP y los pozos en perforación. Actualmente no se cuenta con un sistema capaz de gestionar los datos durante el proceso y que garantice la seguridad, a todos los niveles, en la interacción con estos datos. La mayoría de los procesos que manejan información son realizados por Compañías de Servicio, las cuales pueden ser cubanas y extrajeras. Cada una de estas compañías emite diariamente un reporte de sus servicios. Los Supervisores son los encargados de dirigir el proceso de perforación, además de construir diariamente los reportes en su estación de trabajo. Toda esta labor se realiza de forma manual utilizando la herramienta Microsoft Excel 2003.

Toda información acumulada por el Supervisor diariamente ya sea en formato duro o digital, es guardad en la oficina del mismo, con el objetivo de coleccionarla para su posterior uso si fuera necesario, a lo cual con solo tener acceso a este local se pudiera tener acceso a esta información. Diariamente las compañías que prestan servicios tales como Lodo, Direccional, Mudlogin, Perforación y Registro, reportan una serie de datos o reportes su mayoría en horario nocturno, en caso de la compañía que brinda servicio direccional entrega un reporte con todas las herramientas y sus usos, utilizadas en las actividades de perforación, para así poder hacer el cierre del día de perforación y quedar registrada de forma ordenadas.

1.3.2 Descripción actual del dominio del problema.

La Dirección de Intervención y Perforación de Pozos (DIPP), es la encargada de dirigir y monitorear todos los procesos que se realizan en los pozos en perforación. Esta entidad se encuentra radicada en la Empresa de Producción y Extracción de Petróleo del Centro (EPEPC), perteneciendo a la Dirección de Exploración - Producción de la Unión de CUPET. Aquí se realizan

y se aprueban los proyectos de los pozos a perforar, los cuales se elaboran por un equipo multidisciplinario de diferentes ramas, siendo así la responsable de los pozos desde su preparación hasta la terminación y ensayo.

La DIPP es además la encargada de contratar las Compañías de Servicio que van a trabajar en el pozo. Se contratan compañías que brindan servicios de Direccionales, Lodo, Perforación (Contratistas), Registro, Geología. En Cuba se hace uso del sistema llamado WellWizard, el cual se utiliza para monitoreo en tiempo real de la perforación; este servicio lo presta la compañía mixta CUBALOB. Todas estas compañías trabajan en el pozo y cumplen un rol determinado en el proceso lo cual se refleja en el proceso de seguridad que debe cumplir el sistema. Los supervisores de pozo son los encargados de velar que todo el proceso de flujo de información interno y hacia la DIPP, se realice correctamente y sin errores.

En ninguna de las estaciones de trabajo existe protección alguna contra el acceso indebido o no autorizado a esta información; por lo cual puede ser borrada, copiada o modificada/actualizada erróneamente y es imposible tener conocimiento de quien lo hizo, dado que se trabaja con una sesión común para todos los usuarios. En el contexto del sistema a implantar en la DIPP y en los Pozos, se debe manejar la seguridad de manera tal que exista un conocimiento de los usuarios que tienen acceso a la información, delimitar niveles de acceso y conocer que, cuando y donde acceden los usuario en cada acción que realizan en el sistema.

1.4 Análisis de soluciones Existentes.

En la actualidad existen una gran gama de producciones de software para el sector del Petróleo, encaminadas a resolver disímiles problemáticas dentro de esta esfera. A nivel internacional están muy desarrolladas las aplicaciones relacionadas con distintas fases de desarrollo del crudo. La seguridad en cualquier software destinado a la empresa del petróleo debe ser la línea base para el desarrollo del mismo, un punto de control imposible de penetrar para aquellas personas las cuales no estén autorizados. Debido a la sensibilidad y cantidad de información que se maneja, la cual es vital tanto para el desarrollo del propio pozo como para futuros proyectos, garantizar la integridad y confidencialidad de los datos es imprescindible.

Estos sistemas deben ser capaces de gestionar usuarios, roles, permisos, que ayudaran a la manipulación de la información en dependencia del nivel que corresponda. Al igual de controlar el flujo de actividades de un usuario en el sistema. A continuación se presentan varios sistemas que existen y son aplicados a la rama de la perforación:

1.4.1 Well Wizard:

Sistema de muestreo de todos los factores que son fundamentales en la perforación de pozos como aire comprimido, temperatura de lodo, los cuales son monitoreados por un software que está actualmente instalado en todos los pozos en perforación de Cuba. Este sistema se conecta a un servicio el cual está configurado directamente con la Base de Datos donde se reciben los datos que son interpretados de los diferentes sensores ubicados en el equipo de perforación. Por esto al iniciar este sistema se debe elegir la dirección IP o nombre del ordenador que contiene este servicio. Para esto se necesita también si está configurado en el servidor un usuario y una contraseña. La principal desventaja de este sistema es que no tiene restricciones por usuarios o por roles para ver las diferentes variables que procesa (6).

1.4.2 WellView:

WellView® es un archivo de pozo corporativo completo. Desde la petición de perforar un pozo hasta el abandono, registra todos los cambios y operaciones a través del ciclo de vida del pozo. Con su poderoso estado mecánico, reportes y sistemas de búsqueda, WellView coloca la información en las manos de la gente que más la necesita sin las limitantes de archivos de papel tradicionales o reportes diarios. Peloton[1] comenzó el desarrollo de WellView en 1991, enfocados en las operaciones de producción del pozo y estado mecánico histórico. Para abarcar más en el ciclo de vida de un pozo, Peloton expandió este software para manejo de datos de perforación en 1997. Nuestra versión actual crece con la fortaleza de versiones pasadas mejorando el modelo de datos, herramientas de administración y capacidades analíticas (7).

1.4.3 WellSight:

El software WellSight son un conjunto de software utilizados para captación, informes, monitoreo, seguimiento, reconciliación y exportación de datos .Originalmente este software era para el desarrollo petrolífero en la Cuenca occidental de Canadá pero luego de ver sus beneficios se ha extendido su uso a diferentes usuarios del mundo. Dentro de sus diferentes funciones se hará alusión a la de exportación de datos (8).

Este sistema como muchos otros no maneja la seguridad basada en usuarios o roles, la seguridad proviene por defecto de la misma estación de trabajo donde este instalada, es decir la sesión del sistema operativo usado.

[1] Empresa canadiense que desarrolla sistemas para la industria del petróleo.

En esta investigación se han podido identificar tanto fortalezas como debilidades que tienen los software actualmente utilizados en el proceso de perforación.

Fortalezas:

1. Alta competitividad y confiabilidad.
2. Manipulan altos volúmenes de Información.

Debilidades:

1. Todas son aplicaciones de Escritorio y solo trabajan en Windows.
2. Todos trabajan la seguridad, pero solamente a nivel de usuario y contraseña, debido a la sesión del sistema operativo, no orientado a roles (niveles de acceso a la información).
3. No valoran la seguridad como elemento crítico en el funcionamiento estos sistemas, se concentran más en la integridad y disponibilidad de los datos, no así en su confidencialidad.

Por tales razones se demuestra la necesidad de que el sistema SIPP posea un subsistema de seguridad, que le permita delimitar el acceso a la información orientado a los niveles de acceso definidos en el negocio, así como poder monitorear la actividad de los usuarios, dado que cada uno tendrá un usuario y contraseña de uso personal. Esto será sin duda una ventaja que de este sistema sobre los sistemas que se utilizan en el mundo actualmente.

1.5 Enfoques de la Programación.

La programación web permite la creación de sitios dinámicos en Internet. Esto se consigue generando los contenidos de la aplicación o sistema a través de una base de datos mediante lenguajes de script como pueden ser PHP, ASP o ASP.NET. Con el uso práctico de este tipo de programación se podrá crear sitios dinámicos como periódicos digitales o tiendas virtuales.

El protocolo de transferencia de hipertexto (HTTP, *Hyper Text Transfer Protocol* (Protocolo de transferencia de Hiper Texto)) es el protocolo usado en cada transacción de la Web. Un protocolo se puede entender como el lenguaje utilizado por dos computadoras para comunicarse entre sí mediante la red. *Hyper Text*[2] se refiere al contenido de las páginas, escrito en un lenguaje especial HTML, los navegadores se comunican con los servidores de internet mediante este protocolo, estos le envían páginas en el lenguaje HTML, y a su vez el navegador las interpreta y muestra su contenido (9).

El lenguaje HTML es fundamental cuando se refiere a programación web en la construcción de cualquier sistema, gracias a que permite representar cualquier tipo de información contenida dentro de

[2] Hyper Text: Nombre que recibe el texto que en la pantalla de un dispositivo electrónico conduce a otro texto relacionado. (42)

una página web. Para este tipo de programación los lenguajes más utilizados del lado del cliente, es decir los interpretados por los navegadores son:

- ✓ HTML
- ✓ Javascript
- ✓ CSS

En cambio por parte de los servidores serian:

- ✓ PHP
- ✓ ASP
- ✓ ASP.NET
- ✓ JSP

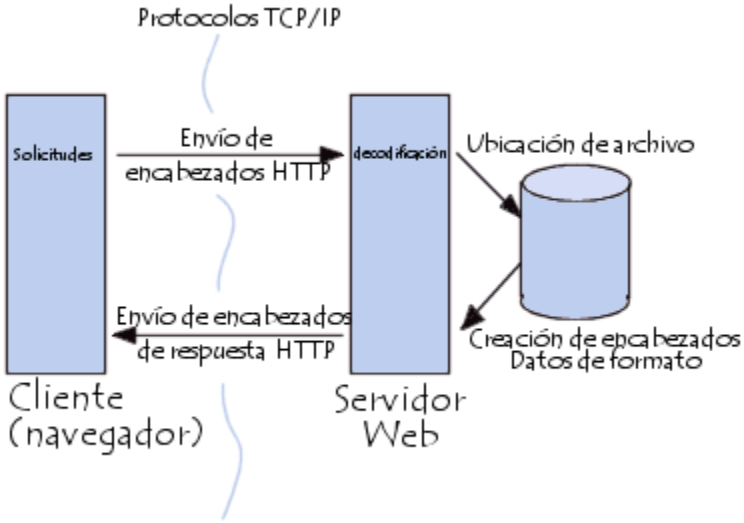


Figura 1: Comunicación entre el navegador y el servidor.

1.5.1 Subsistema Web.

Se entiende por sistema web o aplicación web, a un conjunto de páginas web (estáticas o dinámicas), entiéndase por página web estática a aquella en que es mostrada siempre al cliente de la misma forma sin importar las veces que sea llamada y por página web dinámica a la página cuyo contenido cambia en dependencia de los permisos de acceso que tenga el usuario que la solicita, de la cantidad de veces que sea llamada por el usuario, los parámetros de uso de la página, etc. (10)

Características:

- ✓ Se encuentra alojada en un Servidor Web.
- ✓ Son accesibles mediante el internet, usando un navegador web.
- ✓ La lógica del sistema se ejecuta en el servidor, mientras que el cliente solamente representa los datos.
- ✓ El acceso al sistema puede ser público o restringido.
- ✓ La actualización del sistema no afecta ni depende del cliente.
- ✓ Multiplataforma ya que pueden ejecutarse en cualquier Sistema.

Ventajas:

Los sistemas web tienen numerosas ventajas las cuales bien utilizadas, pudieran dar soluciones a problemas de cualquier empresa:

- ✓ Compatibilidad Multiplataforma: Los sistemas web funcionan de forma independiente a la plataforma que esté usando el cliente. Se cuenta con soporte para la mayoría de los sistemas operativos.
- ✓ Actualización: La actualización de los sistemas web se realiza sin la intervención del cliente, sin necesidad de interferir en su trabajo.
- ✓ Fácil Instalación: Para usar un sistema web no es necesario la descarga e instalación de este, el único requerimiento que piden estos sistemas es un navegador web y una conexión a internet.
- ✓ Bajo consumo de recursos: Toda o gran parte de la aplicación no se encuentra en el ordenador cliente, las tareas principales las ejecuta el ordenador servidor (11).

1.5.2 Estilos y estándares de programación.

Un estándar de codificación se define cuando se trabaja en un proyecto, de forma que los desarrolladores mantengan una estructura en el código del programa. Es decir defina la estructura y organización que deben seguir las instrucciones. Estos estilos de programación definen como deben quedar declaradas las clases, métodos, atributos, comentarios. Otros estándares especifican que datos deben comentarse acerca del desarrollador además de los cambios efectuados en el código fuente. El código de un programa lo leen muchas personas, bien para repararlo, bien para ampliarlo o, simplemente, para evaluarlo. Para estos lectores es fundamental que el código del programa esté bien organizado y estandarizado, con estilo, para que su significado sea claro. Existen una serie de reglas básicas que ayudan a conseguir un texto satisfactorio (12).

1.5.3 Framework de Desarrollo.

Un framework es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

En la actualidad el desarrollo de disimiles framework para la construcción de aplicaciones web ha crecido exponencialmente, permitiendo la rapidez de los procesos de desarrollo de los mismos, trayendo consigo la implementación y facilidad de uso de los patrones de diseño más comunes, favoreciendo una buena práctica de este tipo de programación. Además, de proporcionar una estructura al código fuente, también obliga al desarrollador a crear código más legible y más fácil de mantener. Proporciona que el proyecto se destine más hacia actividades del negocio, dejando a un lado las operaciones complejas, encapsulándolas en instrucciones sencillas.

Características:

- ✓ Abstracción de URL y Sesiones: No es necesario manipular las URL ni las Sesiones, el framework lo hace de forma interna.
- ✓ Acceso a datos: Incluyen las herramientas y clases necesarias para interactuar con la mayoría de la Base de Datos (PostgreSQL, Oracle, MySql, MS SQL, etc.).
- ✓ Control de peticiones: Cuentan con clases controladoras que se encargan de atender las peticiones hechas por los clientes, estas son altamente adaptables.
- ✓ Autenticación y Control de Acceso: Tienen mecanismos de control de autenticación y control de acceso automático, los que se pueden adecuar a las necesidades de los clientes (13).

1.5.4 Programación Orientada a Objetos (POO).

La programación orientada a objetos es un paradigma que utiliza objetos como elementos fundamentales en la construcción de la solución. Paradigma de la programación la cual utiliza los objetos y sus relaciones para el diseño de aplicaciones. Aumenta considerablemente la rapidez con la que se escriben estos programas, enfocada principalmente a la forma de pensar en la vida real, basando los elementos en objetos los cuales tienen propiedades y métodos. El elemento básico de este paradigma no es la función (elemento básico de la programación estructurada), sino un ente denominado objeto, que no es más que un conjunto complejo de datos y programas que poseen estructura y forman parte de una organización. Un objeto contiene varios datos bien

estructurados y pueden ser visibles o no dependiendo del programador y las acciones del programa en ese momento (14).

Características:

En el enfoque orientado a objeto (OO) las propiedades del objeto son claves. Los principios del modelo OO son:

- ✓ Abstracción: Es una descripción simplificada o especificación de un sistema que enfatiza algunos de los detalles o propiedades del sistema, mientras suprime otros.
- ✓ Encapsulación: En el proceso de ocultar todos los detalles de un objeto que no contribuyen a sus características esenciales.
- ✓ Modularidad: Es la propiedad de un sistema que ha sido descompuesto en un conjunto de módulos coherentes e independientes.
- ✓ Jerarquía o herencia: Es el orden de las abstracciones organizado por niveles.

En menor grado:

- ✓ Tipificación: Es la definición precisa de un objeto de tal forma que objetos de diferentes tipos no puedan ser intercambiados o, cuando mucho, puedan intercambiarse de manera muy restringida.
- ✓ Concurrencia: Es la propiedad que distingue un objeto que está activo de uno que no lo está.
- ✓ Persistencia: Es la propiedad de un objeto a través de la cual su existencia trasciende el tiempo (es decir, el objeto continua existiendo después de que su creador ha dejado de existir) y/o el espacio (es decir, la localización del objeto se mueve del espacio de dirección en que fue creado). (15)

También encontramos:

- ✓ Polimorfismo: comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre, al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. O dicho de otro modo, las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado. Cuando esto ocurre en "tiempo de ejecución", esta última característica se llama asignación tardía o asignación dinámica. Algunos lenguajes proporcionan medios más estáticos (en "tiempo de compilación") de polimorfismo, tales como las plantillas y la sobrecarga de operadores de C++.

- ✓ Herencia: las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo. Esto suele hacerse habitualmente agrupando los objetos en clases y estas en árboles o enrejados que reflejan un comportamiento común. Cuando un objeto hereda de más de una clase se dice que hay herencia múltiple (16).

1.6 Conclusiones Parciales.

Luego de haber abordado los aspectos esenciales para el total cumplimiento del problema planteado, se llegan a las siguientes conclusiones:

- ✓ El análisis profundo de las actividades realizadas tanto en la DIPP como en los pozos en perforación, posibilitó comprender la interacción de los actores del negocio con los datos, siendo esto decisivo para implementar el enfoque de seguridad orientado a roles.
- ✓ El estudio del estado del arte arrojó que, no existe un sistema que gestione la seguridad de la información basados en roles.

Capítulo 2: Descripción de las tecnologías.

2.1 Introducción.

Desde las edades más antiguas de la humanidad la creación de herramientas ha proporcionado un aumento en el desarrollo de la economía y recíprocamente, se han podido realizar nuevos descubrimientos científicos gracias al desarrollo de nuevas tecnologías, que han extendido las posibilidades de experimentación y adquisición del conocimiento. En el presente capítulo se describe y caracteriza de forma detallada las tecnologías y tendencias actuales utilizadas para el desarrollo del subsistema.

2.2 Lenguajes de Programación.

2.2.1 PHP 5

PHP (acrónimo de "PHP: *Hypertext Preprocessor*") es un lenguaje interpretado de alto nivel para el desarrollo de páginas Web dinámicas del lado del servidor, cuyos fragmentos de código se intercalan fácilmente en páginas HTML, debido a esto, y a que es de *Open Source* (código abierto), es el más popular y extendido en la Web (17).

Características:

- ✓ Al ser PHP un lenguaje libre, dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas Web dinámicas:
- ✓ Soporte para una gran cantidad de bases de datos: MysSQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras.
- ✓ Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de *Acrobat Reader*) hasta analizar código XML.
- ✓ Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- ✓ Soportado por una gran comunidad de desarrolladores, como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.
- ✓ El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.

- ✓ Con PHP se puede hacer cualquier cosa que podemos realizar con un script CGI, como el procesamiento de información en formularios, foros de discusión, manipulación de *cookies* y páginas dinámicas (18).

PHP incluye además de las versiones anteriores:

- ✓ Velocidad: No solo la velocidad de ejecución, la cual es importante, sino además no crear demoras en la máquina. Por esta razón no debe requerir demasiados recursos de sistema. PHP se integra muy bien junto a otro software, especialmente bajo ambientes Unix, cuando se configura como módulo de Apache, está listo para ser utilizado.
- ✓ Estabilidad: La velocidad no sirve de mucho si el sistema se cae cada cierta cantidad de ejecuciones. Ninguna aplicación es 100% libre de bug[3], pero teniendo de respaldo una increíble comunidad de programadores y usuarios es mucho más difícil para los bug sobrevivir. PHP utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando un sistema robusto y estable.
- ✓ Seguridad: El sistema debe poseer protecciones contra ataques. PHP provee diferentes niveles de seguridad, estos pueden ser configurados desde el archivo.ini
- ✓ Simplicidad: Se les debe permitir a los programadores generar código productivamente en el menor tiempo posible. Usuarios con experiencia en C y C++ podrán utilizar PHP rápidamente (19).

Luego de un estudio profundo por parte del arquitecto de software de los distintos lenguajes de programación para la web, se define utilizar PHP 5.3 para la programación del Subsistema de Seguridad como lenguaje de programación del lado del servidor, debido a que es un lenguaje libre, multiplataforma y soporta conexiones con las Base de Datos más utilizadas en la actualidad. PHP posee además amplia documentación, así como una fuerte comunidad de desarrollo, además es altamente utilizado en la UCI, con una comunidad interna la cual constantemente trabaja para profundizar en el uso y especialización del lenguaje, permite también el uso de la Programación Orientada a Objetos. Una de sus fortalezas es que al ser su código transparente al navegador y al cliente eleva las prestaciones de seguridad y confiabilidad, gracias a que todas las instrucciones se ejecutan por parte del servidor el cual solo envía respuestas en formato HTML.

2.2.2 Javascript

[3] Bug: Defecto de un software, se refiere a errores o problemas de funcionamiento en programas informáticos (40).

Lenguaje de programación interpretado, se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente (client-side[4]), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS) (20).

Características:

- ✓ Lenguaje Interpretado: No requiere compilación. El navegador del usuario se encarga de interpretar las sentencias JavaScript contenidas en una página HTML y ejecutarlas adecuadamente.
- ✓ Orientado a Eventos: Cuando un usuario acciona sobre un enlace o mueve el puntero sobre una imagen se produce un evento. Mediante JavaScript se pueden desarrollar scripts que ejecuten acciones en respuesta a estos eventos.
- ✓ Orientado a Objetos: El modelo de objetos de JavaScript está reducido y simplificado, pero incluye los elementos necesarios para que los scripts puedan acceder a la información de una página y puedan actuar sobre la interfaz del navegador (21).

Debido a que este lenguaje se ejecuta por el lado del cliente, nos facilita las validaciones comunes de formularios, como validar si un campo determinado debe ser un número real o entero, el formato de entrada de campo de correo, así como cálculos entre campos, aunque de todas formas deben hacerse del lado del servidor ya que desde el navegador puede ser deshabilitado la ejecución de scripts. La orientación a eventos facilita la creación y ejecución de efectos visuales, aumentando así la experiencia del usuario en las interfaces gráficas, se puede lograr que una aplicación web tenga algunas características de las aplicaciones de escritorio, tales como menús desplegables, componentes de tiempo como relojes, cronómetro, componentes que se muevan de lado a otro de la página, entre otros.

2.2.3 HTML

HTML, siglas de *HyperText Markup Language* (Lenguaje de Marcado de Hipertexto), es uno de los lenguajes de marcado más utilizado para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto de las páginas, así como para complementar el texto con objetos tales como imágenes. Se describe en forma de etiquetas (<…>), puede describir también hasta cierto punto la apariencia de un documento.

[4] Client-side: En español "Lado del cliente", se refiere a operaciones que son efectuadas por el lado del cliente en una arquitectura cliente-servidor.

Características:

- ✓ Las etiquetas pueden tener atributos los cuales definen las propiedades del elemento.
- ✓ Los espacios, tabulaciones, líneas en blanco y retornos de carro del documento HTML se ignoran, tomándose como un único espacio en blanco. Esto permite añadir espacios para aumentar la claridad del documento.
- ✓ No distingue entre mayúsculas y minúsculas. Cuando es importante hacerlo, como al poner un título o un atributo, hay que ponerlo entre comillas.
- ✓ HTML tiene unas reglas estructurales que indican dónde pueden y no pueden ir los elementos, esto obliga a tener un orden lógico en la escritura del código.
- ✓ Las etiquetas tienen que seguir un orden piramidal, las primeras que se abren son las últimas que se cierran (22).

El HTML es el lenguaje fundamental para la construcción de una aplicación web, aunque desde PHP también se puede generar páginas que el contenido interno será código HTML para que lo puedan interpretar los navegadores. Posee una amplia gama de elementos para poder básicamente colocar cada texto, imagen o video de forma lógica dentro de la vista. Cuando la creación de este lenguaje no se pensó que la web llegara q ser un área de ocio o con carácter multimedia, debido a esto no es posible solamente con el código HTML hacer uso de estos medios, es necesario utilizar componentes de terceros tales como Flash Player o Silverlight.

2.2.4 CSS

Hojas de Estilo en Cascada (*Cascading Style Sheets*), es un mecanismo simple que describe cómo se va a mostrar una página en la pantalla, o cómo va a ser presentada la información en la página a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

Características:

- ✓ Complementariedad con documentos estructurados.
- ✓ Independencia del vendedor, la plataforma y el dispositivo.
- ✓ Simplicidad.
- ✓ Rendimiento de la red.
- ✓ Flexibilidad.
- ✓ Riqueza.
- ✓ Combinación con lenguajes alternativos.

- ✓ Accesibilidad (23).

En otras palabras los estilos CSS nos ayudan mucho más que el HTML estático ha maquetar nuestras páginas, una de sus grandes ventajas es que al ser archivos que contienen las llamadas reglas de estilo, pueden ser reutilizables en cualquier lugar de nuestra aplicación o sistema, disminuyendo el tiempo de evolución de las interfaces.

2.3 Entorno de Desarrollo Integrado.

2.3.1 Zend Studio 6.0

Se trata de un programa de la casa Zend, impulsores de la tecnología de servidor PHP, orientada a desarrollar aplicaciones Web en lenguaje PHP. El programa, además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código. El programa está escrito en Java, lo que le ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows, Linux y MacOS.

Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. La del cliente contiene el interfaz de edición y la ayuda. Permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor que instala Apache y el módulo PHP (24).

Características:

- ✓ Soporte para PHP 4 y PHP 5.
- ✓ phpDoc integrado.
- ✓ Plegado de código (comentarios, bloques de *phpDoc*, cuerpo de funciones y métodos e implementación de clases).
- ✓ Sangrado automático y otras ayudas de formato de código.
- ✓ Funciones de depuración: Botón de ejecución y traza, marcadores, puntos de parada (breakpoints), seguimiento de variables y mensajes de error del intérprete de PHP.
- ✓ Instalación de barras de herramientas para Internet Explorer y Mozilla Firefox (opcional).
- ✓ Soporte para gestión de grandes proyectos de desarrollo.
- ✓ Soporte para control de versiones usando CVS o Subversion.
- ✓ Soporte para navegación en bases de datos y ejecución de consultas SQL (20).

El uso de este Entorno de Desarrollo facilita la documentación de clases, métodos, propiedades, variables, por parte de los desarrolladores, su fácil integración con un control de versiones para el trabajo del equipo de desarrollo. Una de las funcionalidades más fuertes que presenta es la depuración del código, siendo esto posible mediante puntos de interrupción en instrucciones que pudieran estar en prueba por los mismos desarrolladores. Brinda un completamiento de código completo y de fácil entendimiento, donde se puede observar la documentación incrustada de los elementos llamados.

2.4 Framework de Desarrollo.

2.4.1 Symfony.

Symfony es un framework diseñado para la realización de aplicaciones Web de forma óptima desde las más sencillas hasta las más robustas, enfocado principalmente en la arquitectura MVC (Modelo, Vista, Controlador). Además cuanta con herramientas y clases que aseguran la rapidez y la disminución del tiempo de desarrollo. Automatiza acciones comunes como realizaciones de CRUD (*Create* - Crear, *Update* - Actualizar, *Delete* - Eliminar) sobre una o varias tablas de la Base de Datos en dependencia del Modelo de Objeto Relacional (ORM – *Object Relational Model*) utilizado, como *Propel* y *Doctrine*.

Symfony está desarrollado completamente en PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Se puede utilizar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows (25).

Symfony fue diseñado para ajustarse a los siguientes requisitos:

- ✓ Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- ✓ Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de Propel, permiten cambiar con facilidad de SGBD en cualquier fase del proyecto.
- ✓ Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- ✓ Sencillo de usar en la mayoría de casos, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos.
- ✓ Aunque utiliza MVC (Modelo Vista Controlador), tiene su propia forma de trabajo en este punto, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.

- ✓ Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador sólo debe configurar aquello que no es convencional.
- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ Preparado para aplicaciones empresariales, adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- ✓ Código fácil de leer que incluye comentarios de *phpDocumentor* y que permite un mantenimiento muy sencillo.
- ✓ Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes (25).

Producto a que el Subsistema será desarrollado en PHP, Symfony es la mejor opción debido a que por el uso de su ORM, orienta toda la Base de Datos a objetos, lo cual tiene elimina las inyecciones SQL, soporta las base de datos más usadas hoy en día, en caso de ocurrir algún cambio en el gestor de base de datos esto no afectaría en nada el funcionamiento de la aplicación. Además de ser libre, fue diseñado principalmente para sistemas web de alta complejidad y cuenta también con una gran comunidad de desarrollo.

2.5 Estilos y Estándares de Codificación.

Symfony utiliza sus propios estándares de codificación:

- ✓ Nunca use las tabulaciones en el código. La sangría se hace por pasos de 2 espacios:

```php
<?php
class sfFoo
{
  public function bar()
  {
    sfCoffee::make();
  }
}
```

- ✓ No ponga espacios después de un paréntesis de apertura y antes de un cierre:

```php
<?php
if ($myVar == getRequestValue($name))     // correcto
if ( $myVar == getRequestValue($name) )   // incorrecto
```

- ✓ Llaves van siempre en su propia línea.

- ✓ Utilice llaves para indicar la estructura de control del cuerpo, independientemente del número de declaraciones que contiene.

- ✓ Symfony está escrito en PHP5, por lo que cada método de definición de clase o miembro debe declarar explícitamente su visibilidad utilizando las palabras clave privada, protegida o pública.

- ✓ En un cuerpo de la función, las declaraciones de retorno debe tener una línea en blanco antes de él para aumentar la legibilidad.

- ✓ Todo lo que uno comente en una línea debe estar en este formato:

```php
<?php
// space first, with no full stop needed
```

- ✓ Evite la evaluación de las variables dentro de cadenas, en lugar de optar por la concatenación.

- ✓ Utilice minúsculas constantes de PHP: falso, verdadero y nulo. Lo mismo ocurre con array (). Al contrario, siempre use cadenas en mayúsculas con las constantes definidas por el usuario, como la define ('MY_CONSTANT', 'foo / bar "). Mejor, trate de utilizar siempre las constantes de clase:

```php
<?php
class sfCoffee
{
  const HAZ_SUGAR = true;
}
var_dump(sfCoffee::HAZ_SUGAR);
```

- ✓ Para comprobar si una variable es nula o no, no utilice la función de PHP is_null ():

```php
<?php
if (null !== $coffee)
{
  echo 'I can haz coffee';
```

- ✓ Formato de archivo:

  Para aquellos archivos que contienen sólo código PHP los tags de demarcación ("<? ") no estarán permitidos, además no es requerido por PHP y omitirlos nos previene de algún accidente ocasionado por un espacio en blanco.

- ✓ Parámetros:

Los parámetros van siempre en minúsculas.

- ✓ Variables:

  El nombre de las variables debe estar compuesto de caracteres alfanuméricos, el carácter "Underscore" está permitido. Siempre tiene que comenzar con letra minúscula. Además siempre debe inicializarse y sobre todo deben tener nombres significativos.

- ✓ String literales:

  Cuando se le asigna un texto literal (sin contenido de variables) se utilizarán comillas dobles.

```php
<?php
$a = "Texto de ejemplo";
```

- ✓ Concatenación:

  Para concatenar Strings se utilizará el operador "." (Punto), con un espacio entre medio para mejorar la lectura:

```php
<?php
$company = 'Zend' . '' . 'Technologies';
```

- ✓ Control de flujo:

  En las declaraciones if/then/else deberá tener un espacio antes y después del paréntesis condicional, lo mismo se aplica al elseif, a continuación un ejemplo que lo ilustra:

```php
<?php
if ($a != 2)
{
  $a = 2;
}
elseif ($a == 3)
{
  $a = 4;
}
else
{
  $a = 6;
}
```

- ✓ Documentación:

  Las complicadas funciones y métodos deberán tener un bloque de documentación. El mismo será entre /**/ cuando sean de 2 líneas en adelante y // cuando sea una sola línea (12).

El propio Symfony fuerza al desarrollador a crear código legible, fácil de entender y mantener. Por tal motivo se hará uso de estos estándares de codificación definidos por los desarrolladores de este framework.

2.6 Sistema Gestor de Base de Datos.

Un Sistema Gestor de base de datos (SGBD) es un conjunto de programas que permiten crear y mantener una Base de datos, asegurando su integridad, confidencialidad y seguridad. Por tanto debe permitir:

- ✓ Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- ✓ Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- ✓ Manipular la base de datos: realizar consultas, actualizarla, generar informes (26).

2.6.1 PostgreSQL 8.4

PostgreSQL es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD. El desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre, y/o apoyados por organizaciones comerciales.

Este gestor soporta:

- ✓ Alta concurrencia.
- ✓ Claves Ajenas o ForeignKey
- ✓ Disparadores o Triggers.
- ✓ Vistas.
- ✓ Herencia de tablas.
- ✓ Operaciones Geométricas.
- ✓ Amplia variedad de tipos nativos:
    - Números de precisión arbitraria.
    - Texto de largo ilimitado.
    - Figuras geométricas (con una variedad de funciones asociadas).
    - Direcciones IP (IPv4 e IPv6).
    - Bloques de direcciones estilo CIDR.
    - Direcciones MAC.
    - Arrays (Arreglos).
    - Además que los usuarios pueden crear sus tipos de datos personalizados (27).

Características:

- ✓ Atomicidad (Indivisible) es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.
- ✓ Consistencia es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la base de datos.
- ✓ Aislamiento es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generará ningún tipo de error.
- ✓ Durabilidad es la propiedad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema.
- ✓ Es compatible con la mayoría de las familias y sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows, etc.
- ✓ Documentación muy bien organizada, pública y libre, con comentarios de los propios usuarios.
- ✓ Comunidades muy activas, varias comunidades en castellano.
- ✓ Altamente adaptable a las necesidades del cliente.
- ✓ Soporte nativo para los lenguajes más populares del medio: PHP, C, C++, Perl, Python, etc.
- ✓ Soporte de protocolo de comunicación encriptado por SSL (28).

2.7 Servidor Web.

Un servidor es un tipo de software que realiza ciertas tareas en nombre de los usuarios. El término servidor ahora también se utiliza para referirse al ordenador físico en el cual funciona este software, una máquina cuyo propósito es proveer datos de modo que otras máquinas los puedan utilizar. Un servidor web sería entonces un software especial el cual usa http como protocolo de intercambio para enviar páginas HTML y archivos asociados, al ordenador del usuario que ha iniciado la petición.

2.7.1 Apache.

Apache es un programa de servidor HTTP Web de código abierto. Actualmente es uno de los servidores más utilizados en la red. Software libre además de ser multiplataforma, también con soporte de base de datos SQL.

Características:

- ✓ Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- ✓ Apache es una tecnología gratuita de código fuente abierto.
- ✓ Apache es un servidor altamente configurable de diseño modular.
- ✓ Soporta Perl, PHP y otros lenguajes de script.
- ✓ Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
- ✓ Fuertemente configurable en la creación y gestión de logs.
- ✓ Modular: Puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de módulos, para el desarrollo de módulos específicos.
- ✓ Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor (29).

2.8 Técnicas de aseguramiento de la calidad.

La calidad del software es el conjunto de cualidades que lo caracterizan y que determinan su utilidad y existencia. La calidad es sinónimo de eficiencia, flexibilidad, confiabilidad, usabilidad, seguridad e integridad. El aseguramiento de calidad del software es el conjunto de actividades planificadas y sistemáticas necesarias para aportar la confianza que el producto requiere para satisfacer los requerimientos dados por parte del cliente (30).

Existen diferentes métodos o técnicas para asegurar la calidad de un software:

- ✓ Prácticas colaborativas: Son aquellas actividades que, realizadas por equipos de desarrollo y sin poner en juego jerarquías, se utilizan para mejorar la calidad del desarrollo de software, funcionan mejor si la construcción del sistema también es de forma colaborativa.

- ✓ Repositorios Compartidos: Es fundamental contar con un repositorio único de código y documentación, utilizando herramientas automatizadas de control de versiones. Un repositorio centralizado debe tener, al menos, funcionalidades para poder actualizar código fuente de más de un origen y dar marcha atrás en caso de necesitarlo, hacia cualquier versión anterior.

- ✓ Integración continua: Es una de las prácticas más fuertes, que básicamente se ha interpretado como recomendación de integración y prueba de humo una vez al día. La idea es que al día, corridos ya todas las pruebas unitarias, se integre, se compile y se corran las pruebas de integración, todo de forma automatizada. Si en algún momento se encuentra un problema, la prioridad es solucionarlo antes de seguir, de modo tal que las pruebas del día siguiente corran sin problemas heredados.

- ✓ Pruebas de aceptación parciales: Cada vez es más común hacer pruebas de aceptación en forma frecuente, ya que si tenemos un cliente junto con el equipo de desarrollo, lo recomendado es que al igual que en la integración continua se realicen también pruebas de aceptación una vez al día. De hecho, es una buena práctica tener un producto lo más entregable posible en cualquier momento dado, y para eso es fundamental contar con estas pruebas (31).

2.9 Conclusiones Parciales.

El término de este capítulo dedicado a describir las técnicas de calidad, tecnologías y herramientas a utilizar para lograr el resultado obtenido, se llegan a las siguientes conclusiones:

1. Los sistemas actuales presentan deficiencia en la gestión de la seguridad y en el atributo de portabilidad.
2. El uso de técnicas de aseguramiento de la calidad, contribuye a la calidad del resultado.
3. La integración de tecnologías y herramientas del software libre, permiten tener una infraestructura fuerte y sostenible para el desarrollo de SIPP.

Capítulo 3: Enfoque de la Solución Propuesta.

3.1 Introducción.

En este capítulo se presenta el Subsistema de Seguridad así como los módulos por los que está compuesto y los requisitos funcionales asociados a los casos de uso. Se dan a conocer los conceptos del modelo de implementación y componente. Se exponen los diagramas de implementación junto con la vista de implementación, además se definen cada uno de los componentes que forman parte del modelo.

3.2 Requerimientos.

Según la IEEE (*Standard Glossary of Software Engineering Terminology*) un requerimiento es la condición o capacidad que debe poseer un sistema o un componente de un sistema para satisfacer un contrato, un estándar, una especificación u otro documento formalmente impuesto (31). Es la idea de que debe hacer o cómo debe funcionar el sistema, son las necesidades que tendrá un producto, que puede ser sugerido por el mismo cliente o por el equipo de desarrollo.

Los requerimientos funcionales definen las funciones que el sistema será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas para producir salidas, mientras que los requerimientos no funcionales tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (32). Los Requerimientos Funcionales (RF) y no Funcionales (RNF) implementados en el Subsistema se manifiestan a continuación.

3.2.1 Requerimientos Funcionales.

Estos son los requerimientos funcionales para la primera versión del Subsistema de Seguridad.

- ✓ **RF1:** Gestionar Usuario, este requisito es uno de los encargados de la seguridad del sistema; tiene como funcionalidad el trabajo con los datos de los usuarios autorizados a la manipulación de la información en el sistema.
  - **Insertar Usuario:** para ingresar un nuevo usuario al sistema se requiere de un formulario que contenga los campos nombre de usuario ,usuario, CI, sexo, área de trabajo (Pozo, Dipp) y el rol a desempeñar dentro del sistema.
  - **Actualizar Usuario:** Para modificar un usuario existente se debe cargar un formulario con los datos registrados del usuario.

- **Eliminar Usuario:** Para eliminar un usuario debe existir un formulario que muestre a todos los existentes en el sistema y que pueda ser seleccionado aquel que desea eliminarse.

✓ **RF2:** Gestionar Rol, tiene como funcionalidad la creación de nuevos roles, actualización de los ya creados, así como la eliminación de roles que sean desechados por parte del Administrador del Sistema.

- **Crear Rol:** Para crear un nuevo rol en el sistema se requiere de un formulario que contenga los campos nombre del rol, rol, lugar (Pozo, Dipp).
- **Actualizar Rol:** para actualizar un rol existente se debe cargar un formulario para seleccionar el rol a modificar.
- **Eliminar Rol:** para eliminar un rol, debe existir un formulario que muestre a todos los existentes en el sistema y que pueda ser seleccionado aquel que desea eliminarse.

✓ **RF3:** Autenticar Usuario, proporciona el acceso inicial al sistema, solo a aquellos usuarios registrados en el mismo, para lo cual, se necesita un formulario con los campos usuario y contraseña. Para poder acceder al sistema todos los usuarios deben estar autenticados. En el caso de que el usuario intente acceder alguna información no autorizada, el sistema debe denegar el acceso.

✓ **RF4:** Cambiar Contraseña, posibilita cambiar la contraseña por parte del mismo usuario, para lo cual, se necesita un formulario con los campos usuario, contraseña anterior, nueva contraseña, confirmar nueva contraseña.

✓ **RF7:** Asignar a Pozo, se encarga de asignar a un usuario que ya esté agregado al sistema, con un rol determinado, a un pozo, para el manejo de información en el mismo (33).

Además de los requisitos funcionales descritos, se identificaron dos más los cuales fueron pospuestos para la próxima versión de este subsistema. Estos están asociados al control de las actividades de los usuarios dentro del sistema y la gestión de nuevos espacios de trabajo que aumentará la escalabilidad del sistema en el tiempo.

3.3 Diagrama de Caso de Uso del Sistema.

Los diagramas de Casos de Uso del sistema son ubicados dentro de los diagramas de comportamiento de UML y constituyen una representación gráfica de los procesos y su interacción con los actores, se utilizan para ilustrar los requisitos del sistema, al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo.

3.3.1 Diagrama de Caso de Uso del Sistema.

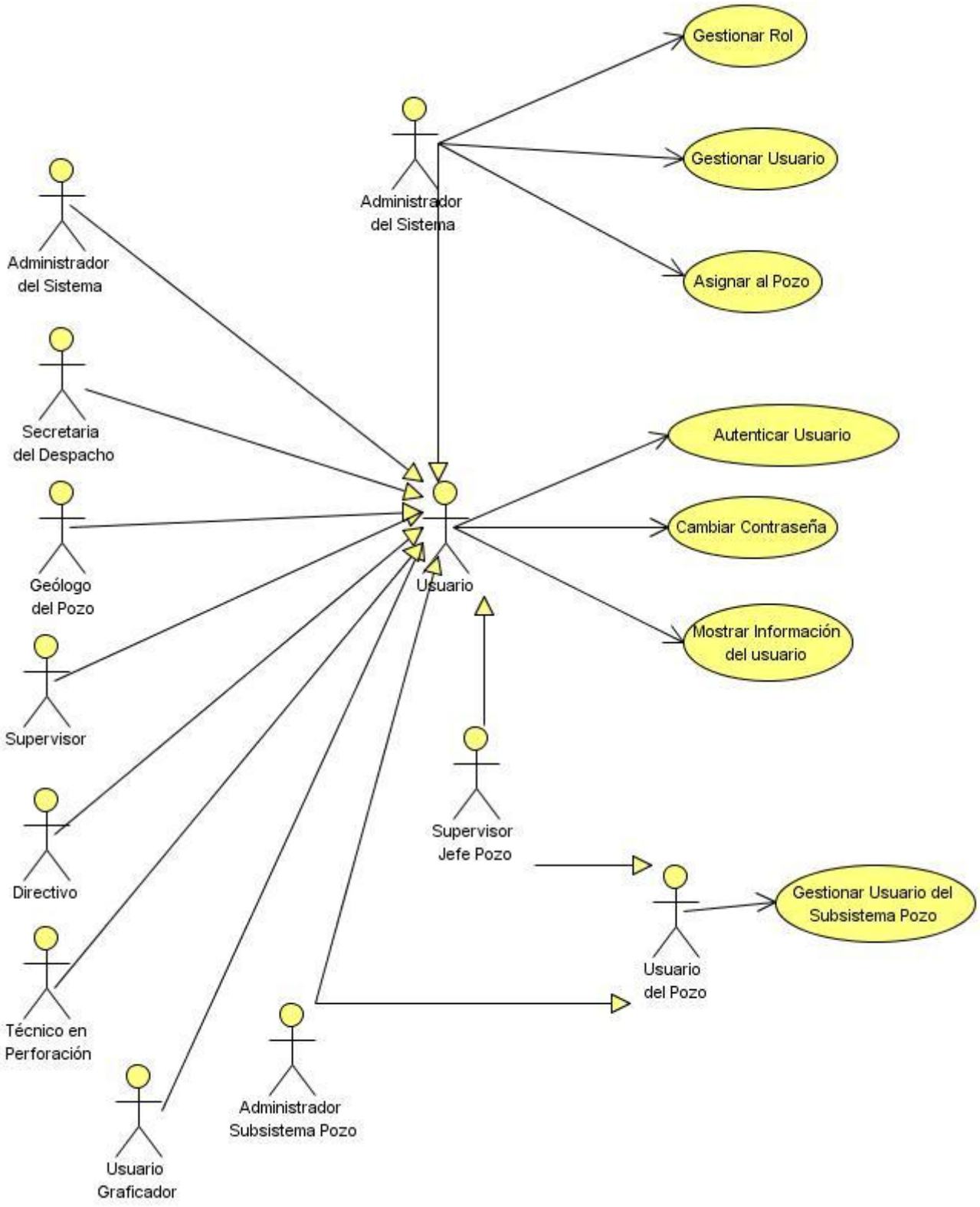


Figura 2: Diagrama de Caso de Uso del Sistema.

3.4 Subsistema de Seguridad.

3.4.1 Seguridad en la Web.

La seguridad en un sistema web debe ser una línea horizontal, abarcando incluso las partes públicas que pudiera contener. La misma debe incluirse dentro del diseño del propio sistema, definiendo, implementado y probando controles necesarios. Se debe otorgar el acceso solo al personal autorizado con los privilegios mínimos que le permitan cumplir con sus funciones. También la implementación de controles de acceso para evitar la ejecución de funciones por parte de usuarios no autorizados o con niveles de permisos insuficientes.

Existen varias vías para infiltrarse en una aplicación web, por esto es necesario un nivel de seguridad el cual permita que se cumplan con los principios y conceptos de seguridad informática además de implementar los mecanismos adecuando tales como la autenticación y autorización. Este Subsistema de Seguridad logra cumplir con lo mencionado anteriormente, para esto muestra interfaces al usuario donde se le explica las razones por las cuales no puede acceder al sistema y posibles soluciones a tener en cuenta, estas son algunas de las acciones que ocurren al violar algunos de estos principios:

- ✓ Al entrar a una dirección válida, a la cual se necesita estar autenticado

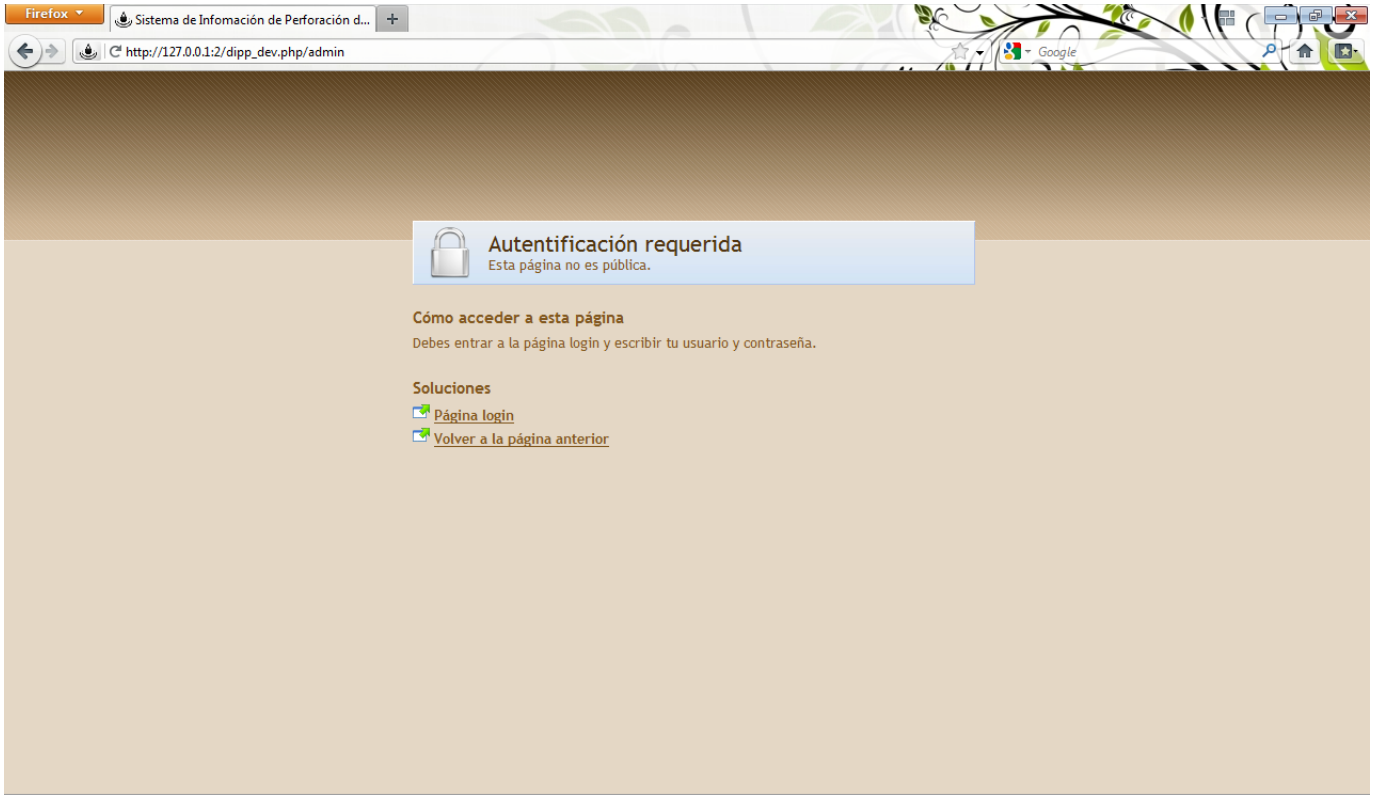


Figura 3: Mecanismo de Autenticación.

- ✓ Al estar autenticado e intenta acceder a otra dirección donde se encuentran recursos a los cuales no tiene permisos

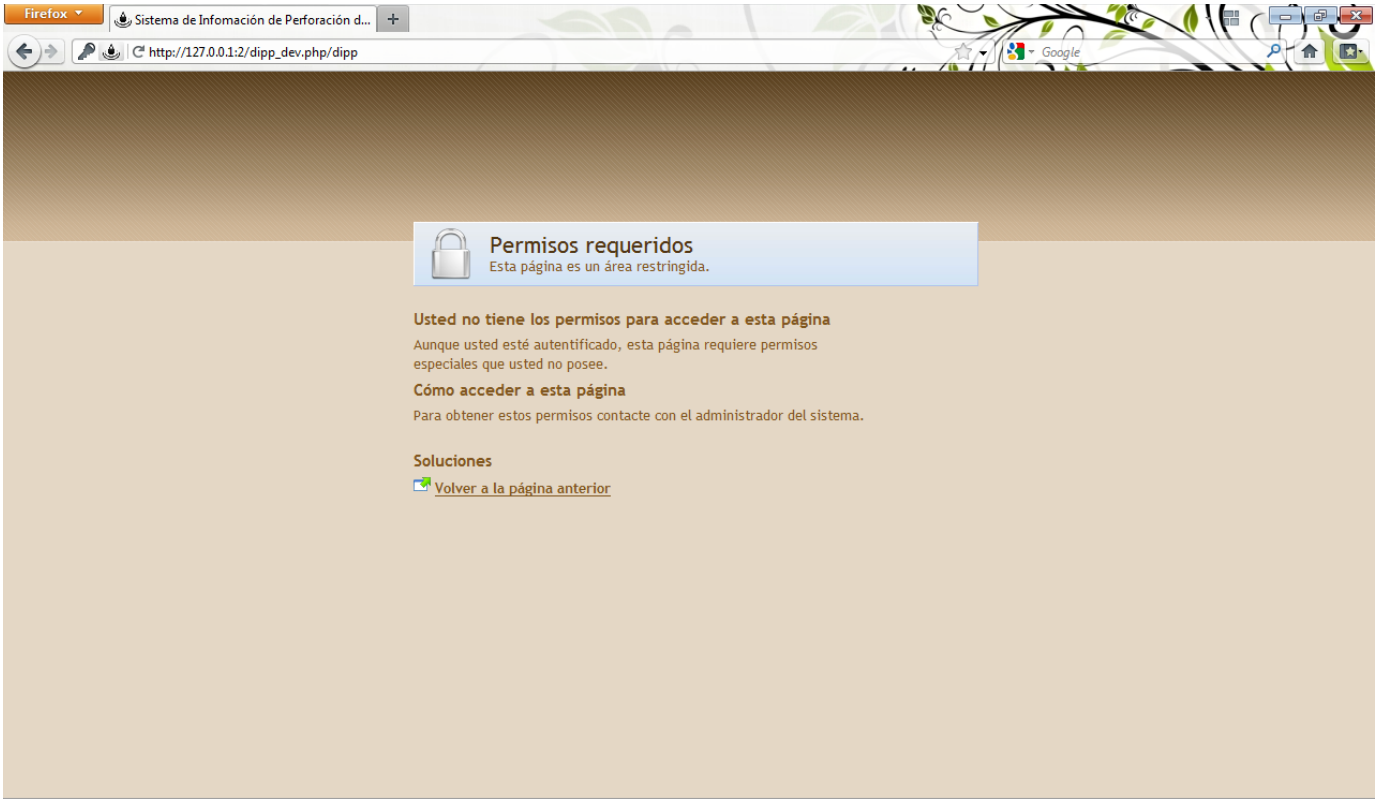


Figura 4: Mecanismo de Autorización.

3.4.2 Características del Sistema de Manejo Integral de Perforación de Pozos (SIPP).

SIPP es una aplicación o sistema distribuido por contar con dos aplicaciones separadas espacialmente, una llamada Dipp la cual se instala en la empresa donde radica la Dirección de Intervención y Perforación de Pozos y otra llamada Pozo la cual es instalada en los pozos que estén en perforación, esto nace de la necesidad del cliente de gestionar toda la información consolidada en el proceso de perforación de pozos. Se puede decir de forma general que la aplicación **Pozo** se centra en la manipulacion de toda la información atómica referente a las actividades que se realizan en un pozo de perforación vinculadas directamente con el negocio, tales como la gestión de los inventarios de herramientas direccionales; productos químicos, control del combustible, gestión de las operaciones diarias; metraje; costo; actividades; torque y arrastre, asi como la automatización de los reportes que intervienen cada una de las variables mencionadas anteriormente. En esta aplicación existen dos estaciones de trabajo o módulos Supervisor y Geólogo, significa que para que el usuario pueda tener acceso a esta aplicación debe tener rol de Supervisor o rol de Geólogo o ambos en conjunto.

La aplicación **DIPP** se enfoca más en la automatización de los reportes y partes de perforación que anteriormente se hacían de forma manual con toda la información recogida de los pozos, además de brindar la posibilidad de exportar estos reportes en formato excel. Es donde se gestionan todos los datos iniciales necesarios para el proceso de perforación, conocidos como los nomencladores, asi como también toda la seguridad del sistema es decir usuarios, roles, permisos. Esta aplicación esta conformada por tres espacios de trabajo Dipp, Nomencladores, Administración. La siguiente tabla muestra la relacion de los roles que debe tener un usuario para poder acceder a una de estas estaciones de trabajo:

Tabla 1: Permiso de los roles a los módulos.

Rol/Estación de Trabajo	Dipp	Nomencladores	Administración
Directivo	X		
Secretaria	X		
Técnico de Perforación	X		
Administrador de Nomencladores		X	
Administrador del Sistema			X

Ya que el Subsistema de Seguridad genera los archivos de configuración de seguridad para cada módulo, entonces cabe preguntarse, si la estación de trabajo de Administración es la que permite establecer los permisos de los roles para acceder a los módulos, y esta se encuentra en la aplicación Dipp, entonces ¿Cómo se establecen los permisos para los módulos o estaciones de trabajo que se encuentran en la aplicación Pozo si ambas están separadas físicamente?

Para darle respuesta a esta interrogante hay que tener en cuenta como es la relación que existe entre usuario, rol y módulo. Un usuario puede tener de uno a varios roles, un rol puede acceder a uno a varios módulos, por lo tanto un usuario podrá acceder a varios módulos en dependencia de sus roles. Por ejemplo si en la estación de trabajo de Administración de la aplicación Dipp, se le

realiza una modificación al rol Supervisor diciendo que ya no accede al espacio del Supervisor, ahora accede al del Geólogo, el sistema trata de buscar los ficheros security.yml de los módulos afectados, pero al pertenecer a la aplicación Pozo que se encuentra en los pozos de perforación entonces el cambio solo se vería afectado a nivel de Base de Datos.

Ya que SIPP trabaja la comunicación mediante de réplica de Base de Datos estos cambios también se verían reflejados en las Bases de Datos de los demás Pozos, por lo que los ficheros de seguridad se generan en el momento antes del usuario entrar en el sistema, desde la misma aplicación Pozo, logrando así la comunicación de ambas aplicaciones en cuanto a cambios realizados en la configuración de seguridad del sistema, todo de forma transparente al usuario.

3.4.3 Módulo de Inicio.

En este módulo es donde se encuentran los casos de usos relacionados con la entrada de los usuarios al sistema, la asignación de credenciales que se les son otorgadas, así como también el cambio de contraseñas de los mismos.

Está compuesto por los siguientes CU:

- ✓ Autenticar Usuario.

Figura 5: Interfaz CU Autenticar Usuario.

- ✓ Cambiar Contraseña.

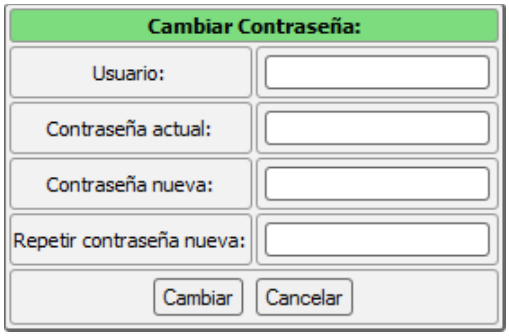


Figura 6: Interfaz CU Cambiar Contraseña.

3.4.4 Módulo de Administración.

Este módulo es el encargado de gestionar toda la información referente a usuarios y roles, también es donde se llevan a cabo todas las operaciones referentes a el establecimiento de permisos que tendrán los roles para acceder a los otros módulos tanto en la aplicación de la **DIPP** como la del **Pozo**, así como también de asignar un usuario a un pozo determinado.

Está compuesto por los siguientes CU:

- ✓ Gestionar Usuario.
  - Adicionar Usuario

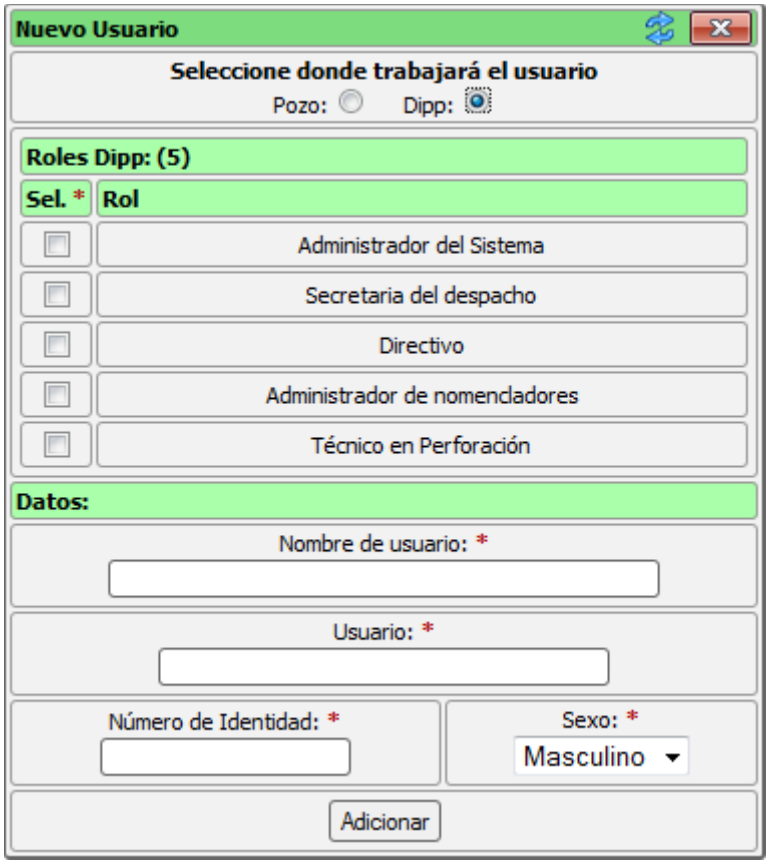


Figura 7: Interfaz CU Gestionar Usuario, Adicionar Usuario.

- Eliminar Usuario

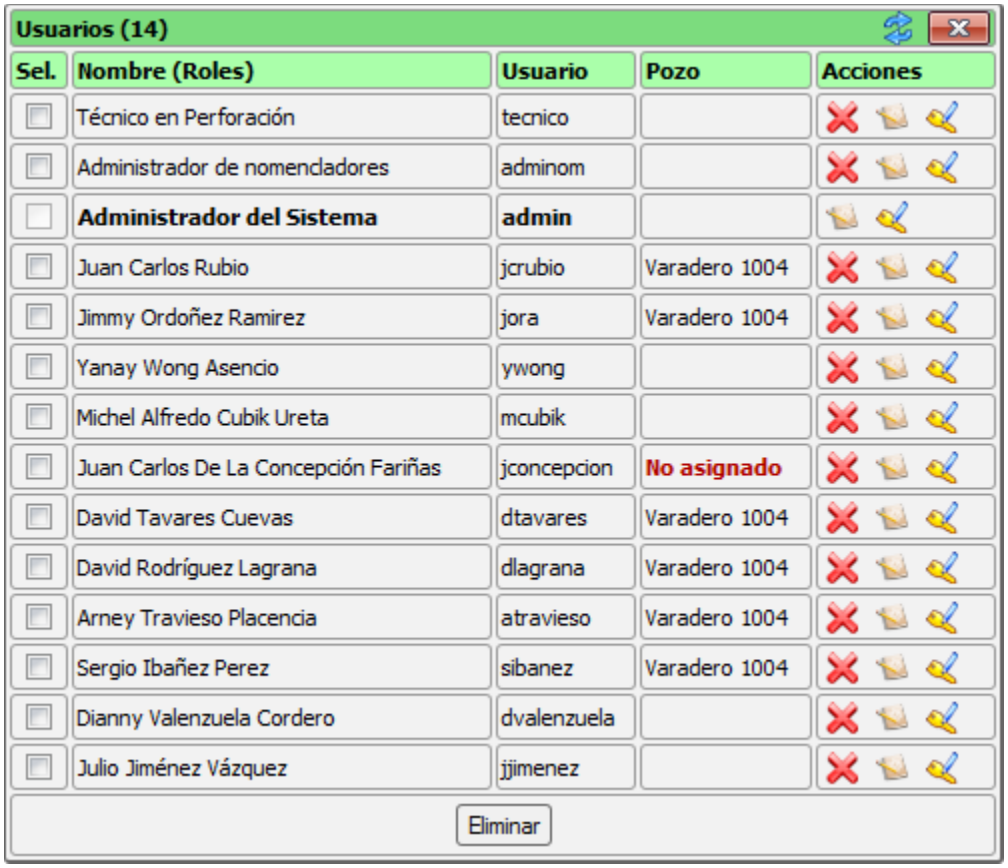


Figura 8: Interfaz CU Gestionar Usuario, Eliminar Usuario.

- Modificar Usuario

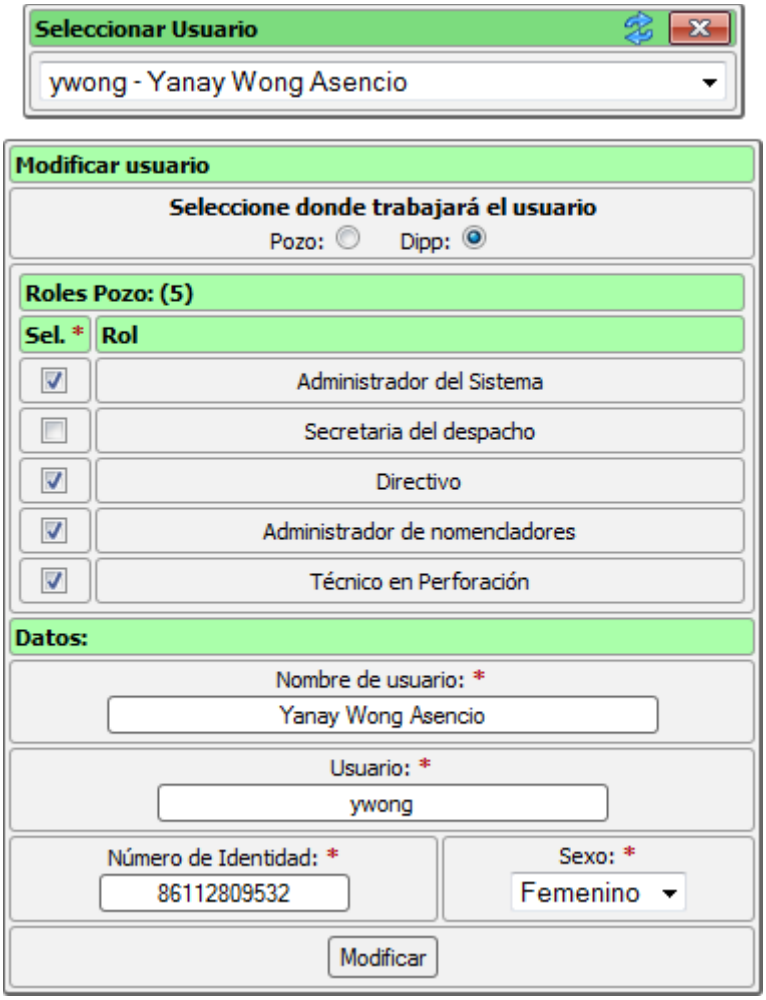


Figura 9: Interfaz CU Gestionar Usuario, Modificar Usuario.

- ✓ Gestionar Rol.
  - Adicionar Rol

Figura 10: Interfaz CU Gestionar Rol, Adicionar Rol.

- Eliminar Rol

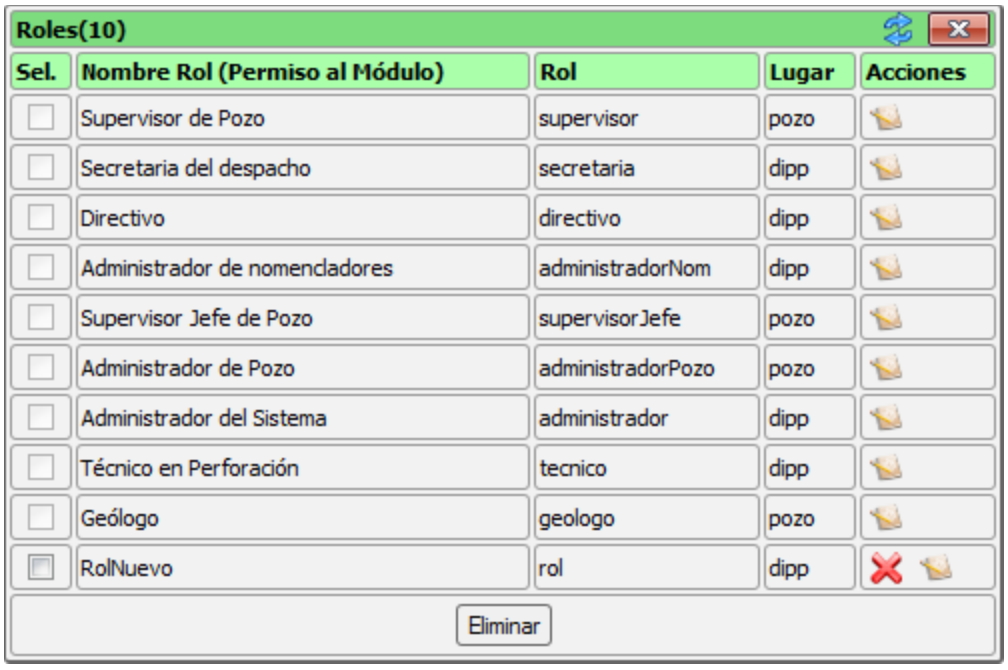


Figura 11: Interfaz CU Gestionar Rol, Eliminar Rol.

- Modificar Rol

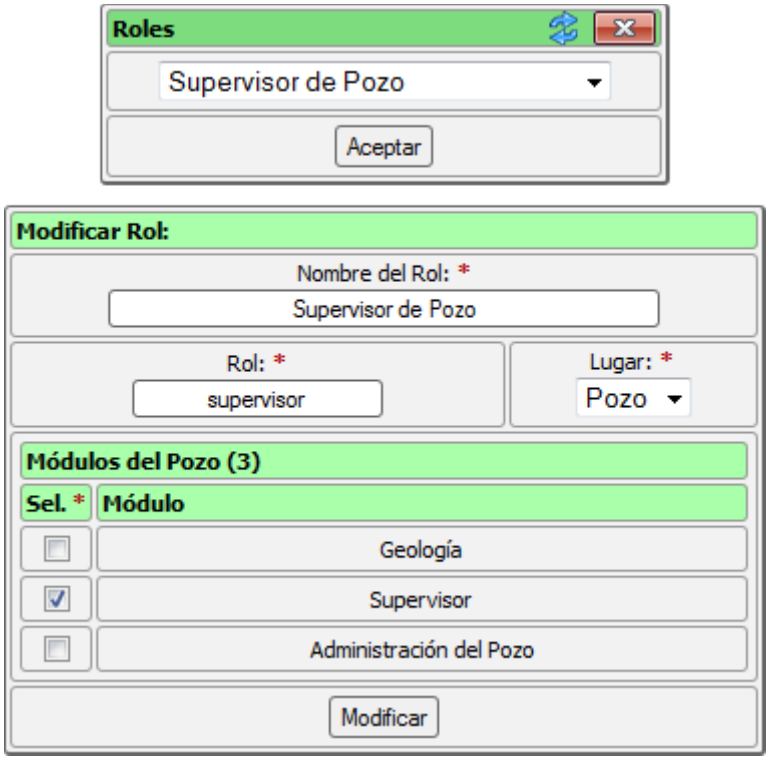


Figura 12: Interfaz CU Gestionar Rol, Modificar Rol.

- ✓ Asignar a Pozo.

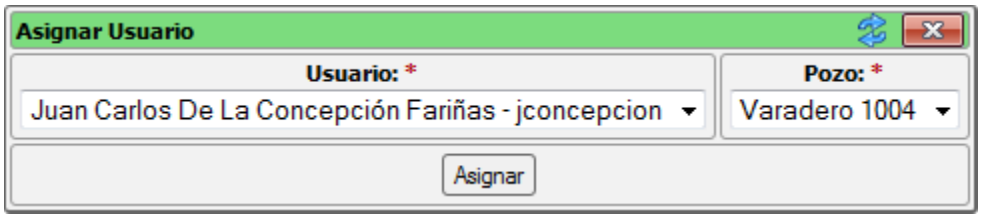


Figura 13: Interfaz CU Asignar a Pozo.

3.4.5 Trabajando la Seguridad con Symfony.

Symfony como framework de desarrollo para sistemas web, propone una estructura de seguridad similar a la estructura de carpetas por las cuales está formado. La raíz de cualquier proyecto desarrollado con Symfony contiene los siguientes directorios:

- ✓ *apps*: Contiene un directorio por cada aplicación del proyecto (en este caso *dipp* y *pozo*).
- ✓ *config*: Almacena la configuración general del proyecto.

- ✓ *lib*: Almacena las clases y librerías externas. Se guardar todo el código común a todas las aplicaciones del proyecto. El subdirectorio model guarda el modelo de objetos del proyecto, que es generado por la ORM Propel.
- ✓ *web*: Contiene la raíz del servidor web, los únicos archivos accesibles desde Internet.

También para cada aplicación existe una estructura de carpetas (apps/nombre_aplicacion/):

- ✓ *config*: Contiene un montón de archivos de configuración creados con YAML. Aquí se almacena la mayor parte de la configuración de la aplicación.
- ✓ *lib*: Contiene las clases y librerías utilizadas exclusivamente por la aplicación.
- ✓ *modules*: Almacena los módulos que definen las características de la aplicación.
- ✓ *templates*: Contiene las plantillas globales de la aplicación, es decir, las que utilizan todos los módulos.

Cada aplicación contiene uno o más módulos. Cada uno tiene su propio subdirectorio dentro de **modules**, así quedaría su estructura (**apps**/nombre_aplicacion/**modules**/nombre_modulo/):

- ✓ *actions*: Normalmente contiene un único archivo llamado *actions.class.php* y que corresponde a la clase que almacena todas las acciones del módulo.
- ✓ *config*: Puede contener archivos de configuración adicionales con parámetros exclusivos del módulo (*security.yml*).
- ✓ *lib*: Almacena las clases y librerías utilizadas exclusivamente por el módulo.
- ✓ *templates*: Contiene las plantillas correspondientes a las acciones del módulo. Cuando se crea un nuevo módulo, automáticamente se crea la plantilla llamada *indexSuccess.php* (34).

El archivo *security.yml* es creado por Symfony para manejar la seguridad del proyecto en todos los niveles tanto a nivel de aplicación como de módulo, este fichero permite restringir el acceso a determinadas acciones, se configura que una página o modulo solamente pueda ser accedida por los usuarios registrados o por un grupo de usuarios registrados con permisos especiales (credenciales). Si se tiene un archivo *security.yml* dentro de la carpeta *config* de la aplicación, esta configuración servirá para los módulos que no contengan un archivo *security.yml* dentro de sus subdirectorios config, sin embargo los módulos que contengan este archivo, su configuración de seguridad ya no será la de la aplicación.

En sistemas que tengan aplicaciones públicas, es decir que pueda ser accedida por cualquier usuario, no es necesario definir un archivo de configuración de seguridad, mientras que en

sistemas como SIPP donde la información es muy sensible se necesita estar autenticado con credenciales para acceder a cualquier de sus aplicaciones. El archivo *security.yml* se encuentra preferiblemente en las carpetas *config* de cada módulo con el objetivo de tener bien identificado los permisos para los usuarios y exista menos probabilidades de fraccionar el sistema.

Ejemplo de restricciones de acceso:
(**apps**/nombre_aplicacion/**modules**/nombre_modulo/**config**/**security.yml**)

```
1  ver:
2    is_secure: off        # Todos los usuarios pueden ejecutar la acción "ver"
3
4  modificar:
5    is_secure: on         # Acción "modificar" sólo para usuarios autenticados
6
7  borrar:
8    is_secure: on         # Sólo para usuarios autenticados
9    credentials: admin    # Con credencial "admin"
10
11 all:
12   is_secure: off        # off es el valor por defecto
```

Figura 14: Contenido de los archivos de seguridad.

El subsistema de seguridad trabaja con estos ficheros de forma dinámica, es decir el administrador de sistema nunca tendrá que abrir uno de estos ficheros manualmente para asignar a cada usuario en dependencia de su rol, al módulo al cual va a acceder.

3.5 Modelo de Implementación.

El modelo de implementación describe cómo los elementos del modelo de diseño, como las clases se implementan en término de componentes (ficheros de código fuente, ejecutables, etc.). El modelo de implementación describe también cómo se organizan los componentes de acuerdo a los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje de programación, y cómo dependen los componentes uno de otros. Además éste define una jerarquía por su sistema de implementación (35).

3.5.1 Vista de Implementación.

Los subsistemas de implementación proporcionan una forma de organizar los artefactos del modelo de implementación en trozos más manejables. Un subsistema puede estar formado por componentes, interfaces y otros subsistemas (recursivamente). Además, un subsistema puede

implementar las interfaces que representan la funcionalidad que exporta en forma de operaciones. Es importante entender que un subsistema de implementación se manifiesta a través de un mecanismo de empaquetamiento concreto en un entorno de implementación determinado (35).

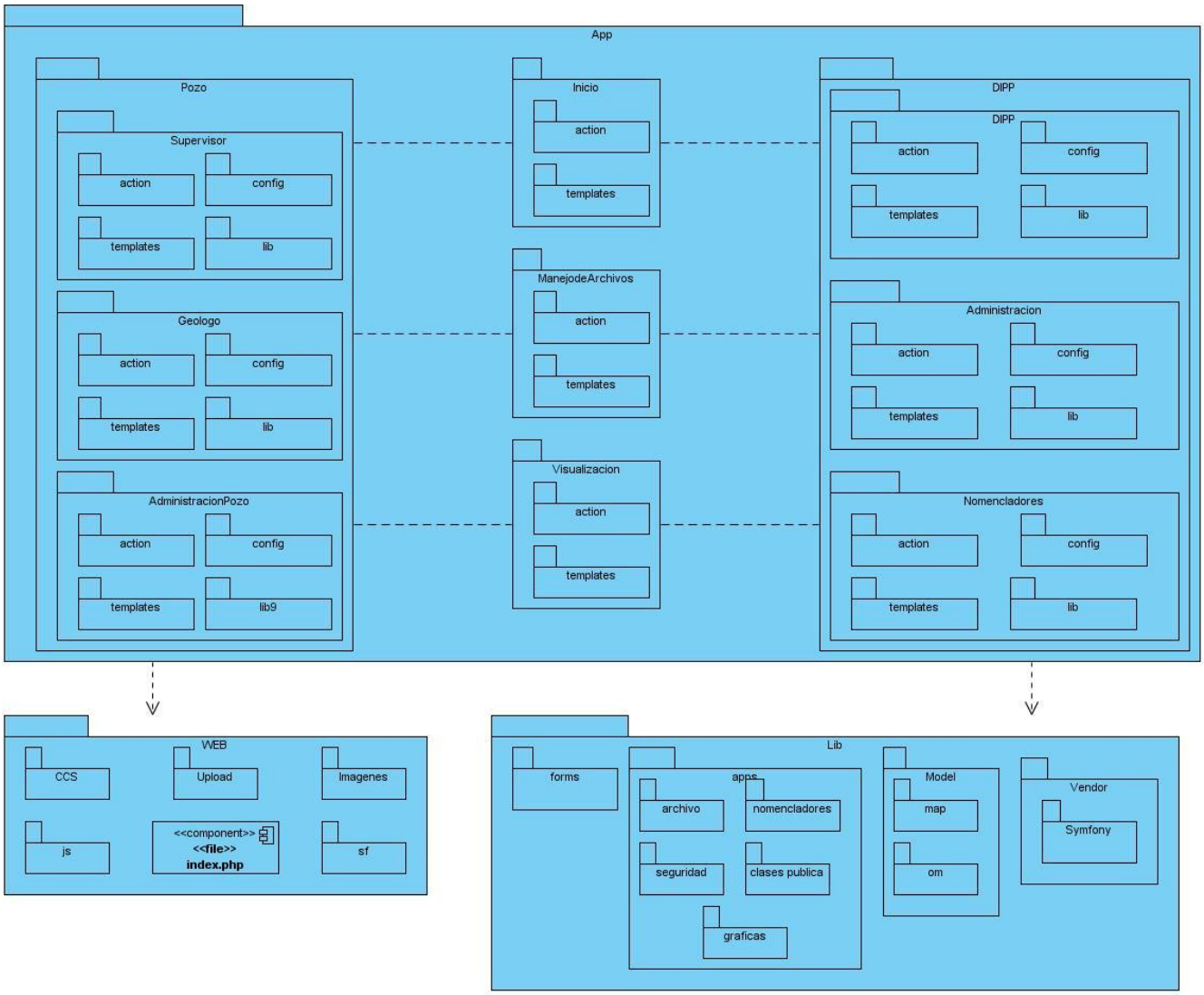


Figura 15: Vista de Implementación de SIPP.

Dentro del sistema lo módulos correspondientes al subsistema de seguridad por la parte de la DIPP está el de Administración, encargándose principalmente de la gestión de usuarios y roles así como de la relación entre ellos dando lugar a los permisos, Por la parte de la aplicación del Pozo encontramos al módulo Administración de Pozo donde solo se gestionan los usuarios de un pozo, este módulo existe debido a la deficiencia que existe en las redes de comunicación entre

los pozos de perforación y la DIPP. Inicio es un módulo común para cualquiera de las dos aplicación su objetivo principal es la autenticación de usuario.

Todos estos módulos interactúan con las diferentes librerías que existen ya sean las propias del framework, como las creadas por el equipo de desarrollo que encapsulan la lógica del negocio, también utilizan las clases del modelo de acceso a datos para la interacción con la Base de Datos. Todo esto puede ser visible desde la capa de Presentación contenida dentro de WEB la cual contiene todo lo referente a la vista del sistema, ficheros CSS, JS, imágenes, todo lo que el usuario final puede observar.

3.5.2 Diagramas de Componentes.

Diagrama dentro del cual se empaquetan físicamente los elementos de un modelo, modelan la vista estática de un sistema. No es necesario que un diagrama incluya todos los componentes del sistema, además se tiene en consideración los requisitos relacionados con la facilidad del desarrollo, así como también Muestran los componentes que constituyen una parte reusable.

3.5.2.1 Descripción general de los componentes.

1. **JavaScript**

Este tipo de fichero se ejecuta por el lado del cliente, se utiliza principalmente para la validación de los campos de un formulario, pero también para la creación de componentes dinámicos.

2. **CSS (Cascading Style Sheets)**

Este fichero es interpretado por el navegador para dar estilo a los componentes de las vistas, enriquecen el diseño del sistema.

3. **Actions Class**

Este fichero contiene la clase que almacena todas las acciones de un módulo, es decir la clase controladora de las vistas que se ejecutan en el módulo. Se encarga principalmente en capturar las peticiones hechas por el usuario mediante el navegador, procesarlas utilizando las librerías del sistema y devolviendo como respuesta el código que será mostrado al usuario.

4. **Model Class**

Son las clases generadas por Symfony que necesita la capa ORM (Object-Relational-Mapping) para agilizar el trabajo con la Base de Datos.

5. **Propel**

Symfony utiliza Propel para el mapeo de objetos a Base de Datos conocido como ORM.

3.5.2.2 Diagramas de componentes del Subsistema Seguridad.

1. **Autenticar Usuario.**

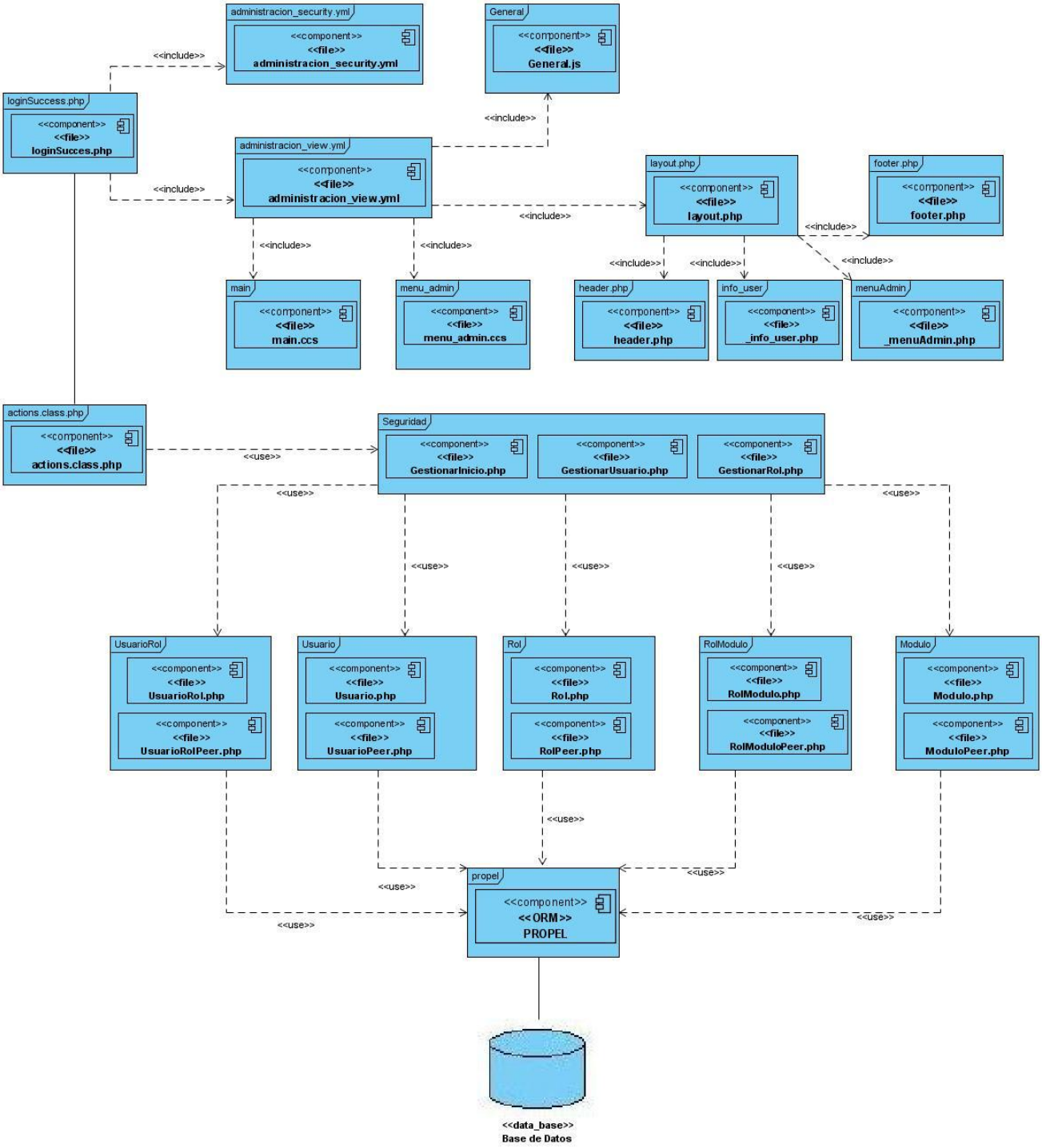


Figura 16: DC1 Autenticar Usuario.

2. Cambiar Contraseña.

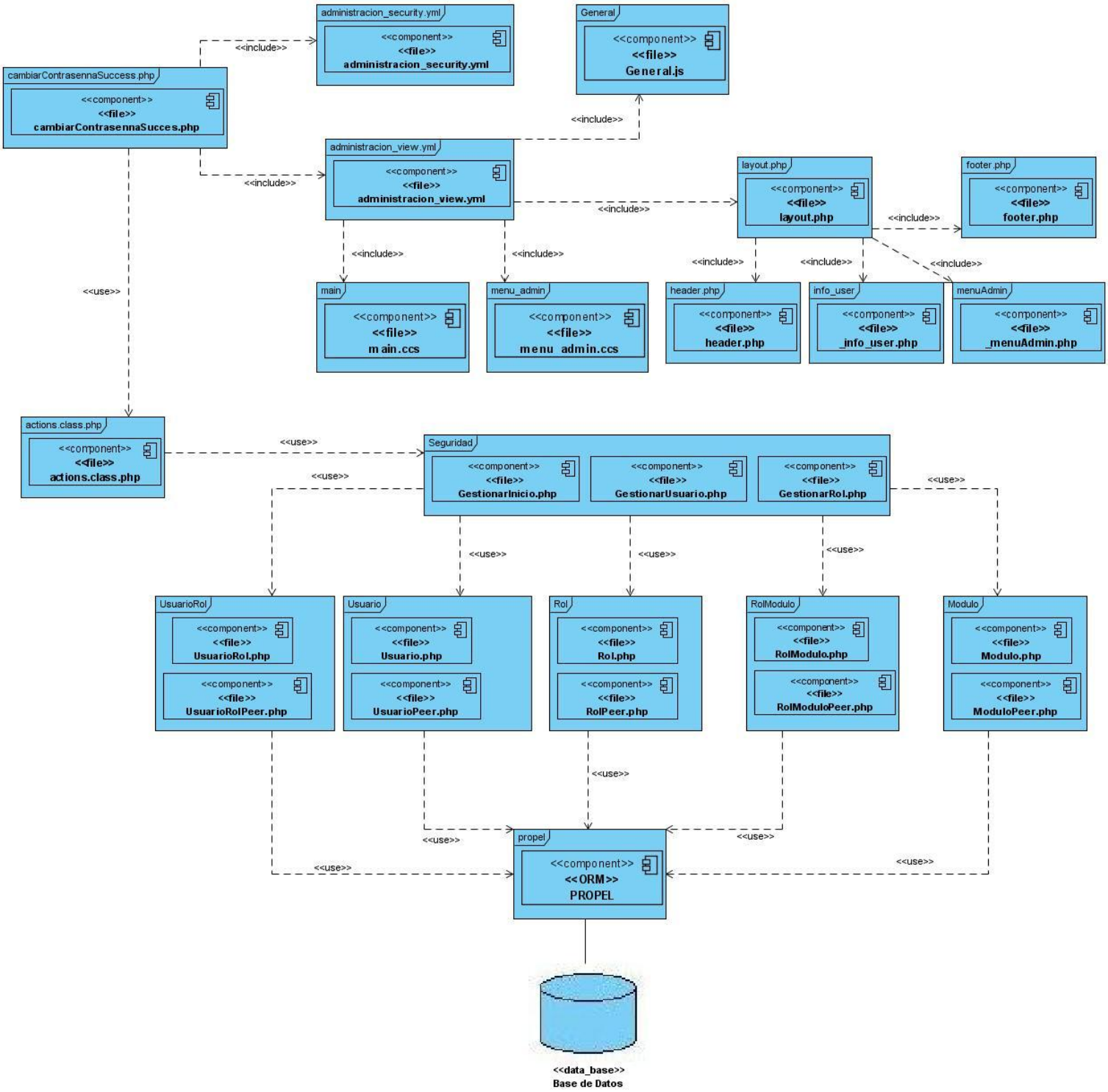


Figura 17: DC2 Cambiar Contraseña.

3. Gestionar Usuario.

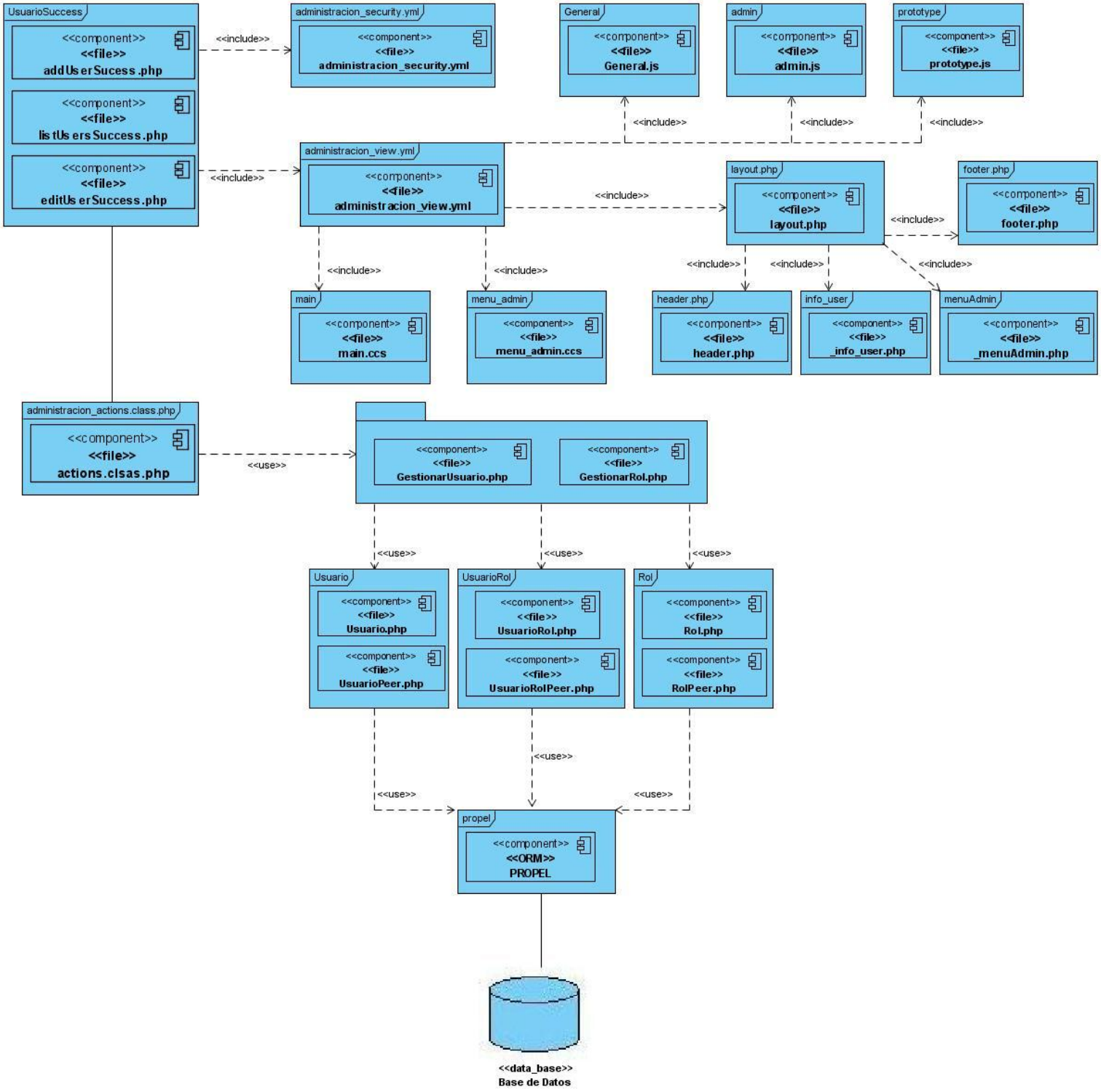


Figura 18: DC3 Gestionar Usuario.

4. Gestionar Rol.

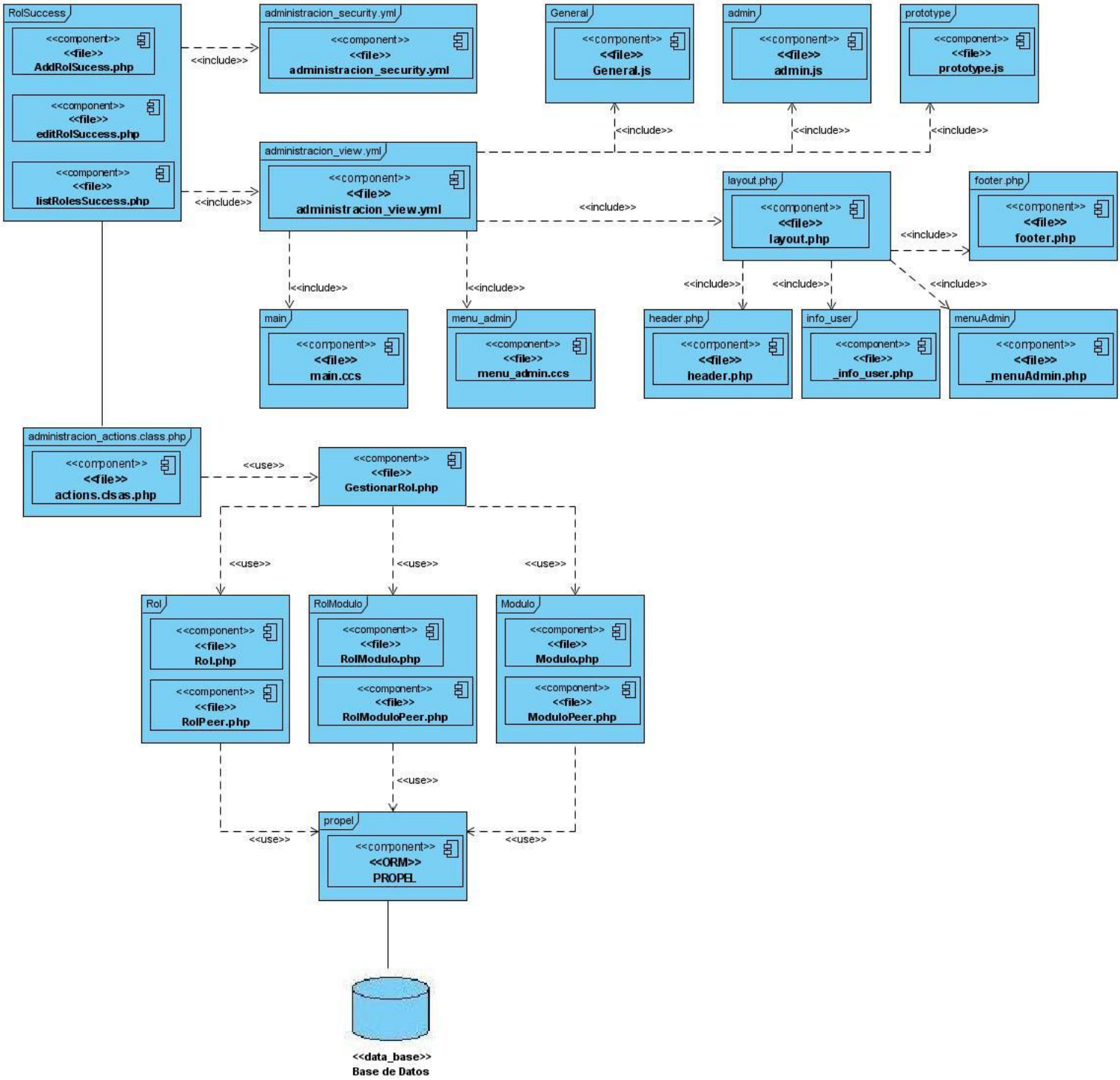


Figura 19: DC4 Gestionar Rol.

5. Asignar a Pozo.

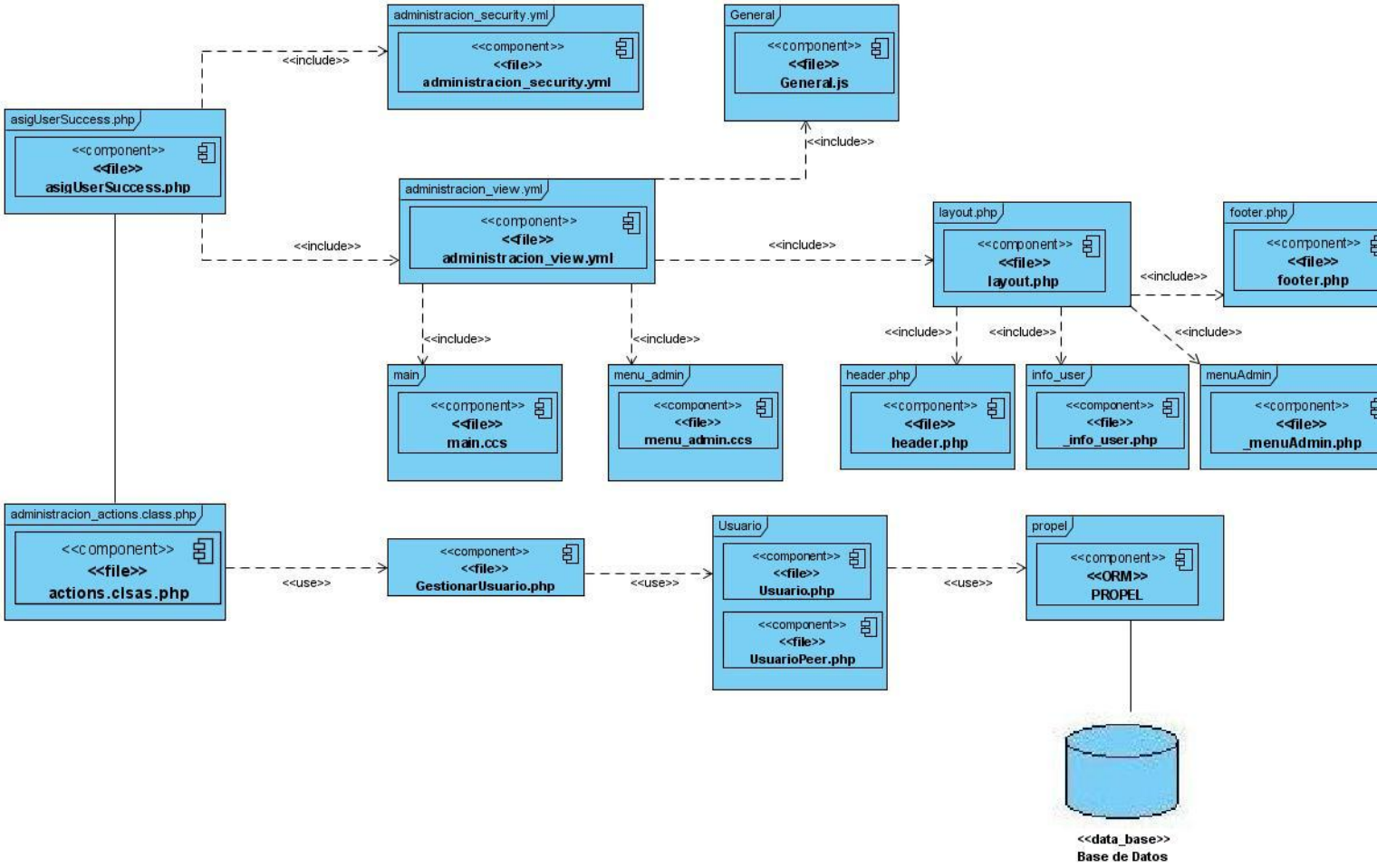


Figura 20: DC5 Asignar a Pozo.

3.6 Validación del Resultado.

Las pruebas de software son los procesos que permiten verificar y revelar la calidad de un producto de software. Son utilizadas para identificar posibles fallos de implementación, calidad, o usabilidad, básicamente es una fase de desarrollo de software (36). Las pruebas deben centrarse en dos objetivos fundamentales, si el software no hace lo que debe hacer y si hace lo que no debe hacer buscando efectos secundarios adversos, todo con el objetivo de medir el grado con que el software cumple con los requerimientos. Existen dos tipos de pruebas, técnicas y funcionales. Las pruebas técnicas son responsabilidad del equipo de desarrollo, mientras que las pruebas funcionales deben ser realizadas por parte del técnico de pruebas, ya que dispone de los conocimientos y aptitudes necesarias para este tipo de tarea (37).

Para poder tener una visión de la calidad del Subsistema de Seguridad se realizaron principalmente pruebas de caja negra y las pruebas de aceptación por parte del cliente.

3.6.1 Pruebas de Caja Negra (Black Box Testing).

Las pruebas de caja negra se dirigen directamente al exterior de un sistema, son pruebas funcionales donde se tratan de encontrar rupturas que no se encuentren dentro de su especificación, como pueden ser de interfaz de usuario, disposición de menús, control de teclas entre otros. Estas pruebas no son aplicables a módulos o componentes que sean transparentes al usuario.

Cuando se diseñan casos de prueba usando el método de Caja Negra, es con el objetivo de verificar que el sistema se comporta según los requerimientos establecidos por el cliente. Para ellos se necesita, como artefacto de entrada, el documento donde quedaron plasmados dichos requisitos, esto en algunos casos se trata de una lista con los requisitos del sistema. En el caso de que los requisitos sean modelados como casos de uso, se necesita la descripción de estos como artefacto de entrada (38).

3.6.2 Pruebas de Aceptación.

Son pruebas realizadas por el cliente, su objetivo es verificar que el software está listo y que puede ser usado por los usuarios finales, aunque determina la conformidad del cliente antes de que sea entregada una versión final o estable.

Cuando se construye un software a medidas para un cliente, es necesario llevar a cabo una serie de pruebas de aceptación para así poder la validación de todos los requisitos por parte del usuario final. El tiempo de duración de una prueba de aceptación puede variar desde semanas hasta meses. No es recomendable cuando un sistema lo utilizarán una gran cantidad de usuarios finales, hacer una prueba de aceptación para cada uno, para mitigar esta situación existe un proceso denominado prueba **alfa** y **beta** para descubrir errores que solo el usuario final podrá encontrar.

- ✓ **Prueba Alfa:** Es llevada a cabo por un cliente en conjunto con el equipo de desarrollo, este cliente usa el sistema de forma natural al lado de un desarrollador como supervisor el cual registra errores y problemas de uso, por lo tanto estas pruebas se llevan a cabo en un ambiente controlado.

- ✓ **Prueba Beta:** Esta se realiza por los usuarios finales del sistema en los lugares de trabajo del cliente, en esta prueba está ausente el equipo de desarrollo, por lo tanto la prueba no es más que el sistema en vivo en un entorno de trabajo que no puede ser controlado por los desarrolladores. Aquí el cliente es el que registra los problemas tanto reales como imaginarios y los informa al equipo de desarrollo a intervalos regulares, esto proporciona la modificación del sistema a intervalos también, para así preparar una versión más estable para cualquier tipo de cliente (39).

"Semejante a la muerte y a los impuestos, la prueba es desagradable e inevitable."

Edward Yourdon.

Durante el proceso de desarrollo se realizaron 3 pruebas de aceptación al sistema. La primera en enero de 2010, la segunda en mayo de 2010, la última realizada desde diciembre de 2010 hasta marzo 2011 durante el despliegue en un pozo en perforación. Como parte de este proceso se firmó un acta de aceptación donde se aceptaron varios subsistemas además del implementado. (Ver anexos 2 y 3)

3.6.3 Descripción de los Casos de Pruebas.

Desde el momento que el equipo de desarrollo elabora los casos de uso del sistema, se pueden elaborar los casos de prueba que no son más que un conjunto de entrada con datos de prueba, unas condiciones de ejecución y unos resultados esperados con el objetivo de identificar y notificar las condiciones que se llevarán a cabo en las pruebas. Los casos de prueba son necesarios para validar el funcionamiento de la aplicación y aceptar los requisitos del producto.

Tabla 2: DCP Autenticar Usuario.

Escenario	Variable 1 Usuario	Variable 2 Contraseña	Respuesta del Sistema	Resultado de la Prueba	Flujo Central
EC1.1: Autenticar Usuario correctamente.	V (ycarias)	V (pruebasipp)	El sistema comprueba que el usuario y contraseña son válidos y permite el acceso satisfactoriamente según el rol asignado al usuario.	*Satisfactorio*	Página de inicio del sistema SIPP

EC1.2: Autenticar Usuario erróneamente	I (pepe)	I (pepe)	El sistema muestra un mensaje: "El usuario o la contraseña son incorrectos", y limpia el campo del usuario	*Satisfactorio*	Página de inicio del sistema SIPP
	I (pepe) (@#$%)	V (pruebasipp) (pruebasipp)	El sistema muestra un mensaje: "El usuario o la contraseña son incorrectos", y limpia el campo del usuario	*No satisfactorio (para caracteres extraños)*	
	V (ycarias)	I (pruebasipp1)	El sistema muestra un mensaje: "El usuario o la contraseña son incorrectos", y limpia el campo del usuario	*Satisfactorio*	
EC 1.3: Autenticar dejando campos vacíos.	I (vacío)	I (vacío)	El sistema muestra un mensaje: Debe llenar todos los campos	*Satisfactorio.*	Página de inicio del sistema SIPP
	I (vacío)	V (pruebasipp)	El sistema muestra un mensaje: Debe llenar todos los campos	*Satisfactorio.*	
	V (ycarias)	I (vacío)	El sistema muestra un mensaje: Debe llenar todos los campos	*No Satisfactorio*	
EC 1.4: Cambiar contraseña	N/A	N/A	El sistema muestra la interfaz para Cambiar la contraseña	*Satisfactorio*	Página de inicio del sistema SIPP
EC 1.5: Autenticar usuario que no tiene pozo asignado	V (UsusarioSinPozo)	V (Ususario SinPozo)	El sistema muestra un mensaje indicando que "El usuario no está asignado a ningún pozo" y limpia los campos.	*Satisfactorio.*	Página de inicio del sistema SIPP
EC 1.6: Autenticar usuario asignado a un pozo que aún no esté en perforación.	V (UsusarioPozoNF)	V (Ususario PozoNF)	El sistema muestra un mensaje indicando "El pozo al que usted está asignado no está en Perforación" y limpia los campos.	*Satisfactorio.*	Página de inicio del sistema SIPP

Tabla 3: DCP Cambiar Contraseña.

Escenario	Variable 1 Usuario	Variable 2 Contraseña antigua	Variable 3 Nueva contraseña	Variable 4 Repetir contraseña nueva	Respuesta del Sistema	Resultado de la Prueba	Flujo Central
EC 1.1: Cambiar contraseña satisfactoriamente.	V (ycarias)	V (pruebasipp)	V (pruebasipp0)	V (pruebasipp0)	El sistema comprueba datos y realiza el cambio de contraseña satisfactoriamente notificando al usuario que "Se ha cambiado su contraseña".	*Satisfactorio*	Varieante1: Página de Inicio de SIPP Variante2: Luego de autenticarse en el sistema seleccionar el ícono "Cambio de contraseña" representado por unas llaves sobre el menú. Observación: Probar las dos variantes.
EC 1.2: Cambiar contraseña introduciendo datos inválidos para el usuario.	I (yisell)	V (pruebasipp0)	V (pruebasipp)	V (pruebasipp)	El sistema comprueba los datos insertados y muestra un mensaje indicando que el usuario no existe en el sistema.	Satisfactorio	Varieante1: Página de Inicio de SIPP Variante2: Luego de autenticarse en el sistema seleccionar el ícono "Cambio de contraseña" representado por unas llaves sobre el menú. Observación: Probar las dos variantes.

	I (K((*$#&@)	V (pruebasipp0)	V (pruebasipp)	V (pruebasipp)	El sistema comprueba los datos insertados y muestra un mensaje indicando que no se pueden introducir caracteres extraños para el usuario	No satisfactorio	Varieante1: Página de Inicio de SIPP Variante2: Luego de autenticarse en el sistema seleccionar el ícono “Cambio de contraseña” representado por unas llaves sobre el menú. Observación : Probar las dos variantes
EC 1.3: Cambiar contraseña introduciendo datos inválidos para la contraseña actual	V (ycarias)	I (123456)	V (Nueva)	V (Nueva)	El sistema verifica los datos y envía un mensaje indicando que la “Contraseña actual es incorrecta para el usuario”		Varieante1: Página de Inicio de SIPP Variante2: Luego de autenticarse en el sistema seleccionar el ícono “Cambio de contraseña” representado por unas llaves sobre el menú. *Observación : Probar las dos variantes*
EC 1.4: Cambiar contraseña dejando campos vacíos.	I (Vacío)	V (pruebasipp)	V (pruebasipp0)	V (pruebasipp0)	El sistema verifica que los campos estén llenos , si no resulta , muestra un mensaje de error ”Se deben llenar todos los campos”	*Satisfactorio.*	Varieante1: Página de Inicio de SIPP Variante2: Luego de autenticarse en el sistema seleccionar el ícono “Cambio de contraseña” representado por unas llaves sobre el

	V (ycarias)	I (vacío)	V (pruebasipp0)	V (pruebasipp0)	El sistema verifica que los campos estén llenos , si no resulta , muestra un mensaje de error ”Se deben llenar todos los campos”	*Satisfactorio.*	menú. *Observación : Probar las dos variantes*
	V (ycarias)	V (pruebasipp1)	I (Vacío)	V (pruebasipp0)	El sistema verifica que los campos estén llenos , si no resulta , muestra un mensaje de error ”Se deben llenar todos los campos”	*Satisfactorio.*	
	V (ycarias)	V (pruebasipp1)	V (pruebasipp0)	I (Vacío)	El sistema verifica que los campos estén llenos , si no resulta , muestra un mensaje de error ”Se deben llenar todos los campos”	*Satisfactorio.*	
EC 1.5: Cambiar contraseña introduciendo datos diferentes en la nueva contraseña y la confirmación de la misma.	V (ycarias)	V (pruebasipp1)	V (pruebasipp0)	I (123456)	El sistema valida la información del usuario y muestra un mensaje señalando que las contraseña nueva y confirmación de contraseña no coinciden.	*Satisfactorio*	Varieante1: Página de Inicio de SIPP Variante2: Luego de autenticarse en el sistema seleccionar el ícono “Cambio de contraseña” representado por unas llaves sobre el menú. *Observación : Probar las dos variantes*

EC 1.6 No Aceptar confirmación de la operación	V (N/A)	V (N/A)	V (N/A)	V (N/A)	El sistema cancela la operación de modificar contraseña y mantiene la interfaz, luego regresa a la página en la que estaba antes de que el usuario solicitara "Cambiar contraseña"	*Satisfactorio*	Varieante1: Página de Inicio de SIPP Variante2: Luego de autenticarse en el sistema seleccionar el ícono "Cambio de contraseña" representado por unas llaves sobre el menú. Observación : Probar las dos variantes

Los faltantes diseños de caso de prueba se pueden encontrar en el Anexo 1.

3.7 Conclusiones Parciales.

Una vez concluido este capítulo, donde se expuso las características del resultado y proceso de solución se han llegado a las siguientes conclusiones:

1. La profundización en el análisis de los requisitos permitió establecer el alcance de la solución.
2. Debido a la estructuración y organización del modelo de implementación se logró una independencia funcional del subsistema.
3. El proceso de prueba realizado permitió validar la calidad del producto.

Conclusiones Generales.

El proceso de investigación realizado en el ambiente de despliegue posibilitó la correcta identificación de las reglas de negocio asociadas a la seguridad del sistema, lo cual se refleja en el resultado obtenido, en su marcada orientación a los roles que intervienen en el proceso de perforación en tierra.

El uso de herramientas y tecnologías, devenidas del software libre, permitió la construcción de una solución con alto grado de portabilidad para su futura distribución y uso.

La disposición modulada del modelo de implementación, le aporto al resultado, la versatilidad e independencia funcional necesaria para poder integrarse con subsistemas o aplicación de similares características.

La realización de pruebas de caja negra permitió validar las funcionalidades implementadas en el subsistema, aumentando la calidad del producto. En el caso de las pruebas de aceptación, posibilitaron validar con los usuarios finales la usabilidad y cumplimiento de los requisitos pactados.

Recomendaciones.

Teniendo como base los resultados obtenidos, se proponen las siguientes recomendaciones; las cuales constituyen una colección de las principales metas a trazar, con el objetivo de mejorar el subsistema en sus próximas versiones:

1. Implementar los casos de uso que fueron pospuestos para la próxima versión.
2. Profundizar en el estudio de las actividades que realizan los usuarios durante el proceso de perforación y fuera del mismo, a fin de encontrar nuevas funcionalidades que aumenten la seguridad del sistema Ej.: Registros de Dominio, manejo de usuarios por cuentas de correo.
3. Investigar sobre nuevas funciones o complementos del framework de desarrollo con respecto a seguridad para posibles inclusiones en el subsistema.

Bibliografía.

1. **Cuevas, David Tavares.** [En línea] Septiembre de 2009. [Citado el: 12 de enero de 2011.] http://bibliodoc.uci.cu/TD/TD_2728_09.pdf..

2. **Definicion.de.** [En línea] [Citado el: 02 de junio de 2011.] http://definicion.de/seguridad-informatica/.

3. **SIPP.** *Glosario de Términos. SIPP.* 2010.

4. monografias.com. [En línea] [Citado el: 13 de enero de 2011.] http://www.monografias.com/trabajos75/seguridad-desarrollo-aplicaciones-web/seguridad-desarrollo-aplicaciones-web.shtml.

5. **Gestión, Sistemas de.** Sistemas de Gestión. *Sistemas de Gestión.* [En línea] [Citado el: 14 de enero de 2011.] http://www.bsigroup.com.mx/es-mx/Auditoria-y-Certificacion/Sistemas-de-Gestion/De-un-vistazo/Que-son-los-sistemas-de-gestion/.

6. **SIPP.** *Estudio del estado del arte de sistemas para el área de perforacón.* 2009.

7. Peloton. *Drilling & Well Data Software Solutions.* [En línea] [Citado el: 15 de enero de 2011.] http://www.peloton.com/es/default.asp?id=13.

8. **WellSight.** WellSight.com. [En línea] [Citado el: 16 de enero de 2011.] http://www.wellsight.com.

9. Mi Tecnológico. *Protocolo HTTP.* [En línea] [Citado el: 27 de marzo de 2011.] http://www.mitecnologico.com/Main/ProtocoloHttp.

10. **Connalen, J.** *Building Web Applications with UML. s.l. : Adison Wesley.* 1999.

11. **Sánchez, Alain y Rodríguez, Darlenis.** *Sistema de gestión para el Programa Nacional de Grupos Electrogenos DENYO.* Ciudad de la Habana : s.n., 2007.

12. Symfony. *HowToContributeToSymfony.* [En línea] [Citado el: 25 de marzo de 2011.] http://trac.symfony-project.org/wiki/HowToContributeToSymfony#CodingStandards.

13. **Agenda, SOA.** SOA Agenda. [En línea] [Citado el: 28 de marzo de 2011.] http://soaagenda.com/journal/articulos/que-son-los-frameworks/.

14. pisuerga. *Introducción a la Programación Orientada a Objeto.* [En línea] [Citado el: 28 de marzo de 2011.] http://pisuerga.inf.ubu.es/lsi/Invest/Java/Tuto/I_1.htm.

15. **Zabala.** angelfire. *La ingeniería de Software.* [En línea] 2000. [Citado el: 29 de marzo de 2011.] http://www.angelfire.com/scifi/jzavalar/apuntes/IngSoftware.html#paradigmaOO.

16. Desarrollo WEB. *Programación orientada a objeto.* [En línea] [Citado el: 28 de marzo de 2011.] http://www.desarrolloweb.com/articulos/499.php.

17. Ciberaula. *Introducción, definición y evolución de PHP.* [En línea] 2006. [Citado el: 25 de marzo de 2011.] http://php.ciberaula.com/articulo/introduccion_php/.

18. **Hinostroza, Raul Rodas.** LinuxCentro.net. *LinuxCentro.net - Características de PHP.* [En línea] 22 de febrero de 2007. [Citado el: 25 de marzo de 2011.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.

19. Software libre para todos. *Liberado PHP 5.0.0 - Software Libre.* [En línea] 14 de julio de 2009. [Citado el: 25 de marzo de 2011.] http://www.somoslibres.org/modules.php?name=News&file=article&sid=178.

20. **Mozilla.** [En línea] [Citado el: 25 de marzo de 2011.] https://developer.mozilla.org/es/JavaScript.

21. El código. *Javascript.* [En línea] [Citado el: 28 de marzo de 2011.] http://www.elcodigo.net/tutoriales/javascript/javascript1.html.

22. Hooping. *HTML.* [En línea] [Citado el: 28 de marzo de 2011.] http://www.hooping.net/faq-caracteristicas.aspx.

23. SiDar. *CSS.* [En línea] [Citado el: 28 de marzo de 2011.] http://www.sidar.org/recur/desdi/mcss/tareas/20011206/slide2-0.html.

24. **Alvarez, Miguel Angel.** desarrolloweb.com. *Zend Studio.* [En línea] 4 de junio de 2003. [Citado el: 25 de marzo de 2011.] http://www.desarrolloweb.com/articulos/1178.php.

25. **Web, Libros.** Symfony, la guía definitiva. [En línea] [Citado el: 25 de marzo de 2011.] http://www.librosweb.es/symfony/.

26. Garbage Collector. *Sistema gestor de base de datos SGBD.* [En línea] 1 de noviembre de 2004. [Citado el: 25 de marzo de 2011.] http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base_de_datos_sgbd.php.

27. PostgreSQL. [En línea] [Citado el: 26 de marzo de 2011.] http://www.postgresql.org/.

28. **A., Ernesto Quiñones.** APESOL. *introduccion_a_postgresql.* [En línea] [Citado el: 10 de febrero de 2010.] http://www.eqsoft.net/presentas/introduccion_a_postgresql.pdf.

29. Ciberaula. *Una Introduccion al Apache.* [En línea] 2006. [Citado el: 26 de marzo de 2011.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

30. I-Sol. *Aseguramiento de la Calidad.* [En línea] [Citado el: 26 de marzo de 2011.] http://www.i-sol.com.ar/pg005.html.

31. **Fontela, Carlos.** Técnicas de aseguramiento de la calidad del producto (Carlos Fontela). *CyS Ingeniería de Software.* [En línea] 19 de Junio de 2009. [Citado el: 27 de marzo de 2011.] http://cysingsoft.wordpress.com/2008/06/24/tecnicas-de-aseguramiento-de-la-calidad-del-producto/.

32. Procesos de la ingeniería de requerimientos. [En línea] [Citado el: 6 de abril de 2011.] http://www.google.com/url?sa=t&source=web&cd=4&ved=0CC0QFjAD&url=http%3A%2F%2Fantares.itmorelia.

edu.mx%2F~jcolivar%2Fcourses%2Fpm10a%2Fpm_u1.doc&rct=j&q=definicion%20de%20requerimiento%20por%20la%20IEEE&ei=ajiOTdaDH-GH0QG1xZSrCw&usg=AFQjCNFW4xWEv7DfOWvMZV.

33. **SIPP.** *Especificación de Requisitos Subsistema de Seguridad.* 2009.

34. **Potencier, Fabien y Zaninotto, François.** *Symfony 1.2, La Guía Definitiva.* 2008.

35. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo.* La Habana : Félix Varela, 2004.

36. **v3, Metrica.** Administración Electrónica. [En línea] [Citado el: 20 de mayo de 2011.] http://administracionelectronica.gob.es/?_nfpb=true&_pageLabel=P60085901274201580632&langPae=es.

37. **PRUEBASDESOFTWARE.** PRUEBASDESOFTWARE. [En línea] [Citado el: 20 de mayo de 2011.] http://www.pruebasdesoftware.com/laspruebasdesoftware.htm.

38. **UCI.** Entorno Virtual de Aprendizaje. [En línea] [Citado el: 20 de mayo de 2011.] http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_10/Clase_Practica_6/Materiales_Basicos/Material_El_Como_disenar_casos_de_pruebas_a_partir_de_los_casos_de_uso.pdf.

39. **Pressman, Roger S.** *Ingeniería del Software Un enfoque práctico.*

40. **Microsiervos.** La polilla que voló dentro de un ordenador y el origen de (bug informático). [En línea] [Citado el: 2 de junio de 2011.] http://www.microsiervos.com/archivo/leyendas-urbanas/polilla-volo-en-ordenador-y-origen-leyenda-bug.html.

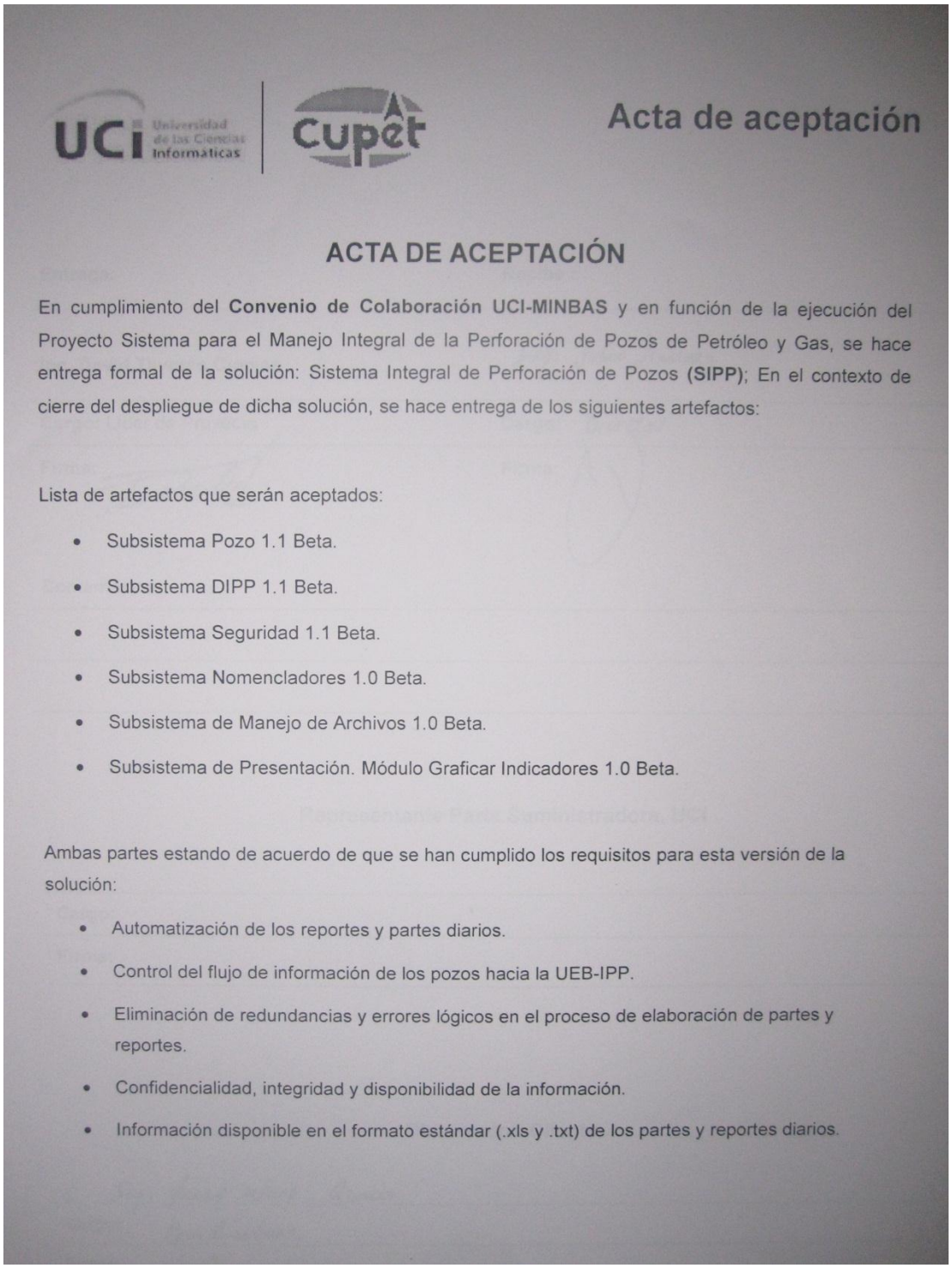
41. **Manager, Free Download.** Well Logger. [En línea] [Citado el: 03 de junio de 2011.] http://www.freedownloadmanager.org/es/downloads/Bien_Maderero_7024_p/.

42. **Texto, Hiper.** HIPERTEXTO: EL NUEVO CONCEPTO DE DOCUMENTO EN LA CULTURA DE LA IMAGEN. [En línea] [Citado el: 03 de junio de 2011.] http://www.hipertexto.info/.

Anexos.

Anexo 1: Corresponde a los documentos de diseño de caso de prueba de seguridad.

Anexo 2: Acta de Aceptación de los artefactos que fueron aceptados.



UCI Universidad de las Ciencias Informáticas | CUPET

Acta de aceptación

ACTA DE ACEPTACIÓN

En cumplimiento del **Convenio de Colaboración UCI-MINBAS** y en función de la ejecución del Proyecto Sistema para el Manejo Integral de la Perforación de Pozos de Petróleo y Gas, se hace entrega formal de la solución: Sistema Integral de Perforación de Pozos **(SIPP)**; En el contexto de cierre del despliegue de dicha solución, se hace entrega de los siguientes artefactos:

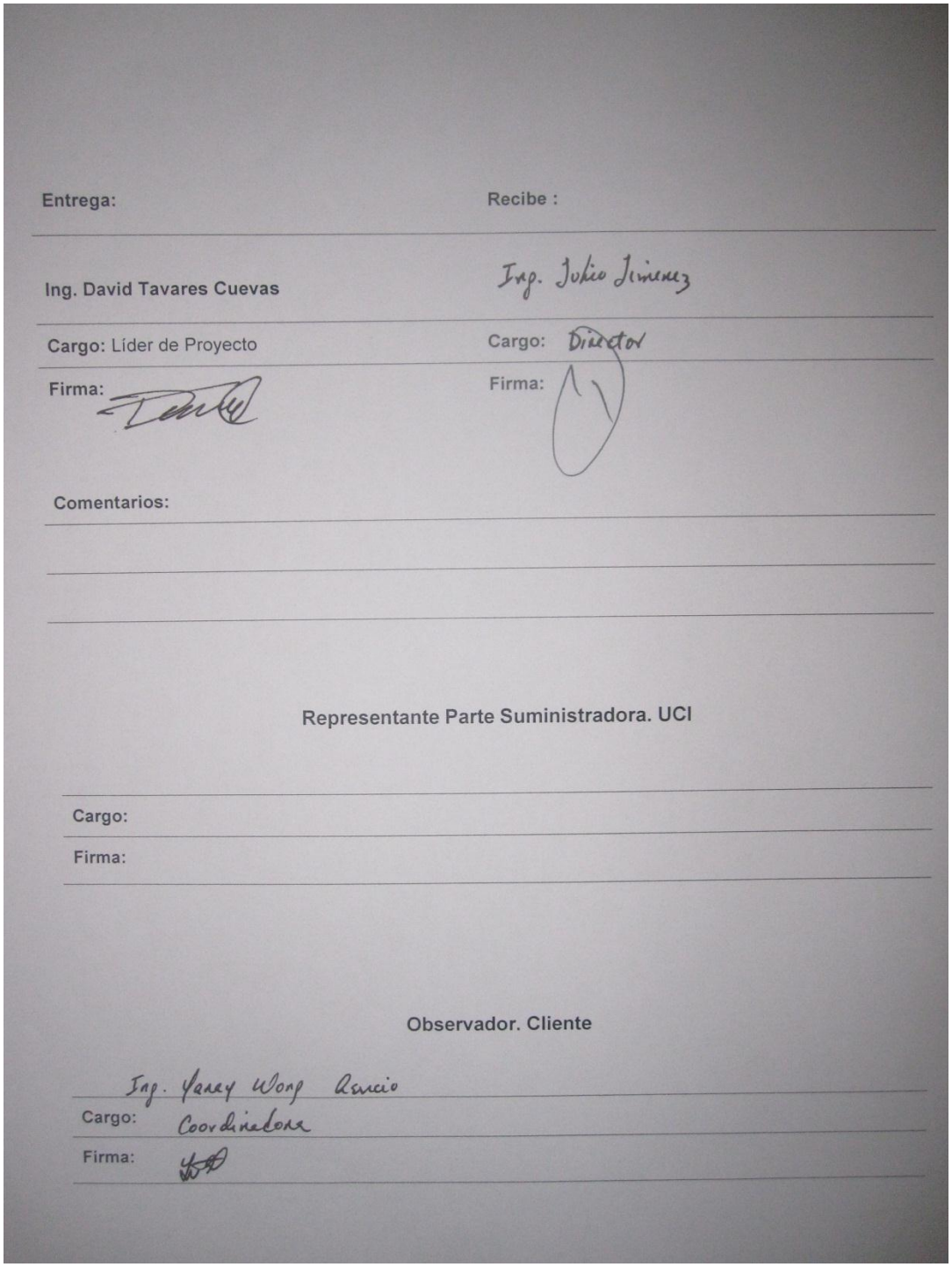
Lista de artefactos que serán aceptados:

- Subsistema Pozo 1.1 Beta.
- Subsistema DIPP 1.1 Beta.
- Subsistema Seguridad 1.1 Beta.
- Subsistema Nomencladores 1.0 Beta.
- Subsistema de Manejo de Archivos 1.0 Beta.
- Subsistema de Presentación. Módulo Graficar Indicadores 1.0 Beta.

Ambas partes estando de acuerdo de que se han cumplido los requisitos para esta versión de la solución:

- Automatización de los reportes y partes diarios.
- Control del flujo de información de los pozos hacia la UEB-IPP.
- Eliminación de redundancias y errores lógicos en el proceso de elaboración de partes y reportes.
- Confidencialidad, integridad y disponibilidad de la información.
- Información disponible en el formato estándar (.xls y .txt) de los partes y reportes diarios.

Anexo 3: Firma del Acta de Aceptación.



Entrega:

Ing. David Tavares Cuevas

Cargo: Líder de Proyecto

Firma:

Recibe :

Ing. Julio Jiménez

Cargo: Director

Firma:

Comentarios:

Representante Parte Suministradora. UCI

Cargo:

Firma:

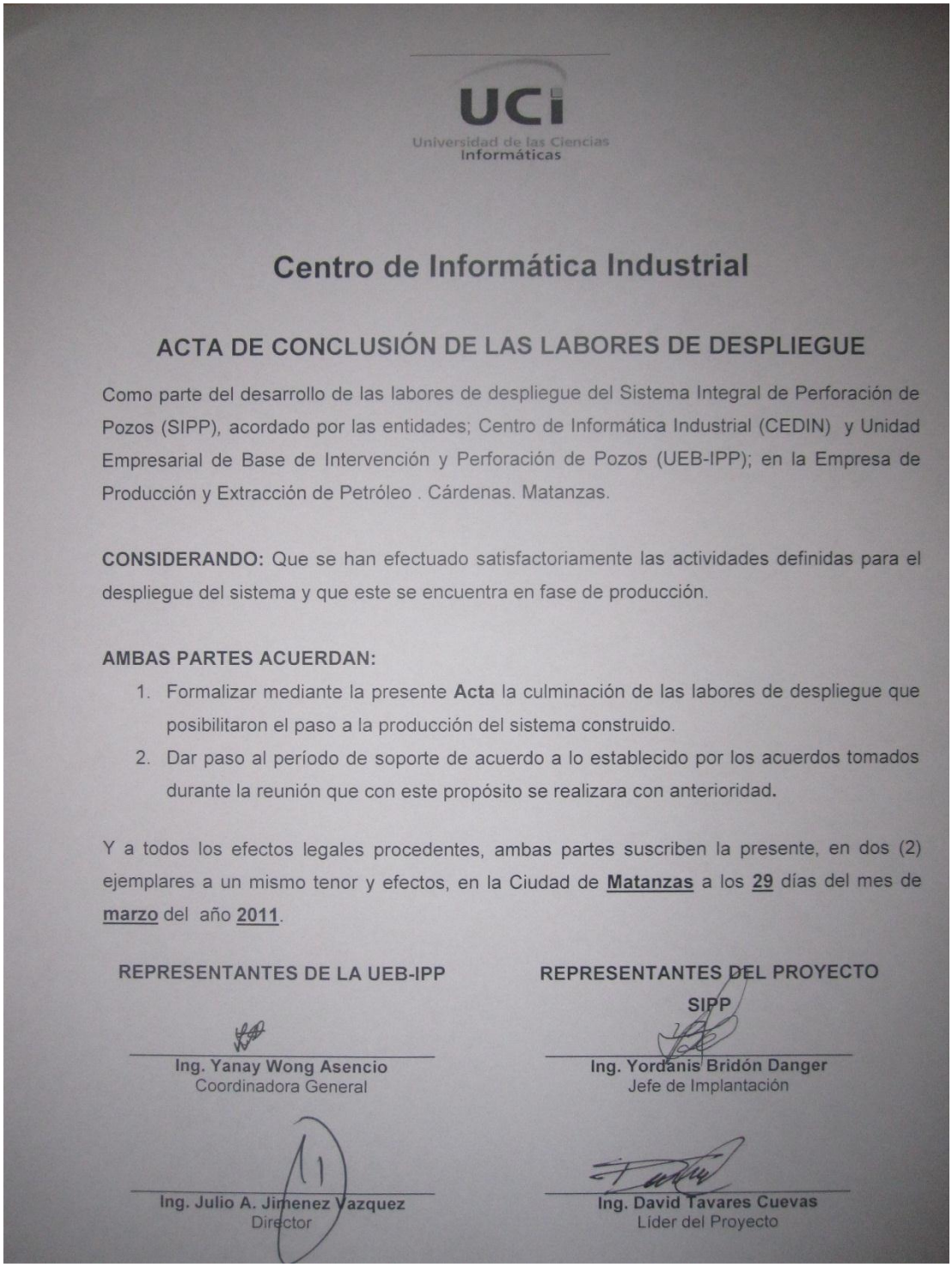
Observador. Cliente

Ing. Yanay Wong Acacio

Cargo: Coordinadora

Firma:

Anexo 4: Acta de conclusión de las labores de despliegue.



UCI
Universidad de las Ciencias Informáticas

Centro de Informática Industrial

ACTA DE CONCLUSIÓN DE LAS LABORES DE DESPLIEGUE

Como parte del desarrollo de las labores de despliegue del Sistema Integral de Perforación de Pozos (SIPP), acordado por las entidades; Centro de Informática Industrial (CEDIN) y Unidad Empresarial de Base de Intervención y Perforación de Pozos (UEB-IPP); en la Empresa de Producción y Extracción de Petróleo . Cárdenas. Matanzas.

CONSIDERANDO: Que se han efectuado satisfactoriamente las actividades definidas para el despliegue del sistema y que este se encuentra en fase de producción.

AMBAS PARTES ACUERDAN:

1. Formalizar mediante la presente **Acta** la culminación de las labores de despliegue que posibilitaron el paso a la producción del sistema construido.
2. Dar paso al período de soporte de acuerdo a lo establecido por los acuerdos tomados durante la reunión que con este propósito se realizara con anterioridad.

Y a todos los efectos legales procedentes, ambas partes suscriben la presente, en dos (2) ejemplares a un mismo tenor y efectos, en la Ciudad de **Matanzas** a los **29** días del mes de **marzo** del año **2011**.

REPRESENTANTES DE LA UEB-IPP

Ing. Yanay Wong Asencio
Coordinadora General

Ing. Julio A. Jimenez Vazquez
Director

REPRESENTANTES DEL PROYECTO SIPP

Ing. Yordanis Bridón Danger
Jefe de Implantación

Ing. David Tavares Cuevas
Líder del Proyecto