Facultad X

Título: Sistema de autenticación y autorización centralizado.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Lázaro Antonio Marín Mártires

Luis Ramón Capriles Alvarado

Tutor: Ing. William Azcuy Morales

Ciudad de la Habana

Julio 2008

"El hombre puede hacer de si mismo muchas cosas producto de su propio esfuerzo físico y espiritual, el que se proponga cultivar la virtud la cultiva, el que se proponga alcanzar los más altos niveles de conocimiento los alcanza."

Fidel Castro Ruz

DECLARACIÓN DE AUTORÍA

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

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

Lázaro Antonio Marín Mártires

_____
Firma del Autor

Luis Ramón Capriles Alvarado

_____
Firma del Autor

Ing. William Azcuy Morales

_____
Firma del Tutor

DATOS DE CONTACTO

Tutor:

Ing. William Azcuy Morales

Universidad de las Ciencias Informáticas, Ciudad de la Habana, Cuba.

william@uci.cu

AGRADECIMIENTOS

A nuestra Revolución y en especial a nuestro Comandante en Jefe, por ser el creador de esta Universidad del Futuro…

A nuestro tutor William Azcuy Morales por la importante ayuda que nos concedió en todo momento y por la dedicación que nos brindó, por el empeño y entusiasmo de realizar este trabajo junto a nosotros.

A toda nuestra familia por el apoyo incondicional que nos brindaron en todo momento para salir hacia adelante, por la confianza depositada en nosotros, y en especial a nuestros padres por ese amor inmenso que nos han dado siempre y por levantar nuestros ánimos en los momentos difíciles.

A todos nuestros amigos y profesores por el apoyo que nos dieron, por habernos ayudado de una forma u otra a obtener un resultado satisfactorio en nuestra carrera y por marcar en nosotros momentos inolvidables y en especial a Jose Ramón Fernández Pérez y Francisco Sanfiel por su ayuda incondicional.

A mi novia Annie Hernández Sánchez por estar siempre al lado mío y brindarme todo su apoyo y cariño.

DEDICATORIA

Dedico este trabajo a:

A la memoria de mi abuelo Romano Mártires, a mi mamá Concepción del Pilar, a mi papá Carlos y a mi abuela Enma (Mami) por ser ellos mi mano derecha en todo momento, por haberme sabido educar de una forma correcta, por darme ese afecto tan grande de familia, por guiarme, aconsejarme y querer siempre lo mejor para mi y sobre todas las cosas por aceptarme así como soy y por saber que siempre van a estar cerca de mi en cualquier parte de este mundo, a toda mi familia y amistades, a Heriberto Ángel por haber sabido mantener esta amistad desde lejos y por estar al lado mío, por brindarme su ayuda y por darme el aliento de un verdadero amigo en todo momento, a Cesarín el mas pequeño de la familia que de una forma u otra con las pocas veces que hable con el me proporcionó la alegría que me hacia falta para seguir adelante y en especial a ese Dios que existe por haberme permitido terminar mi tesis y salir adelante.

Lázaro Antonio Marín Mártires

A mis padres Merice ALvarado y Luis Ramón Capriles, que son mi fuente de inspiración, por su confianza y amor, por el esfuerzo realizado durante toda una vida. Por quererme tanto, le hago realidad hoy sus sueños.

A mi hermano Dayán, que lo quiero mucho y es lo más grande que tengo en esta vida.

A mi niña linda Yaniris por confiar siempre en mí, y brindarme todo su apoyo y cariño.

A mi abuelo Mundo, donde quiera que estés.

A mis abuelitos Zoraida, Febi y Almide, por brindarme todo su amor y ser mis segundos padres.

A mis amigos de siempre Liván, Peña, Yunior y Miguel por comportarse como verdaderos hermanos durante todo este tiempo.

A todos mis amigos, compañeros y profesores de la facultad 10. . . A todos en general muchas gracias de corazón

Luis Ramón Capriles Alvarado

RESUMEN

Como consecuencia a la ausencia de un sistema de autenticación y autorización centralizada en la corporación de PDVSA, actualmente el sistema informático de su arquitectura de computo no es capaz de mapear la credenciales de los usuarios hacia todas las aplicaciones de dicho sistema y otros sistemas heterogéneos, tampoco proporciona una infraestructura para simular un login único para el usuario corporativo frente a una plataforma tecnológica heterogénea dentro del ambiente de aplicativos de integración, en que los usuarios puedan acceder a diferentes aplicaciones con solo un conjunto de credenciales.

Esta falta de mapeo de credenciales de los usuarios hacia todas las aplicaciones del sistema condujo a la realización de esta tesis que se basa en hacer un software de autenticación y control centralizado, con el cual se pretende que el sistema informático de la corporación logre llevar a todas las aplicaciones del sistema las credenciales de los usuarios permitiéndole tener acceso a dichas aplicaciones.

Para obtener un resultado tangible sobre la utilidad de este nuevo sistema de autenticación y autorización, se desarrolló una investigación acerca de los diferentes tipos de sistemas de autenticación y autorización existentes (SSO), así como la parte de administración del sistema. Un resultado positivo que se ha tenido en consideración fueron las ventajas que proporcionan los sistemas de autenticación y autorización centralizados, destacando la importancia de mapeos entre diferentes entornos que manejan diferentes mecanismos de seguridad.

PALABRAS CLAVE

"[Insertar palabras clave que identifican el contenido temático del trabajo]"

TABLA DE CONTENIDOS

INTRODUCCIÓN

En la actualidad, la República Bolivariana de Venezuela es uno de los países donde se encuentran las más grandes reservas de petróleo del planeta. Posee una importante empresa productora de petróleo (PDVSA) que suministra recursos naturales como petróleo y gas natural a su país y ha hecho convenios con Cuba y también con países de Latino América para venderles el petróleo a bajo costo sin ningún tipo de interés. Adicionalmente PDVSA ha reforzado la filial Corporación Venezolana del Petróleo con el fin de acabar con la pobreza.

Se puede decir que esta empresa depende en gran medida de la operación de sistemas informáticos interrelacionados, por lo que se puede concluir que el cerebro de PDVSA abarca los programas de computación que manejan los procesos modulares y estos a su vez, los procesos administrativos, así como los datos referentes a los procesos industriales, los equipos donde recibe la información de control de datos y las redes electrónicas que permiten el flujo de información entre las distintas áreas.

El funcionamiento de PDVSA depende totalmente de sistemas automatizados y relacionados entre sí. En la mayoría de los casos se encuentra que cada uno de los servicios o aplicaciones cuenta con su propio componente de seguridad, el cual generalmente compromete la de todo el sistema, dado que su nivel de seguridad no es superior al nivel de seguridad del componente más vulnerable. Por esta razón PDVSA debe tener la máxima seguridad en el sistema informático, ya que en estos procesos y sistemas no están excluidos de ataques que ponen en peligro su funcionamiento y pueden provocar descalabros tanto en una empresa como en una nación.

Actualmente el Proyecto de desarrollo de la Intranet de PDVSA no cuenta con un sistema de seguridad que mantenga y fortalezca la seguridad informática de los procesos y sistemas con que cuenta la empresa; que ayude y permita a su vez mantener el control interno y ofrecer a los clientes servicios de valor agregado con altos niveles de seguridad y confianza. Además a los usuarios de la corporación les permitiría una autenticación única a la entrada del sistema y que a la vez sus credenciales sean distribuidas a todas las aplicaciones de la empresa, para así evitar que tengan que identificarse cada vez que vayan a acceder a una aplicación o servicio brindado.

PDVSA tampoco cuenta con un sistema que controle la autorización a las aplicaciones a las que son accedidas por los usuarios tanto de la empresa como a otros tipos de usuarios y a los servicios que corren dentro de la identidad. Debido a todo lo planteado anteriormente los usuarios de la corporación se ven obligados a usar sus credenciales en varios sistemas de forma repetida ya que los módulos de autenticación de estos son completamente independientes, por esta misma razón los administradores carecen de una herramienta única que gestione los roles de los usuarios en cada aplicación, tarea que, al encontrarse distribuida, recae sobre los responsables de cada sistema, hecho que contradice las políticas de seguridad de PDVSA.

Anteriormente mencionado y coherente con esta directriz, la autenticación y desarrollo de las aplicaciones corporativas de PDVSA con altos niveles de seguridad informática, establece un reto constante para fortalecer los fundamentos de seguridad informática básicos como son: la confidencialidad, la integridad, la disponibilidad, la autenticación, la autorización y el no repudio y la observación. Por tanto esta tesis presenta como posible solución un análisis de la arquitectura de implementación por credenciales de una estrategia de Single Sign-On para fortalecer los esquemas de seguridad e interacción de las aplicaciones y transferir la funcionalidad y complejidad de todos los componentes de seguridad a un solo servicio así como lograr que los usuarios se autentiquen una sola vez.

Single Sign-On (SSO)

Dentro de los Single Sign-On se encuentran cinco tipos principales, también llamados Reduced Sign-On systems (en inglés, sistemas de autenticación reducida), ejemplo de esto son:

- Enterprise Single Sign-On (E-SSO)
- Web Single Sign-On (Web-SSO
- Kerberos
- Federation
- OpenIDCon

Con la implementación de un sistema único de autenticación(SSO) en las aplicaciones de PDVSA se mejoraría la productividad de los usuarios en cuanto a que ya no estarían demorados por varios accesos y que no estarían obligados a recordar múltiples contraseñas y números de identificación ,además tendrían menos solitudes para establecer contraseñas olvidadas ,mejoraría la productividad

de desarrolladores ya que Single Sign-On proporciona desarrolladores con un marco común de autenticación lo que los desarrolladores no tendría que preocuparse por la autenticación en absoluto, así como la simplificación de la administración.

Además proporcionaría un mecanismo unificado para la gestión de la autenticación de los usuarios y aplica las normas de determinación de acceso de los usuarios a las aplicaciones y datos. De esta manera el sistema seria capaz de mapear las credenciales de los usuarios hacia todas las aplicaciones de dicho sistema y otros sistemas heterogéneos.

El Proyecto de la Intranet de PDVSA

Actualmente en la universidad se está ejecutando el Proyecto de la Intranet de PDVSA. Este proyecto fomentará la integración de los sistemas y aplicaciones que corren dentro de su entorno, en la cual los sistemas únicos de autenticación (Single Sign-On) juegan un papel fundamental. En él se desarrollan varias tareas para la creación del sistema. Una de las tareas principales del proyecto es la creación de un sistema de autenticación y autorización centralizado apropiado para las aplicaciones de PDVSA. Por esta razón, esta tesis se desarrollada como parte de este Proyecto.

Tecnologías aplicadas en el Proyecto de la Intranet de PDVSA

El proyecto de la Intranet de PDVSA está utilizando PHP como lenguaje de desarrollo de portales Web. Otra de las tecnologías que en el Proyecto se emplea es PostgreSQL que es un gestor de Base Datos. Además de estas tecnologías se utiliza para el desarrollo un SSO. Con la utilización de estas tecnologías se desarrolla el sistema de autenticación y autorización centralizado.

Por tanto el objeto de estudio de este trabajo se relaciona principalmente con los procesos de autenticación y autorización centralizados, para trabajadores, aplicaciones y servicios.

El campo de acción se encuentra enmarcado en los procesos de autenticación y autorización centralizados, para trabajadores, aplicaciones y servicios respectivamente, dentro de la infraestructura de PDVSA.

El objetivo general del presente trabajo investigativo es: Desarrollar un sistema de autenticación y autorización centralizado SSO para el entorno de intranet de PDVSA.

Como objetivos específicos de la investigación se tienen:

- Investigar acerca del surgimiento y desarrollo de la metodología de control de acceso SSO (Single Sign On).
- Desarrollar la documentación correspondiente con el estudio realizado de dicha metodología con respecto a sus usos actuales (Estado del Arte).
- Desarrollar la aplicación con la implementación que incluya la metodología de acceso único.
- Documentar el desarrollo paulatino de la aplicación utilizando una metodología de desarrollo de software.

Para cumplir estos objetivos específicos se proponen las siguientes tareas:

- Hacer un estudio de los sistemas únicos de autenticación ya existentes.
- Desarrollar la documentación correspondiente a la investigación antes realizada y que refleje el estado del arte del problema planteado.
- Implementar la aplicación correspondiente a la metodología de acceso estudiada, por módulos.
- Especializar la aplicación al área correspondiente a las aplicaciones de la empresa.
- Documentar el desarrollo del software apoyado en la metodología RUP para la documentación de aplicaciones software.

Posibles resultados

Se espera que al finalizar el desarrollo de la aplicación:

Disminuya la frecuencia con que se autentican los usuarios al solicitar algún servicio cuando emplean las aplicaciones de PDVSA. Además, se pretende que se autoricen las diferentes aplicaciones que funcionan en esta infraestructura, mejorando de esta forma la seguridad de las aplicaciones Web en el entorno de la infraestructura para PDVSA.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Este capítulo se refiere al estado del arte del problema planteado en la introducción. También se abordarán temáticas sobre las tecnologías que existen en el plano internacional y que pueden solucionar dicho problema con el estudio de técnicas de sistemas de autenticación y control centralizado. La integración de todas las tecnologías consultadas provee una vía para solucionar esta dificultad existente: el estudio de un SSO. Sistemas únicos de autenticación.

1.1 Sistemas de autenticación.

En el ecosistema de software de las empresas y corporaciones de hoy en día, la seguridad de acceso a las aplicaciones de la organización constituye un creciente desafío. Implementar y poner en marcha un modelo de seguridad de acceso implica navegar múltiples aplicaciones, en la mayoría de los casos, con muy pocas características comunes entre ellas. La norma hasta el momento ha sido que cada aplicación dentro de una empresa o corporación cuenta con un adecuado sistema de control de accesos y su propio repositorio de datos. En la actualidad dentro del ambiente de tecnología de cualquier empresa o corporación existen diversos métodos de autenticación y autorización de accesos que generan una gran ineficiencia.

Sin embargo, con la implementación de un agente Single Sign-On (SSO) el sistema se encarga de almacenar, en una base datos o directorios protegidos, las credenciales que permiten al usuario acceder a cada una de las aplicaciones o servicios en el momento que lo desee, ya que el proceso de autenticación se realiza de manera transparente para el usuario, una vez que éste ha sido autenticado por medio de la arquitectura SSO. Se puede decir que con dicha implementación el sistema simplificaría y centralizaría el control de accesos a todas las aplicaciones de la empresa o corporación, reduciendo el costo en la administración de seguridad, lo que logra un mejor rendimiento y velocidad de los procesos de autenticación y acceso que facilita a los usuarios la interacción con los sistemas de la empresa, simplificando el manejo de claves y lo fundamental es que aumenta los niveles de seguridad, ya que contará con una plataforma central para el manejo de la seguridad en todos los procesos de autenticación y acceso a sus aplicaciones.

El agente SSO se refiere al acceso a múltiples recursos por medio de un único proceso de ingreso. Gran cantidad de las arquitecturas implementadas en diferentes organizaciones han sido diseñadas con el objeto de dar acceso a los usuarios a múltiples servicios Web y aplicaciones. En la mayoría de los casos se encuentra que cada uno de los servicios o aplicaciones cuenta con su propio componente

de seguridad, el cual generalmente compromete la seguridad de todo el sistema, dado el nivel de seguridad del componente más débil, el cual determina el nivel de confianza del sistema en su conjunto.

1.1.1 Tipos de Sistemas Single Sign-On

Existen cinco tipos principales de sistemas SSO, también conocidos como Reduced Sign-On Systems (Sistemas de Autenticación Reducida). Los cuales consideran a continuación.

- ***Enterprise Single Sign-On (E-SSO)****: también llamado Legacy Single Sign-On, el cual funciona luego de una autenticación primaria, interceptando los requerimientos de autenticación presentados por las aplicaciones secundarias para completarlos con el usuario y la contraseña. Los sistemas E-SSO permiten interactuar con sistemas que pueden deshabilitar la presentación de la pantalla de login.*

- ***Web Single Sign-On (Web-SSO)****: conocido como Web Access Management (Web-AM), trabaja sólo con aplicaciones y recursos que se acceden vía Web. Los accesos son interceptados con la ayuda de un servidor Proxy o de un componente instalado en el servidor Web destino. Los usuarios no autenticados que tratan de acceder son redirigidos a un servidor de autenticación y regresan sólo después de haber logrado un acceso exitoso. Se utilizan cookies, para reconocer aquellos usuarios que acceden y su estado de autenticación.*

- ***Kerberos****: es un método popular de externalizar la autenticación de los usuarios. Los usuarios se registran en el servidor Kerberos y reciben un ticket, que luego utilizan para obtener acceso.*

- ***Federation****: es una nueva manera de concebir este tema, también para aplicaciones Web. Utiliza protocolos basados en estándares para habilitar que las aplicaciones puedan identificar los clientes sin necesidad de autenticación redundante.*

- ***OpenID****: es un proceso de SSO distribuido y descentralizado donde la identidad se compila en una URL de forma que cualquier aplicación o servidor puede verificar.[1]*

1.2 Arquitecturas Single Sign-On

Existen diferentes tipos de arquitecturas que permiten implementar un SSO. Cada una de ellas posee características que la hace más apropiada para determinado tipo de organización. La decisión de adoptar una u otra depende básicamente de los recursos computacionales y económicos disponibles y también de lo que se quiera lograr con el sistema y las decisiones de diseño establecidas por el equipo del proyecto.

Los tres componentes fundamentales que forman la arquitectura son:

- ***Interfase****: es el modo en que el SSO interactúa con una determinada aplicación. Frecuentemente reside en el cliente, y es conocido como Agente SSO.*
- ***Administración****: es el mecanismo que permite configurar, mantener y monitorear el proceso de SSO.*
- ***Credenciales****: cada aplicación a la que accede el usuario requiere información confidencial (nombre de usuario, contraseña, etc.), que agrupadas recibe el nombre de credenciales. Las credenciales deben almacenarse de manera protegida para que sea únicamente el agente SSO quien pueda acceder a ellas [1].*

Como se había planteado anteriormente para la implementación de un sistema SSO se pueden tener en cuenta las diferentes tipos de arquitecturas existentes de las cuales se mencionan a continuación:

1.2.1Password vault

Se trata de la configuración más básica para implementar SSO utilizando credenciales. En este caso los tres elementos de la arquitectura se encuentran ubicados en el cliente y, por lo tanto, es justamente allí desde donde se accede a las aplicaciones, con tal propósito se deben almacenar previamente las credenciales correspondientes, para que puedan suministrarse a las aplicaciones cuando sea necesario, como se ilustra en la figura 1.

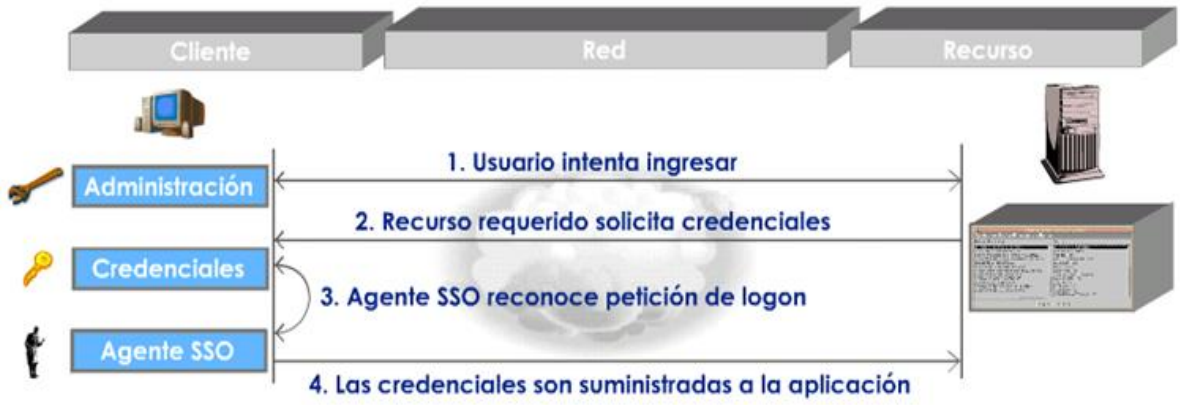


Figura 1. Arquitectura Password vault

Dentro de las características que posee la arquitectura Password vault se encuentran las funciones administrativas limitadas donde la administración de cada uno de los clientes debe realizarse desde la estación correspondiente y por lo tanto generalmente termina quedando a cargo del usuario. No es posible actualizar los clientes de manera masiva en toda la organización, requiere que se realice máquina por máquina.

Dentro de las ventajas de este tipo de arquitectura se encuentran:

- La ventaja de que no es mucho más complicada que la instalación de un nuevo software en el equipo del cliente.
- Requiere pocos recursos computacionales (un servidor central donde residen las diferentes aplicaciones y los clientes necesarios).

Dentro de sus desventajas se pueden mencionar:

- La administración local obliga a tomar medidas adicionales de seguridad informática y control de acceso por parte de la empresa.
- El nivel de transparencia para el usuario es bajo, ya que éste generalmente se encuentra comprometido con la administración del proceso de ingreso.

- El almacenamiento local de credenciales no permite que el usuario acceda a las aplicaciones desde múltiples estaciones.

- La información entre el cliente SSO y el servidor no viaja cifrada.

1.2.2 Administración centralizada con almacenamiento local de credenciales

Con la implementación de la arquitectura de Administración centralizada con almacenamiento local de credenciales se busca solucionar los principales inconvenientes que presenta la arquitectura Password Vault que ofrece un mecanismo para controlar y supervisar el proceso de ingreso y elimina la necesidad de configurar el SSO a cada uno de los clientes. Esta arquitectura se ilustra en la figura 2.

Figura 2. Arquitectura Administración centralizada con almacenamiento local de credenciales.

Características de esta arquitectura:

- Incluye un servidor central que permite realizar labores de administración.

- El software cliente es autónomo durante el proceso de autenticación, debido a que durante este proceso la labor de administración se restringe a realizar el monitoreo de los clientes.
- Las credenciales permanecen en el cliente.

Sus ventajas son:

- Control centralizado de la configuración y monitoreo del software del cliente.
- Las labores de administración tienen un bajo grado de complejidad.

Sus desventajas son:

- El almacenamiento de las credenciales en el cliente hace que se deban tomar medidas de control de acceso y confidencialidad de la información.
- Una vez que el cliente se ha conectado, el administrador del SSO sólo puede monitorear la conexión y no podría efectuar acciones de desconexión o cambio de su configuración.
- La información entre el cliente que utiliza SSO y el servidor no viaja cifrada.

1.2.3 Administración y almacenamiento de credenciales centralizados

La arquitectura SSO con administración y almacenamiento centralizado de credenciales (Figura 3) pretende solucionar los principales inconvenientes encontrados en la arquitectura que almacena las credenciales localmente, la cual ya fue mencionada y se representa en la figura 3.

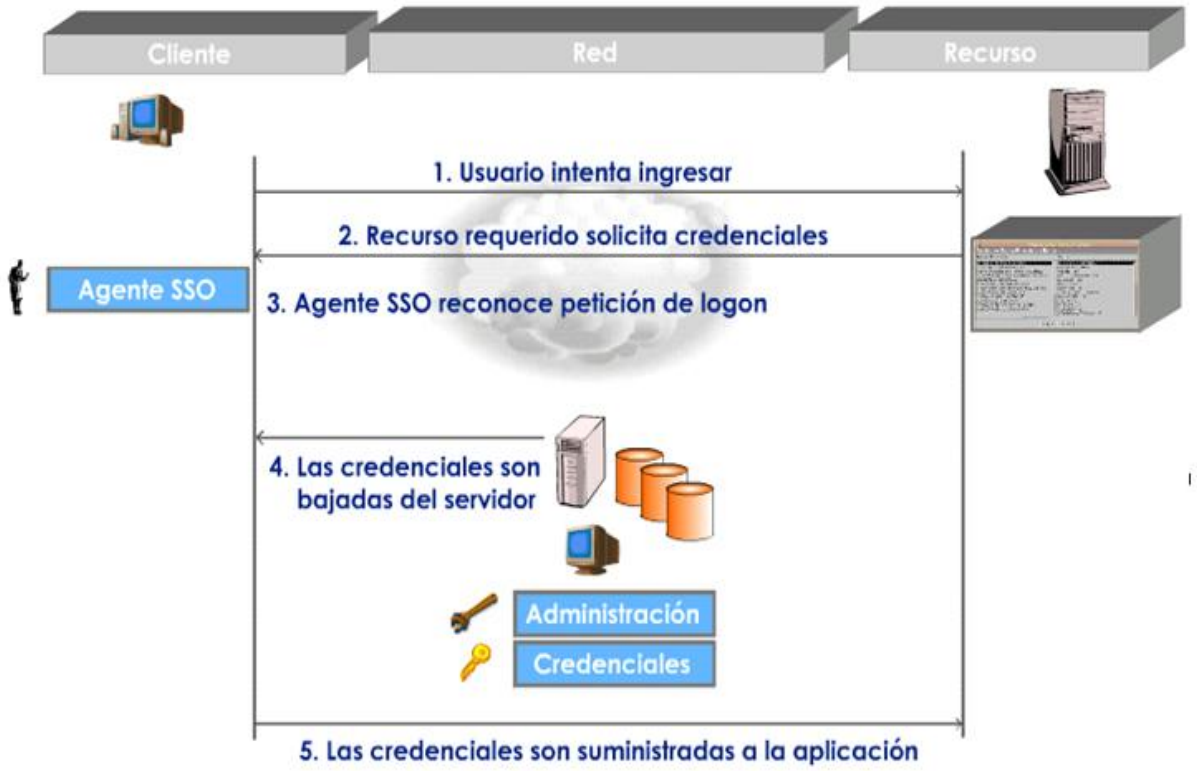


Figura 3. Arquitectura Administración y almacenamiento de credenciales centralizados

Características de esta arquitectura:

- Las credenciales se envían a un servidor central, que entrega las credenciales al cliente correspondiente en el momento que hace el ingreso.
- El administrador determina la frecuencia con que se descargan las credenciales del servidor (por sesión, por login).

Sus ventajas:

- Permite a los usuarios el acceso a las aplicaciones desde cualquier estación, mediante su previa autenticación.
- Ofrece la administración centralizada de credenciales disminuyendo su posible manipulación por el cliente.

Desventajas:

- Se crea un único punto de falla, convirtiendo al SSO en un Gateway para todos los recursos de la organización, ya que el servidor debe contactarse cada vez que se realice un ingreso. El acceso a todas las aplicaciones de la organización depende del servidor central.
- La configuración no presenta redundancia, ni recuperación entre fallas ni respaldo.
- La información entre el cliente que utiliza SSO y el servidor no viaja cifrada.

1.2.4 Arquitectura SSO totalmente distribuida

La arquitectura SSO totalmente distribuida (mostrada en la figura 4) se caracteriza principalmente por separar el servidor de la base de datos, lo cual la hace completamente modular. Esta arquitectura soluciona los problemas encontrados en las arquitecturas anteriormente presentadas y adicionalmente ofrece múltiples ventajas.

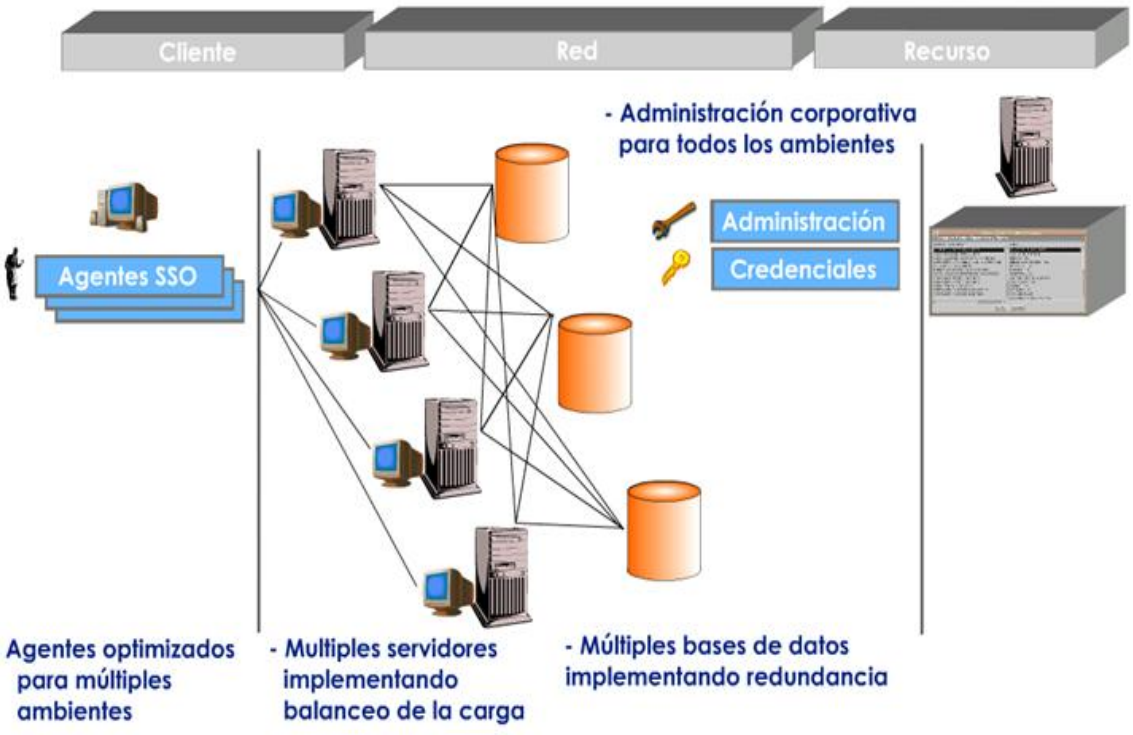


Figura 4.Arquitectura SSO totalmente distribuida

Las características de esta arquitectura son:

- La información se accede en el momento de ingreso.
- Cuenta con SSO avanzados que utilizan bases de datos escalables que soportan redundancia (por ejemplo, SQL Server u Oracle).
- Las bases de datos se encuentran sincronizadas con el fin de lograr redundancia y respaldo.
- El proceso de ingreso se envía a un recurso de red, siempre que el agente SSO pueda establecer conexión IP a un servidor SSO, las credenciales podrán solicitarse (y almacenarse en memoria caché para realizar offline logon) y el ingreso podrá realizarse.
- El servidor resulta ser una aplicación independiente que cuenta con un administrador diferente.
- La información se almacena en bases de datos comerciales o en directorios de manera encriptada. Sin embargo, la información entre el cliente que utiliza SSO y el servidor no viaja cifrada.

Sus ventajas son:

- Los agentes SSO se encuentran optimizados para múltiples ambientes (Terminal Server, Web, Win32).
- Contiene múltiples servidores implementando balanceo de la carga para aumentar la disponibilidad y la atención de los requerimientos de autenticación.
- El hecho de contar con múltiples servidores adicionales hace que disminuya la latencia de la red.
- Contiene múltiples bases de datos sincronizadas, lo que implementa la redundancia.
- Permite realizar funciones de administración corporativa en todos los ambientes.

Desventajas:

- Solución altamente costosa por el ambiente distribuido requerido.

- La implementación técnica se demora por interacción entre múltiples sistemas operacionales.
- Soporte y administración complejos originados por la consideración anterior.

1.2.5 **Administración y almacenamiento de credenciales centralizados que garantiza alta disponibilidad y redundancia.**

La arquitectura SSO con administración y almacenamiento centralizado de credenciales que garantiza alta disponibilidad y redundancia (Vea la figura 5) es una adaptación de la arquitectura presentada en Administración y almacenamiento de credenciales centralizados incorpora algunas de las ventajas de la arquitectura totalmente distribuida, presentada en arquitectura SSO totalmente distribuida, constituye unión de estas arquitectura para lograr un mejor rendimiento.

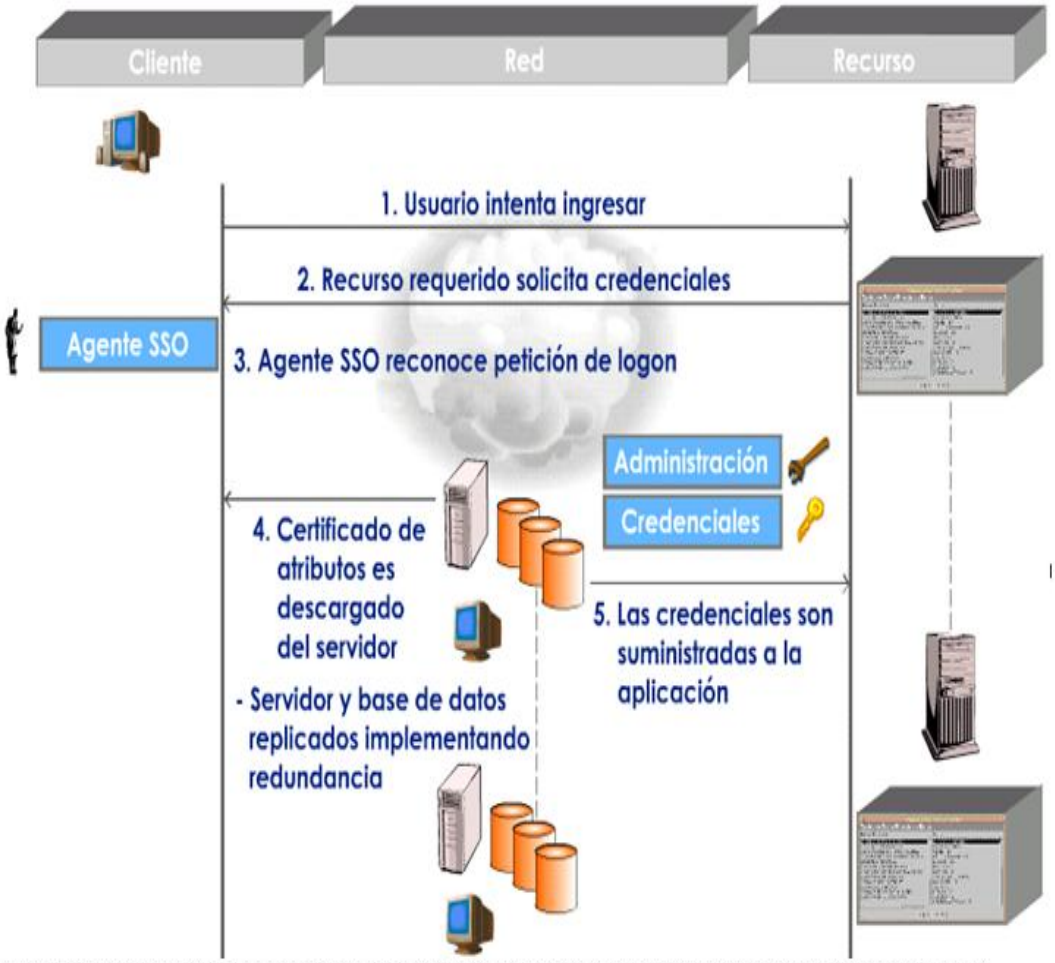


Figura 5.Arquitectura Administración y almacenamiento de credenciales centralizados que garantiza alta disponibilidad y redundancia.

Sus características son:

- Las credenciales se almacenan en un servidor central el cual entrega un certificado al cliente y comprueba las credenciales necesarias para la respectiva aplicación en el momento de hacer el ingreso.
- Incorpora infraestructura replicada con el fin de manejar la contingencia y redundancia en tiempo real.
- Ofrece alta disponibilidad mediante software.

Ventajas de esta arquitectura:

- Permite a los usuarios el acceso a las aplicaciones desde cualquier estación.
- Ofrece administración centralizada.
- Su infraestructura duplicada permite implementar alta disponibilidad y redundancia.
- Tanto el hardware como el software se encuentran debidamente especificados para enfrentar una situación de contingencia.

Sus desventajas son:

- La alta disponibilidad y redundancia que provee se basa en su infraestructura replicada, la cual la hace costosa tanto a nivel de hardware y software como a nivel de su administración y control.
- La información entre el cliente que utiliza SSO y el servidor no viaja cifrada.

1.3 Aplicaciones exitosas que hacen uso del SSO.

Se puede decir que hoy en día existen muchas aplicaciones exitosas que utilizan la implementación de un SSO, ya que permite un acceso fácil a plataformas y aplicaciones a través de una interfaz de login única facilitando la vida de los usuarios e incrementando su productividad .A demás incrementa su el nivel de seguridad usando un buen nivel de autenticación y el manejo de las contraseñas.

1.3.1Google

Google. Es uno de los ejemplos de soluciones de seguridad debido a que se integra con sus sistemas de autenticación y autorización para proporcionar un modo seguro es el que brinda Google Search Appliance. En la actualidad todas las compañías tienen un gran número de información muy importante y el servicio prestado por Google para indexar información de uso público como también restringida, y habilita las políticas de seguridad para la información de la empresa en el momento de la búsqueda. Además, proporciona a los usuarios un mecanismo por medio del cual pueden buscar información restringida de forma muy segura. La tecnología de búsqueda empresarial de Google se integra con sus sistemas de autenticación y autorización para proporcionar una manera de utilización segura a los usuarios.

Se pude decir que Google Search Appliance es un potente motor de búsqueda que rastrea el contenido y una vez encontrado los muestra de forma segura. Por lo general el motor de búsqueda puede configurarse para que guarde tanto el contenido público como protegido. Cuando el usuario accede a un contenido protegido, aparece un cuadro de diálogo en la sección del navegador que solicita del usuario las credenciales necesarias, evento que tiene lugar una vez por sesión. El motor de búsqueda ha estructurado la autenticación que brinda el servicio en dos fases principales: rastreo/indexación y publicación. La fase de rastreo crea un índice de información que adquiere a través de los mecanismos de acceso a contenidos integrados.

Una vez que el servicio rastrea y adquiere la información, la aplicación utiliza credenciales de acceso facilitadas por el administrador del sistema. Éstas pueden ser de inicio de sesión única (SSO) para sistemas basados en formularios donde la autenticación se implanta en entornos Web protegidos por inicio el de sesión única y está basada en formularios mediante solicitudes HEAD, cuyo contenido está basado en una aplicación Web para un servidor Web. El motor de búsqueda puede utilizar tanto el reenvío de cookies como la suplantación absoluta del usuario. En ambos casos el motor captura la información de acceso en una cookie y la reenvía a los sistemas que se están rastreando. Para sistemas de credenciales de autenticación básica como las credenciales NTLM (nombre de usuario, contraseña y dominio) y certificados del cliente, por ejemplo, el X.509, funciona casi en todas las implementaciones del servidor Web que son compatibles al menos con HTTP/1.0. También es admitido por servidores basados en el sistema operativo (SO) Microsoft®. Generalmente, si las credenciales de un usuario residen en un SO Microsoft y se utilizan NTLM, el motor de búsqueda podrá aprovecharlas.

El proceso de publicación tiene lugar cuando los usuarios realizan una consulta y en eso momento pueden especificar, a través de la interfaz de búsqueda, si quieren buscar información de carácter publico o privilegiado. Si el usuario se decide por la privilegiada, el sistema le solicita al usuario sus credenciales de acceso basándose en los métodos de autenticación y autorización configurados, es decir, como ya la aplicación esta integrada por un inicio de sesión única (SSO) el usuario es dirigido directamente al servidor de autenticación en el cual se registrará y luego el sistema se encargará de darle las credenciales correspondientes para poder realizar la búsqueda y si el usuario ya está autenticado podrá continuarla sin necesidad de autenticarse.

El sistema realiza la búsqueda en el índice para obtener una lista de los posibles resultados que coincidan con lo que se esta buscando. Sin embargo, antes de mostrar la lista de los resultados obtenidos, la aplicación utiliza la cookie de SSO en nombre del usuario para autorizar los resultados ante el sistema de origen. Los que no superan la autorización se excluyen de la lista, y únicamente se muestran al usuario aquéllos que hayan sido validados y el sistema asegura que sólo puedan verlos los usuarios que estén correctamente registrados.

El sistema de búsqueda Google Search Appliance cumple con estándares abiertos establecidos. La aplicación puede utilizar un gran número de mecanismos en el sector de la autenticación de usuarios, incluyendo también la autenticación basada en LDAP y Active Directory, sistemas de acceso único (SSO) basados en formularios, como Netegrity y Oblix, y certificados de un cliente, digamos el X.509. La autorización proporciona a la aplicación el nivel del sistema-origen del documento, utilizando de nuevo el sistema SSO empresarial, autenticación HTTP básica o autorización basada en NTLM. También es compatible con interfaces que tiene su base en SAML para la autenticación y la autorización. La interfaz de proveedor basada en SAML 2.0 utiliza los estándares XML emergentes para permitir la autenticación de terceros, así como la autorización de resultados externos. Con las interfaces de proveedor de servicios de autorización y autenticación, Google Search Appliance se puede integrar de forma fácil y segura a todos los entornos de control de acceso empresarial. Además, proporciona seguridad a las empresas con la facilidad de integrarlas de inmediato después de haber implantado el sistema.

Al sistema de búsqueda Google Search Appliance no le hace falta la creación de identidades de usuario nuevos o listas de control de acceso (ACL) para implementar una búsqueda segura en la empresa. En su lugar Google utiliza el sistema de gestión de identidad existente y las políticas de control de acceso de que ya dispone en sus sistemas de contenido.

1.3.2 Windows Live ID

La plataforma .NET contiene los cimientos para la nueva generación del software, ella utiliza los servicios Web como medio para poder interactuar con distintas tecnologías que permiten conectar diversos sistemas operativos, dispositivos físicos, información y usuarios. La idea central de la plataforma ·NET es proveer servicios, a los cuales se puede acceder desde Internet, que constituye una infraestructura global de comunicación. La plataforma ·NET permite usar Internet y su capacidad de distribución para que los usuarios accedan desde cualquier dispositivo, en cualquier sistema operativo y lugar por la funcionalidad que los servicios Web proveen. Unos de los componentes más importantes que lo integran, provistos por Microsoft es ·NET Passport o Windows Live ID, como se denomina actualmente, que es un conjunto de servicios Web.

Desde 1999 Windows Live ID permite que un usuario se autentique una sola vez y luego acceda a los sitios que hacen uso del servicio sin necesidad de volver a identificarse. Estos sitios pueden tener en cuenta datos del usuario como su perfil de usuario para adaptar la interface o proveer los servicios específicos. El sitio debe conectarse al servicio Web .NET Passport y hacer llamadas a componentes del servicio ·NET Passport para hacer uso de un servicio . El usuario, por su parte, debe tener una cuenta en Hotmail o MSN y haber sido autenticado por el servicio ·NET Passport. Una vez hecho esto, él puede navegar por distintos sitios y si éstos utilizan .NET Passport entonces el usuario no tiene necesidad de identificarse o de ingresar sus datos personales.

La mayoría de los sitios Web y las aplicaciones que utilizan Windows Live ID son los sitios de Microsoft tales como Hotmail, MSNBC, MSN, la Xbox 360, Xbox Live, el .NET Messenger Service, MSN Zune o las suscripciones como MSN Music. Mediante el pasaporte Microsoft crea una base de datos tipo cliente en extremo poderosa. Microsoft puede recolectar información de cualquier sitio habilitado con el pasaporte para que Microsoft sepa a cuáles acciones les da seguimiento en Investor.com, las páginas Web que visualiza en MSN.com y hacia donde viaja a través de Expedia.com. Cuando se mueve de un sitio habilitado con pasaporte a otro, ellos también comparten esa información. También se debe decir que controla las suscripciones a los sitios que utilicen este sistema de registro, entre los que se encuentra el Microsoft Center para Empresas y Profesionales. De este modo, podrá gestionar sus suscripciones, darse de alta, de baja o modificar sus datos de registro. Este servicio que brinda Microsoft es personal ya que contiene la información que se proporciona durante el proceso de registro, además brinda seguridad porque las paginas de inicio son seguras, esto se refleja en el candado que tiene la barra del explorador, lo que implica que la conexión a esas páginas es segura.

1.3.3OpenID

Hoy en día se sabe que en Internet se encuentran aplicaciones Web que utilizan OpenID que permanecen accesibles al servicio prestado por Internet, por esto elegir un método de autenticación y autorización de los usuarios a dichas aplicaciones y servicios es siempre un factor determinante en la seguridad de los sistemas informáticos.

El proyecto OpenID no es más que un estándar que posee un sistema descentralizado de identificación, él permite a los sitios Web que le brindan soporte el acceso de los usuarios sin tener necesidad de crear su propia cuenta. Simplemente se necesita un proveedor de identidad (IdP), un ejemplo de esto es Yahoo que acaba de adquirir el proveedor de cuentas (OpenID 2.0) como estándar y gracias a eso ha ganado casi 250 millones de usuarios potenciales.

Este estándar (OpenID) es un sistema de autentificación seguro, libre y muy fácil de usar, ya que facilita la identificación en muchas páginas Web con sólo introducir la dirección URL de OpenID, de forma que sólo mantendrá una ID global para los accesos a Web con soporte OpenID, que es un registro rápido y único, pues posee la facilidad de autenticarse una sola ves y tener acceso a muchas aplicaciones tipo Web.

Se puede decir que el usuario en el sistema de autenticación crea un identificador en un servidor que verifique OpenID y el proveedor de identidad confirma la identificación de los usuarios a los sitios que soporten este sistema. Seguidamente el usuario, después de haberse registrado, obtiene la URL (o más bien la XRI) que él puede usar para identificarse, además si posee una pagina Web puede incluir unas etiquetas en ella y usar su propio dominio como código de identificación.

La utilización de OpenID implica una arquitectura Single Sign-On, que no especifica el mecanismo de autenticación por lo que la seguridad de la conexión dependerá de la confianza del cliente en el proveedor de identidad. Se debe destacar que según sea la implementación de éste ofrece una autenticación segura y mayor seguridad en el sistema.

Una implementación sencilla del acceso a un proveedor de identidad consiste, básicamente, en solicitar al usuario su identificador y requerir del proveedor de identidad los datos necesarios. En el proveedor, el usuario gestionará si quiere proporcionar a nuestra aplicación sus datos siempre, o sólo en esa ocasión. Los datos que proporciona el usuario dependerán de su elección, siendo obligatorio facilitar su e-mail y su nombre, y es opcional el resto de información de su perfil. A continuación, en la figura 6, se puede apreciar el esquema de identificación que sigue OpenID.

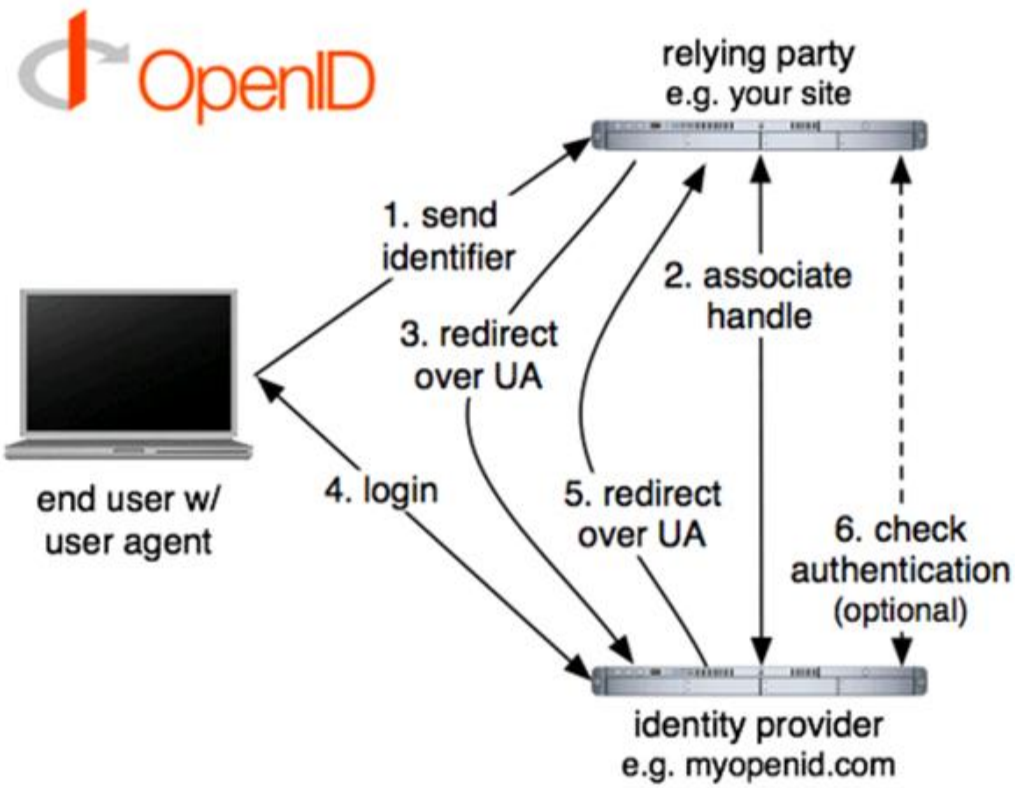


Figura 6. Esquema de identifcación que sigue OpenID

1.4 Kerberos

Éste es un método popular que permite hacer externa la autenticación de los usuarios. Kerberos sigue la arquitectura SSO basada en el paso de tokens, llamados tickets en su última versión (KerberosV).

La autenticación en este método viene suministrada, cuando el usuario se valida en el servidor de autenticación Kerberos y él devuelve una clave de sesión, o sea, un ticket general de comunicación con el servidor de autenticación y el cliente cada vez que quiera acceder a un recurso del servidor de autenticación genera un ticket para ese recurso, luego el servidor comprueba que el ticket enviado por el cliente es válido, y permite el acceso.

Sus ventajas son:

- Los servicios de redes más aceptados usan esquemas de autenticación basados en contraseñas lo que implica que los esquemas requieran nombre de usuario y contraseña

para una autenticación en un servidor en la red, para que sea seguro la red tiene que estar inaccequible para usuarios que sean externos y todos los usuarios de la red tienen que ser de mucha confianza.

- Elimina los servicios de redes más convencionales que usan esquemas de autenticación basados en contraseñas.

Destacar que Kerberos, como protocolo de seguridad, usa una criptografía de claves simétricas, lo que significa que la clave utilizada para cifrar es la misma clave utilizada para descifrar o autenticar usuarios. Esto permite a dos computadores en una red insegura, demostrar su identidad mutuamente de manera segura y además restringe los accesos sólo a usuarios autorizados y autentica los requerimientos a servicios, asumiendo un entorno distribuido abierto, en el cual usuarios ubicados en estaciones de trabajo acceden a estos servicios en servidores distribuidos a través de una red.

1.5 Lenguaje de Programación para la Web.

PHP es el acrónimo de Hipertext Preprocesor. ***Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma****, rápido, con una gran librería de funciones y mucha documentación.[2]*

Este lenguaje fue creado en 1994 por Rasmus Lerdorf y como PHP está desarrollado con política de código abierto. A lo largo de su historia ha tenido muchas contribuciones de otros desarrolladores.

La última versión es PHP5, que utiliza el motor Zend-2 y presenta mejoras significativas y un entorno de programación orientado a objetos (POO) que lo hace mucho más completo y permite que el PHP proporcione un alto rendimiento a las aplicaciones Web empresariales a nivel de las plataformas J2EE y .NET. El principal objetivo de PHP5 ha sido mejorar los mecanismos de POO para solucionar las carencias de las anteriores versiones.

Sus características son:

- Posee una sintaxis semejante a la de C.
- Dispone de una alta conectividad con la mayoría de los Sistemas de Gestión de Bases de Datos.
- Es Open-Source y de obtención gratuita.

- Portátil y con multiplataforma (W95, 98, 2000, XP, NT, Unix, Linux y otras), lo cual permite su desarrollo desde sistemas operativas con bases heterogéneas.
- Proporciona soporte para la mayoría de los protocolos de comunicación de Internet (HTTP, IMAP, FTP, LDAP, INMP y otros).

Sus ventajas son:

- Muy fácil de aprender.
- Se caracteriza por ser un lenguaje muy rápido.
- Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- Soporta en cierta medida la orientación a objeto, clases y herencia.
- Capacidad de conexión con la mayoría de los manejadores de base de datos como MysSQL, PostgreSQL, Oracle, MS SQL Server, entre otros.
- Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- Capacidad de expandir su potencial utilizando módulos.
- Incluye gran cantidad de funciones.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- No requiere definición de tipos de variables ni manejo detallado de bajo nivel.

1.6 PostgreSQL

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) basado en el proyecto POSTGRES, de la universidad de Berkeley, que ha sido desarrollado de varias formas desde 1977.

El proyecto PostgreSQL tiene actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto. Incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones,

disparadores, reglas e integridad transaccional. A pesar de esto, PostgreSQL no es un sistema de gestión de base de datos puramente orientado a objetos. Está ampliamente considerado como el sistema de base de datos de código abierto más avanzado del mundo.

Se pueden encontrar principales características de este gestor de bases de datos como son:

- Implementación del estándar SQL92/SQL99.
- Soporta distintos tipos de datos, y además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP…), cadenas de bits, etc. También permite la creación de tipos propios.
- Incorpora una estructura de datos array.
- Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etcétera.
- Permite la declaración de funciones propias, así como la definición de disparadores.
- Soporta el uso de índices, reglas y vistas.
- Incluye herencia entre tablas, aunque no entre objetos, ya que no existen, por eso a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

Las ventajas son:

- Costo Es libre
- Rapidez.
- Es compatible con cualquier Sistema Operativo
- Se puede modificar el motor acorde con las necesidades del usuario, ya que este permite entrar a las fuentes, sin embargo, los creadores no se responsabilizan con los errores introducidos.

1.7 Que es un Web Service?

Un Web Service es un componente de software que se comunica con otras aplicaciones codificando los mensaje en XML y enviando estos mensaje a través de protocolos estándares de Internet tales como el Hypertext Transfer Protocol (HTTP). Intuitivamente un Web Service es similar a un sitio Web que no cuenta con una interfaz de usuario y que da servicio a las aplicaciones en vez de a las personas. Un Web Service, en vez de obtener solicitudes desde el navegador y retornar páginas Web como respuesta, lo que hace es recibir solicitudes a través de un mensaje formateado en XML desde una aplicación, realiza una tarea y devuelve un mensaje de respuesta también formateado en XML.[3]

1.7.1 Web Service (SOAP)

SOAP (*Simple Object Access Protocol*) es un protocolo estándar creado por Microsoft, IBM y otros, está actualmente bajo el pronóstico de la W3C que define cómo dos objetos en diferentes procesos que pueden comunicarse por medio de intercambio de datos XML. SOAP es uno de los protocolos utilizados en los servicios Web. Es independiente de la plataforma, y del lenguaje. Esta basado en XML y es la base principal de los Web Services. Los mensajes SOAP son documento XML propiamente dicho. Un mensaje SOAP se parece mucho a una carta: es un sobre que contiene una cabecera con la dirección del receptor del mensaje, un conjunto de opciones de entrega (tal como la información de encriptación), y un cuerpo o body con la información o data del mensaje.

Ventajas de la utilización de SOAP:

- Es sencillo de implementar, probar y usar
- Atraviesa "firewalls" y routers, pues estos "piensan" que es una comunicación HTTP.
- Tanto los datos como las función es se describen en XML, lo que permite que el protocolo no sólo sea más fácil de utilizar sino que también sea muy sólido.
- Es independiente del sistema operativo y procesador
- Se puede utilizar tanto de forma anónima como con autenticación (nombre/clave).
- Facilidad para utilizar cualquier lenguaje.
- No se encuentra fuertemente asociado a ningún protocolo de transporte.

- No está atado a ninguna infraestructura de objeto distribuido.
- Aprovecha los estándares existentes en la industria.
- Permite la interoperabilidad entre múltiples entornos

1.8 Arquitectura orientada a servicios

La Arquitectura Orientada a Servicios (en inglés Service-Oriented Architecture o SOA), Define la utilización de servicios para dar soporte a los requerimientos de software del usuario.

SOA es una arquitectura de software que permite la creación y/o cambios de los procesos de negocio desde la perspectiva de TI de forma ágil ,mediante la constitución de nuevos procesos utilizando las funcionalidades de negocio que están contenidas en la infraestructura de aplicaciones actuales o futuras (expuestas bajo la forma de webservices).

SOA precisa las siguientes capas de software:

- Aplicativa básica, sistemas desarrollados bajo cualquier arquitectura o tecnología, geográficamente dispersos y bajo cualquier figura de propiedad.
- De exposición de funcionalidades, donde las funcionalidades de la capa aplicativas son expuestas en forma de servicios (webservices).
- De integración de servicios, facilitan el intercambio de datos entre elementos de la capa aplicativa orientada a procesos empresariales internos o en colaboración.
- De composición de procesos, que define el proceso en términos del negocio y sus necesidades, y que varia en función del negocio.
- De entrega, donde los servicios son desplegados a los usuarios finales.

Los beneficios que podemos obtener con la adopción de SOA son:

- Mejora en los tiempos de realización de cambios en procesos.
- Facilidad para evolucionar a modelos de negocios basados en tercerización.

- Facilidad para abordar modelos de negocios basados en colaboración con otros entes (socios, proveedores).
- Poder para reemplazar elementos de la capa aplicativa SOA sin disrupción en el proceso de negocio.
- SOA proporciona una metodología y un marco de trabajo para documentar las capacidades de negocio y puede dar soporte a las actividades de integración y consolidación.

Cuando se refieren de una arquitectura orientada a servicios, están hablando de un juego de servicios residentes en Internet o en una intranet, usando servicios Web. Hay un juego de estándares de los que se habla ligados a los servicios Web. Incluyen los siguientes:

- XML
- HTTP
- SOAP
- WSDL
- UDDI

Hay que considerar, sin embargo, que un sistema SOA no necesariamente necesita utilizar estos estándares para ser "orientado a servicios" pero es altamente recomendable su uso.

Toda lo tratado hasta aquí es el asiento para la construcción de una aplicación que puede ser de gran apoyo al trabajo que realizan las aplicaciones y servicios con que cuenta la infraestructura de PDVSA , esto se logra mediante la implementación de Web Single Sign-On (Web-SSO) ya que este facilita el trabajo con las aplicaciones que se acceden por vía Web y estos accesos son interceptados con la ayuda de un proxy y los usuarios que estén autenticados son redirigidos a un servidor de autenticación y logran un acceso exitoso a las aplicaciones después de haber realizado una satisfactoria autenticación.

Se utiliza para este SSO una arquitectura de Administración y almacenamiento de credenciales centralizados y se propone que sea con redundancia, y esto se logra agregando otro servidor de base de datos e insertando una aplicación para replicar los datos, esto se logra con Postgret y el

SLONY, es decir con este nuevo servidor de base de datos instalado si se cae algún otro servidor de base de datos el sistema sigue trabajando porque entonces pasa automáticamente para el otro.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

La teoría con las generalidades para diseñar un sistema de autenticación y autorización centralizada fue el primer objetivo del capítulo anterior. En este capítulo nos apoyaremos en dicha teoría para describir un producto que posea las condiciones que debe cumplir un sistema automatizado y que sea fácilmente vinculable a los sistemas de la intranet de PDVSA.

Este producto debe trabajar en conjunto con un mecanismo de autenticación que permita acceso a múltiples aplicaciones o servicios dentro de la infraestructura donde se aplique. A partir de esta integración los usuarios de la corporación contarán con una autenticación única que les facilitará acceder a los servicios y aplicaciones existentes en la red interna de PDVSA autenticándose una sola vez.

Además fortalecerá los esquemas de seguridad e interacción con las aplicaciones. Los accesos son interceptados con la ayuda de un componente instalado en la aplicación destino. Los usuarios no autenticados que tratan de acceder son redirigidos a un servidor de autenticación y regresan solo después de haber logrado un acceso exitoso. Se utilizan cookies y sesiones, para reconocer aquellos usuarios que acceden y su estado de autenticación.

2.1 Estructura y funcionalidad.

Con este trabajo de diploma se pretende realizar una aplicación que facilite la integración de los servicios y aplicaciones que corren dentro la infraestructura corporativa de PDVSA, así como también una infraestructura robusta, escalable y segura. De esta mezcla se tiene el SSO: Single Sign-On.

Este SSO funciona con base a la definición de mapeos de credenciales entre diferentes entornos que maneja diferentes mecanismos de seguridad de forma tal que los aplicativos de *Enterprise Application Integration* (EAI) puedan hasta cierto punto filtrar las credenciales de seguridad a través de toda la infraestructura tecnológica de la organización, sin importar que diferentes sistemas utilicen diferentes conjuntos de credenciales de seguridad.

El Single Sign-On debe vincularse a una arquitectura implementada con el objeto de dar acceso a los usuarios a múltiples servicios o aplicaciones Web. Uno de los principales objetivos que implementa una arquitectura SSO es transferir la funcionalidad y complejidad de todos los componentes de seguridad a un solo servicio de Single Sign-On. En una arquitectura SSO, todos los mecanismos de seguridad se encuentran centrados en el SSO, siendo este el único punto de autenticación y registro

del sistema. Se pude mencionar como otro de los principales objetivos de una arquitectura Single Sign-On es que los usuarios deben hacer el proceso de ingreso una sola vez, a pesar de que continúan interactuando con múltiples componentes de seguridad del sistema.

Un sistema Single Sign-on está compuesto por 5 módulos que son básicos para la implementación de la misma:

- **Mapping Subservice**: Se encarga de mapear entre dos conjuntos de credenciales diferentes.
- **Lookup Subservice**: Se encarga de hacer las consultas sobre las credenciales almacenadas en el SSO.
- **Administration Subservice**: Permite administrar el servicio de SSO y las credenciales allí almacenados.
- **Secret Subservice**: Es quien se encarga de controlar la encriptación de credenciales y generación de tickets de autenticación.
- **Credentials Store**: Es una base de datos que almacena las credenciales y los mapeos entre las mismos, con la información apropiadamente encriptada

Se puede afirmar que la arquitectura del SSO es una arquitectura esencialmente distribuida, compuesta por un Master Secret Server (único en la estructura del dominio de SSO), un conjunto de uno o más Single Sign-on Servers, y el Credential Store (una DB de SQL Server) como se muestra en la figura 7.

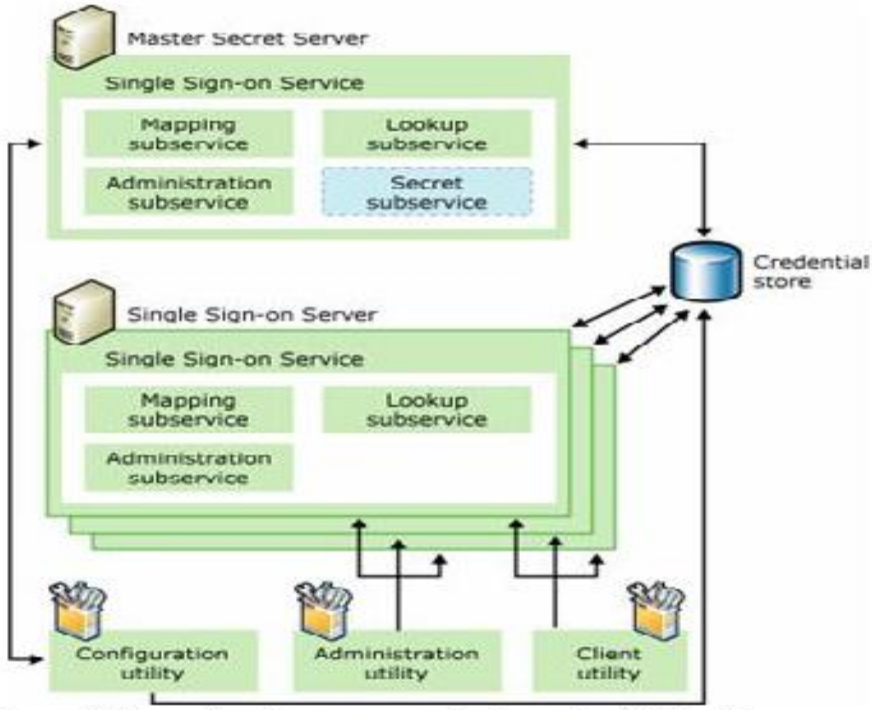


Figura 7 Arquitectura escencialmente distribuida

A continuación se muestra como funcionan los componentes vistos en la figura anterior:

El Credentials Store de SSO no solo es responsable de almacenar la configuración del SSO misma, sino que es responsable también de almacenar los diferentes usuarios y grupos asociados a cada aplicación o sistema contra el que se desea mapear credenciales. Estos mapeos pueden incluir tanto nombres de usuarios como de grupos, y adicionalmente contraseñas u otros tokens de autenticación (conocidos como “secretos”), es crítico que la información allí almacenada este completamente encriptada para que este segura.

El Master Secret Server (MSS) lo diferencia de los demás servidores en la infraestructura de SSO, ya que es precisamente la clave primaria con que se encriptan los contenidos en el Credentials Store, el MSS es único servidor autorizado en la infraestructura a encriptar los datos que se van almacenar allí.

Se puede decir además que los demás servidores en la infraestructura solo pueden acceder el Credentials Store para leer de allí las credenciales almacenadas, ya que solo pueden realizar la desencriptación de sus contenidos (solo conocen la clave de decodificación).

Ya se pueden visualizar varios elementos claves para que esta arquitectura sea realmente funcional. El punto quizá más importante es que la clave maestra debe mantenerse segura a toda costa, por varias razones:

Si llegara a perderse, se perderían todos los datos almacenados en el Credentials Store, ya que no sería posible desencriptarlos. Por esto es crítico que una vez establecida la clave maestra, se resguarde esta apropiadamente en un medio externo y se almacene dicho resguardo en un sitio físicamente seguro.

- Debe ser posible cambiar periódicamente la clave maestra de encriptación, lo cual puede ser necesario, por ejemplo, en caso que la clave actual fuera comprometida (ej.: se robaron el resguardo). Cambiar la clave maestra implica, en primera instancia, re-encriptar toda la información almacenada en el *Credentials Store*, lo cual es función de las herramientas administrativas de SSO y función únicamente realizable por el *Master Secret Server*.

La segunda de estas razones es clave para entender el segundo requerimiento de esta arquitectura: Si cambia la clave maestra, manejada por el *Master Secret Server*, este cambio debe notificarse a los demás servidores en la infraestructura de SSO; en caso contrario estos no podrían acceder la información de mapeos almacenada y la estructura colapsaría.

Para resolver este detalle, el Enterprise Single Sign-on tiene un mecanismo interno de replicación de esta información, que se hace en forma automática cada 30 segundos sobre protocolos de RPC seguros (canales encriptados).

Se puede mencionar como punto crítico de esta arquitectura es que aunque el *Master Secret Server* es crítico para la administración y encriptación de la información en el Credentials Store, no es crítico para la operación de la infraestructura de SSO en tiempo de ejecución. Es decir, que si por alguna razón el *Master Secret Server* saliera de línea, la infraestructura restante de SSO seguiría operando pues tendría acceso al *Credentials Store* (siempre y cuando esta base de datos de SQL Server no esté desplegada en la misma maquina que el MSS) y podría seguir accediendo y desencriptando los mapeos allí almacenados, que es la operación más común en una infraestructura de SSO.

2.2 Autenticación y Autorización

Se puede decir también que la autenticación y la autorización que brinda este sistema centralizado de gestión por credenciales juega un papel fundamental en el funcionamiento del Single Sing-On ya que

estos dos factores están muy interrelacionados entre si, pues cada aplicación o servicio esta configurado para que se conecte a una base de datos central y allí realizar las consultas para la autenticación de usuario y autorización.

La autenticación es el primer componente de la AAA (autenticación, autorización, contabilidad) para el modelo de control de acceso, ella es capaz de consolidar las identidades del sistema operativo y las aplicaciones dentro del directorio de control activo para disponer de una única identidad valida para fines de autenticación y login. Se puede mencionar que el uso de una identidad única facilita la integración de tecnologías avanzadas como tokens y tecnologías de login unificado que en este caso son la que se utilizarán en el trabajo.

Inicialmente el usuario antes de acceder a una aplicación o servicio del sistema se tiene que autenticar introduciendo sus credenciales , luego las credenciales son enviadas a un servidor de autenticación para verificar que el usuario esta autenticado en caso de ser así ,el sistema le brinda un token de seguridad que se va a mover en conjunto con sus credenciales hacia todos las aplicaciones y servicios del sistema, en caso de que los usuarios no estén registrados, estos son redirigidos al servidor de autenticación y regresan solo después de haber conseguido un acceso exitoso.

Ya una vez realizado el proceso de autenticación se debe preceder a la autorización ya que el usuario le ha demostrado al sistema la gestión de identidad que es una disciplina que abarca todas las tareas necesarias para crear y administrar en un entorno informático. Destacar que la autorización es el proceso de gestión de una identidad en el sistema, ya que este lo utiliza para determinar las actividades que el usuario puede realizar, primeramente el sistema debe de saber quien es usted para brindarle los permisos necesarios. La autorización es el segundo componente de la AAA para el modelo de control de acceso.

El sistema debe ser capaz de ya una vez que el usuario este autenticado devolverle el rol o los roles que se le han asignado al usuario en una aplicación o servicio del sistema, y así el usuario conoce los privilegios que se le han asignado, luego el sistema le permite el acceso con la identidad del usuario a las aplicaciones o servicios correspondientes estableciendo al usuario controles para la apertura de sesiones con privilegios de administrador, y habilitando la modalidad de acceso con el mínimo nivel de privilegios ,autorización para el acceso a la Web y segregación de tareas.

2.3 RUP como metodología de desarrollo para SSO

El Proceso Unificado de Rational (RUP, el original inglés Rational Unified Process) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, utiliza el paradigma de orientación orientada a objetos para su descripción lo que constituye una metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos". La documentación que se le hace a las diferentes aplicaciones es la base para futuras modificaciones y mejoras que las hagan más prácticas y manejables. Además de eso un marco de proceso configurable para satisfacer necesidades especificas e implementa las mejoras practicas de desarrollo de software.[4]

2.3.1 Objeto de automatización

Como se ha explicado anteriormente, la intranet corporativa de PDVSA en conjunto de las aplicaciones y servicios que brinda la infraestructura informática de la corporación, necesita de un sistema de autenticación único que garantice un acceso múltiple ha dichas aplicaciones y servicios. Para eso esta trabajando el Proyecto de la Intranet Corporativa de PDVSA, pero además, la utilización de un sistema de autenticación y autorización centralizado (Single Sing.-On) pude ser de gran utilidad para: primero para evitar que los usuarios tengan que autenticarse varias veces al acceder a una aplicación o servicio de la infraestructura, y segundo evitar problemas de seguridad.

Esencialmente, en la estructura del SSO existen varios elementos que deben funcionar de forma automática. Uno de ellos es el proceso de autenticar usuario en la que el sistema tiene que enviar las credenciales del usuario a un servidor de autenticación para verificar que el usuario este autenticado y además tiene que chequear si la aplicación tiene permiso para usar el sistema de autenticación. Otro de los elementos automatizar es validar token, en este caso una aplicación o servicio le solicita al sistema de validar un token de seguridad y el sistema entonces chequea que la aplicación tenga permisos para usar el sistema de autenticación y luego que chequea que el token exista y que la sesión sea valida. Otro proceso automatizar es listar las sesiones, donde se deben de mostrar todas las sesiones al administrador, cerrar sesión es otro proceso que se debe de automatizar ya que una aplicación le solicita al sistema cerrar una sesión enviándole el token de seguridad, otros procesos que se debe mencionar es el de gestionar roles y listar dichos roles, también se pretende automatizar devolver roles aplicación-usuario en el cual una aplicación solicita los roles que un usuario tiene en ella enviando el identificador del usuario y por ultimo se debe de automatizar gestionar aplicaciones.

2.3.2 Objetivos estratégicos del sistema

Realizar una aplicación de fácil acceso y con una interfaz cómoda y no intrusiva para el usuario del sistema.

Contribuir al desarrollo tecnológico de la Corporación de PDVSA.

Realizar un sistema de autenticación única que sea eficiente y responda a las necesidades de los usuarios de la corporación.

Fortalecer la seguridad en la infraestructura tecnológica.

2.3.3 Propuesta del sistema

El sistema de autenticación y control centralizado (SSO) debe funcionar como un mecanismo de autenticación única, que le brinde al usuario el acceso a las diferentes aplicaciones y servicios con que cuenta la infraestructura mediante un único acceso y además que el sistema sea capaz de brindar altos niveles de confiabilidad a los usuarios y que brinde un robusto sistema de seguridad.

2.3.3.1 Modelo de dominio

El sistema de autenticación y control centralizado se desarrolla en el ambiente de las aplicaciones y servicios con que cuenta la infraestructura de la corporación de PDVSA. EL sistema de autenticación y control centralizado es un sistema que permite al usuario autenticarse una sola vez, además proporciona un mecanismo de autenticación que permite el acceso a los usuarios a múltiples aplicaciones, así como también establecer un componente de seguridad y un alto nivel de confiabilidad a los usuarios, a las aplicaciones y a los servicios que se ejecutan en el sistema.

El sistema presenta una lista de las aplicaciones que pueden hacer uso de él, para que cuando una aplicación haga uso del sistema el automáticamente busque en la lista que tiene si la aplicación esta registrada para entonces dejarla usar los servicios que posee el sistema. Ya una ves que la aplicación hace uso del sistema, esta re-direcciona al usuario a un formulario de autenticación para poder hacer uso de la aplicación.

Este acceso se da por medio de la inserción de un login especificado y un password que a su vez debe validarse .Una vez registrado el usuario y después de haberse validado el registro y contraseña del usuario, el sistema le da un token de seguridad y es almacenado junto con las credenciales del

usuario. El sistema devuelve a la aplicación el token de seguridad junto con el rol o roles que se le fueron asignados al usuario para así ella saber hasta donde el usuario tiene los permisos necesarios para hacer uso de los servicios que brinda la aplicación.

En caso de que el usuario ya este autenticado y tenga la necesidad de hacer uso de otra aplicación o servicio, el sistema verifica si el usuario esta registrado, de ser así entonces el sistema comprueba que exista una sesión abierta y si es cierto, el sistema valida el token de seguridad que se le fue enviado a la aplicación comprobando que el token de la aplicación coincidan con los datos almacenados, luego de este procedimiento la aplicación necesita que el sistema le devuelva el rol o los roles del usuario para saber que permisos tiene, le pasa el usuario e identificador de la aplicación y el sistema le devuelve el rol o roles que se le fueron dados al usuario. Después de esto el sistema autoriza a la aplicación hacer uso de él como se muestra en la figura 8.

Representación del modelo de Dominio

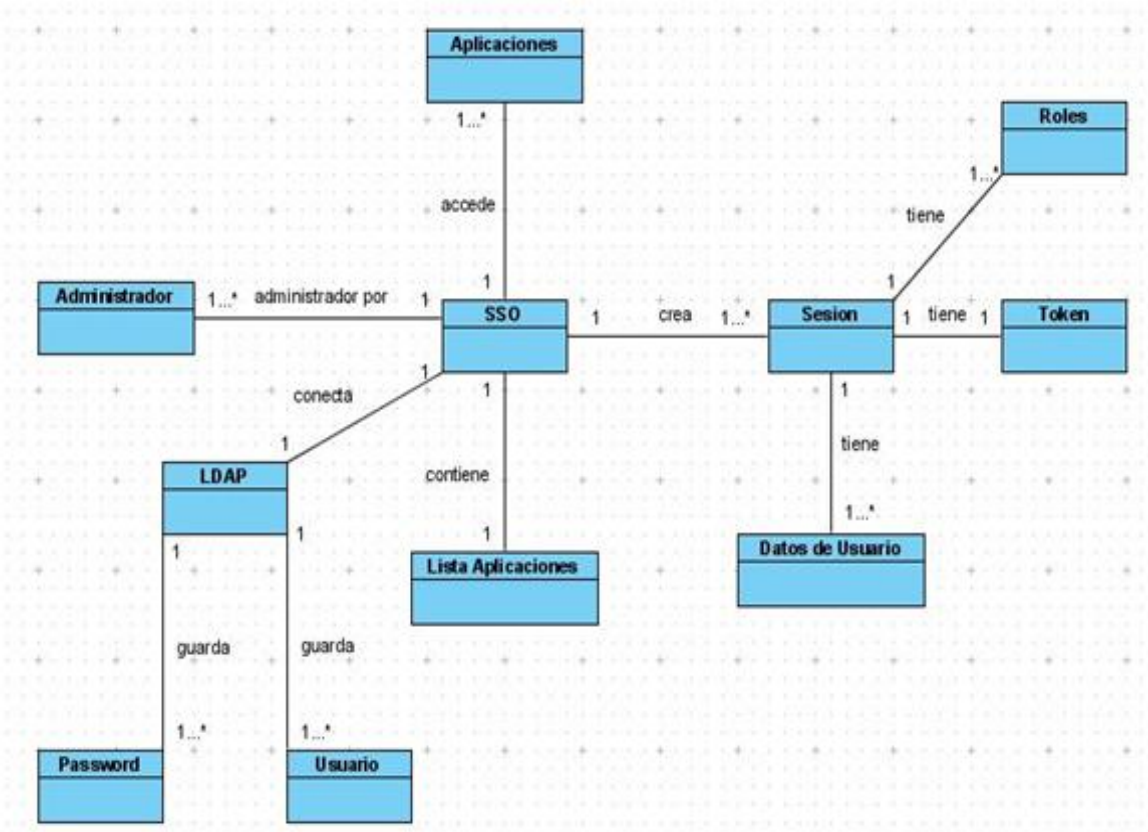


Figura 8.Modelo de Dominio

Entidades y Conceptos fundamentales

Concepto	Descripción
Aplicaciones	Forman parte de los procesos modulares del sistema informático con que cuenta la infraestructura de PDVSA que los usuarios utilizan accediendo mediante un servidor Web.
SSO	Sistema que permite un múltiple acceso mediante un login único.
Lista Aplicaciones	Lista con que cuenta el sistema, de las aplicaciones que pueden hacer uso de él.
Administrador	Es el encargado de administrar todas las funcionalidades con que cuenta el sistema.
LDAP	Implementa un servicio de directorio jerárquico y distribuido para acceder a repositorios de información referentes a usuarios y contraseñas y otras entidades en torno de red, ofreciendo una amplia capacidad de filtrado y mecanismos de seguridad.
Password	Conjunto de caracteres alfanuméricos que le permite al usuario el acceso a determinado recurso o a la utilización de un servicio dado.
Usuario	Persona que esta registrada en el sistema y que puede acceder al mismo.
Sesión	La conexión que se establece entre el usuario con el sistema, mediante el inicio de sesión utilizando los datos de acceso suministrados mediante el registro.
Roles	Es el nombre que se le confiere al conjunto de perfiles que le son asignados al usuario para la realización de sus funciones en la aplicación.
Token	Es un tipo de identificador que se les brinda a las aplicaciones para que el sistema les pueda dar los permisos.
Datos usuarios	Es todo lo que identifica al usuario.

Tabla 1. Entidades y conceptos fundamentales

2.3.3.2 Requerimientos de la aplicación

Al poder identificar los requisitos funcionales de la aplicación, se van identificando las funcionalidades y cualidades con las que debe contar la aplicación. De esta forma se llega a un mejor entendimiento entre los usuarios y el trabajo y el trabajo de los desarrolladores. Además se puede decir que una buena elección de los requerimientos hace posible que el mantenimiento futuro de la aplicación lleve un orden y un seguimiento detallado.

Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir .Las condiciones que debe cumplir el sistema son:

RF1: El sistema debe permitir que un usuario se autentique satisfactoriamente.

RF2: El sistema debe ser capaz de validar un token de seguridad.

RF3: El sistema debe listar las secciones activas al administrador.

RF4: El sistema debe permitir a una aplicación cerrar sesión.

RF5: El sistema debe brindar la gestión de roles.

RF5.1: Adicionar rol.

RF5.2: Listar rol.

RF5.3: Modificar rol

RF5.4: Eliminar rol.

RF6: El sistema debe ser capaz de verificar si una aplicación tiene permisos para hacer uso de él.

RF7: El sistema sea capaz de devolver los roles aplicación-usuario.

RF8: El sistema debe de facilitar la gestión de las aplicaciones.

RF8.1: Adicionar aplicación.

RF8.2: Listar aplicación.

RF8.3: Modificar aplicación.

RF8.4: Eliminar aplicación.

Requisitos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe de tener .Se debe pensar en estas propiedades como las características que hacen el producto mas atractivo, usable, rápido y confiable.

Las propiedades que debe cumplir el sistema son:

Usabilidad:

EL sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente Web en sentido general.

Rendimiento:

Tiempos de respuestas rápidos al igual que la velocidad de procesamiento de la información.

Soporte:

Se requiere de un servidor de base de datos que de soporte para grandes volúmenes de datos y velocidad de procesamiento, además que muestre un tiempo de respuesta rápido en accesos concurrentes.

Seguridad:

Identificar al usuario antes de realizar cualquier acción.

Crear diferentes cuentas de usuario y asignarle a cada uno los permisos pertinentes.

Mostrar a cada usuario solo las funcionalidades del sistema sobre las cuales tiene permiso de acceso.

Ofrecer mensajes de verificación antes de ejecutar acciones irreversibles (eliminaciones de datos).

El servidor donde se encuentre instalado el sistema debe de estar ubicado en un local protegido contra hurto y desastres naturales.

Portabilidad:

Necesidad de que el sistema sea multiplataformas.

Confiabilidad:

La herramienta de implementación utilizada debe de tener soporte para la recuperación ante fallos y errores.

Funcionalidad:

Reducir el mínimo de tiempo en que carga el sistema.

Guardar en cache páginas de contenido para agilizar la navegación.

Software:

Navegador superior o compatible con Internet Explorer 6, o Mozilla Firefox.

Definición de los actores

Actores	Justificación
Aplicación	Es la encargada de interactuar con el sistema, además de realizar determinadas funciones en él.
Administrador del Sistema	Es el encargado de gestionar el rol o los roles del usuario, así como también gestionar las aplicaciones a las que pueden acceder al sistema y listar las sesiones que están abiertas.

Tabla 2. Definición de los actores

Listado de Casos de Uso

CU-1	Autenticar Usuario
Actores	Aplicación.
Descripción	En este caso de uso la aplicación inicia la acción de autenticar al usuario en el sistema, luego se recepciona el identificador y la contraseña, se autentica el usuario y se devuelve un token de seguridad que identifique la sesión.
Referencia	RF1

Tabla 3.CU Autenticar usuario

CU-2	Validar token
Actores	Aplicación.
Descripción	La aplicación le solicita al sistema validar un token de seguridad, luego el sistema chequea que la aplicación tengo permisos para hacer uso de él, después el sistema elimina todas las sesiones que hayan espirado o que no sean validas, el sistema chequea que el token exista y devuelve una respuesta de si de que el login de usuario es correspondiente.
Referencia	RF2

Tabla 4.CU Válidar token

CU-3	Listar sesiones
Actores	Administrador del sistema
Descripción	El caso de usos se inicia cuando el administrador le solicite al sistema que le muestre todas las sesiones que se encuentren activas en ese momento.
Referencia	RF3

Tabla 5.CU Listar sesiones

CU-4	Cerrar sesión
Actores	Aplicación
Descripción	La aplicación le solicita al sistema cerrar una sesión
Referencia	RF4

Tabla 6.CU Cerrar sesión

CU-5	Gestionar roles
Actores	Administrador del sistema.

Descripción	Este caso de uso se inicia cuando el administrador necesita tanto adicionar, listar, modificar, así como también eliminar roles del usuario.
Referencia	RF5

Tabla 7.CU Gestionar roles

CU-6	Verificar aplicación.
Actores	Aplicación
Descripción	El caso de usos se inicia cuando una aplicación accede al sistema y este verifica que la aplicación ese registrada.
Referencia	RF6

Tabla 8. CU Verificar aplicación

CU-7	Solicitar roles
Actores	Aplicación
Descripción	El caso de uso se inicia cuando una aplicación le solicita al sistema el rol o roles que se le fueron asignados a un usuario en la aplicación.
Referencia	RF7

Tabla 9. CU Solicitar roles

CU-8	Gestionar aplicaciones
Actores	Administrador del sistema
Descripción	Permite al administrador adicionar, listar, modificar o eliminar una aplicación.
Referencia	RF8

Tabla 10. CU Gestionar aplicaciones

Diagrama de Casos de uso del sistema

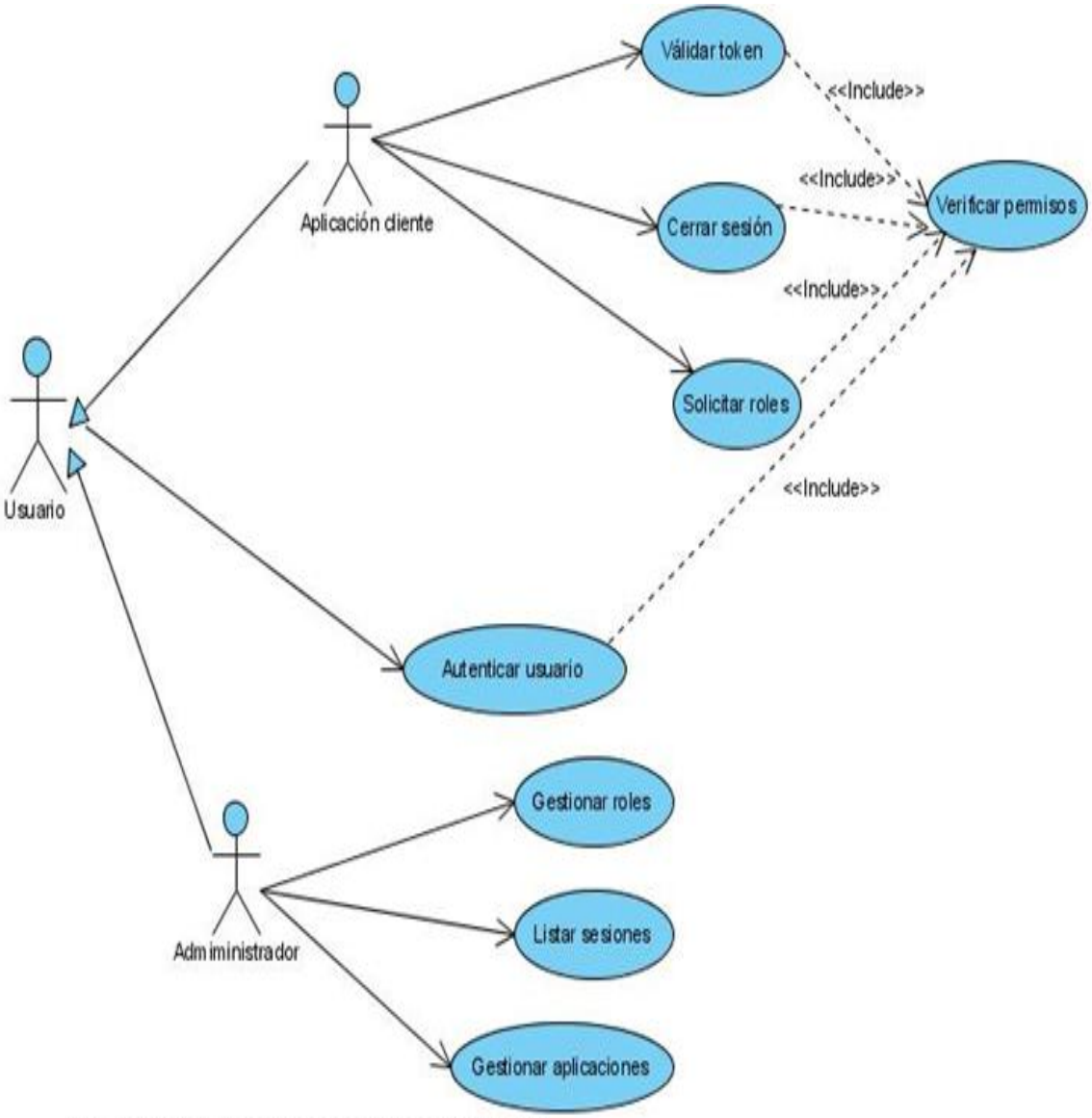


Figura 9.Diagrama de casos de uso del sistema.

Descripción de los casos de uso del sistema.

Caso Uso		
CU 1	Autenticar Usuario	
Propósito:	Comprobar que los datos del usuario son válidos y autenticarlo en el sistema.	
Actor:	Aplicación Cliente	
Resumen:	El caso de uso de inicia cuando una aplicación desea autenticar a un usuario. Se recepciona el identificador y la contraseña, se autentica al usuario y se devuelve un token de seguridad que identifique la sesión.	
Precondiciones:	Que la aplicación tenga permisos	
Referencias:	RF1	
CU asociados:	CU-6	
Prioridad	Critico	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Sistema
1. Una aplicación solicita autenticar a un usuario enviando los datos de autenticación.		1.1Se ejecuta el caso de uso incluido (Verificar permisos). 1.2 El sistema verifica que los datos introducidos sean correctos. 1.3 El sistema autentica el usuario. 1.4 El sistema crea token de seguridad. 1.5 El sistema devuelve token de seguridad.
Flujos Alternos		
Acción del Actor		Respuesta del Sistema
		1.1 El sistema devuelve un mensaje de error “La aplicación no tiene permiso para hacer uso del sistema”. 1.2 El sistema devuelve un mensaje de error “Los datos de autenticación son incorrectos”
Poscondiciones	El usuario está autenticado	

Tabla 11. CU Autenticar usuario

Caso Uso		
CU-2	Validar Token	
Propósito:	Comprobar que una aplicación tenga los permisos correspondientes para hacer uso del sistema.	
Actor:	Aplicación Cliente	
Resumen:	El caso de uso se inicia cuando una aplicación le solicita al sistema de autenticación validar un token de seguridad, el sistema chequea que la aplicación tenga permisos para usar el sistema de autenticación, el sistema elimina todas las sesiones que hayan expirado o no sean válidas, el sistema chequea que el token exista y devuelve una respuesta de si (login del usuario correspondiente a la sesión) o no.	
Precondiciones:	La aplicación tenga permisos para usar el sistema.	
Referencias:	RF2	
CU asociados:	CU-6	
Prioridad	Critico	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Sistema
1. Una aplicación le solicita al sistema validar un token de seguridad.		1.1Se ejecuta el caso de uso incluido (Verificar permisos). El sistema chequea que la aplicación tenga una sesión abierta. El sistema elimina todas las sesiones que hayan expirado o que no sean validas. El sistema chequea que el token de seguridad exista. El sistema devuelve una respuesta afirmativa.
Flujos Alternos		
Acción del Actor		Respuesta del Sistema
		El sistema muestra un mensaje de error ”La aplicación no tiene permisos para utilizar el sistema” El sistema muestra un mensaje de error en caso de que la sesión no este abierta. El sistema muestra un mensaje de error en caso de que el token de seguridad no exista.
Poscondiciones	El sistema valida el token de seguridad y brinda los permisos.	

Tabla 12. CU Válidar token

Caso Uso		
CU-3	Listar sesiones	
Propósito:	Que el sistema muestre una lista con las sesiones abiertas.	
Actor:	Administrador del Sistema.	
Resumen:	El caso de uso se inicia cuando el administrador le solicita al sistema que le muestre todas las sesiones que se encuentren activas .El caso de uso termina cuando el sistema le muestra al usuario una lista con todas las sesiones abiertas.	
Precondiciones:	EL administrador debe de estar autenticado.	
Referencias:	RF3	
CU asociados:		
Prioridad	Secundario	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Sistema
1. El administrador decide conocer las sesiones abiertas y presiona el botón "Listar sesiones".		El sistema le muestra al administrador una lista con todas las sesiones abiertas y finaliza el caso de uso.
Flujos Alternos		
Acción del Actor		Respuesta del Sistema
Poscondiciones	Obtener una lista con todas las sesiones abiertas.	

Tabla 13. CU Listar sesiones

Caso Uso		
CU-4	Cerrar sesión	
Propósito:	Que una aplicación le solicite al sistema cerrar la sesión.	
Actor:	Aplicación Cliente	
Resumen:	El caso de uso se inicia cuando una aplicación le solicita al sistema cerrar sesión enviándole el token de seguridad.	
Precondiciones:	La aplicación debe de tener permisos para usar el sistema.	
Referencias:	RF4	
CU asociados:	CU-6	
Prioridad	Secundario	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Sistema
1. La aplicación le solicita al sistema cerrar sesión enviándole el token de seguridad.		1.1 1Se ejecuta el caso de uso incluido (Verificar permisos). 1.2 El sistema comprueba que el token de seguridad sea valido. 1.3 El sistema cierra la sesión y finaliza el caso de uso.
Flujos Alternos		
Acción del Actor		Respuesta del Sistema
		1.1 El sistema muestre un mensaje de error “La aplicaron no tiene permisos para usar el sistema”. 1.2 El sistema muestra un mensaje de error en caso de que el token de seguridad no sea valido.
Poscondiciones	El sistema cierra la sesión.	

Tabla 14. CU Cerrar sesión

Caso Uso	
CU-5	Gestionar roles
Propósito:	Permitir al administrador adicionar, listar, modificar y eliminar rol.
Actor:	Administrador del Sistema
Resumen:	El caso de uso se inicia cuando el administrador necesita tanto

	adicionar, listar, modificar como eliminar los roles del usuario. Terminando el CU al lograr su objetivo.
Precondiciones:	Administrador, autenticado satisfactoriamente.
Referencias:	RF5
CU asociados:	
Prioridad	Secundario

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
El Administrador, puede necesitar: - Adicionar un rol (ir a Escenario Adicionar rol). -: Listar rol(ir a Escenario: Listar rol) -Modificar rol (ir a Escenario: Modificar rol) -Eliminar rol(ir a Escenario: Eliminar rol)	
Escenario: Adicionar rol	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El administrador decide crear un rol y presiona el botón “Crear”.	1.1 El sistema muestra un formulario de solicitud de los datos necesarios para crear un rol.
2. El administrador suministra al sistema los datos del rol a crear.	2.1 El sistema valida que los datos sean entrados correctamente. 2.2 El sistema verifica que el rol no exista. 2.3. El sistema guarda el rol creado correctamente en la BD. 2.4 El sistema le muestra un mensaje el administrador informándola que ya ha sido efectuado el registro del rol y finaliza el caso de uso.
Curso alternativo de los eventos	
Acción del Actor	Respuesta del Sistema
2. a Si el administrador no llena todos los campos obligatorios para crear el evento y presiona el botón: “Crear”.	2.1 Muestra una notificación en la página que indica que no se han completado todos los campos obligatorios. 2.2 El sistema muestra un mensaje de que el rol ya existe.
Escenario: Listar roles	
Acción del Actor	Respuesta del Sistema
1. El administrador ejecuta la acción Listar roles	1.1 Se ejecuta la funcionalidad listar roles, mostrándose una página con todas los roles que existen, mostrando la opción de escoger el rol a adicionar.

Escenario: Modificar rol	
Acción del Actor	Respuesta del Sistema
1. El administrador selecciona el rol que quiere modificar y presiona el botón “Editar”.	1.1 Se ejecuta la funcionalidad Modificar rol, mostrándose una página con los datos de los roles, habilitados para ser modificados.
2. El administrador modifica los datos que crea necesario, al terminar presiona el botón: “Modificar”.	2.1 El sistema verifica que los campos obligatorios estén llenos. 2.2 Guarda los nuevos datos en la BD y termina el CU.
Curso alternativo de los eventos	
Acción del Actor	Respuesta del Sistema
2. Si el administrador modifica los datos de forma incorrecta y presiona el botón: “Modificar”.	2.1 Muestra una notificación en la que indica donde está el error.
Escenario: Eliminar rol	
Acción del Actor	Respuesta del Sistema
1. El administrador escoge el rol que desee eliminar y presiona el botón: “Eliminar”.	1.1 Se ejecuta la funcionalidad Eliminar, mostrándose previamente un cuadro de diálogo en el que se pregunta si el usuario está seguro de querer ejecutar esta acción.
2. El administrador está seguro de haber escogido el rol correcto a eliminar y presiona el botón: “Aceptar”.	2.1 Elimina el rol, finalizando el CU.
Curso alternativo de los eventos	
Acción del Actor	Respuesta del Sistema
2. a Si el administrador no está seguro de haber escogido el rol correcto, presiona el botón: “Cancelar”.	1.1 No se lleva a cabo ninguna acción.
Poscondiciones	El sistema gestiona los roles.

Tabla 15.CU Gestionar Roles

Caso Uso		
CU-6	Verificar permisos.	
Propósito:	Que la aplicación tenga permisos para hacer uso del sistema.	
Actor:	Aplicación Cliente	
Resumen:	El caso de uso se inicia cuando una aplicación intenta acceder al sistema, este busca en la lista de aplicaciones si la aplicación está registrada y si lo esta el sistema le posibilita hacer uso de él.	
Precondiciones:	La aplicación tiene que estar registrada en el sistema.	
Referencias:	RF6	
CU asociados:		
Prioridad	Critico.	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Sistema
1. La aplicación intenta acceder al sistema.		1.1 El sistema chequea en la lista de aplicaciones que la aplicación esté registrada. 1.2 El sistema le da los permisos para hacer uso de él.
Flujos Alternos		
Acción del Actor		Respuesta del Sistema
		1.1 El sistema envía un mensaje de error “La aplicación no tiene permisos para usar el sistema”.
Poscondiciones	El sistema le brinda los permisos	

Tabla 16. CU Verificar acplicación

Caso Uso		
CU-7	Solicitar roles	
Propósito:	Devuelve los roles que tiene el usuario en una aplicación determinada.	
Actor:	Aplicación Cliente	
Resumen:	El caso de uso se inicia cuando una aplicación solicita los roles que un usuario determinado tiene en ella. Se le devuelven dichos roles.	
Precondiciones:	La aplicación debe tener los permisos para usar el sistema.	
Referencias:	RF7	
CU asociados:	CU-6	
Prioridad	Secundario	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Sistema
1. Una aplicación solicita los roles de un usuario.		1.1Se ejecuta el caso de uso incluido (Verificar permisos). 1.2 El sistema chequea qué roles tiene el usuario que intenta acceder a una aplicación. 1.2 El sistema devuelve los roles del usuario para esa aplicación.
Flujos Alternos		
Acción del Actor		Respuesta del Sistema
		1.1 El sistema devuelve un mensaje de error “La aplicación no tiene permiso para hacer uso del sistema”.
Poscondiciones	El sistema le devuelve los roles a la aplicación.	

Tabla 17. CU Solicitar roles

Caso Uso	
CU-8	Gestionar aplicaciones
Propósito:	Permitir al administrador adicionar, listar, modificar y eliminar una aplicación.
Actor:	Administrador del Sistema.
Resumen:	El caso de uso se inicia cuando el administrador necesita tanto adicionar, modificar y eliminar las aplicaciones que puedan hacer uso del sistema. Terminando el CU al lograr su objetivo.
Precondiciones:	EL administrador tiene que estar autenticado.
Referencias:	RF8
CU asociados:	
Prioridad	Secundario

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
El Administrador, puede necesitar: - Adicionar una aplicación (ir a Escenario Adicionar aplicación). -Listar aplicación (ir a Escenario Listar aplicación). -Modificar aplicación (ir a Escenario Modificar aplicación). -Eliminar aplicación (ir a Escenario Eliminar aplicación).	

Escenario: Adicionar aplicación	
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El administrador decide adicionar una aplicación presiona el botón “Adicionar”.	1.1 El sistema muestra un formulario de solicitud de los datos necesarios para adicionar una aplicación..
2. El administrador suministra al sistema los datos de la aplicación que se va adicionar.	2.1 El sistema valida que los datos sean entrados correctamente. 2.2 El sistema verifica que la aplicación no exista. 2.3. El sistema guarda la aplicación

	añadida correctamente en la BD.. 2.4 El sistema le muestra un mensaje el administrador informándola que ya ha sido efectuado el registro de la aplicación y finaliza el caso de uso.
Curso alternativo de los eventos	
Acción del Actor	Respuesta del Sistema
2. a Si el administrador no llena todos los campos obligatorios para adicionar la aplicación y presiona el botón: "Adicionar".	2.1 Muestra una notificación en la página que indica que no se han completado todos los campos obligatorios. 2.2 El sistema muestra un mensaje de que la aplicación ya existe.
Escenario: Listar aplicaciones.	
Acción del Actor	Respuesta del Sistema
1. El administrador ejecuta la acción Listar aplicaciones.	1.1 Se ejecuta la funcionalidad listar aplicaciones, mostrándose una página con todas las aplicaciones que existen.
Escenario: Modificar aplicación.	
Acción del Actor	Respuesta del Sistema
1. El administrador selecciona la aplicación que quiere modificar y presiona el botón "Editar".	1.1 Se ejecuta la funcionalidad Modificar aplicación, mostrándose una página con los datos de las aplicaciones, habilitados para ser modificados.
2. El administrador modifica los datos que crea necesario, al terminar presiona el botón: "Modificar".	2.1 El sistema verifica que los campos obligatorios estén llenos. 2.2 Guarda los nuevos datos en la BD y termina el CU.
Curso alternativo de los eventos	
Acción del Actor	Respuesta del Sistema
2. Si el administrador modifica los datos de forma incorrecta y presiona el botón: "Modificar".	2.1 Muestra una notificación en la que indica donde está el error.
Escenario: Eliminar aplicación	
Acción del Actor	Respuesta del Sistema
1. El administrador escoge la aplicación que desee eliminar y presiona el botón: "Eliminar".	1.1 Se ejecuta la funcionalidad Eliminar, mostrándose previamente un cuadro de diálogo en el que se pregunta si el usuario está seguro de querer ejecutar esta acción.
2. El administrador está seguro de haber escogido la aplicación correcta a eliminar y presiona el botón: "Aceptar".	2.1 Elimina la aplicación, finalizando el CU.
Curso alternativo de los eventos	
Acción del Actor	Respuesta del Sistema

2. a Si el administrador no está seguro de haber escogido la aplicación correcta, presiona el botón: “Cancelar”.	1.1 No se lleva a cabo ninguna acción.
Poscondiciones	El sistema proporciona la gestión de roles

Tabla 18. CU Gestionar aplicaciones

En este capitulo se han visto las características del sistema en general en conjunto con el proceso incremental e iterativo de RUP relacionado con el modelo de dominio y las características que son propias del sistema de autenticación y control centralizado que se va a desarrollar.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

En este capitulo se vera que los flujos de trabajo análisis y diseño son los que se encargan de cumplir con el objetivo principal de esta disciplina que no es mas que transformar los requerimientos a una especificación que describa de cómo se va a implementar el sistema. El análisis fundamentalmente consiste en obtener una visión que se preocupa de ver que hace el sistema de software a desarrollar y se interesa por los requisitos funcionales. Por otro lado el diseño esta basado en refinamiento que toma en cuenta los requisitos no funcionales, por lo cual se centra en como el sistema cumple sus objetivos.

3.1 Análisis

Se puede decir que las clases que se identifican en el flujo de trabajo de análisis y diseño, especialmente en el análisis, están asociadas con el contexto del dominio del problema .Por esta razón se representan conceptos y relaciones. En el flujo de trabajo de análisis se logra una abstracción del sistema, lo que esto constituye la entrada fundamental a la hora de modelar la aplicación.

Clases interfaz

Estas clases se encuentran al modelar las interacciones entre el sistema y su(s) o autor(es) y se pueden identificar a partir de:

Al menos una clase que modele la interacción entre el actor y el sistema, es decir una clase para cada interacción actor-caso de uso sin preocuparse de que la solución puede que presente más de una pantalla dentro de un caso de uso para un mismo usuario.

Una clase para cada caso sistema externo que será el responsable de la relación del sistema con cada uno de ellos.

Una clase par cada actor que represente un dispositivo sobre el cual actúa o recibe información.

Clase control

Las clases de control son las encargadas de coordinar el trabajo de los casos de uso, coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso, por lo que se puede decir que las clases de control definen el flujo de control y las transacciones dentro de un caso de uso delegando el trabajo a otros objetos.

Diagrama de clases de análisis

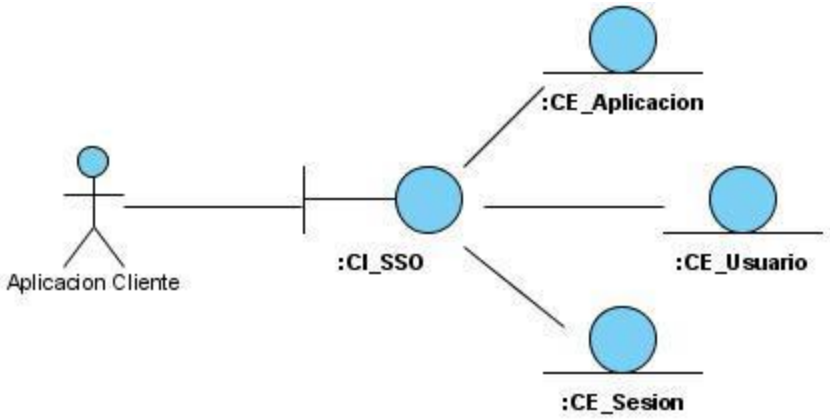


Figura 10.Diagrama de clases de análisis:Autenticar usuario

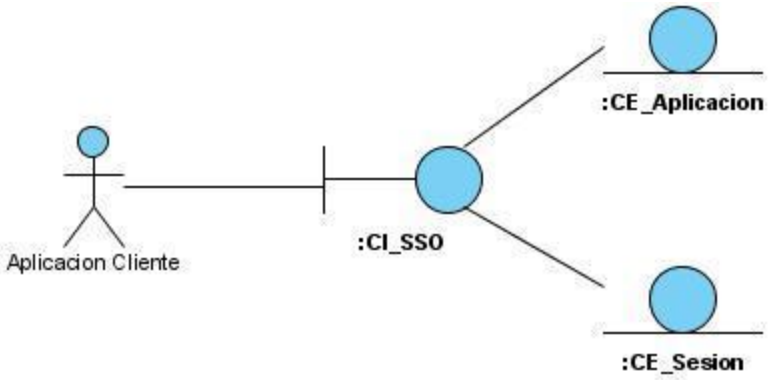


Figura 11.Diagrama de clases de análisis:Válidar token

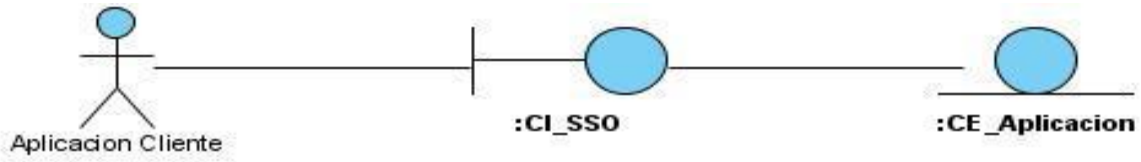


Figura 12.Diagrama de clases de análisis:Verficar permisos

Diagramas de colaboración

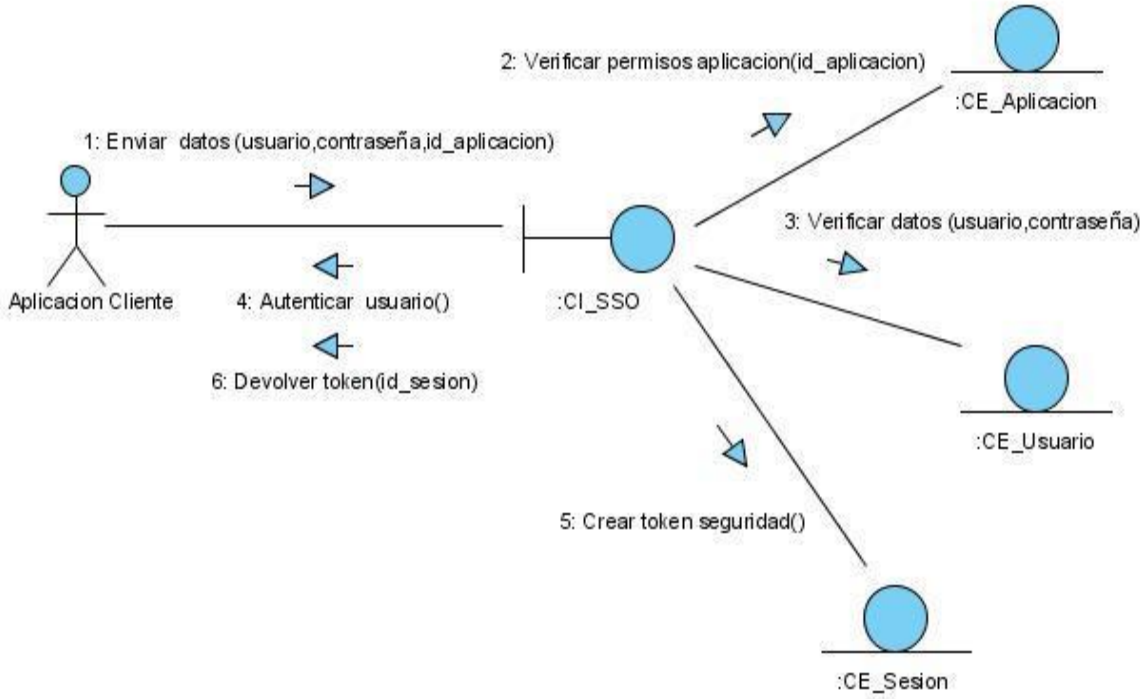


Figura 13.Diagrama de colaboración:Autenticar usuario

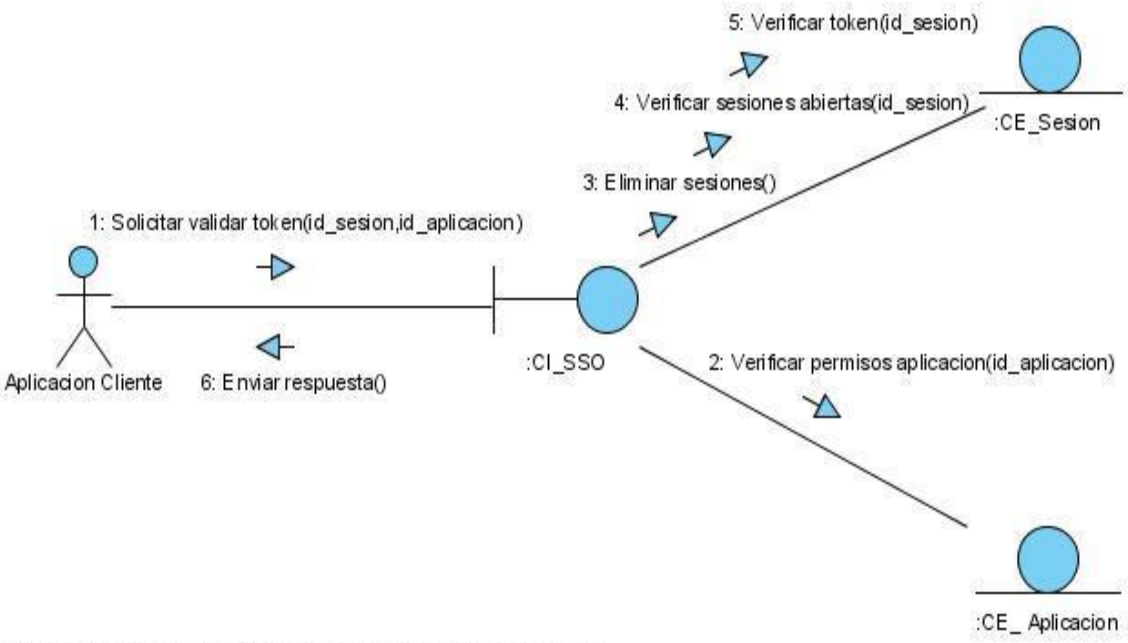


Figura 14.Diagrama de colaboración:Verficar token

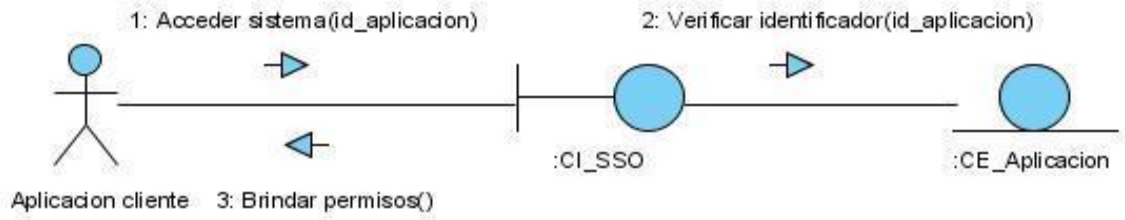


Figura 15.Diagrama de colaboración:Verficar permisos

Flujo de análisis de los diagramas de colaboración

Los diagramas de colaboración y todos en general pueden ser de difícil interpretación por si mismos. En estos casos pude ser útil un documento adicional que los explique .Este texto debe obviar los atributos, responsabilidades de los objetos que se manejen en él: Este documento se conoce como:

- Flujo de sucesos-análisis para el diagrama de colaboración: Autenticar usuario

Para lograr autenticar a un usuario, la Aplicación Cliente envía datos (1) del usuario. La CI_SSO que a la misma ves se comporta como interfaz controladora, realiza un conjunto de operaciones como: Verificar si la aplicación tiene los permisos (2) para utilizar mi sistema, para esto busca en la base de datos (tabla de aplicación) si la aplicación existe.Luego la CI_SSO verifica los datos del usuario (3) ,de manera que los comprueba con una LDAP(dominio).Una ves que los datos de usuario son correctos, la CI_SSO autentica al usuario(4) y continuamente crea un token de seguridad(5) de la sesión en la base de datos(tabla de sesión).Una ves creado en token de seguridad de la sesión, la CI_SSO se lo devuelve (6) a la Aplicación Cliente.

- Flujo de sucesos-análisis para el diagrama de colaboración: Validar token

La Aplicación Cliente le solicita a la CI_SSO validar token de seguridad (1). La CI_SSO que a la misma vez se comporta como interfaz controladora, realiza un conjunto de operaciones como: Verificar si la aplicación tiene los permisos (2) para utilizar mi sistema, para esto busca en la base de datos (tabla de aplicación) si la aplicación existe. Luego la CI_SSO verifica que haya una sesión abierta (3) y elimina las sesiones (4) que hayan expirado o que no sean validas de la base de datos (tabla sesión).Una vez eliminadas todas las sesiones fallidas, la CI_SSO verifica el token (5), para esto busca en la base datos (tabla de sesión) .La CI_SSO envía respuesta (6) a la Aplicación Cliente.

- Flujo de sucesos-análisis para el diagrama de colaboración: Verificar permisos

La Aplicación Cliente le solicita a la CI_SSO acceder al sistema (1). La CI_SSO que a la misma vez se comporta como interfaz controladora, realiza un conjunto de operaciones como: Verificar el identificador de la aplicación (2), se dirige a la base de datos y comprueba si el identificador de la aplicación esta registrado en el sistema, si esta le brinda los permisos (3) para hacer uso de el.

Requisitos especiales

Los requisitos especiales son descripciones textuales que se recogen en los requisitos no funcionales. Y uno de los mas importantes que se debe cumplir es ,cuando una aplicación cualquiera intenta acceder al sistema para hacer uso de las funcionalidades que brinda, el sistema debe verificar inmediatamente que la aplicación que este accediendo este registrada , las respuestas que debe mostrar el sistema deben de ser rápidas y no deben de demorar más de cinco segundos.

3.2 Diseño

El diseño es el encargado de modelar como se implementara el sistema, o sea, la definición de un plano para construir la aplicación y encontrar su forma incluida (incluida la arquitectura) para que soporte todos los requisitos incluyendo los requisitos no funcionales. Se puede decir que el diseño adquiere una comprensión de gran profundidad de los aspectos relacionados con los requisitos funcionales y restricciones relacionadas con los lenguajes de programación ,componentes reutilizables, sistemas operativos , tecnologías de distribución y concurrencia ,tecnologías de interfaz de usuario y tecnologías de gestión de transacciones, además el diseño facilita la descomposición de trabajos de implementación en partes, mas manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo ,teniendo en cuenta la posible concurrencia.

En la aplicación que se desarrolla la mayor parte del código de implementación es en el lenguaje PHP orientado a objetos. PHP funciona tanto en sistemas Unix como Linux con un servidor Web Apache de forma que el código generado por cualquiera de éstas plataformas no debe ser modificado al pasar a la otra , por lo que se puede decir que es un lenguaje multiplataforma. La principal ventaja de este conjunto es que, se puede compilar el interprete PHP como un módulo de Apache lo que provoca que la velocidad de ejecución de una página PHP sea elevada y que el consumo de recursos sea bajo ya que el interprete PHP se encarga una sola vez en memoria. En este caso se programa del lado del servidor que se ejecuta en el servidor Web junto antes de que se envié la página del cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a base de datos, conexiones en red y otras páginas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la página PHP. Dado que la página resultante contiene únicamente código HTML, y es compatible con otros navegadores.

Diagramas de clases del Diseño

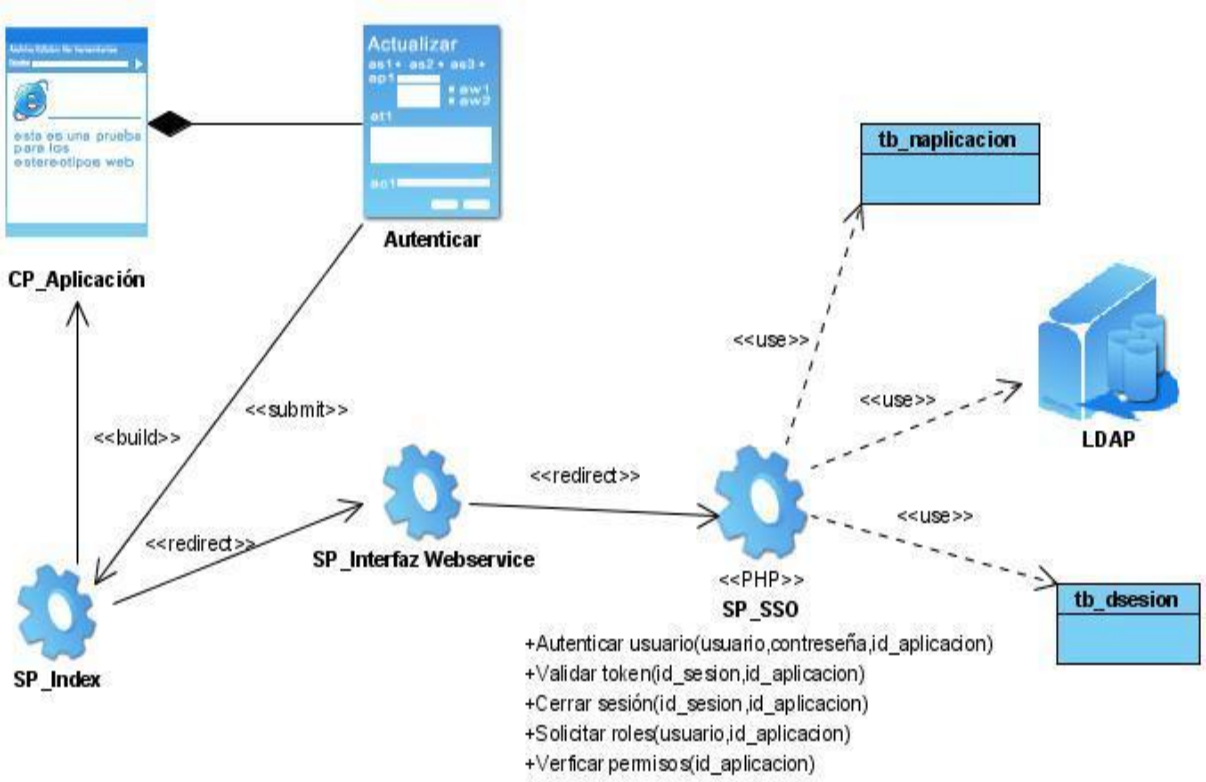


Figura 16.Diagrama de clases del diseñ:Autenticar usuario

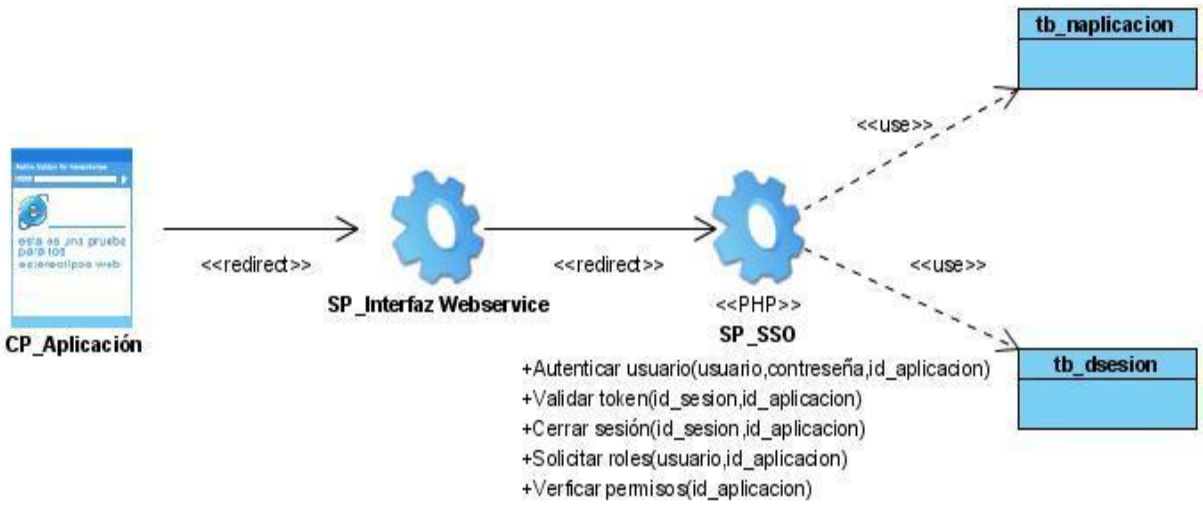


Figura 17.Diagrama de clases del diseño:Válidar token

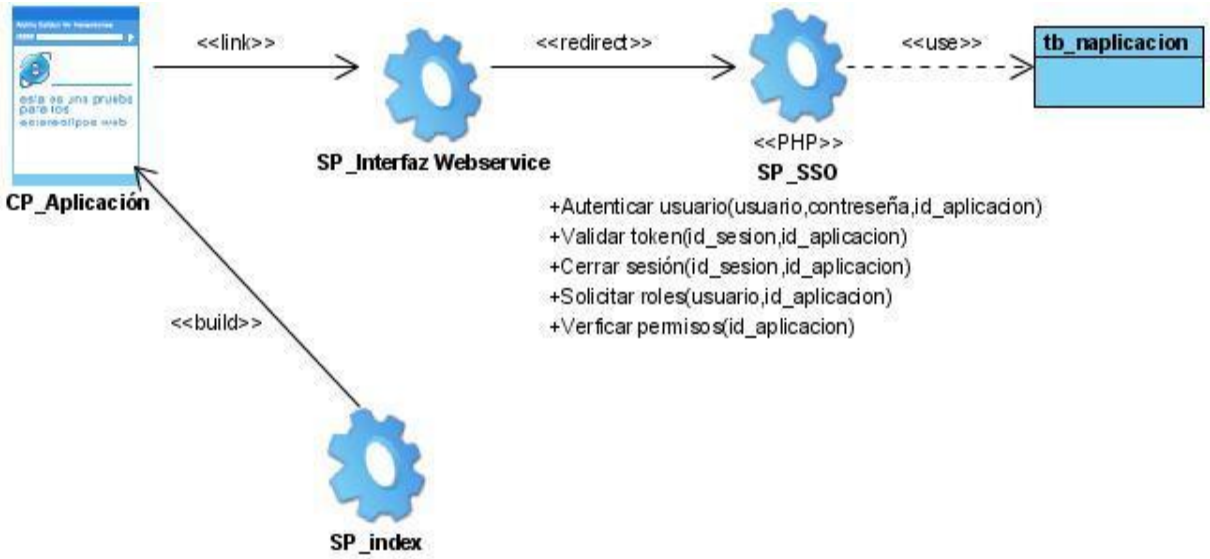


Figura 18.Diagrama de clases del diseño:verficar permisos

Diagramas de secuencia del diseño.

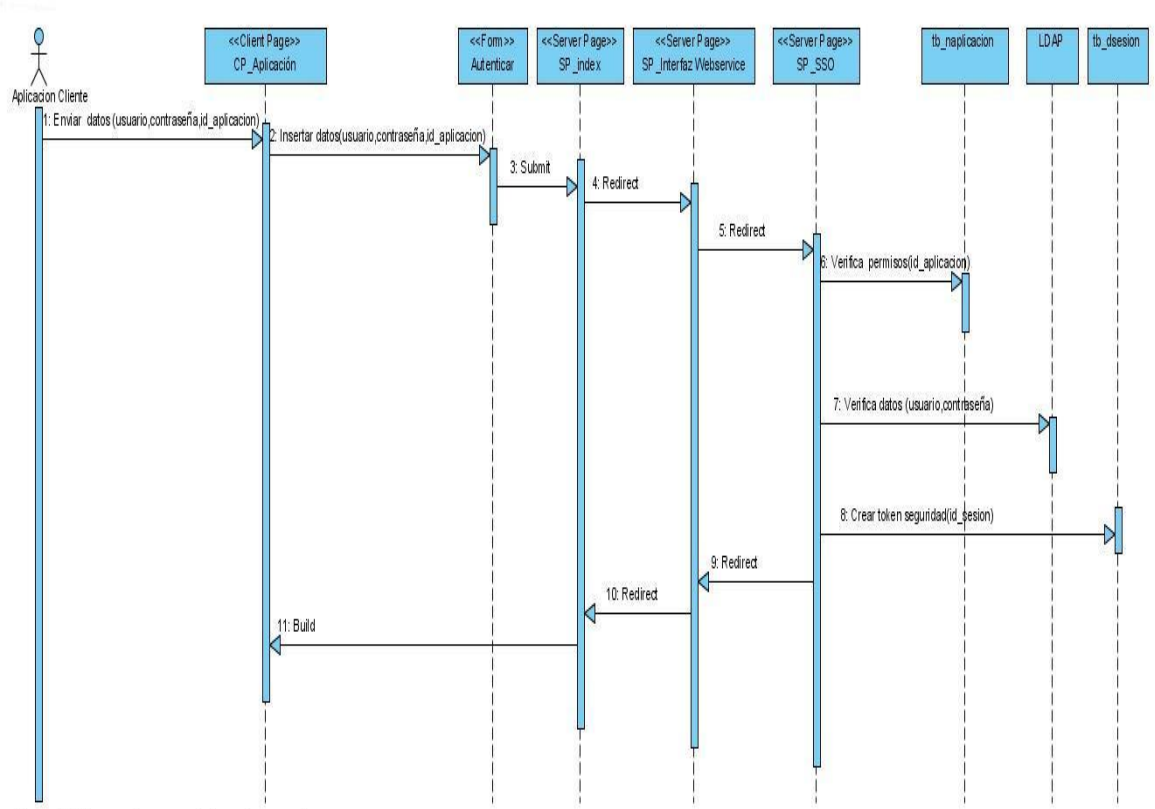


Figura 19.Diagrama de secuencia:Autenticar usuario

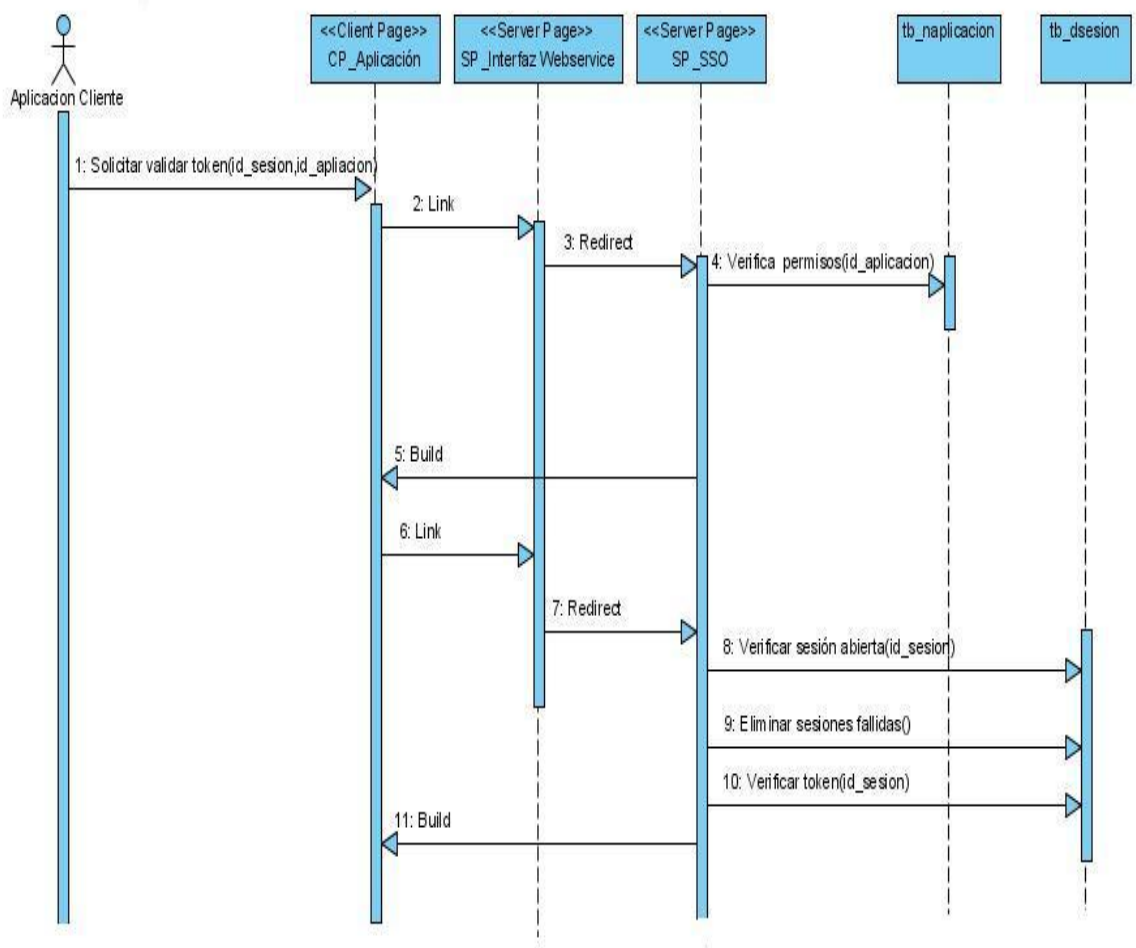


Figura 20.Diagrama de secuencia:Válidar token

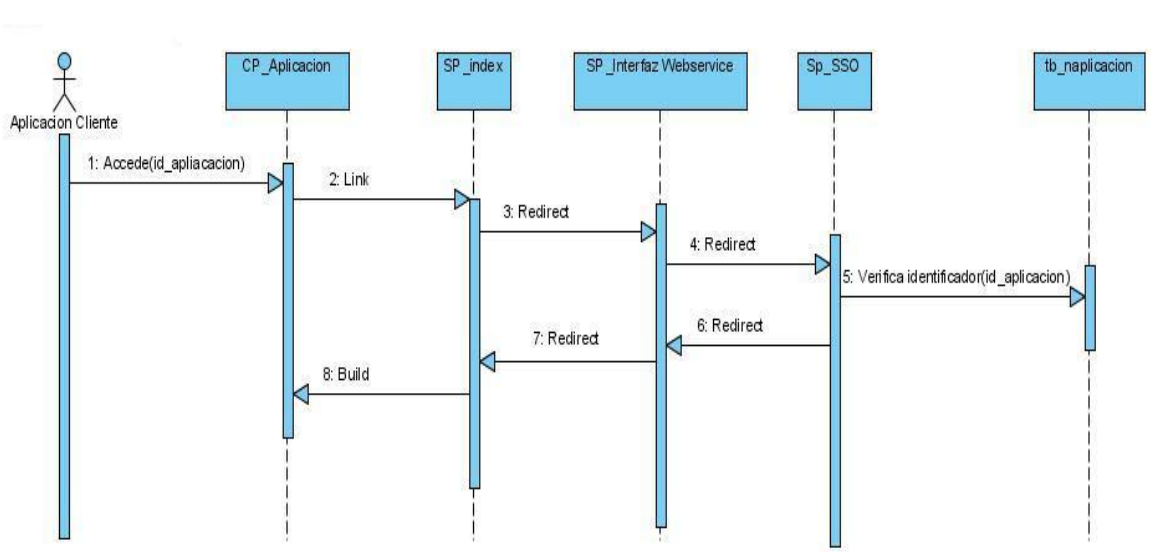


Figura 21.Diagrama de secuencia:Verficar permisos.

Descripciones de las clases del diseño.

Nombre: SSO	
Tipo de clase : Clase Controladora	
Atributo	Tipo
Métodos	
Nombre:	_Construct()
Descripción:	Construye la clase.
Nombre:	Verificar permisos($id _ aplicación)
Descripción:	Verifica que la aplicación tenga permisos para usar el sistema.
Nombre:	Autenticar usuario($usuario, contraseña)
Descripción:	Autenticar a los usuarios en el sistema.
Nombre:	Validar token(id _ sesión)
Descripción:	Verifica que el identificador de la sesión que esta accediendo al sistema coincida con el que esta guardado en la base de datos, luego válida ese token de seguridad y brinda los permisos a la aplicación.
Nombre:	Cerrar sesión(id _ sesión)
Descripción:	Cierra las sesiones que estén activas conociendo el identificador de la sesión.
Nombre:	Solicitar roles(id _ usuario)
Descripción:	Devuelve a una aplicación el rol o los roles que tiene un usuario en ella.

Tabla 18.Descripción de clases del diseño:SSO

Nombre: Gestionar roles	
Tipo de clase : Clase Controladora	
Atributo	Tipo
Métodos	

Nombre:	_Construct()
Descripción:	Construye la clase.
Nombre:	Getrol()
Descripción:	Devuelve los datos de los roles
Nombre:	Adicionar rol($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de los roles con los valores que recibe como parámetro.
Nombre:	Listar roles($array)
Descripción:	Lista todos los roles que existen.
Nombre:	Modificar rol ($array)
Descripción:	Modifica los datos de un rol sustituyéndolos por los datos que recibe como parámetro de entrada.
Nombre:	Eliminar rol(($id_rol)
Descripción:	Elimina un rol.

Tabla 19. Descripción de las clases del diseño:Gestionar roles

Nombre: Listar_sesiones	
Tipo de clase : Clase Controladora	
Atributo	Tipo
Métodos	
Nombre:	_Construct()
Descripción:	Construye la clase.
Nombre:	Listar sesiones($id _ sesión)
Descripción:	Lista todas las sesiones que se encuentren abiertas.

Tabla 20. Descripción de las clases del diseño:Listar sesiones

Nombre: Gestionar_aplicaciones	
Tipo de clase : Clase Controladora	
Atributo	Tipo
Métodos	
Nombre:	_Construct()
Descripción:	Construye la clase.
Nombre:	Getaplicación()
Descripción:	Devuelve los datos de los roles
Nombre:	Adicionar aplicación($array)
Descripción:	Crea una nueva instancia en la tabla de la base de datos que guarda los datos de las aplicaciones con los valores que recibe como parámetro de entrada.
Nombre:	Listar roles($array)
Descripción:	Lista todas las aplicaciones que existen.
Nombre:	Modificar aplicación($array)
Descripción:	Modifica los datos de una aplicación sustituyéndolos por los datos que recibe como parámetro de entrada.
Nombre:	Eliminar aplicación(id _ aplicación)

Descripción:	Elimina una aplicación.

Tabla 21.Descripción de las clases del diseño:Gestionar aplicaciones

Diagrama de Base de datos

Para el diseño de la base de datos del sistema se utilizó el diagrama de clases persistentes y el modelo de datos, representando las clases que simbolizan los datos que se obtienen y almacenan durante los procesos de la aplicación, siendo estos los que pueden modelarse a través de un diagrama de clases persistentes, lo que permitirá ver la relación entre los datos, y completará el modelado de la lógica del negocio.

Diagrama de Clases Persistentes

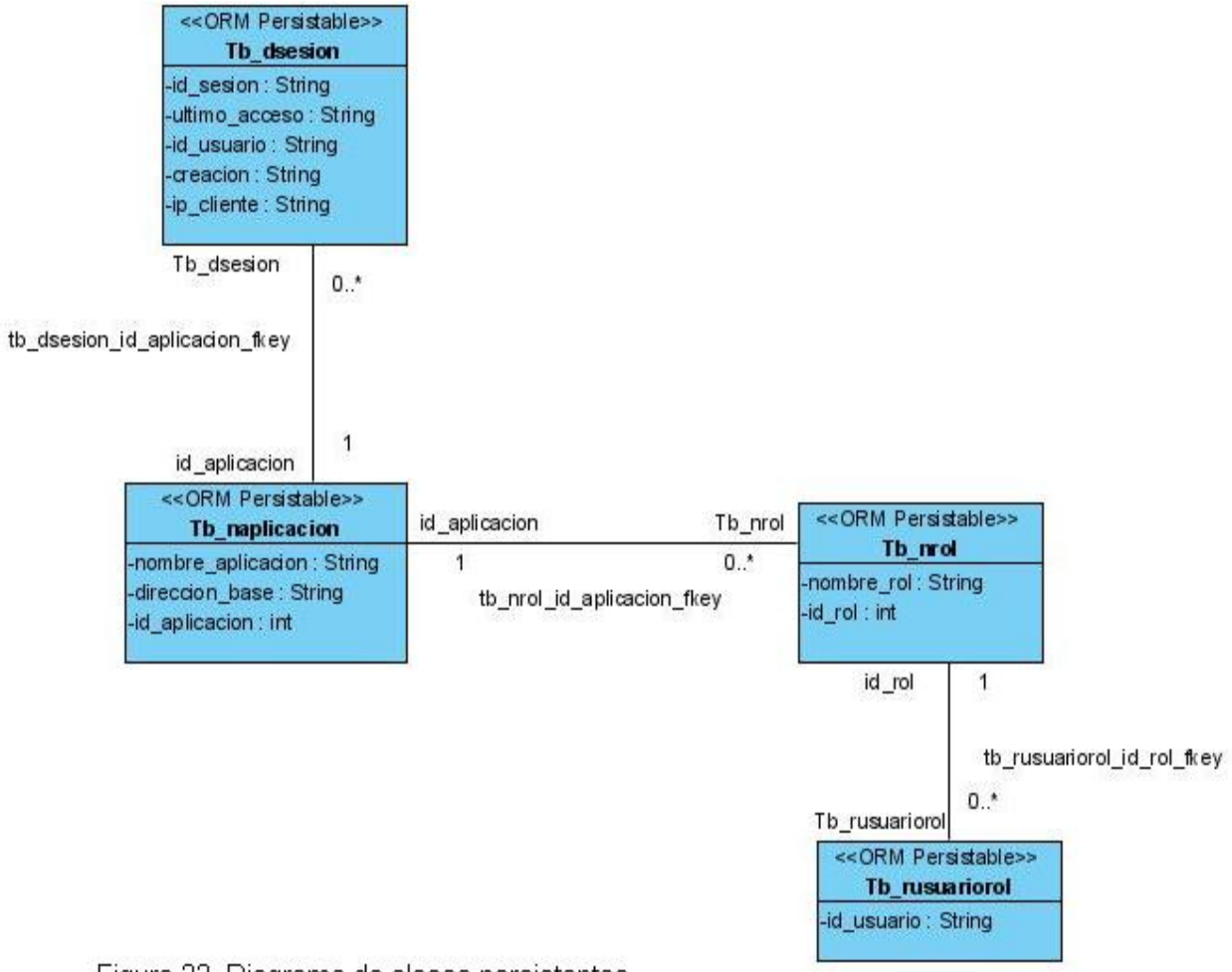


Figura 22. Diagrama de clases persistentes

Modelo de Datos

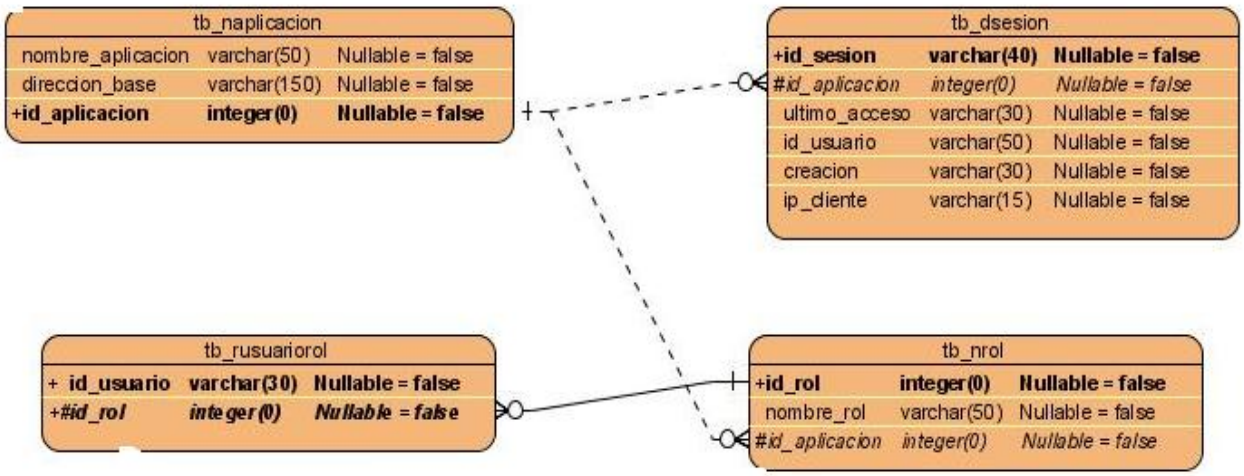


Figura 23.Diagrama de modelo de datos

Definiciones del diseño

EL diseño del sistema de autenticación y control centralizado para la infraestructura de PDVSA esta basado en un estilo arquitectónico que tiene como objetivo principal la organización estructural en términos y describir los componentes y relaciones que existen entre ellos con las restricciones de la aplicación.

Se puede decir que para el desarrollo del producto se utilizo para facilitar la aplicación, el patrón arquitectónico: Arquitectura en capaz que posibilita organizar el modelo de diseño en capaz y simplifica la comprensión y la organización del desarrollo del sistema, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores, dentro de este patrón arquitectónico se hace uso del estilo, Arquitectura en tres niveles que divide la aplicación en tres capaz lógicas cada una de ellas bien definidas.

La primera capa se denomina, capa de presentación que se va a componer de una interfaz interna para la aplicación cliente que es la que va posibilitar hacer los pedidos al sistema y así poder hacer uso de las funcionalidades que brinda, esta interfaz interna es parte de la arquitectura de la infraestructura PDVSA que en el algún momento cambie a SOAP.

La segunda capa se denomina lógica de aplicación, es básicamente el código al que recurre la capa de presentación para recuperar los datos deseados y esta está compuesta por las clases gestoras que contienen los métodos para realizar las acciones que desee la aplicación cliente sobre los datos.

La tercera capa se denomina, la capa de datos, esta contiene los datos necesarios de la aplicación, estos datos consisten en cualquier fuente de información incluyendo una base de datos.

El diseño propuesto cumple con el patrón de alta cohesión, pues las clases poseen responsabilidades estrechamente relacionada y colaboran entres ellas en un área determinada para llevar a cabo las tareas, por ejemplo la clase controladora “Gestionar Rol” es la que coordina y encapsula todo el comportamiento solamente del caso de uso Gestionar Rol. Otro de los patrones con que cumple el diseño es el patrón de bajo acoplamiento, pues el funcionamiento de una clase, no depende del funcionamiento de muchas otras clases.

Con el modelo de clases del diseño realizado y con las descripciones correspondientes a las clases del diseño, es muy fácil llegar a la etapa de codificación de la aplicación: la implementación .Este flujo de trabajo abarca todo lo que corresponde a la escritura del código fuente de la aplicación que ayuda a la ves a complementar prácticamente el trabajo teórico argumentado en los anteriores flujos de trabajo.

Tratamiento de errores

Mediante el tratamiento de errores en las aplicaciones se consigue validar las posibles respuestas que pueden darse en debido momento por el sistema. Además, evita la revelación datos sensibles de este o de la ocurrencia de errores en tiempo de ejecución que afecten la imagen real del sistema.

En caso particular del producto SSO si es necesario controlar los errores que se produzcan y unos de los fundamentales a tratar es la conexión a la base de datos, que sea de modo seguro ,que no halla ningún fallo a la hora de hacer las consultas ni de verificar algún dato. Otro de los errores que hay que tener bien en cuenta es a la hora de verificar el usuario y contraseña en el dominio de la infraestructura, que las verificaciones sean seguras y bien realizadas.

Interfaz

Se puede decir que la interfaz del sistema es interna y que está regida por las pautas de diseño aprobadas por la gerencia de asuntos públicos corporativos de PDVSA y fueron creados por la

dirección de diseño de la universidad de las ciencias informáticas y parte del proyecto de desarrollo de la Intranet corporativa de PDVSA y también se hizo uso de ajax con el objetivo de disminuir la sobrecarga de peticiones al servidor, y además de eso hace posible que las interacciones entre el cliente y el servidor se hagan transportando solamente los datos necesarios e involucrados en la petición que se esta haciendo en ese momento por el cliente con el objetivo de no sobrecargar el servidor de datos innecesarios y así hacer la interfaz más agradable y manejable para el cliente.

Seguridad

Se puede decir en cuanto a la seguridad de que se va a utilizar el protocolo SCL para la publicación de los servicios, que además garantiza suma confidencialidad de los datos personales de los usuarios de la infraestructura y los datos proporcionados por las aplicaciones serán guardados en la base de datos gestionada bajo la responsabilidad del sistema de autenticación, creada para facilitar el uso de los servicios que brinda. Otro aspecto a tener en cuenta es que el código se prepara para evitar ataques de inyección a SQL y se validan todos los datos introducidos cliente –servidor.

Ayuda

EL sistema cuenta con una ayuda de orientación al usuario. Esta constituida por una página FAQ online, donde el usuario puede acceder a un curso de entrega de capacitación para transferencias tecnológicas y una documentación de cómo usar los servicios.

El análisis y el diseño que se realizo en este capítulo tuvo como objetivo modelar y describir todas las clases que integran la estructura del sistema (SSO), lo que facilitó un mejor entendimiento de la estructura del sistema como base para la realización de la ingeniería del producto. Con estos modelos y descripciones se obtienen artefactos con datos suficientes para pasar a la implementación del producto.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

En este capítulo se llevaran a cabo los flujos de trabajo de implementación y prueba. Con sus definiciones correspondientes a estos se pretende obtener el código fuente de la aplicación y en la parte de pruebas se realizaran una serie de pruebas que condicionen las funcionalidades con las que contara el sistema.

4.1 Implementación

Se conoce que el objetivo principal de esta disciplina es convertir los elementos del diseño encontrados en elementos de implementación que dichos elementos se pueden decir que son: códigos fuentes, ejecutables, binarios, entre otros .Otro aspecto a tener en cuenta como disciplina son las pruebas de unidad, las cuales son capaces de limitarse a los componentes del software implementado .De esta disciplina se obtiene un sistema ejecutable estable ,constituidos de los resultados de los programadores individuales .

Diagrama de despliegue

En el diagrama de despliegue se muestra la situación física de los componentes lógicos que fueron desarrollados, es decir sitúa el software en el hardware y además muestra las relaciones que existen entre ellos. La configuración de los elementos de procesamiento de tiempo de ejecución y los componentes de software, estarán formados por instancias de los componentes de software que representan manifestaciones del código en tiempo de ejecución (los componentes que sólo sean utilizados en tiempo de compilación deben mostrarse en el diagrama de componentes) como se muestra en la figura 24.

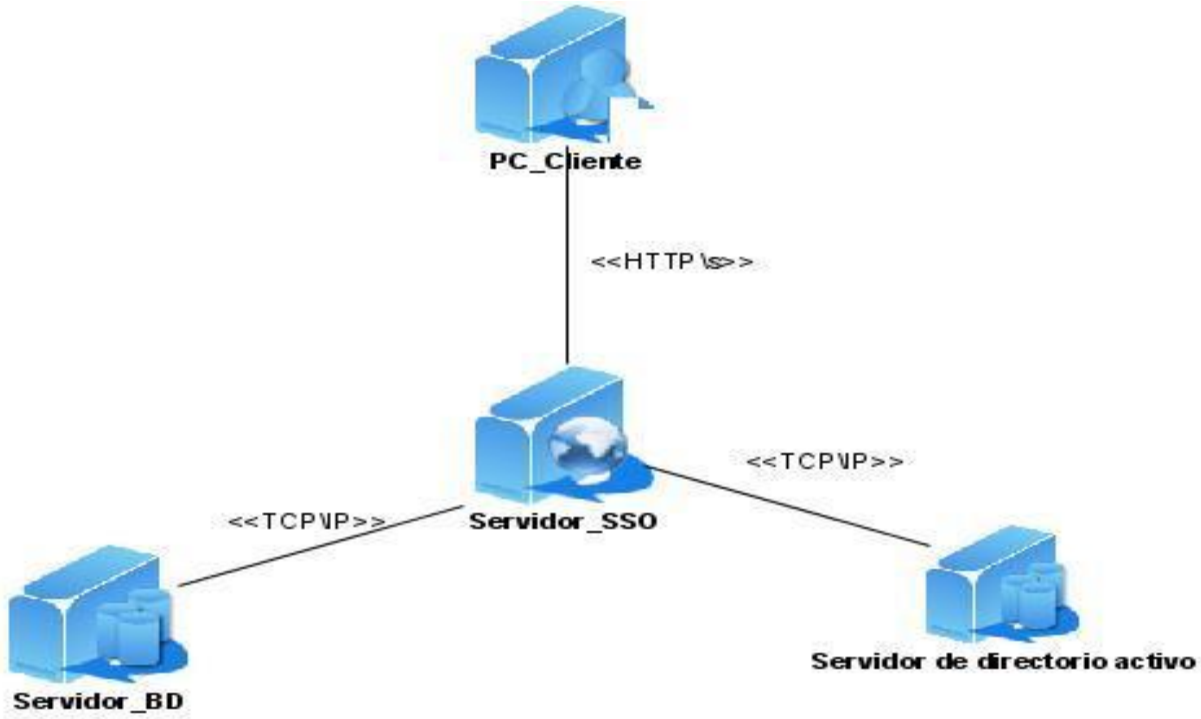


Figura 24.Diagrama de despliegue

Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones entre componentes de software, sean estos componentes de código fuente, binarios o ejecutables. Se puede decir que el diagrama de componentes tiene en consideración los requisitos que se relacionan con la facilidad de desarrollo, la gestión de software, la reutilización, y las restricciones de los lenguajes de programación y de las herramientas utilizadas en el desarrollo. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Los elementos de modelado dentro de un diagrama de componentes serán componentes y paquetes. Los distintos tipos de componentes que existen se pueden agrupar en paquetes según el criterio lógico lo que esto facilita simplificar a la hora de implementar el sistema, los paquetes representan una división física del sistema y ellos se organizan en una jerarquía de capas donde capa tiene una interfaz bien definida.

Los diagramas de componentes se utilizan para modelar código fuente, versiones ejecutables, bases de datos físicas, entre otros:

Código fuente: En el modelado de código fuentes se suelen utilizar para representar las dependencias entre los ficheros de código fuente, o para modelar las diferentes versiones de estos ficheros. Para ello se deben identificar el conjunto de archivos de código fuente de interés y estereotiparlos como archivos file, agruparlos en paquetes, utilizar valores etiquetados para la información de versiones y modelar las dependencias de compilación.

Código ejecutable: Se utiliza para modelar la distribución de una nueva versión a los usuarios. Para tal propósito se identifican el conjunto de componentes ejecutables que intervienen, se utilizan estereotipos para los diferentes tipos de componentes (ejecutables, bibliotecas, tablas, archivos, documentos), se consideran las relaciones entre dichos componentes que la mayoría de las veces incluirán interfaces que son exportadas (realizadas) por ciertos componentes e importadas (utilizadas) por otros.

Bases de datos física: UML permite el modelado de bases de datos físicas así como de los esquemas lógicos de bases de datos. [5]

Para la realización del diagrama de componentes UML define cinco estereotipos estándar que se aplican a los componentes encontrados:

Ejecutable: Especifica un componente que se puede ejecutar

en un nodo.

Library: Especifica una biblioteca de objetos estática o dinámica.

Table: Especifica un componente que representa una tabla de una base de datos.

File: Especifica un componente que representa un documento que contiene código fuente o datos.

Document: Especifica un componente que representa un documento.

Diagrama de componentes: Paquete de servicios.

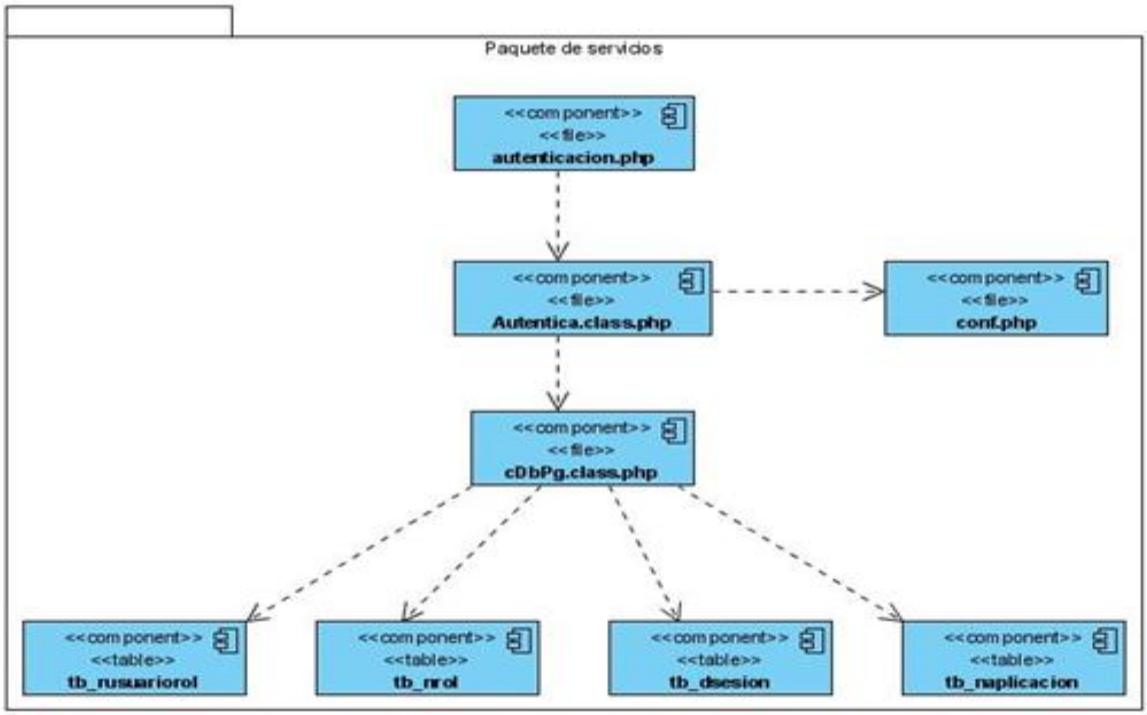


Figura 25.Diagrama de componentes:Paquete deservicios

Diagrama de componentes: Paquete de administración

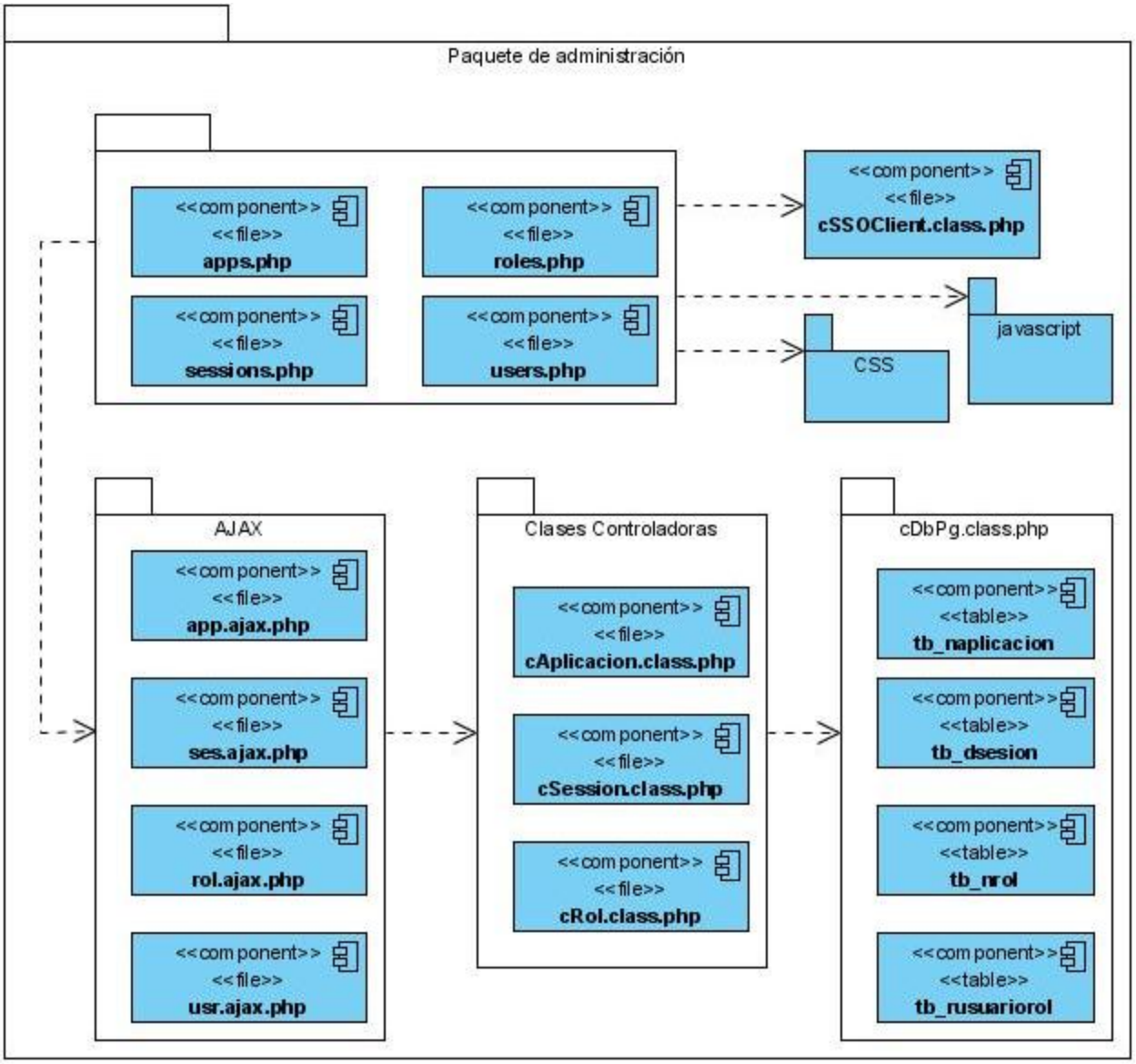


Figura 26.Diagrma de componentes:Paquete de administración

4.2 Pruebas

El Sistema de autenticación y autorización esta sujeto a pruebas, debido a la necesidad se hacer las mismas en un servidor real de la compañía de PDVZA. Producto que de esta forma es como se pueden medir la calidad del producto, así como verificar su rendimiento y eficiencia.

CONCLUSIONES

En este documento hemos plasmado y cumplido los objetivos trazados en el trabajo de diploma para optar por el título de ingenieros en ciencias informáticas. Nos propusimos desarrollar una aplicación vinculada al proyecto la Intranet de PDVSA que ayudar a los usuarios de la corporación a tener acceso a múltiples aplicaciones de la infraestructura. Este objetivo se cumplió eficazmente con la elaboración de este producto.

También se trazaron objetivos específicos, a los cuales se le dio cumplimiento en el transcurso del desarrollo de la aplicación. La construcción del marco teórico de la utilización del SSO a partir del estudio del estado del arte, se materializó con la consulta a múltiples bibliografías que ampliaron los conocimientos relacionados con el tema de la utilidad del SSO y los servicios Web.

Se logró implementar la aplicación con el nombre SSO: ingle Sing-On. La codificación se realizo con el empleo de PHP como lenguaje de programación, aunque algunos ficheros fueron implementados en Ajax para lograr un producto mejor y de fácil manejo.

Con la introducción del nuevo producto en la infraestructura de PDVSA se pretende facilitar un manejo mejor de los procesos modulares de la corporación lo cual se espera que trabaje con mas eficiencia y lograr alcanzar un alto componente de seguridad tanto por parte de los usuarios como también por parte de las aplicaciones.

Con la realización de este trabajo, los autores aprendimos un tema completamente desconocido y es el de “mapeo de credenciales” Profundizamos en el análisis de los tipos de arquitecturas a utilizar en la implantación del SSO y por parte de la implementación el estudio de los lenguajes de programación a utilizar como son: PHP, Ajax para el desarrollo de aplicaciones Web. Otro de los estudios realizados fue referente a la utilización de Web Service.

RECOMENDACIONES

Se recomienda a los seguidores de esta idea empezada en la universidad, que continúen con el estudio de los diferentes tipos de sistemas de autenticación y autorización centralizada existentes en la actualidad, así como también profundizar más en el estudio de los diferentes tipos de arquitecturas que implementa un SSO para así escoger la adecuada a implementar y que este acorde con la necesidad donde se valla a poner en marcha dicho sistema. La introducción de sistemas de autenticación y autorización centralizados en otras infraestructuras, organizaciones y en la universidad de las ciencias informáticas.

En el caso del desarrollo de esta aplicación, recomendamos que se amplíe el uso del SSO como parte de próximas versiones de este producto. Además, la intención es incrementar las facilidades de uso para los usuarios y administradores, que puedan emplear el sistema de autenticación que más agradable les sea.

BIBLIOGRAFÍA

Cómo Utilizar Microsoft .NET Passport. [En línea] [Citado el: 20 de enero de 2008.] http://www.dummies.com/WileyCDA/DummiesArticle/C-oacute-mo-Utilizar-Microsoft-NET-Passport.id-2117.html.

Soluciones empresariales. [En línea] [Citado el: 10 de diciembre de 2007.] http://www.google.es/enterprise/security.html.

2008. *OpenID en ASP.NET.* [En línea] 2008. [Citado el: 20 de febrero de 2008.] http://www.lobosoft.es/2008/01/26/openid-en-aspnet/.

2007. *Openid.* [En línea] 2007. [Citado el: 10 de enero de 2008.] http://openid.es/.

Análisis y Diseño. [En línea] [Citado el: 20 de enero de 2008.] http://merinde.rinde.gob.ve/index.php?option=com_content&task=view&id=137&Itemid=192.

Implementación. [En línea] [Citado el: 20 de enero de 2008.] http://merinde.rinde.gob.ve/index.php?option=com_content&task=view&id=138&Itemid=193.

Modelo de Implementación: Diagramas de Componentes y Despliegue. [En línea] [Citado el: 10 de enero de 2008.] http://www.info-ab.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

PostGreSQL vs. MySQL. [En línea] [Citado el: 15 de diciembre de 2007.] http://www.netpecos.org/docs/mysql_postgres/x15.html .

Caballero.I, Cano.J. 2003. *Consideraciones para implementar una arquitectura single sign-on .* [En línea] julio de 2003. [Citado el: 15 de enero de 2008.] http://www.criptored.upm.es/guiateoria/gt_m142j.htm.

Eliurkis. 2007. *Single Sign On: Sistema de Autenticación Único.* [En línea] 1 de septiembre de 2007. [Citado el: 10 de enero de 2008.] http://www.deepinphp.com/2007/09/01/single-sign-on-sistema-de-autenticacion-unico/.

Fernandez.A. 2001. *Diagrama de Componentes .* [En línea] 20 de marzo de 2001. [Citado el: 10 de diciembre de 2007.] http://www-gris.det.uvigo.es/~avilas/UML/node49.html.

2005. Newcomlab. *Arquitectura en 3 capas de la solución.* [En línea] 2005. [Citado el: 15 de diciembre de 2007.] http://www.newcomlab.com/servicios/b2c_arquitectura.asp.

Restrepo.T. 2004. *Administración del Enterprise Single Sign-On.* [En línea] 2004. [Citado el: 11 de diciembre de 2007.] http://www.winterdom.com/dev/esp/BTS_SSO_1.htm.

Rodríguez.D. 2006. *Sistema para el Control y Gestión de la contratación y los servicios de Auditoría en Conas.* [En línea] junio de 2006. [Citado el: 10 de enero de 2008.] http://ftp.jovenclub.cu/Tesis/IF06T025.pdf.

Torvalds.L. *LINUX MAGAZINE 10.* [En línea] [Citado el: 15 de diciembre de 2007.] http://www.scribd.com/doc/209465/LINUX-MAGAZINE-10.

REFERENCIA

1. **Eliurkis.** Single Sign On: Sistema de Autenticación Único. [En línea] 1 de Septiembre de 2007. [Citado el: 20 de febrero de 2008.] http://www.deepinphp.com/2007/09/01/single-sign-on-sistema-de-autenticacion-unico/.

2. **Álvarez, Miguel Ángel.** DesarrolloWeb. "Qué es PHP". [En línea] 9 de Mayo de 2001. [Citado el: 2008 de enero de 15.] http://www.desarrolloweb.com/articulos/392.php.

3. **González, Benjamín.** XML Web Services. [En línea] 24 de Junio de 2004. [Citado el: 2008 de Enero de 18.]

4. **Rodríguez Esquijarosa, Duniesky.** Sistema para el Control y Gestión de. [En línea] Junio de 2006. [Citado el: 2008 de Marzo de 24.]

5. **Fernandez Vilas, Ana.** Diagrama de Componentes. [En línea] 20 de Marzo de 2001.

ANEXOS

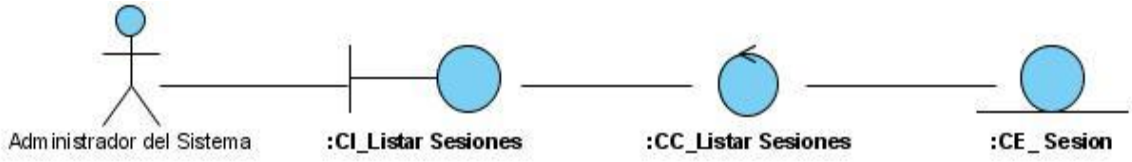


Diagrama de clases de análisis: Listar sesiones

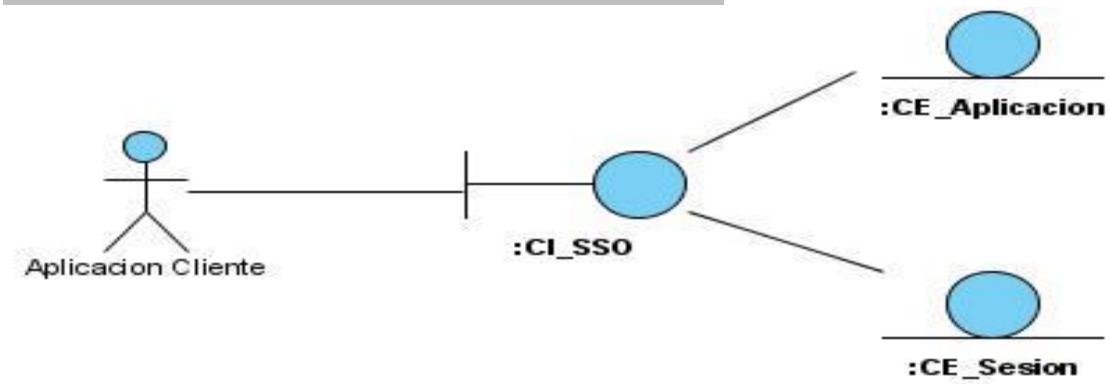


Diagrama de clases de análisis: Cerrar sesión

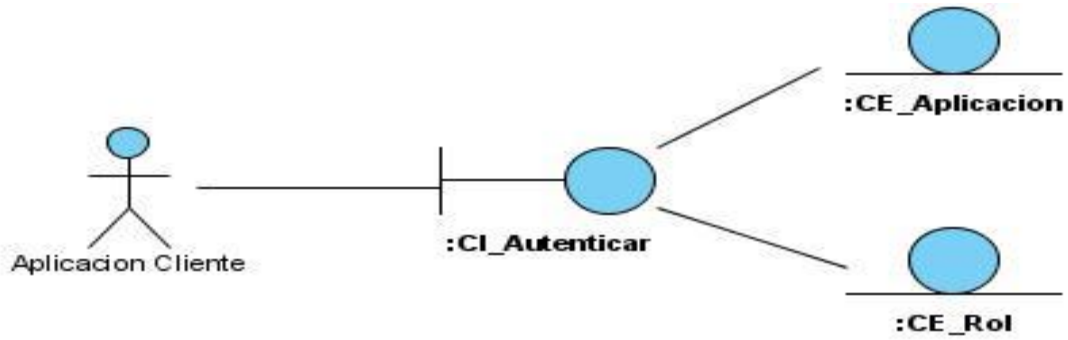


Diagrama de clases de análisis: Solicitar roles

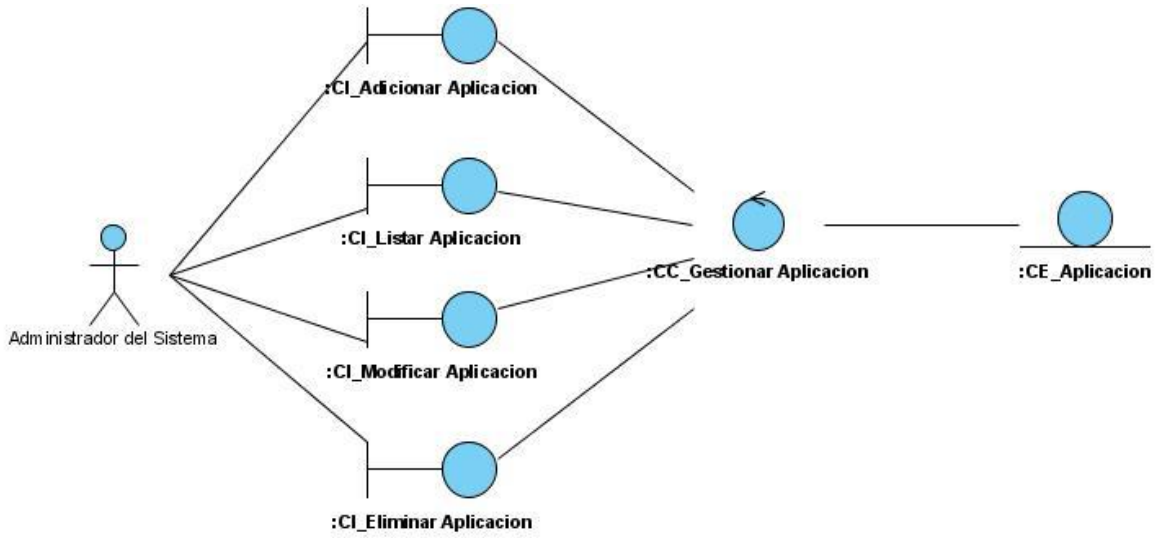


Diagrama de clases de análisis: Gestionar aplicaciones

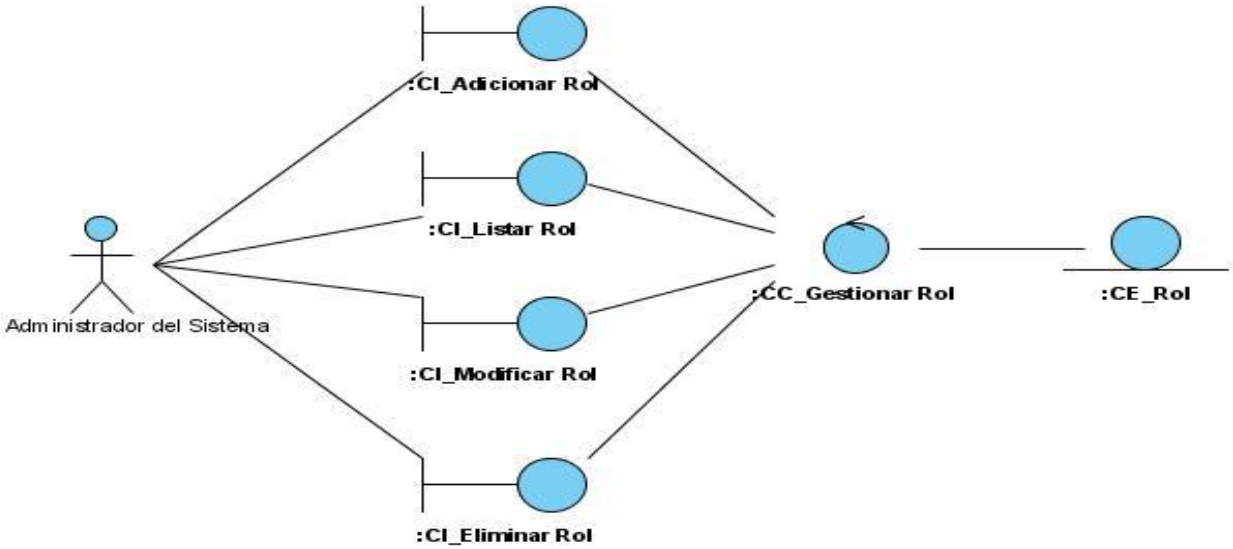
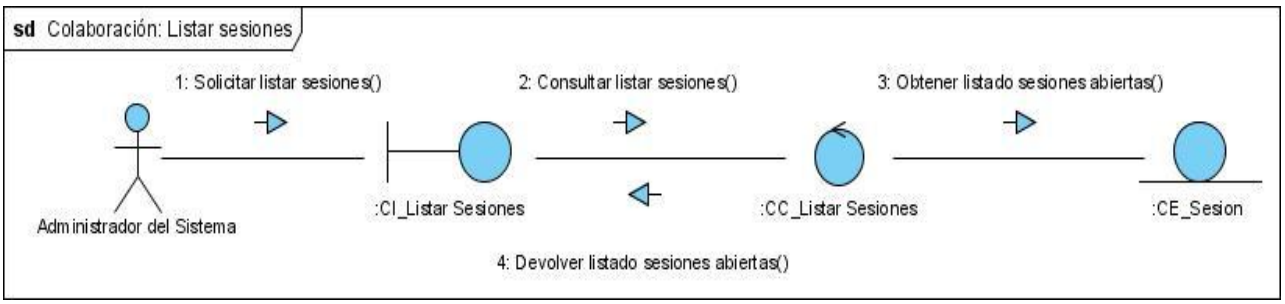
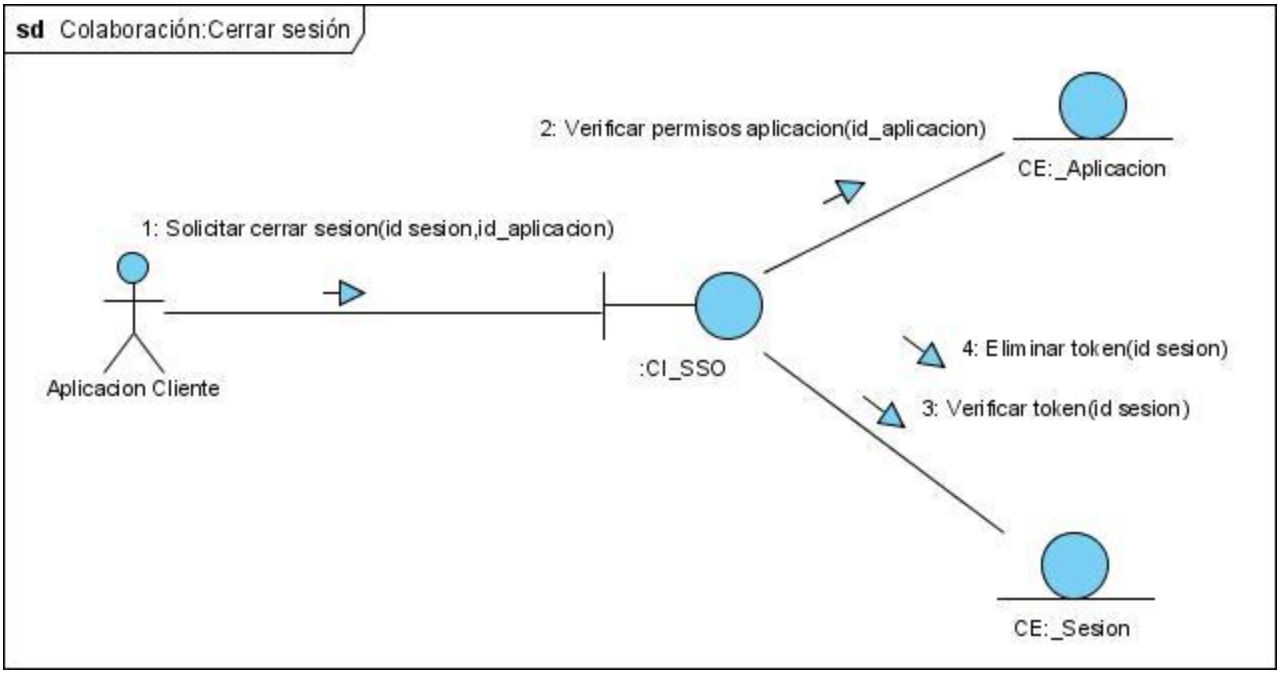
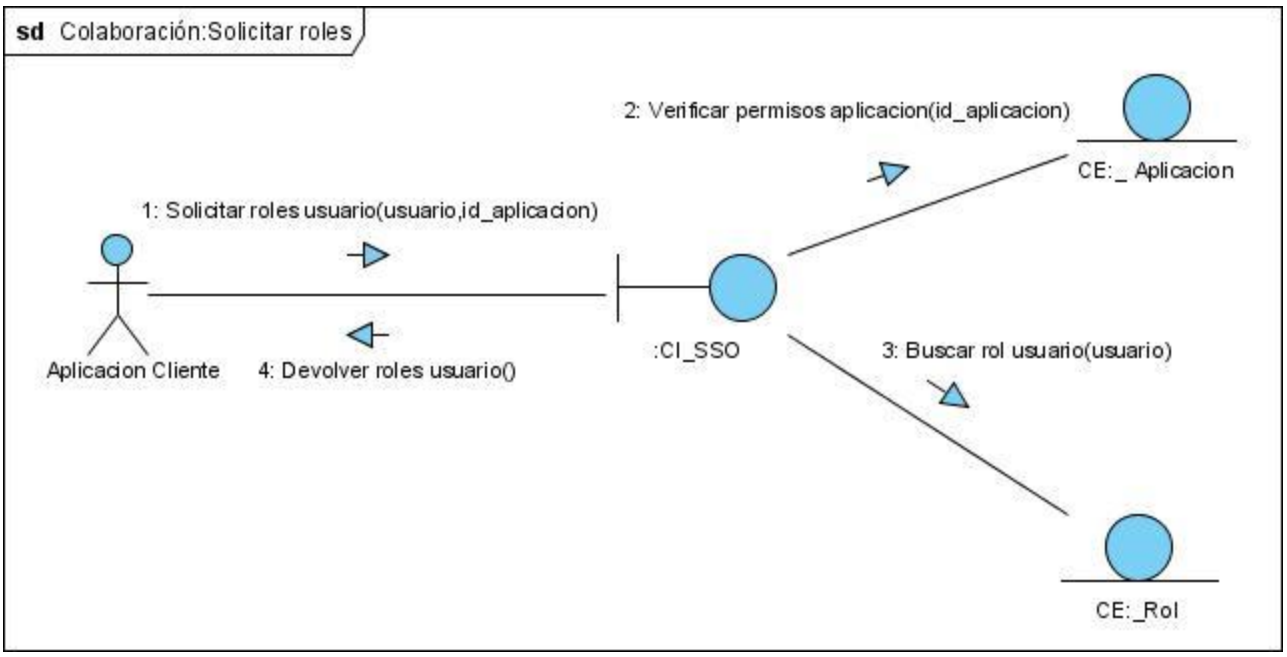
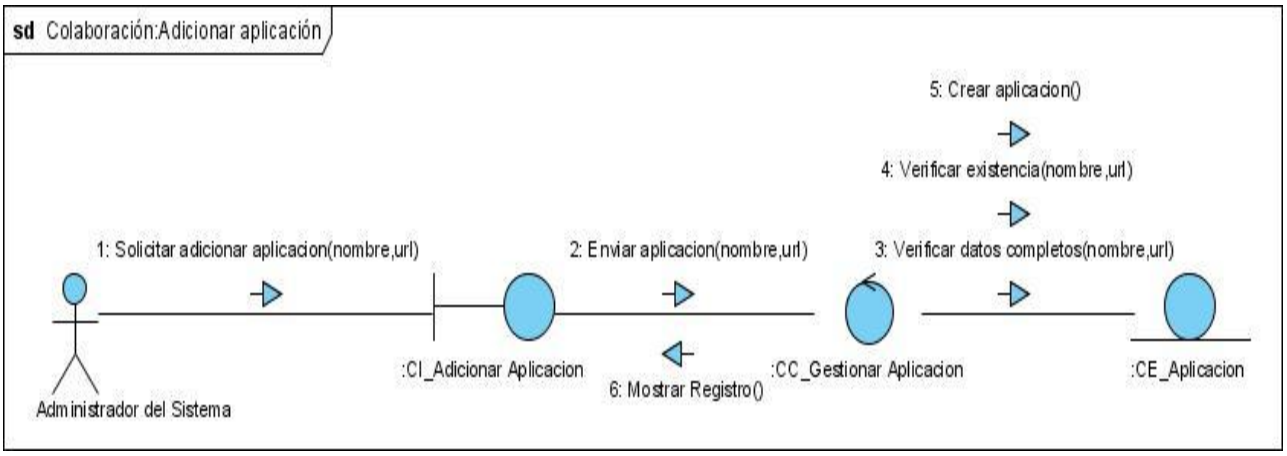


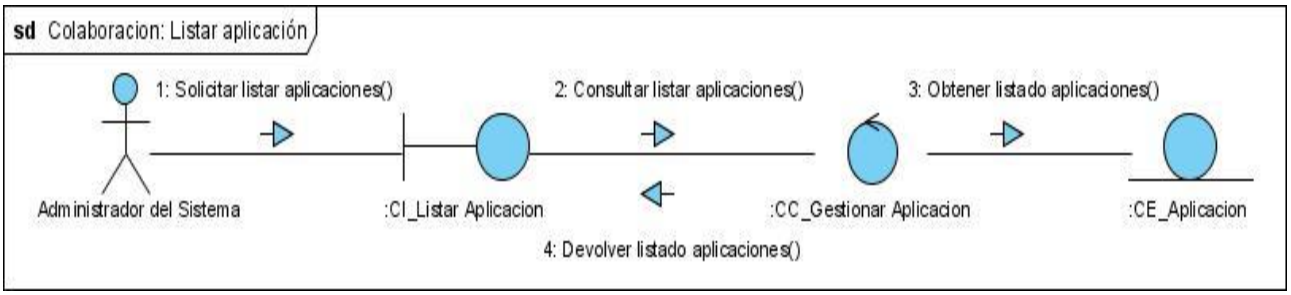
Diagrama de clases de análisis: Gestionar roles

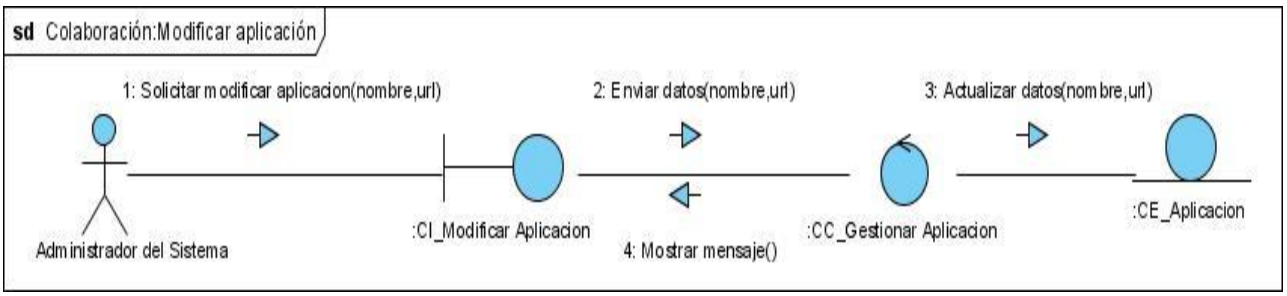

sd Colaboración: Listar sesiones
1: Solicitar listar sesiones()
2: Consultar listar sesiones()
3: Obtener listado sesiones abiertas()
Administrador del Sistema
:CI_Listar Sesiones
:CC_Listar Sesiones
:CE_Sesion
4: Devolver listado sesiones abiertas()

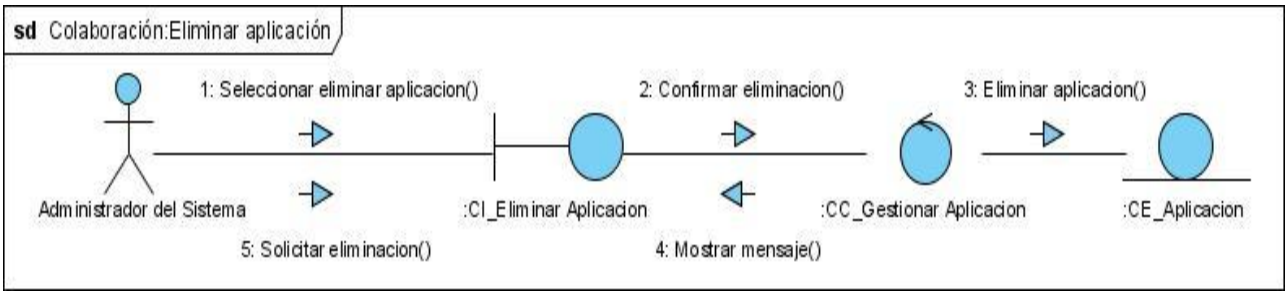

sd Colaboración:Cerrar sesión
2: Verificar permisos aplicacion(id_aplicacion)
CE:_Aplicacion
1: Solicitar cerrar sesion(id sesion,id_aplicacion)
Aplicacion Cliente
:CI_SSO
4: Eliminar token(id sesion)
3: Verificar token(id sesion)
CE:_Sesion

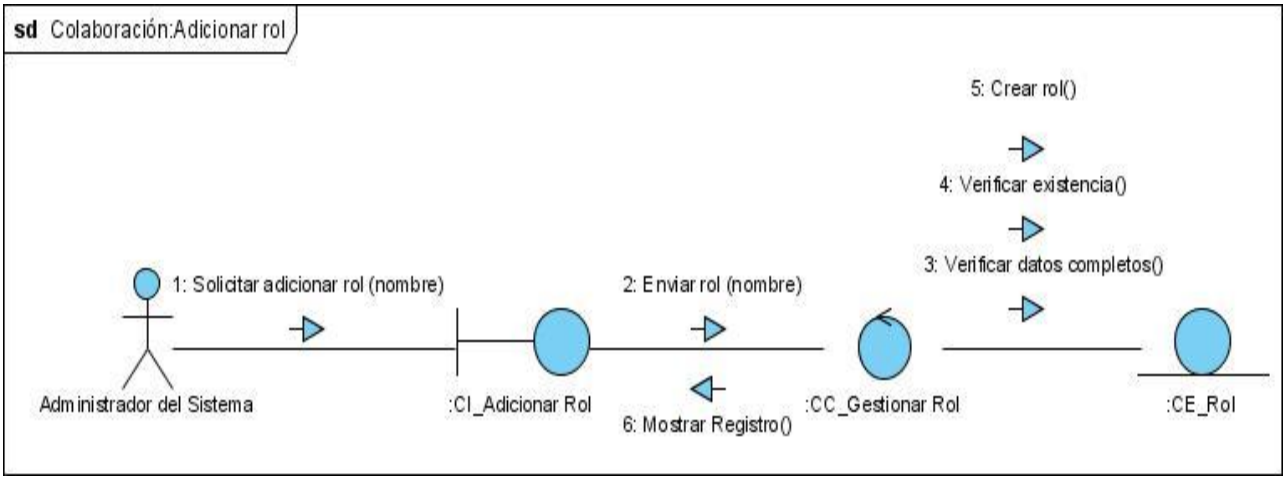

sd Colaboración:Solicitar roles
2: Verificar permisos aplicacion(id_aplicacion)
CE:_ Aplicacion
1: Solicitar roles usuario(usuario,id_aplicacion)
Aplicacion Cliente
4: Devolver roles usuario()
:CI_SSO
3: Buscar rol usuario(usuario)
CE:_Rol

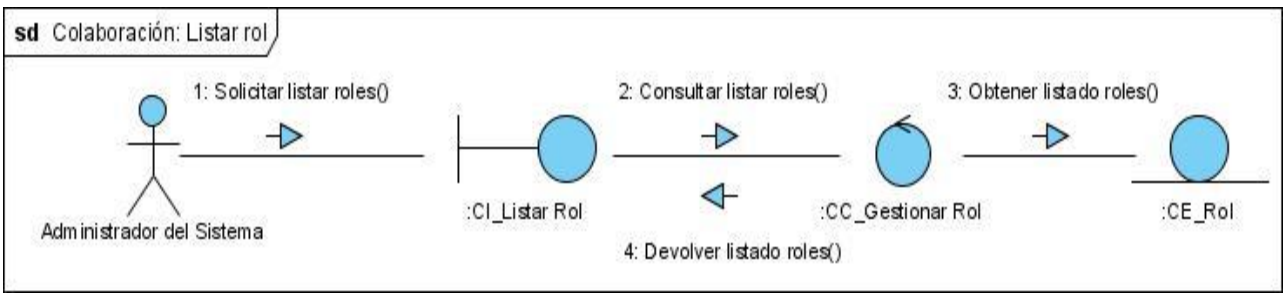

sd Colaboración:Adicionar aplicación
5: Crear aplicacion()
4: Verificar existencia(nombre,url)
1: Solicitar adicionar aplicacion(nombre,url)
2: Enviar aplicacion(nombre,url)
3: Verificar datos completos(nombre,url)
Administrador del Sistema
:CI_Adicionar Aplicacion
6: Mostrar Registro()
:CC_Gestionar Aplicacion
:CE_Aplicacion

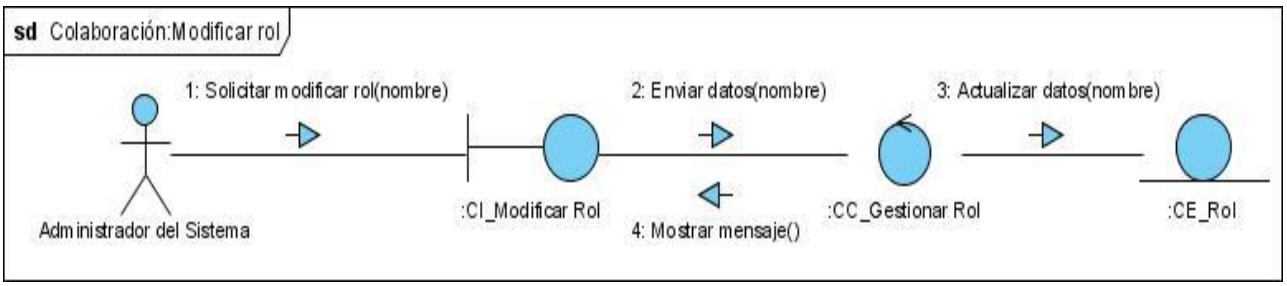

sd Colaboracion: Listar aplicación
1: Solicitar listar aplicaciones()
2: Consultar listar aplicaciones()
3: Obtener listado aplicaciones()
Administrador del Sistema
:CI_Listar Aplicacion
:CC_Gestionar Aplicacion
:CE_Aplicacion
4: Devolver listado aplicaciones()

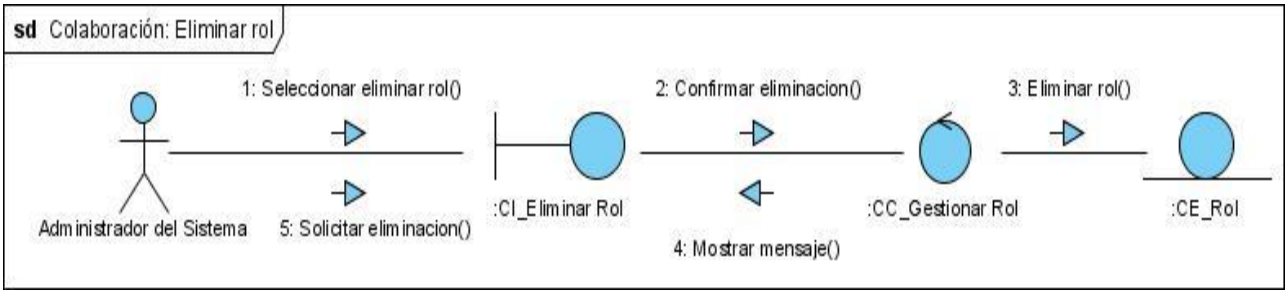

sd Colaboración:Modificar aplicación
1: Solicitar modificar aplicacion(nombre,url)
2: Enviar datos(nombre,url)
3: Actualizar datos(nombre,url)
Administrador del Sistema
:CI_Modificar Aplicacion
:CC_Gestionar Aplicacion
:CE_Aplicacion
4: Mostrar mensaje()


sd Colaboración:Eliminar aplicación
1: Seleccionar eliminar aplicacion()
2: Confirmar eliminacion()
3: Eliminar aplicacion()
Administrador del Sistema
:CI_Eliminar Aplicacion
:CC_Gestionar Aplicacion
:CE_Aplicacion
5: Solicitar eliminacion()
4: Mostrar mensaje()


sd Colaboración:Adicionar rol
5: Crear rol()
4: Verificar existencia()
3: Verificar datos completos()
1: Solicitar adicionar rol (nombre)
2: Enviar rol (nombre)
Administrador del Sistema
:CI_Adicionar Rol
:CC_Gestionar Rol
:CE_Rol
6: Mostrar Registro()


sd Colaboración: Listar rol
1: Solicitar listar roles()
2: Consultar listar roles()
3: Obtener listado roles()
Administrador del Sistema
:CI_Listar Rol
:CC_Gestionar Rol
:CE_Rol
4: Devolver listado roles()





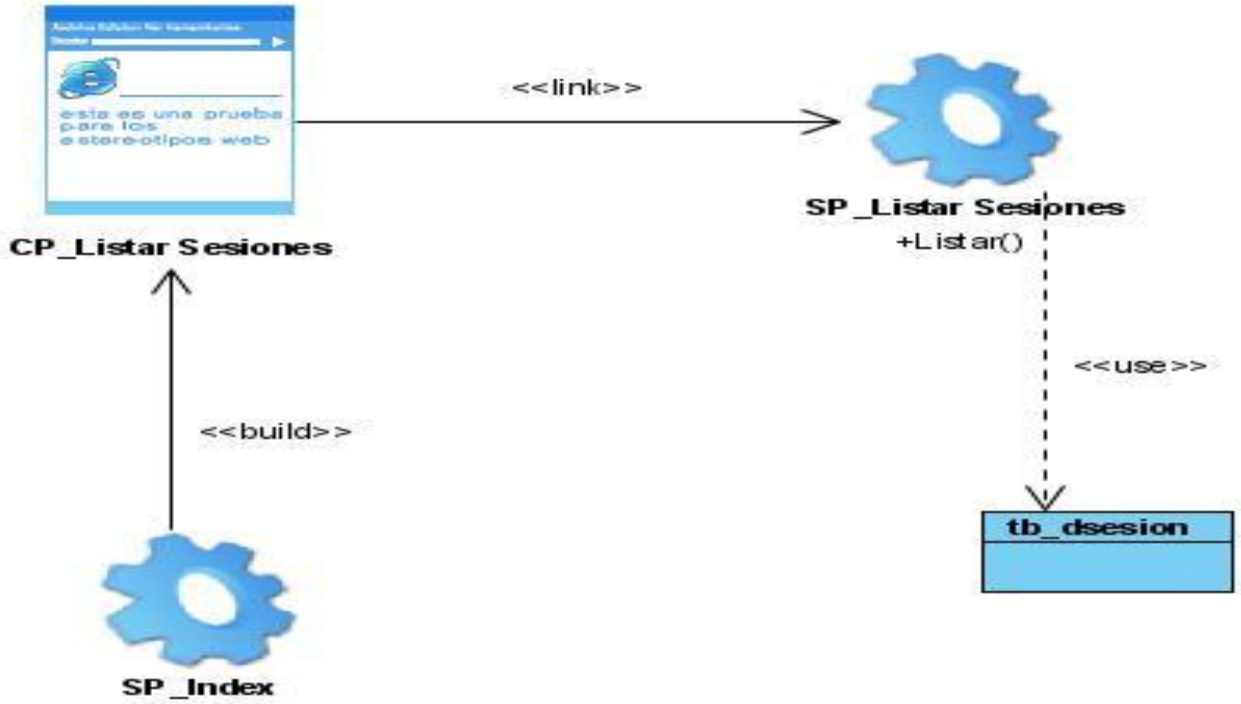


Diagrama de clases del diseño Web: Listar sesiones

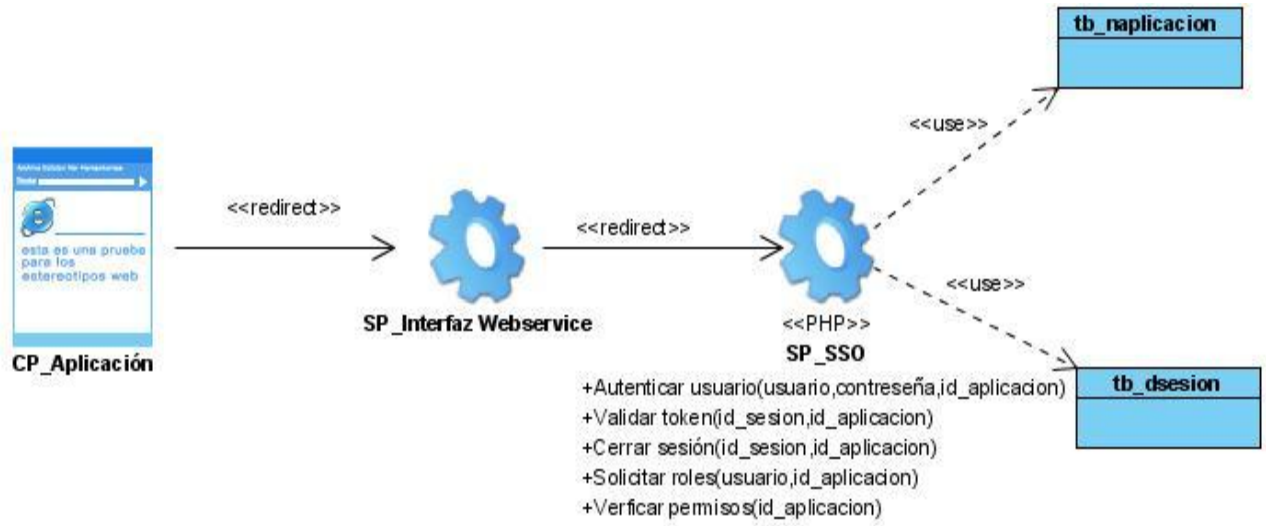


Diagrama de clases del diseño Web: Cerrar Sesión

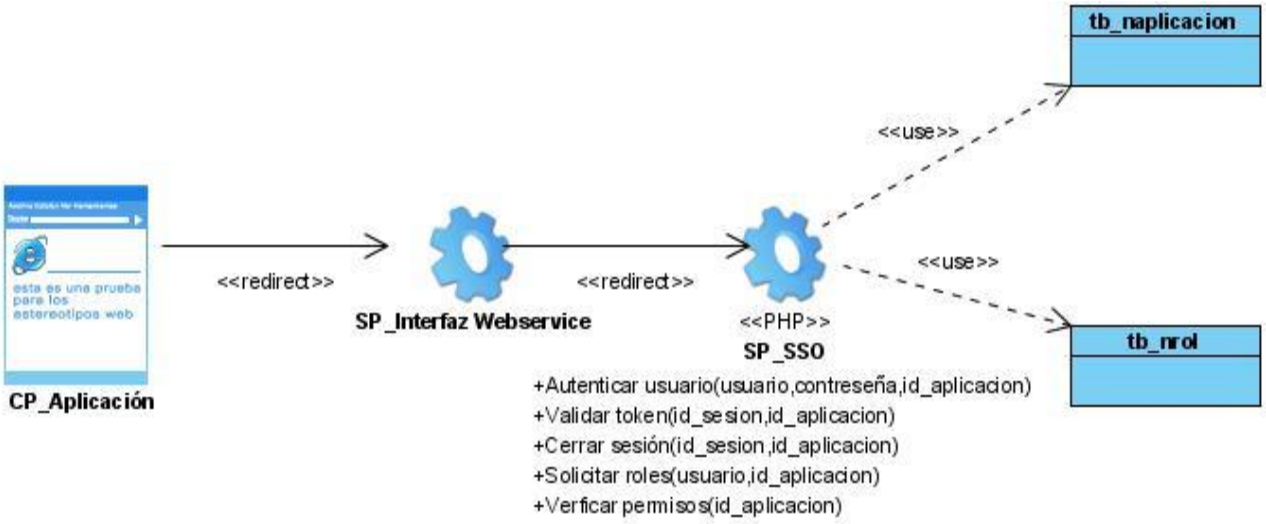


Diagrama de clases del diseño Web: Solicitar roles

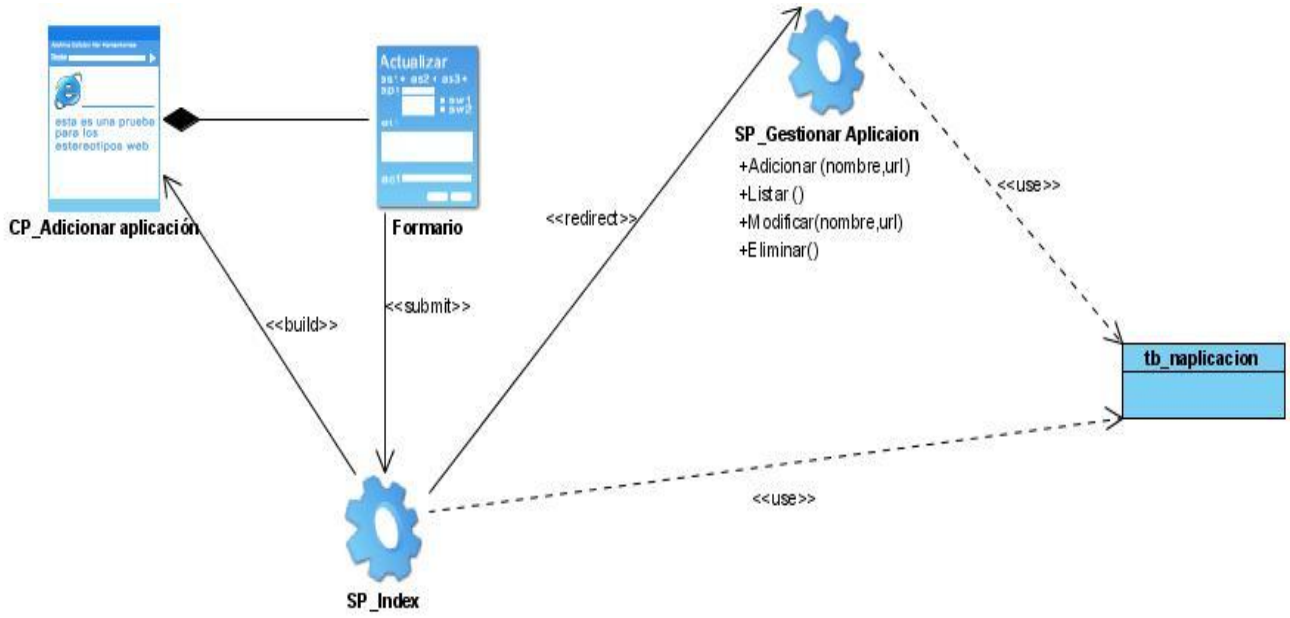


Diagrama de clases del diseño Web: Adicionar aplicación

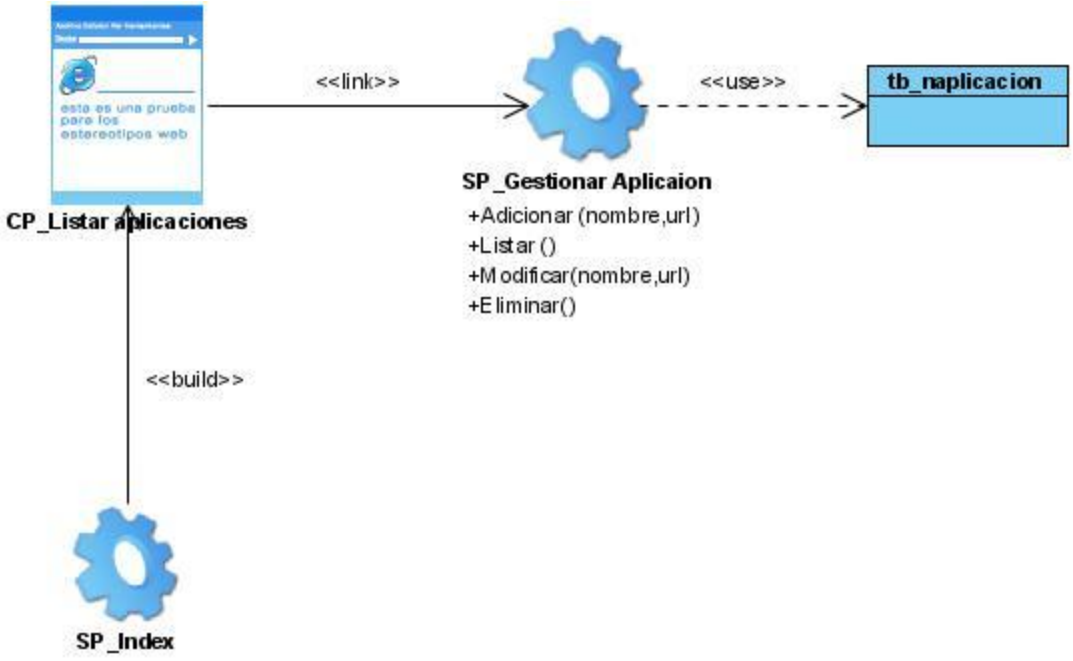


Diagrama de clases del diseño Web: Listar aplicación

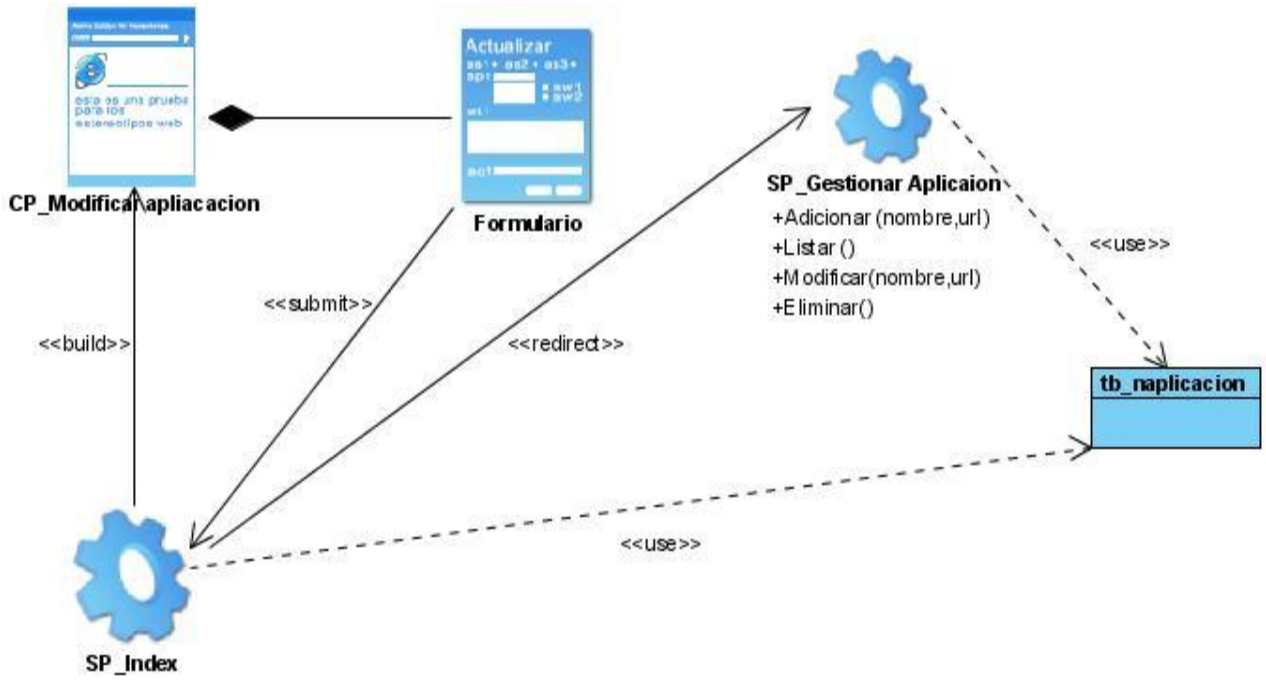


Diagrama de clases del diseño Web: Modificar aplicación

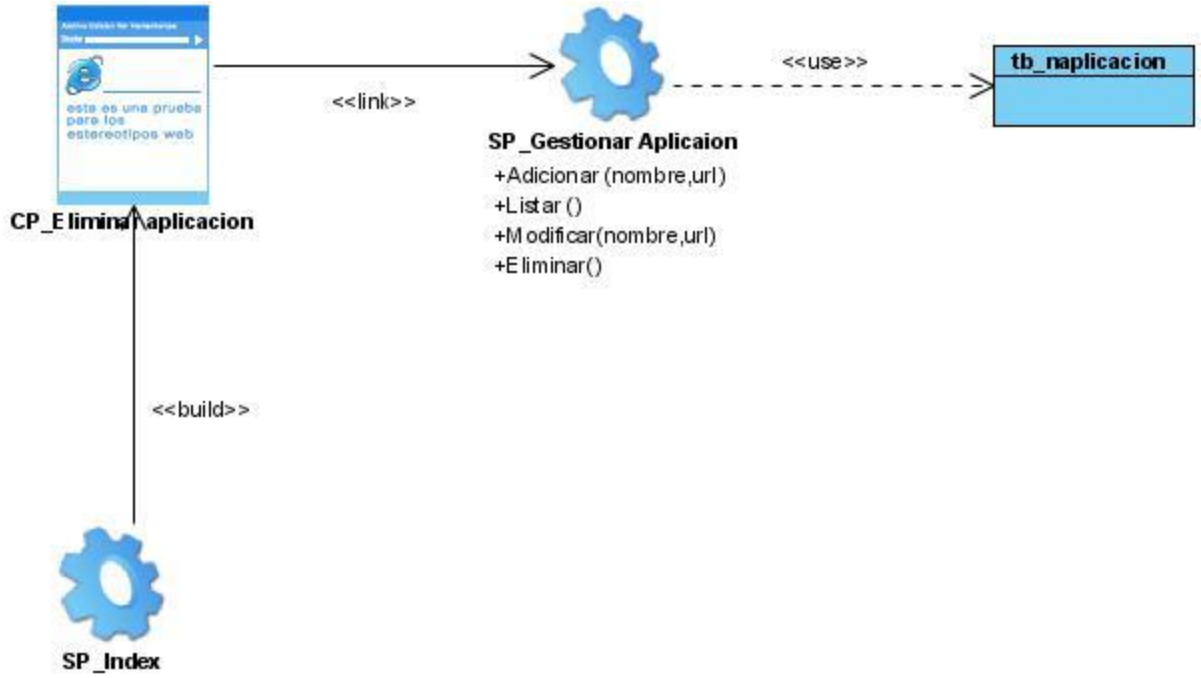


Diagrama de clases del diseño Web: Eliminar aplicación

GLOSARIO DE TERMINOS

PDVSA: Es la corporación estatal de la República Bolivariana de Venezuela que se encarga de la exploración, producción, manufactura, transporte y mercadeo de los hidrocarburos, de manera eficiente, rentable, segura, transparente y comprometida con la protección ambiental.

PostgreSQL: Es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional y es servidor de base de datos relacional libre.

Gateway: Aplicación que permite la transmisión de datos de un servidor en red a otro.

SQL Server: Es un software que permite manipular/administrar bases de datos.

Oracle: Es un sistema de gestión de base de datos relacional (o RDBMS por el acrónimo en inglés de Relational Data Base Management System) para base de datos empresariales.

Terminal Server: Son un componente de los sistemas operativos Windows que permite a un usuario acceder a las aplicaciones y datos almacenados en otro ordenador mediante un acceso por red.

Win32: Es el API de Windows para desarrollar aplicaciones de 32 bits.

NTLM: Es un protocolo de autenticación que utiliza el cifrado para la transmisión segura de contraseñas.

X.509: Es un estándar para infraestructuras de claves publicas y define formatos específicos de llaves publicas.

LDAP: Es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado.

Active Directory: Es el nombre utilizado por Microsoft para referirse a su implementación de un servicio de directorio en una red distribuida.

XML: Acrónimo de eXtensible Markup Language (lenguaje de marcas extensible), no es un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.

SAML 2.0: Es un estándar XML para el intercambio de autenticación y autorización de datos de seguridad entre dominios.

URL: Es el Localizador Uniforme de Recursos, o dicho mas claramente, es la dirección que localiza una información dentro de Internet.

http: Es el protocolo utilizado para realizar la transferencia de archivos (texto, imágenes, sonido, video, y otros archivos multimedia) en la web.