

IMPLEMENTACIÓN Y MONTAJE DE UN SISTEMA MEDIADOR DE USUARIOS
EN EQUIPOS CISCO DE LA EMPRESA DE TELECOMUNICACIONES DE
PEREIRA

JHON ALEXANDER MARÍN SÁNCHEZ
ANDERSON LENIS PAREJA

UNIVERSIDAD LIBRE
FACULTAD DE INGENIERÍAS
INGENIERÍA DE SISTEMAS
PEREIRA

2011

IMPLEMENTACIÓN Y MONTAJE DE UN SISTEMA MEDIADOR DE USUARIOS
EN EQUIPOS CISCO DE LA EMPRESA DE TELECOMUNICACIONES DE
PEREIRA

JHON ALEXANDER MARÍN SÁNCHEZ

ANDERSON LENIS PAREJA

Trabajo de grado para optar

Título de ingeniería de sistemas

Director: Ing. Raúl Alberto Gaviria

UNIVERSIDAD LIBRE

FACULTAD DE INGENIERÍAS

INGENIERÍA DE SISTEMAS

PEREIRA

2011

Firma del presidente del jurado

Firma del presidente del jurado

Firma del presidente del jurado

Pereira, 2 de Mayo de 2011

Nota de aceptación:

Gracias a todas las personas que intervinieron en el proceso de formación académica en especial a mi compañera sentimental que me dio lo más hermoso de la vida y que es mi razón de vivir, gracias a mis amigos que siempre me apoyaron y me brindaron sus conocimientos para cumplir con las metas trazadas.

Jhon Alexander Marín

Agradecer a dios y a mi familia por darme la posibilidad de realizar todos mis estudios, no solo universitarios sino en general.

Agradecer a mis amigos más cercanos los cuales me dieron apoyo y conocimientos para la realización de este proyecto, y por ultimo le agradezco a la universidad Libre por los conocimientos que me brindaron mediante sus docentes.

Anderson Lenis Pareja

Agradecimientos

Queremos agradecer sinceramente el apoyo y la asesoría académica del Ingeniero Raúl Alberto Gaviria y a todos los docentes que de manera directa o indirecta contribuyeron para el desarrollo de este proyecto. A todos aquellos que nos acompañaron en momentos de alegría y en horas difíciles. A la Universidad Libre por el desarrollo de habilidades académicas y humanas. Queremos también agradecer a la Empresa de Telecomunicaciones de Pereira por brindarnos la posibilidad de implementar nuestro proyecto de grado en el área de las redes informáticas, el cual era el campo al cual estaba orientado nuestro trabajo.

TABLA DE CONTENIDO

	Pag
INTRODUCCIÓN	16
1. DEFINICIÓN DEL PROBLEMA.....	17
1.1. TITULO	17
1.2. DESCRIPCIÓN DEL ÁREA DEL PROBLEMA	17
1.3. DIAGNÓSTICO DEL PROBLEMA	18
2. JUSTIFICACIÓN.....	23
3. OBJETIVOS	25
3.1. OBJETIVO GENERAL	25
3.2. OBJETIVOS ESPECÍFICOS	25
4. DELIMITACIÓN DEL PROBLEMA	26
5. METODOLOGÍA	27
5.1. RESEÑA DEL MARCO INSTITUCIONAL	28
6. MARCO CONCEPTUAL	29
6.1. SOFTWARE	29
6.1.1. Formas del software.....	32
6.1.2. El proceso de creación del software	32
6.1.3. Seguridad.....	33
6.2. DATOS DE SOFTWARE.....	33
6.3. DEFINICIONES DE INGENIERÍA DE SOFTWARE.....	35
6.4. ACTIVIDADES DE DESARROLLO DE INGENIERÍA DE SOFTWARE.....	35
6.4.1. Obtención de requerimientos.....	36
6.4.2. Análisis.....	36
6.4.3. Diseño del sistema	36

6.4.4. Diseño de objetos	37
6.4.5. Implementación	37
6.5. PRODUCTO DE SOFTWARE	38
6.6. PROCESOS DE SOFTWARE	38
6.7. LA INGENIERÍA DE SOFTWARE	38
6.8. DEFINICIÓN DE MODULO DE CICLO DE VIDA	39
6.8.1. Etapas en el ciclo	39
6.8.1.1. Expresión de necesidades	40
6.8.1.2. Especificaciones	40
6.8.2. Algunos médelos de ciclos de vida	40
6.8.2.1. Modelo cascada	40
6.8.2.2. Modelo en espiral	42
6.8.3. Descripción del ciclo de vida orientado a objetos según Grady Booch ...	44
6.9. ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS	47
6.10. CONCEPTOS DE LA PROGRAMACIÓN ORIENTADA A OBJETOS	49
6.11. INGENIERÍA DE REQUERIMIENTOS	51
6.11.1. Beneficios	51
6.11.2. ¿Por qué ingeniería de requerimientos?	51
6.12. DEFINICIÓN DEL MODELO DE MARCO LÓGICO	52
6.13. ARQUITECTURA CLIENTE/SERVIDOR.....	55
6.13.1. Introducción	55
6.13.2. Ventajas	55
6.13.3. Modelo cliente/servidor de dos capas	56
6.14. METODOLOGÍA PARA LA ELICITACIÓN DE REQUERIMIENTOS DEL SOFTWARE	57
6.15. ETAPAS DE LA ELICITACIÓN SEGÚN AMADOR DURAN Y BEATRIZ BERNARDEZ.....	58
6.15.1. Etapa 1	58
6.15.2. Etapa 2	59

6.15.3. Etapa 3	60
6.15.4. Etapa 4	61
6.14.5. Etapa 5	62
6.15.6. Etapa 6	63
6.16. ¿QUÉ ES JAVA?.....	65
6.17. ¿QUÉ ES PHP?	67
6.18. ¿QUÉ FRAMEWORK?	68
7. MARCO TEÓRICO	70
8. MARCO DE ANTECEDENTES.....	74
9. MARCO LEGAL.....	75
9.1. DECRETO 1360/1989 PROPIEDAD INTELECTUAL.....	75
9.2. CIRCULAR NO. 06 DE 15 DE ABRIL DE 2002 DE UNIDAD ADMINISTRATIVA ESPECIAL DIRECCIÓN NACIONAL DE DERECHOS DE AUTOR.....	75
9.3. LEY 1273 DE 2009 PROTECCIÓN DE LA INFORMACIÓN Y DE LOS DATOS.....	76
9.4. ACUERDO 003/2003 MODELO DE EVALUACIÓN DOCENCIA-SERVICIO....	76
9.5. MODELO DE EVALUACIÓN DE LA RELACIÓN DOCENCIA-SERVICIO: CRITERIOS BÁSICOS DE CALIDAD PARA CENTROS DE PRACTICAS FORMATIVAS	76
10. DESARROLLO METOLÓGICO.....	78
10.1. INTRODUCCIÓN AL ANÁLISIS DEL SOFTWARE.....	78
10.2. POBLACIÓN	78
10.2.1. Participantes.....	78
10.2.2. Beneficiarios	79
10.3. DESCRIPCIÓN DEL SISTEMA ACTUAL	79
10.4. OBJETIVOS DEL ANÁLISIS DEL SOFTWARE.....	79
10.4.1. Objetivo general	79
10.4.2. Objetivos específicos	80
10.4.2.1. Generación de informes.....	80

10.4.2.2. Gestión de usuarios	80
10.4.2.3. Validar usuarios.....	81
10.4.2.4. Gestión de perfil de usuarios y permisos	81
10.4.2.5. Gestión de comandos	82
10.4.2.6. Gestión de administradores.....	82
10.4.2.7. Seguridad	82
10.5. REQUISITOS DE INFORMACIÓN	83
10.5.1. Requisitos de almacenamiento de información	83
10.5.1.1. Información de los usuarios	83
10.5.1.2. Información de comandos.....	84
10.5.1.3. Relación entre usuarios y comandos	85
10.6. REQUISITOS FUNCIONALES	86
10.6.1. Definición de actores	86
10.6.1.1. Usuario.....	86
10.6.1.2. Base de datos	86
10.6.1.3. Interfaz de usuarios	87
10.6.2. Diagramas y documentación casos de uso	88
10.6.2.1. Modulo Validación de Usuarios	88
10.6.2.2. Modulo Gestión de Usuarios	92
10.6.2.2.1. Caso de uso Crear Usuario	92
10.6.2.2.2. Caso de uso Modificar Usuario.....	96
10.6.2.2.3. Caso de uso Consultar Usuario	100
10.6.2.2.4. Caso de uso Eliminar Usuario	103
10.6.2.3. Modulo gestión Comandos	107
10.6.2.3.1. Caso de uso Crear Comandos.....	107
10.6.2.3.2. Caso de uso modificar Comando.....	110
10.6.2.3.3. Caso de uso Consultar Comando	113
10.6.2.3.4. Caso de uso Eliminar Comando	116
10.6.2.4. Modulo gestión Logs.....	120

10.6.2.4.1. Caso de uso Consultar Logs	120
10.7. REQUERIMIENTOS NO FUNCIONALES.....	122
10.8. DIAGRAMA DE CLASES.....	126
11. DISEÑO.....	127
11.1. DISEÑO ARQUITECTÓNICO	127
11.2. DIAGRAMA DE DESPLIEGUE	128
11.3. PROTOTIPOS DE INTERFAZ GRAFICA DE USUARIO	129
11.3.1. Interfaz Validar Usuarios.....	129
11.3.2. Interfaz Menú Principal.....	131
11.3.3. Interfaz Asociar Usuario-Comando	132
11.3.4. Interfaz Editar Comando.....	133
11.3.5. Interfaz Comandos.....	135
11.3.6. Interfaz New Comando.....	136
11.3.7. Interfaz Log de Usuarios y Comandos.....	137
11.3.8. Interfaz Usuarios.....	138
11.3.9. Interfaz Crear Usuarios.....	139
11.3.10. Interfaz Editar Usuarios	142
11.3.11. Interfaz inicio de conexión	143
11.3.12. Interfaz Web para acceder al cli de los equipos Cisco	145
11.3.13. Interfaz dirección IP	146
11.3.14. Interfaz sesión cli	148
11.4. DISEÑO DE LA BASE DE DATOS	149
11.4.1. Modelo Entidad Relación.....	149
11.4.2. Descripción de la base de datos	150
11.4.3. Lenguaje de definición de datos.....	154
12. CONCLUSIONES	157
13. RECOMENDACIONES.....	159
14. BIBLIOGRAFÍA.....	160
15. GLOSARIO.....	162

LISTA DE TABLAS

	Pag
Tabla 1. Servicios de la empresa de telecomunicaciones de Pereira	19
Tabla 2. Personas entrevistadas	21
Tabla 3. Ingeniería de software	38
Tabla 4. Participantes del proyecto	78
Tabla 5. Beneficiarios del proyecto	79
Tabla 6. Objetivo 1 Generación de informes	80
Tabla 7. Objetivo 2 Gestión de usuarios	80
Tabla 8. Objetivo 3 Validar usuario	81
Tabla 9. Objetivo 4 Gestión de perfil de usuario y permisos.....	81
Tabla 10. Objetivo 5 Gestión de comandos	82
Tabla 11. Objetivo 6 Seguridad	82
Tabla 12. Requisito de información 01 Información de Usuario.....	83
Tabla 13. Requisito de información 02 Información de comandos.....	84
Tabla 14. Requisito de restricción de información 03 Relación entre usuarios y comandos	85
Tabla 15. Actor 01 Usuario.....	86
Tabla 16. Actor 02 Base de datos	86
Tabla 17. Actor 03 Interfaz de Usuario.....	87
Tabla 18. Documentación caso de uso 01 Validar Usuario	88
Tabla 19. Documentación caso de uso 02 Crear Usuario	92
Tabla 20. Documentación caso de uso 03 Modificar Usuario	96
Tabla 21. Documentación caso de uso 04 Consultar Usuario	100
Tabla 22. Documentación caso de uso 05 Eliminar Usuario	104
Tabla 23. Documentación caso de uso 06 Crear Comandos	107

Tabla 24. Documentación caso de uso 07 Modificar Comandos.....	110
Tabla 25. Documentación caso de uso 08 Consultar Comando.....	113
Tabla 26. Documentación caso de uso 09 Eliminar Comando.....	116
Tabla 27. Documentación caso de uso 10 Consultar Logs.....	120
Tabla 28. Requisito no funcional 01 Herramientas de desarrollo	122
Tabla 29. Requisito no funcional 02 Visualización	123
Tabla 30. Requisito no funcional 03 Requerimientos de interacción.....	124
Tabla 31. Requisito no funcional 04 Robustez.....	124
Tabla 32. Requisito no funcional 05 Control de acceso	125
Tabla 33. Evento interfaz Validar Usuario	130
Tabla 34. Evento interfaz Menú Principal.....	131
Tabla 35. Evento interfaz Asociar Usuario-Comando	133
Tabla 36. Evento interfaz Editar Comando.....	134
Tabla 37. Evento interfaz Comandos.....	135
Tabla 38. Evento interfaz New Comando.....	136
Tabla 39. Evento interfaz Log de Usuarios y Comandos	137
Tabla 40. Evento interfaz Usuarios.....	138
Tabla 41. Evento interfaz Crear Usuario	141
Tabla 42. Evento interfaz Editar Usuarios.....	142
Tabla 43. Evento interfaz Inicio de Conexión	144
Tabla 44. Evento interfaz Web para acceder al cli de los equipos Cisco	145
Tabla 45. Evento interfaz Dirección IP.....	147
Tabla 46. Evento interfaz Sesión cli	148
Tabla 47. Log	150
Tabla 48. Comandos.....	151
Tabla 49. Usuarios.....	152
Tabla 50. Comandos_Usuarios.....	153

LISTA DE FIGURAS

	Pag
Figura 1 Modelo de cascada	42
Figura 2 Modelo en espiral	44
Figura 3 Análisis de objetos.....	49
Figura 4 Definición de programa orientado a objetos	50
Figura 5 Modelo Cliente/Servidor de 2 capas	57
Figura 6 Modelo Arquitectura Java.....	67
Figura 7 Modelo Arquitectura PHP	68
Figura 8 Modelo Arquitectura FrameWork	69
Figura 9 Diagrama caso de uso Validar Usuario	88
Figura 10 Diagrama de secuencia Validar Usuario.....	90
Figura 11 Diagrama de colaboración Validar Usuario.....	91
Figura 12 Diagrama caso de uso Crear Usuario.....	92
Figura 13 Diagrama de secuencia Crear Usuario	94
Figura 14 Diagrama de colaboración Crear Usuario.....	95
Figura 15 Diagrama caso de uso Modificar Usuario	96
Figura 16 Diagrama de secuencia Modificar Usuario.....	98
Figura 17 Diagrama de colaboración Modificar Usuario	99
Figura 18 Diagrama caso de uso Consultar Usuario.....	100
Figura 19 Diagrama de secuencia Consultar Usuario	102
Figura 20 Diagrama de colaboración Consultar Usuario.....	103
Figura 21 Diagrama caso de uso Eliminar Usuario.....	103
Figura 22 Diagrama de secuencia Eliminar Usuario	105
Figura 23 Diagrama de colaboración Eliminar Usuario.....	106

Figura 24 Diagrama caso de uso Crear Comando.....	107
Figura 25 Diagrama de secuencia Crear Comando	108
Figura 26 Diagrama de colaboración Crear Comando	109
Figura 27 Diagrama caso de uso Modificar Comando	110
Figura 28 Diagrama de secuencia Modificar Comandos	111
Figura 29 Diagrama de colaboración Modificar Comando.....	112
Figura 30 Diagrama caso de uso Consultar Comando.....	113
Figura 31 Diagrama de secuencia Consultar Comando	114
Figura 32 Diagrama de colaboración Consultar Comando	115
Figura 33 Diagrama caso de uso Eliminar Comando.....	116
Figura 34 Diagrama de secuencia Eliminar Comando	118
Figura 35 Diagrama de colaboración Eliminar Comando	119
Figura 36 Diagrama caso de uso Consultar Logs.....	120
Figura 37 Diagrama de secuencia Consultar Logs	121
Figura 38 Diagrama de colaboración Consultar Logs	122
Figura 39 Diagrama de clases.....	126
Figura 40 Diseño arquitectónico	127
Figura 41 Diagrama de despliegue.....	128
Figura 42 Interfaz Validar Usuario 1.....	129
Figura 43 Interfaz Validar Usuario 2.....	130
Figura 44 Interfaz Menú Principal	131
Figura 45 Interfaz Asociar Usuario-Comando.....	132
Figura 46 Interfaz Editar Comando	133
Figura 47 Interfaz Comandos	135
Figura 48 Interfaz New Comando	136
Figura 49 Interfaz Log de Usuario y Comandos	137
Figura 50 Interfaz Usuarios	138
Figura 51 Interfaz Crear Usuario 1.....	139
Figura 52 Interfaz Crear Usuario 2.....	140

Figura 53 Interfaz Crear Usuario 3.....	141
Figura 54 Interfaz Editar Usuario	142
Figura 55 Interfaz Inicio de Conexión 1.....	143
Figura 56 Interfaz Inicio de Conexión 2.....	144
Figura 57 Interfaz Web para acceder al cli	145
Figura 58 Interfaz dirección IP 1.....	146
Figura 59 Interfaz dirección IP 2.....	147
Figura 60 Interfaz sesión cli	148
Figura 61 Diagrama Entidad Relación.....	149

INTRODUCCIÓN

El presente Proyecto está basado en la necesidad que se presentó en el área de soporte de la Empresa de Telecomunicaciones de Pereira y se convirtió en un requerimiento hacia el área de administración de la red de Core. El requerimiento exigía tener la posibilidad de prestar un mejor soporte hacia los usuarios de los diferentes servicios que presta la compañía.

Para darle respuesta al requerimiento y mejorar el servicio de soporte hacia los usuarios, se planteó la implementación de un sistema mediador de usuarios para equipos Cisco, el cual nos permitiera realizar las labores que ya se vienen realizando, pero con la posibilidad de otorgar muchos más privilegios de consulta y administración hacia los agentes de soporte, los cuales son los que atienden todas las solicitudes de soporte de todos los usuarios de la Empresa de Telecomunicaciones de Pereira. Adicionalmente también tener la posibilidad de guardar los registros de todos los soportes que realicen y así, realizar estadísticas acerca de si los agentes tienen los conocimientos necesarios para dar soporte a todos los usuarios.

De igual manera, mediante este sistema se tendrá control de los agentes que tendrán acceso a los diferentes equipos Cisco que operan en la Empresa de Telecomunicaciones de Pereira.

El requerimiento radica básicamente en la prestación de un mejor servicio de soporte hacia todos los usuarios, ya sean de Internet banda ancha, transmisiones de datos, telefonía IP, televisión IP, etc.

1. DEFINICIÓN DEL PROBLEMA

1.1 TÍTULO

Implementación de un sistema mediador para la interacción entre usuario y los equipos routers cisco, bajo un entorno web, en la empresa de UNE Telefónica de Pereira.

1.2. DESCRIPCIÓN DEL ÁREA DEL PROBLÉMA.

Debido a que la administración y monitoreo de los equipos de telecomunicaciones se encuentran en constante crecimiento dentro de una organización, en especial las orientadas hacia servicios de ISP (Proveedor de servicios de Internet), se ve la necesidad de automatizar ciertos procesos.

En este caso tenemos el proceso de monitoreo de consultas y cambios de configuración hacia los equipos CISCO de la empresa de Telecomunicaciones de Pereira.

Dichos equipos se encuentran distribuidos de acuerdo a la tecnología que manejan y el tipo de tráfico que procesan.

La distribución de dichos equipos es la siguiente:

- Equipos de acceso para la red de cable modem llamados CMTS (Servidor de terminales de cable modem) al cual tienen acceso el personal de soporte a manera de solo consultas y el personal de administración, el cual realiza los cambios de configuración que se requieran realizar en dichos equipos.

- Equipos de acceso para la red de DSL (Línea digital del suscriptor) llamados DSLAM (Multiplexor digital de acceso a la línea digital de abonado), a los cuales se tienen los mismos accesos mencionados anteriormente.

- Equipos de Core (Corazón de la red IP de la empresa) llamados XR-12410, los cuales son los que procesan todo el tráfico de toda la red de Telefónica y lo publican hacia Internet. A estos equipos solo se tiene acceso el personal de administración.

De acuerdo a lo mencionado anteriormente, se ve la necesidad de crear un programa mediador que nos permita realizar consultas y cambios de configuración básicos en los equipos Cisco sin la necesidad de interactuar directamente con los equipos por medio de la línea de comando sino por medio de un entorno Web.

Adicionalmente se tendrá una base de datos donde se almacenarán los registros de comandos utilizados por cada tipo de usuario que utilice dicha aplicación.

1.3. DIAGNÓSTICO DEL PROBLEMA.

La Empresa de Telecomunicaciones de Pereira tiene un área de soporte para todos los usuarios, ya sean de Internet, Transmisiones de datos, telefonía IP, televisión IP, a los cuales se tienen destinados agentes para cada tipo de servicio, pero con acceso a los mismos equipos Cisco que operan en la compañía.

Debido a la creciente demanda de usuarios nuevos en dicha entidad, se presentan todo tipo de inconvenientes, tales como:

- Incremento de usuarios por cada servicio, lo cual demanda más personal de soporte para los usuarios.

- La cantidad concurrente de accesos a los diferentes equipos por los cuales se les provee servicio a los usuarios.
- Capacitación del manejo de los equipos, al igual de los servicios que maneja la compañía.
- En la asignación de privilegios a los agentes de acuerdo al tipo de soporte que van a realizar (Internet, Voz IP, Televisión IP, Transmisiones de Datos)

De esta forma se establece que para la ejecución de las tareas en cuanto al soporte de los usuarios, se necesita un sistema que permita la realización de éstas de una manera efectiva, fácil y rápida.

Estos inconvenientes fueron detectados por medio de la administración de la red de Core, los cuales tienen entre sus funciones dar soporte a los agentes, cuando estos no tienen acceso a ciertos equipos, esto es crucial para sus labores.

En la actualidad, el área de soporte de la Empresa de Telecomunicaciones de Pereira está en la capacidad de realizar soporte técnico en los siguientes servicios que la compañía entrega a la comunidad:

Tabla 1. Servicios de la Empresa de Telecomunicaciones de Pereira.

Servicios	Cantidad de agentes por turno
Internet Banda Ancha	64
Internet Dedicado y Transmisiones de Datos	16
Telefonía IP	16
Televisión IP	16

Fuente: los autores

En estos servicios se ubican los diferentes agentes que estén con la capacidad y conocimientos necesarios para lograr un efectivo soporte a todos los usuarios que los requieran en el momento que estos lo necesiten.

Preparación, realización y análisis de entrevista para el estudio del problema

Estudio del problema:

Se realizaron visitas al área de soporte de la Empresa de Telecomunicaciones de Pereira con el objetivo de conocer el funcionamiento de las diversas labores de soporte de acuerdo a los diferentes servicios que la compañía ofrece a la comunidad.

Las personas entrevistadas fueron las siguientes:

Carolina Paque, encargada de coordinar a todo el personal de soporte de la Empresa de Telecomunicaciones de Pereira.

Leonardo Valero, encargado de verificar la calidad de prestación del servicio de soporte hacia con los usuarios.

Se determinaron los objetivos de la entrevista:

El objetivo principal de la entrevista fue conocer detalladamente cuales son los problemas actuales que tiene la Empresa de Telecomunicaciones de Pereira en el área de soporte técnico hacia los usuarios

Como interactúan los agentes con los equipos de la red de acceso y proponer una herramienta informática como solución al problema planteado.

Así proponer una herramienta informática más efectiva para que los técnicos de soportes puedan interactuar y mejorar el rendimiento del soporte.

Planificar las entrevistas:

Tabla 2. Personas entrevistadas.

Nombre	Fecha	Hora	Lugar	Duración
Carolina Paque	Miércoles 15 de Agosto de 2007	2:00 pm	Oficina Agregación & Backbone de la Empresa de Telecomunicaciones de Pereira.	1 Hora
Leonardo Valero	Jueves 29 de Agosto de 2007	10:00 am	Oficina Agregación & Backbone de la Empresa de Telecomunicaciones de Pereira.	1 Hora

Fuente: los autores

Análisis de las entrevistas:

Conclusiones

1. Se obtuvo información básica sobre la interacción que tienen los agentes de soporte con los usuarios de la Empresa de Telecomunicaciones de Pereira.
2. Se conoció la forma en que los agentes de soporte interactúan con todos los equipos tanto de acceso como de Core de la Empresa de Telecomunicaciones de Pereira.
3. Se vio la necesidad de implementar un sistema que permitiera a los agentes de soporte de la Empresa de Telecomunicaciones de Pereira tener más posibilidades de realizar diagnósticos y soluciones hacia los usuarios mediante los equipos a los cuales estos acceden.

Descripción

La Empresa de Telecomunicaciones de Pereira tiene un área de soporte hacia todos los usuarios que tengan alguno de los servicios que dicha compañía les este otorgando.

La empresa tiene el compromiso de prestar este servicio de soporte a todos sus usuarios, teniendo muy en cuenta la eficacia de dicha labor ya que esto es fundamental con lo que tiene que ver con la calidad del servicio.

Actualmente, el área de soporte realiza las consultas, diagnósticos y soluciones de manera directa con los equipos que manejen los diferentes servicios, con el pequeño inconveniente de no poseer en la mayoría de las veces los privilegios necesarios para realizar una buena labor para con los usuarios.

Debido a esto muchas veces se ven en la necesidad de recurrir a personal del área de agregación para que les presten la ayuda que ellos necesiten haciendo más lenta la atención hacia el usuario final.

2. JUSTIFICACIÓN

La idea de este proyecto surge debido a la necesidad de implementar un sistema mediador de usuarios para realizar diagnósticos y soportes a todos los usuarios de los diferentes servicios ofrecidos por la Empresa de Telecomunicaciones de Pereira.

Este sistema mediador está diseñado para brindar una conexión hacia los equipos Cisco que operan en la red de **Core** mediante un entorno gráfico bastante similar al que se puede visualizar por medio de un programa que permita realizar conexiones de **Telnet** ó de **SSH**, permitiendo de está manera un ambiente más amigable para realizar las labores de soporte.

El desarrollo de este proyecto tiene unas implicaciones hacia todo el personal de soporte debido al cambio que se debe asumir, por medio de este sistema podemos definir de manera más específica, los privilegios que van a tener cada uno de los agentes.

Este sistema también está diseñado para dar cuenta de todos los comandos utilizados por los agentes durante las sesiones que tengan durante el día, ya que la información podrá ser almacenada en una base de datos, permitiendo así la posibilidad de sacar estadísticas de los comandos más utilizados, conocimientos de los agentes en cuanto al soporte que le otorgan a un usuario.

Se justifica el sistema por las diferentes características que podemos aprovechar de él y sobre todo por el mejor servicio que se le podrá otorgar hacia los usuarios de la Empresa de Telecomunicaciones de Pereira.

Hay que tener muy en cuenta que una Empresa prestadora de servicios públicos, está en la obligación de tener a todos sus usuarios lo más satisfechos posible, no solo con el servicio que se le este otorgando, sino también al momento en que alguno de sus servicios adquiridos presenten cualquier tipo de dificultad.

3. OBJETIVOS.

3.1. OBJETIVO GENERAL.

Implementar un sistema orientado a interactuar por medio de un entorno Web hacia los equipos Routers CISCO para la realización de consultas y configuraciones básicas en la Empresa de Telecomunicaciones de Pereira.

3.2 OBJETIVOS ESPECÍFICOS

- Diagnosticar las necesidades técnicas que se requieren para la realización de la aplicación.
- Adecuar un equipo mediante software GNU, el cual tendrá conexión con todos los equipos Routers CISCO a los cuales se realizarán las consultas.
- Facilitar a los agentes de soporte los privilegios necesarios para realizar un correcto diagnóstico de las fallas que se presenten con los usuarios y su respectiva solución.
- Enfocar la aplicación hacia el área del ISP (Proveedor de Servicios de Internet) a la Empresa de Telecomunicaciones de Pereira.

4. DELIMITACIÓN DEL PROBLEMA

El sistema mediador de usuarios está dirigido hacia el área del ISP (Proveedor de Servicios de Internet) con el principal motivo de satisfacer las necesidades de los usuarios al momento de solicitar un soporte técnico cuando se les presente algún tipo de inconveniente con el servicio que tenga contratado con la Empresa de Telecomunicaciones de Pereira.

5. METODOLOGÍA

Este proyecto está basado en un principio de investigación aplicada (Teórico - Práctica) orientada al área de las telecomunicaciones de la Empresa de Telecomunicaciones de Pereira.

También es importante resaltar la necesidad de determinar qué elementos intervienen en el sistema a desarrollar, tales como sus relaciones, su estructura, un detalle acerca de las diferentes funcionalidades, las cuales nos dan una descripción más clara acerca del sistema que se va a realizar, qué funcionalidades va a desempeñar y el comportamiento que este tendrá. Para esto, el sistema será enfocado desde los siguientes puntos de referencia: El funcional y el dinámico

En las primeras etapas de este proyecto, se utilizó la observación directa y la interacción con el personal involucrado en el área de soporte hacia los usuarios de la Empresa de Telecomunicaciones de Pereira, este proceso permitió realizar la recolección de información para así lograr el diagnóstico y formulación del proyecto de investigación con el cual se apunta a resolver las dificultades que se estaban presentando en proceso de soporte técnico hacia los usuarios.

Para conocer la manera en que funcionaba el área de soporte de la Empresa de Telecomunicaciones de Pereira, se realizaron diversas encuestas con el personal que labora en el área. Mediante esta información recolectada, se pudo realizar la viabilidad y diseño que tendría el software a implementar.

Se utilizaran herramientas de software GNU como PHP para la creación del entorno Web, UML para la ingeniería de requerimientos, SQL para la base de datos, Java para el desarrollo del software, seguridad con protocolos SNMP¹, listas de acceso por IP² y listas de acceso por MAC³, y para la autenticación se maneja usuario y contraseña con algoritmos MD5⁴.

5.1. RESEÑA DEL MARCO INSTITUCIONAL.

Telefónica de Pereira es una empresa ubicada en el centro de la ciudad de Pereira, que cuenta con una oferta de productos de telecomunicaciones y calidad en la prestación de servicios.

Es una empresa de economía mixta por acciones, que ha acompañado a la ciudad de Pereira desde 1929, empujando su crecimiento y desarrollo y actualmente cuenta con aproximadamente 48.000 usuarios de Internet banda ancha.

¹ http://es.wikipedia.org/wiki/Simple_Network_Management_Protocol

² http://es.wikipedia.org/wiki/Direcci%C3%B3n_IP

³ http://es.wikipedia.org/wiki/Direcci%C3%B3n_MAC

⁴ <http://es.wikipedia.org/wiki/MD5>

6. MARCO CONCEPTUAL.

6.1. SOFTWARE.

Se refiere al equipamiento lógico o soporte lógico de una computadora digital, y comprende el conjunto de los componentes lógicos necesarios para hacer posible la realización de tareas específicas; en contraposición a los componentes físicos del sistema, llamados hardware. En otras palabras, es un conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Probablemente la definición más formal de software es la atribuida a la “IEEE”^{*} en su estándar 729: «la suma total de los programas de cómputo, procedimientos, reglas documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo». Bajo esta definición, el concepto de software va más allá de los programas de cómputo en sus distintas formas: código fuente, binario o ejecutable, además de su documentación: es decir, todo lo intangible.

El término «software» fue usado por primera vez en este sentido por “John W. Tukey en 1957”^{**}. En las ciencias de la computación y la ingeniería de software, el

^{*} www.ieee.org/
IEEE, (The Institute of Electrical and Electronics Engineers Inc.: Organización Responsable de dictar los estándares internacionales.)

^{**} http://es.wikipedia.org/wiki/John_W._Tukey

software es toda la información procesada por los sistemas informáticos: programas y datos. El concepto de leer diferentes secuencias de instrucciones de la memoria de un dispositivo para controlar cálculos fue inventado por “Charles Babbage”^{*} como parte de su máquina diferencial. La teoría que forma la base de la mayor parte del software moderno fue propuesta por vez primera por “Alan Turing”^{**}, Los números computables, con una aplicación al problema de decisión.”⁵

- Tipologías.

“Si bien esta distinción es, en cierto modo, arbitraria, y, a veces, difusa y confusa, se puede distinguir al software de la siguiente forma:

Software, es la parte que permite funcionar al hardware. Su objetivo es aislar tanto como sea posible al programador de aplicaciones de los detalles del computador particular que se use, especialmente de las características físicas de la memoria, dispositivos de comunicaciones, impresoras, pantallas, teclados, etcétera. Incluye entre otros: Sistemas operativos, Controladores de dispositivo, Herramientas de diagnóstico, Servidores, Sistemas de ventanas, Utilidades.”⁶

John Wilder Tukey fue un estadístico nació en Massachusetts. Tukey obtuvo un Bachiller en Artes en 1936 y una Maestría en Ciencias en 1937, ambas en química, en la Universidad de Brown, antes de trasladarse a la universidad de Princeton recibió un Doctorado en Matemáticas.

^{*} http://es.wikipedia.org/wiki/Charles_Babbage

Charles Babbage (Gran Bretaña, 26 de diciembre de 1791 - 18 de octubre de 1871) fue un matemático británico y científico de la computación. Diseñó e implementó una máquina a vapor, también diseñó, pero nunca construyó, la máquina analítica para ejecutar programas de tabulación o computación.

^{**} http://es.wikipedia.org/wiki/Alan_Turing

Alan Mathison Turing (23 de junio de 1912 - 7 de junio de 1954). Fue un matemático, informático teórico, criptógrafo y filósofo inglés. Es considerado uno de los padres de la Ciencia de la computación siendo el precursor de la informática moderna. Proporcionó una influyente formalización de los conceptos de algoritmo y computación: la máquina de Turing.

⁵ Introducción a la programación

<http://www.slideshare.net/questdf1874/introduccion-a-la-programacion>

⁶Software y Hardware

<http://nastik.blogcindario.com/2007/10/00019-software-y-hardware.html>

“Software de programación, que proporciona herramientas para ayudar al programador a escribir programas informáticos y a usar diferentes lenguajes de programación de forma práctica. Incluye entre otros: Editores de texto, Compiladores, Intérpretes, Enlazadores, Depuradores.

Los entornos integrados de desarrollo (IDE) agrupan estas herramientas de forma que el programador no necesite introducir múltiples comandos para compilar, interpretar, depurar, etcétera, gracias a que habitualmente cuentan con una interfaz gráfica de usuario (GUI) avanzada.

Software de aplicación, que permite a los usuarios llevar a cabo una o varias tareas más específicas, en cualquier campo de actividad susceptible de ser automatizado o asistido, con especial énfasis en los negocios. Incluye entre otros:

- Aplicaciones de automatización industrial.
- Aplicaciones ofimáticas.
- Software educativo.
- Software médico.
- Bases de datos.
- Videojuegos.
- Archivos de datos”⁷

⁷ Software y Hardware
<http://nastik.blogcindario.com/2007/10/00019-software-y-hardware.html>

6.1.1. Formas del software.

“El software adopta varias formas en distintos momentos de su ciclo de vida:

- Código fuente: escrito por programadores. Contiene el conjunto de instrucciones destinadas a la computadora.
- Código objeto: resultado del uso de un compilador sobre el código fuente. Consiste en una traducción de éste último. El código objeto no es directamente inteligible por el ser humano, pero tampoco es directamente entendible por la computadora. Se trata de una representación intermedia del código fuente.
- Código ejecutable: resultado de enlazar uno o varios fragmentos de código objeto. Constituye un archivo binario con un formato tal que el sistema operativo es capaz de cargarlo en la memoria de una computadora, y proceder a su ejecución. El código ejecutable es directamente inteligible por la computadora.”⁸

6.1.2. El proceso de creación de software.

“El proceso de creación de software es materia de la ingeniería del software. Es un proceso complejo que involucra diversas tareas de gestión y desarrollo. Como resumen de las etapas para la creación de un software, se pueden mencionar:

- Análisis
- Desarrollo

⁸ Software y Hardware
<http://nastik.blogcindario.com/2007/10/00019-software-y-hardware.html>

- Construcción
- Pruebas (unitarias e integradas)
- Paso a Producción”⁹

6.1.3. Seguridad.

La seguridad informática es el área que se enfoca en la protección de la infraestructura conceptual y todo lo relacionado con esta (incluyendo la información contenida). Para ello existen una serie de estándares, protocolos, métodos, reglas, herramientas y leyes concebidas para minimizar los posibles riesgos a la infraestructura o a la información la seguridad informática comprende software, base de datos, archivos y todo lo que signifique un riesgo.

6.2. DATOS DE SOFTWARE.

“¿Qué es?

El software de computadora es el producto que los ingenieros de software construyen y después mantienen en el largo plazo. Incluye los programas que se ejecutan dentro de una computadora de cualquier tamaño y arquitectura, el contenido que se presenta conforme los programas se ejecutan y los documentos, tanto físicos como virtuales, que engloban todas las formas de medios electrónicos.”¹⁰

⁹ Qué es software

<http://marketingnet.com/net/software/que-es-software.html>

¹⁰ Ingeniería del software un enfoque práctico, Sexta Edición. Roger S. Pressman, Editorial Mc Graw Hill. 2005.

“¿Quién lo hace?”

Los ingenieros de software lo construyen y lo mantienen, y casi todos en el mundo industrializado lo usan de manera directa o indirecta

¿Por qué es importante?

Porque afecta de forma muy cercana todos los aspectos de la vida y se ha vuelto omnipresente en el comercio, la cultura y las actividades cotidianas.

¿Cuáles son los pasos?

El software de computadora se construye de la misma forma que cualquier producto de éxito mediante la aplicación de un proceso que conduzca a un resultado de alta calidad que satisfaga las necesidades de la gente que usará el producto. Se aplica un enfoque de Ingeniería de software.

¿Cuál es el producto obtenido?

Desde el punto de vista del ingeniero de software, el producto obtenido lo forman los programas, el contenido (datos) y los documentos que constituyen el software. Pero desde el enfoque de usuario, el producto obtenido es la información resultante que de alguna manera mejora el mundo del usuario.

6.3. DEFINICIONES DE INGENIERÍA DE SOFTWARE.

“Es la disciplina tecnológica y administrativa dedicada a la producción sistemática de productos de Software, que son desarrollados y modificados a tiempo y dentro de un presupuesto definido (Fairley).”¹¹

6.4. ACTIVIDADES DE DESARROLLO DE INGENIERÍA DE SOFTWARE.

“Las actividades de desarrollo manejan la complejidad mediante la construcción de modelos de los dominios del problema o del sistema.

Las actividades de desarrollo incluyen:

- Obtención de requerimientos.
- Análisis.
- Diseño del sistema.
- Diseño de objetos.
- Implementación.”¹².

¹¹ Modelos y estándares de calidad de software
http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/garcia_r_ci/capitulo2.pdf

¹² Ingeniería de software orientado a objetos, Primera edición. Bernd Bruegge – Allen H. Dutoit. Editorial Prentice Hall. 2002

6.4.1. Obtención de requerimientos.

“Durante la obtención de requerimientos el cliente y los desarrolladores definen el propósito del sistema. El resultado de esta actividad es una descripción del sistema en términos de actores y casos de uso.

6.4.2. Análisis.

“Durante el análisis, los desarrolladores tratan de producir un modelo del sistema que sea correcto, completo, consistente, claro, realista y verificable.

Los desarrolladores transforman los casos de uso producidos durante la obtención de requerimientos en un modelo de objeto que describa por completo al sistema. Durante esta actividad, los desarrolladores descubren ambigüedades e inconsistencias en el modelo de casos de uso y las resuelven con el cliente.

6.4.3. Diseño del sistema.

“Durante el diseño del sistema, los desarrolladores definen los objetivos de diseño del proyecto y descomponen el sistema en subsistemas más pequeños que pueden realizar los equipos individuales. Los desarrolladores también seleccionan estrategias para la construcción del sistema, como la plataforma de hardware y software en la que ejecutará el sistema, la estrategia de almacenamiento de datos, persistentes, el flujo de control global, la política de control de acceso y el manejo de las condiciones de frontera. El resultado de sistema es una descripción clara de cada una de estas estrategias, una descomposición en subsistemas y un

diagrama de organización que representa el mapeo en hardware y software del sistema.”¹³

6.4.4. Diseño de objetos.

“Durante el diseño de objetos, los desarrolladores definen objetos personalizados para cubrir el hueco entre el modelo de análisis y la plataforma de hardware y software definida durante el diseño del sistema. Esto incluye definir con precisión los objetos e interfaces de subsistemas, la selección de componentes hechos, la reestructuración del modelo de objeto para lograr los objetivos de diseño, tales como extensibilidad o comprensión.

6.4.5. Implementación.

“Durante la implementación, los desarrolladores traducen el modelo de objetos en código fuente. Esto incluye la implementación de los atributos y métodos de cada objeto y la integración de todos los objetos de forma tal que funcionen como un solo sistema. La actividad de implementación cubre el hueco entre el modelo de diseño de objetos detallado y el conjunto completo de archivos de código fuente que pueden ser compilados juntos.

¹³ Ingeniería de software orientado a objetos, Primera edición. Bernd Bruegge – Allen H. Dutoit. Editorial Prentice Hall. 2002

6.5. PRODUCTO DE SOFTWARE.

“Es el conjunto de programas (fuentes y ejecutables), procedimientos, reglas y documentación posible asociada, así como los datos pertenecientes a la operación del sistema.”¹⁴

6.6. PROCESOS DE SOFTWARE.

“Es el conjunto de actividades técnicas y administrativas necesarias para la fabricación de un producto de software.

La madurez de una organización es proporcional a su capacidad para poner en práctica procesos de software que produzcan los resultados previstos.”¹⁵

6.7. LA INGENIERÍA DE SOFTWARE.

Es una disciplina pragmática que utiliza:

Tabla 3. Ingeniería de software.

Ciencias de la Computación	Fundamentos Científicos
Administración	Fundamentos para administración de proyectos
Economía	Fundamentos para estima de recursos y control de costos
Sicología y técnicas de comunicación	Fundamentos para trabajo interdisciplinario y buena comunicación interpersonal entre grupo

Fuente: los autores

¹⁴ es.wikipedia.org/wiki/Software

¹⁵ www.acis.org.co/index.php?id=547

6.8. DEFINICIÓN DE MODELO DE CICLO DE VIDA.

“Un modelo de ciclo de vida de software es una vista de las actividades que ocurren durante el desarrollo de software, intenta determinar el orden de las etapas involucradas y los criterios de transición asociadas entre estas etapas.

Así, los modelos por una parte suministran una guía para los ingenieros de software con el fin de ordenar las diversas actividades técnicas en el proyecto, por otra parte suministran un marco para la administración del desarrollo y el mantenimiento, en el sentido en que permiten estimar recursos, definir puntos de control intermedios, monitorear el avance, etc.”¹⁶

6.8.1. Etapas en el ciclo.

Se puede ver a grandes rasgos, una pequeña descripción de las etapas con las que se puede contar a lo largo del ciclo de vida del software; una vez delimitadas de cierta manera las etapas, habrá que establecer la forma en que éstas se afrontan.

6.8.1.1. Expresión de necesidades.

Esta etapa tuvo como objetivo la consecución de un primer documento, en que quedaron reflejados los requerimientos y funcionalidades del software, por tal razón se ofrecen al usuario la solución desde las ciencias informáticas, teniendo en cuenta las necesidades de la Empresa de Telecomunicaciones de Pereira en la relación Docencia-Servicio.

¹⁶ www.mundodescargas.com/.../decargar_el-ciclo-de-vida-del-software.pdf

6.8.1.2. Especificaciones.

El paso siguiente es formalizar los requerimientos; el documento obtenido en la etapa anterior se tomó como punto de partida para esta fase.

Por medio de esta etapa se obtuvo un nuevo documento que permitió definir con más precisión la herramienta informática requerida por la Empresa de Telecomunicaciones de Pereira.

6.8.2. Algunos modelos de ciclos de vida.

6.8.2.1. El modelo cascada.

“El primer modelo de proceso de desarrollo de software que se publicó se derivó de procesos de ingeniería de sistemas más generales (Royce, 1970). Este modelo se muestra en la figura debido a la cascada de una fase a otra, dicho modelo se conoce como modelo en cascada o como ciclo de vida de software. Las principales etapas de este modelo se transformaron en actividades fundamentales de desarrollo:

1. Análisis y definición de requerimientos: los servicios, restricciones y metas del sistema se definen a partir de las consultas con los usuarios. Entonces, se definen en detalle y sirven como especificación del sistema.

2. Diseño del sistema y del software: El proceso del diseño del sistema divide los requerimientos en sistemas hardware o software. Establece una arquitectura completa del sistema. El diseño del software identifica y define las abstracciones fundamentales del sistema software y sus relaciones.

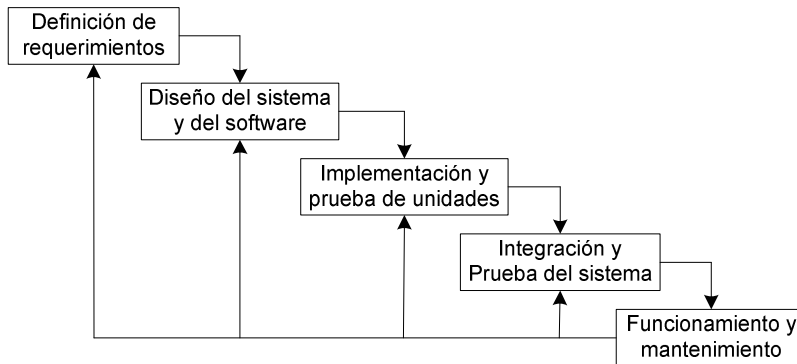
3. Implementación y prueba de unidades: Durante esta etapa, el diseño del software se lleva a cabo como un conjunto o unidades de programas. La prueba de unidades implica verificar que cada una cumpla su especificación.

4. Integración y prueba del sistema: Los programas o las unidades individuales de programas se integran y prueban como un sistema completo para asegurar que se cumplan los requerimientos del software. Después de la pruebas, el sistema software se entrega al cliente.

5. Funcionamiento y mantenimiento: Por lo general (aunque no necesariamente), ésta es la fase más larga del ciclo de vida. El sistema se instala y se pone en funcionamiento práctico. El mantenimiento implica corregir errores no descubiertos en las etapas anteriores del ciclo de vida, mejorar la implementación de las unidades del sistema y resaltar los servicios del sistema una vez que se descubren nuevos requerimientos”¹⁷.

¹⁷ Ingeniería del software, Séptima edición. IAN SOMMERVILLE. Editorial PEARSON Addison Wesley. 2006

Figura 1. Modelo de cascada.



Fuente: Ingeniería del software, Séptima edición. IAN SOMMERVILLE. Editorial PEARSON Addison Wesley. 2006

6.8.2.2. Modelo en espiral.

“El modelo en espiral del proceso de software fue originalmente propuesto por Boehm (Boehm, 1988). Más que representar el modelo del software como una secuencia de actividades con retrospectiva de una actividad a otra, se representa como un espiral. Cada ciclo en la espiral representa una fase del proceso del software. Así, el ciclo más interno podría referirse a la viabilidad del sistema, el siguiente ciclo a la definición de requerimientos, el siguiente ciclo al diseño del sistema, y así sucesivamente. Cada ciclo en espiral se divide en cuatro sectores:

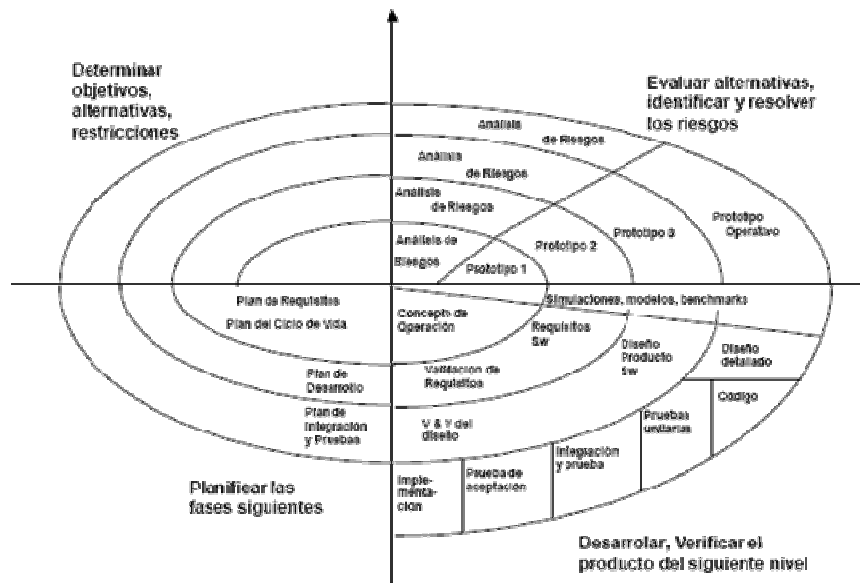
1. Definición de objetivos: Para esta fase del proyecto se definen los objetivos específicos. Se identifican las restricciones del proceso y del producto, y se traza un plan detallado de gestión. Se identifican los riesgos del proyecto. Dependiendo de estos riesgos, se plantean alternativas.
2. Evaluación y reducción de riesgos: Se lleva a cabo un análisis detallado para cada uno de los riesgos del proyecto identificados. Se definen los

pasos para reducir dichos riesgos. Por ejemplo. Si existe el riesgo de tener requerimientos inapropiados, se puede desarrollar un prototipo del sistema.

3. Desarrollo y validación: Después de la evaluación de riesgos, se elige un modelo para desarrollar el sistema. Por ejemplo, si los riesgos de la interfaz de usuario son dominantes, un modelo de desarrollo apropiado podría ser la construcción de prototipos evolutivos. Si los riesgos de seguridad son la principal consideración, un desarrollo basado en transformaciones formales podría ser el más apropiado, y así sucesivamente. El modelo en cascada puede ser el más apropiado para el desarrollo si el mayor riesgo identificado es la integración de subsistemas.
4. Planificación: El proyecto se revisa y se toma la decisión de si se debe continuar con un ciclo posterior de la espiral. Si se decide continuar, se desarrollan los planes para la siguiente fase del proyecto.”¹⁸

¹⁸ Ingeniería del software, Séptima edición. IAN SOMMERVILLE. Editorial PEARSON Addison Wesley. 2006

Figura 2. Modelo en espiral



Fuente: Ingeniería del software, Séptima edición. IAN SOMMERVILLE. Editorial PEARSON Addison Wesley. 2006

6.8.3. Descripción del ciclo de vida orientado a objetos según grady booch.

“Según la experiencia de” Booch”^{*}, todos los sistemas orientados a objeto que analizó y que terminaron con éxito tienen dos características en común:

^{*} http://es.wikipedia.org/wiki/Grady_Booch

Grady Booch (nació el 27 de febrero de 1955) es un diseñador de software, un metodologista de software y entusiasta de diseño de patrones. Es director científico de Rational Software (ahora parte de IBM) y editor de una serie de Benjamín/Cummings. En 1995 se recibió como miembro de la Asociación de Maquinaria Computacional (ACM). Fue nombrado socio de IBM en 2003.

Booch es mejor conocido por el desarrollo del Lenguaje Unificado de Modelado con Ivar Jacobson y James Rumbaugh. También desarrolló el método Booch de desarrollo de software, el que presenta en su libro, Análisis y Diseño Orientado a Objetos. Él aconseja la adición de más clases para simplificar códigos complejos.

Obtuvo su licenciatura en 1977 en la Academia de la Fuerza Aérea de los Estados Unidos y un máster en ingeniería eléctrica en 1979 de la Universidad de California, Santa Barbará.

- La existencia de una fuerte visión arquitectónica con buena integridad conceptual
- La aplicación de un ciclo de vida del desarrollo bien dirigido, iterativo e Incremental: cuando dice bien dirigido lo dice en el sentido de que el proceso pueda controlarse y medirse, aunque no debe ser tan rígido como para que limite la creatividad.

Según sus observaciones, cualquier actividad humana que requiera creatividad e innovación demanda un proceso incremental e iterativo que se apoya en la experiencia, inteligencia y talento de todos los miembros del equipo. El primer problema del análisis orientado a objetos es un problema de clasificación (creación de clases), y esa clasificación inteligente es un trabajo intelectualmente difícil de por sí y la parte más difícil de un diseño orientado a objetos. La mejor forma de abordarlo es por un proceso iterativo e incremental.

De esta forma la descomposición orientada a objetos reduce en gran medida el riesgo que representa construir sistemas de software complejos, porque están diseñados para evolucionar de forma incremental partiendo de sistemas más pequeños en los que ya se tiene confianza.

El diseño orientado a objetos entonces, es fundamentalmente diferente a los enfoques de diseño estructurado tradicionales, requiere un modo distinto de pensar acerca de la descomposición, y produce arquitecturas software muy alejadas del dominio de la cultura del diseño estructurado.

Esas diferencias Booch las atribuye a que los métodos estructurados se basan en programación estructurada y los orientados a objetos en programación orientada a objetos Booch desaconseja, a título personal, el uso del análisis estructurado

como punto de partida para el diseño orientado a objetos. Dice que falla a menudo cuando el desarrollador no es capaz de resistir la incitación al recaer en el abismo de la mentalidad del diseño estructurado.

Otro peligro habitual es que analistas tiendan a escribir diagramas de flujos de datos que reflejan un diseño en vez de un modelo esencial del problema.

En el proceso orientado a objetos, las fronteras entre el análisis y diseño son difusas aunque el objetivo principal de ambos es bastante diferente. En el análisis se descubren las clases y objetos, y en el diseño se plantean las abstracciones y mecanismos que proporcionan el comportamiento.

Para el análisis, Booch recomienda fuertemente el análisis de casos de uso y la técnica de escenarios para guiar el proceso de identificación de clases y objetos. El análisis se desarrolla entonces como un estudio de cada escenario fundamental, usando técnicas de presentación.

En cuanto al diseño, los mecanismos son el alma del mismo. Durante el proceso de diseño el desarrollador debe considerar no sólo el diseño de clases individuales, sino también cómo trabajan juntas las instancias de esas clases. Nuevamente el uso de escenarios dirige este proceso.

Así, los modelos por una parte suministran una guía para los ingenieros de software con el fin de ordenar las diversas actividades técnicas en el proyecto, por otra parte suministran un marco para la administración del desarrollo y el

mantenimiento, en el sentido en que permiten estimar recursos, definir puntos de control intermedios, monitorear el avance, etc.”¹⁹

6.9. ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS.

“Es un enfoque de la ingeniería de software que modela un sistema como un grupo de objetos que interactúan entre sí. Este enfoque representa un dominio en términos de conceptos compuestos por verbos y sustantivos, clasificados de acuerdo a su dependencia funcional.

En éste método de análisis y diseño se crea un conjunto de modelos utilizando una notación acordada como, por ejemplo, el lenguaje unificado de modelado “(UML)”*. Análisis y Diseño Orientado a Objetos aplica técnicas de modelado de objetos para analizar los requerimientos para un contexto, por ejemplo, un sistema de negocio, un conjunto de módulos de software y para diseñar una solución para mejorar los procesos involucrados. No está restringido al diseño de programas de computadora, sino que cubre sistemas enteros de distinto tipo.

Las metodologías de análisis y diseño más modernas son casos de uso guiados a través de requerimientos, diseño, implementación, pruebas, y despliegue.

¹⁹ generacionesweb.com/.../Programacion-Orientada-Objetos.jsp

* <http://es.wikipedia.org/wiki/UML>

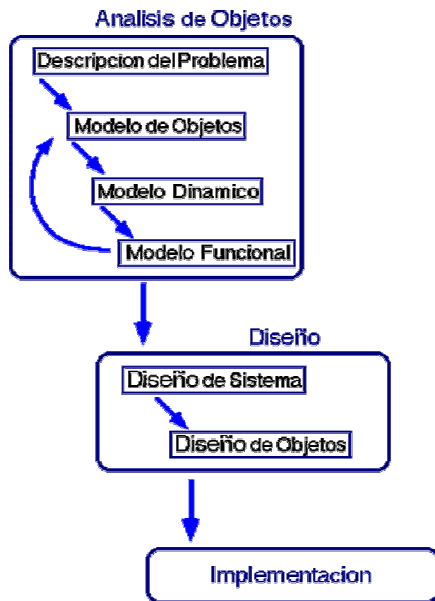
Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

El lenguaje unificado de modelado se ha vuelto el lenguaje de modelado estándar usado en análisis y diseño orientado a objetos.

Diseño orientado a objetos: es una fase de la metodología orientada a objetos para el desarrollo de Software. Su uso induce a los programadores a pensar en términos de objetos, en vez de procedimientos, cuando planifican su código. Un objeto agrupa datos encapsulados y procedimientos para representar una entidad. La 'interfaz del objeto', esto es las formas de interactuar con el objeto, también es definida en esta etapa. Un programa orientado a objetos es descrito por la interacción de esos objetos. El diseño orientado a objetos es la disciplina que define los objetos y sus interacciones para resolver un problema de negocio que fue identificado y documentado durante el análisis orientado a objetos.”²⁰

²⁰ [es.wikipedia.org/.../Análisis_y_diseño_orientado_a_objetos](https://es.wikipedia.org/wiki/Análisis_y_diseño_orientado_a_objetos)

Figura 3. Análisis de objetos.



Fuente: <http://pisuerga.inf.ubu.es/lisi/Docencia/TFC/ITIG/icruzadn/Memoria/23.htm>

6.10. CONCEPTOS DE LA PROGRAMACIÓN ORIENTADA A OBJETOS

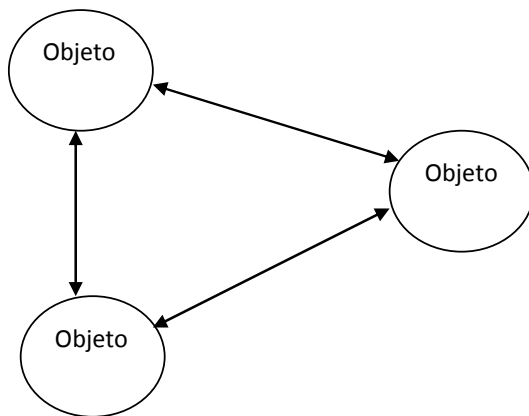
“A diferencia de la programación tradicional la programación orientada a objetos define una estructura de más alto nivel llamada objeto, que ofrece dos ventajas sobre la programación tradicional:

- a. La primera es permitir al programador que organice su programa de acuerdo con abstracciones de más alto nivel, siendo éstas más cercanas a la manera de pensar de la gente. En otras palabras los objetos son las unidades de representación de las aplicaciones, por ejemplo, cuentas de banco, reservación de vuelos, etc.

- b. La segunda es que los datos globales desaparecen, siendo estos junto con las funciones parte interna de los objetos. Por lo tanto, cualquier cambio en la estructura de alguno de los datos solo debiera afectar las funciones definidas en ese mismo objeto y no en los demás.

En general, un programa orientado a objetos se define exclusivamente en términos de objetos y sus relaciones, como se muestra en la figura.”²¹

Figura 4. Definición de programa orientado a objetos.



Fuente: Ingeniería de software orientada a objetos con UML, java e internet. Alfredo Weitzenfeld. Editorial Thomson. 2005.

²¹ Ingeniería de software orientada a objetos con UML, java e internet. Alfredo Weitzenfeld. Editorial Thomson. 2005.

6.11. INGENIERÍA DE REQUERIMIENTOS.

“La construcción de requerimientos incluye un análisis de factibilidad del sistema deseado, la adquisición y análisis de las necesidades de las personas interesadas, la creación de una descripción precisa de lo que el sistema debe y no debe hacer, junto con cualquier restricción en su operación e implementación, y la validación de esta descripción o especificación por las personas interesadas.”²²

6.11.1. Beneficios.

- “Acuerdo entre desarrolladores, clientes y usuarios sobre el trabajo que se debe realizar y los criterios de aceptación del sistema
- Tener elementos básicos para mejores estimaciones
- Mejoras en atributos de software
- Alcanzar los objetivos con los recursos justos (pocas omisiones, menos malentendidos y menos cambios)”

6.11.2. ¿Por qué Ingeniería de Requerimientos?

“Cada aplicación de software tiene usuarios que confían en ella para mejorar sus vidas. El tiempo que se pasa entendiendo las necesidades de los usuarios es una inversión de alto nivel en el éxito del proyecto.”

²² Ingeniería de software orientada a objetos con UML, java e internet. Alfredo Weitzenfeld. Editorial Thomson. 2005.

6.12. DEFINICIÓN DEL MODELO DE MARCO LÓGICO.

“El ML, modelo de Marco Lógico, es un instrumento para la planificación de proyectos orientada a objetivos. El método también puede usarse para análisis, valoración, seguimiento y evaluación de proyectos.

La utilización del método depende del rol de los usuarios y de sus necesidades.

Propósito del modelo de marco lógico.

El Marco Lógico se usa para:

1. Identificar problemas y necesidades en un sector de la sociedad.
2. Facilitar la selección y priorización entre proyectos.
3. Planificar e implementar proyectos de desarrollo efectivamente.
4. Seguimiento y evaluación de proyectos de desarrollo.

El uso del método depende del rol y de las necesidades de sus usuarios.

El modelo de marco lógico fue desarrollado durante los años 60 y ha sido difundido ampliamente por todo el mundo desde la década del 70.

En la actualidad es usado por empresas privadas, municipalidades y por casi todas las organizaciones internacionales de desarrollo para valorar y hacer seguimiento y evaluación de proyectos/programas.

Según Kari Örtengren el modelo de marco lógico es:

- Un instrumento de análisis lógico y pensamiento estructurado en la planificación de proyectos.
- Un marco, una batería de preguntas que, proporciona una estructura para el diálogo entre las diferentes partes interesadas en un proyecto.
- Un instrumento de planificación que incluye los diferentes elementos en un proceso de cambio (problemas, objetivos, partes interesadas, plan de implementación, etc.).
- Un instrumento para crear participación/responsabilidad/propiedad.

Con el concepto planificación orientada a objetivos se indica que el punto de partida del proceso de planificación es el análisis del problema, el que guía hacia los objetivos y hace posible, finalmente, la elección de actividades relevantes. Por lo tanto, antes de hacer el plan de actividades, es necesario un análisis de los problemas y objetivos.

El método ML es una herramienta de uso adecuado para el desarrollo de capacidades en “el esfuerzo de facilitar a individuos, grupos u organizaciones para que puedan identificar y hacer frente a los desafíos del desarrollo”, al facilitar la discusión entre las partes interesadas para identificar los obstáculos del cambio. Durante el análisis del problema, las necesidades y modelos sobre diferentes

formas de desarrollo de capacidades pueden ser reconocidas y volverse transparentes.

El análisis de problema muestra si es necesario fortalecer el conocimiento profesional de los individuos involucrados, o si es necesario utilizar un enfoque más amplio.

El modelo de marco lógico contiene nueve pasos diferentes:

1. Análisis del contexto del proyecto.
2. Análisis de las partes interesadas.
3. Análisis de problema/ análisis de situación.
4. Análisis de objetivos.
5. Plan de actividades.
6. Planificación de recursos.
7. Indicadores/ mediciones de los objetivos.
8. Análisis de riesgos y manejo de riesgos.
9. Análisis de supuestos.

Un proceso de planificación orientada a objetivos, como es el ML, se hace en los nueve pasos diferentes arriba presentados.”²³

²³ Método de Marco Lógico
www.sida.se/shared/jsp/download.jsp?f=SIDA1489es_Metodo+de+marco+Logico.+web.pdf&a=2379

6.13. ARQUITECTURA CLIENTE/SERVIDOR.

6.13.1. Introducción.

“Es un sistema informático en el que las tareas se distribuyen entre diferentes aplicaciones. Es decir, en vez de construir una sola aplicación en la que se contemple la realización de todos los trabajos, estos trabajos se realizan por varias aplicaciones que incluso pueden ejecutarse en máquinas diferentes.

Existen dos tipos de aplicaciones:

- La aplicación servidora que suele realizar las tareas de más trabajo y consumo de recursos.
- La aplicación cliente que se encarga fundamentalmente de hacer peticiones de datos a la aplicación servidora.

Los sistemas cliente/servidor se suelen ejecutar en máquinas distintas: una máquina en la que se ejecuta la aplicación servidora y otra u otras en la que se ejecuta la aplicación cliente. Por eso, hay que disponer de un PC potente para la aplicación servidora y varios PCs domésticos donde se ejecute la aplicación cliente.”²⁴

6.13.2. Ventajas.

1. Los desarrollos son más cortos.

²⁴ <http://es.wikipedia.org/wiki/Cliente-servidor>

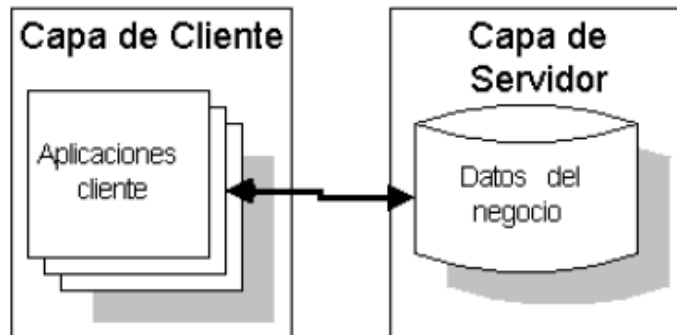
2. Mayor duración de las aplicaciones.
3. Es posible reconfigurar el servidor o los clientes sin la necesidad de alterar los programas que constituyen la aplicación.
4. Aumento de rendimiento y reducción de gastos, debido a que la aplicación se puede redistribuir por distintas máquinas.

6.13.3. Modelo Cliente /Servidor de dos capas.

“Tradicionalmente la arquitectura cliente/servidor está basada en un modelo de computación de dos capas. Este modelo consiste de un cliente y un servidor de base de datos. El procesamiento de tareas y la lógica de la aplicación son compartidas entre el servidor de base de datos y el cliente.

En este modelo los clientes son clientes gordos o clientes pesados, en donde reside mucho del poder de procesamiento y de la lógica de la aplicación. Esto hace que el mantenimiento del cliente sea costoso. Adicionalmente, los clientes pueden estar operando en diferentes plataformas, dando como resultado una distribución compleja de versiones de aplicaciones específicas de las plataformas.

Figura 5. Modelo Cliente/ Servidor de 2 capas.



Fuente: <http://es.wikipedia.org/wiki/Cliente-servidor>

6.14. METODOLOGÍA PARA LA ELICITACIÓN DE REQUISITOS DEL SOFTWARE

“El objetivo de esta metodología es la definición de las tareas a realizar, los productos a obtener y las técnicas a emplear durante la actividad de Elicitación de requisitos de la fase de Ingeniería de requisitos del desarrollo de software.

En esta metodología se distinguen dos tipos de productos: los productos entregables y los productos no entregables o internos. Los productos entregables son aquellos que se entregan oficialmente al cliente como parte del desarrollo en fechas previamente acordadas, mientras que los no entregables son productos internos al desarrollo que no se entregan al cliente.”²⁵

²⁵ Metodología para la Elicitación de Requisitos de Sistemas Software Versión 2.3. www.lsi.us.es/informes/lsi-2000-10.pdf

6.15. ETAPAS DE LA ELICITACIÓN SEGÚN AMADOR DURAN Y BEATRIZ BERNÁRDEZ.

6.15.1. Etapa 1.

“Obtener información sobre el dominio del problema y el sistema actual.

Objetivos Etapa 1.

- Conocer el dominio del problema.
- Conocer la situación actual.

Descripción Etapa 1.

Antes de mantener las reuniones con los clientes y usuarios e identificar los requisitos es fundamental conocer el dominio del problema y los contextos organizacional y operacional, es decir, la situación actual.

Enfrentarse a un desarrollo sin conocer las características principales ni el vocabulario propio de su dominio suele provocar que el producto final no sea el esperado por clientes ni usuarios.

Por otro lado, mantener reuniones con clientes y usuarios sin conocer las características de su actividad hará que probablemente no se entiendan sus necesidades y que su confianza inicial hacia el desarrollo se vea deteriorada enormemente.

Esta etapa es opcional, ya que puede que no sea necesario realizarla si el equipo de desarrollo tiene experiencia en el dominio del problema y el sistema actual es conocido.”²⁶

6.15.2. Etapa 2.

“Preparar y realizar las sesiones de Elicitación/negociación.

Objetivos Etapa 2.

- Identificar a los usuarios participantes.
- Conocer las necesidades de clientes y usuarios.
- Resolver posibles conflictos.

Descripción Etapa 2.

Teniendo en cuenta la información recopilada en la etapa anterior, en esta etapa se deben preparar y realizar las reuniones con los clientes y usuarios participantes con objeto de obtener sus necesidades y resolver posibles conflictos que se hayan detectado en iteraciones previas del proceso.

²⁶ Metodología para la Elicitación de Requisitos de Sistemas Software Versión 2.3. pág. 3
www.lsi.us.es/informes/lsi-2000-10.pdf

Esta etapa es especialmente crítica y ha de realizarse con especial cuidado, ya que generalmente el equipo de desarrollo no conoce los detalles específicos de la organización para la que se va a desarrollar el sistema y, por otra parte, los clientes y posibles usuarios no saben qué necesita saber el equipo de desarrollo para llevar a cabo su labor.

6.15.3. Etapa 3.

“Identificar/revisar los objetivos del sistema.

Objetivos Etapa 3.

- Identificar los objetivos que se esperan alcanzar mediante el sistema software a desarrollar.
- Revisar, en el caso de que haya conflictos, los objetivos previamente identificados.

Descripción Etapa 3.

A partir de la información obtenida en la etapa anterior, en esta etapa se deben identificar qué objetivos se esperan alcanzar una vez que el sistema software a desarrollar se encuentre en explotación o revisarlos en función de los conflictos identificados. Puede que los objetivos hayan sido proporcionados antes de comenzar el desarrollo.

6.15.4. Etapa 4.

“Identificar/revisar los requisitos de información.

Objetivos Etapa 4.

- Identificar los requisitos de almacenamiento de información que deberá cumplir el sistema software a desarrollar.
- Identificar los requisitos de restricciones de información o reglas de negocio que deberá cumplir el sistema software a desarrollar.
- Revisar, en el caso de que haya conflictos, los requisitos de almacenamiento y/o de restricciones de información previamente identificados.

Descripción Etapa 4.

A partir de la información obtenida en las etapas 1 y 2, y teniendo en cuenta los objetivos identificados en la etapa 3 y el resto de los requisitos, en esta etapa se debe identificar, o revisar si existen conflictos, qué información relevante para el cliente deberá gestionar y almacenar el sistema software a desarrollar así como qué restricciones o reglas de negocio debe cumplir dicha información.

Inicialmente se partirá de conceptos generales para posteriormente ir detallándolos hasta obtener todos los datos relevantes.

6.15.5. Etapa 5.

“Identificar/revisar los requisitos funcionales

Objetivos Etapa 5

- Identificar los actores del sistema software a desarrollar.
- Identificar los requisitos funcionales, expresados de forma tradicional o como casos de uso, que deberá cumplir el sistema software a desarrollar.
- Revisar, en el caso de que haya conflictos, los requisitos funcionales previamente identificados.

Descripción Etapa 5.

A partir de la información obtenida en las etapas 1 y 2, y teniendo en cuenta los objetivos identificados en la etapa 3 y el resto de los requisitos, en esta etapa se debe identificar, o revisar si existen conflictos, qué debe hacer el sistema a desarrollar con la información identificada en la etapa anterior.

Inicialmente se identificarán los actores que interactuarán con el sistema, es decir aquellas personas u otros sistemas que serán los orígenes o destinos de la información que consumirá o producirá el sistema a desarrollar y que forman su entorno.

A continuación se identificarán los casos de uso asociados a los actores, y posteriormente se detallarán las posibles excepciones hasta definir todas las situaciones posibles.

En el caso de que se considere necesario, se podrá optar por expresar algunos o todos los requisitos funcionales de la forma tradicional, es decir, mediante un párrafo en lenguaje natural, en lugar de hacerlo mediante casos de uso.

6.15.6. Etapa 6.

“Identificar/revisar los requisitos no funcionales.

Objetivos Etapa 6.

- Identificar los requisitos no funcionales del sistema software a desarrollar.
- Revisar, en el caso de que haya conflictos, los requisitos no funcionales previamente identificados.

Descripción Etapa 6.

A partir de la información obtenida en las etapas 1 y 2, y teniendo en cuenta los objetivos identificados en la etapa 3 y el resto de los requisitos, en esta etapa se deben identificar, o revisar si existen conflictos, los requisitos no funcionales, normalmente de carácter técnico o legal.

Algunos tipos de requisitos que se suelen incluir en esta sección son los siguientes:

- Requisitos de comunicaciones del sistema.

Son requisitos de carácter técnico relativos a las comunicaciones que deberá soportar el sistema software a desarrollar. Por ejemplo: el sistema deberá utilizar el protocolo TCP/IP para las comunicaciones con otros sistemas.

- Requisitos de interfaz de usuario.

Este tipo de requisitos especifica las características que deberá tener el sistema en su comunicación con el usuario. Por ejemplo: la interfaz de usuario del sistema deberá ser consistente con los estándares definidos en *IBM's Common User Access*.

Es importante ser cuidadoso con este tipo de requisitos, ya que en esta fase de desarrollo todavía no se conocen bien las dificultades que pueden surgir a la hora de diseñar e implementar las interfaces, por esto no es conveniente entrar en detalles demasiado específicos.

- Requisitos de fiabilidad.

Los requisitos de fiabilidad deben establecer los factores que se requieren para la fiabilidad del software en tiempo de explotación. La fiabilidad mide la probabilidad del sistema de producir una respuesta satisfactoria a las demandas del usuario.

Por ejemplo: la tasa de fallos del sistema no podrá ser superior a 2 fallos por semana.

- Requisitos de entorno de desarrollo

Este tipo de requisitos especifican si el sistema debe desarrollarse con un producto específico. Por ejemplo: el sistema deberá desarrollarse con Oracle 7 como servidor y clientes Visual Basic 4.

- Requisitos de portabilidad

Los requisitos de portabilidad definen qué características deberá tener el software para que sea fácil utilizarlo en otra máquina o bajo otro sistema operativo. Por ejemplo: el sistema deberá funcionar en los sistemas operativos Windows 95, Windows 98 y Windows NT 4.0, siendo además posible el acceso al sistema a través de Internet usando cualquier navegador compatible con HTML 3.0.

6.16. ¿QUÉ JAVA?

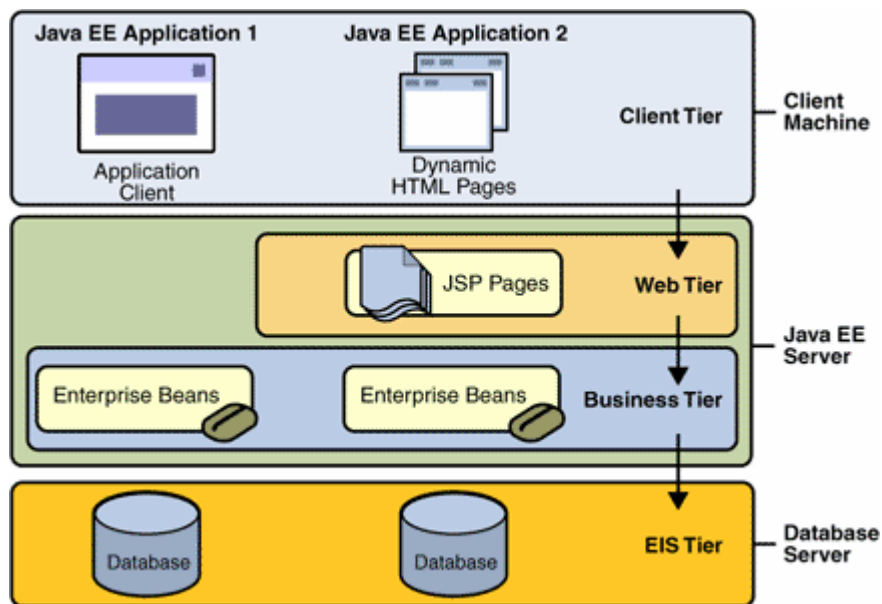
“Java es un lenguaje de programación con el que podemos realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de internet como en la informática en general. Esta desarrollado por la compañía Sun Microsystems.

Una de las principales características es que es un lenguaje independiente de la plataforma. Eso quiere decir que si hacemos un programa en Java podrá funcionar

en cualquier ordenador. Es una ventaja significativa para los desarrolladores de software, pues antes tenían que hacer un programa para cada sistema operativo, por ejemplo, Linux, Aple, etc. Esto lo consigue por que se ha creado una Maquina de java y posibilita que este último se entienda perfectamente.

Actualmente Java se utiliza en un amplio abanico de posibilidades y casi cualquier cosa que se pueda hacer en cualquier lenguaje se puede hacer también en Java y muchas veces con grandes ventajas, con java se puede programar paginas web dinámicas, con acceso a base de datos, utilizando XML, con cualquier tipo de conexión a red entre cualquier sistema.

Figura 6. Modelo Arquitectura Java.



Fuente: <https://wiki.zylk.net:448/zwiki/CursoJboss/J2EE>

6.17. ¿QUÉ PHP?

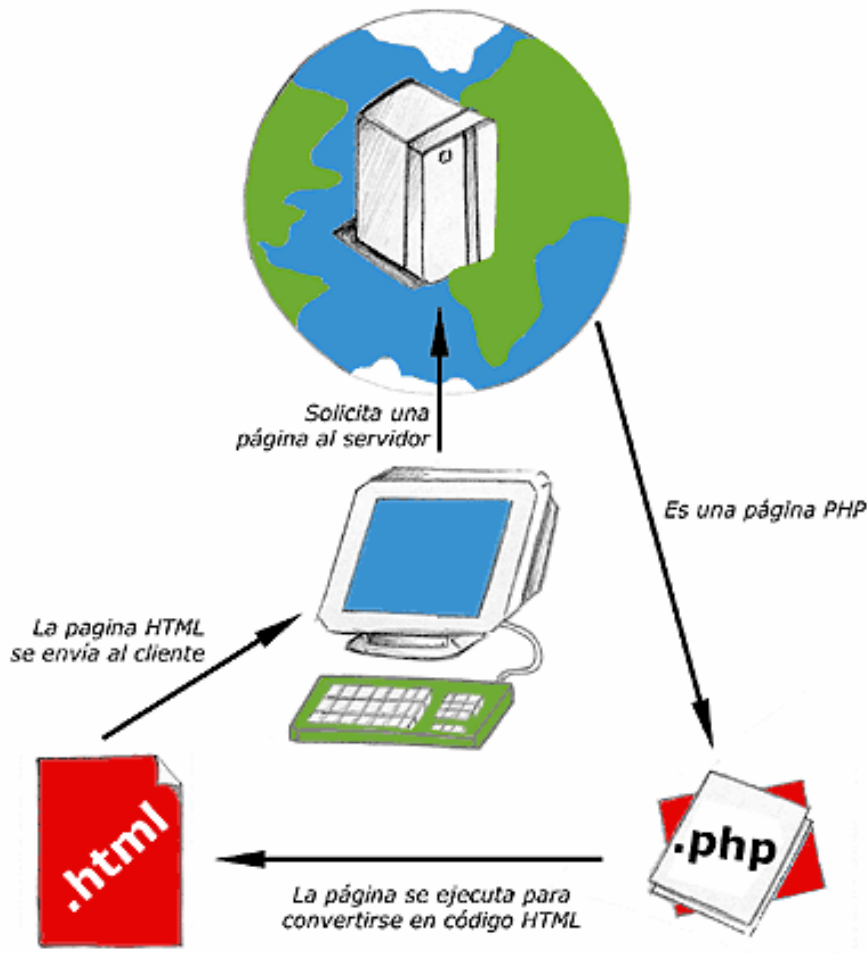
Es un lenguaje para programar scripts del lado del servidor, que se incrustan dentro del código HTML. Este lenguaje es gratuito y multiplataforma.

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar acceso a base de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML, es compatible con todos los navegadores.

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red.

Algunas de las importantes capacidades de PHP son la compatibilidad con las bases de datos más comunes, como MySQL, mSQL, Oracle; Informix y ODBC, por ejemplo. Incluye funciones para envío de correo electrónico, upload de archivos, crear dinámicamente en el servidor de imágenes en formato GIF, incluso animadas y una lista interminable de utilidades adicionales.

Figura 7. Modelo Arquitectura PHP.



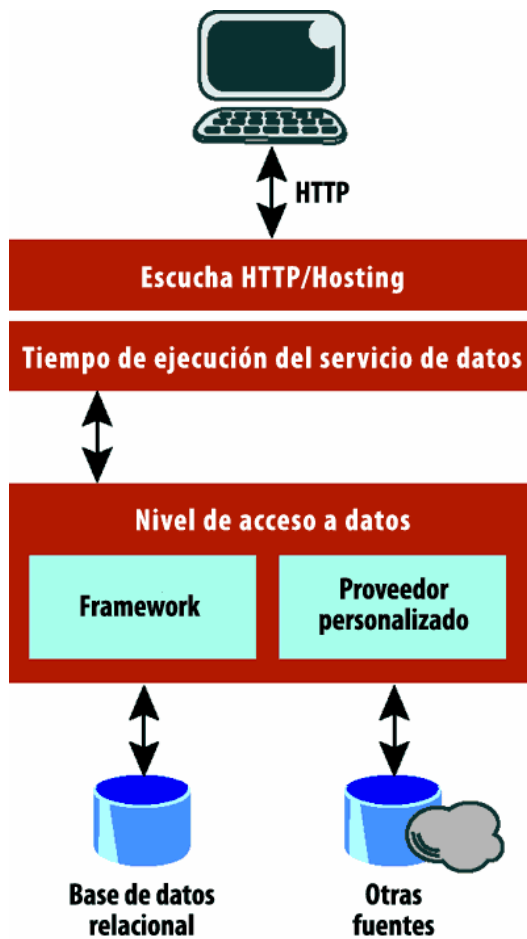
Fuente: <http://www.miguelmatas.es/blog/2007/11/01/buscando-una-arquitectura-preparada-para-hacerse-mayor/>

6.18. ¿QUÉ FRAMEWORK?

Es un concepto sumamente genérico, se refiere a ambiente de trabajo, y ejecución para desarrollar aplicaciones. En general los Framework son soluciones completas que contemplan herramientas de apoyo a la construcción ambiente de trabajo o desarrollo y motores de ejecución.

En general con el termino FrameWork, se refiere a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. Se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta.

Figura 8. Modelo Arquitectura FrameWork.



Fuente: <http://ivanex.wikidot.com/patron-arquitectura>

7. MARCO TEÓRICO.

La Elicitación es el proceso de adquirir o sonsacar todo el conocimiento relevante necesario para producir un modelo de los requerimientos de un dominio del problema.

Partiendo de esta teoría Amador Durán y Beatriz Bernárdez profesores titulares de la Universidad de Sevilla España crearon un informe técnico sobre la metodología para la Elicitación de requisitos de sistemas software, el cual sirve de apoyo en la elaboración del análisis y el diseño del mismo. El objetivo de esta metodología es la definición de las tareas a realizar, los productos a obtener y las técnicas a emplear durante la fase de ingeniería de requisitos del desarrollo de software. En esta metodología se distinguen dos tipos de productos: los productos entregables y los no entregables o internos que se obtienen a partir de las siete tareas recomendadas. Los productos entregables son aquellos que se entregan oficialmente al cliente como parte del desarrollo, mientras que los no entregables son productos internos al desarrollo que no se entregan al cliente, y permiten obtener la fase de ingeniería de requerimientos de software.

En el desarrollo de este proyecto de investigación se optó por esta metodología debido a que es muy ordenada, completa y facilita todos los procesos requeridos en la Ingeniería de software.

De igual manera se utilizó la metodología orientada a objetos que se apoya “UML”^{*} (Lenguaje Modelado Unificado), el cual fue el resultado de unificar 3 métodos de análisis y diseño:

^{*} <http://es.wikipedia.org/wiki/UML>

Lenguaje Unificado de Modelado (UML), por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. funciones del sistema, y

- La metodología de “Grady Booch”^{*} para la descripción de conjuntos de objetos y sus relaciones.
- La técnica de modelado orientada a objetos de “James Rumbaugh”.^{**}
- La metodología de casos de uso de “Ivar Jacobson”^{***}.

El desarrollo de UML comenzó a finales de 1994 y en 1997 fue aprobada la versión 1.1 por la “OMG”^{****} convirtiéndose así en la notación estándar para el análisis y el diseño del software.

UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir, se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software, pero no especifica en sí mismo qué metodología o proceso usar. El UML se enfoca en tres modelos diferentes; “el modelo funcional

aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

^{*} http://es.wikipedia.org/wiki/Grady_Booch

Grady Booch es mejor conocido por el desarrollo del Lenguaje Unificado de Modelado con Ivar Jacobson y James Rumbaugh. También desarrolló el método Booch de desarrollo de software, el que presenta en su libro, Análisis y Diseño Orientado a Objetos. Él aconseja la adición de más clases para simplificar códigos complejos.

^{**} http://es.wikipedia.org/wiki/James_Rumbaugh

James Rumbaugh es un científico de la computación y un metodologista de objeto. Es mejor conocido por su trabajo en la creación de la Técnica de Modelado de Objetos y el Lenguaje Unificado de Modelado (UML). Doctorado en ciencias de la computación por el M.I.T.

^{***} http://es.wikipedia.org/wiki/Ivar_Jacobson

Ivar Hjalmar Inventó el diagrama de secuencia y desarrolló los diagramas de colaboración. También impuso el uso de diagramas de estado de transición para describir los flujos de mensajes entre los componentes. Fue uno de los desarrolladores originales del SDL (lenguaje de especificación), que se convirtió en estándar en 1967.

^{****} <http://www.lamdasi.com.ar/textocomp.asp?id=326>

La OMG (Object Management Group) es una asociación sin fines de lucro formada por grandes corporaciones, muchas de ellas de la industria del software, como IBM, Apple, Sun Microsystems y HP.

es la representación a alto nivel del flujo de información, utiliza diagramas de casos de uso, el modelo de objetos es la representación de objetos, clases, jerarquías y relaciones, describe la estructura del sistema, y el modelo dinámico es la representación del comportamiento interno del sistema y los objetos, utiliza diagramas de secuencia.”²⁷

Se eligió UML (lenguaje de modelado unificado) debido a que la metodología de Elicitación de requisitos de sistemas software mencionada anteriormente así lo recomienda, por su capacidad para especificar, visualizar, construir y documentar un sistema informático.

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principio de los años 90`s. Esta herramienta tiene constantes mejoras en el desempeño, escalabilidad y seguridad con respecto a las versiones anteriores, Java es un entorno de desarrollo para la mayoría de los sistemas operativos.

PHP es un lenguaje de script creado por una gran comunidad de personas. El sistema fue desarrollado originalmente en el año 1994 por Rasmus Lerdorf como un CGI escrito en C que permitía la interpretación de un número limitado de comandos.

Estas herramientas se eligieron debido a que es orientada a objetos y es pertinente para este tipo de estudios.

SQL server 5.1.49-1 permite crear, desplegar y administrar aplicaciones empresariales más seguras, escalables y confiables, es un sistema de gestión de bases de datos relacionales capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea, permite trabajar la arquitectura cliente-servidor, donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información, característica

²⁷ Ingeniería del Software Orientado a Objetos. Bernd Bruegge – Allen H Putoit. Editorial Prentice Hall. Primera Edición 2002

de gran importancia debido a que la herramienta informática requiere de esta arquitectura.

MD5 permite codificar las contraseñas utilizadas para así garantizar un mayor nivel de seguridad en la aplicación, y a nivel de equipos CISCO están las listas de acceso²⁸ con privilegios y manejo del protocolo SNMP.

“El software se realizó bajo el modelo de ciclo de vida en espiral el cual consiste en una serie de ciclos que se repiten, cada uno tiene las mismas fases y cuando termina da un producto ampliado con respecto al ciclo anterior, su mayor ventaja es que no necesita una definición completa de los requisitos para empezar a funcionar”²⁹

²⁸ http://www.guillesql.es/Articulos/Manual_Cisco_CCNA_Listados_Acceso.aspx

²⁹ Ingeniería del Software. Ian Somerville. Editorial Pearson Addison Wesley. Séptima Edición 2007

8. MARCO DE ANTECEDENTES.

Realizando una revisión en las principales Universidades de la región no se encontraron trabajos o documentación relacionada con software mediador de equipos CISCO. En la Internet se encuentra documentación del manejo de los equipos pero no se ha encontrado un mediador para estos equipos.

9. MARCO LEGAL.

El desarrollo de software esta regido por los siguientes decretos, circulares y leyes.

9.1 DECRETO 1360/1989 PROPIEDAD INTELECTUAL.

“Este decreto reglamenta la inscripción del soporte lógico en el registro nacional de derecho de autor.”³⁰

9.2 CIRCULAR NO. 06 DE 15 DE ABRIL DE 2002 DE UNIDAD ADMINISTRATIVA ESPECIAL DIRECCIÓN NACIONAL DE DERECHO DE AUTOR.

“Por el cual se reglamenta la inscripción del soporte lógico (Software) en el registro Nacional del Derecho de Autor. Esta circular se encarga de aclarar los conceptos de derechos de autor en el ámbito Universitario.

³⁰ http://www.unal.edu.co/viceinvestigacion/normatividad/decreto_1360.pdf

9.3 LEY 1273 DE 2009 PROTECCIÓN DE LA INFORMACIÓN Y DE LOS DATOS.

“Por medio del cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado - denominado “de la protección de la información y de los datos”- y se preservan integralmente los sistemas que utilicen las tecnologías de la información y las comunicaciones, entre otras disposiciones.”³¹

9.4. ACUERDO 003/2003 MODELO DE EVALUACIÓN DOCENCIA-SERVICIO.

“La relación Docencia-Servicio fue reglamentada por el ministerio de la protección social según este acuerdo por medio del cual se adoptaron criterios para evaluar y verificar los convenios Docente-Asistencial” ³²

9.5. MODELO DE EVALUACIÓN DE LA RELACIÓN DOCENCIA-SERVICIO: CRITERIOS BÁSICOS DE CALIDAD PARA CENTROS DE PRACTICAS FORMATIVAS.

“Este modelo presenta criterios para la evaluación de la relación Docencia-Servicio, que se genera en el desarrollo de la interacción del componente de “prácticas formativas” de los programas y procesos educativos del personal de salud que se llevan a cabo en las instituciones prestadoras de servicios de salud.

³¹http://www.mindefensa.gov.co/descargas2/anexos/10077_1273.pdf?PHPSESSID=834cfea003c32e32ea89945e7f6235d9

³²<http://www.minproteccionsocial.gov.co/VbeContent/library/documents/DocNewsNo14546DocumentNo1426.PDF>

En la característica 11 del factor IV se reglamenta el desarrollo de un sistema de información veraz, ágil y oportuna, que permita la toma de decisiones. Igualmente se deben optar mecanismos para la producción, emisión y divulgación de la comunicación generada en la relación Docencia-Servicio.”³³

³³<http://www.minproteccionsocial.gov.co/VbeContent/library/documents/DocNewsNo14546DocumentNo1426.PDF>

10. DESARROLLO METODOLÓGICO.

10.1. INTRODUCCIÓN AL ANÁLISIS DEL SOFTWARE.

La aplicación que está en desarrollo permite a la Empresa de Telefónica de Pereira tener un control más preciso de los agentes de Help Desk en la manipulación y control de los equipos y tener un reporte detallado de los comandos que utiliza cada uno de ellos.

10.2. POBLACIÓN.

10.2.1. Participantes.

Tabla 4. Participantes del proyecto.

Nombre	Papel	Organización
Jhon Alexander Marín Sánchez	Analista y Diseñador	Universidad Libre Pereira
Anderson Lenis Pareja	Analista y Diseñador	Universidad Libre Pereira
Raúl Alberto Gaviria	Asesor en el diseño técnico del Proyecto	Universidad Libre Pereira

Fuente: los autores

10.2.2. Beneficiarios.

Tabla 5. Beneficiarios del proyecto.

Nombre	Papel	Organización
Jhon Jairo Zapata Pachón	Técnico en Instalación y Mantenimiento	Telefónica de Pereira
Cesar Augusto Melo Rivera	Técnico en Instalación y Mantenimiento	Telefónica de Pereira

Fuente: los autores

10.3. DESCRIPCIÓN DEL SISTEMA ACTUAL.

En el desarrollo de la metodología se realizó un diagnóstico del problema identificados en la Empresa de Telecomunicaciones de Pereira los cuales se describen en el numeral 1.2.

10.4. OBJETIVOS DEL ANÁLISIS DEL SOFTWARE.

10.4.1. Objetivo General.

Analizar, diseñar y desarrollar una aplicación informática que funcione como mediador entre usuarios y equipos CISCO de la Empresa de Telecomunicación de Pereira. Que facilite la manipulación de los equipos y permita la generación de eventos registrados en ellos.

10.4.2. Objetivos Específicos.

10.4.2.1. Generación de Informes.

Tabla 6. Objetivo 1 Generación de informes.

OBJ- 1	Generación de informes
Versión	2010-06-25 / 2010-07-15
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	El sistema deberá generar informes de: listado de comandos utilizados por los usuarios.
Sub objetivos	Permitir obtener información de los usuarios cada vez que se requiera.
Importancia	Alta

Fuente: los autores

10.4.2.2. Gestión de usuarios.

Tabla 7. Objetivo 2 Gestión de usuarios.

OBJ- 2	Gestión de usuarios
Versión	2010-06-24
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	El sistema deberá gestionar la información correspondiente a los usuarios que van a manejar el sistema: ingresar, modificar, consultar y eliminar.
Importancia	Alta

Fuente: los autores

10.4.2.3. Validar usuarios.

Tabla 8. Objetivo 3 Validar usuario.

OBJ- 3	Validar usuario
Versión	2010-06-24
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	El sistema deberá validar la información del usuario que desea ingresar al sistema
Importancia	Alta

Fuente: los autores

10.4.2.4. Gestión de perfil de usuarios y permisos.

Tabla 9. Objetivo 4. Gestión de perfil de usuarios y permisos.

OBJ- 4	Gestión de perfil de usuarios y permisos
Versión	2010-06-24
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	El sistema deberá gestionar la información correspondiente a los perfiles y permisos de los usuarios.
Importancia	Alta

Fuente: los autores

10.4.2.5. Gestión de Comandos.

Tabla 10. Objetivo 5 Gestión de comandos.

OBJ- 5	Gestión de comandos
Versión	2010-10-28 / 2010-06-15
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	El sistema deberá gestionar la información correspondiente a los comandos: ingresar, modificar, consultar y eliminar.
Importancia	Alta

Fuente: los autores

10.4.2.6. Gestión de Administradores.

10.4.2.7. Seguridad.

Tabla 11. Objetivo 6 Seguridad.

OBJ- 6	Seguridad
Versión	2010-06-23
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	Garantizar la seguridad a través del ingreso de los usuarios registrados en el sistema.
Importancia	Alta

Fuente: los autores

10.5. REQUISITOS DE INFORMACIÓN.

10.5.1. Requisitos de almacenamiento de información.

10.5.1.1. Información de los Usuarios.

Tabla 12. Requisito de información 01 Información de usuarios.

IRQ- 01	Información de usuarios
Versión	2010-06-24 2010-07-07
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Objetivos asociados	OBJ.4 Gestión de usuarios OBJ.6 Gestión de perfiles de usuario y permisos
Requisitos asociados	UC-01 Validar usuario UC-02 Crear usuario UC-03 Modificar usuario UC-04 Consultar usuario UC-05 Eliminar Usuario UC-06 Asociar Comandos UC-07 Modificar perfil

Continúa en la página siguiente.

Continuación de la tabla anterior.

Descripción	El sistema deberá almacenar la información correspondiente a los usuarios que lo van a utilizar. Como mínimo	
Datos específicos	Nombres, Apellidos, Nick, Contraseña, Comandos.	
Tiempo de vida	Indefinido	
Ocurrencias simultaneas	Medio	Máximo
	20	100
Importancia	Alta	

Fuente: los autores

10.5.1.2. Información de comandos.

Tabla 13. Requisito de Información 02 Información de comandos.

IRQ- 02	Información de comandos
Versión	2010-06-18 / 2010-07-07
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Requisitos asociados	UC-16 Ingresar Comandos UC-17 Modificar Comandos UC-19 Eliminar Comandos UC-18 Consultar Comandos

Continúa en la página siguiente.

Continuación de la tabla anterior.

Descripción	El sistema deberá almacenar la información correspondiente a los comandos. Como mínimo.	
Datos específicos	Comando, Descripción.	
Tiempo de vida	Indefinido	
Ocurrencias simultaneas	Medio	Máximo
	20	100
Importancia	Alta	

Fuente: los autores

10.5.1.3. Relación entre usuarios y comandos.

Tabla 14. Requisito de restricción de información 03 Relación entre usuarios y comandos.

CQR- 03	Relación entre usuarios y comandos
Versión	2010-06-15 / 2010-07-07
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	La información almacenada por el sistema deberá satisfacer la siguiente restricción: para ubicar a un usuario se debe tener en cuenta que comandos tiene asociados.
Importancia	Alta
Comentario	Ninguno

Fuente: los autores

10.6. REQUISITOS FUNCIONALES.

10.6.1. Definición de actores.

10.6.1.1. Usuario.

Tabla 15. Actor 01 Usuario.

ACT – 01	Usuario
Versión	2010-10-25 / 2010-06-15
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	Este actor representa el usuario del sistema.
comentarios	El usuario del sistema tiene definido perfiles que determinan las acciones que pueden realizar.

Fuente: los autores

10.6.1.2. Base de datos.

Tabla 16. Actor 02 Base de datos.

ACT – 02	Base de Datos (BDD)
Versión	2010-10-25 / 2010-06-15
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja

Continúa en la página siguiente.

Continuación de la tabla anterior.

Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	Este actor representa la base de datos del software
comentarios	Aquí se almacenan todos los datos del software

Fuente: los autores

10.6.1.3. Interfaz de usuario.

Tabla 17. Actor 03 Interfaz de usuario.

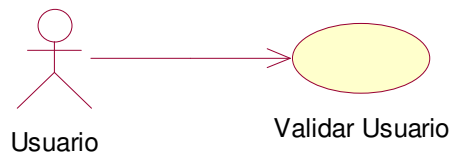
ACT – 03	Interfaz de usuario
Versión	2010-10-25 / 2010-06-15
Autores	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín Sánchez – Anderson Lenis Pareja
Descripción	Este actor representa las interfaces gráficas de usuario del software
comentarios	Aquí se visualiza gráficamente las peticiones y validaciones del sistema.

Fuente: los autores

10. 6. 2 Diagramas y Documentación de casos de uso.

10. 6. 2.1. Módulo Validación de Usuarios.

Figura 9. Diagrama caso de uso Validar Usuario.



Fuente: los autores

Tabla 18. Documentación caso de uso 01 Validar Usuario

UC-01	Validar Usuario	
Versión	2010-06-29	
Actores	Usuarios	
Propósito	Validar al usuario que desea ingresar al sistema	
Resumen	El usuario ingresa al sistema a través de un nombre de usuario y contraseña, el sistema se encarga de verificar y consultar su perfil y comandos asociados.	
Precondición	El usuario debe haberse creado previamente para poder ingresar al sistema	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El caso de uso inicia cuando el usuario desea ingresar al sistema.	2. La herramienta pide el nombre de usuario y contraseña.

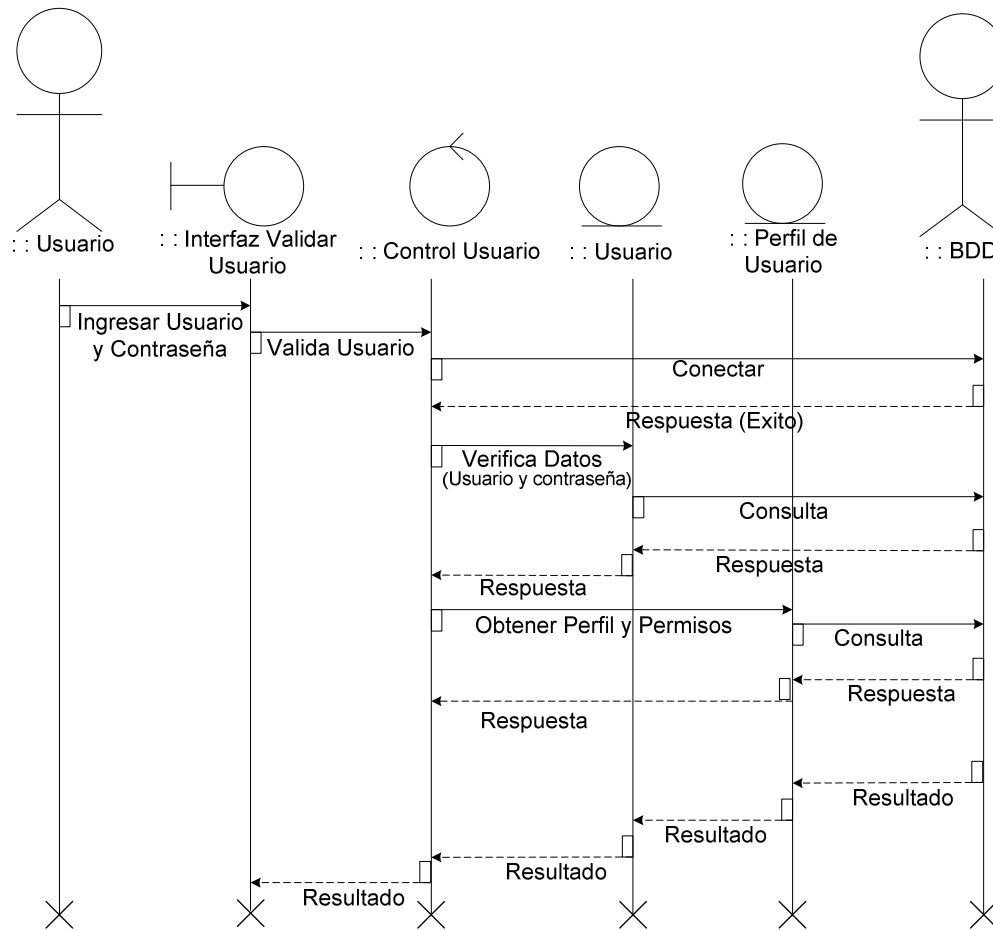
Continúa en la página siguiente.

Continuación de la tabla anterior.

	3. El usuario digita el nombre de usuario y contraseña.	4. La herramienta informática valida los datos proporcionados por el usuario.
		5. La herramienta informática obtiene el perfil y permisos del usuario.
		6. Inicia la herramienta informática en la pantalla principal dependiendo del perfil y permisos del usuario.
Excepciones de eventos	Si la herramienta detecta que el nombre de usuario y contraseña son incorrectos envía un mensaje de nombre de usuario y/o contraseña no validos	

Fuente: los autores

Figura 10. Diagrama de secuencia Validar Usuario.

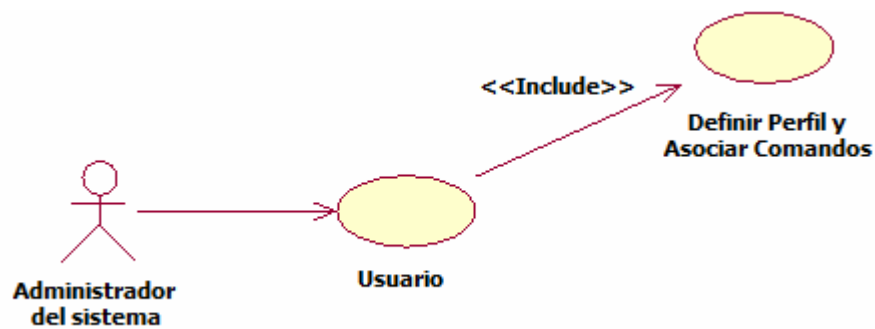


Fuente: los autores

10.6.2.2. Módulo Gestión de Usuarios.

10.6.2.2.1. Caso de uso Crear Usuario.

Figura 12. Diagrama caso de uso Crear Usuario



Fuente: los autores

Tabla 19. Documentación caso de uso 02 Crear Usuario.

UC-02	Crear Usuario
Versión	2010-06-29 / 2010-10-02
Actores	Administrador del Sistema
Propósito	Registrar los datos de un nuevo usuario. Asociar Comandos y Contraseña
Resumen	Se ingresan los datos, se crea la contraseña y se definen los comandos de los del usuario
Precondición	Ninguna

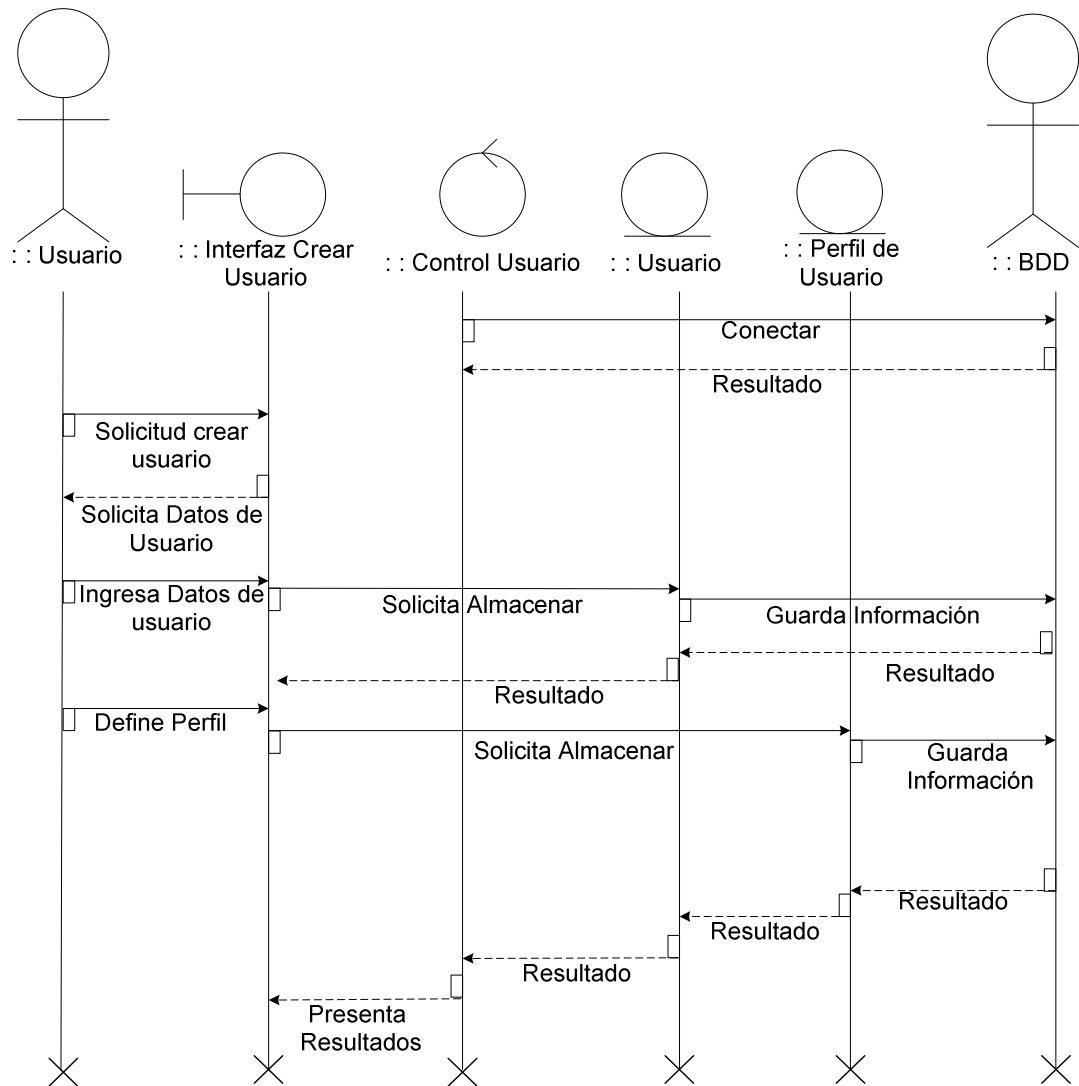
Continúa en la página siguiente.

Continuación de la tabla anterior.

	Acción del actor	Respuesta del sistema
Secuencia normal de eventos	1. El administrador solicita a la herramienta informática la creación de un nuevo usuario	2. La herramienta informática solicita los siguientes datos del nuevo usuario: nombres, apellidos, Nick, contraseña, asociar comandos
	3. El administrador del sistema proporciona los datos a la herramienta informática y solicita que los almacene.	4. La herramienta informática realiza las validaciones necesarias.
		5. La herramienta informática almacena los datos.
Excepciones de eventos	Si la herramienta informática detecta que el nuevo usuario ya ha sido ingresado, informa de la situación al administrador, permitiéndole modificar los datos proporcionados.	

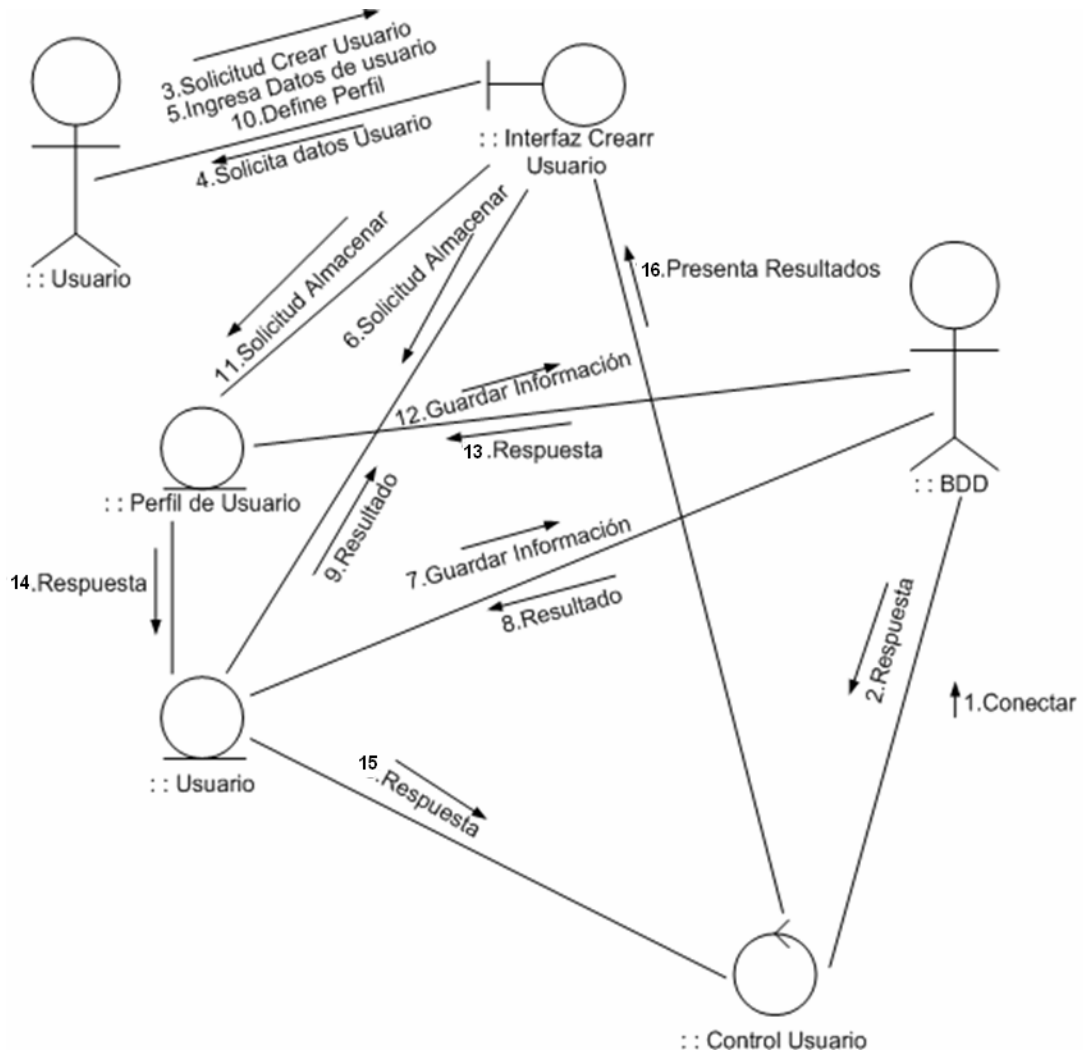
Fuente: los autores

Figura 13. Diagrama de secuencia Crear Usuario.



Fuente: los autores

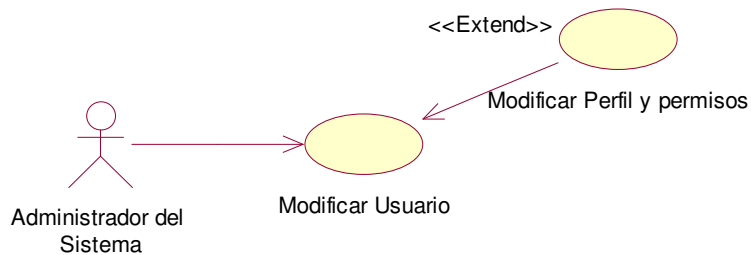
Figura 14. Diagrama de colaboración Crear Usuario.



Fuente: los autores

10.6.2.2.2. Caso de uso Modificar Usuario.

Figura 15. Diagrama caso de uso Modificar Usuario.



Fuente: los autores

Tabla 20. Documentación caso de uso 03 Modificar Usuario.

UC-03	Modificar Usuario	
Versión	2010-06-29 / 2010-10-02	
Actores	Administrador del Sistema	
Propósito	Modificar los datos y comandos asociados de un usuario.	
Resumen	El administrador del sistema modifica los datos y comandos asociados a un usuario.	
Precondición	El usuario debe estar registrado previamente. El perfil del administrador debe permitir modificar usuarios.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta informática modificar los datos de un usuario.	2. La herramienta informática solicita al administrador ingresar el ID del usuarios a modificar.

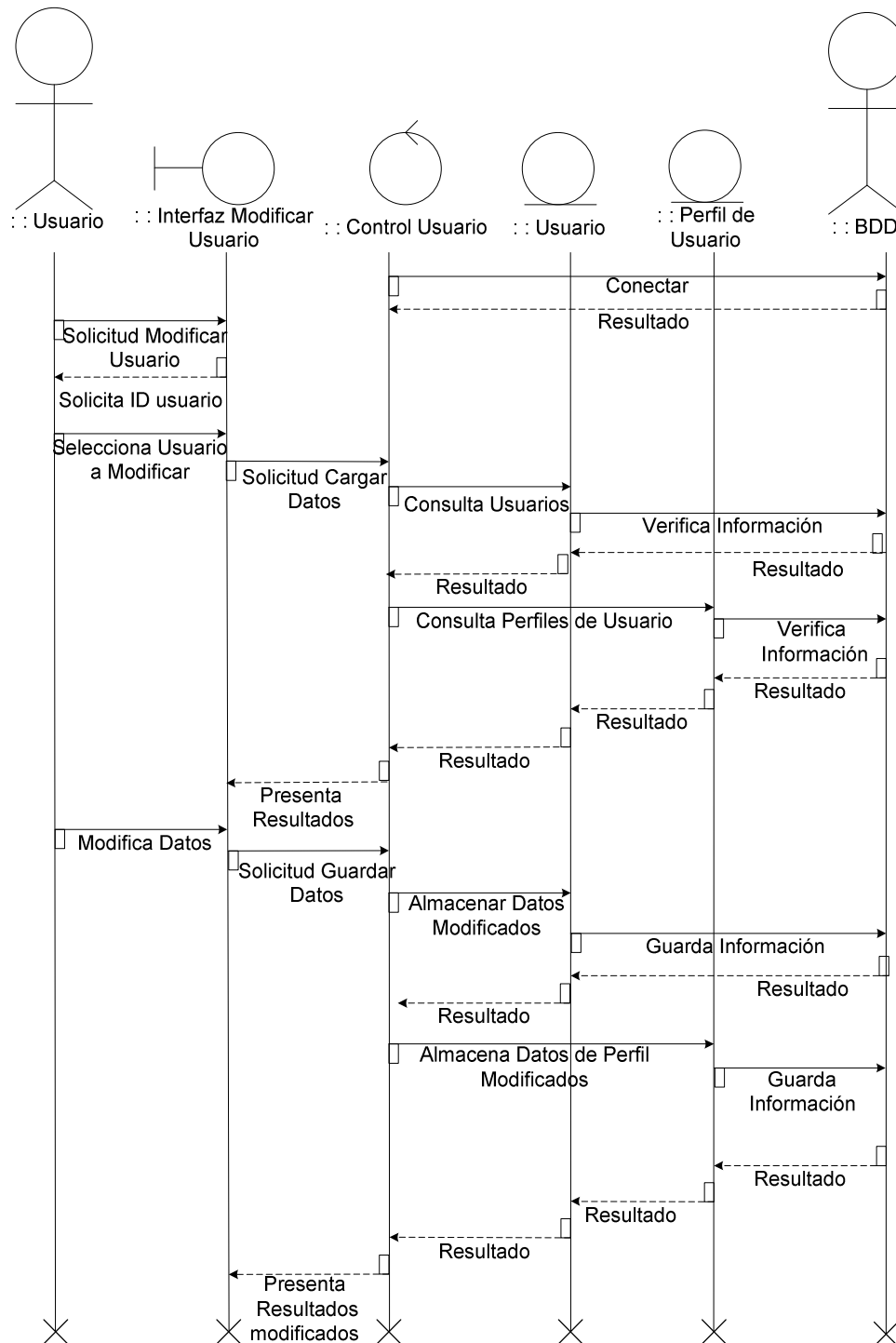
Continúa en la página siguiente.

Continuación de la tabla anterior.

Secuencia normal de eventos		3. La herramienta informática realiza las validaciones necesarias.
		4. La herramienta informática muestra los siguientes datos correspondientes al usuario a modificar: nombres apellidos, Nick, Contraseña y comandos asociados.
		5. La herramienta informática permite al administrador modificar los siguientes datos: nombres, apellidos, Nick, contraseña y comandos asociados.
	6. El administrador modifica los datos que la herramienta informática le permita y solicita que los almacene.	7. La herramienta informática modifica los datos correspondientes al usuario y los almacena.
Excepciones de eventos	Ninguna.	

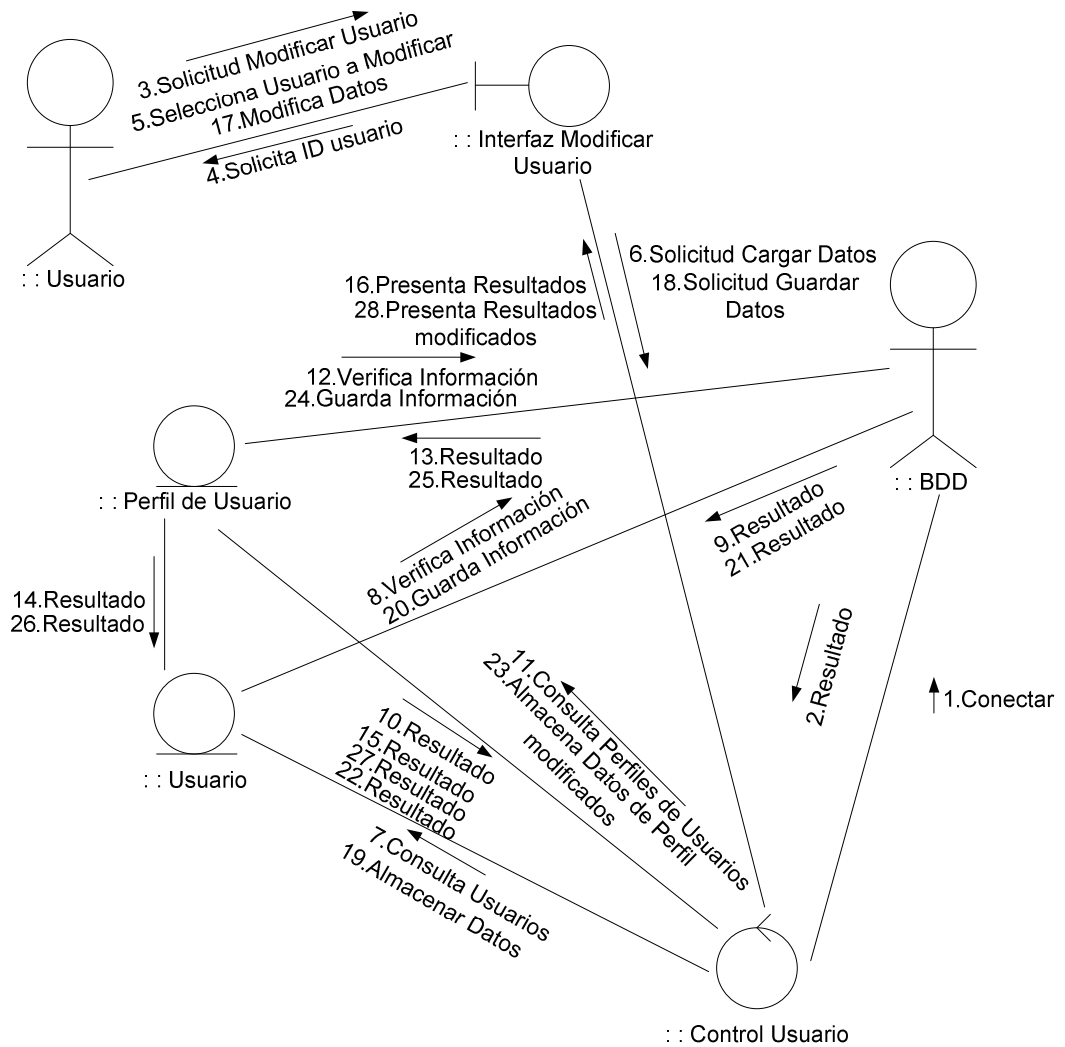
Fuente: los autores

Figura 16. Diagrama de secuencia Modificar Usuario.



Fuente: los autores

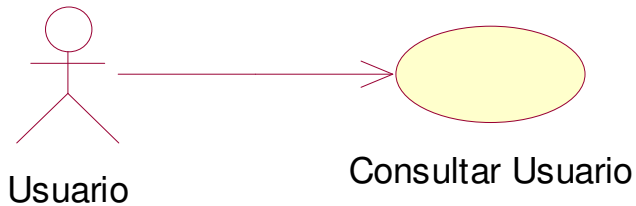
Figura 17. Diagrama de colaboración Modificar Usuario.



Fuente: los autores

10.6.2.2.3. Caso de uso Consultar Usuario.

Figura 18. Diagrama caso de uso Consultar Usuario.



Fuente: los autores

Tabla 21. Documentación caso de uso 04 Consultar Usuario.

UC-04	Consultar Usuario	
Versión	2010-06-19 / 2010-07-02	
Actores	Administrador del Sistema	
Propósito	Consultar usuarios.	
Resumen	El administrador del sistema modifica los datos y comandos asociados a un usuario.	
Precondición	El administrador del sistema se debe haber validado previamente. El usuario debe estar registrado previamente.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta consultar a los usuarios.	2. La herramienta solicita al administrador ingresar el Nick del usuario a consultar.

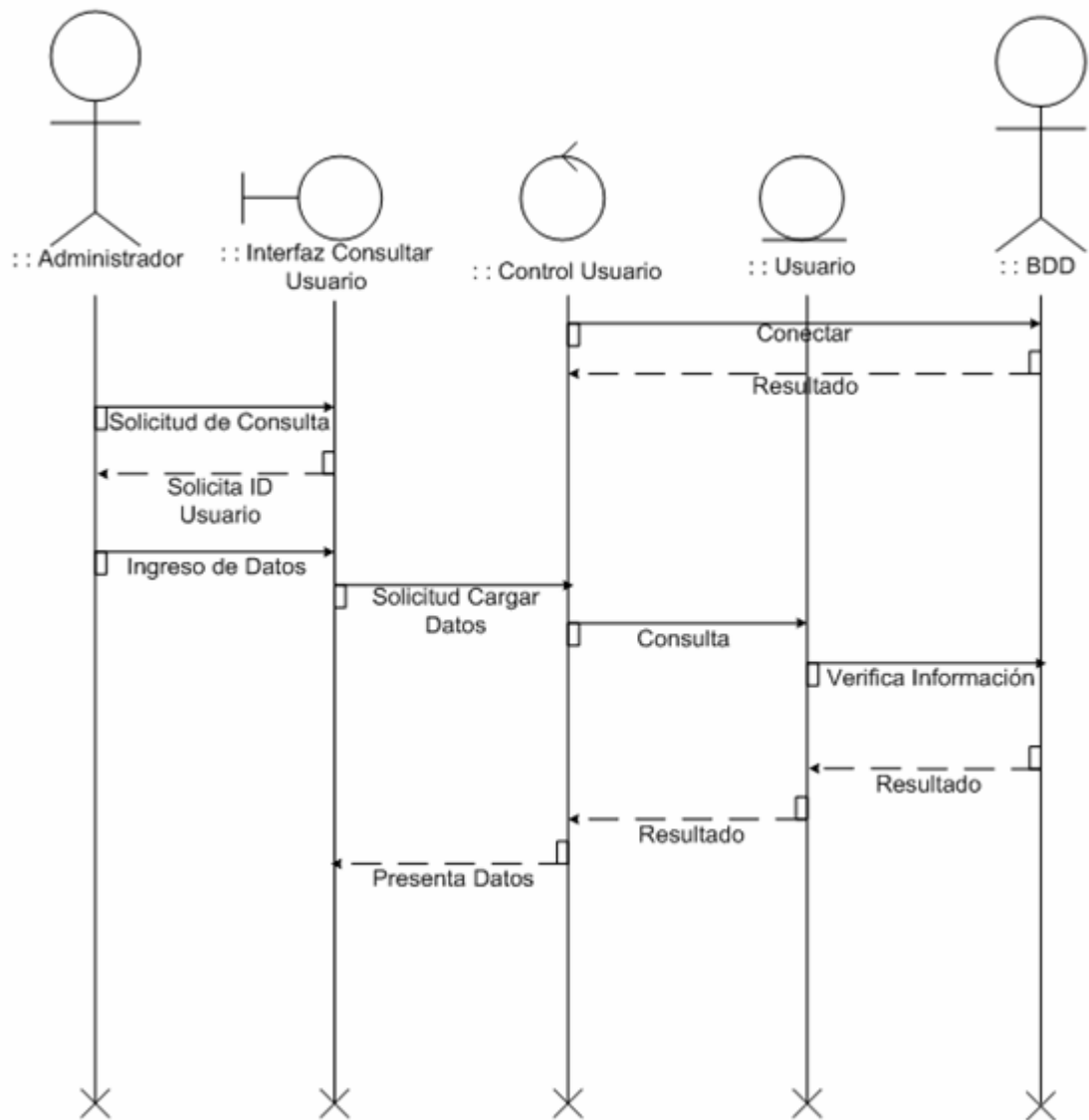
Continúa en la página siguiente.

Continuación de la tabla anterior.

	3. El administrador solicita a la herramienta consultar la información de usuarios.	4. La herramienta informática realiza las validaciones necesarias.
		5. La herramienta informática muestra los datos correspondientes al usuario: nombres, apellidos, Nick contraseña y comandos asociados.
Excepciones de eventos	Ninguna.	
Importancia	Alta.	

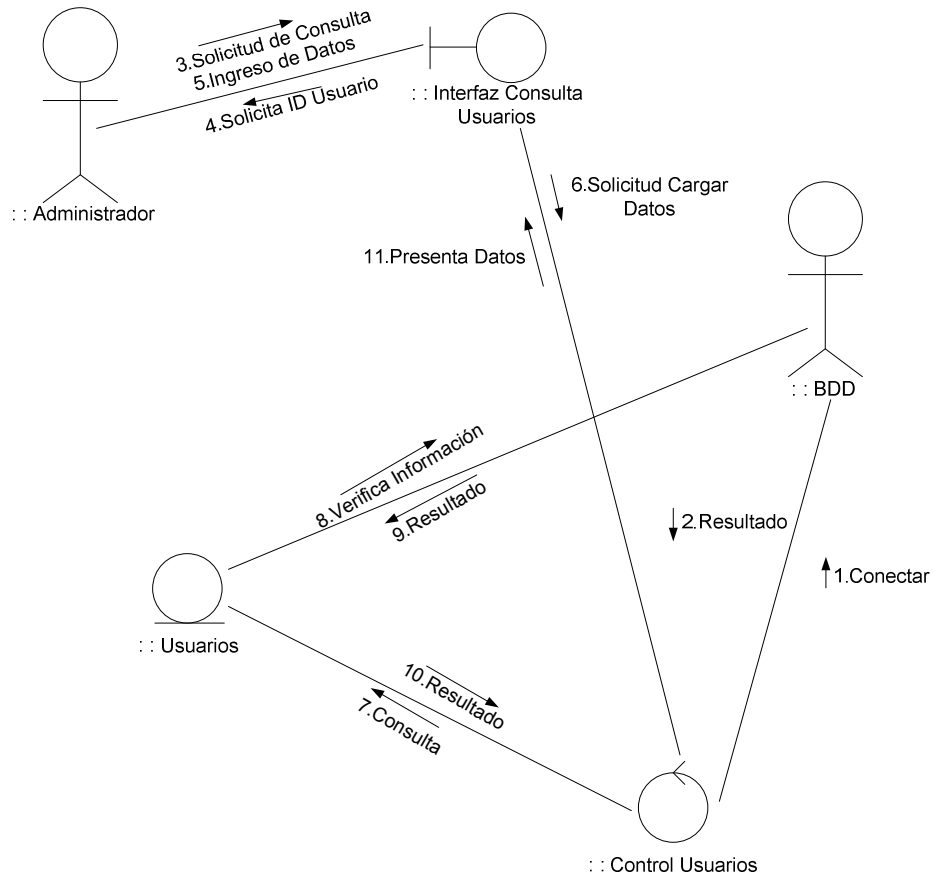
Fuente: los autores

Figura 19. Diagrama de secuencia Consultar Usuario.



Fuente: los autores

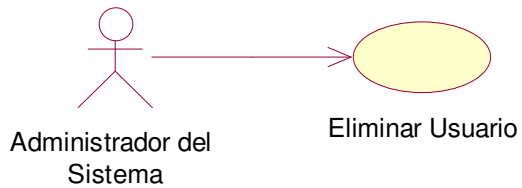
Figura 20. Diagrama de colaboración Consultar Usuario.



Fuente: los autores

10.6.2.2.4. Caso de uso Eliminar Usuario.

Figura 21. Diagrama caso de uso Eliminar Usuario.



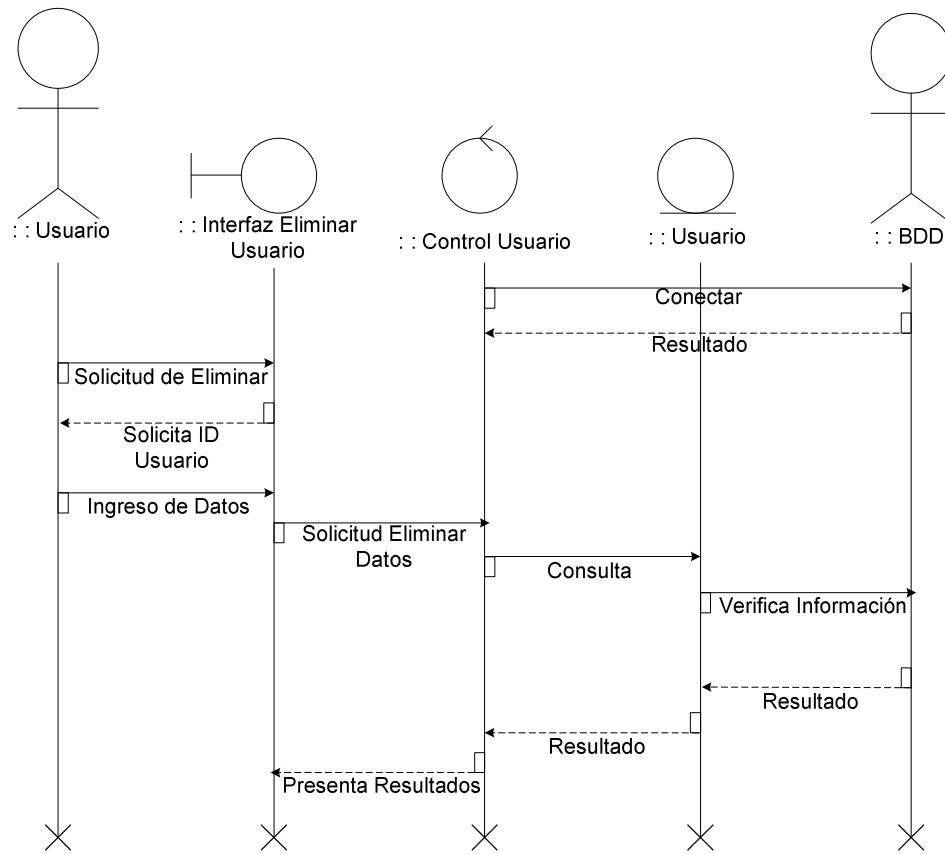
Fuente: los autores

Tabla 22 Documentación caso de uso 05 Eliminar Usuario.

UC-05	Eliminar Usuario	
Versión	2010-06-29	
Actores	Administrador del Sistema	
Propósito	Eliminar usuarios.	
Resumen	El administrador del sistema elimina un usuario del sistema.	
Precondición	El administrador del sistema debe estar registrado previamente. El perfil del administrador debe permitir eliminar usuarios.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta informática comenzar el proceso de eliminar un usuario.	2. La herramienta informática solicita al administrador ingresar el Nick del usuario que se requiere eliminar.
		3. La herramienta informática realiza las validaciones necesarias.
	4. El administrador solicita a la herramienta informática que elimine la información correspondiente al usuario.	5. La herramienta informática elimina los datos correspondientes al usuario.
Excepciones de eventos	Ninguna.	

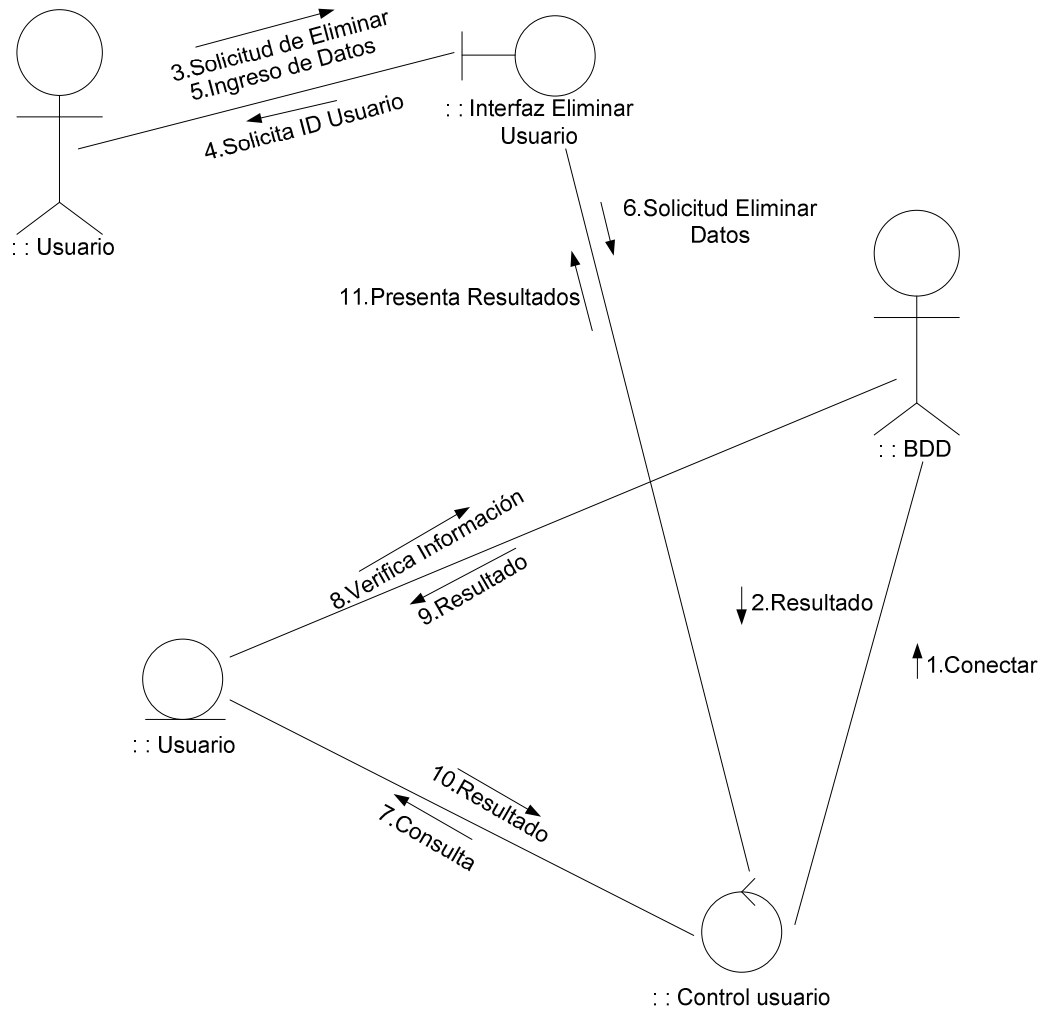
Fuente: los autores

Figura 22. Diagrama de secuencia Eliminar Usuario.



Fuente: los autores

Figura 23. Diagrama de colaboración Eliminar Usuario.

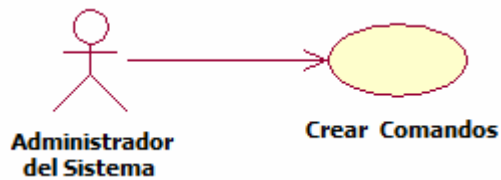


Fuente: los autores

10. 6. 2.3. Módulo gestión de Comandos.

10.6.2.3.1. Caso de uso Crear Comandos.

Figura 24. Diagrama caso de uso Crear Comandos.



Fuente: los autores

Tabla 23. Documentación caso de uso 06 Crear Comandos.

UC-06	Crear Comandos	
Versión	2010-06-29	
Actores	Administrador del Sistema	
Propósito	Crear comandos.	
Resumen	El administrador del sistema crea los comandos.	
Precondición	El administrador del sistema debe estar registrado previamente.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta informática la creación de un nuevo comando.	2. La herramienta informática solicita los siguientes datos del nuevo comando: comando, descripción.

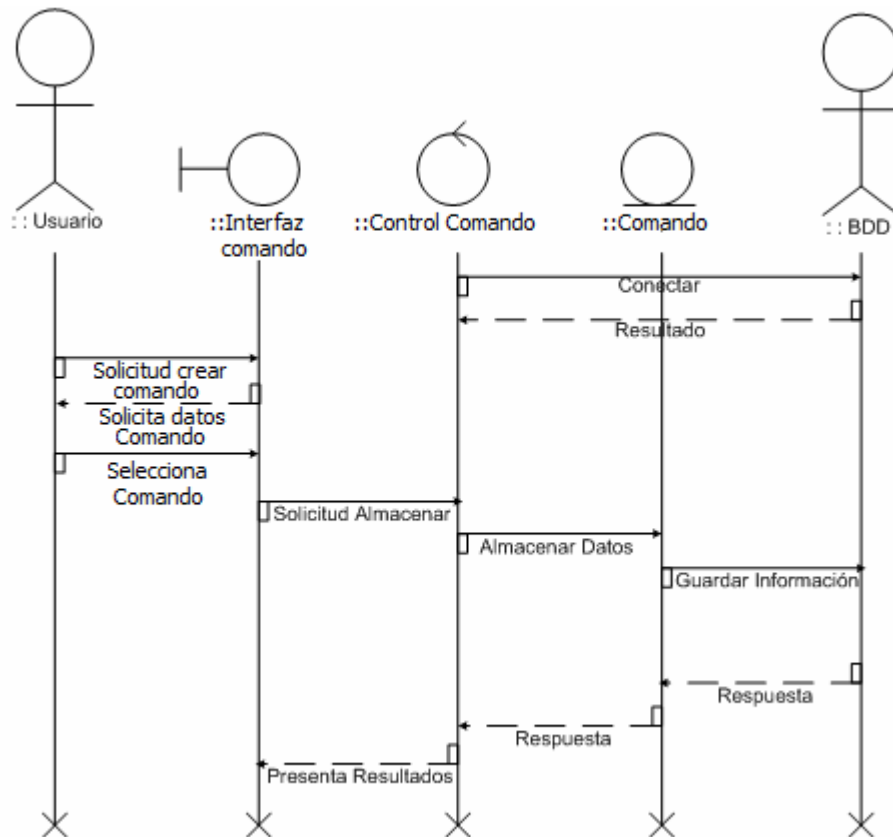
Continúa en la página siguiente.

Continuación de la tabla anterior.

	3. El administrador del sistema proporciona los datos a la herramienta informática y solicita que los almacene.	4. La herramienta informática realiza las validaciones necesarias.
		5. La herramienta informática almacena los datos.
Excepciones de eventos	Ninguna.	

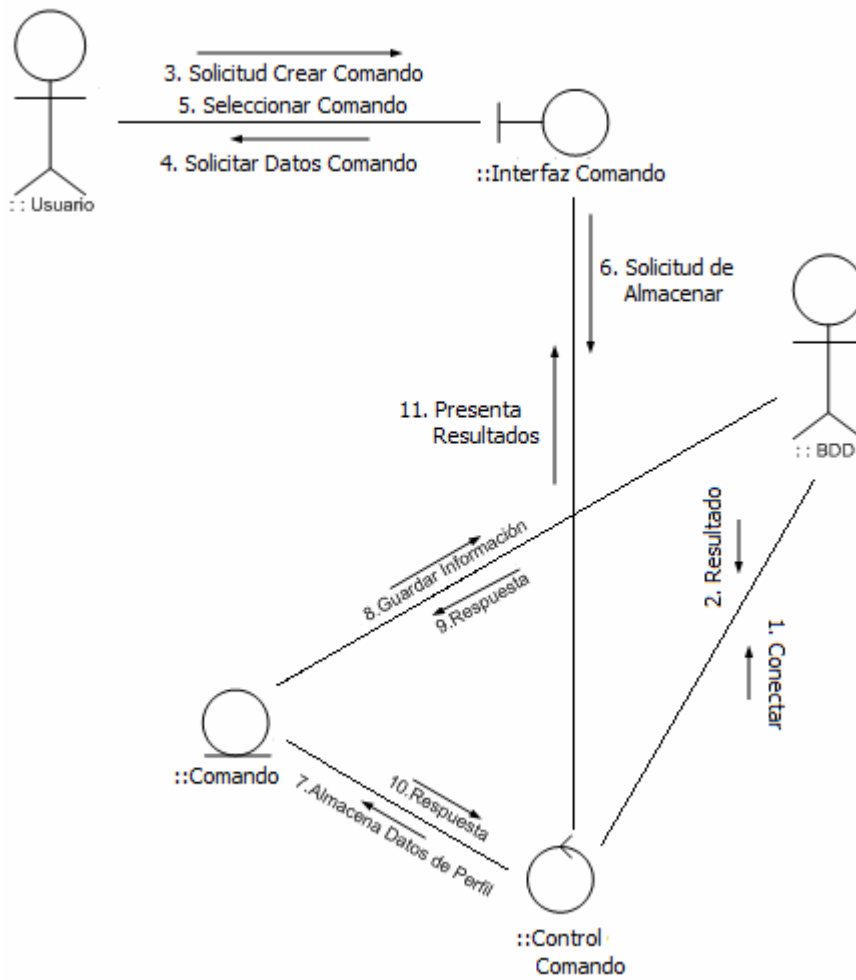
Fuente: los autores

Figura 25. Diagrama de secuencia Crear Comando.



Fuente: los autores

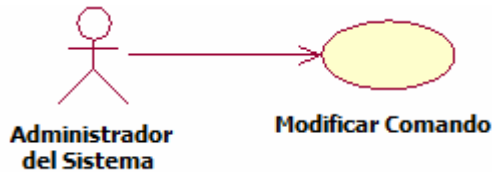
Figura 26. Diagrama de colaboración Crear Comando.



Fuente: los autores

10.6.2.3.2 Caso de uso Modificar Comandos.

Figura 27. Diagrama caso de uso Modificar Comando.



Fuente: los autores

Tabla 24. Documentación caso de uso 07 Modificar Comandos.

UC-07	Modificar Comandos	
Versión	2010-06-29	
Actores	Administrador del Sistema	
Propósito	Modificar comandos.	
Resumen	El administrador del sistema modifica los comandos.	
Precondición	El administrador del sistema debe estar registrado previamente.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta informática el comando a modificar.	2. La herramienta informática presenta en la pantalla los comandos.
	3. El administrador del sistema selecciona el comando que desea modificar.	4. La herramienta informática muestra la información del comando.

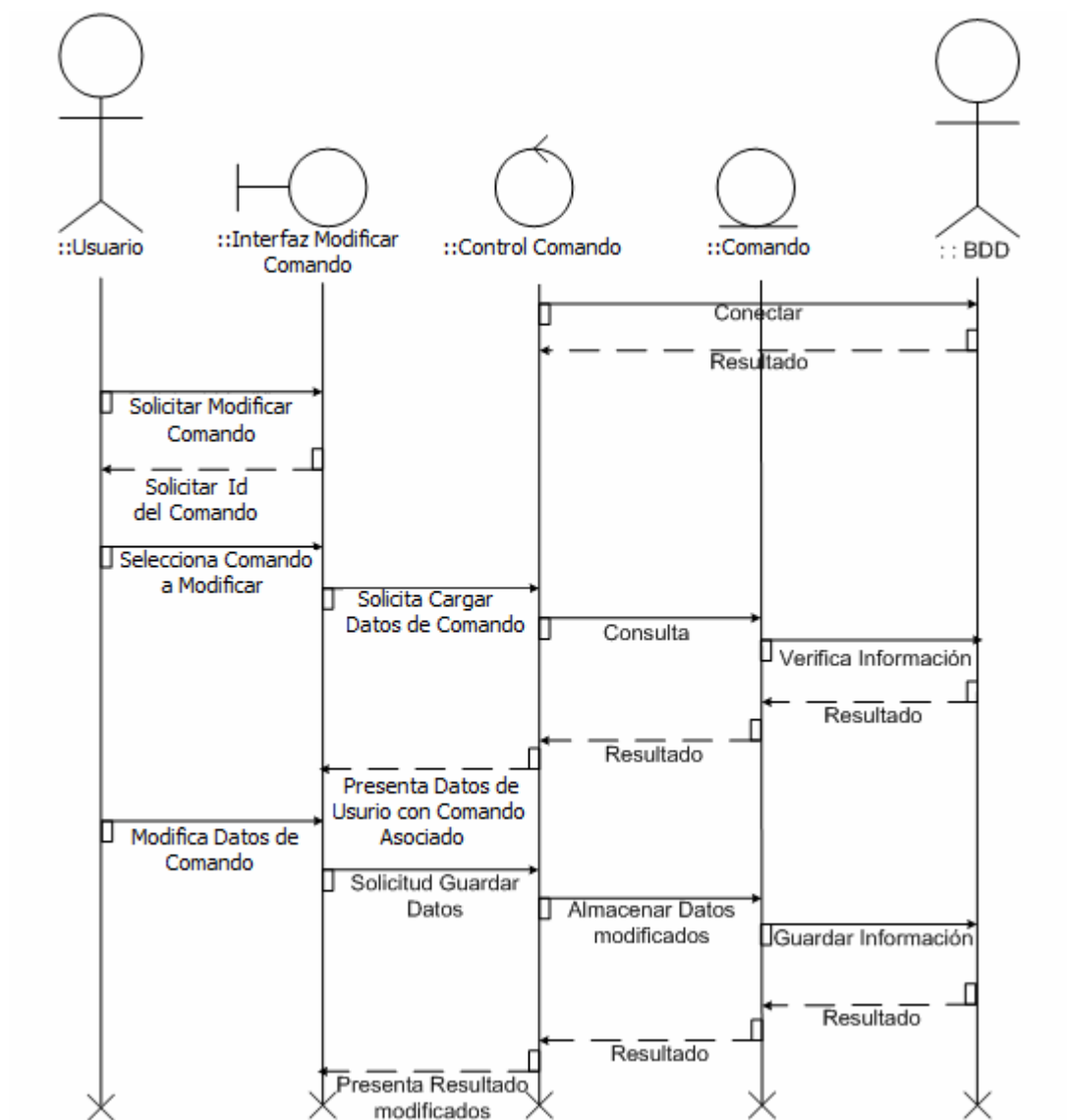
Continúa en la página siguiente.

Continuación de la tabla anterior.

		5. La herramienta informática actualiza los datos.
Importancia	Alta.	

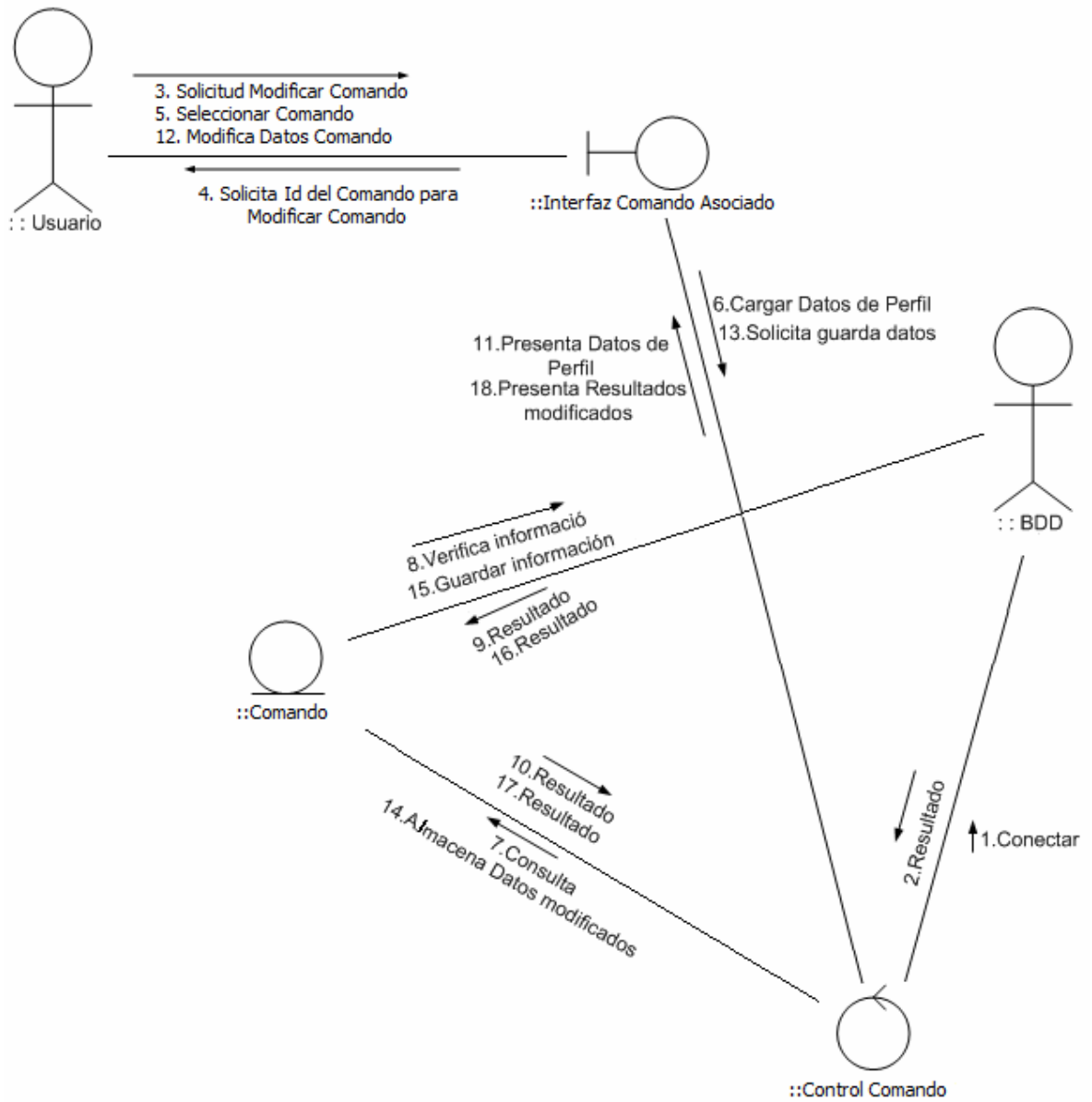
Fuente: los autores

Figura 28. Diagrama de secuencia Modificar Comandos.



Fuente: los autores

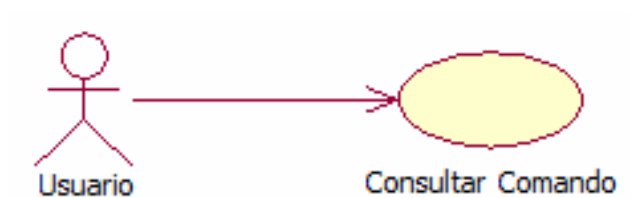
Figura 29. Diagrama de colaboración Modificar Comando.



Fuente: los autores

10.6.2.3.3. Caso de uso Consultar Comando.

Figura 30. Diagrama de caso de uso Consultar Comando.



Fuente: los autores

Tabla 25. Documentación caso de uso 08 Consultar Comando.

UC-07	Modificar Comandos	
Versión	2010-06-29	
Actores	Administrador del Sistema	
Propósito	Modificar comandos.	
Resumen	El administrador del sistema modifica los comandos.	
Precondición	El administrador del sistema debe estar registrado previamente.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta informática el comando a modificar.	2. La herramienta informática presenta en la pantalla los comandos.
	3. El administrador del sistema selecciona el comando que desea modificar.	4. La herramienta informática muestra la información del comando.

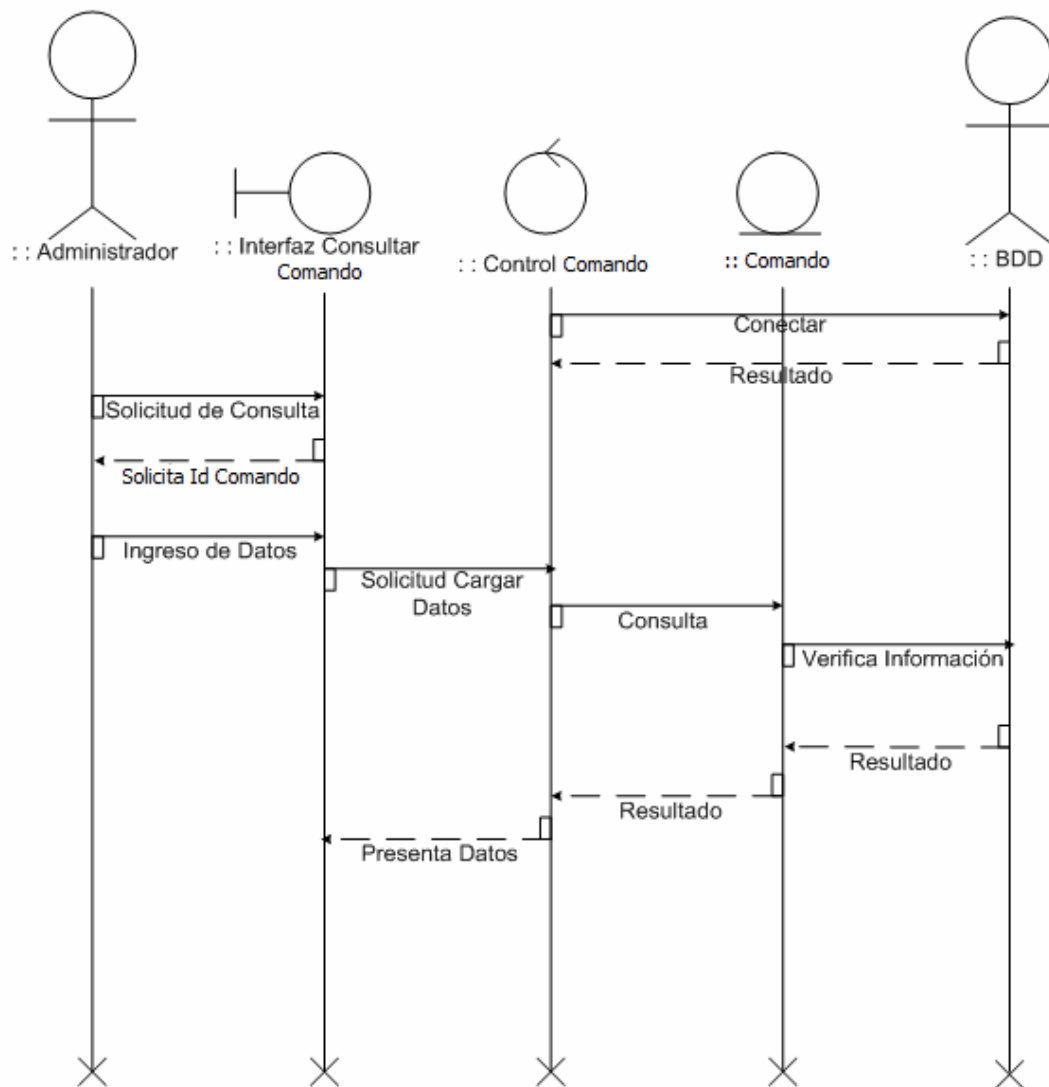
Continúa en la página siguiente.

Continuación de la tabla anterior.

		5. La herramienta informática actualiza los datos.
Importancia	Alta.	

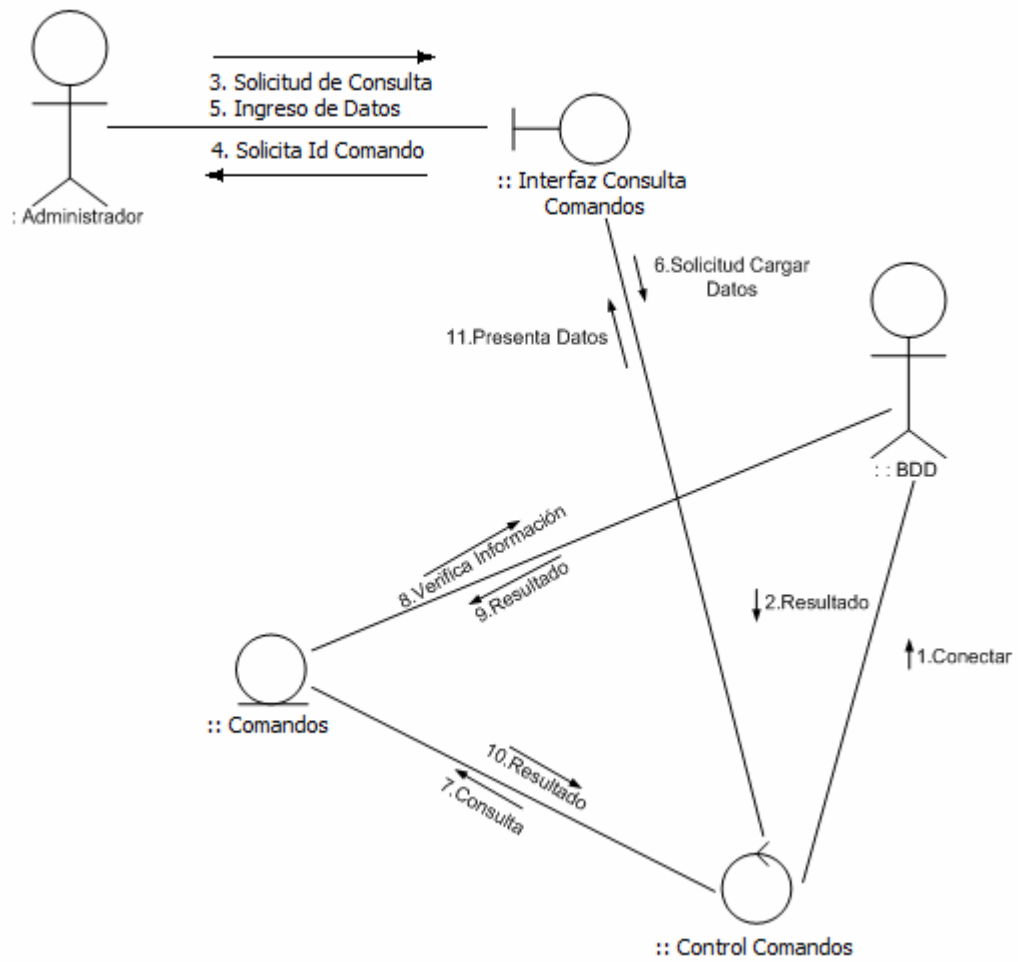
Fuente: los autores

Figura 31. Diagrama de secuencia Consultar Comando.



Fuente: los autores

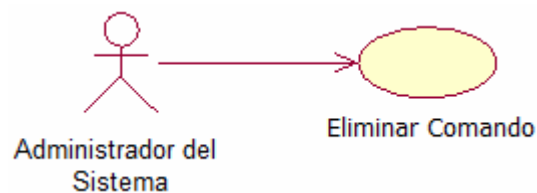
Figura 32. Diagrama de colaboración Consultar Comando.



Fuente: los autores

10.6.2.3.4. Caso de uso Eliminar Comando.

Figura 33. Diagrama caso de uso Eliminar Comando.



Fuente: los autores

Tabla 26. Documentación caso de uso 09 Eliminar Comando.

UC-09	Eliminar Comando	
Versión	2010-06-29	
Actores	Administrador del Sistema	
Propósito	Eliminar comando.	
Resumen	El administrador del sistema elimina un comando del sistema.	
Precondición	El administrador del sistema debe estar registrado previamente.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta informática comenzar el proceso de eliminar un comando.	2. La herramienta informática solicita al administrador ingresar el id del comando que se requiere eliminar.

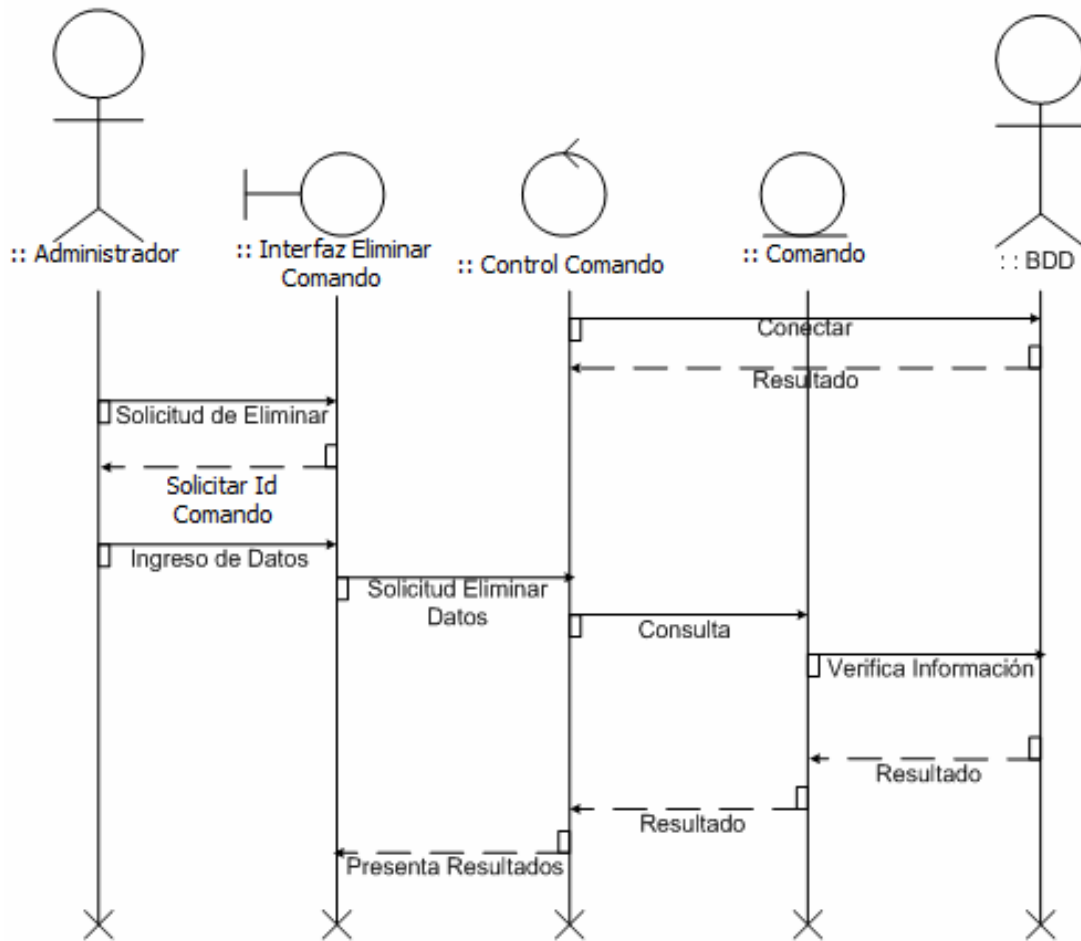
Continúa en la página siguiente.

Continuación de la tabla anterior.

		3. La herramienta informática realiza las validaciones necesarias.
	4. El administrador solicita a la herramienta informática que elimine la información correspondiente al comando.	5. La herramienta informática elimina los datos correspondientes al comando.
Excepciones de eventos	Ninguna.	

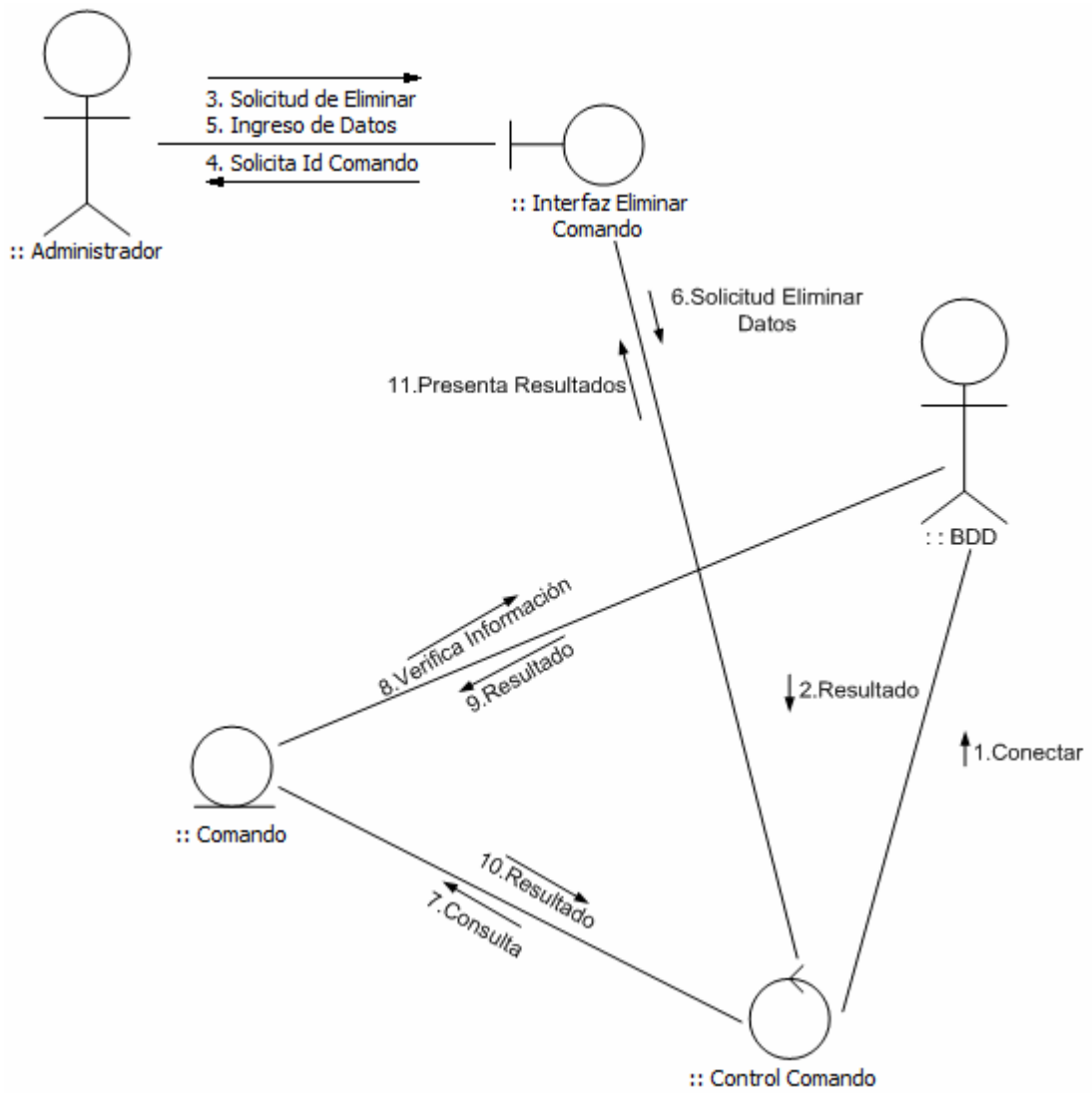
Fuente: los autores

Figura 34. Diagrama de secuencia Eliminar Comando.



Fuente: los autores

Figura 35. Diagrama de colaboración Eliminar Comando.

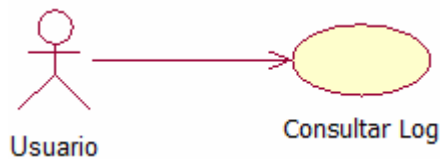


Fuente: los autores

10. 6. 2.4. Módulo gestión Logs.

10.6.2.4.1. Caso de uso Consultar Logs.

Figura 36. Diagrama de caso de uso Consultar Logs.



Fuente: los autores

Tabla 27. Documentación caso de uso 10 Consultar Logs.

UC-10	Consultar Logs	
Versión	2010-07-25 / 2010-08-16	
Actores	Administrador del Sistema	
Propósito	Consultar los logs de los usuarios.	
Resumen	El administrador del sistema consulta los logs de los usuarios.	
Precondición	El administrador del sistema debe estar registrado previamente.	
Secuencia normal de eventos	Acción del actor	Respuesta del sistema
	1. El administrador solicita a la herramienta informática que consulte la información del log.	2. La herramienta informática solicita al administrador ingresar al log de usuario y comando.

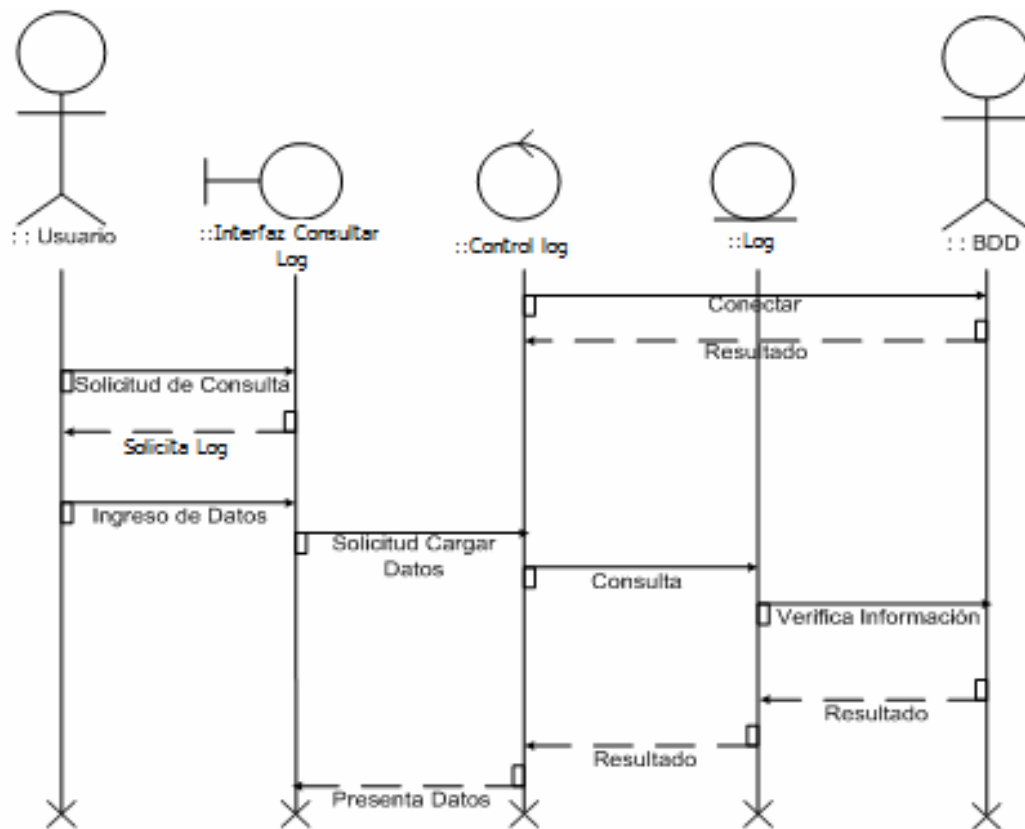
Continúa en la página siguiente.

Continuación de la tabla anterior.

		3. La herramienta informática muestra los datos correspondientes al log: nombres apellidos, Nick, comando, fecha y hora.
Excepciones de eventos	Ninguna.	
Importancia	Alta	

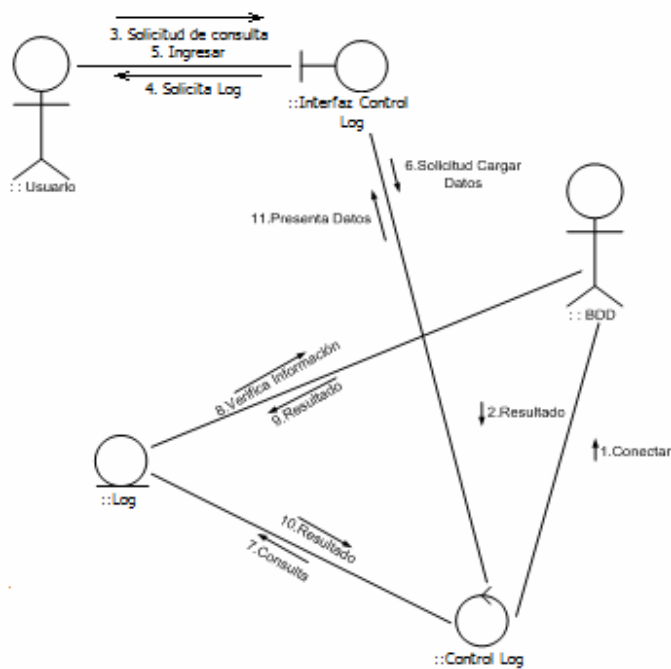
Fuente: los autores

Figura 37. Diagrama de secuencia Consultar Logs.



Fuente: los autores

Figura 38. Diagrama de colaboración Consulta Logs.



Fuente: los autores

10.7. REQUISITOS NO FUNCIONALES.

Tabla 28. Requisito no funcional 01 Herramientas de desarrollo

NFR 01	Herramienta de desarrollo
Versión	2010-02-21
Autores	Jhon Alexander Marín – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín – Anderson Lenis Pareja
Descripción	El sistema será creado de acuerdo a los siguientes requerimientos

Continúa en la página siguiente.

Continuación de la tabla anterior.

Requerimientos sobre la construcción	Plataforma de desarrollo	Herramientas	Versiones
	Java	Cake PHP	2008
	SQL Server	SQL Server	2005
Importancia	Alta		

Fuente: los autores

Tabla 29. Requisito no funcional 02 Visualización.

NFR - 02	Visualización
Versión	2010-03-19
Autores	Jhon Alexander Marín – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín – Anderson Lenis Pareja
Descripción	El sistema será creado de acuerdo a los siguientes requerimientos de la interfaz grafica de usuario.
Requerimientos la GUI	Menús de doble clic
Importancia	Alta

Fuente: los autores

Tabla 30. Requisito no funcional 03 Requerimientos de interacción.

NFR - 03	Requerimientos de interacción
Versión	2010-03-29
Autores	Jhon Alexander Marín – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín – Anderson Lenis Pareja
Descripción	El sistema será creado de acuerdo a los siguientes requerimientos de interacción.
Requerimientos de interacción	Computadores de escritorio.
Importancia	Alta

Fuente: los autores

Tabla 31. Requisito no funcional 04 Robustez.

NFR - 04	Robustez
Versión	2010-04-29
Autores	Jhon Alexander Marín – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín – Anderson Lenis Pareja
Descripción	El sistema será creado de acuerdo a los siguientes requerimientos mínimos: Capacidad mínima del disco duro de 40Gb, un procesador Pentium 4 de 2,6 GHz 1 Gb de memoria RAAM.
Importancia	Alta

Fuente: los autores

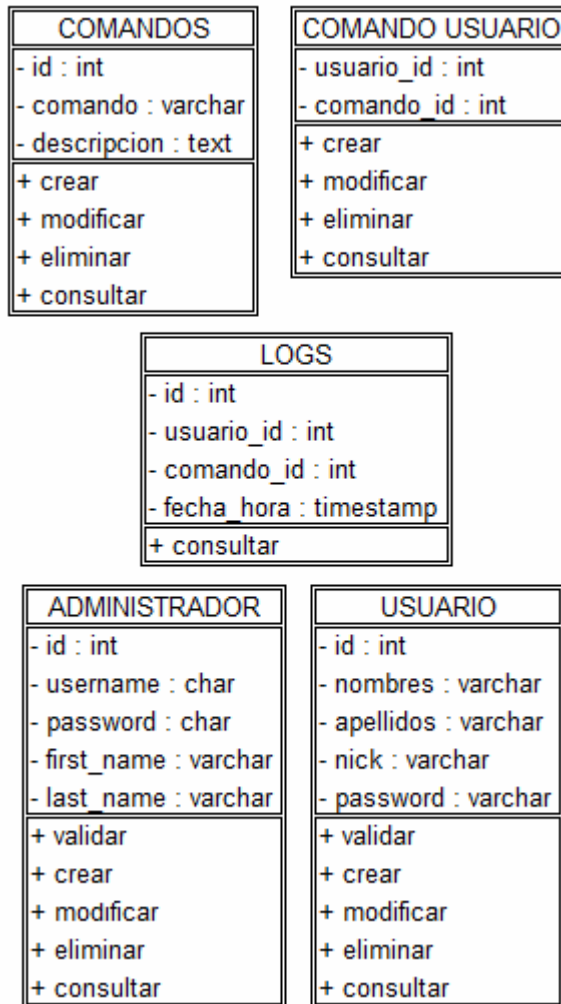
Tabla 32. Requisito no funcional 05 Control de acceso.

NFR - 05	Control de acceso
Versión	2010-04-21
Autores	Jhon Alexander Marín – Anderson Lenis Pareja
Fuentes	Jhon Alexander Marín – Anderson Lenis Pareja
Descripción	El sistema deberá verificar las contraseñas de los usuarios.
Importancia	Alta

Fuente: los autores

10.8 DIAGRAMA DE CLASES

Figura 39. Diagrama de clases.

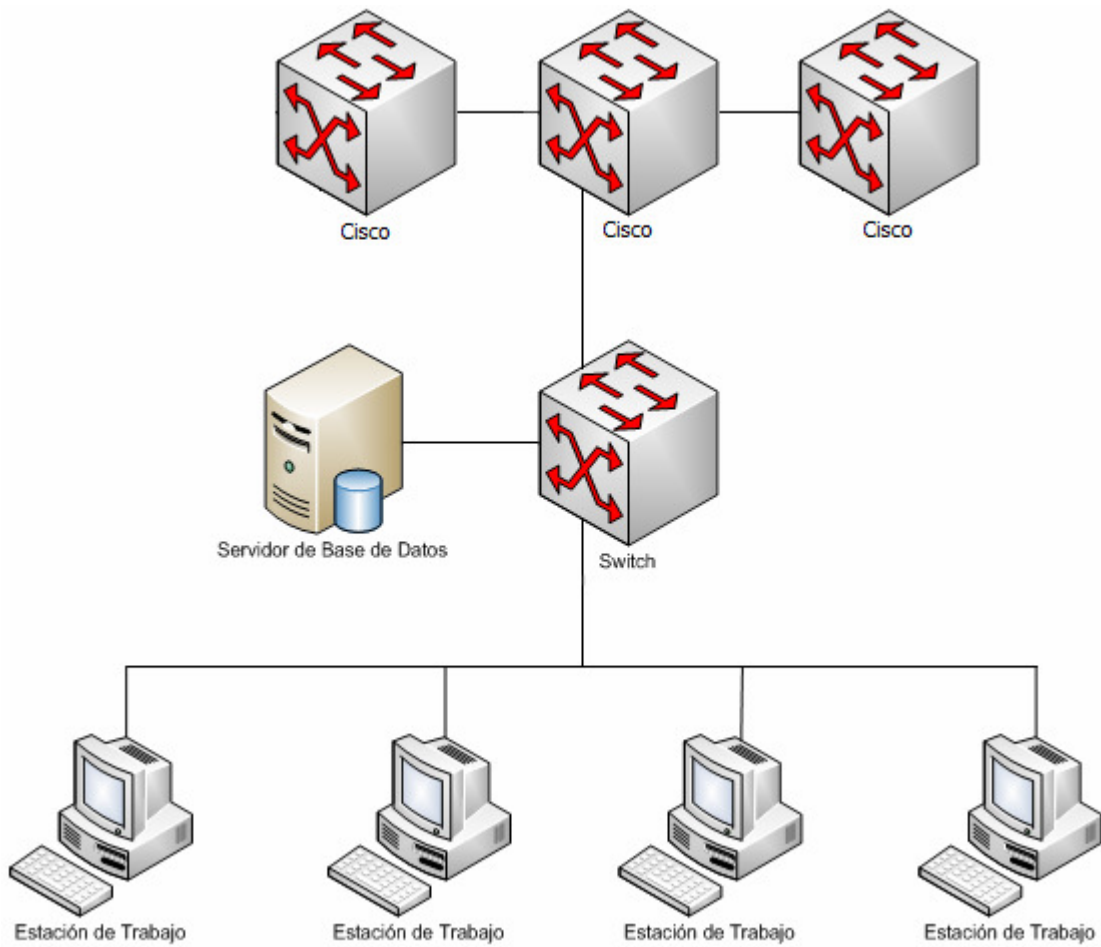


Fuente: los autores

11. DISEÑO.

11.1. DISEÑO ARQUITECTÓNICO.

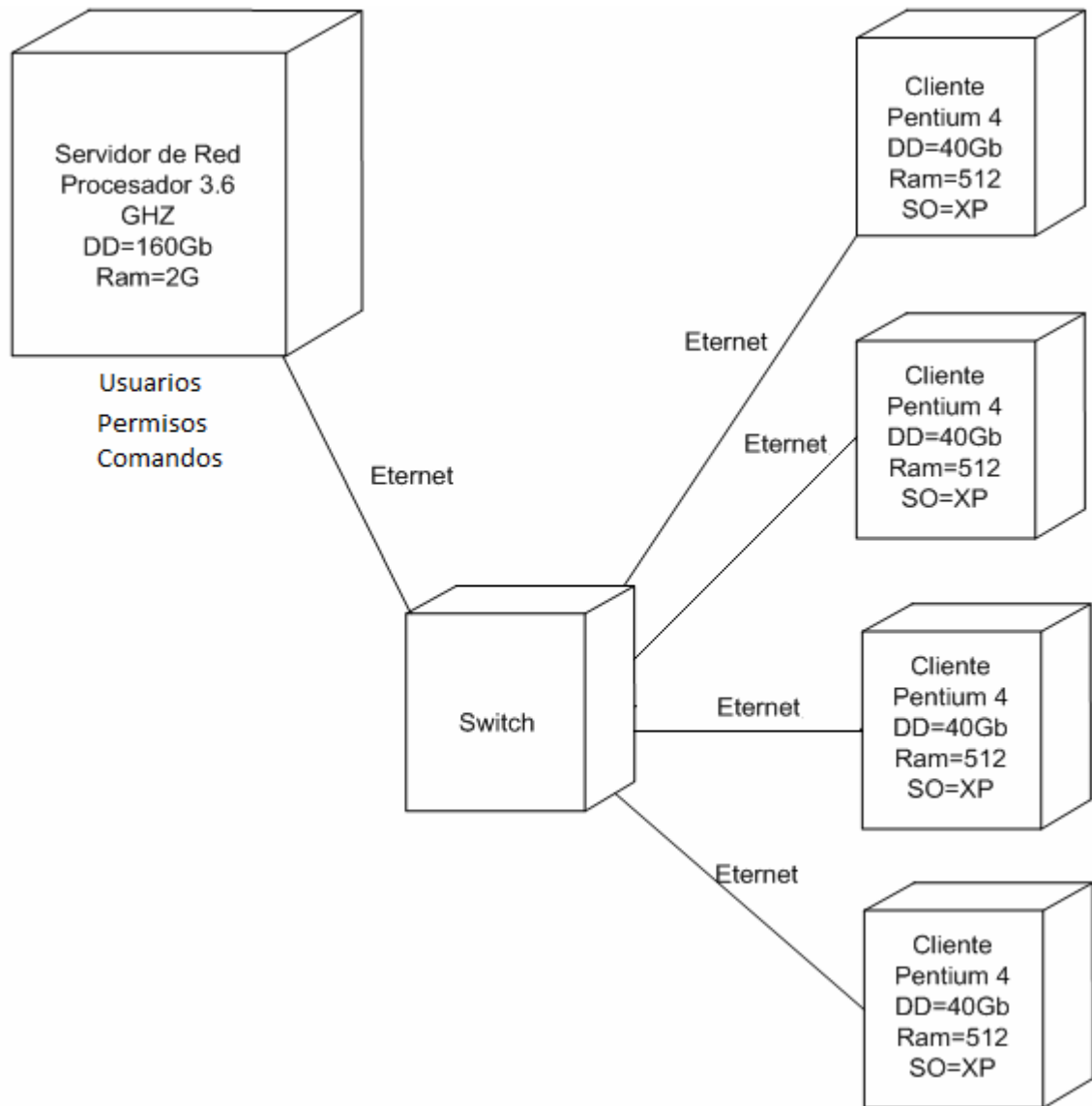
Figura 40. Diseño arquitectónico.



Fuente: los autores

11.2. DIAGRAMA DE DESPLIEGUE.

Figura 41. Diagrama de despliegue.



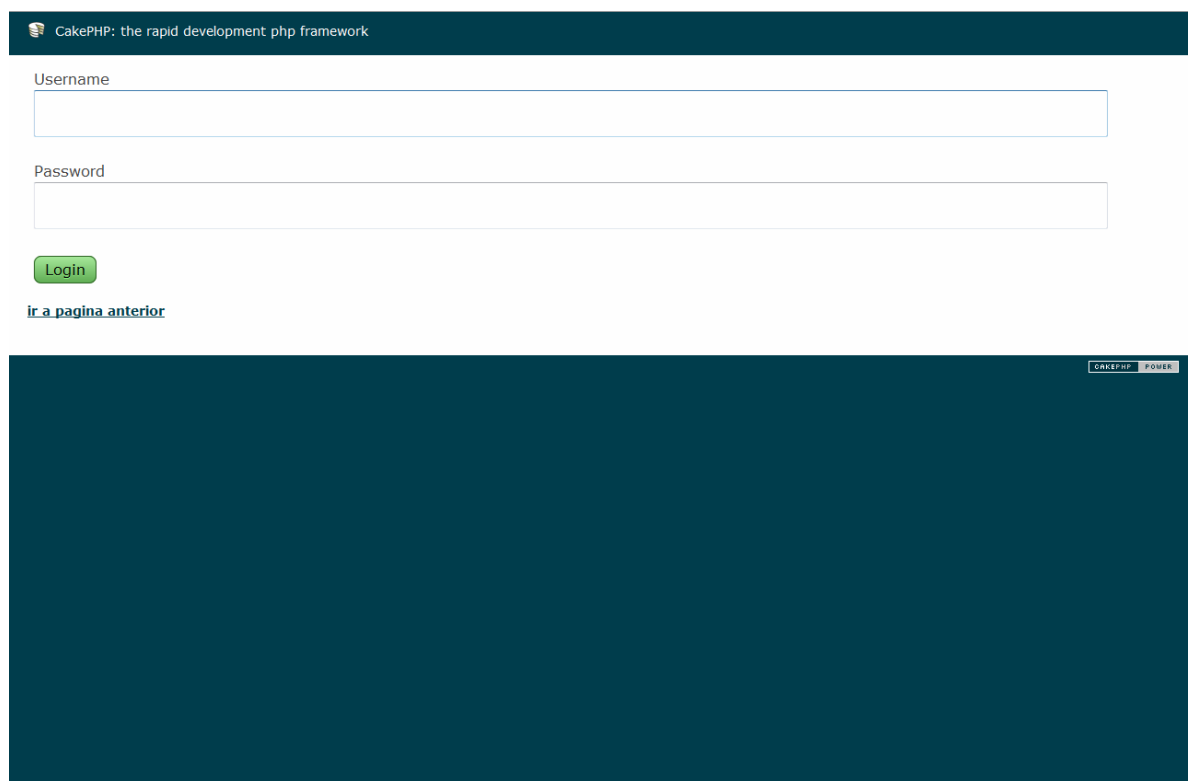
Fuente: los autores

11.3. PROTOTIPOS DE INTERFAZ GRÁFICA DE USUARIO.

11.3.1. Interfaz Validar Usuario.

Descripción: Esta pantalla se utiliza para la validación del usuario en el sistema, definiendo el perfil de acceso.

Figura 42. Interfaz Validar Usuario 1.



The image shows a screenshot of a web application interface for user validation. At the top, there is a dark teal header with the text "CakePHP: the rapid development php framework" and a small logo. Below the header, there are two input fields: "Username" and "Password". The "Username" field is a simple text input, and the "Password" field is a text input with a small eye icon on the right side. Below the password field is a green "Login" button. Underneath the button is a link that says "ir a pagina anterior". At the bottom of the page, there is a dark teal footer with the text "CAKEPHP POWER" in the top right corner.

Fuente: los autores

Figura 43. Interfaz Validar Usuario 2.

CakePHP: the rapid development php framework

Username
lenolenis

Password
●●●●●●●●

Login

[ir a pagina anterior](#)

CAKEPHP POWER

Fuente: los autores

Tabla 33. Evento interfaz Validar Usuario

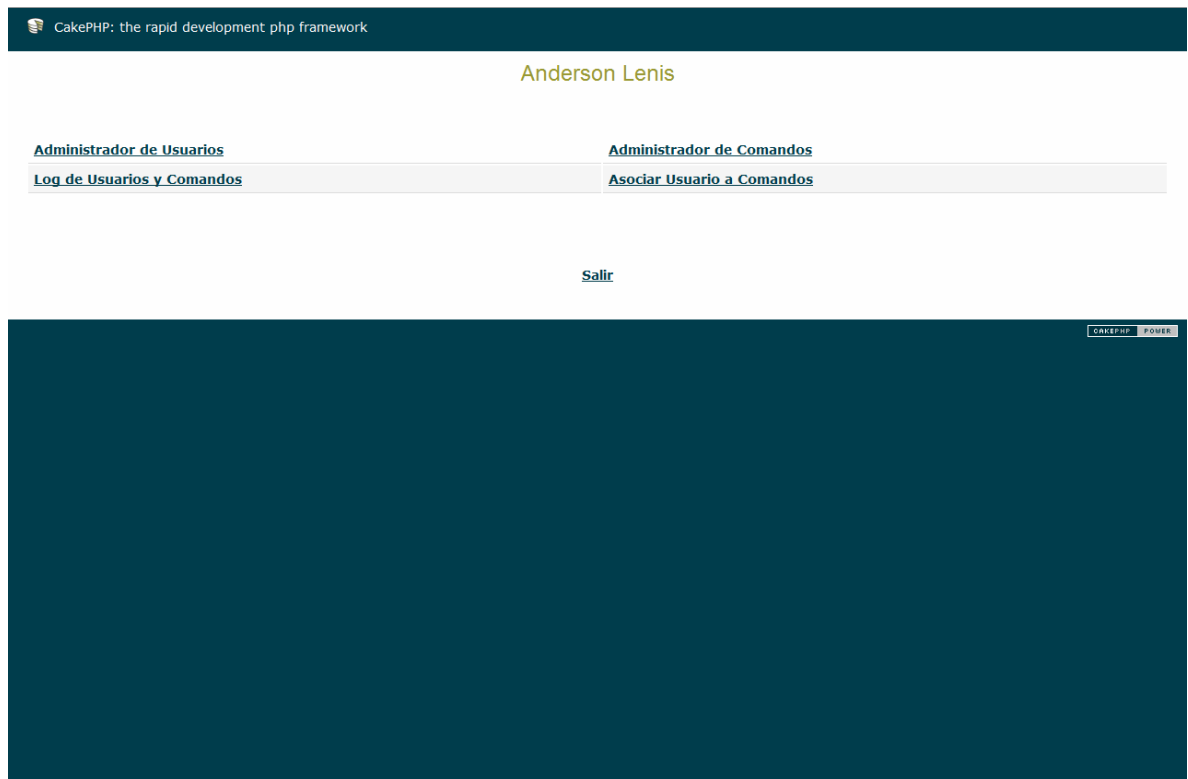
Evento	Acción
Aceptar	Valida el usuario.
Ir a la página anterior.	Cierra la ventana validar usuario y sale del sistema

Fuente: los autores

11.3.2. Interfaz Menú Principal.

Descripción: Esta pantalla se utiliza para acceder a las opciones del sistema.

Figura 44. Interfaz Menú Principal.



Fuente: los autores

Tabla 34. Evento interfaz Menú Principal.

Evento	Acción
Administrador de usuarios	Permite al administrador interactuar con los usuarios
Log de Usuarios y comandos	Permite al administrador observar los eventos registrados en el sistema

Continúa en la página siguiente.

Continuación de la tabla anterior.

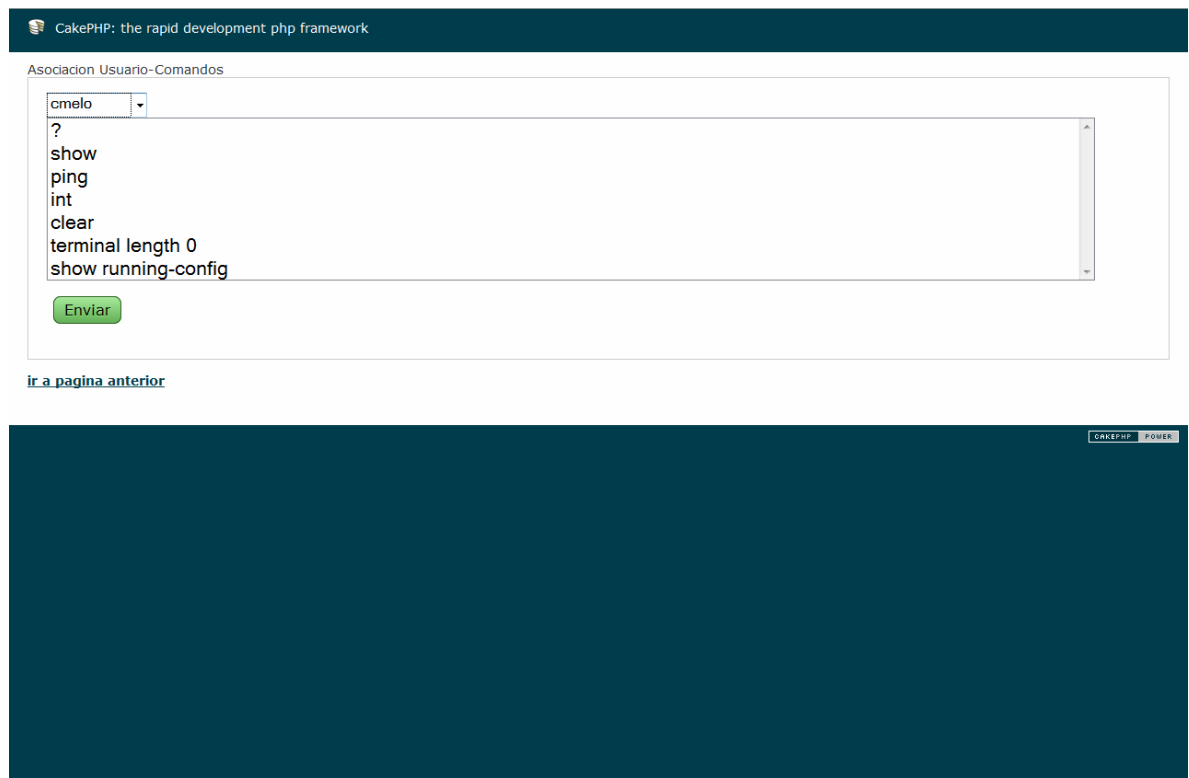
Administrador de comandos	Permite al administrador interactuar con los comandos
Asociar usuarios a comandos	Permite al administrador asociar los usuarios a los comandos

Fuente: los autores

11.3.3. Interfaz Asociar Usuario-Comando.

Descripción: Esta pantalla se utiliza para asociar los comandos a los usuarios.

Figura 45. Interfaz Asociar Usuario-Comando.



Fuente: los autores

Tabla 35. Eventos interfaz Asociar Usuario-Comando.

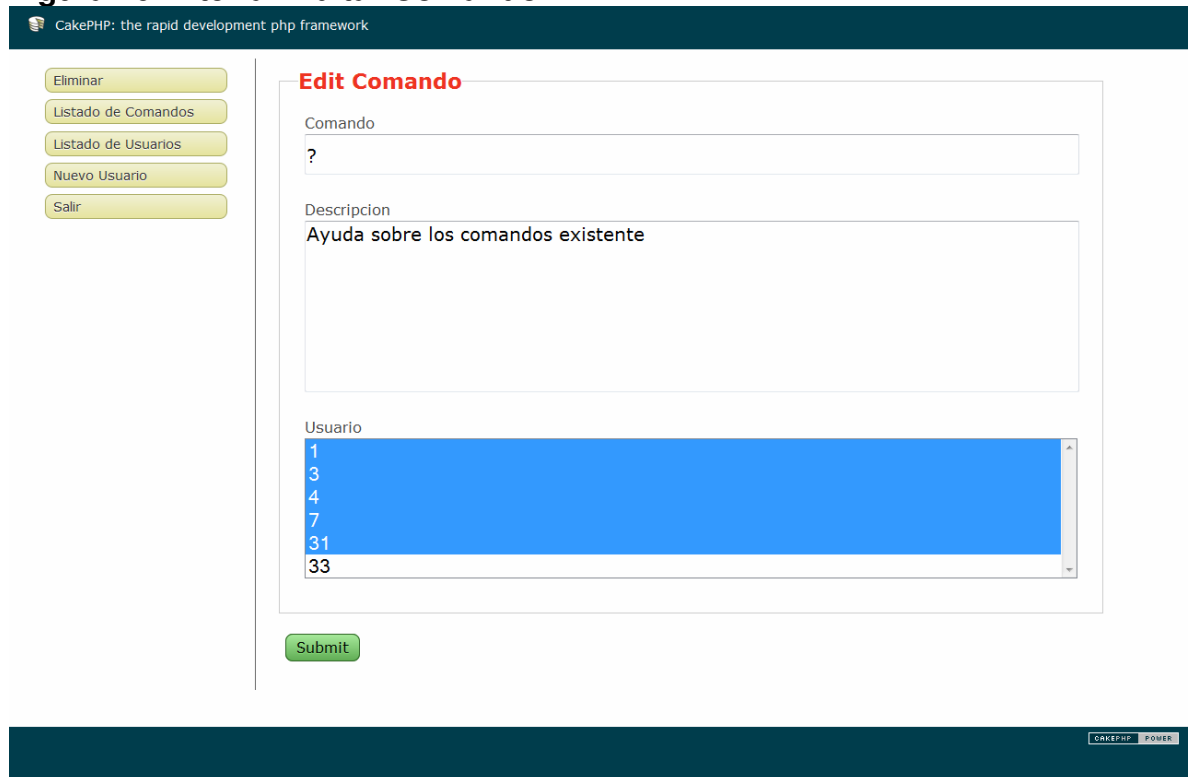
Evento	Acción
Asociar Usuarios a Comandos	Permite al Administrador asociar los comandos a los usuarios.
Enviar	Permite al administrador guardar los cambios que se realicen.

Fuente: los autores

11.3.4 Interfaz Editar Comando.

Descripción: Esta pantalla se utiliza para modificar la descripción de los comandos.

Figura 46. Interfaz Editar Comando.



Fuente: los autores

Tabla 36. Evento interfaz Editar Comando.

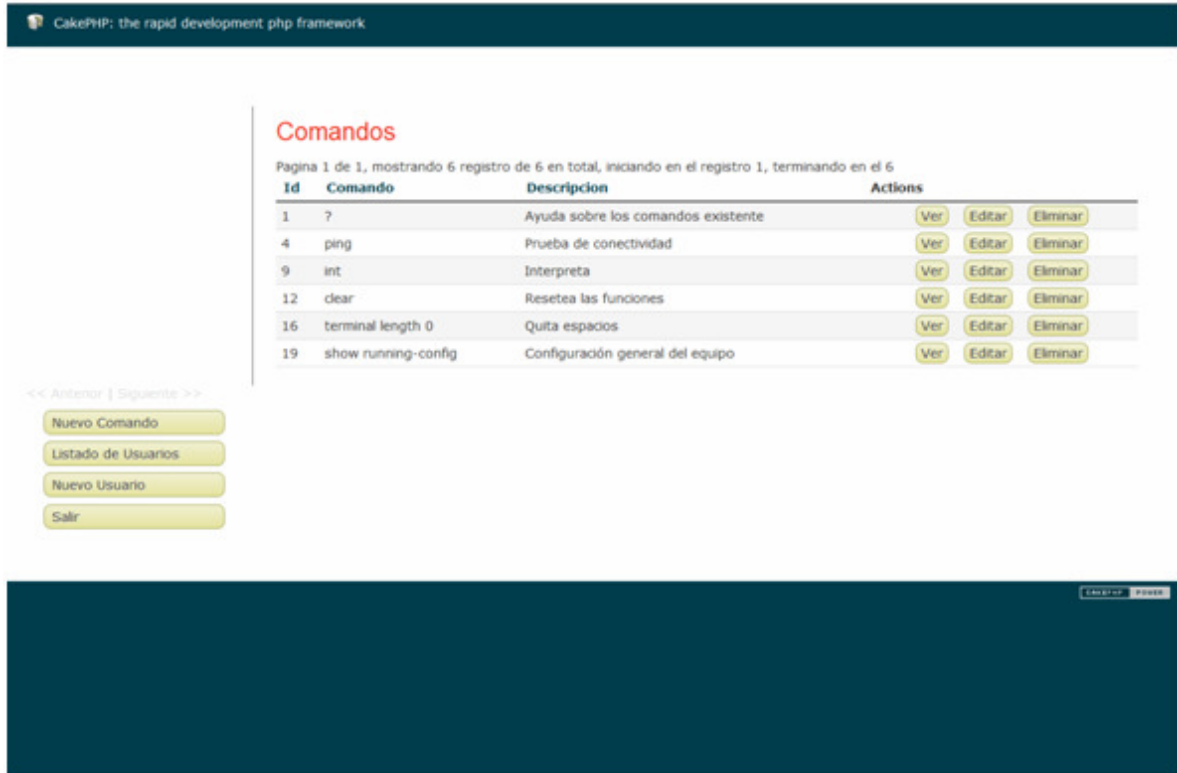
Evento	Acción
Eliminar	Permite al administrador eliminar comando
Listado de comando	Permite al Administrador listar los comandos existentes en el sistema.
Listado de Usuarios	Permite al Administrador listar los usuarios para asociarles un comando
Nuevo Usuario	Permite al Administrador crear un nuevo usuario.
Submit	Permite al administrador guardar los cambios que se realicen a los comandos.
Salir	Permite al Administrador salir del sistema

Fuente: los autores

11.3.5. Interfaz Comandos.

Descripción: Esta pantalla se utiliza para administrar los comandos.

Figura 47. Interfaz Comandos.



Fuente: los autores

Tabla 37. Evento interfaz Comandos.

Evento	Acción
Ver	Permite al administrador ver el Comando con sus características.
Editar	Permite al Administrador editar el comando.
Eliminar	Permite al Administrador eliminar el comando.
Salir	Permite al Administrador salir del sistema.

Fuente: los autores

11.3.6 Interfaz New Comando.

Descripción: Esta pantalla se utiliza para crear un nuevo comando.

Figura 48. Interfaz New Comando.

Fuente: los autores

Tabla 38. Evento interfaz New Comando.

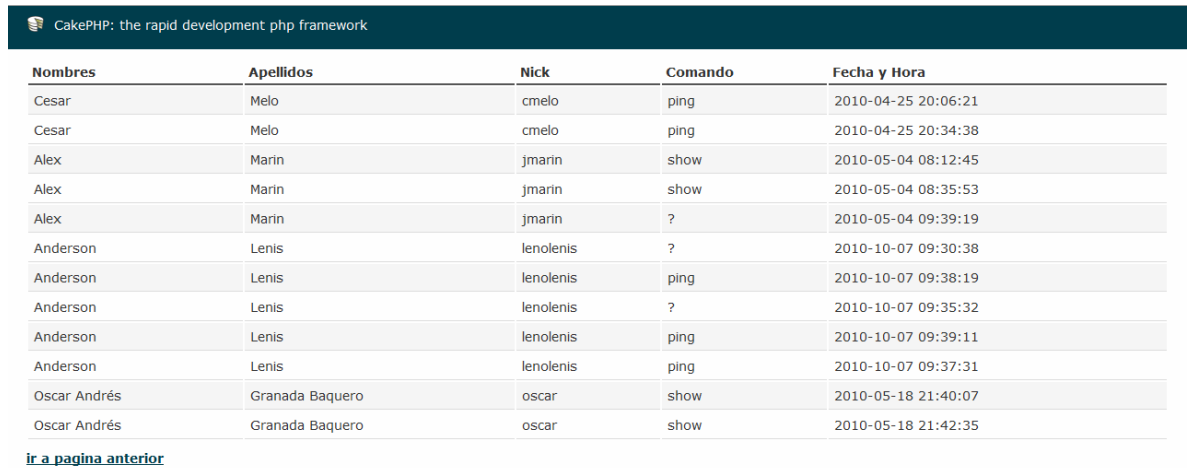
Evento	Acción
Submit	Permite al Administrador guardar los datos del nuevo comando

Fuente: los autores

11.3.7. Interfaz Log de Usuarios y Comandos.

Descripción: Esta pantalla se utiliza para Visualizar los logs de los usuarios.

Figura 49. Interfaz Log de Usuarios y Comandos.



Nombres	Apellidos	Nick	Comando	Fecha y Hora
Cesar	Melo	cmelo	ping	2010-04-25 20:06:21
Cesar	Melo	cmelo	ping	2010-04-25 20:34:38
Alex	Marin	jmarin	show	2010-05-04 08:12:45
Alex	Marin	jmarin	show	2010-05-04 08:35:53
Alex	Marin	jmarin	?	2010-05-04 09:39:19
Anderson	Lenis	lenolenis	?	2010-10-07 09:30:38
Anderson	Lenis	lenolenis	ping	2010-10-07 09:38:19
Anderson	Lenis	lenolenis	?	2010-10-07 09:35:32
Anderson	Lenis	lenolenis	ping	2010-10-07 09:39:11
Anderson	Lenis	lenolenis	ping	2010-10-07 09:37:31
Oscar Andrés	Granada Baquero	oscar	show	2010-05-18 21:40:07
Oscar Andrés	Granada Baquero	oscar	show	2010-05-18 21:42:35

[ir a pagina anterior](#)



Fuente: los autores

Tabla 39. Evento interfaz Log de Usuarios y Comandos.

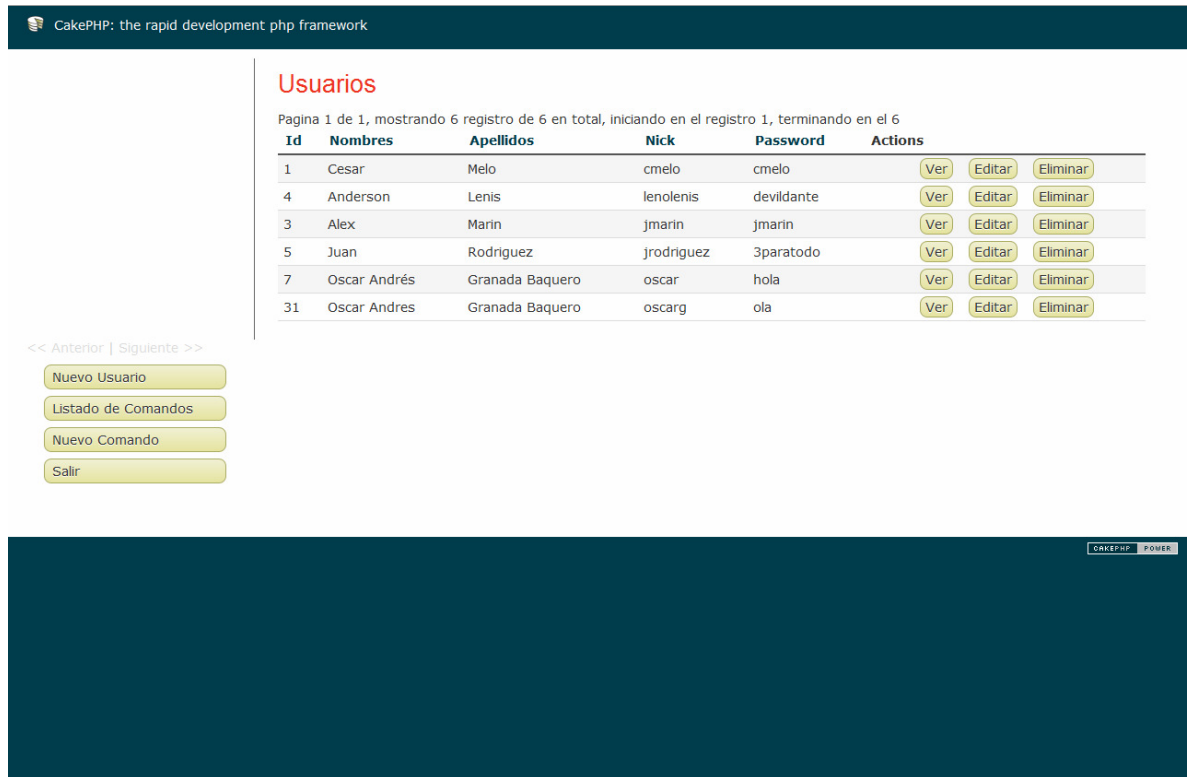
Evento	Acción
Ir a la pagina anterior	Permite al Administrador ir al menú principal

Fuente: los autores

11.3.8. Interfaz Usuarios.

Descripción: Esta pantalla se utiliza para Visualizar los usuarios.

Figura 50. Interfaz Usuarios.



Fuente: los autores

Tabla 40. Evento interfaz Usuarios.

EVENTO	ACCIÓN
Ver	Permite al administrador ver las características de los usuarios
Editar	Permite al administrador editar las opciones y características de los usuarios.

Continúa en la página siguiente.

Continuación de la tabla anterior.

Eliminar	Permite al administrador eliminar los usuarios.
Salir	Permite al administrador salir del menú principal

Fuente: los autores

11.3.9. Interfaz Crear Usuarios.

Descripción: Esta pantalla se utiliza para crear nuevos usuarios.

Figura 51. Interfaz Crear Usuario 1.

Nuevo Usuario

Nombres

Apellidos

Nick

Password

Comando
?
show
ping
int
clear
terminal length 0
show running-config

Enviar

Salir

Fuente: los autores

Figura 52. Interfaz Crear Usuario 2.

CakePHP: the rapid development php framework

Nuevo Usuario

Nombres
Francisco

Apellidos
Flores

Nick
pacho

Password
•••••

Comando
?
show
ping
int
clear
terminal length 0
show running-config

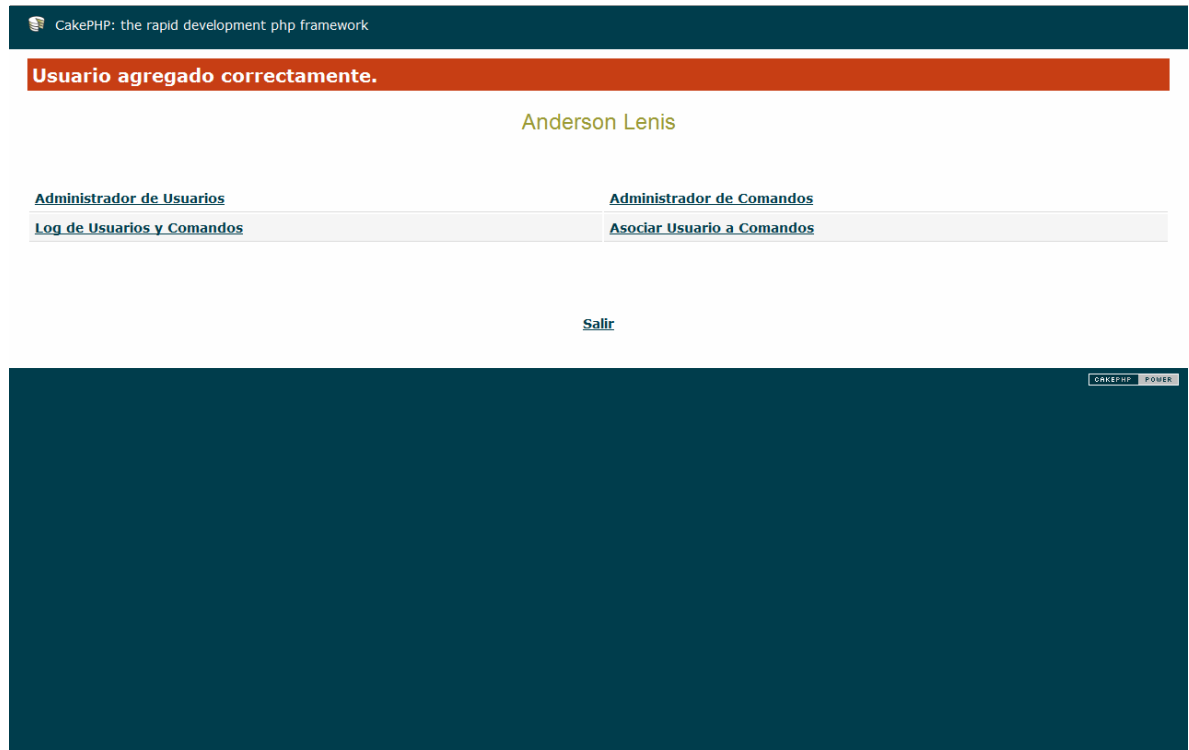
Enviar

[Salir](#)

CakePHP POWER!

Fuente: los autores

Figura 53. Interfaz Crear Usuario 3.



Fuente: los autores

Tabla 41. Evento interfaz Crear Usuario.

EVENTO	ACCIÓN
Enviar	Permite al administrador guardar los datos de los usuarios creados.
Salir	Permite al administrador volver al menú principal

Fuente: los autores

11.3.10. Interfaz Editar Usuarios.

Descripción: Esta pantalla se utiliza para Editar los Usuarios.

Figura 54. Interfaz Editar Usuario.

CakePHP: the rapid development php framework

Eliminar
Listado de Usuarios
Listado de Comandos
Nuevo Comando
Salir

Edit Usuario

Nombres
Cesar

Apellidos
Melo

Nick
cmelo

Password
•••••

Comando
1
4
9
12
14
16
19

Submit

CAKEPHP POWER

Fuente: los autores

Tabla 42. Evento interfaz Editar Usuarios.

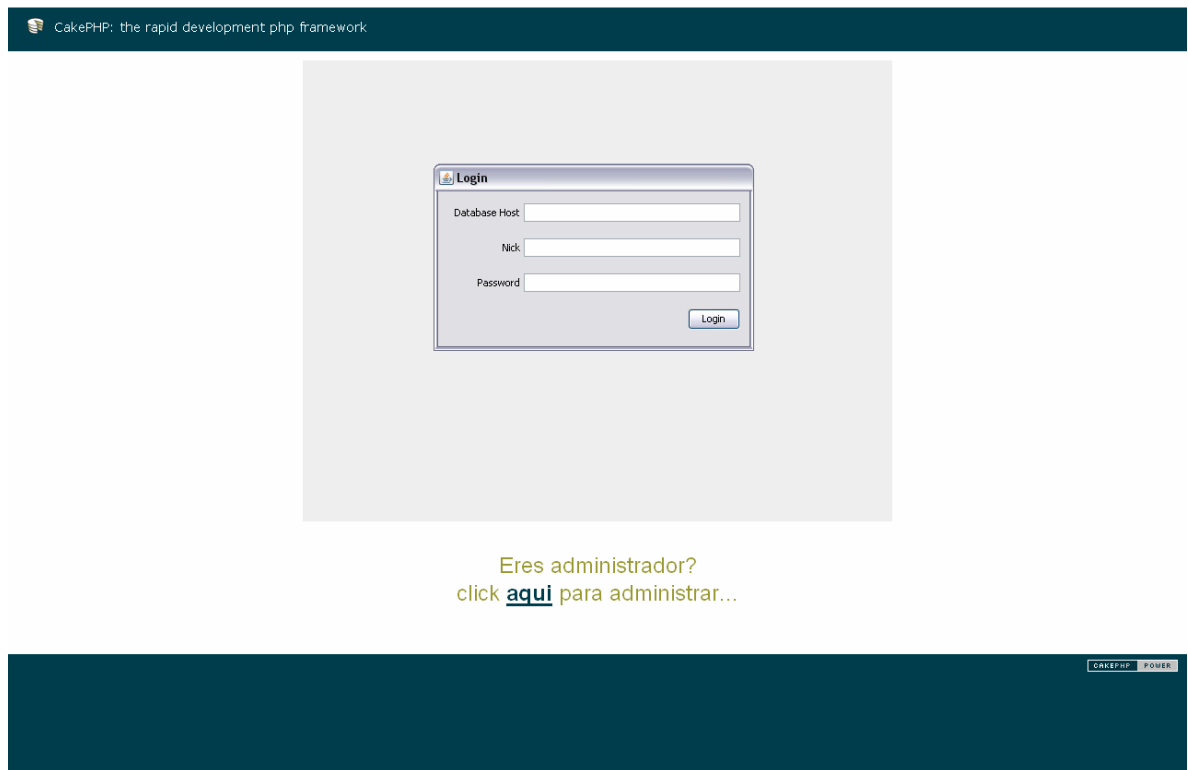
EVENTO	ACCIÓN
Submid	Permite al administrador guardar los datos de los usuarios modificados.
Salir	Permite al administrador volver al menú principal

Fuente: los autores

11.3.11. Interfaz inicio de conexión.

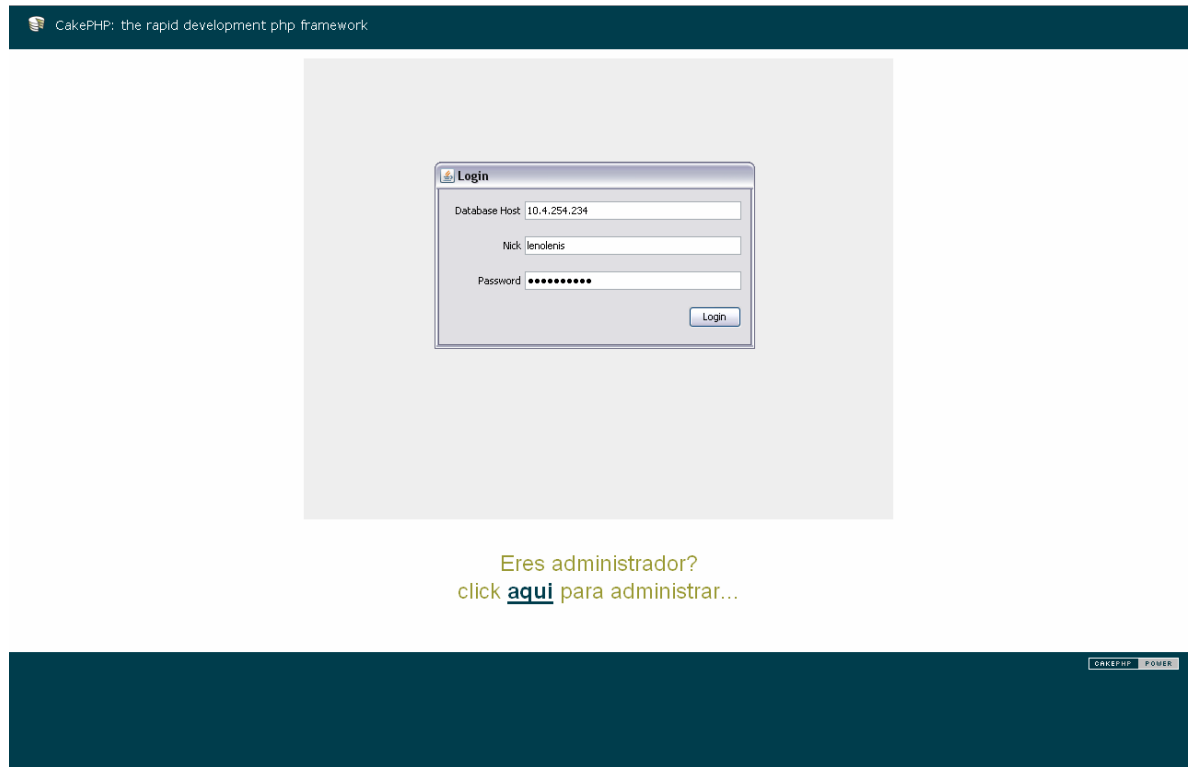
Descripción: Esta pantalla se utiliza para ingresar a los Swich o Routers Cisco.

Figura 55. Interfaz inicio de conexión 1.



Fuente: los autores

Figura 56. Interfaz inicio de conexión 2.



Fuente: los autores

Tabla 43. Evento interfaz Inicio de Conexión.

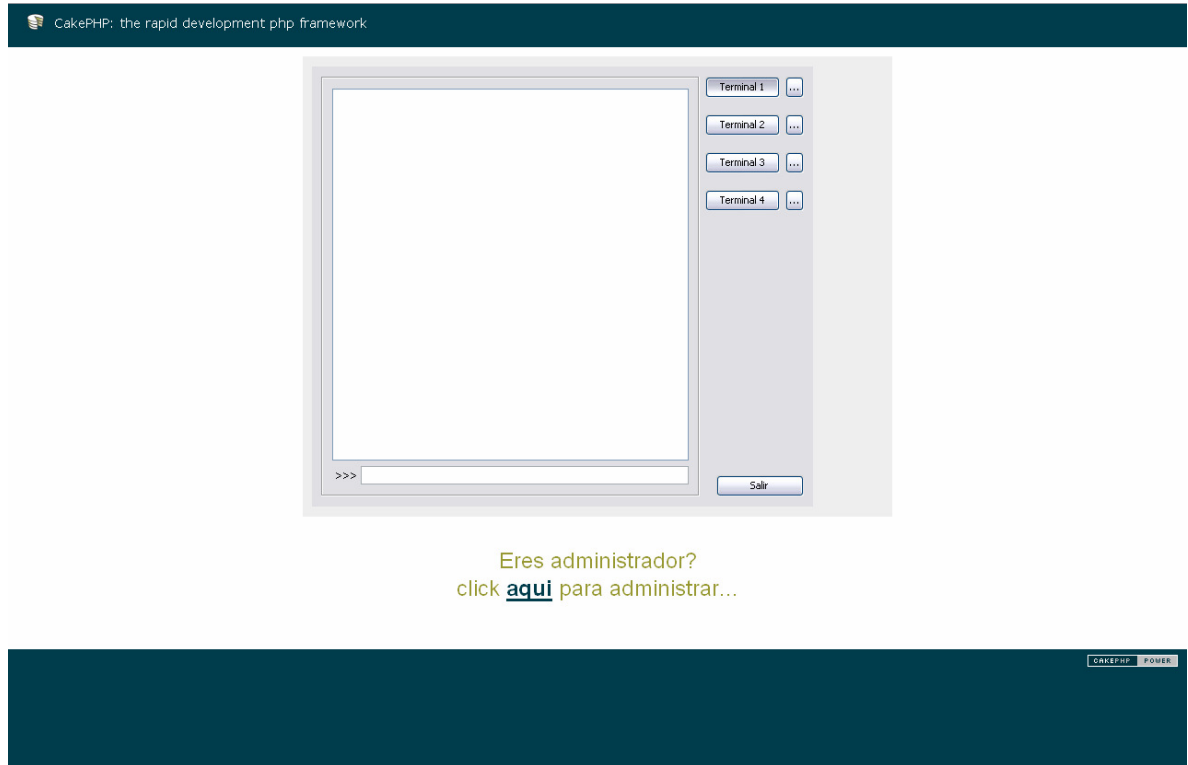
EVENTO	ACCIÓN
Loguin	Permite al usuario ingresar a los Switch o Router Cisco.
Cambiar usuario	Permite al usuario cambiar a modo Administrador

Fuente: los autores

11.3.12. Interfaz Web para acceder al cli de los equipos Cisco.

Descripción: Esta pantalla se utiliza para gestionar los equipos Cisco mediante líneas de comandos.

Figura 57. Interfaz Web para acceder al cli.



Fuente: los autores

Tabla 44. Evento interfaz Web para acceder al cli de los equipos Cisco.

EVENTO	ACCIÓN
Terminal 1	Permite al usuario acceder a la Terminal 1.
Terminal 2	Permite al usuario acceder a la Terminal 2.

Continúa en la página siguiente.

Continuación de la tabla anterior.

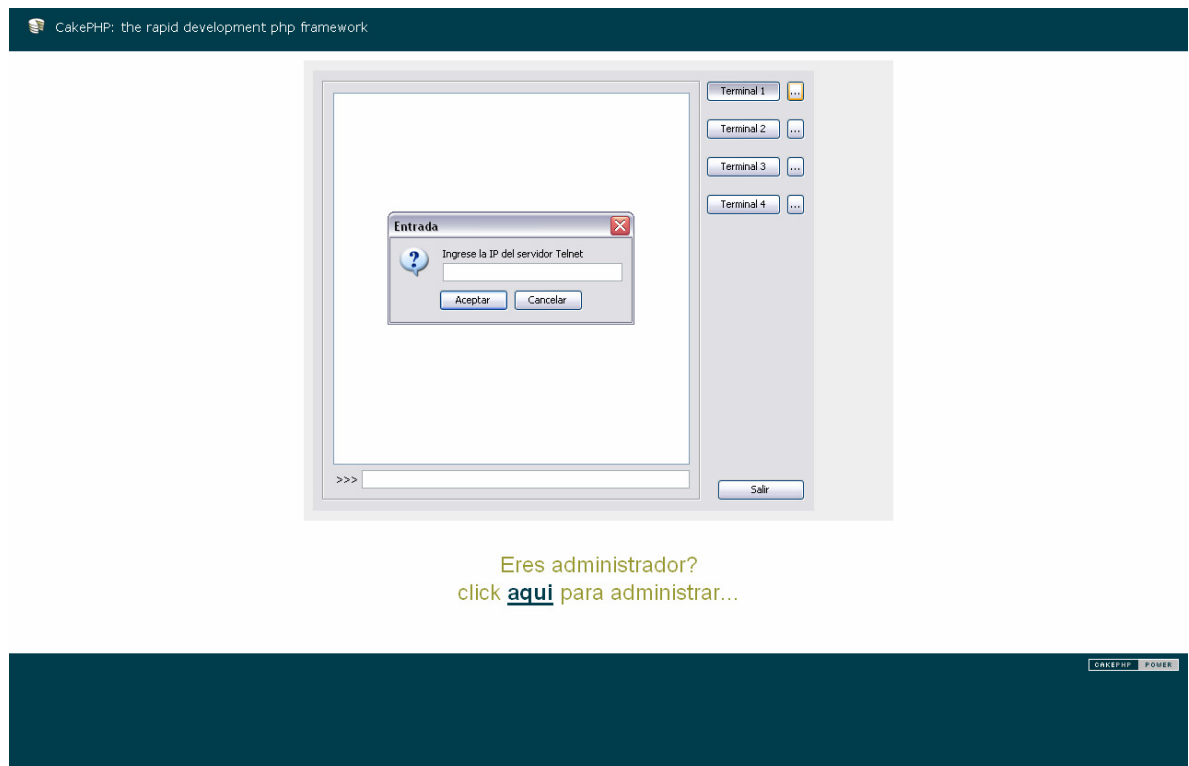
Terminal 3	Permite al usuario acceder a la Terminal 3.
Terminal 4	Permite al usuario acceder a la Terminal 4.
Salir	Permite al usuario salir del sistema.

Fuente: los autores

11.3.13. Interfaz dirección IP.

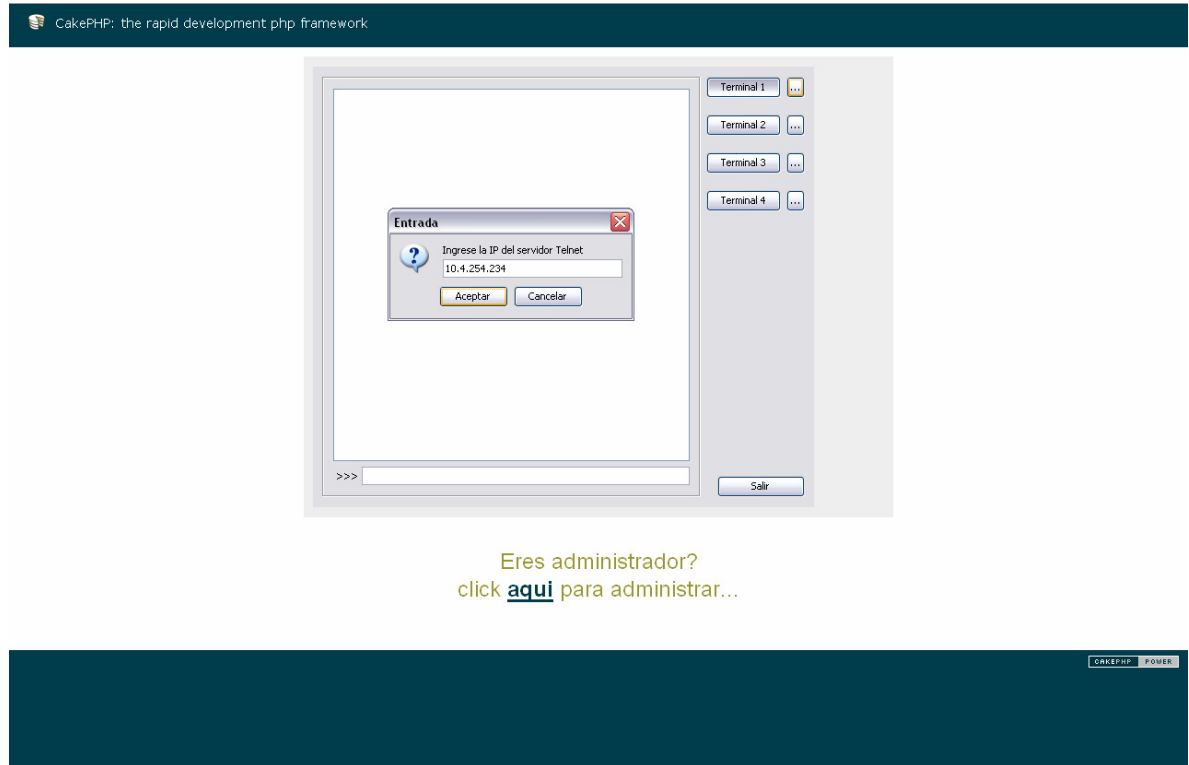
Descripción: Esta pantalla se utiliza para ingresar la dirección IP del equipo Cisco al que se desea ingresar.

Figura 58. Interfaz dirección IP 1.



Fuente: los autores

Figura 59. Interfaz dirección IP 2.



Eres administrador?
click [aqui](#) para administrar...

Fuente: los autores

Tabla 45. Evento interfaz Dirección IP.

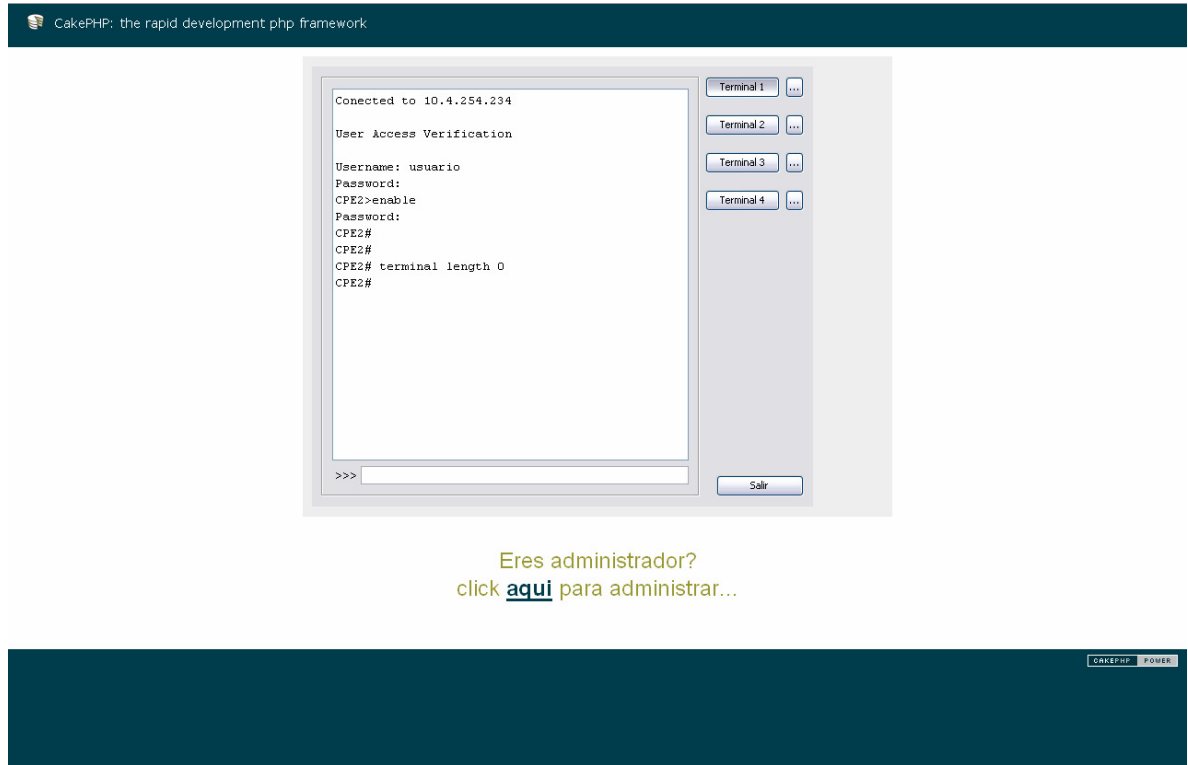
EVENTO	ACCIÓN
Aceptar	Permite al usuario ingresar la IP del equipo Cisco que se desea acceder.
Cancelar	Permite al usuario cancelar la acción de ingresar la IP.

Fuente: los autores

11.3.14. Interfaz sesión cli.

Descripción: Esta pantalla se utiliza ejecutar los comandos dentro la sesión iniciada de los equipos Cisco.

Figura 60. Interfaz sesión cli.



Fuente: los autores

Tabla 46. Evento interfaz Sesión cli.

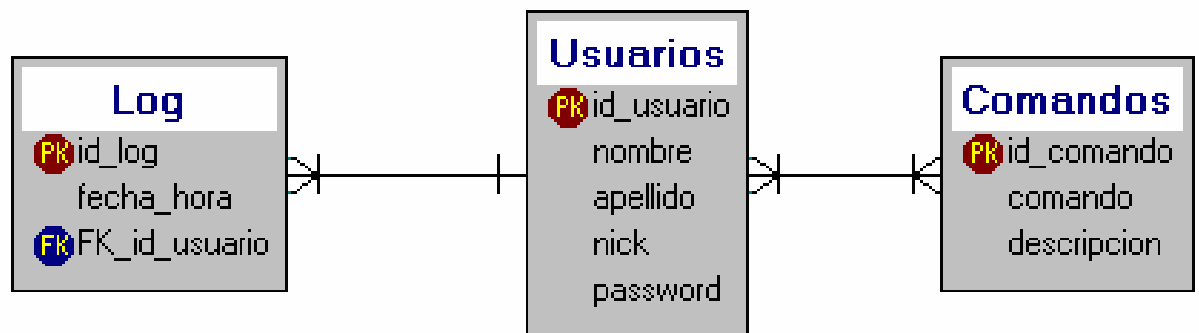
EVENTO	ACCIÓN
Terminal	Permite al usuario intercambiar entre terminales.
Salir	Permite al usuario salir de la aplicación.

Fuente: los autores

11.4 DISEÑO DE LA BASE DE DATOS.

11.4.1. Modelo Entidad Relación.

Figura 61. Diagrama Entidad Relación.



Fuente: los autores

11.4.2. Descripción de la base de datos.

Tabla 47. Log

Log					
Descripción: esta tabla contiene la información correspondiente a los Log					
Atributos:					
PK	Attribute Name	Data Type	Not Null	Unique	Descripción
Yes	Id_log	NVARC HAR (30)	Yes	No	Este campo contiene la información correspondiente al id del Log
No	Fecha_hora	NVARC HAR (20)	Yes	No	Este campo contiene la información correspondiente a la fecha y hora
Relaciones					
Relationship Name		Type	Parent	Child	Descripción
Id_log		identifying	Usuarios	Id_usuarios	Esta relación resulta entre la tabla Log y la tabla Usuarios.

Fuente: los autores

Tabla 48. Comandos.

Comandos					
Descripción: Esta tabla contiene la información correspondiente a los comandos.					
Atributos:					
PK	Attribute Name	Data Type	Not Null	Unique	Descripción
Yes	Id_comando	NVARCHAR (40)	Yes	No	Este campo contiene la información correspondiente al id del comando.
No	Comando	NVARCHAR (40)	Yes	No	Este campo contiene la información correspondiente al comando.
No	Descripción	NVARCHAR (40)	Yes	No	Este campo contiene la información correspondiente a la descripción de l comando.
Relaciones:					
Relationship Name	Type	Parent	Child	Description	
id_comando	Identifying	usuarios	Id_usuarios	Esta relación resulta entre la tabla comando y la tabla usuarios.	
id_comando	Identifying	comandos_usuarios	Id_comando	Esta relación resulta entre la tabla comando y la tabla comando_usuarios.	

Fuente: los autores.

Tabla 49. Usuarios.

Usuarios					
Descripción: Esta tabla contiene la información correspondiente a los usuarios.					
Atributos:					
PK	Attribute Name	Data Type	Not Null	Unique	Descripción
Yes	Id_usuario	NVARCHAR (40)	Yes	Yes	Este campo contiene la información correspondiente al id del usuario.
No	nombre	NVARCHAR (40)	Yes	No	Este campo contiene la información correspondiente al nombre del usuario.
No	apellido	NVARCHAR (40)	No	No	Este campo contiene la información correspondiente al apellido del usuario.
No	Nick	NVARCHAR (40)	No	Yes	Este campo contiene la información correspondiente al nick del usuario.
No	Password	NVARCHAR (40)	No	No	Este campo contiene la información correspondiente al password del usuario.

Continúa en la página siguiente.

Continuación de la tabla anterior.

Relaciones:				
Relationship				
Name	Type	Parent	Child	Descripción
id_usuario	identifying	Log	FK_id_usuario	Esta relación resulta entre la tabla Usuario y la tabla Log.
id_usuario	identifying	Comando	FK_id_usuario	Esta relación resulta entre la tabla Usuario y la tabla. Comando_usuario

Fuente: los autores

Tabla 50. Comandos_Usuarios.

Comandos_usuarios				
Descripción: Esta tabla contiene la relación que existe entre los usuarios y los comandos.				
Relaciones:				
Relationship				
Name	Type	Parent	Child	Descripción
FK_id_usuario	identifying	Comandos_usuarios	FK_id_comando	Esta relación resulta entre la tabla Usuario y la tabla Comandos.

Fuente: los autores

11.4.3. Lenguaje de definición de datos

SCRIPT

```
/*=====*/  
/* Tables */  
/*=====*/
```

```
CREATE TABLE Comandos(  
id_comando varchar CONSTRAINT PK_Comandos1 PRIMARY KEY,  
comando varchar,  
descripcion varchar,  
CONSTRAINT UC_Comandos1 UNIQUE(id_comando))  
go
```

```
CREATE TABLE Usuarios(  
id_usuario varchar CONSTRAINT PK_Usuarios1 PRIMARY KEY,  
nombre varchar,  
apellido varchar,  
nick varchar,  
password varchar,  
CONSTRAINT UC_Usuarios1 UNIQUE(id_usuario))  
go
```

```
CREATE TABLE comandos_usuarios(  
FK_id_usuario int NOT NULL,  
FK_id_comando int NOT NULL)  
go
```

```
CREATE TABLE Usuarios_Comandos(  

```

```

FK_id_comando int NOT NULL,
FK_id_usuario int NOT NULL,
CONSTRAINT          PK_Usuarios_Comandos1          PRIMARY          KEY
(FK_id_comando,FK_id_usuario))
go

```

```

CREATE TABLE Log(
id_log int CONSTRAINT PK_Log1 PRIMARY KEY,
fecha_hora varchar,
FK_id_usuario varchar NOT NULL,
CONSTRAINT UC_Log1 UNIQUE(id_log))
go

```

```

/*=====*/
/* Foreign Keys */
/*=====*/

```

```

ALTER TABLE comandos_usuarios
ADD CONSTRAINT FK_comandos_usuarios_1
FOREIGN KEY (FK_id_comando) REFERENCES Comandos (id_comando)
go

```

```

ALTER TABLE comandos_usuarios
ADD CONSTRAINT FK_comandos_usuarios_2
FOREIGN KEY (FK_id_usuario) REFERENCES Usuarios (id_usuario)
go

```

```

ALTER TABLE Usuarios_Comandos
ADD CONSTRAINT FK_Usuarios_Comandos_1
FOREIGN KEY (FK_id_comando) REFERENCES Comandos (id_comando)

```

```
go
ALTER TABLE Usuarios_Comandos
ADD CONSTRAINT FK_Usuarios_Comandos_2
FOREIGN KEY (FK_id_usuario) REFERENCES Usuarios (id_usuario)
go
```

```
ALTER TABLE Log
ADD CONSTRAINT FK_Log_1
FOREIGN KEY (FK_id_usuario) REFERENCES Usuarios (id_usuario)
go
```

12. CONCLUSIONES

- La herramienta informática llegó a su etapa de implementación en equipos de soluciones para clientes corporativos para verificar su funcionamiento y estabilidad, quedando pendiente la aprobación de la implantación hacia los equipos de CORE por parte del personal de Agregación & Backbone de la Empresa de Telecomunicaciones de Pereira.
- El apoyo de una herramienta informática es de gran ayuda en la optimización del proceso de monitoreo de sesiones hacia los equipos Cisco, de esta manera evitamos problemas de desempeño en los equipos que se desean monitorear.
- Una de las ventajas de haber seleccionado la herramienta JAVA como lenguaje de programación es que es bastante simple y poderoso, es seguro, es independiente de la arquitectura del hardware, es rápido, fácil de aprender, entre otras, lo cual nos permitió plantear una solución ágil y fácil de entender para el usuario final.
- El uso de framework nos facilitó mucho el desarrollo de la aplicación para administrar todo lo que tiene ver con los usuarios, comandos, logs, etc, gracias a que este nos liberó mucha carga de escritura de código.
- Una ventaja de usar UML es permitir que los cambios en los requerimientos del sistema no causen grandes modificaciones en los diagramas, permitiendo que la herramienta informática sea más fácil de mantener.

- Este es un logro más obtenido dentro del campo de las telecomunicaciones lo cual fue una de las prioridades desde un principio de su desarrollo e implementación.
- Por lo general esta tesis se realizó para tener una perspectiva general en el campo de las telecomunicaciones acerca de los modelos de trabajo más usuales que se presentan cuando interactúan con equipos CISCO y las necesidades que se deben de tener en cuenta con cada uno de ellos.

13. RECOMENDACIONES

Para futuros proyectos de desarrollo de software se recomienda tener en cuenta las metodologías aplicadas en este trabajo para así facilitar los procesos de análisis y diseño de los mismos.

Al momento de la implantación del sistema de información se recomienda tener en cuenta el diseño arquitectónico y los requisitos no funcionales plasmados en este documento.

Los ingenieros de software deben tener conocimiento detallado del objeto de estudio de manera que al momento del levantamiento de la información se puedan resolver las diferencias con el cliente.

Usar metodologías en el análisis del sistema de información que permitan generar cambios sin que se causen grandes alteraciones en el diseño.

El mantenimiento de la herramienta informática hace parte del ciclo de vida del software permitiendo la mejora y optimización de la misma. Por tanto es de gran importancia que se tengan en cuenta las recomendaciones expresadas por sus desarrolladores para realizar este proceso.

14. BIBLIOGRAFÍA.

BERNÁRDEZ JIMÉNEZ, Beatriz y DURÁN TORO, Amador “Metodología para la Elicitación de requisitos de sistemas software”. Internet:

(www.lsi.us.es/informes/lsi-2000-10.pdf).

BRUEGGE, Bernd y DUTOIT, Allen H. ingeniería de software orientada a objetos. 1ed. Editorial Prentice Hall. 2002.

CEBALLOS SIERRA, Francisco Javier. Microsoft Visual Basic.Net lenguaje y aplicaciones. 2ed. Editorial Alfaomega Ra.ma.2007.

COLOMBIA. El Congreso de la República. Ley 1273. (5 enero de 2009). Por medio de la cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado - denominado de la protección de la información y de los datos. Internet: (http://www.fiscalia.gov.co/justiciapaz/Documentos/Normativa/LEY_975_250705.htm)

COLOMBIA. MINISTERIO DE LA PROTECCIÓN SOCIAL, “Modelo de evaluación de la relación Docencia-Servicio”. Internet:

<http://www.minproteccionsocial.gov.co/VbeContent/library/documents/DocNewsNo14546DocumentNo1426.PDF>)

PRESSMAN, Roger S. ingeniería del software un enfoque práctico. 6ed. Editorial Mc Graw Hill. 2005.

SOMMERVILLE, Ian. Ingeniería del software. 7ed. Editorial Pearson Addison Wesley. 2007.

WEIZENFELD, Alfredo. Ingeniería de software orientada a objetos con UML. Editorial Thomson. 2005.

15. GLOSARIO.

RED DE CORE: Es el conjunto de equipos principales de una red de telecomunicaciones donde se converge todo el tráfico que se esté transportando.

TELNET: Telecommunication Network (Telecomunicación de Red), es el nombre de uno de los protocolos de red, el cual nos permite acceder mediante una red a un equipo en especial y así controlarlo de manera remota como si se estuviera en frente de esta.

SWITCH: Es un dispositivo con el cual se pueden realizar interconexiones entre diferentes redes de computadores, operando a nivel de capa 2 (Enlace de datos) del modelo OSI.

ROUTER: Es un dispositivo con el cual se pueden realizar interconexiones entre diferentes redes de computadores, operando a nivel de capa 3 (Nivel de Red).

BRAS: Es un dispositivo situado en la categoría de Router, pero con la característica principal de manejar todo el acceso de usuarios que están conectados mediante la tecnología DSL (Línea Digital del Subscriptor) ó para mayor comprensión, por la red de telefonía convencional.

CMTS: Es un dispositivo situado en la categoría de Router, pero con la característica principal de manejar todo el acceso de usuarios que están conectados mediante la tecnología de Cable (HFC – red híbrida de fibra óptica y cable coaxial) ó para mayor comprensión, por la misma red de televisión.

FIBRA OPTICA: Es un medio de transmisión que se emplea especialmente en las redes de datos. Las fibras permiten enviar gran cantidad de datos e información a grandes distancias con velocidades muy similares a las de radio y/o cable.

USUARIO: Se denomina de esta manera, al nombre clave con el cual se le da acceso a una persona a un sistema de información ó equipo.

CONTRASEÑA: Se denomina así al método de seguridad que se utiliza para identificar a un usuario. Es frecuente su uso en redes. Se utiliza para dar acceso a personas con determinados permisos (perfil).

DISEÑO: proceso de convertir los requisitos de un sistema en una manera de resolver el problema y lograr así implementación que cumpla con los objetivos que se desean alcanzar.

IMPLANTACIÓN: es el proceso de verificar e instalar, en nuestro caso, un sistema con la respectiva capacitación hacia las personas que vayan a interactuar con dicho sistema.

IMPLEMENTACIÓN: en programación, la implementación es la programación de un determinado algoritmo en un lenguaje específico, en función de lograr realizar las funciones que se desean alcanzar.

INFORMACIÓN: A nivel general podemos decir que es un conjunto organizado de todo tipo de datos los cuales han sido procesados. Es el elemento principal que se maneja, en nuestro caso, en la base de datos de todos los comandos utilizados por los agentes de soporte.

CLI: Command Line Interface (Interface de Línea de Comando), como su nombre lo indica, es una interfaz gráfica muy simple donde solo podemos visualizar los comandos que estemos ejecutando, al igual que los resultados que estos nos despliegan pero a manera de solo texto.

INTERFACE: Es un programa creado para permitir la comunicación entre dos o más aplicaciones diferentes, o entre el usuario y las aplicaciones. Las podemos tener de maneras gráficas complejas ó más simples.

LLAVE PRIMARIA: Es la que identifica en forma única.

MENSAJE: señal enviada desde un objeto emisor a otro objeto receptor para que éste lleve a cabo uno de sus métodos.

PROCESO: Conjunto ordenado de métodos, procedimientos, tareas y actividades, relacionados entre sí y que contribuyen a determinar las diferentes funciones.

PROGRAMACIÓN ORIENTADA A OBJETOS: programación basada en objetos y no en acciones, en la que los datos se modelan en clases y antes están los datos que la lógica.

REQUISITO: Características con las que debe cumplir un bien o servicio, implícitas u obligatorias.

UML: lenguaje unificado de modelado. Lenguaje gráfico y notacional usado en la especificación, la visualización, la construcción y la documentación de aplicaciones basadas en la orientación a objetos.