SISTEMA DE IDENTIFICACIÓN AUTOMÁTICA DE CLIENTES CON REGISTRO DE DESPACHO DE TAXIS Y ADMINISTRACIÓN DE INFORMACIÓN HISTÓRICA.

DIXON FERNANDO CANO LARGO
LINA MARÍA MUÑOZ TORRES

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA, FÍSICA Y DE SISTEMAS Y COMPUTACIÓN
PROGRAMA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PEREIRA
2010

SISTEMA DE IDENTIFICACIÓN AUTOMÁTICA DE CLIENTES CON REGISTRO DE DESPACHO DE TAXIS Y ADMINISTRACIÓN DE INFORMACIÓN HISTÓRICA.

DIXON FERNANDO CANO LARGO
LINA MARÍA MUÑOZ TORRES

Proyecto de Grado presentado como requisito para optar al título de INGENIERO EN SISTEMAS Y COMPUTACIÓN

Director del Proyecto
Guillermo Solarte
Ingeniero de Sistemas

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA, FÍSICA Y DE SISTEMAS Y COMPUTACIÓN
PROGRAMA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PEREIRA
2010

Nota de Aceptación:

_____
_____
_____
_____
_____

_____
Firma del presidente del jurado

_____
Firma del jurado

_____
Firma del jurado

Pereira, 2 de Septiembre de 2010

DEDICATORIA

Dedicado a nuestras familias que supieron tener paciencia y colaborarnos en cuanto estuvo a su alcance.

AGRADECIMIENTOS

Gracias al Ingeniero Guillermo Solarte y al Ingeniero Jesús Ernesto Duque Ocampo Gerente de la empresa PrimerTax S.A., quienes con toda su capacidad y conocimiento pudieron asesorarnos de la mejor manera posible para culminar esta presentación con mayor claridad y comprensión posible.

Al ingeniero Juan Carlos Tamayo por su apoyo a esta idea.

CONTENIDO

DEDICATORIA 4
AGRADECIMIENTOS....5
CONTENIDO 6
LISTA DE TABLAS 11
LISTA DE ILUSTRACIONES 12
GLOSARIO 15
OBJETIVOS....19
OBJETIVO GENERAL 19
OBJETIVOS ESPECÍFICOS 19
RESUMEN....20
INTRODUCCIÓN....21
1. MARCO TEÓRICO....22
1.1 METODOLOGÍAS ÁGILES....22
1.1.1 El Manifiesto Ágil.....23
1.1.2 Comparación.....24
1.2 PROGRAMACIÓN EXTREMA (EXTREME PROGRAMMING, XP) 25
1.2.1 Valores.....25
1.2.1.1 Simplicidad. 25
1.2.1.2 Comunicación.....26
1.2.1.3 Retroalimentación (feedback).....26
1.2.1.4 Coraje o valentía.....27
1.2.1.5 Respeto. 27
1.2.2 Proceso XP. 27
1.2.3 Prácticas XP.....28
1.2.4 Alcance de XP.....30
1.2.5 Planeación. 31
1.2.5.1 Historias de Usuario. 31
1.2.5.2 Velocidad del proyecto. 32
1.2.5.3 Iteraciones. 32
1.2.5.4 Entregas Pequeñas. 33
1.2.5.5 Reuniones. 33
1.2.5.6 Roles XP.....34
1.2.5.7 Traslado de personal. 35
1.2.5.8 Ajustar XP.....36
1.2.6 Diseño. 36
1.2.6.1 Simplicidad en el diseño. 37
1.2.6.2 Metáfora del sistema. 37
1.2.6.3 Tarjetas de clase, responsabilidad, colaboración (CRC cards). ..37
1.2.6.4 Soluciones puntuales (Spike Solution). 38
1.2.6.5 No solucionar antes de tiempo. 38
1.2.6.6 Refactorización (Refactoring). 38
1.2.7 Codificación.....39

1.2.7.1 Cliente siempre presente. 39
1.2.7.2 Codificar primero la prueba. 39
1.2.7.3 Programación en parejas. 40
1.2.7.4 Integración secuencial. 40
1.2.7.5 Integraciones frecuentes. 41
1.2.7.6 Estándares y propiedad colectiva del código. 41
1.2.8 Pruebas. 42
1.2.8.1 Pruebas unitarias 42
1.2.8.2 Pruebas de aceptación. 42
1.2.8.3 Cuando se encuentra un error. 43
1.2.9 Proceso de desarrollo en XP. 43
2. OBTENIENDO E INTERPRETANDO LA INFORMACIÓN DEL PROBLEMA. 45
2.1 PROCESO MANUAL DE PRESTACIÓN DE SERVICIO DE TRANSPORTE INDIVIDUAL URBANO 45
2.1.1 Registro de información del servicio 45
2.1.2 Sistema identificador de llamadas. 45
2.1.3 Móvil o vehículo 46
2.1.4 Conductor. 46
2.1.5 Cliente. 46
2.1.6 Operador. 46
2.2 INTERPRETACIÓN DEL PROBLEMA 46
3. PRESENTACIÓN 50
3.1 HERRAMIENTAS EMPLEADAS 50
3.1.1 Visual Studio 2005 Express Editions (Microsoft). 50
3.1.2 MySQL. 50
3.1.3 StarUML. 51
3.2 DESCRIPCIÓN DEL NEGOCIO 51
3.3 DESCRIPCIÓN DEL CLIENTE 52
4. PLANEACIÓN 53
4.1 HISTORIAS DE USUARIO 53
4.1.1 Lo que dice la metodología XP. 53
4.1.1.1 Características 54
4.1.2 Experiencia en el Proyecto. 54
4.2 VELOCIDAD DEL PROYECTO 56
4.2.1 Lo que dice la metodología XP. 56
4.2.2 Experiencia en el Proyecto. 57
4.3 ENTREGAS EN ITERACIONES 59
4.3.1 Lo que dice la metodología XP. 59
4.3.2 Experiencia en el Proyecto. 59
4.4 PLAN DE ENTREGAS 60
4.4.1 Lo que dice la metodología XP. 60
4.4.2 Experiencia en el Proyecto. 60
4.5 REUNIÓN INICIAL DE ITERACIÓN 61
4.5.1 Lo que dice la metodología XP 61
4.5.2 Experiencia en el Proyecto. 61

4.6 REUNIÓN MATINAL....62
4.6.1 Lo que dice la metodología XP. 62
4.6.2 Experiencia en el Proyecto....62
4.7 MOVER PERSONAL 64
4.7.1 Lo que dice la metodología XP. 64
4.7.2 Experiencia en el Proyecto....64
4.8 MODIFICAR XP CUANDO SEA NECESARIO 65
4.8.1 Lo que dice la metodología XP 65
4.8.2 Experiencia en el Proyecto....65
5. DISEÑO....66
5.1 SIMPLICIDAD....66
5.1.1 Lo que dice la metodología XP 66
5.1.2 Experiencia en el Proyecto....66
5.2 METÁFORA DEL SISTEMA 67
5.2.1 Lo que dice la metodología XP 67
5.2.2 Experiencia en el Proyecto....67
5.3 TARJETAS CRC....68
5.3.1 Lo que dice la metodología XP 68
5.3.2 Experiencia en el Proyecto....68
5.4 SPIKE SOLUTION (SOLUCIÓN RÁPIDA)....69
5.4.1 Lo que dice la metodología XP 69
5.4.2 Experiencia en el Proyecto....69
5.5 NO ADICIONE FUNCIONALIDAD ANTES DE TIEMPO 70
5.5.1 Lo que dice la metodología XP 70
5.5.2 Experiencia en el Proyecto....71
5.6 REFACTORIZACIÓN....72
5.6.1 Lo que dice la metodología XP 72
5.6.2 Experiencia en el Proyecto....72
6. CODIFICACIÓN....74
6.1 CLIENTE SIEMPRE PRESENTE 74
6.1.1 Lo que dice la metodología XP 74
6.1.2 Experiencia en el Proyecto....74
6.2 EL CÓDIGO SE ESCRIBE SIGUIENDO ESTÁNDARES 75
6.2.1 Lo que dice la metodología XP 75
6.2.2 Experiencia en el Proyecto....75
6.3 CODIFICAR PRIMERO LA PRUEBA 76
6.3.1 Lo que dice la metodología XP 76
6.3.2 Experiencia en el Proyecto....76
6.4 TODA LA PRODUCCIÓN DE CÓDIGO DEBE SER HECHA EN PAREJAS 77
6.4.1 Lo que dice la metodología XP 77
6.4.2 Experiencia en el Proyecto....78
6.5 SOLO UNA PAREJA HACE INTEGRACIÓN A LA VEZ (INTEGRACIÓN SECUENCIAL) 78
6.5.1 Lo que dice la metodología XP 78

6.5.2 Experiencia en el Proyecto. 79
6.6 INTEGRACIONES FRECUENTES 79
6.6.1 Lo que dice la metodología XP 79
6.6.2 Experiencia en el Proyecto. 80
6.7 PROPIEDAD COLECTIVA DEL CÓDIGO 80
6.7.1 Lo que dice la metodología XP 80
6.7.2 Experiencia en el Proyecto. 81
6.8 NO TRABAJAR HORAS EXTRAS 82
6.8.1 Lo que dice la metodología XP 82
6.8.2 Experiencia en el Proyecto. 82
7. PRUEBAS 84
7.1 EL USO DE LOS TESTER EN XP 84
7.2 PRUEBAS DE UNIDAD 84
7.2.1 Lo que dice la metodología XP 84
7.2.2 Experiencia en el Proyecto. 84
7.3 PRUEBAS DE ACEPTACIÓN 85
7.3.1 Lo que dice la metodología XP. 85
7.3.2 Experiencia en el Proyecto. 85
7.4 MANEJO DE ERRORES 85
7.4.1 Lo que dice la metodología XP. 86
7.4.2 Experiencia en el Proyecto. 86
8. COMENTARIOS 87
8.1 PLANEACIÓN 87
8.1.1 Historias de Usuario. 87
8.1.2 Número de horas de trabajo por semana no laborar horas extras. 87
8.1.3 Velocidad del proyecto. 88
8.2 DISEÑO 88
8.2.1 Diseños simples para facilitar la capacitación. 88
8.2.2 Agregar funcionalidad antes de tiempo. 89
8.3 CODIFICACIÓN 89
8.3.1 Costos del Cliente In Situ. 89
8.3.2 Programación en parejas. 90
8.3.3 Trabajar horas extras. 90
8.4 PRUEBAS 91
8.4.1 Pruebas Autónomas. 91
8.4.2 Pruebas antes de la construcción. 91
9. CONCLUSIONES 92
7.1 MOTIVACIÓN E INTERÉS DEL CLIENTE 92
9.2 CÓDIGO DE PROPIEDAD COLECTIVA 92
9.3 CONTROL DE CAMBIO Y MANEJO DE RIESGOS 92
9.5 SPIKE SOLUTION COMO PRÁCTICA UNIVERSAL 93
10. RECOMENDACIONES 94
11. BIBLIOGRAFÍA 95
12. ANEXOS 98
ANEXO A: ESTÁNDARES 98

ANEXO B: HISTORIAS DE USUARIO 100
ANEXO C: MODELO ENTIDAD-RELACIÓN FINAL 131
ANEXO D: DIAGRAMAS DE CASOS DE USO Y ACTIVIDADES 132
ANEXO E: TARJETAS CRC 140
ANEXO F: MANUAL DE USUARIO 154
ANEXO G: ARQUITECTURA DEL SISTEMA 190

LISTA DE TABLAS

Tabla 1. Diferencias entre metodologías ágiles y no ágiles 24
Tabla 2. Casos de Uso Vs Historias de Usuario 53
Tabla 3. Iteraciones y velocidad del proyecto 57
Tabla 4. Estándares 98

LISTA DE ILUSTRACIONES

Ilustración 1. Las prácticas se refuerzan entre sí.....29
Ilustración 2. Rotación de Personal 36
Ilustración 3. Historia de Usuario No 1.....48
Ilustración 4. Proceso de reuniones.....63
Ilustración 5. Historia de Usuario No 1.....100
Ilustración 6. Historia de Usuario No 2.....101
Ilustración 7. Historia de Usuario No 3.....102
Ilustración 8. Historia de Usuario No 4.....103
Ilustración 9. Historia de Usuario No 5.....104
Ilustración 10. Historia de Usuario No 6.....105
Ilustración 11. Historia de Usuario No 7.....106
Ilustración 12. Historia de Usuario No 8.....107
Ilustración 13. Historia de Usuario No 9.....108
Ilustración 14. Historia de Usuario No 10.....109
Ilustración 15. Historia de Usuario No 11.....110
Ilustración 16. Historia de Usuario No 12.....111
Ilustración 17. Historia de Usuario No 13.....112
Ilustración 18. Historia de Usuario No 14.....113
Ilustración 19. Historia de Usuario No 15.....114
Ilustración 20. Historia de Usuario No 16.....115
Ilustración 21. Historia de Usuario No 17.....116
Ilustración 22. Historia de Usuario No 18.....117
Ilustración 23. Historia de Usuario No 19.....118
Ilustración 24. Historia de Usuario No 20.....119
Ilustración 25. Historia de Usuario No 21.....120
Ilustración 26. Historia de Usuario No 22.....121
Ilustración 27. Historia de Usuario No 23.....122
Ilustración 28. Historia de Usuario No 24.....123
Ilustración 29. Historia de Usuario No 25.....124
Ilustración 30. Historia de Usuario No 26.....125
Ilustración 31. Historia de Usuario No 27.....126
Ilustración 32. Historia de Usuario No 28.....127
Ilustración 33. Historia de Usuario No 29.....128
Ilustración 34. Historia de Usuario No 30.....129
Ilustración 35. Historia de Usuario No 31.....130
Ilustración 36. Modelo Entidad-Relación Final.....131
Ilustración 37. Diagrama de Caso de Uso de Despacho de Taxis.....132
Ilustración 38. Diagrama de Actividades - Prestación de Servicio 132
Ilustración 39. Diagrama de Actividades de la Verificación de Dirección del Cliente 133
Ilustración 40. Diagrama de Actividades de la Consulta de Historial 133
Ilustración 41. Diagrama de Actividades de la Consulta de Información de un Móvil

.....134
Ilustración 42. Diagrama de Caso de Uso de Administración del Sistema 135
Ilustración 43. Diagrama de Actividades de la Gestión de Datos.....135
Ilustración 44. Diagrama de Actividades de la Creación de un registro.....136
Ilustración 45. Diagrama de Actividades de la Modificación de un registro. 137
Ilustración 46. Diagrama de Actividades de la Eliminación de un registro 138
Ilustración 47. Diagrama de Actividades de la Generación de Reportes/SuperBackup.....139
Ilustración 48. CRC Clase Cliente.....140
Ilustración 49. CRC Clase Log.....140
Ilustración 50. CRC Clase Servidor 140
Ilustración 51. CRC Formulario Configuración.....141
Ilustración 52. CRC Formulario Propiedades.....141
Ilustración 53. CRC Modulo Variables 141
Ilustración 54. CRC Formulario Principal 142
Ilustración 55. CRC Formulario Cambio Clave 143
Ilustración 56. CRC Formulario Editar Registro 143
Ilustración 57. CRC Formulario Login 143
Ilustración 58. CRC Formulario Súper Backup 143
Ilustración 59. CRC Formulario Reportes 144
Ilustración 60. CRC Modulo Variables 144
Ilustración 61. CRC Formulario Principal 145
Ilustración 62. CRC Clase Cliente.....147
Ilustración 63. CRC Clase Log.....147
Ilustración 64. CRC Clase Servidor 147
Ilustración 65. CRC Formulario Acerca De 147
Ilustración 66. CRC Formulario Configuración Puerto 148
Ilustración 67. CRC Formulario Dialogo Asignar.....148
Ilustración 68. CRC Formulario Dialogo Eliminación Manual.....149
Ilustración 69. CRC Formulario Emergencias.....149
Ilustración 70. CRC Formulario Login 150
Ilustración 71. CRC Formulario Número Líneas 150
Ilustración 72. CRC Formulario Reporte 150
Ilustración 73. CRC Formulario Propiedades.....151
Ilustración 74. CRC Formulario Principal 151
Ilustración 75. Interfaz del Servidor.....154
Ilustración 76. Ítem para configurar el puerto serial 155
Ilustración 77. Ítem para ocultar la ventana servidor.....155
Ilustración 78. Ítem para mostrar la ventana servidor. 156
Ilustración 79. Ítem para cerrar el servidor.....156
Ilustración 80. Validación para el Administrador 157
Ilustración 81. Configurar el lugar de la base de datos.....158
Ilustración 82. Pestaña Usuarios, Menú de Opciones 159
Ilustración 83. Pestaña Usuarios, Crear Nuevo Usuario.....160
Ilustración 84. Pestaña Usuarios, Privilegios 161

Ilustración 85. Pestaña Usuarios, Modificar Usuario.....162
Ilustración 86. Pestaña Usuarios, Eliminar Usuario 162
Ilustración 87. Pestaña Móviles, Menú de Opciones 163
Ilustración 88. Pestaña Móviles, Registrar Nuevo Usuario 164
Ilustración 89. Pestaña Usuarios, Modificar Usuario.....165
Ilustración 90. Pestaña Usuarios, Eliminar Usuario 165
Ilustración 91. Pestaña Registros, Menú de Opciones 166
Ilustración 92. Pestaña Registros, Agregar Registros.....167
Ilustración 93. Pestaña Registros, Modificar Registro.....168
Ilustración 94. Pestaña Registros, Eliminar Registro 168
Ilustración 95. Pestaña Sanciones, Menú de Opciones.....169
Ilustración 96. Pestaña Sanciones, Registrar Sanción 170
Ilustración 97. Pestaña Sanciones, Modificar Registro.....171
Ilustración 98. Pestaña Sanciones, Eliminar Sanción.....171
Ilustración 99. Pestaña Reportes, Proceso de creación de un reporte. 173
Ilustración 100. Pestaña Reportes, SuperBackup.....174
Ilustración 101. Validación para el Identificador.....175
Ilustración 102. Configurar el lugar de la base de datos.....176
Ilustración 103. Distribución de la interfaz del Identificador 177
Ilustración 104. Registrar Servicio 178
Ilustración 105. Log de Emergencias.....179
Ilustración 106. Configuración del puerto serial 180
Ilustración 107. Número de Líneas 181
Ilustración 108. Ocultar Información de Sanciones.....182
Ilustración 109. Mostrar Información de Sanciones 182
Ilustración 110. Acerca de los autores.....183
Ilustración 111. Proceso para la asignación de un servicio 184
Ilustración 112. Historial de servicios prestados 184
Ilustración 113. Modificar dirección.....185
Ilustración 114. Menú clic derecho.....186
Ilustración 115. Proceso para realizar consultas.....188
Ilustración 116. Consultar Información de Móvil. 189
Ilustración 117. Diagrama de Paquetes - Servidor.....191
Ilustración 118. Diagrama de Paquetes - Identificador. 192
Ilustración 119. Diagrama de Paquetes - Administrador.....194
Ilustración 120. Diagrama de Componentes.....195
Ilustración 121. Diagrama de Despliegue. 196

GLOSARIO

A continuación se darán algunas definiciones necesarias para comprender el contenido del documento.

ANÁLISIS: Es una de las etapas del ciclo de vida de un sistema informático, en esta se recopilan, identifican, clasifican, y documentan los requerimientos del sistema, se estudian los diversos escenarios o tipos de interacción de los usuarios con el sistema, en el análisis estructurado, el resultado de este proceso es el modelo del sistema.

ANÁLISIS DE RIESGOS: Proceso establecido por el equipo de desarrollo mediante el cual se realiza el estudio de viabilidad de una Historia de Usuario expresada por el cliente. A través de este proceso, el equipo de desarrollo verifica el impacto en tiempo, costo y alcance del proyecto; al finalizar el análisis, el equipo tiene como resultado una respuesta que se le presenta al cliente mediante una exposición, en donde se especifican los resultados en términos de costos y de funcionalidad, igualmente se otorgan soluciones operativas, propuestas o recomendaciones según sea adecuado para el caso concreto.

APLICACIÓN: Es un tipo de programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de trabajo. Esto lo diferencia principalmente de otros tipos de programas como los sistemas operativos (que hacen funcionar al ordenador), las utilidades (que realizan tareas de mantenimiento o de uso general), y los lenguajes de programación (con el cual se crean los programas informáticos). Son aplicaciones los procesadores de textos, hojas de cálculo, bases de datos, programas de dibujo, paquetes estadísticos, etc.

BASE DE DATOS: Es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Un conjunto de información almacenada en memoria auxiliar que permite acceso directo y un conjunto de programas que manipulan esos datos.

CLASE: Es una construcción que se utiliza como un modelo (o plantilla) para crear objetos de esa clase. Este modelo describe el estado y el comportamiento que todos los objetos de la clase comparten. Un objeto de una determinada clase se denomina una instancia de la clase. La clase que contiene (y se utilizó para crear) esa instancia se puede considerar como del tipo de ese objeto, por ejemplo, una instancia del objeto de la clase "Personas" sería del tipo "Personas".
CLIENTE: Persona que realiza una llamada telefónica a la central de la empresa de despachos de taxis para solicitar un móvil que lo lleve a un destino.

DESPACHO: Envío de uno o varios móviles hacia un cliente, por petición de él,

para la prestación del servicio.

DESTINO: Punto geográfico de la ciudad que corresponde a una dirección que se refiere a una calle, carrera, barrio, conjunto residencial entre otros.

DISEÑO: Se define como el proceso previo de configuración mental, "pre-figuración", en la búsqueda de una solución. Es el proceso de convertir los requisitos de un sistema en una manera de resolver el problema con el objetivo de posibilitar una implementación que cumpla el costo, prestaciones y calidad deseados.

EVENTO: Hecho que ocurre en momento definido. Suceso que ocurre instantáneamente en un punto del tiempo.

GUI (*GRAPHICS USER INTERFACE*): La interfaz gráfica de usuario es un programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso, consiste en proporcionar un entorno visual sencillo para permitir la comunicación con el sistema operativo de una máquina o computador.

LICENCIA DE USO: Es un contrato entre el licenciante (autor/titular de los derechos de explotación/distribuidor) y el licenciatario del programa informático (usuario consumidor /usuario profesional o empresa), para utilizar el software cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas.

MÉTODO: En la programación orientada a objetos, un método es una subrutina asociada exclusivamente a una clase (llamados métodos de clase o métodos estáticos) o a un objeto (llamados métodos de instancia).

MODELO: En diseño orientado a objetos, una representación del mundo real en unas abstracciones de software denominadas clases y la relación entre ellas.

MODELO CLIENTE-SERVIDOR: Esta arquitectura consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta. Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.
En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa.

MOTOR DE BASE DE DATOS: denominados también *Sistemas Gestores de Bases de Datos (SGBD),* son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada.

MÓVIL: Vehículo que se usa como medio de transporte de personas o cosas.

OBJETO: En el paradigma de programación orientada a objetos (POO), un objeto se define como la unidad que en tiempo de ejecución realiza las tareas de un programa. También a un nivel más básico se define como la instancia de una clase.

OPERADOR: Es la persona encargada de atender a los clientes y de interactuar con el sistema.

PARADIGMA ORIENTADO A OBJETOS: La programación orientada a objetos o POO (OOP según sus siglas en inglés) es un paradigma de programación que usa objetos y sus interacciones, para diseñar aplicaciones y programas informáticos. Está basado en varias técnicas, incluyendo herencia, abstracción, polimorfismo y encapsulamiento. Su uso se popularizó a principios de la década de los años 1990. En la actualidad, existe variedad de lenguajes de programación que soportan la orientación a objetos.

PRESTACIÓN DE SERVICIO: Acción que ejecuta un operador cuando asigna al conductor de un móvil la posición geográfica de un cliente, para que este le ofrezca el servicio de transporte hasta un destino.

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.

PROTOTIPO: Primer ejemplar de alguna cosa que se toma como modelo para crear otros de la misma clase con funcionalidades más complejas y que se integra con otras partes del sistema. La creación de prototipos es fundamental para la ejecución de pruebas funcionales.

SISTEMA EXTERNO: Ente que interactúa con el sistema. Entra, manipula o recibe información del sistema, pero es externo al sistema. Puede ser un hardware u otro programa.

SO (SISTEMA OPERATIVO): Es un software que actúa de interfaz entre los dispositivos de hardware y los programas de usuario o el usuario mismo para utilizar un computador. Es responsable de gestionar, coordinar las actividades y llevar a cabo el intercambio de los recursos y actúa como intermediario para las aplicaciones que se ejecutan.

SOLICITUD DE SERVICIO: Acción que ejecuta un cliente cuando le solicita al operador la prestación de un servicio.

SUBSISTEMA: Sistema que es parte de otro sistema. Un sistema puede estar constituido por múltiples partes y subsistemas. En general, desde el punto de vista de un sistema determinado, un subsistema es fundamental para el funcionamiento del sistema que lo contiene.

VENTANA: Es un área visual, normalmente de forma rectangular, que contiene algún tipo de interfaz de usuario, mostrando la salida y permitiendo la entrada de datos para uno de varios procesos que se ejecutan simultáneamente. Las ventanas se asocian a interfaces gráficas, donde pueden ser manipuladas con un puntero. Su uso permite superponerlas para ver aquello que deseamos, manteniendo a la vez otras ventanas en superposición o en otros sectores de la pantalla. El sistema de ventanas ha proporcionado al usuario una mayor facilidad y rapidez de uso al no tener que recurrir a comandos difíciles de recordar, puesto que se transmiten las órdenes a través de iconos.

OBJETIVOS

OBJETIVO GENERAL

Realizar el análisis, diseño e implementación de un software que sirva como soporte para el registro y la administración de la información correspondiente al despacho de taxis de la empresa PrimerTax S.A.

OBJETIVOS ESPECÍFICOS

1. Realizar un trabajo de campo donde se pueda entender cómo se llevan a cabo los procesos de registro de los despachos realizados en la empresa y plasmar ese conocimiento adquirido en el trabajo de campo, de tal forma que facilite la implementación de un software de soporte para agilizar el registro de dicha información y permitir la administración ordenada de la misma.
2. Registrar los requerimientos del cliente en plantillas de historias de Usuario, se dejarán especificados los detalles de las necesidades en cuanto a mejoras y a la operatividad en el proceso de despacho.
3. Hacer el diseño de cada historia de usuario y realizar una plantilla de aceptación en donde se especifican los resultados esperados para considerar que el diseño y desarrollo de la historia de usuario será exitoso cuando se acoplen las partes para formar el sistema que el usuario espera como producto final.
4. Realizar el acoplamiento de las partes realizadas como subsistemas de manera que se tenga un producto final estable y que ofrezca las soluciones esperadas por la empresa de despachos.

RESUMEN

En el documento se expondrá la problemática concerniente a la habilidad operativa que demanda el registro de una prestación de servicio, en la central de despachos de PrimerTax S.A. Seguido de esto se hará una descripción de la forma en la que la empresa hacía frente a este problema, se mostrarán los antecedentes y la metodología aplicada en el registro del despachos.

En este documento también se mostrará el análisis, diseño y desarrollo de una solución sistematizada que además de ser de utilidad en la lucha contra el problema planteado, ayuda a los procesos administrativos de la empresa, seguido por una muestra de los resultados del software desarrollado. Por último, se darán algunas conclusiones del trabajo realizado y una mención de los aportes que este trabajo entrega al mecanismo de mejoramiento continuo de los procesos operativos de una empresa de despachos.

Este trabajo se realizó con un objetivo fijo, el cual era producir una herramienta de apoyo que permitiera agilizar y mejorar la atención de un cliente, lo cual es una base sólida para el desarrollo de una compañía con éxito y con clientes satisfechos.

INTRODUCCIÓN

Las tecnologías de la informática han desempeñado un importante papel a lo largo de la historia como facilitadores de procesos. Consiguiente a esto, los procesos de mayor impacto dentro de una empresa de servicios son: la toma de decisiones y ofrecer un buen servicio a los clientes.

Dentro de la toma adecuada de decisiones en una empresa, es fundamental la calidad y consistencia de la información y dentro de la calidad del servicio se encuentra un importante factor como lo es la agilidad en el servicio prestado.

En una sociedad tan competitiva, las empresas buscan mejorar sus procesos y dichos procesos están ligados al modelo de negocio, es por esto que una implementación de un software a la medida, que satisfaga unos requerimientos precisos de gestión de información y de mejoramiento en los tiempos de servicio, puede ser de gran ayuda para la obtención oportuna de información necesaria para el proceso administrativo y el mejoramiento de los servicios ofrecidos por la empresa.

Cada día los sistemas de información ofrecen mayores capacidades a bajo costo, el uso de los mismos es fundamental en el desarrollo de actividades administrativas y operativas, ya que agilizan el trabajo y permiten un mayor control en el uso y almacenamiento de la información referente al negocio.

Algunas empresas de despacho de taxis no cuentan con sistemas de información, que apoyen el proceso de prestación de sus servicios y que al mismo tiempo guarde de manera adecuada la información referente a los servicios prestados. Sin embargo, sí tienen la infraestructura necesaria para poder obtener la información desde consolas de identificación de llamadas, que pueden ser conectadas a equipos de cómputo a través del puerto serial para luego procesar esta información. Igualmente, se ha determinado la posibilidad de que estas empresas puedan adquirir equipos con capacidades medias; teniendo en cuenta que el sistema no requiere de características demasiado exigentes.

Por las razones expuestas, nos hemos dado a la tarea de investigar al respecto, con el fin de realizar una herramienta de software, que cumpla con los requerimientos; brindando una solución a los problemas de información que tienen estas empresas. Otra característica esencial, será desarrollar dicha herramienta, con una alta facilidad de uso para que el rendimiento en el préstamo de servicio de despacho de móviles, no afecte de manera negativa el negocio durante el periodo de adaptación al sistema por parte de los operadores.

1. MARCO TEÓRICO

En este capítulo se darán los conceptos relacionados con el despacho de móviles, necesarios para que el lector comprenda en que consiste el proceso relacionado con el préstamo de servicio de transporte individual urbano; adicionalmente, se realiza una presentación teórica de XP (*Extreme Programming*) como metodología de desarrollo con el cual fue realizado el proyecto.

Se inicia con una descripción del proceso de prestación de servicio de transporte individual urbano, enumerando los pasos seguidos durante el proceso, los elementos que componen el proceso y la información registrada durante el mismo.

Finalmente, se realiza una descripción del manifiesto ágil, seguido de una conceptualización de XP como metodología de desarrollo, detallando sus etapas de desarrollo (planeación, diseño, codificación y pruebas) junto a los aspectos que componen cada etapa.

1.1 METODOLOGÍAS ÁGILES

En febrero de 2001, tras una reunión celebrada en Utah-EEUU, nace el término ágil aplicado al desarrollo de software. En esta reunión participan un grupo de 17 expertos de la industria del software, incluyendo algunos de los creadores o impulsores de metodologías de software. Su objetivo fue esbozar los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y respondiendo a los cambios que puedan surgir a lo largo del proyecto.

Se pretende ofrecer una alternativa a los procesos de desarrollo de software tradicionales, caracterizados por ser rígidos y dirigidos por la documentación que se genera en cada una de las actividades desarrolladas.

Tras esta reunión, se creó The Agile Alliance, una organización sin ánimo de lucro, dedicada a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones para que adopten dichos conceptos. El punto de partida es fue el Manifiesto Ágil, un documento que resume la filosofía "ágil".

1.1.1 El Manifiesto Ágil.

Según el Manifiesto se valora:

- **Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas.** La gente es el principal factor de éxito de un proyecto software. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente. Es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades.
- **Desarrollar software que funciona más que conseguir una buena documentación.** La regla a seguir es no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante. Estos documentos deben ser cortos y centrarse en lo fundamental.
- **La colaboración con el cliente más que la negociación de un contrato.** Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito.
- **Responder a los cambios más que seguir estrictamente un plan.** La habilidad de responder a los cambios que puedan surgir a los largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, etc.) determina también el éxito o fracaso del mismo. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

Los valores anteriores inspiran los doce principios del manifiesto. Son características que diferencian un proceso ágil de uno tradicional. Los dos primeros principios son generales y resumen gran parte del espíritu ágil. El resto tienen que ver con el proceso a seguir y con el equipo de desarrollo, en cuanto metas a seguir y organización del mismo. Los principios son:

I. La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
II. Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
III. Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
IV. La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
V. Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.
VI. El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.

VII. El software que funciona es la medida principal de progreso.
VIII. Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
IX. La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
X. La simplicidad es esencial.
XI. Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.

En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

1.1.2 Comparación.

La Tabla 1 recoge esquemáticamente las principales diferencias de las metodologías ágiles con respecto a las tradicionales (no ágiles). Estas diferencias que afectan no sólo al proceso en sí, sino también al contexto del equipo así como a su organización.

Tabla 1. Diferencias entre metodologías ágiles y no ágiles

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura del	La arquitectura del software es esencial

software	y se expresa mediante modelos

1.2 PROGRAMACIÓN EXTREMA (EXTREME PROGRAMMING, XP)

XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. Kent Beck, el padre de XP, describe la filosofía de XP sin cubrir los detalles técnicos y de implantación de las prácticas. Posteriormente, otras publicaciones de experiencias se han encargado de dicha tarea. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

1.2.1 Valores.

Mas que una metodología, XP se considera una disciplina, la cual está sostenida por valores y principios propios de las metodologías ágiles. Los valores originales de la programación extrema son:

- Simplicidad
- comunicación
- retroalimentación (*feedback*)
- coraje.

Un quinto valor, respeto, fue añadido en la segunda edición de *Extreme Programming Explained*. Los cinco valores se detallan a continuación:

1.2.1.1 Simplicidad.

La simplicidad es la base de la programación extrema. Se simplifica el *diseño* para agilizar el desarrollo y facilitar el mantenimiento. Un diseño complejo del código junto a sucesivas modificaciones por parte de diferentes desarrolladores, hacen

que la complejidad aumente exponencialmente. Para mantener la simplicidad es necesaria la refactorización del código, ésta es la manera de mantener el código simple a medida que crece. También se aplica la simplicidad en la *documentación*, de esta manera el código debe comentarse en su justa medida, intentando eso sí, que el código esté autodocumentado. Para ello se deben elegir adecuadamente los nombres de las variables, métodos y clases. Los nombres largos no decrementan la eficiencia del código ni el tiempo de desarrollo gracias a las herramientas de autocompletado y refactorización que existen actualmente. Aplicando la simplicidad junto con la autoría colectiva del código y la programación por parejas se asegura que cuanto más grande se haga el proyecto, todo el equipo conocerá más y mejor el sistema completo.

1.2.1.2 Comunicación.

La comunicación se realiza de diferentes formas. Para los programadores el *código* comunica mejor cuanto más simple sea. Si el código es complejo hay que esforzarse para hacerlo inteligible. El código autodocumentado es más fiable que los comentarios, ya que éstos últimos pronto quedan desfasados con el código a medida que es modificado. Debe comentarse sólo aquello que no va a variar, por ejemplo el objetivo de una clase o la funcionalidad de un método. Las *pruebas unitarias* son otra forma de comunicación ya que describen el diseño de las clases y los métodos, al mostrar ejemplos concretos de cómo utilizar su funcionalidad. Los programadores se comunican constantemente gracias a la *programación por parejas*. La comunicación con el *cliente* es fluida ya que el cliente forma parte del equipo de desarrollo. El cliente decide que características tienen prioridad y siempre debe estar disponible para solucionar dudas.

1.2.1.3 Retroalimentación (feedback).

Al estar el cliente integrado en el proyecto, su opinión sobre el estado del proyecto se conoce en tiempo real. Al realizarse ciclos muy cortos tras los cuales se muestran resultados, se minimiza el tener que rehacer partes que no cumplen con los requisitos y ayuda a los programadores a centrarse en lo que es más importante. Considérense los problemas que derivan de tener ciclos muy largos. Meses de trabajo pueden tirarse por la borda debido a cambios en los criterios del cliente o malentendidos por parte del equipo de desarrollo. El código también es una fuente de retroalimentación gracias a las herramientas de desarrollo. Por ejemplo, las pruebas unitarias informan sobre el estado de salud del código. Ejecutar las pruebas unitarias frecuentemente permite descubrir fallos debidos a cambios recientes en el código.

1.2.1.4 Coraje o valentía.

Los puntos anteriores parecen tener sentido común, entonces, ¿por qué coraje? Para los gerentes la programación en parejas puede ser difícil de aceptar, porque les parece como si la productividad se fuese a reducir a la mitad ya que solo la mitad de los programadores está escribiendo código. Hay que ser valiente para confiar en que la programación por parejas beneficia la calidad del código sin repercutir negativamente en la productividad. La simplicidad es uno de los principios más difíciles de adoptar. Se requiere coraje para implementar las características que el cliente quiere ahora, sin caer en la tentación de optar por un enfoque más flexible que permita futuras modificaciones. No se debe emprender el desarrollo de grandes marcos de trabajo (*frameworks*) mientras el cliente espera. En ese tiempo el cliente no recibe noticias sobre los avances del proyecto y el equipo de desarrollo no recibe retroalimentación para saber si va en la dirección correcta. La forma de construir marcos de trabajo es mediante la refactorización del código en sucesivas aproximaciones.

1.2.1.5 Respeto.

El respeto se manifiesta de varias formas. Los miembros del equipo se respetan los unos a otros, porque los programadores no pueden realizar cambios que hacen que las pruebas existentes fallen o que demore el trabajo de sus compañeros. Los miembros respetan su trabajo porque siempre están luchando por la alta calidad en el producto y buscando el diseño óptimo o más eficiente para la solución a través de la refactorización del código. Los miembros del equipo respetan el trabajo del resto no haciendo menos a otros, si no orientándolos a realizarlo mejor, obteniendo como resultado una mejor autoestima en el equipo y elevando el ritmo de producción en el equipo.

1.2.2 Proceso XP.

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

- El cliente define el valor de negocio a implementar.
- El programador estima el esfuerzo necesario para su implementación.
- El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
- El programador construye ese valor de negocio.
- Vuelve al paso 1.

En todas las iteraciones de este ciclo, tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio

posible con cada iteración.

El ciclo de vida ideal de XP consiste de seis fases:

- Exploración
- Planificación de la Entrega (Release)
- Iteraciones
- Producción
- Mantenimiento y
- Muerte del Proyecto.

1.2.3 Prácticas XP.

A partir de los valores se plantea una serie de prácticas que sirven de guía para los desarrolladores en esta metodología. Una de los aspectos más importantes para XP son las doce reglas que se plantean, las cuales se caracterizan por su grado de simplicidad y por su enfoque en la practicidad, además de que cada regla se complementa con las demás.

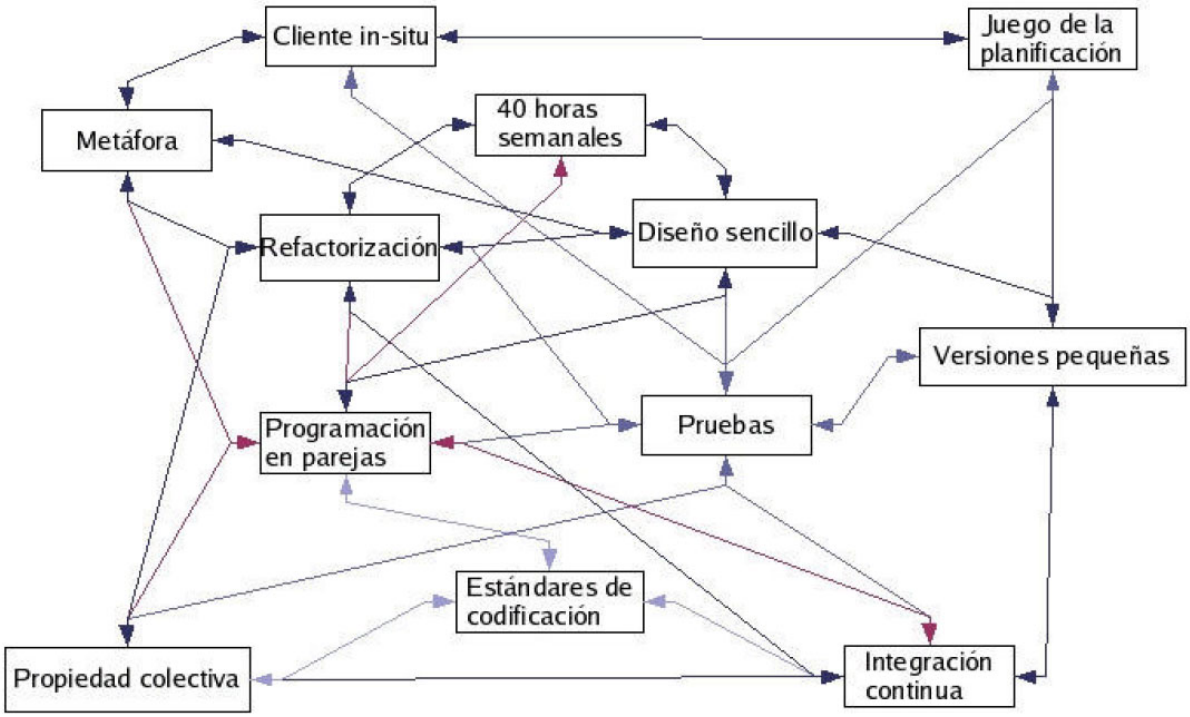
La principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo, convirtiéndola en una curva logarítmica del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de las siguientes prácticas.

- **El juego de la planificación.** Hay una comunicación frecuente entre el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración.
- **Entregas pequeñas**. Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más 3 meses.
- **Metáfora**. El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema (conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema, ayudando a la nomenclatura de clases y métodos del sistema).
- **Diseño simple**. Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.

- **Pruebas**. La producción de código está dirigida por las pruebas unitarias. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.
- **Refactorización (Refactoring).** Es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. Se mejora la estructura interna del código sin alterar su comportamiento externo.
- **Programación en parejas**. Toda la producción de código debe realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores,…).
- **Propiedad colectiva del código.** Cualquier programador puede cambiar cualquier parte del código en cualquier momento.
- **Integración continua.** Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.
- **40 horas por semana**. Se debe trabajar un máximo de 40 horas por semana. No se trabajan horas extras en dos semanas seguidas. Si esto ocurre, probablemente está ocurriendo un problema que debe corregirse. El trabajo extra desmotiva al equipo.
- **Cliente in-situ.** El cliente tiene que estar presente y disponible todo el tiempo para el equipo. Éste es uno de los principales factores de éxito del proyecto XP. El cliente conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada. La comunicación oral es más efectiva que la escrita.
- **Estándares de programación.** XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.

El mayor beneficio de las prácticas se consigue con su aplicación conjunta y equilibrada puesto que se apoyan unas en otras. Esto se puede apreciar en la Ilustración 1, donde una línea entre dos prácticas significa que las dos prácticas se refuerzan entre sí. La mayoría de las prácticas propuestas por XP no son novedosas sino que en alguna forma ya habían sido propuestas en ingeniería del software e incluso demostrado su valor en la práctica. El mérito de XP es integrarlas de una forma efectiva y complementarlas con otras ideas desde la perspectiva del negocio, los valores humanos y el trabajo en equipo.

Ilustración 1. Las prácticas se refuerzan entre sí



1.2.4 Alcance de XP.

La programación extrema es conveniente en ciertas situaciones, pero también es necesario saber que presenta controversia en otras. Esta metodología es aplicable con resultados positivos a proyectos de mediana y pequeña envergadura, donde los grupos de trabajo no superan 20 personas.

Otro aspecto importante en la selección de esta metodología radica en el ambiente cambiante que se presenta en los requerimientos de la aplicación. La metodología XP está encaminada hacia los desarrollos que requieren de cambios continuos en el transcurso de un proyecto. La metodología es recomendada para proyectos en los cuales el costo de cambio no se incremente a medida que transcurre la vida del mismo.

Los proyectos realizados bajo esta metodología cumplen con lo estrictamente necesario en funcionalidad en el momento necesario: hacer lo que se necesita cuando se necesita. En XP no es conveniente precipitarse o adelantarse a las tareas que se han establecido previamente sin el consentimiento del cliente, estos hechos conllevan a inyectar complejidad al sistema, alejándolo del concepto de simplicidad.

1.2.5 Planeación.

La planeación es la etapa inicial de todo proyecto en XP. En este punto se comienza a interactuar con el cliente y el resto del grupo de desarrollo para descubrir los requerimientos del sistema. En este punto se identifican el número y tamaño de las iteraciones al igual que se plantean ajustes necesarios a la metodología según las características del proyecto.

En este apartado se tendrán en cuenta ocho elementos, los cuales son los siguientes:

- Historia de usuario
- Velocidad del proyecto
- Iteraciones
- Entregas pequeñas
- Reuniones
- Roles en XP
- Traslado del personal y
- Ajuste a XP.

1.2.5.1 Historias de Usuario.

Son la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Este planteamiento se desarrolla a lo largo del proyecto: el cliente es quien decide que hacer. Como primer paso, se debe proporcionar una idea clara de lo que será el proyecto en sí.

Se puede considerar que las historias de usuario en XP juegan un papel similar a los casos de uso en otras metodologías (ver Anexo D: Diagramas De Casos de Uso), pero en realidad son muy diferentes. Las historias de usuario sólo muestran la silueta de una tarea a realizarse. Por esta razón es fundamental que el usuario o un representante del mismo se encuentren disponibles en todo momento para solucionar dudas, estas no proporcionan información detallada acerca de una actividad específica.

Las historias de usuario también son utilizadas para estimar el tiempo que el equipo de desarrollo tomará para realizar las entregas. En una entrega se puede desarrollar una o varias historias de usuario, esto depende del tiempo que demore la implementación de cada una de las mismas.

Beck en su libro[1] presenta un ejemplo de ficha (story card) en la cual pueden reconocerse los siguientes contenidos:

- Fecha
- Tipo de actividad (nueva, corrección, mejora)
- Prueba funcional
- Número de historia
- Prioridad técnica y del cliente
- Referencia a otra historia previa
- Riesgo
- Estimación técnica
- Descripción
- Notas y
- Lista de seguimiento con la fecha, estado cosas por terminar y comentarios.

A efectos de planificación, las historias pueden ser de una a tres semanas de tiempo de programación (para no superar el tamaño de una iteración). Las historias de usuario son descompuestas en tareas de programación (task card) y asignadas a los programadores para ser implementadas durante una iteración.

1.2.5.2 Velocidad del proyecto.

Es una medida de la capacidad que tiene el equipo de desarrollo para evacuar las historias de usuario en una determinada iteración. Esta medida se calcula totalizando el número de historias de usuario realizadas en una iteración. Para la iteración siguiente se podrá (teóricamente) implementar el mismo número de historias de usuario que en la iteración anterior.

Cabe recordar que la velocidad del proyecto ayuda a determinar la cantidad de historias que se pueden implementar en las siguientes iteraciones, aunque no de manera exacta. La revisión continua de esta métrica en el transcurso del proyecto se hace necesaria, ya que las historias varían según su grado de dificultad, haciendo inestable la velocidad de la realización del sistema.

1.2.5.3 Iteraciones.

En la metodología XP, la creación del sistema se divide en etapas para facilitar su realización. Por lo general los proyectos constan de más de tres etapas, las cuales toman el nombre de iteraciones, de allí se obtiene el concepto de *metodología iterativa*. La duración ideal es de una a tres semanas.

[1] BECK, Kent. Extreme Programming Explained. En: Chapter 15. Planning Strategy. p. 72.

Para cada iteración se define un módulo o conjunto de historias que se van a implementar. Al final de la iteración se obtiene como resultado la entrega del módulo correspondiente, el cual debe haber superado las pruebas de aceptación que establece el cliente para verificar el cumplimiento de los requisitos. Las tareas que no se realicen en una iteración son tomadas en cuenta para la próxima iteración, donde se define, junto al cliente, si se deben realizar o si se deben ser removidas de la planeación del sistema.

1.2.5.4 Entregas Pequeñas.

La duración de una iteración varía entre una y tres semanas, al final de la cual habrá una entrega de los avances del producto, los cuales deberán ser completamente funcionales. Estas entregas deben caracterizarse por ser frecuentes.

1.2.5.5 Reuniones.

El planeamiento es esencial para cualquier tipo de metodología, es por ello que XP requiere de una revisión continua del plan de trabajo. A pesar de ser una metodología que **evita la documentación exagerada**, es muy estricta en la organización del trabajo.

- **Plan de entregas**

Al comenzar el proyecto se realiza una reunión entre el equipo de trabajo y los clientes. En dicha reunión se define el marco temporal de la realización del sistema. El cliente expone las historias de usuario a los integrantes de grupo, quienes estimarán el grado de dificultad de la implementación de cada historia.

Las historias de usuario son asignadas a las diferentes iteraciones según su orden de relevancia para el proyecto. En el proceso de selección de las historias de usuario para cada iteración, se tiene en cuenta que la suma de las estimaciones sea aproximada a la velocidad del proyecto de la iteración pasada.

En esta reunión se predicen los tiempos que se utilizaran en la realización de las diferentes etapas del proyecto, los cuales no son datos exactos pero proporcionan una base del cronograma.

Finalmente, a partir de las historias de usuario, el cliente plantea las pruebas de aceptación, con las cuales se comprueba que cada una de las historias de usuario ha sido correctamente implementada.

- **Inicial de Iteración**

Al comenzar una iteración se realiza una reunión de la misma, donde se organizan las actividades de programación a realizar. Las historias de usuario son traducidas a tareas y asignadas a los desarrolladores.

Los desarrolladores estiman los tiempos para la realización de las tareas. Cada tarea se estima de uno a tres días de programación ideales o sin distracciones. Estas estimaciones son más exactas que las realizadas en la planeación de entregas, por lo tanto no deben exceder la velocidad de proyecto de la iteración anterior. De ser así, se consulta con el cliente para determinar que historias de usuario se pospondrán para iteraciones futuras.

- **Diarias o "stand-up meeting"**

Estas reuniones se realizaran al comenzar la jornada laboral. Todo el equipo de desarrollo se reúne para exponer los problemas e ideas que se estén presentando, esto con el fin que el equipo en conjunto construya una mejor solución.

Es de vital importancia evitar las discusiones largas, ya que se está utilizando tiempo laboral que puede ser destinado a la construcción del sistema. También debe evitarse las conversaciones separadas, las dudas que se presenten serán solucionadas por el equipo en conjunto.

1.2.5.6 Roles XP.

En esta metodología se utiliza el concepto de roles para organizar quienes se encargan de cada una de las actividades que deben realizarse en el transcurso del proyecto. Cada uno de estos papeles son desempeñados por uno o varios integrantes del grupo, sin descartar la posibilidad de rotar entre el equipo durante la realización del sistema.

Los roles de acuerdo con la propuesta original de Beck son:

- **Programador.** El programador escribe las pruebas unitarias y produce el código del sistema. Cuando surgen dudas o preguntas que afectan decisiones sobre funcionalidad del sistema (las decisiones técnicas son solucionadas gracias a las habilidades de los programadores), el programador no debe hacer suposiciones acerca de lo que el cliente quiere; en este caso, debe dirigirse al mismo y aclarar la situación.

- **Cliente.** Escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en

aportar mayor valor al negocio. Es parte fundamental del equipo XP (se menciona su importancia como una de las prácticas), en todo el proyecto debe existir un cliente.

- **Encargado de pruebas (Tester).** Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.

- **Encargado de seguimiento (Tracker).** Tiene como tarea observar la realización del sistema. Proporciona realimentación al equipo, varias veces por semana cuestiona a los integrantes del equipo para anotar sus logros y avances. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración. Mantiene datos históricos del proyecto.

- **Entrenador (Coach).** Es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente. Se asegura que los conceptos de la metodología se apliquen al proyecto, además de brindar ayuda continua a los demás integrantes del equipo.

- **Consultor.** Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.

- **Gestor (Big boss).** Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación. Tiene como responsabilidad la dirección y organización de las reuniones que se realizan durante el proyecto. Es erróneo afirmar que entre sus tareas se encuentra decir que hacer, cuando hacer y de revisar como se desarrolla el sistema, para ello se cuenta con el apoyo del cliente, el tracker y los demás miembros del grupo.

1.2.5.7 Traslado de personal.

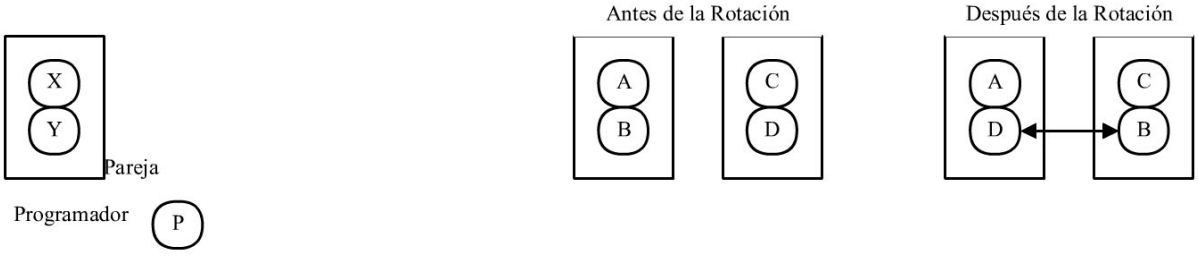
Al mover el personal se evitan problemas relacionados con la pérdida de conocimiento y cuellos de botellas. Todos los miembros del grupo deben tener suficiente conocimiento de la estructura del código de modo tal que se eviten las islas de conocimiento las cuales son susceptibles de generar pérdidas de información importante.

En la medida que todos los programadores entiendan todas las partes del programa se evita que unos tengan una carga de trabajo muy alta mientras que

otros no tengan mucho trabajo por hacer.

La programación en parejas se convierte en una herramienta muy importante para lograr el objetivo del traslado de personal sin que se pierda el rendimiento. Esto se logra haciendo que un miembro de la pareja se traslade mientras el otro continúe el desarrollo con un nuevo compañero.

Ilustración 2. Rotación de Personal



1.2.5.8 Ajustar XP.

Todos los proyectos tienen características específicas por lo cual XP puede ser modificado para ajustarse bien al proyecto en cuestión. Al iniciar el proyecto se debe aplicar XP tal como es, sin embargo no se debe dudar en modificar aquellos aspectos en que no funcione. Eso no quiere decir que los desarrolladores pueden hacer lo que se les antoje. Antes de implementarse un cambio, este debe ser discutido y aprobado por el grupo.

1.2.6 Diseño.

En XP solo se diseñan aquellas historias de usuario que el cliente ha seleccionado para la iteración actual por dos motivos: por un lado se considera que no es posible tener un diseño completo del sistema y sin errores desde el principio. El segundo motivo es que dada la naturaleza cambiante del proyecto, el hacer un diseño muy extenso en las fases iniciales del proyecto para luego modificarlo, se considera un desperdicio de tiempo.

Es importante resaltar que esta tarea es permanente durante la vida del proyecto partiendo de un diseño inicial que va siendo corregido y mejorado en el transcurso del proyecto.

Los aspectos que se tratarán en esta fase se especifican a continuación:

- Simplicidad en el diseño
- Metáfora del sistema
- Tarjetas CRC
- Spike solution
- No solucionar antes de tiempo y
- Refactoring.

1.2.6.1 Simplicidad en el diseño.

Una de las partes más importantes de la filosofía XP es la simplicidad en todos los aspectos. Se considera que un diseño sencillo se logra más rápido y se implementa en menos tiempo, por lo cual esto es lo que se busca. La idea es que se haga el diseño más sencillo que cumpla con los requerimientos de las historias de usuario.

Sobre los diagramas, se es muy claro que se pueden usar siempre que no tome mucho tiempo en realizarlos, que sean de verdadera utilidad y que se esté dispuesto a "tirarlos a la basura". En XP se prefiere tener una descripción del sistema o parte de él, en lugar de una serie de complejos diagramas que probablemente tomen más tiempo y sean menos instructivos.

1.2.6.2 Metáfora del sistema.

Se trata de plasmar la arquitectura de sistema en una "historia" con la cual se le dé al grupo de desarrollo una misma visión sobre el proyecto además brindarles un primer vistazo muy completo a los nuevos integrantes del grupo para hacer su adaptación más rápida.

Es muy importante dentro del desarrollo de la metáfora darle nombres adecuados a todos los elementos del sistema constantemente, y que estos correspondan a un sistema de nombres consistente. Esto será de mucha utilidad en fases posteriores del desarrollo para identificar aspectos importantes del sistema.

1.2.6.3 Tarjetas de clase, responsabilidad, colaboración (CRC cards).

La principal funcionalidad que tienen estas, es ayudar a dejar el pensamiento procedimental para incorporarse al enfoque orientado a objetos. Cada tarjeta representa una clase con su nombre en la parte superior, en la sección inferior izquierda están descritas las responsabilidades y a la derecha las clases que le sirven de soporte.

En el proceso de diseñar el sistema por medio de tarjetas CRC como máximo dos personas se ponen de pie adicionando o modificando las tarjetas, prestando atención a los mensajes que estas se transmiten mientras los demás miembros del grupo que permanecen sentados, participan en la discusión obteniendo así lo que puede considerarse un *diagrama de clases* preliminar.

1.2.6.4 Soluciones puntuales (Spike Solution).

En muchas ocasiones los equipos de desarrollo se enfrentan a requerimientos de los clientes (en este caso historias de usuario) los cuales generan problemas desde el punto de vista del diseño o la implementación. Spike Solution, es una herramienta de XP para abordar este inconveniente.

Se trata de una pequeña aplicación completamente desconectada del proyecto con la cual se intenta explorar el problema y propone una solución potencial. Puede ser burda y simple, siempre que brinde la información suficiente para enfrentar el problema encontrado.

1.2.6.5 No solucionar antes de tiempo.

Los desarrolladores tienden a predecir las necesidades futuras e implementarlas antes. Según mediciones, esta es una práctica ineficiente, concluyendo que tan solo el 10% de las soluciones para el futuro son utilizadas, desperdiciando tiempo de desarrollo y complicando el diseño innecesariamente.

En XP sólo se analiza lo que se desarrollará en la iteración actual, olvidando por completo cualquier necesidad que se pueda presentar en el futuro, lo que supone uno de los preceptos más radicales de la programación extrema.

1.2.6.6 Refactorización (Refactoring).

Como se trató al principio de este apartado, el diseño es una tarea permanente durante toda la vida del proyecto y la refactorización concreta este concepto. Como en cualquier metodología tradicional en XP se inicia el proceso de desarrollo con un diseño inicial. La diferencia es que en las metodologías tradicionales este diseño es tan global y completo que si se ven forzados a modificarlo será un fracaso para el grupo de desarrollo. El caso de XP es el opuesto. Se parte de un diseño muy general y simple que no debe tardar en conseguirse, al cual se le hacen adiciones y correcciones a medida que el proyecto avanza, con el fin de mantenerlo tanto correcto como simple.

La *refactorización en el código* pretende conservarlo tan sencillo y fácil de mantener como sea posible. En cada inspección que se encuentre alguna redundancia, funcionalidad no necesaria o aspecto en general por corregir, se debe rehacer esa sección de código, con el fin de lograr las metas de sencillez tanto en el código en sí mismo como en la lectura y mantenimiento.

Estas prácticas son difíciles de llevar a cabo cuando se está iniciando en XP por varios motivos. En primer lugar debido al temor que genera en los equipos de desarrollo cambiar algo que ya funciona bien, sea a nivel de diseño o implementación. Sin embargo si se cuenta con un esquema de pruebas completo y un sistema de automatización para las mismas se tendrá éxito en el proceso. El otro motivo es la creencia que es más el tiempo que se pierde en refactoring que el ganando sencillez y mantenimiento. Según XP la ganancia obtenida en refactoring es tan relevante que justifica suficientemente el esfuerzo extra en corrección de redundancias y funcionalidades innecesarias.

1.2.7 Codificación.

La codificación es un proceso que se realiza en forma paralela con el diseño, está sujeta a varias observaciones por parte de XP consideradas controversiales por algunos expertos tales como, la rotación de los programadores o la programación en parejas.

Además de los mencionados temas, se tiene a continuación la descripción de los siguientes temas:

- Cliente siempre presente
- Codificar primero la prueba
- Integración secuencial e
- Integraciones frecuentes.

1.2.7.1 Cliente siempre presente.

Uno de los requerimientos de XP es que el cliente esté siempre disponible. No solamente para solucionar las dudas del grupo de desarrollo, debería ser parte de éste. En este sentido se convierte en una gran ayuda al solucionar todas las dudas que puedan surgir, especialmente cara a cara, para garantizar que lo implementado cubre con las necesidades planteadas en las historias de usuario.

1.2.7.2 Codificar primero la prueba.

Cuando se crea primero una prueba, se ahorra mucho tiempo elaborando el

código que la haga pasar, siendo menor el tiempo de hacer ambos procesos que crear el código solamente.

Una de las ventajas de crear una prueba antes que el código es que permite identificar los requerimientos de dicho código. En otras palabras, al escribir primero las pruebas se encuentran de una forma más sencilla y con mayor claridad todos los casos especiales que debe considerar el código a implementar. De esta forma, el desarrollador sabrá con completa certeza en qué momento ha terminado, ya que habrán pasado todas las pruebas.

1.2.7.3 Programación en parejas.

Todo el código debe ser creado por parejas de programadores sentados ambos en frente de un único computador, lo que en principio representa una reducción de un 50% en productividad, sin embargo, según XP no es tal la pérdida. Se entiende que no hay mucha diferencia, en lo que a la cantidad se refiere, entre el código producido por una pareja bajo estas condiciones, que el creado por los mismos miembros trabajando en forma separada, con la excepción que uno o ambos programadores sean muy expertos en la herramienta en cuestión.

Cuando se trabaja en parejas se obtiene un diseño de mejor calidad y un código más organizado y con menos errores que si se trabajase solo, además de la ventaja que representa contar con un compañero que ayude a solucionar inconvenientes en tiempo de codificación, los cuales se presentan con mucha frecuencia.

Se recomienda que mientras un miembro de la pareja se preocupa del método que se está escribiendo el otro se ocupe de cómo encaja éste en el resto de la clase.

1.2.7.4 Integración secuencial.

Uno de los mayores inconvenientes presentados en proyectos de software tiene que ver con la integración, sobre todo si todos los programadores son dueños de todo el código. Para saldar este problema han surgido muchos mecanismos, como darle propiedad de determinadas clases a algunos desarrolladores, los cuales son los responsables de mantenerlas actualizadas y consistentes. Sin embargo, sumando al hecho que esto va en contra de la propiedad colectiva del código no se solucionan los problemas presentados por la comunicación entre clases.

XP propone que se emplee un esquema de turnos con el cual solo una pareja de programadores integre a la vez. De esta forma se tiene plena seguridad de cuál es la última versión liberada y se le podrán hacer todas las pruebas para garantizar

que funcione correctamente. A esto se le conoce como integración secuencial.

1.2.7.5 Integraciones frecuentes.

Se deben hacer integraciones cada pocas horas y siempre que sea posible no debe transcurrir más de un día entre una integración y otra. De esta forma se garantiza que no surjan problemas como que un programador trabaje sobre versiones obsoletas de alguna clase.

Es evidente que entre más se tarde en encontrar un problema más costoso será resolverlo y con la integración frecuente se garantiza que dichos problemas se encuentren más rápido o aún mejor, sean evitados por completo.

1.2.7.6 Estándares y propiedad colectiva del código.

Así como se recomienda que la programación se haga siempre en parejas ubicadas en un único computador, también se aconseja que estas se vayan rotando no solo de compañero, sino también de partes de proyecto a implementar, con el fin de lograr tener una propiedad colectiva del código. Todos y cada uno de los programadores tienen suficiente conocimiento del código de los demás de modo tal que, en cualquier momento puedan continuar la codificación que alguien más empezó sin que represente un traumatismo para nadie.

Uno de los principales motivos por los que se promueve esta práctica dentro de la programación extrema es la posibilidad que brinda de evitar los cuellos de botella. Si una pareja de programadores se retrasa debido a inconvenientes no estimados, pueden ser ayudados o reemplazados por otra pareja que al conocer el código no tendrá que familiarizarse con él.

Para lograr lo anterior se recomienda el establecimiento de *estándares en la codificación*, de modo tal que todo el código escrito por el grupo de desarrollo parezca hecho por una sola persona. No se establecen los aspectos específicos a tener en cuenta dentro de los estándares, sin embargo se aconseja que sean de total aceptación por parte del equipo.

Si bien en la actualidad existen herramientas de soporte en la integración tales como CVS, las cuales ayudan a sobrellevar algunos de los inconvenientes del trabajo en paralelo, es recomendable prestar atención al mecanismo de integración, para evitar problemas en el proyecto que reduzcan bien sea la calidad del proyecto o el rendimiento del equipo de desarrollo.

1.2.8 Pruebas.

XP enfatiza mucho los aspectos relacionados con las pruebas, clasificándolas en diferentes tipos y funcionalidades específicas, indicando quién, cuándo, y cómo deben ser implementadas y ejecutadas.

Del buen uso de las pruebas depende del éxito de otras prácticas, tales como la propiedad colectiva del código y la refactorización. Cuando se tiene bien implementadas las pruebas no habrá temor de modificar el código del otro programador en el sentido que si se daña alguna sección, las pruebas mostrarán el error y permitirán encontrarlo. El mismo criterio se aplica a la refactorización. Uno de los elementos que podría obstaculizar que un programador cambie una sección de código funcional es precisamente hacer que esta deje de funcionar. Si se tiene un grupo de pruebas que garantice su buen funcionamiento, este temor se mitiga en gran medida.

Según XP, se debe ser muy estricto con las pruebas. Sólo se deberá liberar una nueva versión si esta ha pasado con el cien por ciento de la totalidad de las pruebas. En caso contrario se empleará el resultado de estas para identificar el error y solucionarlo con mecanismos ya definidos.

1.2.8.1 Pruebas unitarias

Estas pruebas se aplican a todos los métodos no triviales de todas las clases del proyecto con la condición que no se liberará ninguna clase que no tenga asociada su correspondiente paquete de pruebas. Uno de los elementos más importantes en estas es que idealmente deben ser construidas antes que los métodos mismos, permitiéndole al programador tener máxima claridad sobre lo que va a programar antes de hacerlo, así como conocer cada uno de los casos de prueba que deberá pasar, lo que optimizará su trabajo y su código será de mejor calidad.

Deben ser construidas por los programadores con el empleo de algún mecanismo que permita automatizarlas de modo tal que, tanto su implementación y ejecución consuman el menor tiempo posible permitiendo sacarles el mejor provecho.

El empleo de pruebas unitarias completas facilitan la liberación continua de versiones, por cuanto al implementar algo nuevo y actualizar la última versión, solo es cuestión de ejecutar de forma automática las pruebas unitarias ya creadas para saber que la nueva versión no contiene errores.

1.2.8.2 Pruebas de aceptación.

Las pruebas de aceptación, también llamadas pruebas funcionales son

supervisadas por el cliente, basándose en los requerimientos tomados de las historias de usuario. En todas las iteraciones, cada una de las historias de usuario seleccionadas por el cliente deberá tener una o más pruebas de aceptación, de las cuales deberán determinar los casos de prueba e identificar los errores que serán corregidos.

Las pruebas de aceptación son pruebas de caja negra, que representan un resultado esperado de determinada transacción con el sistema. Para que una historia de usuario se considere aprobada, deberá pasar todas las pruebas de aceptación elaboradas para dicha historia.

Es importante resaltar la diferencia entre las pruebas de aceptación y las unitarias en lo que al papel del usuario se refiere. Mientras que en las pruebas de aceptación juega un papel muy importante seleccionando los casos de prueba para cada historia de usuario e identificando los resultados esperados, en las segundas no tienen ninguna intervención por ser de competencia del equipo de programadores.

1.2.8.3 Cuando se encuentra un error.

Al momento de encontrar un error debe escribirse una prueba antes de intentar corregirlo. De esta forma, tanto el cliente logrará tener completamente claro cuál fue y dónde se encontraba el mismo como el equipo de desarrollo podrá enfocar mejor sus esfuerzos para solucionarlo. Por otro lado se logrará evitar volver a cometerlo.

Si el error fue reportado por el cliente y este creó la correspondiente prueba de aceptación junto al equipo de desarrollo, el programador encargado podrá a su vez producir nuevas pruebas unitarias que le permita ubicar la sección específica donde el error se encuentra.

1.2.9 Proceso de desarrollo en XP.

Todo proyecto de software en XP inicia con una o varias reuniones con el cliente, en las cuales se da claridad a la necesidad puntual del mismo a través de las historias de usuario. Estas también, sirven de base para crear una metáfora del sistema con el cual todo el equipo de trabajo tendrá una idea general de la aplicación a implementar. Con base en las historias de usuario, se crean las pruebas de aceptación las cuales deben ser diseñadas antes de iniciar la codificación.

Concluida esta etapa, se debe acordar un plan de entregas con el cliente, del cual

surge un número inicial de iteraciones y duración de las mismas. Esta reunión de entregas puede repetirse en el transcurso del proyecto, siempre que la velocidad del mismo cambie lo suficiente para tener que replantear el plan de entregas o que surjan nuevas historias de usuario que justifiquen la alteración de dicho plan. Dentro de esta(s) reunión(es) de planeación de entregas debe considerarse la realización de algunos Spike Solution para tener claridad sobre la dificultad y tiempo necesario para implementar determinada historia de usuario.

Toda iteración debe iniciar con una reunión en la que se da claridad a las tareas a desarrollar, basándose en el plan de entregas, la velocidad del proyecto y las historias de usuario sin concluir de la iteración anterior. De esta reunión se obtiene un plan que sirve de hoja de ruta en el transcurso de la iteración.

Todos los días debe hacerse una reunión corta en la cual se discute el avance de la iteración, basándose en el plan obtenido de la reunión de inicio de iteración y las tareas concluidas con el cual se acuerda el trabajo del día.

2. OBTENIENDO E INTERPRETANDO LA INFORMACIÓN DEL PROBLEMA

En este capítulo se dará un recuento de la información recopilada en el trabajo de campo realizado, se hará un análisis de la misma con miras a establecer el modelo y diseño de la aplicación de software propuesta para la solución del problema.

2.1 PROCESO MANUAL DE PRESTACIÓN DE SERVICIO DE TRANSPORTE INDIVIDUAL URBANO

El proceso en la empresa PrimerTax S.A., consta de los siguientes pasos:

- Un cliente llama a la empresa.
- Un operador atiende la llamada registrando manualmente en su cuaderno la dirección de donde se está solicitando el servicio.
- El operador se comunica por radio con los taxis en servicio, comunicándoles el sector al que pertenece la dirección del servicio solicitado y espera la respuesta al llamado por el mismo medio, para el cual se pueden presentar varios móviles.
- Los móviles cerca de la zona se reportan.
- El operador escoge el que esté más cerca del destino y le da la dirección exacta del servicio.
- El operador comunica al cliente cual es el número del móvil que le fue asignado y lo registra anexo a la dirección.

2.1.1 Registro de información del servicio.

Los datos registrados por cada servicio son:

- Dirección que solicita el servicio.
- Número de lateral del móvil (Es un identificador único de la empresa. PrimerTax tiene la letra H y el número, ejemplo H-123).

Los datos se pueden consultar posteriormente, para saber que móvil fue asignado a determinado servicio. Estas consultas son realizadas generalmente pocos minutos después del registro, la consulta a más largo plaza se realiza esporádicamente.

2.1.2 Sistema identificador de llamadas.

Hardware que recibe una llamada telefónica e identifica el código telefónico de origen de la llamada. Dependiendo del modelo del aparato, la cadena será interpretada y enviada a través de una cadena con una determinada estructura, la

mayoría de los modelos diseñados permite obtener la información decodificada por medio de un puerto serial o USB.

2.1.3 Móvil o vehículo.

Automóvil con el cual se presta servicio de transporte individual urbano, los móviles usados por la empresa PrimerTax S.A. son de color amarillo, con letras negras que informan el número telefónico a través del cual se puede solicitar servicio de transporte.

2.1.4 Conductor.

Es la persona que se encuentra inscrita en la empresa y que cumple con los requerimientos para conducir un móvil y ofrecer servicio de transporte a los clientes que llaman a la empresa para solicitar servicio de transporte.

2.1.5 Cliente.

Se refiere a los clientes como las personas que solicitan servicio de transporte a la empresa PrimerTax S.A. Los clientes pueden solicitar servicio de transporte a través de diferentes medios: línea telefónica, radios instalados en unidades residenciales y/o el radio de otro móvil.

2.1.6 Operador.

Se refiere a los trabajadores de la empresa PrimerTax S.A. que atienden a los clientes que solicitan servicio por vía telefónica o por radios de las unidades.

2.2 INTERPRETACIÓN DEL PROBLEMA

A partir de la información recopilada mediante una reunión realizada a los operadores de la empresa PrimerTax S.A., se pudo ver que es necesario desarrollar un sistema que permita agilizar la atención de una llamada y el despacho de un móvil a un destino, un despacho consiste en atender una llamada, solicitar una dirección origen del cliente, informar dicha dirección a través de un radio a un conductor para que este preste el servicio de transporte al cliente. Es importante denotar que en dicho proceso se realiza el registro del préstamo del servicio, el registro se realizaba manualmente por lo que en ocasiones el dictado de la dirección por parte del cliente se realiza más de una vez.

En los registros de los servicios prestados que se realizan por parte del operador que atiende la llamada del cliente, se encuentran especificados los datos que se guardan como históricos de los servicios prestados, la información es almacenada en cuadernos en donde se escribe la fecha, la dirección y el móvil que realiza la prestación del servicio.

Dentro de las funciones operativas que se realizan para el registro de una prestación de servicio o el despacho de taxis, el operador de la empresa PrimerTax S.A. solicita la dirección del cliente en donde el conductor lo recogerá para transportarlo a un determinado sitio. El sistema deberá contar con un mecanismo que permita asociar una dirección de origen y un móvil a un servicio prestado y que dicha dirección sea obtenida de manera automática por el sistema a partir del número telefónico del cliente. Otra parte del software permitirá administrar la información registrada en el sistema.

Adicionalmente, la información recopilada en las primeras reuniones de requerimientos realizados tanto con el gerente de la empresa como con los operadores, a los operadores de la empresa PrimerTax S.A., que se encuentra en la Historia de Usuario No 1 (véase Ilustración 3. Historia de Usuario No 1, página 28) de este documento, se nota claramente que la mayoría de los operadores no usa con frecuencia los sistemas de cómputo para realizar alguna tarea cotidiana o realizar algún tipo de actividad fuera del entretenimiento y no conocen de aplicaciones informáticas para el registro de información corporativa. Por tanto, también se ve la necesidad de capacitar y explicar a los operadores de los beneficios que tiene el uso de esta tecnología y las ventajas que les puede aportar en su proceso operativo.

Ilustración 3. Historia de Usuario No 1

HISTORIA DE USUARIO

Número:	1	Nombre de Historia de Usuario:	Requerimiento Inicial Formulación del Proyecto
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			——— a ———
Usuario:	Lina Ma Muñoz.	Iteración Asignada:	1. Propuesta de Sistema. V1.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se requiere diseñar una propuesta para el sistema de Identificación automático de llamadas con las siguientes caracteristicas: - Se debe reconocer automaticamente la dirección luego de consultar el número telefónico en la base de datos. - La dirección podrá ser modificada si el cliente confirma una ubicación distinta en la base de datos. - Se permite asignar un móvil para prestar el servicio - Se debe guardar en la base de datos la información de prestamo de servicio, se debe guardar: Fecha y hora, movil, dirección en la q' se prestó el servicio. - La información de los servicios prestados, debe poderse consultar en pantalla (para confirmar datos) - De la información histórica se deben poder generar reportes.		
Observaciones:	- Los reportes generados podran consultar: • moviles con mayor numero de servicios prestados • operador con mayor numero de servicios prestados • Cliente con mayor número de servicios solicitados. • Las consultas de los reportes podrán ser ejecutadas para una fecha orango de fechas. • Se requiere que el sistema controle el acceso al sistema mediante una ventana de validación de usuarios. NOTA: Solo se requiere diseño de navegabilidad y propuesta de entorno grafico.		

Por último, este sistema pretende ser de gran ayuda para los gerentes y administradores de la empresa, ya que estos no cuentan con suficientes

herramientas para realizar análisis de la productividad en cuanto se trata de prestación de servicio de transporte individual urbano, como lo expresa el Ingeniero Jesús Ernesto Duque en la Historia de Usuario No 1(véase Ilustración 3. Historia de Usuario No 1, página 28) también puede presentar una gran ventaja puesto que no se necesitará tener almacenado los cuadernos físicos de registro de los servicios prestados y los operadores no requerirán la solicitud de la dirección por parte de un cliente.

3. PRESENTACIÓN

En este capítulo se hace una introducción a las condiciones del entorno que rodearon el proyecto entre las cuales se resaltan aspectos relacionados con el cliente, el negocio y los usuarios finales para los cuales se desarrolló.

Adicionalmente, se realiza una descripción de las herramientas empleadas para el desarrollo del proyecto y los criterios por los cuales fueron elegidas.

3.1 HERRAMIENTAS EMPLEADAS

Se optó por seleccionar herramientas de *Microsoft* para el desarrollo del GUI de la aplicación y para el motor de la base de datos *PostgreSQL*, a continuación se detalla cada una de las herramientas especificando los motivos por los cuales fueron seleccionadas.

3.1.1 Visual Studio 2005 Express Editions (Microsoft).

Es una versión libre de la herramienta de desarrollo, permite crear aplicaciones para el Sistema Operativo Microsoft Windows. Se optó por la versión 2005 la cual, al momento de la selección de herramientas era la versión más reciente.

El motivo por el cual se seleccionó Visual Studio como herramienta de desarrollo, es que permite desarrollar aplicaciones con un entorno familiar a Microsoft Windows, lo que otorgaría una aplicación que en su interfaz gráfica fuera más aceptable por los usuarios finales (*operadores*), quienes manifiestan poca experiencia en el uso de sistemas de computación.

3.1.2 MySQL.

Es el motor de base de datos usado para el proyecto, ha sido la base de datos más popular de código abierto debido a su consistente y rápido rendimiento, fiabilidad y fácil uso. Es usado en todos los continentes por pequeños desarrolladores web tanto como por grandes compañías para ahorrar tiempo y dinero.

Es poderoso en el manejo de grandes volúmenes de datos, sitios Web, sistemas críticos de negocio y paquetes de software incluyendo industrias reconocidas

como: Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, and Zappos.com.

MySQL corre en más de 20 plataformas incluyendo: Linux, Windows, Mac OS, Solaris, HP-UX, IBM AIX.

Los motivos por los cuales seleccionamos este motor de base de datos fueron sus buenas características, el conocimiento de los desarrolladores del motor y adicionalmente, que no causaba costos al proyecto gracias a ser completamente libre.

3.1.3 StarUML.

Es un proyecto abierto para el desarrollo rápido, flexible, extensible, y libre plataforma de UML/MDA que funciona en la plataforma Win32. Es un reemplazo de las herramientas comerciales de UML tales como RationalRose.

Esta herramienta fue seleccionada porque no causaba costos al proyecto gracias a ser una herramienta de libre distribución, adicionalmente es una herramienta de fácil uso que le proporcionaba al proyecto el suficiente soporte.

3.2 DESCRIPCIÓN DEL NEGOCIO

Se trata de una empresa de despacho de taxis que atiende toda la zona metropolitana de la ciudad de Pereira.

Al momento de dar inicio a la construcción del proyecto, la empresa realizaba el registro de la información en cuadernos, cada operador de la empresa tenía un cuaderno propio, por lo tanto se tenía un cuaderno por operador, adicionalmente en cada cuaderno el operador guardaba el registro de prestación de servicios por un tiempo no definido, en algunos casos un cuaderno podía tener el registro de tres o más semanas de despachos realizados.

El mecanismo de almacenamiento de los cuadernos en donde se tenían los históricos no era confiable, y en algunos casos los operadores no los guardaban adecuadamente. Se presentaron casos en los que se requería obtener información de registros históricos de una determinada fecha, pero no era posible. Por razones administrativas y de evolución de los procesos operativos de la empresa, se acordó desarrollar un software que permitiera un registro confiable de la información histórica y que permitiera la consulta y administración de dicha información de una manera práctica.

3.3 DESCRIPCIÓN DEL CLIENTE

Para este proyecto se contó con el cliente y la asistencia de 4 operadores seleccionados por él mismo.

Las reuniones con el cliente se realizaban esporádicamente, pero los operadores o usuarios finales del sistema se encontraban presentes en las entregas de las iteraciones y colaboraban con las pruebas de aceptación de las iteraciones. La participación de los usuarios finales, permitió a los desarrolladores evaluar la facilidad de uso y la aceptación de los diseños propuestos en las interfaces, igualmente, se aportaban muchas mejoras de bajo impacto para el proyecto que a su vez, enriquecían los resultados esperados por el cliente.

4. PLANEACIÓN

A partir de este capítulo se describe la experiencia vivida a lo largo del desarrollo del proyecto. Inicialmente se expone cada uno de los aspectos que propone XP para la etapa de planeación. Para cada uno de los pasos indicados en las etapas, se enuncia las instrucciones dadas por XP contrastándolo con la experiencia real vivida durante el desarrollo del proyecto. Entre los elementos a discutir para este capítulo se encuentran las historias de usuario, el plan de entregas, lo relacionado a las iteraciones del proyecto y las modificaciones aplicadas a la metodología XP para adaptarla al proyecto.

4.1 HISTORIAS DE USUARIO

4.1.1 Lo que dice la metodología XP.

Las Historias de Usuario son de interés para la entrega del Producto, son construidas con la información proporcionada por los clientes y el equipo de desarrollo comenzará a familiarizarse con las herramientas, metodología y prácticas que serán usadas para realizar el proyecto.

La siguiente tabla permite realizar un comparativo entre las *Historias de Usuario y los Casos de uso*[2] usados en metodologías tradicionales.

Tabla 2. Casos de Uso Vs Historias de Usuario

CONCEPTO	CASOS DE USO	HISTORIA DE USUARIO
OBJETIVO	Modelar la iteración entre un "Actor" y el sistema.	Redactar una descripción breve de una funcionalidad tal y como la percibe el usuario.
ESTRUCTURA	Texto detallado donde se sigue una plantilla predefinida a completar con conceptos técnicos (objetivo, resumen, actor, evento, disparador, extensiones, etc.)	Corta y consistente en una o dos frases escritas en el lenguaje del usuario.
PLANIFICACIÓN	No se utilizan para planificar	Se utilizan para planificar las entregas.
AGILIDAD	Requieren tiempo para análisis y redacción de plantillas predefinidas.	Se pueden escribir en pocos minutos.

[2] GROSJEAN Claude Jean, Use cases - User Stories: so precious but not the same!, Disponible en Internet: http://www.agile-ux.com/2009/01/23/use-cases-user-stories-so-precious-but-not-the-same/

COMPRENSIÓN	Suelen ser de difícil comprensión, incluso para personal técnico.	Fáciles de leer y comprender.
MANTENIMIENTO	Suelen pertenecer a documentos con cientos de páginas, difíciles de mantener.	Muy fáciles de mantener.
COMUNICACIÓN	Modelo textual asociado con diagramas: todo tiene que estar escrito.	Basada en la comunicación verbal y orientada a la colaboración y discusión para clarificar detalles.
SOPORTE	Escritos en documentos con el objetivo de que estos sean archivados como documentos de referencia.	Escritas en tarjetas (teóricas o reales) con el objetivo de que sean usadas directamente.
DURACIÓN	Debe ser implementada y probada en una única iteración.	Puede ser implementada en varias iteraciones.
AUTORES	Definidos por "intérpretes" (analistas, consultores, etc.)	Posibilidad de ser definidas por usuarios y clientes.
PRUEBAS	La definición de pruebas, se redacta en documentación separada.	Contienen "Pruebas de Aceptación" en el reverso de la tarjeta.
CONTEXTO	Proporciona una visión más general del sistema y la integración en él.	Proporciona una visión menos obvia, por eso las pruebas y el *feedback* de los usuarios es tan importante en las metodologías ágiles.
METODOLOGÍA	Asociado con RUP.	Asociados con la programación extrema (aunque pueden usarse en RUP)

4.1.1.1 Características

- Escritas por el usuario
- Terminología del Cliente
- Bajo nivel de detalle
- Sirve de base para estimar los tiempos de implementación.[3]

4.1.2 Experiencia en el Proyecto.

Las historias de usuario fueron diligenciadas por alguno de los desarrolladores, con el fin de que el cliente pudiera concentrar su atención en el análisis del

[3] WELL, Don, Extreme Programming: A gentle introduction. Extreme Programming – Agile Process [en línea]. <http://www.extremeprogramming.org/rules/userstories.html>

requerimiento o en el caso de que se estuviera evaluando el diseño o una entrega de iteraciones. Por otro lado se contó con la participación de los usuarios finales (operadores) para la redacción y recolección de información para algunas historias de usuario. Pese a que el cliente no fue quien escribió y diligenció la historia de usuario, siempre se contó con su revisión previa antes de finalizar la reunión.

Desde el punto de vista de nivel de detalle, las historias de usuario se realizaron de manera jerárquica en la que las historias de usuario más específicas contenían los aspectos importantes de diseño, pero no se profundizaban en procesos o con descripciones de carácter técnico, las historias de usuario más generales solo mencionan los requerimientos con el mínimo nivel de detalle. A continuación se mencionan algunas de las historias de usuario ordenadas de manera jerárquica:

1. Requerimiento Inicial
    1.1. Información de Llamadas Entrantes
        1.1.1. Desplegar Información de Llamadas
            1.1.1.1. Búsqueda de Dirección del Cliente
        1.1.2. Desplegar la información de llamadas entrantes en un número de líneas constante.
        1.1.3. Conteo de tiempo de una línea en disposición de ser atendida.
        1.1.4. Despliegue de Información de varias Llamadas Consecutivas.
        1.1.5. Estadísticas de llamadas atendidas y no atendidas, el registro se hace con base en el tiempo máximo de atención de una llamada entrante.
    1.2. Usuarios del Sistema
        1.2.1. Datos de Usuarios del Sistema
        1.2.2. Ventana de Validación de Usuarios en el sistema.
        1.2.3. Usuarios Administradores
            1.2.3.1. Adición de Usuarios al Sistema
            1.2.3.2. Modificar información de Usuarios en el Sistema
            1.2.3.3. Eliminar Usuarios del Sistema
            1.2.3.4. Adición de Móviles en el Sistema
            1.2.3.5. Modificar Información de Móviles en el Sistema
            1.2.3.6. Eliminar Móviles del Sistema
            1.2.3.7. Adición de Registros al Sistema (Registros de servicios prestados)
            1.2.3.8. Modificar Registros del Sistema (Registros de Servicios prestados)
    1.3. Móviles del Sistema
        1.3.1. Información de los Móviles en el Sistema
    1.4. Históricos de Servicios Prestados
    1.5. Generación de Reportes
        1.5.1. Reporte de Registro de Llamadas Atendidas por un operador.
        1.5.2. Reporte de Servicios prestados por un móvil.
        1.5.3. Reporte de todos los servicios prestados.

1.5.4. Reporte de servicios prestados a un cliente.
1.5.5. Reporte de Operador con mayor número de Servicios prestados.
1.5.6. Reporte de Móvil con mayor número de Servicios Prestados.
1.5.7. Reporte de operador con mayor número de despachos realizados.

Posterior a la recolección de la historia de usuario, el equipo de desarrollo se reunía a estimar los tiempos de desarrollo, las historias de usuario no se realizaban con detalles técnicos, por lo tanto, la implementación del mismo se hacía según lo convenían los desarrolladores. Los tiempos estimados para los desarrollos se cumplieron sustancialmente, lo que permitía cumplir con las entregas comprometidas en las fechas especificadas al cliente (ver ANEXO B: HISTORIAS DE USUARIO).

El éxito tenido en las entregas en los tiempos planeados a partir de las historias de usuario, ayudaron a mantener el interés y la confianza en el proyecto por parte del cliente y de los usuarios finales.

4.2 VELOCIDAD DEL PROYECTO

4.2.1 Lo que dice la metodología XP.

> Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración.[4]

La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias.

Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. Al planificar según alcance del sistema, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación.

[4] SÁNCHEZ GONZÁLEZ, Carlos. ONess: un proyecto open source para el negocio textil mayorista desarrollado con tecnologías open source innovadoras. Disponible desde Internet: <http://oness.sourceforge.net/proyecto/html/ch05s02.html>

Puntos Clave:

- Número de Historias de Usuario realizadas en una iteración.
- Número de Puntos asociados a las historias de usuario realizadas en una iteración.
- Sirve para estimar el número de iteraciones necesarias para implementar una historia de usuario o la cantidad de historias implementadas en una iteración.

4.2.2 Experiencia en el Proyecto.

La estimación del número de iteraciones requeridas para implementar una historia de usuario se realizó conforme a los puntos estimados de las historias de usuario de mayor detalle, por lo que era posible estimar de manera aproximada el tiempo requerido. Sin embargo el cálculo en la velocidad del proyecto, estimando la cantidad de historias de usuario implementadas por iteración no fue constante, por lo que la velocidad del proyecto no fue útil; la velocidad fue directamente dependiente de la dificultad asignada a las historias de usuario que se iban desarrollando.

Tabla 3. Iteraciones y velocidad del proyecto

Iteración	Resumen	Horas	Semanas	Horas Semanales	Historias de Usuario
1	Entrega de prototipo inicial. (propuesta PowerPoint)	30	1	30	3
2	Entrega de Prototipo inicial en Visual Studio, navegación de ventanas GUI inicial.	60	3	20	1
3	ARQUITECTURA DEL SISTEMA Creación de Esquema de Base de datos. Implantación de Esquema de Base de Datos. Implantación de Esquema de Base de Datos. Creación de datos iniciales del sistema.	65	3	35	2
4	Recepción de cadenas desde el aparato identificador de llamadas. Corrección de cadenas de entrada.	90	3	30	2
5	Editar direcciones nuevas en la base de datos	50	2	30	1
6	Despliegue de llamadas entrantes. Alta y baja de identificadores en el sistema.	30	1	30	2
7	Atención de llamadas, despacho de servicios.	70	2	35	1

8	Atención de líneas, cambio en mecanismo de selección de líneas a atender.	75	3	30	1
9	Búsqueda en historial. Asignación de pendientes. Obtención automática de Móviles	89	3	30	3
10	Administrador del sistema. Generación de reportes	90	3	30	2
11	Registro de llamadas no atendidas. Reporte de llamadas no atendidas.	65	3	30	2
12	Sanciones. Llenado de datos. Asignación de varios Móviles.	60	2	30	4
13	Consulta de información de móviles. Contador de Servicios prestados.	35	1	35	2
14	Planillas	25	1	30	1
15	Registro manual de servicios. Registro de Emergencias.	60	2	31	2
16	Súper Backup Mejora en seguridad de emergencias.	90	3	30	2

Durante la realización del proyecto no se tuvo en cuenta la estimación de la velocidad del proyecto para la planeación de entregas de versiones o la duración de las iteraciones. El grupo de desarrollo decidió planear las entregas y el número de historias de usuario por cada iteración, según el número de puntos asignados a las historias de usuario (el número de puntos asociados a las historias de usuario fueron asignados según la dificultad de la implementación de la historia).

Durante la estimación de los puntos por cada historia de usuario y el orden de la entrega en las iteraciones se tuvo en cuenta: la puntuación del riesgo en el desarrollo y la prioridad en negocio, de manera cualitativa y cuantitativa. El riesgo causado por cambios en requerimientos y por reprocesos en el desarrollo debido a errores de comunicación con los componentes externos de hardware o errores de lógica fueron cuantificados de la siguiente forma:

- BAJO: corresponde a cambios o reprocesos que modifican los puntos estimados para la historia de usuario en un tiempo no superior a 8 horas.
- MEDIO: corresponde a cambios o reprocesos que modifican los puntos estimados para la historia de usuario en un tiempo superior a 8 horas y menor a 16 horas.
- ALTO: corresponde a cambios o reprocesos que modifican los puntos estimados para la historia de usuario en un tiempo superior a 16 horas

El orden dado a la entrega y desarrollo de las historias de usuario, se planeó conforme a la importancia de la funcionalidad específica a la que hacía referencia la historia de usuario. De la siguiente manera se calificó la prioridad de las historias de usuario:

- ALTA: corresponde a funcionalidades que hacen parte del proceso operativo diario del sistema.
- MEDIA: corresponde a funcionalidades que hacen parte del proceso administrativo periódico del sistema.
- BAJA: corresponde a funcionalidades que hacen parte de mejoras que no afectan con gran impacto las funcionalidades de alta o mediana prioridad.

4.3 ENTREGAS EN ITERACIONES

4.3.1 Lo que dice la metodología XP.

El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio).

Al final de la última iteración el sistema estará listo para entrar en producción. Los elementos que deben tomarse en cuenta durante la elaboración del Plan de la Iteración son: historias de usuario no abordadas, velocidad del proyecto, pruebas de aceptación no superadas en la iteración anterior y tareas no terminadas en la iteración anterior. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores.[5]

4.3.2 Experiencia en el Proyecto.

El proyecto fue dividido en iteraciones que no superaban 3 semanas, en la primera iteración se desarrollo una propuesta general del sistema que fue solicitada por el cliente, las iteraciones siguientes fueron desarrolladas y expuestas al cliente y a los usuarios finales (operadores). Para el caso de este proyecto a partir de la tercera iteración se tenía una versión del proyecto. Cada versión se instalaba en

[5] SÁNCHEZ GONZÁLEZ, Carlos. ONess: un proyecto open source para el negocio textil mayorista desarrollado con tecnologías open source innovadoras. Disponible desde Internet: <http://oness.sourceforge.net/proyecto/html/ch05s02.html>

los equipos del cliente, esto se realizó con el propósito de que los usuarios finales se adaptarán al sistema evolutivamente y que al mismo tiempo desarrollaran propuestas para discutirlas con el cliente, proponer los cambios necesarios para hacer que el software fuera de fácil uso y fácil acceso a las herramientas que los usuarios necesitarían, para ejecutar correctamente el proceso operativo del negocio.

Las Iteraciones relacionadas con la entrega y llenado de los datos del directorio se atrasaron hasta el final del proyecto, esta tarea estaba asignada a un tercero que tuvo dificultades para adquirir la información.

4.4 PLAN DE ENTREGAS

4.4.1 Lo que dice la metodología XP.

Las entregas son realizadas conforme a lo estimado en el tiempo de implementación de las historias de usuario y las prioridades del cliente. Criterios de planificación de entregas:

- Reunión Inicial del proyecto: en donde el cliente expresa la idea de cómo será el orden lógico según el negocio para la puesta en marcha del proyecto.
- Cuales historias de usuario serán implementadas en cada entrega.
- Grado de relevancia, prioridad y dificultad de la historia de usuario.
- Se hace un estudio aproximado de la entrega de las iteraciones.[6]

4.4.2 Experiencia en el Proyecto.

Se realizaron dos reuniones iniciales, en la primera nos reunimos con el señor Juan Carlos Tamayo, quien nos planteó a grandes rasgos los requerimientos del sistema. La segunda reunión se llevó a cabo en las oficinas de PrimerTax S.A en donde el Ingeniero Jesús Ernesto Duque Ocampo gerente de PrimerTax S.A. expresó con mayor detalle el sistema que tenía en mente. Adicionalmente, se tuvieron en cuenta temas como la experticia de los usuarios finales en el manejo de sistemas de información, el tiempo que disponía el gerente para la realización de las reuniones y se asignaron los colaboradores que para este caso fueron dos operadores que realizarían las pruebas del sistema.

[6] CASTILLO, Oswaldo; FIGUEROA Daniel y SEVILLA Hector. Fases de la Programación Extrema. Programación Extrema [en línea]. <http://programacionextrema.tripod.com/fases.htm#primeraFase>

XP recomienda que el orden de las historias implementadas en las iteraciones sean determinadas por el cliente, sin embargo, las historias de usuario entregadas en las iteraciones fueron seleccionadas según la disponibilidad de los sistemas externos (hardware), la dificultad de la implementación y el riesgo de los cambios; estos aspectos también fueron mezclados con la prioridad del negocio. La razón por la que se tomaban primero las historias de usuario de mayor dificultad, fue porque las mejoras podrían ser entregadas en más de una futura iteración, dando más tiempo de perfeccionarlas antes de dar por finalizado el proyecto.

Para aproximar el tiempo de entrega se tuvo en cuenta los siguientes aspectos:

- XP recomienda que una semana conste de 40 horas semanales, es decir 8 horas laborando de lunes a viernes. Sin embargo en el proyecto se estableció que la semana estaría compuesta por 30 horas semanales que serían distribuidas en 4 horas de lunes a viernes, 5 horas sábado y domingo, estos horarios de trabajo fueron convenidos debido a conocimiento de que el grupo de desarrollo tenía obligaciones adicionales al proyecto.
- Las reuniones se harían en horarios luego de las seis (6) de la tarde, esto para evitar coincidir con horarios críticos en cuanto a la demanda del servicio en la empresa de despacho.
- Las reuniones con el cliente para evaluar las entregas realizadas y la evaluación de nuevas historias de usuario se realizaban los días sábado.

4.5 REUNIÓN INICIAL DE ITERACIÓN

4.5.1 Lo que dice la metodología XP

“En la primera iteración se establece una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra traduciendo las historias en tareas de cada iteración; se estiman los tiempos para cada iteración en días ideales.”[7]

4.5.2 Experiencia en el Proyecto.

Luego de la recolección de las historias de usuario, el equipo de desarrollo se

[7] SÁNCHEZ GONZÁLEZ, Carlos. ONess: un proyecto open source para el negocio textil mayorista desarrollado con tecnologías open source innovadoras. Disponible desde Internet: <http://oness.sourceforge.net/proyecto/html/ch05s02.html>

disponía a crear tareas plasmadas en las tarjetas CRC (más adelante se detalla la creación y formato de estas tarjetas), se asignaban las tareas a un responsable para la construcción. Esta actividad fue realizada por los dos desarrolladores del proyecto y se realizó la revisión de pares para hacer ajustes o detectar falencias en los diseños de la construcción, los reprocesos se asignaban a ambos integrantes, dependiendo del error, el reproceso podría ser realizado por otro desarrollador distinto al que causó el reproceso.

4.6 REUNIÓN MATINAL

4.6.1 Lo que dice la metodología XP.

- "Se realiza una reunión al iniciar cada día.
- La reunión se realiza en el sitio de trabajo del equipo.
- Se evitan las discusiones largas."[8]

4.6.2 Experiencia en el Proyecto.

Se practicaron reuniones a diario antes de iniciar la jornada laboral planeada, estas reuniones fueron cortas y estaban planeadas para un tiempo máximo de 30 minutos, sin embargo, en situaciones en las que se requería discutir los resultados en pruebas no exitosas realizadas previamente, el informe se extendía a 2 horas como máximo.

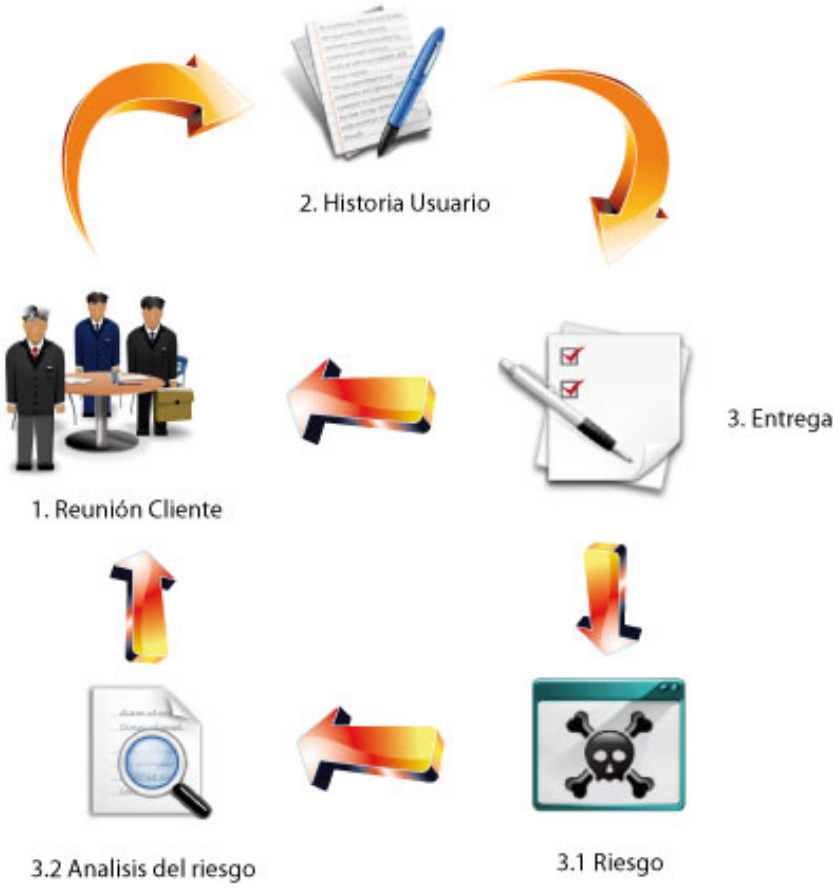
Las reuniones diarias permitían compartir obligaciones, ideas para solucionar problemas de diseño y para atacar las situaciones no controladas que se presentaban al momento de poner en marcha la implementación de una historia de usuario. Sumado a esto, se tenía un seguimiento preciso del avance en las obligaciones asignadas a cada desarrollador, en caso de que el desarrollo se estuviera desarrollando por ambos, se establecían los objetivos del día al igual que se podía hacer un corto seguimiento a lo realizado el día previo.

Cuando las discusiones se extendían, tenían como resultado un avance en la solución para la implementación de alguna tarea, o el estudio de algún riesgo, de lo contrario solo se trataban temas de bajo detalle y sobre todo, se trataba de solucionar dudas con respecto a herramientas de codificación y de pruebas.

[8] WELL, Don, Extreme Programming: A gentle introduction. Extreme Programming – Agile Process [en línea]. <http://www.extremeprogramming.org/rules/standupmeeting.html>

Como se observa en la Ilustración 4. Proceso de reuniones, el tema de las reuniones giraba en torno a las historias de usuario, sea una reunión con el usuario, cliente o equipo de desarrollo, se trataban temas para la entrega, sus posibles riegos con su respectivo análisis de riesgos que luego eran socializados en la reunión.

Ilustración 4. Proceso de reuniones



La anterior ilustración representa el ciclo de las reuniones, este ciclo fue definido y seguido por los miembros del equipo de desarrollo. Los pasos de este ciclo se describen a continuación:

- Inicia con una reunión con el cliente, durante esta reunión el cliente manifiesta los requerimientos que tiene en mente al equipo que se encuentra recolectando información.
- En este proyecto, uno de los miembros del equipo que recolecta la información es quien toma nota de los requerimientos, a medida que el cliente relata sus necesidades se arma la historia de usuario con

esta información.

- Dados los detalles de la historia de usuario, el grupo de desarrollo comenta una propuesta ideada durante la descripción de la historia de usuario; en el caso de que sea posible el equipo define un tiempo de entrega de la historia de usuario, en caso de que no se pueda debido a la detección de un riesgo entonces, pospone su respuesta hasta un par de días más.
- Cuando se detecta un riesgo, el equipo analiza si la implementación de la historia de usuario implica mayores costos de tiempo o financieros. En caso de que la implementación efectivamente afecte los tiempos o costos planeados, procede a informar al cliente los resultados del análisis. El informe dado al cliente se realizaba ya sea, personalmente con una corta visita o telefónicamente.
- Reunión posterior o de cierre, en esta reunión se presentaba el desarrollo de la historia de usuario, en caso de que el cliente

4.7 MOVER PERSONAL

4.7.1 Lo que dice la metodología XP.

"El código es de conocimiento de todo el grupo de desarrollo, todos los desarrolladores deben conocer el código de todos los módulos necesarios en el proyecto, para lograr este objetivo se recomienda hacer rotación de los desarrolladores, de esta manera los desarrolladores aprenderán todos los módulos a medida que se va desarrollando el proyecto."[9]

4.7.2 Experiencia en el Proyecto.

El conocimiento del código fue compartido por ambos desarrolladores, siendo solo dos personas las encargadas en el proyecto, el conocimiento del código se dio con el desarrollo en pareja o por la revisión de pares. Adicionalmente, el proyecto estaba compuesto de pocos módulos principales y cada miembro del grupo estuvo directamente relacionado con él en algún momento del desarrollo del proyecto.

[9] WELL, Don, Extreme Programming: A gentle introduction. Extreme Programming – Agile Process [en línea]. <http://www.extremeprogramming.org/rules/standupmeeting.html>

4.8 MODIFICAR XP CUANDO SEA NECESARIO

4.8.1 Lo que dice la metodología XP

"Corrija las reglas de XP cuando se detecten fallas que ocasionen las reglas de XP no apliquen por completo al proyecto.
Las reglas deben ser modificadas cuando todos los miembros del equipo aprueben los cambios y validen la conveniencia de los cambios."[10]

4.8.2 Experiencia en el Proyecto.

La principal causa de los cambios aplicados a la metodología se dieron con base en que el proyecto fue realizado solo por dos personas, las cuales tenían compromisos de tipo académico que les impedía laborar en el tiempo y horario sugerido por la metodología. Los cambios realizados a la metodología fueron:

- Jornada Laboral: Se adaptó el horario de trabajo y el número de horas laboradas por día, esto fue definido según la disponibilidad de horario de los dos desarrolladores ya que aún se tenían obligaciones académicas.
- Programación en Parejas: no siempre los dos miembros laboraban en el mismo computador ni en el mismo sitio físico, cuando se hacían desarrollos por separado, se realizaba una revisión de pares, esto no se encuentra establecido ni propuesto en la metodología, pero permitían suplir el tracker del proyecto.
- Cliente en Sitio: El cliente no se encontraba en el sitio de desarrollo, a cambio de esto se realizaban visitas muy seguidas (2 por semana) para hablar con los usuarios finales autorizados por el cliente, quienes ayudaban a definir cambios requeridos antes de la reunión con el cliente y durante la implementación de una iteración. Los operadores como usuarios finales fueron de gran utilidad gracias al conocimiento del negocio y por la claridad que tenían con respecto a lo que se pretendía tener al finalizar el proyecto.

[10] WELL, Don, Extreme Programming: A gentle introduction. Extreme Programming – Agile Process [en línea]. <http://www.extremeprogramming.org/rules/fixit.html>

5. DISEÑO

A diferencia de las metodologías pesadas, el diseño se realiza durante todo el tiempo de vida del proyecto, siendo constantemente revisado y muy probablemente modificado debido a cambios presentados durante el desarrollo.

Tal como se presentó en el capitulo anterior, este capítulo presenta una estructura similar a la sección de diseño del marco teórico donde se observará para cada uno de los elementos constitutivos de dicha etapa una serie de ideas que describen la teoría contrastada con lo vivenciado durante la ejecución del proyecto.

Entre los elementos más importantes que menciona XP referentes al diseño esta la simplicidad, las tarjetas CRC, el refactoring y Spike Solution. A continuación se detalla la experiencia vivida con cada uno de ellos.[11]

5.1 SIMPLICIDAD

5.1.1 Lo que dice la metodología XP

- *El diseño debe ser sencillo*
- *Sólo se crearán diagramas útiles.*[12][13]

5.1.2 Experiencia en el Proyecto.

En lo que respecta a la sencillez del diseño, se acogió la recomendación de XP, sólo invirtiendo el tiempo exclusivamente necesario en elaboración de diagramas y diseño de interfaz grafica. Como consecuencia de esta decisión, se debieron hacer algunos sacrificios. Al no haberse realizado muchos diagramas, la orientación a objetos no fue tan completa, sacrificando de esta forma escalabilidad, versatilidad y elegancia del diseño, lo que fue considerado un precio justo a cambio del cumplimiento de los plazos. Desde el punto de vista de las interfaces, se trabajó para que fuera sencilla de usar para los usuarios, se prestó mucha atención a ubicar los elementos de una forma intuitiva y de fácil manejo, el usuario fue de mucha ayuda al especificar lo que era más amigable para él. Como resultado se obtuvo una reacción muy positiva del cliente, manifestando

[11] http://www.agile-process.org/proverbs.html
[12] http://www.extremeprogramming.org/rules/simple.html
[13] http://www.xprogramming.com/Practices/PracSimplest.html

conformidad con la apariencia visual de la aplicación. Es importante aclarar que, estos sacrificios en ningún momento representaron una baja en la calidad de la aplicación en cuando a funcionalidad se refiere.

En lo que refiere a los diagramas, se crearon la tarjetas CRC, algunos diagramas de secuencia, aunque éstos fueron desechados por su poco uso, y el modelo Entidad Relación, del cual surgieron varias versiones en la medida que se incorporaban funcionalidades a la aplicación. Si bien no fueron muchos diagramas, si fueron muy útiles y se convirtieron en la columna vertebral del desarrollo. Todos estos diagramas fueron elaborados a mano y sin prestar mucha atención a la estética de los mismos tal y como lo platea XP. La única excepción fueron los diagramas Entidad Relación los cuales se construyeron en una herramienta UML con la cual se elaboraron mucho más fácil que a mano.

5.2 METÁFORA DEL SISTEMA

5.2.1 Lo que dice la metodología XP

- *Plasmar la arquitectura del sistema en una "historia"*
- *Convención de nombres para los objetos del sistema.*[14]

5.2.2 Experiencia en el Proyecto.

Debido que el programa es una aplicación sencilla y de fácil comprensión tanto para los desarrolladores como para el cliente, se creó una pequeña historia (metáfora) para entender cómo funciona el sistema al inicio, manteniendo todos los nombres en contexto con el negocio, y fue usada para que los desarrolladores entendieran el sistema, pero después su uso no fue muy frecuente ya que se entendía el sistema y además no hubo nuevos miembros al equipo de desarrollo.

Para ver con detalle los estándares adoptados, remítase al ANEXO A: ESTÁNDARES.

El poseer una metáfora que incluya una convención de nombres clara facilita enormemente el logro del objetivo de la propiedad colectiva del código, ya que con solo ver el nombre de un objeto o de un método se puede tomar una claridad bastante amplia sobre la función que este desempeña y el lugar que ocupa dentro del sistema.

[14] http://www.agile-process.org/model.html

5.3 TARJETAS CRC

5.3.1 Lo que dice la metodología XP

- *Su principal utilidad es dejar el enfoque procedimental y entrar al modelo orientado a objetos.*
- *Todo el grupo participa en su elaboración.*[15]

5.3.2 Experiencia en el Proyecto.

Una de las principales piezas de diseño empleada en el proyecto fueron las tarjetas CRC, que no sólo sirvieron como columna vertebral de este, sino que también, tal y como lo hace el diagrama de clases en las metodologías tradicionales, fueron la base del modelo Entidad Relación, elaborado para modelar la base de datos.

Cada Tarjeta CRC se convirtió en un objeto, sus responsabilidades en métodos públicos y sus colaboradores en llamados a otras clases, además se decidió hacer una modificación a lo que aconseja XP y fue tener los atributos en la misma sección de las responsabilidades para ayudar a encontrar con mayor facilidad a quien le correspondía tener una nueva responsabilidad.

En el proceso de elaboración de las tarjetas CRC los dos miembros del equipo estuvieron presentes manipulándolas, de modo tal, que tanto el diseño fue producto de la participación de los dos desarrolladores, como el resultado del mismo fue ampliamente asimilado por ambos, favoreciendo la propiedad colectiva del código.

En XP el proceso de diseño es iterativo, por lo cual las tarjetas CRC no fueron creadas todas en la primera iteración. Al inicio de cada iteración se les fueron agregando atributos, responsabilidades, llamados, o fueron creadas otras CRC nuevas, de modo tal que el diseño se convirtió en un proceso dinámico que se adaptaba a las necesidades planteadas para el momento. Sin embargo, su utilidad no fue la misma durante todo el proceso de desarrollo. En las primeras iteraciones fueron supremamente útiles, dando una idea clara de las responsabilidades sobre la lógica del negocio, pero en las últimas iteraciones donde ya se tenía claridad

[15] http://www.extremeprogramming.org/rules/crccards.html

sobre todos estos elementos, las tarjetas CRC fueron menos empleadas.

XP no propone una estrategia para afrontar la implementación de las tarjetas CRC, por lo cual se creó una con la cual se garantizó el poder correr las pruebas desde el mismo momento que inició la implementación. Primero fueron implementadas las clases más sencillas, aquellas que no hacían llamados a ninguna otra, para seguir con las que hacían llamados a las ya implementadas y así sucesivamente. Aunque XP no plantea una metodología para implementar un modelo de CRC, fue importante adoptar esta metodología debido a que era la forma más cómoda de poder aplicar las pruebas en todo momento. Cuando se empieza por codificar las clases que sólo hacen llamados a clases de JDK*, las pruebas pueden correrse desde el mismo momento que inicia el proceso de codificación, y al ir avanzando en el proceso de implementación en la forma ya indicada, las pruebas irán corriendo todo el tiempo de modo tal que se mantiene absoluto control sobre el desarrollo, en lugar de tener que esperar un largo tiempo antes de ejecutar las pruebas.

5.4 SPIKE SOLUTION (SOLUCIÓN RÁPIDA)

5.4.1 Lo que dice la metodología XP

- *Se trata de una prueba que se hace para resolver un problema técnico o de diseño.*
- *Es un programa muy simple que explora una solución potencial.*[16]

5.4.2 Experiencia en el Proyecto.

El uso de Spike Solution se convirtió en una práctica de vital en el desarrollo del proyecto, debido a la falta de experiencia en el lenguaje usado, lo cual generaba muchas dudas técnicas. Las de mayor importancia para el proyecto fueron dos situaciones las cuales, se le entregó el proyecto a un desarrollador diferente, de igual forma de realizaron todas las demás situaciones generadas durante el desarrollo de cada iteración.

Para implementar la comunicación con los dispositivos electrónicos externos, se debía crear una interfaz por el puerto serial, la cual debía ser configurada según

[16] http://www.extremeprogramming.org/rules/spike.html

las necesidades de cada dispositivo y además se debía filtrar los mensajes recibidos por el puerto serial. El estudio del completo uso del puerto serial, fue encargado a uno de los desarrolladores, el cual destinó aproximadamente doce horas a su estudio, al término de las cuales en un periodo no mayor a dos horas capacitó en el empleo del puerto al otro desarrollador. Por otro lado se requirió de un medio para realizar reportes impresos de calidad y de forma sencilla. La solución adoptaba fue ReportViewer, la cual fue estudiada en el transcurso de la tercera iteración por un desarrollador, al término de este tiempo se compartió la información al otro desarrollador tal como en el caso anterior, sin que retrasara sus demás responsabilidades con el proyecto.

XP recomienda asignar estos Spike Solution por parejas, sin embargo tal como se explicó en el capítulo de planeación no fue hecho de esta forma, lo que no representó un problema. El objetivo de lograr una comprensión rápida de cada uno de estos asuntos fue logrado, asegurando el cumplimiento de los plazos del proyecto.

Un aspecto importante relacionado con los Spike Solution, es que el conocimiento adquirido con la elaboración de estos debe ser compartido con el resto del grupo, debido a que es uno de los puntos más sensibles de convertirse en islas de conocimiento y posteriormente cuellos de botella, lo que definitivamente procura evitar XP. En este sentido se fue garantizando cuidadosamente que ambos miembros pudieran hacer implementaciones de ambas librerías si así lo necesitaban, esto no fue necesario en ambos casos. Si bien ambos desarrolladores participaron activamente en el empleo del puerto serial, no fue el mismo caso para ReportViewer que por comodidad solo fue empleada por quien la estudió con esporádicas intervenciones del otro desarrollador.

Uno de los aportes más importantes del concepto de Spike Solution, fue la posibilidad de desligarse completamente del proyecto mientras se trabaja en él, lo que facilita la concentración del esfuerzo personal en adquirir un conocimiento, en lugar de disiparlo intentando solucionar un problema sin saber cómo enfrentarlo.

5.5 NO ADICIONE FUNCIONALIDAD ANTES DE TIEMPO

5.5.1 Lo que dice la metodología XP

- *El adicionar funcionalidades que no se han acordado para la iteración conlleva una pérdida de tiempo que es inaceptable.*[17]

[17] http://www.extremeprogramming.org/rules/early.html

5.5.2 Experiencia en el Proyecto.

La idea de no adicionar funcionalidad antes de tiempo es uno de los conceptos más polémicos que tiene XP, por lo tanto posee lados positivos como negativos. En el desarrollo del proyecto no se fue muy estricto con el cumplimento de esta recomendación, lo que produjo beneficios e inconvenientes a la vez.

Al inicio del proyecto, esta idea se cumplía al pie de la letra y fue evidente el rendimiento en términos de funcionalidades implementas versus el tiempo. Este fue logrado gracias a la claridad que se tenía en términos de los objetivos que se debían cumplir durante cada una de las iteraciones, claridad obtenida en gran medida a la idea de no adicionar funcionalidad antes de tiempo.

Es importante resaltar que en muchas ocasiones durante la realización del proyecto uno o ambos desarrolladores se vieron tentados de crear alguna funcionalidad extra, que se presumía no consumiría mucho tiempo y posiblemente ofrecería una utilidad importante, intensión que fue "reprimida" al recordar la premisa de "no adicionar nada que el cliente no haya pedido explícitamente". Producto de este concepto no solo se logró optimizar al máximo el tiempo, sino también se vio reflejado en la experiencia que el cliente tuvo con la aplicación en cuanto tenía contacto con ella. Al no tener funcionalidades que no se había solicitado explícitamente, el programa estaba libre de botones, menús o demás funcionalidades que presentaran confusión, encontrándose solamente con elementos que resultaron familiares, lo que facilitó el proceso de aprendizaje de la herramienta. Este resultado positivo no es mencionado por XP, sin embargo es importante resaltarlo.

La idea de no adicionar funcionalidad antes de tiempo comenzó a generar pequeños inconvenientes. En las reuniones iniciales donde se diseñó el plan de entregas quedaron algunos aspectos claros sobre el proyecto, sin embargo, al inicio de cada iteración solo se discutió la parte del proyecto concerniente a dicha iteración, por cuanto cualquier detalle de diseño o implementación que concerniera a una iteración posterior fue omitido, aún si se tenía completamente claro que tarde o temprano debía ser considerado. Por ejemplo, en la primera iteración se construyó en el modelo Entidad Relación una tabla llamada historial de la cual debía contener una llave foránea de la tabla usuario, pero esto no fue implementado en esta iteración debido a que correspondía a otra iteración. El costo de incluirlas en la iteración que le correspondía fue considerablemente superior al requerido si se hubieran implementado de una vez en la anterior iteración, sobre todo en lo que tiene que ver con las pruebas. Es importante aclarar que cuando se hace una modificación en el diseño de la base de datos, también hay que modificar las pruebas y el código de la aplicación, por lo que su

impacto es el más alto de todos. En otros elementos como las interfaces, o la codificación misma, el impacto es menos importante.

Después de presentarse esta situación, el equipo de desarrolladores se reunió para hacer la modificación correspondiente a la metodología XP, en esta reunión se decidió que, al tener una visión de todo el sistema, cuando se presente la necesidad/oportunidad de realizar una funcionalidad que no está contemplada en la presente iteración, se realizarán los cambios mínimos para que en la futura iteración no se haga traumatizante desarrollarla. También, en dicha reunión se resalto el gran número de mejoras en cuando a usabilidad del aplicativo, para ellas se creó un mecanismo de recolección de dichas mejoras para ser comunicadas al cliente y no dejarlas en el olvido.

La idea de no implementar antes de tiempo es interesante y se convierte en una herramienta muy importante para optimizar el tiempo de desarrollo, sin embargo, no debe abusarse y administrarse con sentido crítico. Si se sabe que algo con toda seguridad va a requerir ser implementado, sobre todo si se trata de la base de datos, y se sabe también que el costo de implementarlo después será superior al requerido ahora, probablemente no sea tan mala idea considerar adelantarlo.

5.6 REFACTORIZACIÓN

5.6.1 Lo que dice la metodología XP

- *Diseño como tarea permanente*
- *Se conserva el código sencillo*
- *Rehacer secciones de código si es necesario*[18]

5.6.2 Experiencia en el Proyecto.

Al transcurrir el desarrollo de la aplicación, se revisó constantemente el diseño de la misma, surgiendo situaciones que no fueran tomadas en cuenta al comienzo del proyecto en el diseño general. Como salida a estos problemas se optó por la refactorización de las partes afectadas, buscando las soluciones más convenientes y sencillas, conservando la simplicidad del código. Aunque estos cambios fueron extensos, en ningún momento se convirtieron en cuellos de botella, gracias a que el código era de propiedad colectiva completamente.

[18] http://www.extremeprogramming.org/rules/refactor.html

Una de estas situaciones se refirió a la decisión que se tomó de no cambiar la forma en que se presentaba la información al usuario, que paso de ser una tabla dinámica a ser una más estática por preferencia del usuario. Las medidas tomadas para realizar este cambio no tomaron más de 2 horas de trabajo. En una metodología pesada, este tipo de situaciones podría generar costos extras, mientras en XP no tuvo la menor trascendencia, sólo demandó la implementación de una estrategia de solución en el menor tiempo posible.

Otra situación a la que se le aplicó refactoring, fue a la adición de una cuenta regresiva a la misma tabla en la cual se presenta la información de la llamada al usuario, esta característica fue adicionada casi al finalizar el proyecto. Dicha adición/modificación afectaba la base de datos en gran medida, se crearon nuevas tablas y nuevas referencias, las cuales afectaban un considerable número de funcionalidades dentro del aplicativo. Aunque este requerimiento fue tomado a última hora, los programadores realizaron el cambio en las clases afectadas y se adicionaron las tablas y las nuevas referencias, en un tiempo aproximado de 4 horas. Este refactoring fue logrado con éxito gracias a la estructura del programa que permitió la adición del código nuevo, y al plan de organización que se ideó antes de dicha modificación.

6. CODIFICACIÓN

En metodologías pesadas, la codificación es un proceso al cual solo se llega después de largas fases de análisis y diseño de las que queda una gran cantidad de documentación a partir de la cual el proceso de codificación es relativamente sencillo. En XP el proceso es muy diferente. Prácticamente desde un principio se inicia con la codificación, favoreciendo el logro del objetivo de estar haciendo entregas frecuentes al cliente.

Algunos de los elementos más importantes en cuanto a la codificación son que, el cliente debe estar presente en esta, se debe trabajar en parejas y debe haber una propiedad colectiva del código. Todos estos elementos representan paradigmas nuevos en lo que a la ingeniería del software se refiere, planteando entornos de discusión sobre la conveniencia de adoptarlas.

A continuación encontrará para cada uno de los aspectos que conforman la etapa de codificación una comparación entre las ideas teóricas de la codificación y lo aplicado en la ejecución del ejercicio práctico.

6.1 CLIENTE SIEMPRE PRESENTE

6.1.1 Lo que dice la metodología XP

- *El cliente debe estar disponible en el sitio de trabajo*
- *El cliente es fundamental para solucionar dudas cara a cara*[19]

6.1.2 Experiencia en el Proyecto.

La idea de tener al cliente, un representante de éste o a un usuario no es fácil de asimilar si se consideran los costos que esto representa. En el caso de este proyecto, el cliente no podía desplazarse a ninguno de los lugares de trabajo de los desarrolladores dado que debía estar al frente de su negocio. Pretender tener al otro usuario implicaría pagarle el tiempo que estuviera acompañando al grupo, gasto que ni el grupo de desarrollo podía asumir ni el cliente le interesaba pagar. Por tal motivo, se debió implementar una estrategia de comunicación distinta en la cual los programadores podían llamar vía telefónica al cliente en el momento que

[19] http://www.extremeprogramming.org/rules/customer.html

requirieran solucionar cualquier duda en el proceso de implementación. Cuando la duda era demasiado complicada para ser comprendida por vía telefónica, se acordaba una reunión para solucionar las dudas existentes. Si bien esta estrategia no fue igual de efectiva que haber tenido al usuario acompañando el desarrollo, fue suficiente para lograr una buena comunicación con él.

Es importante tener en cuenta que contar con un representante del cliente en las instalaciones del equipo de desarrollo, demandará una inversión representada en la remuneración económica para el representante. En muchos proyectos susceptibles de ser desarrollados por medio de XP, este gasto se vuelve inaceptable por elevar de forma importante el costo del proyecto. Sin embargo, se puede plantear una solución intermedia donde el representante del cliente solo esté presente durante un periodo de tiempo acordado con el grupo de desarrollo por día, que deberá ser aprovechado para disipar cualquier duda que haya surgido durante el resto del tiempo de desarrollo en que no estuvo. Esta es una modificación importante a XP, ya que el "Cliente In Situ" es uno de los elementos más relevantes de esta metodología y la hace mucho más aplicable en la práctica.

6.2 EL CÓDIGO SE ESCRIBE SIGUIENDO ESTÁNDARES

6.2.1 Lo que dice la metodología XP

- *Al escribir el código del programa se deben seguir estándares de programación.*[20]

6.2.2 Experiencia en el Proyecto.

La estandarización del código fue asumida desde el mismo momento en que se inició la codificación. Debido que el grupo de desarrollo había estado trabajando unido por largo tiempo, ya tenían un esquema de estándares acordados de forma táctica, sin embargo, por seguir una disciplina se formalizaron estos en un documento.

Se debe resaltar que el programar empleando estándares es una práctica que no solo se recomienda en XP, es una buena práctica que debe ser seguida en cualquier metodología de desarrollo lo que no implica algo muy novedoso en XP.

[20] http://www.extremeprogramming.org/rules/standards.html

Tal como se ha presentado en varias oportunidades en el transcurso de este documento, el programar siguiendo estándares no es un fin en sí mismo. Se trata de un medio con el cual se pretende facilitar la propiedad colectiva del código.

En el caso de este proyecto se aplicaron todos los estándares con gran éxito debido a dos motivos. En primer lugar, la herramienta empleada para desarrollar facilitaba aplicar los estándares pactados y en segundo, que los mismos venían siendo empleados desde antes por el equipo de desarrollo.

6.3 CODIFICAR PRIMERO LA PRUEBA

6.3.1 Lo que dice la metodología XP

- *Escribir primero la prueba que el código.*
- *El tiempo de escribir una prueba y luego el código del programa para dicha prueba es menor que solo escribir el código.*
- *Escribir pruebas primero permite identificar los casos especiales que deberá pasar el código.*[21]

6.3.2 Experiencia en el Proyecto.

No es fácil seguir este planteamiento de XP por varios motivos. En primero lugar, el framework escogido para el desarrollo no provee una facilidad para realizar las pruebas, ni tampoco se encontró una herramienta, con licencia de libre uso, para este fin.

Se realizaban códigos con pequeñas pruebas para el funcionamiento de las funciones, pero se encontraron muchas limitaciones, por ejemplo al tratar de hacer pruebas a elementos gráficos tales como Ventanas y Botones, o realizar pruebas a partes del código que se comunican con dispositivos externos por medio del puerto serial. Los reportes son otro obstáculo para realizar las pruebas, ¿Cómo probar un reporte que fue diseñado en papel? o ¿Cómo saber si el programa obtuvo la hora del sistema de forma correcta?, si no es comparándola con la propia hora del sistema.

El carácter privado de muchos métodos representa un obstáculo insalvable para hacerle pruebas, ya que como solo puede ser accedido desde el interior de la

[21] http://www.extremeprogramming.org/rules/testfirst.html

clase no puede ser probado independientemente. La solución que se tomó en este sentido fue probarlo a través del método público que hace uso de él y permite generar una traza del comportamiento. Si el método público pasa determinada prueba, se asume que el método privado también la ha pasado.

Todos los elementos anteriores representaron obstáculos en el desarrollo de las pruebas y plantearon una inquietud importante sobre el alcance del concepto "codificar primero las pruebas". ¿Se trata de codificar SIEMPRE una prueba antes que el código? o ¿solo aquellas clases encargadas de realizar la lógica del negocio? Debido que XP no tiene una respuesta clara a esta inquietud, el grupo de desarrollo optó por probar solo aquellas clases que ejecutan la lógica del negocio, que en definitiva son las más importantes de las cuales se debe tener garantía de estar bien construidas.

Es importante resaltar las ventajas que representa hacer las pruebas antes que el código de la aplicación. En primer lugar, el tiempo que toma escribir determinado código después de haber implementado la prueba es considerablemente menor que si no hubiera escrito la prueba antes. En segundo lugar, al hacer la prueba se identifican de manera precisa cuales son los casos especiales y rutas alternas que deben ser consideradas dentro del código haciendo de este un producto más robusto y tolerante a fallos. Finalmente, una ventaja no expuesta de forma vehemente por XP es la estética del código de la aplicación. Al tener tal claridad sobre cómo debe ser escrito determinado código, la tarea de realizarlo es más sencilla y el mismo queda organizado de forma más estética.

6.4 TODA LA PRODUCCIÓN DE CÓDIGO DEBE SER HECHA EN PAREJAS

6.4.1 Lo que dice la metodología XP

- *Toda la producción de código debe ser hecha en parejas sentadas frente a un único computador.*
- *Al trabajar en parejas se tiene un diseño de mejor calidad y un código más organizado.*
- *Al trabajar en parejas se solucionan los problemas más fácilmente.*[22]

[22] http://www.extremeprogramming.org/rules/pair.html

6.4.2 Experiencia en el Proyecto.

El no contar con una sede permanente complicó seriamente el cumplimiento del objetivo de programar en parejas. Por otro lado, al solo haber una pareja de programadores se hacía completamente imposible cumplir con el objetivo de tener varias parejas de programadores trabajando uniformemente. Sin embargo, se procuró trabajar en pareja tal como lo plantea XP en un solo computador, aunque en algunas iteraciones se necesitó trabajar cada uno en su propio computador, con la salvedad de mantener el mayor nivel de comunicación posible, para avanzar con mayor rendimiento en el proyecto.

Para las iteraciones en las cuales se trabajo por separado, fue extremadamente importante el uso de los estándares, el cual generó que no importara que los dos programadores trabajaran por aparte, pues el código sería siendo legible para los dos.

Es importante resaltar que el concepto de programación en parejas ni es la gran panacea, ni debe ser descartado de plano. Es muy probable que dos programadores que no dominan la herramienta en la cual estén desarrollando, sean mucho más productivos trabajando bajo un mismo computador que estando solos. También es de resaltar la empatía que se requiere en la pareja para que el proceso dé resultados exitosos. Se requiere que ambos programadores tengan concepciones similares en términos de cómo enfrentar un problema de programación para que sean productivos.

6.5 SOLO UNA PAREJA HACE INTEGRACIÓN A LA VEZ (INTEGRACIÓN SECUENCIAL)

6.5.1 Lo que dice la metodología XP

- *Antes de integrar nuevo código a un proyecto se debe garantizar que la última versión halla pasado todas las pruebas.*
- *Solo se debe hacer una integración a la vez.*
- *Se debe tener claro cuál es la última versión funcional.*[23]

[23] http://www.extremeprogramming.org/rules/sequential.html

6.5.2 Experiencia en el Proyecto.

A diferencia de otras metodologías donde se definía propiedad sobre algunas clases, en esta ocasión se le dio la propiedad de todo el proyecto a dos programadores, alternando ésta posesión durante todo el proceso de desarrollo según como la necesidad de implementación lo iba requiriendo.

Durante todo el tiempo se tenía completa claridad de quién tenía la última versión, de modo que la otra persona pasaba permanentemente el código a la poseedora de la mencionada última versión, haciéndola responsable de hacer la integración y garantizar que la nueva última versión no tuviera errores. Esto no va en contra de la propiedad colectiva del código debido a que cualquier desarrollador podía modificar cualquier clase, siempre y cuando coordinara esto con el dueño del proyecto para tenerla en cuenta en la siguiente integración.

Esta metodología resultó no solo efectiva sino también flexible. Se garantizó que solo una persona hiciera integración a la vez, que siempre se supiera no solo cuál era sino también donde estaba ubicada ésta y que se pudiera traspasar la responsabilidad de integrar según como fuera necesario.

Se siguió la directiva de XP, en el sentido de no dar posesión de clases o elementos de código a nadie, que solo se hiciera una integración a la vez y siempre que se tuviera claro cuál era la última versión. Al cumplir a cabalidad con estas tres instrucciones se evitaron muchos problemas haciendo que el proceso de integración fuera fluido y sin inconvenientes.

6.6 INTEGRACIONES FRECUENTES

6.6.1 Lo que dice la metodología XP

- *Se deben hacer integraciones cada pocas horas o en lo posible no tardar más de un día entre una y otra integración.*
- *Entre más se tarde en encontrar un problema, resultará más costoso resolverlo.*
- *Integrar frecuentemente evita problemas como el trabajar sobre una clase obsoleta.*[24]

[24] http://www.extremeprogramming.org/rules/integrateoften.html

6.6.2 Experiencia en el Proyecto.

Se siguieron los lineamientos de XP, durante las iteraciones donde la pareja trabajó por separado, se hacían una o dos integraciones diarias, y en las otras iteraciones, simplemente se trabajaba siempre sobre la última versión, garantizando de esta forma que en todo momento se estuviera trabajando sobre la última versión del proyecto.

Al no emplear software de versionado por no contar con un servidor conectado a Internet permanentemente se recurrió a un estándar para conocer cuál era la última versión. Todos componentes del proyecto se almacenan dentro de una carpeta con la siguiente estructura de nombres: Material PrimerTax S.A versión X.Y.Z. De esta forma se garantizaba que cualquier desarrollador en forma autónoma encontrara la última versión del proyecto.

La persona encargada de la integración no era siempre quién había construido la clase que iba a ser añadida o reemplazada. Esta responsabilidad era de quién en el momento fuera el dueño del proyecto tal y como se planteó en el apartado de integración secuencial. Las tareas que tenía dicha persona eran la de integrar el código y las pruebas, realizar y supervisar dichas pruebas y garantizar la funcionalidad del programa antes de liberarlo. Al momento de realizar una integración se le enviaba el resultado de esta al otro programador para que de ese momento en adelante trabajara sobre ella.

Al realizar en forma cuidadosa este procedimiento, permanentemente se evitaron problemas con versiones obsoletas de elementos del sistema y se permitió encontrar problemas, que de no haberse hecho integraciones frecuentes habían sido mucho más costosos.

Al tener componentes que no se podían probar por la limitante de dependencia de otros dispositivos externos para su funcionamiento, se tenía especial cuidado con dichos componentes cuando eran modificados para tratar de evitar errores de lógica y se tenían pendientes sus pruebas para cuando se estuviera en la empresa probando la versión liberada. Existieron casos en los que dichos componentes fallaban, pero en su gran mayoría era por errores de configuración del dispositivo electrónico externo con el cual se hacia la comunicación.

6.7 PROPIEDAD COLECTIVA DEL CÓDIGO

6.7.1 Lo que dice la metodología XP

- *Se debe procurar rotar a los programadores no solo de compañero, también de partes del proyecto a desarrollar.*
- *Cualquier programador debería poder continuar la codificación que alguien más empezó sin muchas dificultades.*[25]

6.7.2 Experiencia en el Proyecto.

Sobre la propiedad colectiva del código se ha discutido bastante durante el documento, pero solo en este momento se afronta el tema de forma directa. XP propone muchas estrategias destinadas a facilitar la propiedad colectiva del código, debido a la dificultad de lograrse en proyectos pequeños, y se complica aún más en la medida que el proyecto crece.

Estrategias como la rotación del personal, el empleo de estándares y la programación en parejas van destinadas a la consecución de la propiedad colectiva del código, de modo tal que solo se logrará este objetivo en la medida que las estrategias planteadas sean ejecutadas cuidadosamente.

Analizando el proyecto que se presenta en este documento, el rotar a los programadores por diferentes partes de la aplicación no fue fácil de lograr, principalmente porque no se pudo desarrollar la estrategia de la programación en parejas, sin embargo, gracias a la aplicación de las estrategias ya discutidas anteriormente fue posible rotar a los programadores en diferentes partes del proyecto según como las necesidades de este así lo requirieran.

En la medida que transcurrían las iteraciones se requería dedicar más horas de trabajo en determinadas partes de la aplicación, pudiéndose enfocar el trabajo de alguno de los programadores en partes del programa que habitualmente no trabajaba. Esta flexibilidad en la distribución de tareas fue fundamental para evitar la sobrecarga de trabajo en uno de los programadores, agilizando aún más el desarrollo. De no ser porque había obtenido muy buenos resultados en la propiedad colectiva del código, no habría sido posible esto. No obstante, sería demasiado optimista pretender lograr la propiedad colectiva del código al cien por ciento, además de un desgaste excesivo para el equipo de desarrollo.

Se prefería que quién había creado una clase continuara trabajando en su desarrollo, se temía que otro desarrollador no tuviera el mismo rendimiento que el creador. El motivo es muy simple, por más que se sigan estándares y se haga un diseño muy cuidadoso, la labor de programar es creativa y cada programador enfrenta un mismo problema de diferentes formas para facilitarse el trabajo, lo que probablemente podría confundir a otro.

[25] http://www.extremeprogramming.org/rules/collective.html

Una disciplina que se adoptó para lograr la propiedad colectiva del código fue notificar al otro desarrollador cuando se había modificado una clase e intercambiar dichos cambios. La importancia de esta medida era evitar que un desarrollador creyera conocer un código cuándo la última versión de este difería de la que conocía.

Probablemente uno de los grandes aportes de XP a la Ingeniería del Software es la propiedad colectiva del código, pero no como concepto, el cual es deseable desde el inicio de programación. El aporte de XP radica en los métodos que se deben emplear para alcanzarlo con algún éxito. Pese a esto, es importante aclarar que solo es posible lograr la propiedad colectiva del código en proyectos pequeños. En la medida que estos crecen, el intentar alcanzarla deja de ser productivo para el proyecto y empieza a convertirse en lastre para el equipo de desarrollo.

6.8 NO TRABAJAR HORAS EXTRAS

6.8.1 Lo que dice la metodología XP

- *No trabajar horas extras.*
- *El dedicar horas extras a un proyecto retrasado, no lo va a poner al día.*
- *Es preferible replantear los plazos a trabajar horas extras.*[26]

6.8.2 Experiencia en el Proyecto.

Debido a que los desarrolladores no se dedicaron exclusivamente al proyecto, el concepto de horas extras se vuelve más subjetivo. Desde un contexto laboral, horas extras se entiende como el tiempo trabajado después de ocho horas diarias, concepto que no es aplicable al proyecto en cuestión.

Para cualquier persona que haya programado es claro que se trata de una tarea de importante esfuerzo mental y como tal requiere no ser desempeñada por muchas horas consecutivas, en tal sentido, el plantearse trabajar horas extras después de una jornada completa de desarrollo sugiere más una pérdida de tiempo que una recuperación de los atrasos del proyecto.

En el caso de este proyecto no se trabajaron horas extras debido a que se tuvo la

[26] http://programacionextrema.tripod.com/fases.htm#terceraFase

intención de seguir los lineamientos de XP. Pero uno de los grandes problemas que se tuvo durante el proyecto fue el definir los plazos para las entregas. Esto se debe a la falta de experiencia que tenían los desarrolladores en planear un proyecto.

Debido a lo anterior se la estrategia que se tomo fue la de replantear los plazos en vez de trabajar horas extras, los cuales fueron hablados con el cliente, con su debida explicación. En este punto el cliente fue bastante flexible, gracias a que tenía experiencia en el tema de la iteración con proyectos de software y sus conocidas demoras, aun usando metodologías tradicionales. Además el cliente estaba satisfecho con las entregas realizadas, las cuales cumplían con lo pactado.

Uno de los elementos que considera XP para hacer la mayoría de sus planteamientos, es el hecho que la labor de programación es muy difícil de estimar y proyectar en el tiempo. La experiencia tanto en este proyecto como en otros anteriores es que el rendimiento en la labor de programación varía mucho según un sin número de circunstancias que afectan no sólo al proyecto en sí mismo sino también a quién lo esté implementando. En tal sentido, se debería poder tanto flexibilizar en lo posible los plazos como diseñarlos con suficiente holgura para contemplar estas posibles demoras. Si bien XP plantea que debería ser posible replantear algunos plazos, esta posibilidad debería quedar clara desde el principio con el cliente para evitar malos entendidos.

7. PRUEBAS

XP recomienda la realización del mayor número de pruebas posibles a lo largo del desarrollo del proyecto, con el fin de asegurar que los resultados se mantengan a medida que avanza el proyecto. En este proceso no solo participa el equipo de desarrollo, sino que además, el cliente es importante y especialmente en las pruebas de aceptación.

Las pruebas deben ser realizadas con el fin de garantizar el funcionamiento correcto de las historias de usuario construidas. No importa el mecanismo por el cual se realicen o si se utiliza algún módulo específico para ejecutarlas.[27]

7.1 EL USO DE LOS TESTER EN XP:

- Se deben crear las aplicaciones que realizarán los test con un entorno de desarrollo específico para test.
- Hay que someter a test las distintas clases del sistema omitiendo los métodos más triviales.
- Se deben crear los test que pasarán los códigos antes de implementarlos. En el apartado anterior se explicó la importancia de crear antes los test que el código.

Los test permiten verificar que un cambio en la estructura de un código no tiene porqué cambiar su funcionamiento.

7.2 PRUEBAS DE UNIDAD

7.2.1 Lo que dice la metodología XP

- *Las pruebas deben ser escritas antes que los métodos.*
- *Su implementación y ejecución deben consumir el menor tiempo posible.*[28]

7.2.2 Experiencia en el Proyecto.

La creación de pruebas para el sistema desarrollado en el proyecto fue difícil de

[27] http://programacionextrema.tripod.com/fases.htm#cuartaFase
[28] http://www.extremeprogramming.org/rules/unittests.html

soportar a través de programas de prueba debido a la dependencia con el hardware especializado.

Se realizaron pruebas de caja blanca, mediante seguimiento de trazas en la ejecución manual de varios casos de prueba. Se permitió mediante este mecanismo detectar problemas, casos de falla y de éxito que podrían surgir durante las pruebas de aceptación o durante la puesta en marcha del proyecto.

7.3 PRUEBAS DE ACEPTACIÓN

7.3.1 Lo que dice la metodología XP.

Test de aceptación sirven para evaluar las distintas tareas en las que ha sido dividida una historia de usuario. Para asegurar el funcionamiento final de una determinada historia de usuario se deben crear "Test de aceptación". Estos test son creados y usados por los clientes para comprobar que las distintas historias de usuario cumplen su cometido. [29]

7.3.2 Experiencia en el Proyecto.

La realización de las matrices de prueba, se realizaron conforme a los diseños planteados durante la planeación de las historias de usuario.

En las matrices y plantillas de pruebas de aceptación se registraron los resultados esperados, los resultados esperados fueron especificados para cada historia de usuario.

No fueron realizados programas de prueba para las funcionalidades que se comunicaban con sistemas externos (hardware), estas pruebas por lo tanto se realizaron en los equipos del cliente en horarios de poca concurrencia para no afectar el rendimiento operativo de la empresa.

7.4 MANEJO DE ERRORES

[29] http://www.extremeprogramming.org/rules/functionaltests.html

7.4.1 Lo que dice la metodología XP.

Al momento de encontrarse un error durante las pruebas de unidad o las pruebas de aceptación, se debe documentar con detalle le caso de prueba y el resultado obtenido erróneamente.[30]

7.4.2 Experiencia en el Proyecto.

Cuando se detectaba un resultado incorrectamente inesperado durante la ejecución de pruebas de caja blanca o caja negra, el analista de pruebas documentaba con cuidado las condiciones en las cuales se generó el error; los datos de entrada y la salida generada. En caso de que fuera posible de realizaba una copia de una traza que permitiera hacer el diagnóstico aproximado del error.

[30] http://www.extremeprogramming.org/rules/bugs.html

8. COMENTARIOS

En este capítulo se realizan comentarios acerca de la experiencia vivida luego de aplicar la metodología XP al desarrollo de una solución de software.
Algunos de estos comentarios agrupan observaciones, sugerencias y críticas que se adquirieron durante el desarrollo del proyecto. Se resaltan de manera especial los cambios realizados a la metodología para adaptarla al tamaño del proyecto, al número de personas que integraron el grupo de desarrollo y al tiempo que se disponía por parte del grupo de desarrollo para trabajar en el proyecto.

Para facilitar la comprensión de los comentarios, se han agrupado según las fases de la metodología XP.

8.1 PLANEACIÓN

Los comentarios expuestos a continuación tienen que ver con la fase de planeación en donde se sugiere la construcción de historias de usuario, estimación de tiempos, historias de usuario realizadas por iteración y la estimación de la velocidad del proyecto.

8.1.1 Historias de Usuario.

Las historias de usuario no fueron escritas directamente por el cliente, se tomó la decisión de que las historias fueran escritas a medida que se realizaban las reuniones con el cliente, pero se realizaba por alguno de los miembros del grupo de desarrollo. Esto se hizo con el fin de que el cliente solo se preocupara de expresar las ideas libremente. Al finalizar cada reunión el cliente realizaba una lectura de la historia de usuario creada, hacía los ajustes o aclaraciones que creía convenientes y luego el equipo de desarrollo se dedicaba a realizar las historias de usuario que se necesitaran para dividir la historia en tareas específicas que serían las asignadas a un responsable del grupo de desarrollo.

8.1.2 Número de horas de trabajo por semana no laborar horas extras.

Para el caso de este proyecto, la estimación de tiempo para desarrollar una historia de usuario dada en semanas era una estimación de poca utilidad, a causa de que existían semanas en las cuales no era posible cumplir el número de horas semanales; como consecuencia de esto la semana siguiente el grupo de desarrollo debía compensar el tiempo trabajando más de lo planeado el fin de

semana y en las noches.

Finalmente se estimaban con mayor exactitud el tiempo de desarrollo en horas planeadas para cada historia de usuario y el control interno de cumplimiento se realizaba con base en las horas trabajadas contra el avance en la historia de usuario, de manera que si se presentaba algún atraso en el avance del proceso se detectara con prontitud para evaluar las posibles causas, corregirlas, recuperar el tiempo perdido e informar al cliente del evento si es necesario.

8.1.3 Velocidad del proyecto.

La velocidad del proyecto no fue una medida de importancia ni para el equipo de desarrollo ni para el cliente. La razón principal por la que la medida fue inútil fue por el gran número de cambios que se presentaron a lo largo de la evolución del proyecto, esto fue porque a medida que se realizaban entregas el cliente y los usuarios finales iban adicionando nuevos requerimientos que variaban el porcentaje de avance en el proyecto calculado con base en el número de Historias de Usuario implementadas en un determinado tiempo (Véase Tabla 2. Iteraciones y velocidad del proyecto).

8.2 DISEÑO

En este apartado se discuten puntos de vista opuestos sobre el uso de diseños simples y la adición de funcionalidades antes de tiempo, identificando los casos en los cuales fue conveniente y aquellos en los que no lo fue.

8.2.1 Diseños simples para facilitar la capacitación.

Al aplicar los lineamientos de simplicidad en el diseño y de no adicionar funcionalidad antes de tiempo, se evitaron incorporar elementos no solicitados por el cliente, lo cual genero una interfaz de usuario libre botones, etiquetas y otros elementos que se hubieran podido convertir en distractores para el usuario, facilitando el proceso de aprendizaje del mismo, trayendo como beneficio un interés positivo por el proyecto y un apoyo mayor por parte del cliente.

XP afirma que no se debe adicionar funcionalidad antes de tiempo, pero no manifiesta lo anterior como un motivo para hacerlo, siendo una razón importante para justificar esta práctica.

8.2.2 Agregar funcionalidad antes de tiempo.

El uso estricto de esta práctica no conllevo a tener algunos problemas. Sin importar qué metodología de desarrollo se esté empleando, la base de datos es un elemento esencial y por consiguiente su modelo. Cualquier cambio que se realice en éste durante el proyecto tiene un impacto superior que si fuera solamente en el código. Al alterar un modelo de base de datos, se tendrá que modificar tanto la propia base de datos como los métodos relacionas a esos en la aplicación, lo que conlleva mayor consumo de tiempo.

Por la experiencia que los el equipo ha tenido, pueden existir algunos elementos cuya presencia en el proyecto se presenta como obvia y por tal motivo deberán ser considerados en el modelo de datos en algún momento. Cabe aclarar que con lo mencionado en el párrafo anterior, el impacto de incluirlos ya avanzando el proyecto es alto. Por tal motivo se hace conveniente su adición al modelo de datos desde un principio, aunque no sean utilizados en el código hasta fases avanzadas del proyecto. Si bien, este juicio sólo se presenta para el modelo de datos, pueden existir algunos otros elementos ajenos a éste que requieran la misma consideración.

Esta conclusión se contradice directamente con lo manifestado por XP, la cual sugiere no adicionar funcionalidad antes de tiempo, y solo implementar estrictamente aquello planeado para determinada iteración, con lo cual se sugiere aceptar este planteamiento de XP, con algunas excepciones consideradas por el equipo en las primeras fases de diseño.

8.3 CODIFICACIÓN

La metodología XP plantea un esquema de trabajo muy diferente al usado en las metodologías tradicionales presentando puntos positivos y negativos. A continuación se discuten algunos puntos resaltando la forma como se experimentaron en la ejecución del proyecto.

8.3.1 Costos del Cliente In Situ.

XP resalta la importancia de la presencia del cliente durante el proyecto, aunque esa presencia del cliente o de un representante de él representa unos costos económicos complejos y que vulneran en gran parte su viabilidad. Estos costos deberán ser asumidos por el cliente o por el equipo de trabajo, lo cual es inaceptable por las dos partes, lo que en últimas es un costo adicional al proyecto que se considera elevado.

Estos costos tienen un impacto importante para el proyecto, principalmente porque el tipo de proyectos para los que está recomendado XP son los pequeños y medianos donde el dinero se convierte en una gran limitante.

8.3.2 Programación en parejas.

El concepto de programación en parejas es más complejo que lo planteado por XP. Antes de tomar la decisión de implementarla o rechazarla de plano hay que analizar a profundidad el nivel de conocimientos que tengan en la herramienta, compatibilidad personal y laboral y enfoques de programación que empleen. Además se debe definir una metodología clara de cómo enfrentar los problemas de programación de programación. De lo contrario, más que una ayuda, sería un obstáculo el uno para el otro.

Cabe recordar que XP si plantea una metodología para desarrollar la programación en parejas, pero ésta no es lo suficientemente completa para considerar todos los aspectos, tanto técnicos como humanos, que implica a dos personas que trabajan en un mismo monitor y teclado.

Por ello se debe crear un plan de acción dentro del equipo de trabajo para subsanar las falencias que tiene la metodología ante este concepto. De esta forma se garantiza la continuidad del proyecto, sin anomalías que van de la mano con el trabajo entre dos humanos.

8.3.3 Trabajar horas extras.

El proceso de programación requiere de un alto esfuerzo mental, por lo cual el desgaste que produce una larga jornada de trabajo disminuye el rendimiento de los desarrolladores al punto en que los avances logrados en horas extras se consideran irrelevantes en comparación con el tiempo que toma lograrlos.

Comúnmente cuando los proyectos se retrasan, se comete el error de alargar las jornadas de trabajo con el fin de recuperar tiempo, pero al contrario en lugar de lograr el objetivo, se aumenta el nivel de desgaste y estrés del equipo de trabajo, convirtiéndose en una bola de nieve que en vez de solucionar el problema, lo empeora.

En el mundo laboral definitivamente no es recomendable replantear los plazos de entrega con el cliente cuando se presenta una situación de retraso, sin embargo para este tipo de proyectos puede convertirse en la opción más adecuada. Para evitar el riesgo de retrasos en el proyecto es mejor planear cronogramas más flexibles que consideren los posibles inconvenientes que se presenten en el

transcurso del proyecto.

8.4 PRUEBAS

8.4.1 Pruebas Autónomas.

XP sugiere que el desarrollo de las pruebas se realicen sin la intervención humana y que el mismo sistema mediante un análisis de resultados determine si los resultados fueron o no exitosos. Las condiciones dadas por el tipo de proyecto y la dependencia de sistemas externos (hardware) especializado no permitía realizar las pruebas con sistemas autónomos. Como mecanismo alternativo, las pruebas fueron realizadas mediante la compilación del código y el ingreso de parámetros para crear los casos de prueba y lograr respuestas del sistema para analizarlas y aplicar las correcciones necesarias a la funcionalidad.

Las pruebas realizadas al GUI de la aplicación fueron realizadas mediante revisión de pares. La revisión de pares fue uno de los mecanismos aplicados por el grupo de desarrollo, la metodología no especifica este tipo de herramientas, sin embargo fue de gran utilidad para el proceso de pruebas.

8.4.2 Pruebas antes de la construcción.

La planeación del esquema de pruebas ayuda a establecer con mayor detalle el funcionamiento del sistema, esto ayuda a que el desarrollador encargado tenga en cuenta mayor cantidad de situaciones a controlar en el sistema, evitando así el reproceso por errores de aplicación de lógica en el sistema.

Para concluir, el proceso de implementación de las historias de usuario, es más organizado, preciso y controlado si se planean los casos de prueba previos a la construcción.

9. CONCLUSIONES

El desarrollo e implementación del software desarrollado fue exitosamente concretado, y adicionalmente permitió adquirir conocimientos del negocio que permitirán en un futuro desarrollar herramientas más competitivas y prácticas que complementen la actual y proliferen los resultados obtenidos actualmente.

A continuación se describen los aspectos importantes que tuvieron lugar en el transcurso del desarrollo del proyecto.

7.1 MOTIVACIÓN E INTERÉS DEL CLIENTE

El cliente mostró un gran interés por el desarrollo ininterrumpido del proyecto, antes de dar inicio al desarrollo del proyecto, el cliente manifestó la necesidad de poder tener control o información del avance del proyecto; esto fue posible gracias a las entregas realizadas al finalizar cada iteración, el cliente se encontraba muy satisfecho gracias a que podía tener la posibilidad de apreciar el avance y manifestar libremente los cambios que deseara en el sistema.

Al momento de plantear un cambio el cliente estuvo dispuesto a esperar el análisis del mismo, por lo que recibía con aceptación aquellos resultados negativos, en los que el cambio era considerado de alto riesgo para el proyecto y no se implementaba en el alcance actual del proyecto.

9.2 CÓDIGO DE PROPIEDAD COLECTIVA

La distribución del conocimiento del código es una ventaja significativa, ya que existen funcionalidades en las que se puede recurrir a lluvias de ideas y el entendimiento mutuo del código facilitó que las ideas plasmadas en lógica de programación sean correctamente contextualizadas y entendidas por todos los integrantes del equipo de desarrollo.

9.3 CONTROL DE CAMBIO Y MANEJO DE RIESGOS

La metodología XP permitió que el cliente y los usuarios finales afinaran el proyecto conforme iba avanzando el desarrollo, a medida que surgían nuevos requerimientos los riesgos asociados al proyecto aumentaban, sin embargo, se

estableció con un cliente un mecanismo de análisis de riesgos que permitían establecer si un requerimiento se encontraba o no dentro del alcance del proyecto. Sumado a esto, cada cambio solicitado y que se convenía implementar era transformado en una historia de usuario, esto conllevaba a la planeación de un tiempo asociado a esta implementación y una prioridad.

9.5 SPIKE SOLUTION COMO PRÁCTICA UNIVERSAL

El concepto de Spike Solution se convierte en una práctica que puede ser usada en cualquier metodología para la solución de problemas técnicos, ya que es la oportunidad que brinda a un desarrollador de solucionar un problema o adquirir un nuevo conocimiento abstrayéndose totalmente cualquier particularidad del proyecto. El hecho de trabajar sobre un tema en particular permite llegar a un grado de profundización en el tiempo que no se logra otra forma, dándole al programador la habilidad de inyectar dichos conocimientos al proyecto.

10. RECOMENDACIONES

Deben ser planteados desde un inicio el control de los cambios, la detección de los riesgos y la forma en cómo serán valorados los nuevos requerimientos, esto con el fin de evitar que se pierda el alcance del proyecto.

Se debe definir la participación del cliente dentro del proceso, en caso de que el cliente no tenga la suficiente disponibilidad, entonces se puede plantear la posibilidad de tener la asistencia de otra u otras personas autorizadas y seleccionadas por el cliente para dicha participación.

Es importante aclarar al cliente la forma de trabajo de la metodología, de manera que el cliente conozca el mecanismo de trabajo, los horarios y la distribución de las tareas.

El cliente debe ser informado de cualquier novedad presentada en cualquiera de las fases de desarrollo.

11. BIBLIOGRAFÍA

BECK, Kent. Extreme Programming Explained. 2 ed. Addison-Wesley Professional; US ed edition, October 5, 1999. ISBN-10: 0201616416.

NEWKIRK, James y MARTIN, Robert. La programación extrema en la práctica. Traducido por Francisco Javier Zapata Molida 1 ed. Pearson Education, Madrid, 2002. ISBN: 8478290575

ECHEVERRY TOBÓN, Luis Miguel y DELGADO CARMONA, Luz Elena. Caso práctico de la metodología ágil XP al desarrollo de software. Trabajo de grado Ingeniero de Sistemas y Computación. Pereira: Universidad Tecnológica de Pereira. Facultad de Ingenierías Eléctrica, Electrónica, Física y Ciencias de la Computación. Programa Ingeniería de Sistemas y Computación, 2007. p. 28-43

SÁNCHEZ GONZÁLEZ, Carlos. ONess: un proyecto open source para el negocio textil mayorista desarrollado con tecnologías open source innovadoras. Ingeniero Informática. Coruña: Universidad de Coruña. Facultad de Informática - Departamento de Tecnologías de Información y las Comunicaciones. 2004. Disponible desde Internet: <http://oness.sourceforge.net/proyecto/html/index.html>

CASTILLO, Oswaldo; FIGUEROA Daniel y SEVILLA Hector. Fases de la Programación Extrema. Programación Extrema [en línea]. <http://programacionextrema.tripod.com/index.htm> [citado en 10 de Febrero de 2009]

GROSJEAN, Jean Claude. Use cases - User Stories: so precious but not the same!. Agile UX [en línea]. <http://www.agile-ux.com/2009/01/23/use-cases-user-stories-so-precious-but-not-the-same/> [citado en 15 de Noviembre de 2010]

LEYVA, Juan. Extreme Programming. Planeta Código [en línea]. <http://planetacodigo.com/wiki/glosario:extreme_programming> [citado en 21 de Octubre de 2010]

WELL, Don, Extreme Programming: A gentle introduction. Extreme Programming – Agile Process [en línea]. <http://www.extremeprogramming.org/> [citado en 28 de Septiembre de 2009]

Jimmy Wales. Wikcionario, La edición en castellano de Wiktionary [en línea]. <http://es.wiktionary.org/wiki/an%C3%A1lisis> [citado en 15 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea].

<http://es.wikipedia.org/wiki/Aplicaci%C3%B3n_%28computaci%C3%B3n%29> [citado en 31 de Octubre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Base_de_datos> [citado en 19 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Clase_%28inform%C3%A1tica%29> [citado en 10 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Dise%C3%B1o> [citado en 27 de Octubre de 2010]

Jimmy Wales. Wikcionario, La edición en castellano de Wiktionary [en línea]. <http://es.wiktionary.org/wiki/evento> [citado en 9 de Octubre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/GUI> [citado en 12 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Licencia_de_software> [citado en 17 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/M%C3%A9todo_%28inform%C3%A1tica%29> [citado en 12 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Cliente-servidor> [citado en 12 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Sistemas_gestores_de_bases_de_datos> [citado en 12 de Octubre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Objeto_%28programaci%C3%B3n%29> [citado en 8 de Octubre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos> [citado en 17 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Prototipo> [citado en 24 de Septiembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Sistema_operativo> [citado en 11 de Noviembre de 2010]

ALEGSA. Diccionario Informático / Tecnológico [en línea]. <http://www.alegsa.com.ar/Dic/subsistema.php> [citado en 21 de Septiembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Ventana_%28inform%C3%A1tica%29> [citado en 1 de Noviembre de 2010]

Jimmy Wales. Wikipedia, La enciclopedia libre [en línea]. <http://es.wikipedia.org/wiki/Programaci%C3%B3n_extrema> [citado en 31 de Octubre de 2010]

12. ANEXOS

ANEXO A: ESTÁNDARES

Los siguientes son los estándares usados dentro del proyecto.

Los objetos tendrán la siguiente estructura para ser nombrados:

tipo_Nombre_Complementos

Donde "tipo" es la referencia al objeto al que se está nombrando, el cual será especificado más adelante, el "Nombre" debe ir con la primera letra en mayúscula y el resto en minúscula, y se debe colocar el guion bajo para adicionarle más palabras también con letra capital las cuales son las denominadas "Complementos" y estos pueden ser cuantos puedan ser necesarios para identificar al objeto. A continuación se muestra la lista de tipos objetos usados en el proyecto junto a su forma de escribirlo dentro del código:

Tabla 4. Estándares

Objeto	Forma de escribirlo
Button	btn
CheckBox	chkbx
Clases	Clss
ComboBox	cmbx
ContextMenuStrip	cntxtmnstrp
CrystalReportViewer	crystlRprtVwr
DataGridView	dtg
DateTimePicker	dttmpckr
Form	Frm
GroupBox	grpbx
Label	lbl
MenuStrip	mnstrp
Module	mdl
PictureBox	pctrbx
ProgressBar	prgrssbr
RadioButton	rdbtn

SerialPort	srlprt
StatusStrip	sttsstrp
SplitContainer	spltcntnr
TabControl	tbcntrl
TabPage	tbpg
TextBox	txt
Timer	tmr
ToolStripMenuItem	mnItm
ToolStripStatusLabel	tlstrpstlbl
ToolStripSeparator	tlstrpsprtr
ToolTip	tltp

A continuación se define la forma de nombrar los demás elementos usados dentro del proyecto:

- **Eventos:** tipo_Nombre_Complemento_Evento

Los eventos tienen una estructura similar a la anterior pero adicionándole al final el tipo de evento al que pertenece.

- **Funciones y Variables:** nombreComplementos

En estas no se hace uso de los guiones bajos para tener una diferencia entre un objeto y una variable y/o función.

ANEXO B: HISTORIAS DE USUARIO

Ilustración 5. Historia de Usuario No 1

HISTORIA DE USUARIO

Número:	1	Nombre de Historia de Usuario:	Requerimiento Inicial Formulación del Proyecto
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			——— o ———
Usuario:	Lina Ma Muñoz	Iteración Asignada:	1. Propuesta de Sistema. V1.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se requiere diseñar una propuesta para el sistema de Identificación automático de llamadas con las siguientes características: - Se debe reconocer automaticamente la dirección luego de consultar el número telefónico en la base de datos. - La dirección podrá ser modificada si el cliente confirma una ubicación distinta en la base de datos. - Se permite asignar un móvil para prestar el servicio - Se debe guardar en la base de datos la información de prestamo de servicio, se debe guardar: Fecha y hora, movil, dirección en la q' se prestó el servicio. - La información de los servicios prestados, debe poderse consultar en pantalla (para confirmar datos) - De la información histórica se deben poder generar reportes.		
Observaciones:	- Los reportes generados podran consultar: • moviles con mayor número de servicios prestados • operador con mayor número de servicios prestados • Cliente con mayor número de servicios solicitados. • Las consultas de los reportes podrán ser ejecutadas para una fecha orango de fechas. • Se requiere que el sistema controle el acceso al sistema mediante una ventana de validación de usuarios. NOTA: Solo se requiere diseño de navegabilidad y propuesta de entorno gráfico.		

Ilustración 6. Historia de Usuario No 2

HISTORIA DE USUARIO

Número:	2.	Nombre de Historia de Usuario:	Registro de usuario. Validación de Usuario.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			1. Requerimiento Inicial.
Usuario:	Dixón Fdo Cano.	Iteración Asignada:	1. Propuesta de Sistema.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	——— o ———
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	——— o ———
Descripción:	Se requiere que la ventana de validación de usuario solicite nombre de usuario, contraseña. Los usuarios registrados son registrados por otro con rol administrativo. -Cada usuario es responsable de la cuenta y de proteger la contraseña. Usuario Administrador: Este usuario es el que adiciona los usuarios al sistema, más adelante se detallará la interfaz administrativa		
Observaciones:	• Se almacenarán los usuarios en la base de datos, podrán ser creados, modificados y eliminados		

Ilustración 7. Historia de Usuario No 3

HISTORIA DE USUARIO

Número:	3	Nombre de Historia de Usuario:	Ventana Principal.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			1. Requerimiento Inicial.
Usuario:	Dixon Fdo Cano	Iteración Asignada:	1. Propuesta Inicial del Sistema.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	——— 0 ———
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO	Puntos Reales:	——— 0 ———

Descripción:

Datos a desplegar en las ventanas principal:

- Principal
  - Línea (entrada desde el identificador de llamadas)
  - Dirección (obtenida automáticamente de la base de datos)
  - Teléfono (entrada desde el identificador de llamadas)
  - Hora (Hora de registro desde el identificador de llamadas)
  - Fecha (Fecha de registro desde el identificador de llamadas)
  - Campo Movil para asignar el servicio.
  - Botón de salida.
  - Fecha y hora del sistema (Reloj).

- Historial
  - Línea
  - Dirección
  - Teléfono
  - Hora
  - Fecha
  - Movil (Que ha sido asignado)
  - Botón de salida.

Observaciones:

- Consultas / Reportes.
  - Objeto di consulta. (moviles, cliente, operador)
  - Criterio de consulta (Cliente, móvil, Operador)
  - Rango de consulta (Hoy, Rango de Fechas)

Ilustración 8. Historia de Usuario No 4

HISTORIA DE USUARIO

Número:	4.	Nombre de Historia de Usuario:	Comunicación con Sistemas Externos.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Dixon Fdo Cano.	Iteración Asignada:	2. Prototipo Inicial (instalador)
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	3
Riesgo en Desarrollo: (Alto / Medio / Bajo)	ALTA	Puntos Reales:	3.
Descripción:	• Se requiere implementar las historias de usuario 1, 2, 3, 4 en un prototipo que permita interacción con el cliente. • Se requiere diseñar un esquema inicial de bases de datos para entregar como propuesta, esto con el fin de verificar que datos serán almacenados para cada elemento del sistema y proceso.		
Observaciones:	Se propone al cliente el uso de herramientas gratuitas teniendo en cuenta las licencias, términos y condiciones de uso del software seleccionado. → Crear esquema inicial de Base de Datos, se evaluan posibles motores MySQL y PostgreSQL.		

Ilustración 9. Historia de Usuario No 5

HISTORIA DE USUARIO

Número:	5	Nombre de Historia de Usuario:	Arquitectura del Sistema.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina Ma Muñoz.	Iteración Asignada:	3. Arquitectura
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	2.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO	Puntos Reales:	3.
Descripción:	→ Se requiere dar inicio a la arquitectura necesaria para permitir que el sistema reciba en un equipo central la información desde sistema externos (identificador de llamadas) y permita distribuir dicha información a varios equipos diferentes → Se debe crear una comunicación serial para conectarse con el dispositivo identificador de líneas telefónicas		
Observaciones:	→ Se establece con el cliente la necesidad de una red interna antes de poder implantar esta entrega. → Los equipos cliente y principal (servidor) no requieren especificaciones técnicas de importancia. → Se aplica en el proyecto una arquitectura cliente - servidor. * Se presentaron retrazos por demora en la instalación y configuración de la red interna.		

Ilustración 10. Historia de Usuario No 6

HISTORIA DE USUARIO

Número:	6.	Nombre de Historia de Usuario:	Mejoras Servidor del sistema.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			5. Arquitectura del sistema.
Usuario:		Iteración Asignada:	3.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	El cliente solicita que se adicionen las siguientes características al programa que inicia la comunicación en el sistema: • Luego de inicializarse el ícono y el programa deben ocultarsen para evitar manipulación por accidente. • No cerrarse con la "X" roja de la esquina de la ventana, sino seleccionar SALIR. Cuando se haga click en la "X" roja, la ventana se oculta. • Al momento de cerrar el servidor, se debe limpiar la caché de todos los servicios no prestados. • Cuando se salga el servidor, este envíe una notificación a todos los servicios cliente que tenga asociados.		
Observaciones:	Adicionalmente, se cambia íconos de la ventana y diseño de la ventana de validación de usuario.		

Ilustración 11. Historia de Usuario No 7

HISTORIA DE USUARIO

Número:	7.	Nombre de Historia de Usuario:	Recepción de aparato decodificadora/ identificador.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz.	Iteración Asignada:	4.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	3.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	ALTA.	Puntos Reales:	3.
Descripción:	→ Se requiere extraer la información del número de teléfono desde la cadena enviada por el dispositivo electrónico identificador de llamadas → Se deben omitir las cadenas enviadas por la inicialización del dispositivo, cuando este se inicializa. →		
Observaciones:	Se toman muestras de las cadenas para iniciar la implementación del parseo de la información en la cadena.		

Ilustración 12. Historia de Usuario No 8

HISTORIA DE USUARIO

Número:	8	Nombre de Historia de Usuario:	Correcciones en Cadenas.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			7. Recepción de aparato identificador
Usuario:	Dixon Fdo Cano.	Iteración Asignada:	4.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	ALTA.	Puntos Reales:	1.
Descripción:	→ Se corrigen las cadenas y el parseo que se realiza para obtener la información necesario para el sistema. → Se identifica que como nuevo requerimiento y consideración para funcionamiento del sistema es necesario que el hardware se configure correctamente antes de poner en marcha el sistema. → En la pantalla principal del identificador se debe recibir el número telefónico identificado y mostrarse la línea por la cual ingresó la llamada, la hora, la fecha y buscar en la base de datos la dirección. • Si la dirección se encuentra en la base de datos, se debe desplegar en la grilla principal. • Si la dirección no se encuentra en la base de datos mostrar que no tiene dirección.		
Observaciones:	→ Cuando una dirección no se encuentra en la base de datos el operador asignará una dirección al servicio manualmente.		

Ilustración 13. Historia de Usuario No 9

HISTORIA DE USUARIO

Número:	9	Nombre de Historia de Usuario:	Editar Direcciones Nuevas en la base de datos.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz	Iteración Asignada:	5.
Prioridad en Negocio: (Alta / Media / Baja)	MEDIA.	Puntos Estimados:	2.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	2.
Descripción:	Para los direcciones no existentes en la base de datos, se requiere que el operador pueda asignar la dirección a la base de datos. → la dirección podrá ser guardada definitivamente → la dirección solo se asignará al servicio, pero en la base de datos el número telefónico no tendrá una dirección asignada. Para las direcciones existentes en la base de datos se requiere que el operador pueda asignar la dirección que se requiera. con las siguientes opciones. → La dirección podrá ser asignada al servicio definitivamente (permite cambio posterior), esto es, que la dirección ingresada reemplazará la dirección que se tiene en el momento. → la dirección solo se asignará al servicio, esto es que no reemplaza la dirección que se tenga asignada al número telefónico en ese momento.		
Observaciones:	* Se creará una ventana que permita la edición de la dirección, y seleccionar la opción de almacenamiento.		

Ilustración 14. Historia de Usuario No 10

HISTORIA DE USUARIO			
Número:	10	Nombre de Historia de Usuario:	Despliegue de llamadas entrantes.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			7. Recepción del aparato identificados.
Usuario:	Dixón Fdo Cano L.	Iteración Asignada:	6.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	1.
Descripción:	Se requiere modificar la forma en como se despliega la información de las llamadas entrantes. → las llamadas no se desplegaran hacia abajo, se tendran solo 10 líneas en la grilla, y cuando se detecte la entrada de una llamada por una línea determinada esta se desplegará en la línea correspondiente de la ventana.		
Observaciones:	.		

Ilustración 15. Historia de Usuario No 11

HISTORIA DE USUARIO			
Número:	11.	Nombre de Historia de Usuario:	Alta y Baja de Identificadores.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz	Iteración Asignada:	6.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	1.
Descripción:	Se requiere que el servidor detecte automáticamente la entrada y salida de un cliente en el sistema Se debe avisar a los sistemas ó a los operadores cuando el servidor se caiga y se vuelva a levantar, para esto el servidor debe encontrar las personas que se encuentran activas en el sistema.		
Observaciones:			

Ilustración 16. Historia de Usuario No 12

HISTORIA DE USUARIO

Número:	12.	Nombre de Historia de Usuario:	Atención de lineas (Despacho de servicio)
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz	Iteración Asignada:	7.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	2
Riesgo en Desarrollo: (Alto / Medio / Bajo)	ALTA.	Puntos Reales:	2.
Descripción:	Se requiere que el sistema permita asignar un servicio, asociando un móvil con una determinada llamada, el móvil será ingresado por teclado en un campo de la ventana principal del sistema. → Guardar Histórico de los servicios prestados: Se requiere disponer de un historial de todos los servicios prestados a través del sistema, para que puedan ser consultados en cualquier momento. El historial debe desplegar la siguiente información • Linea. • dirección • Teléfono • Hora • Fecha • Movil. que fue' asignado.		
Observaciones:			

Ilustración 17. Historia de Usuario No 13

HISTORIA DE USUARIO

Número:	13.	Nombre de Historia de Usuario:	Atención de líneas (selección de líneas)
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			12. Atención de líneas. Despacho Servicio.
Usuario:	Dixon Folo Cano Largo	Iteración Asignada:	8.
Prioridad en Negocio: (Alta / Media / Baja)	MEDIA	Puntos Estimados:	2.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	ALTO.	Puntos Reales:	3.
Descripción:	• Se requiere que se permita avisar a todos los operadores que un servicio ya fué atendido para que sea eliminado de los servicios pendientes. • Se debe avisar a todos los operadores que un servicio está siendo atendido por otro operador. • Se debe revertir la acción anterior de manera que si un operador desea liberar un servicio pendiente, otro operador lo atienda.		
Observaciones:	→ Se define que para los servicios que están siendo atendidos por otros operadores las líneas en pantalla se vean color gris claro. → Se define que para los servicios que está atendiendo el usuario, las líneas en pantalla sean verdes. * De manera que cuando mis líneas son verdes, los demás operadores las ven de color gris claro. → Las líneas sin atender son de color Blanco		

Ilustración 18. Historia de Usuario No 14

HISTORIA DE USUARIO

Número:	14.	Nombre de Historia de Usuario:	Busqueda en Historial.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			12. Atención de lineas (Despacho servicio)
Usuario:	Dixon Fdo Cano.	Iteración Asignada:	9.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se requiere poder realizar busquedas en el historial con el numero de móvil.		
Observaciones:	Este cambio se realiza por la dificultad para encontrar información de un movil especifico. Con la busqueda cuando el código del móvil ingresado es vacio, el historial muestra todos los registros.		

Ilustración 19. Historia de Usuario No 15

HISTORIA DE USUARIO

Número:	15	Nombre de Historia de Usuario:	Asignación de Pendientes.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			13. Atención de líneas.
Usuario:	Lina María Muñoz	Iteración Asignada:	9.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	1.
Descripción:	Se requiere que la asignación de una línea pendiente que está siendo atendida por un operador pueda hacerse a varias líneas mediante una acción controlada también por el operador. La liberación de una línea pendiente deberá poder realizarse también mediante este mecanismo. - Se adiciona que el número de líneas de la grilla que despliega los servicios pendientes sea configurable por el usuario.		
Observaciones:			

Ilustración 20. Historia de Usuario No 16

HISTORIA DE USUARIO

Número:	16.	Nombre de Historia de Usuario:	Obtención Automática de móviles
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Dixon Fdo Cano	Iteración Asignada:	9.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	2
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	2.
Descripción:	Se requiere crear una comunicación serial para conectarse con el dispositivo electrónico que identifica a los móviles que se reportan a la central, se deben desplegar en una lista con el mismo orden con que se reportan. Esta lista debe dar la posibilidad de seleccionar un movil. y así realizar una asignación de un servicio.		
Observaciones:	Este requerimiento permite adicionar dinámicamente el código de un movil a una prestación de servicio.		

Ilustración 21. Historia de Usuario No 17

HISTORIA DE USUARIO

Número:	17.	Nombre de Historia de Usuario:	Administrador del sistema.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz	Iteración Asignada:	10.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	3.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	ALTA.	Puntos Reales:	3.
Descripción:	Se requiere crear la interfaz administrativa del Sistema que permita: Crear, modificar, consultar y eliminar un determinado elemento u objeto en el sistema. - Usuarios: • Crear usuarios • Modificar usuarios: Nombre, Apellido, Usuario, Contraseña, privilegios. • Eliminar Usuarios - Moviles: • Crear móviles • Modificar móviles. • Eliminar móviles. - Registros: • Crear Registros (Sistema fuera de servicio).		
Observaciones:	• Modificar registros (Por error del operador) • Eliminar Registros: . - Generación de Reportes: • Objeto del Reporte (Cliente, moviles). • Rango de la consulta (Hoy, rango de Fechas).		

Ilustración 22. Historia de Usuario No 18

HISTORIA DE USUARIO

Número:	18	Nombre de Historia de Usuario:	Generación de Reportes.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Dixon Fernando C.	Iteración Asignada:	10.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	3
Riesgo en Desarrollo: (Alto / Medio / Bajo)	ALTA.	Puntos Reales:	3.
Descripción:	Se requieren los siguientes Reportes: • Registro de servicios prestados por: - Cliente (número telefónico) - Operador (cédula) - Todos. • Mayor numero de servicios prestados por: - Cliente - Operador - ~~Todos~~ Móviles.		
Observaciones:			

Ilustración 23. Historia de Usuario No 19

HISTORIA DE USUARIO

Número:	19.	Nombre de Historia de Usuario:	Registro de llamadas no atendidas.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Dixon Fdo Cano	Iteración Asignada:	11.
Prioridad en Negocio: (Alta / Media / Baja)	MEDIA	Puntos Estimados:	2.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	3.
Descripción:	Se requiere un tiempo de vencimiento de las llamadas en espera de ser atendidas el tiempo se establece en 1 minuto. Luego de acabarse el minuto de la llamada en espera, esta linea será eliminada de pantalla y se registrará automáticamente en la base de datos como perdida. Se debe permitir la eliminación de llamadas y asignarles una causa. (no será llamada atendida ni perdida).		
Observaciones:	Los operadores no aceptaron el plazo de un minuto y el contador se modifica a 3 minutos de espera. Las causas de eliminación serían por ejemplo: - llamada de consulta. - llamada equivocada. - Otra: Se especifica la causa por el operador manualmente. - Se adiciona ventana de configuración de puerto serial.		

Ilustración 24. Historia de Usuario No 20

HISTORIA DE USUARIO

Número:	20.	Nombre de Historia de Usuario:	Reporte de llamadas no atendidas.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			19: Registro llamadas no atendidas
Usuario:	Lina Ma Muñoz	Iteración Asignada:	11
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	2.
Descripción:	Se requiere generar reportes con las siguientes condiciones. → Registro llamadas Perdidas consultadas por: • Cliente • Operador • Todos. → Registro llamadas Eliminadas consultadas por. • Cliente • Operador • Razón • Todos. → Rendimiento consultados por. • Servicio completo. • Operador.		
Observaciones:			

Ilustración 25. Historia de Usuario No 21

HISTORIA DE USUARIO

Número:	21	Nombre de Historia de Usuario:	SANCIONES. (Administrador)
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			17 Administrador del sistema.
Usuario:	Lina María Muñoz	Iteración Asignada:	12.
Prioridad en Negocio: (Alta / Media / Baja)	MEDIA.	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se requiere que el usuario administrador pueda asignar sanciones a un móvil en una jornada determinada, la sanción está asociada a una dependencia que puede ser: • Gerencia. • Junta Vigilancia • Oficina.		
Observaciones:			

Ilustración 26. Historia de Usuario No 22

HISTORIA DE USUARIO

Número:	22	Nombre de Historia de Usuario:	Sanciones. (Principal).
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			16. Obtención automatica de móviles
Usuario:	Dixon Fdo Cano.	Iteración Asignada:	12.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA.	Puntos Estimados:	1
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	1.
Descripción:	Se requiere que los móviles que se encuentren Sancionados se puedan identificar al momento de ser enlistados en pantalla para esto se define el siguiente código de colores: • Sanciones de Gerencia = Rojo. • " " Oficina = AMARILLO. • Sanciones de Junta de Vigilancia = VERDE		
Observaciones:	El campo de movil sancionado será pintado del color establecido y la fuente será de color negro. - Se solicita por parte de los operadores que el código de colores para las sanciones puedan se ocultados a través de una opción en el menú de configuraciones.		

Ilustración 27. Historia de Usuario No 23

HISTORIA DE USUARIO

Número:	23	Nombre de Historia de Usuario:	Llenado de Datos
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Dixon Fdo Cano.	Iteración Asignada:	12.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se requiere el llenado de datos correspondientes a. - Usuarios administradores - Moviles. - Sanciones en el momento. - Directorio telefonico : PENDIENTE		
Observaciones:	Se realiza llenado de datos en una cantidad pequeña los demás datos son migrados por los usuarios administradores.		

Ilustración 28. Historia de Usuario No 24

HISTORIA DE USUARIO

Número:	24	Nombre de Historia de Usuario:	Asignación de varios móviles
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			16. Obtención automatica de móviles
Usuario:	Lina María Moñoz T	Iteración Asignada:	12.
Prioridad en Negocio: (Alta / Media / Baja)	ALTA	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se permite asignar varios móviles para prestar un servicio, esto cuando se solicitan varios taxis en una misma llamada.		
Observaciones:	Para asignar varios móviles se separan los móviles con un signo "+" y se coloca como cadena en el campo movil antes de asignar y aceptar el servicio.		

Ilustración 29. Historia de Usuario No 25

HISTORIA DE USUARIO

Número:	25	Nombre de Historia de Usuario:	Consulta de Información de Moviles.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz	Iteración Asignada:	13.
Prioridad en Negocio: (Alta / Media / Baja)	MEDIA.	Puntos Estimados:	1
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se requiere adicionar en la ventana principal. una pestaña que permita consultar información de los móviles, se requiere desplegar la siguiente información: • No Móvil. • Placa • Marca • Modelo del vehículo. • Dirección del propietario del vehículo. • Teléfono del propietario del vehículo.		
Observaciones:	La consulta se realiza ingresando el código del lateral del móvil.		

Ilustración 30. Historia de Usuario No 26

HISTORIA DE USUARIO

Número:	26.	Nombre de Historia de Usuario:	Planillas.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			25 Consulta de Inf. de móviles.
Usuario:	Dixon Fdo Cano.	Iteración Asignada:	14.
Prioridad en Negocio: (Alta / Media / Baja)	MEDIA	Puntos Estimados:	2.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO	Puntos Reales:	1.
Descripción:	Se requiere llevar un control de planillas que se asigna a un móvil. El número máximo de planillas que se deben tener es de 3 por mes. → Se debe adicionar en la consulta de información de móviles un campo para las planillas.		
Observaciones:	• Cambio de Esquema. • Se modifica el GUI principal / pestaño de móviles. •		

Ilustración 31. Historia de Usuario No 27

HISTORIA DE USUARIO

Número:	27.	Nombre de Historia de Usuario:	Contador de servicios prestados.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Dixón Fdo Cano	Iteración Asignada:	13.
Prioridad en Negocio: (Alta / Media / Baja)	MEDIO.	Puntos Estimados:	1.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	1.
Descripción:	Se requiere adicionar un contador para el número de servicios prestados por cada operador. El contador debe poderse recuperar en caso de una caida del sistema.		
Observaciones:			

Ilustración 32. Historia de Usuario No 28

HISTORIA DE USUARIO

Número:	28	Nombre de Historia de Usuario:	Super Backup.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz	Iteración Asignada:	16.
Prioridad en Negocio: (Alta / Media / Baja)	BAJA.	Puntos Estimados:	3.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO.	Puntos Reales:	3.
Descripción:	Se requiere que el sistema haga un súper backup, el cual consiste en generar todos los reportes posibles del sistema y guardarlos en una carpeta seleccionada por el usuario y para que el usuario los pueda almacenar en otros medios. Se debe poder eliminar la información de los datos a los que se les ha sacado super backup. y borrarlos definitivamente del sistema.		
Observaciones:	Se debe restringir el acceso solo a usuarios administrativos		

Ilustración 33. Historia de Usuario No 29

HISTORIA DE USUARIO

Número:	29.	Nombre de Historia de Usuario:	Registro manual de servicios.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Lina María Muñoz	Iteración Asignada:	15
Prioridad en Negocio: (Alta / Media / Baja)	BAJA	Puntos Estimados:	1
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIA.	Puntos Reales:	1.
Descripción:	Se debe brindar la posibilidad de registrar un servicio así no sea codificado por el identificador. Se adicionó esta funcionalidad en el menu de Archivo.		
Observaciones:			

Ilustración 34. Historia de Usuario No 30

HISTORIA DE USUARIO

Número:	30	Nombre de Historia de Usuario:	Registro de Emergencias.
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			
Usuario:	Dixon Fdo Cano.	Iteración Asignada:	13
Prioridad en Negocio: (Alta / Media / Baja)	MEDIA	Puntos Estimados:	2.
Riesgo en Desarrollo: (Alto / Medio / Bajo)	BAJO.	Puntos Reales:	2.
Descripción:	→ Se requiere poder capturar el llamado de emergencia de un movil y tenerlo en pantalla de forma que el operador del sistema se percate de la emergencia. → Se adicionan seguimientos en la misma forma que se muestra la emergencia. → Se puede sacar tanto una emergencia como un seguimiento en pantalla.		
Observaciones:			

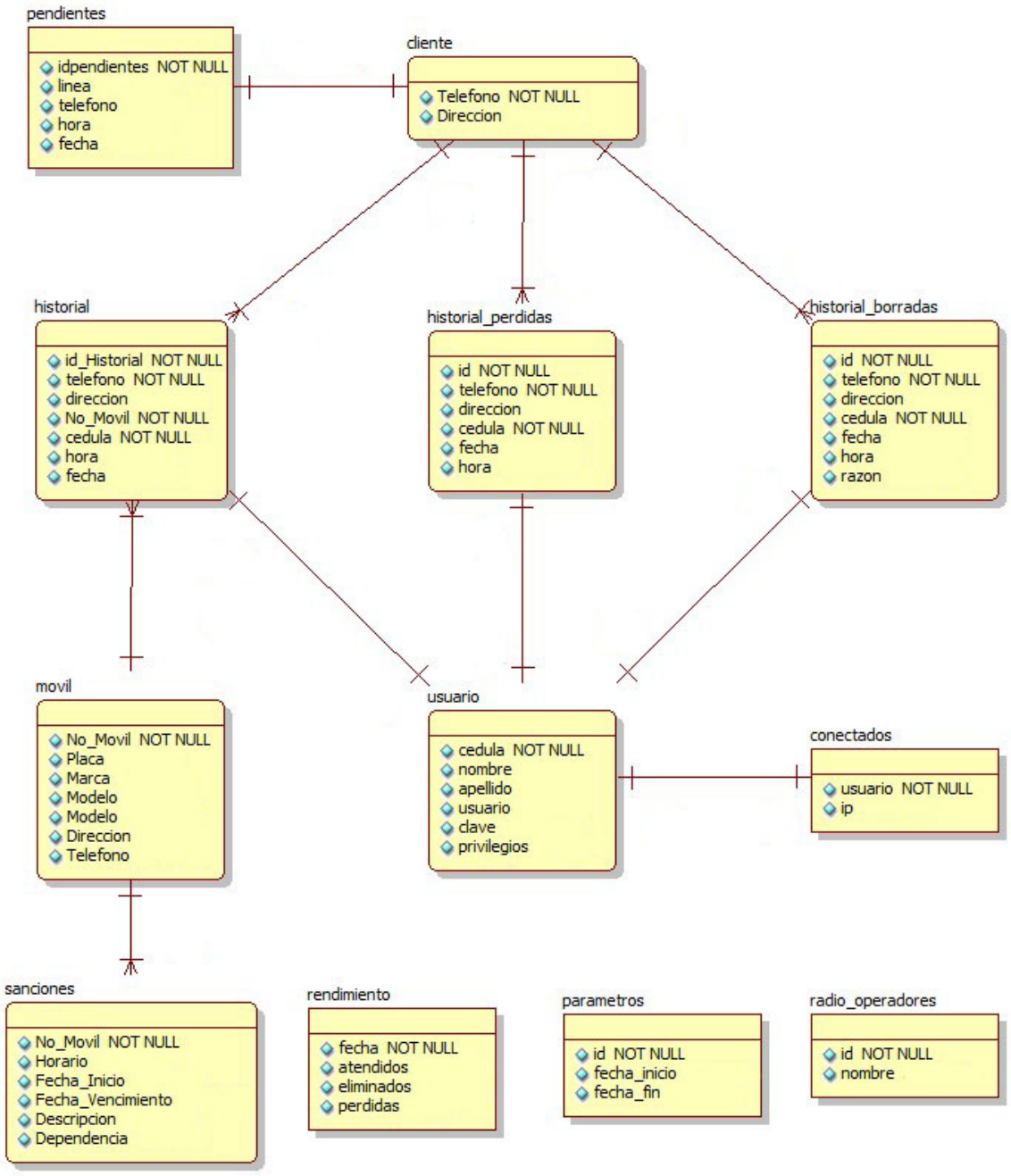
Ilustración 35. Historia de Usuario No 31

HISTORIA DE USUARIO

Número:	31	Nombre de Historia de Usuario:	Mejora a Seguimiento emergencias
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):			30. Registro de emergencias.
Usuario:	Lina María Muñoz To	Iteración Asignada:	16.
Prioridad en Negocio: (Alta / Media / Baja)	BAJO	Puntos Estimados:	1
Riesgo en Desarrollo: (Alto / Medio / Bajo)	MEDIO	Puntos Reales:	1
Descripción:	Las emergencias y los seguimientos debe quedar en un archivo de log el cual debe brindar la posibilidad de editarse para adicionar notas a dicho evento.		
Observaciones:			

ANEXO C: MODELO ENTIDAD-RELACIÓN FINAL

Ilustración 36. Modelo Entidad-Relación Final



ANEXO D: DIAGRAMAS DE CASOS DE USO Y ACTIVIDADES

A continuación se presentan los diagramas de los casos de uso más importantes para el sistema. Para cada caso de uso dan los diagramas de actividades que se consideraron necesarios y suficientes para su entendimiento.

Ilustración 37. Diagrama de Caso de Uso de Despacho de Taxis

Ilustración 38. Diagrama de Actividades - Prestación de Servicio

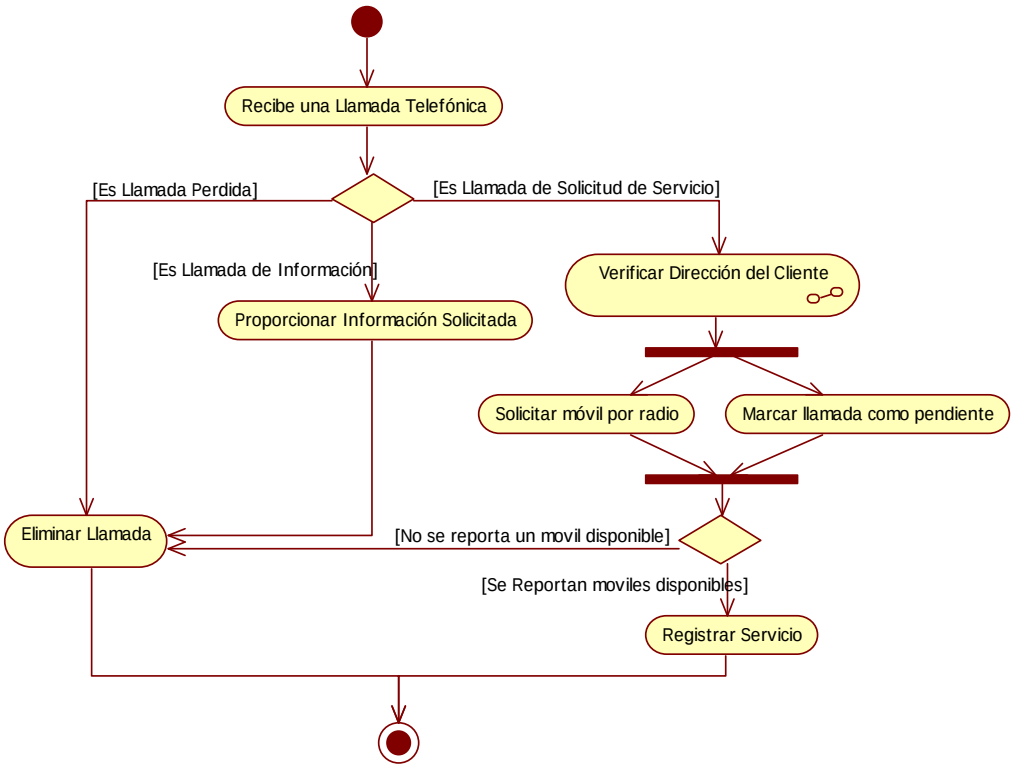


Ilustración 39. Diagrama de Actividades de la Verificación de Dirección del Cliente

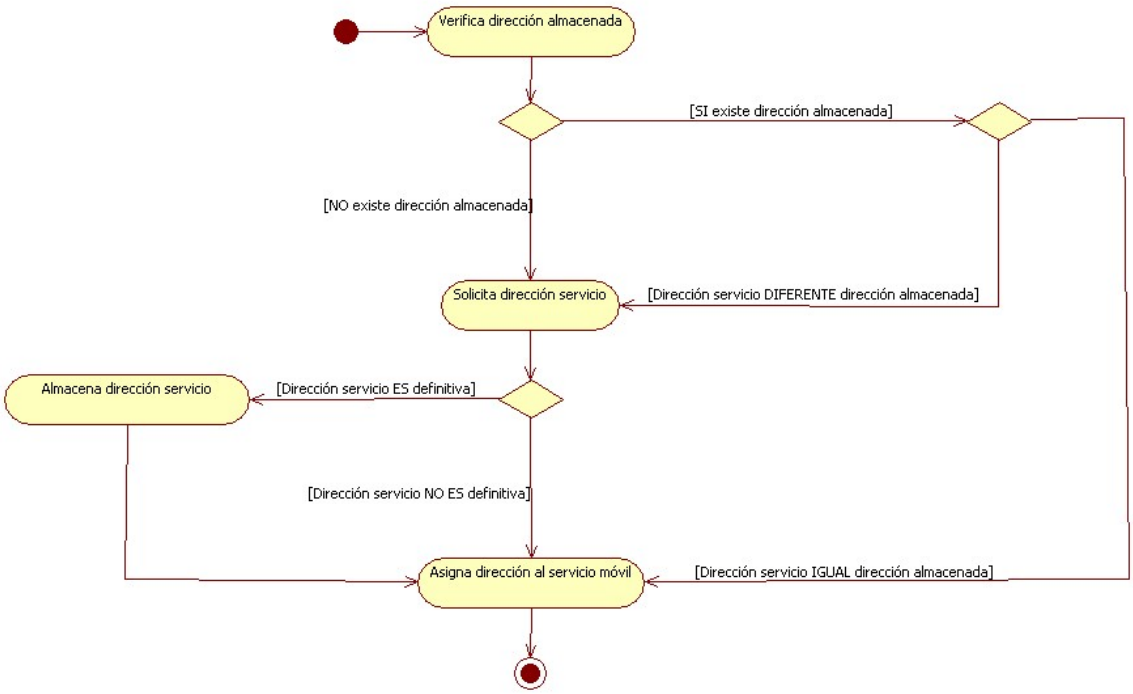


Ilustración 40. Diagrama de Actividades de la Consulta de Historial

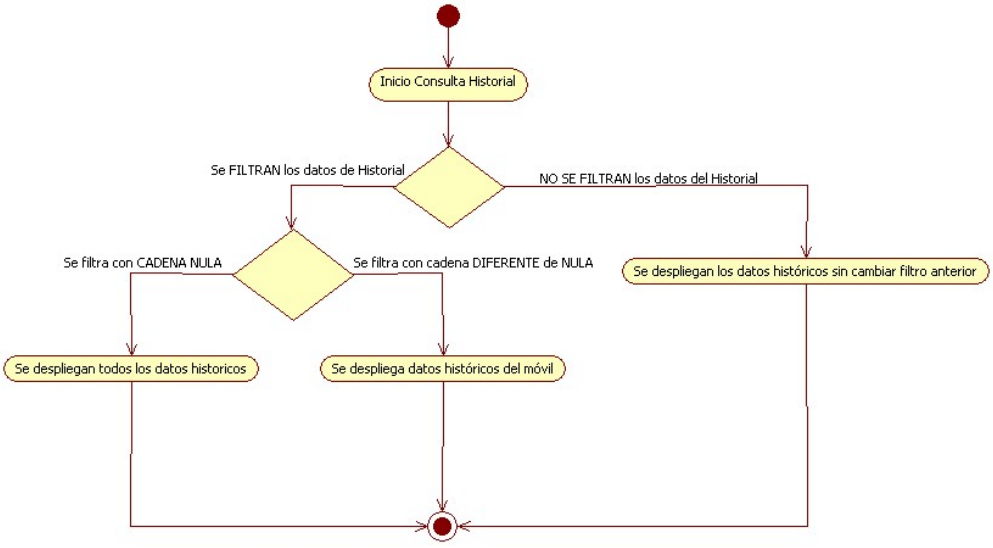


Ilustración 41. Diagrama de Actividades de la Consulta de Información de un Móvil

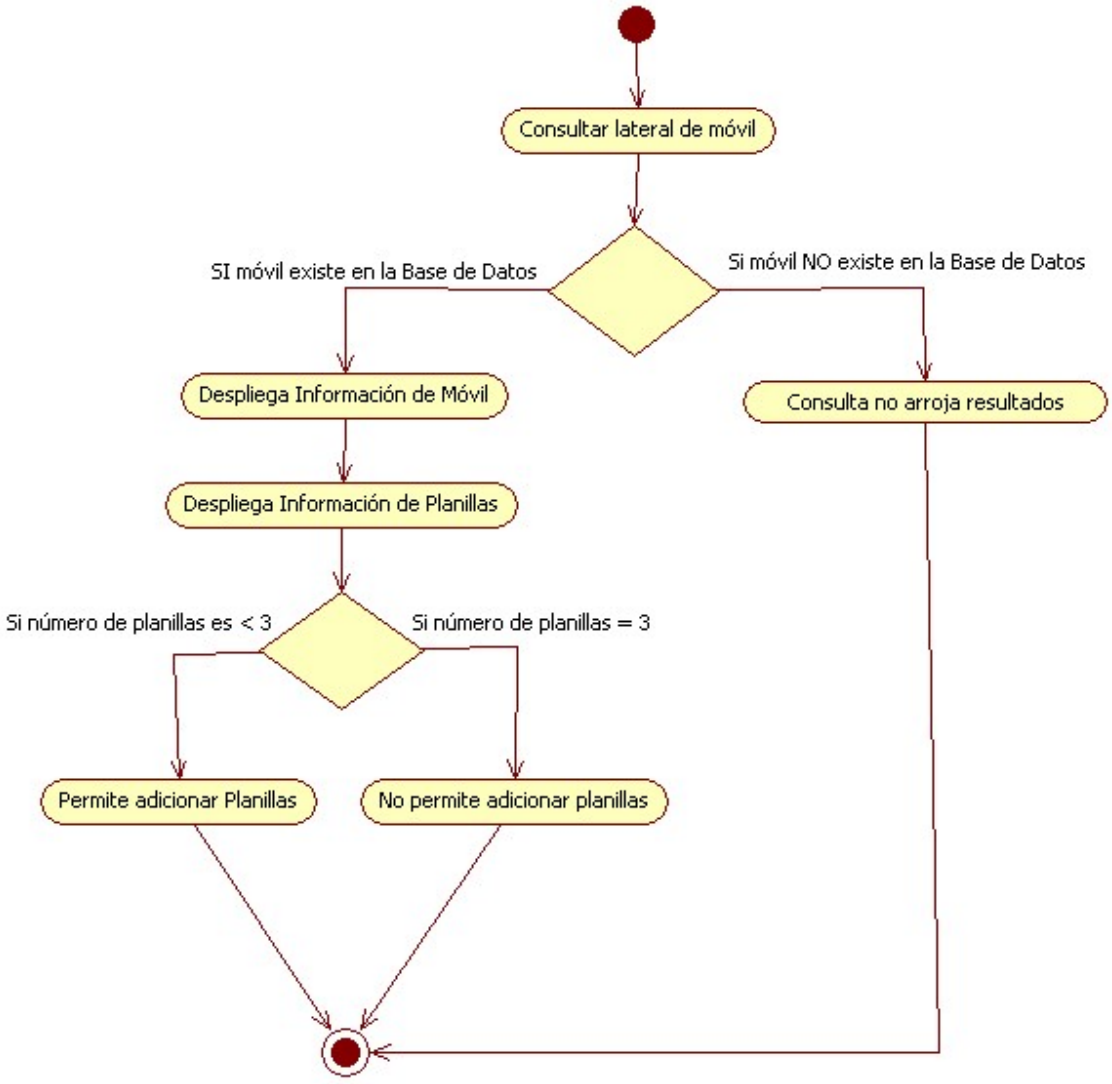


Ilustración 42. Diagrama de Caso de Uso de Administración del Sistema

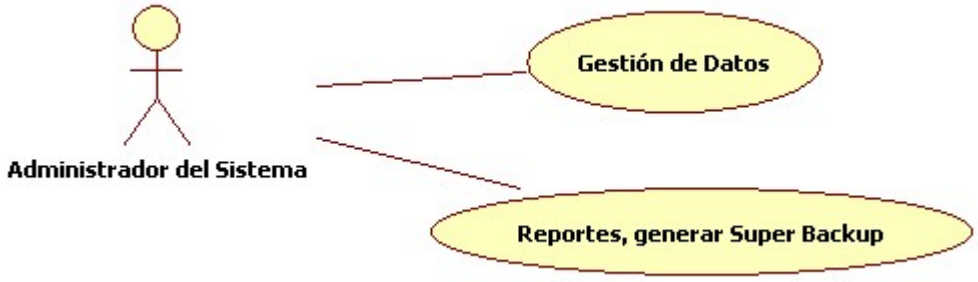


Ilustración 43. Diagrama de Actividades de la Gestión de Datos

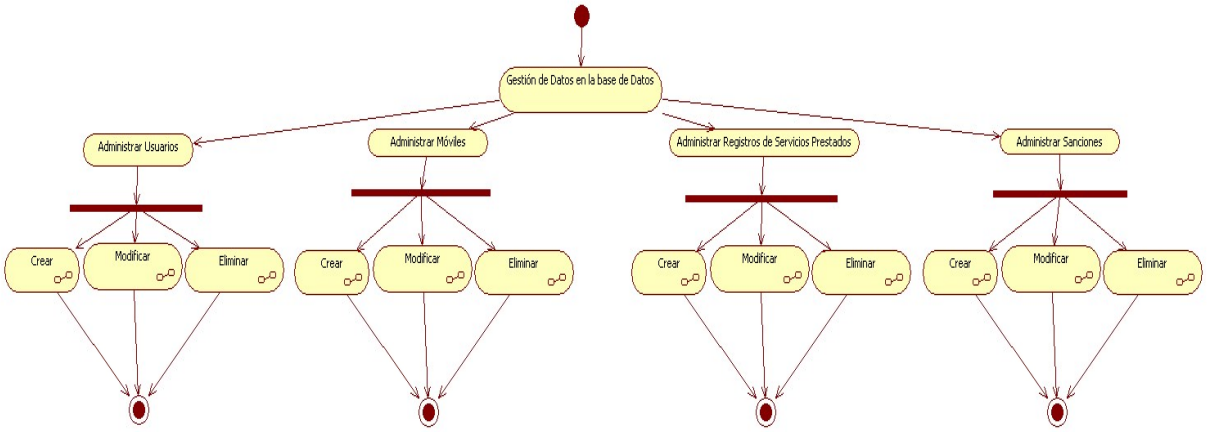


Ilustración 44. Diagrama de Actividades de la Creación de un registro

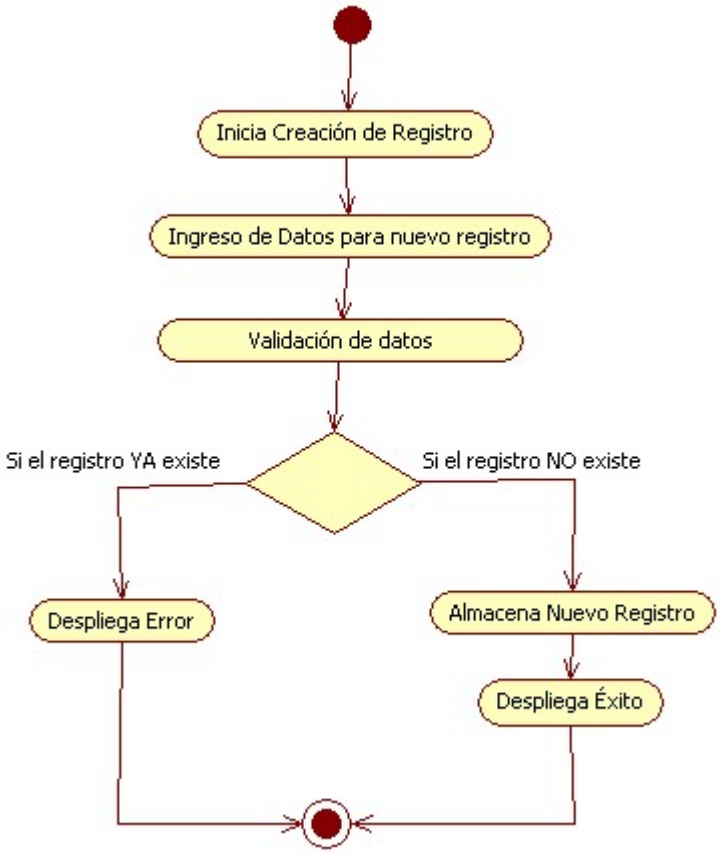


Ilustración 45. Diagrama de Actividades de la Modificación de un registro.

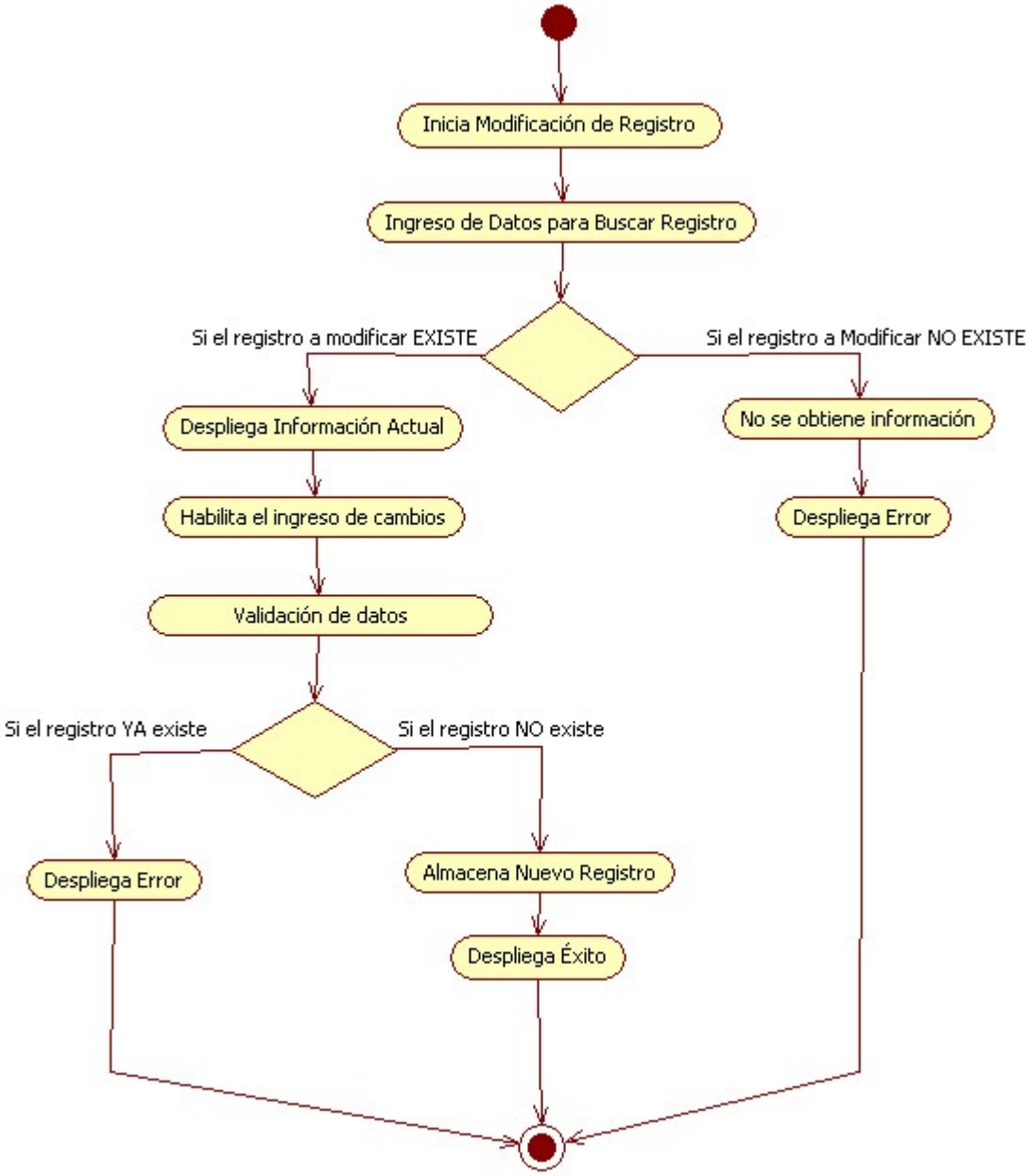


Ilustración 46. Diagrama de Actividades de la Eliminación de un registro

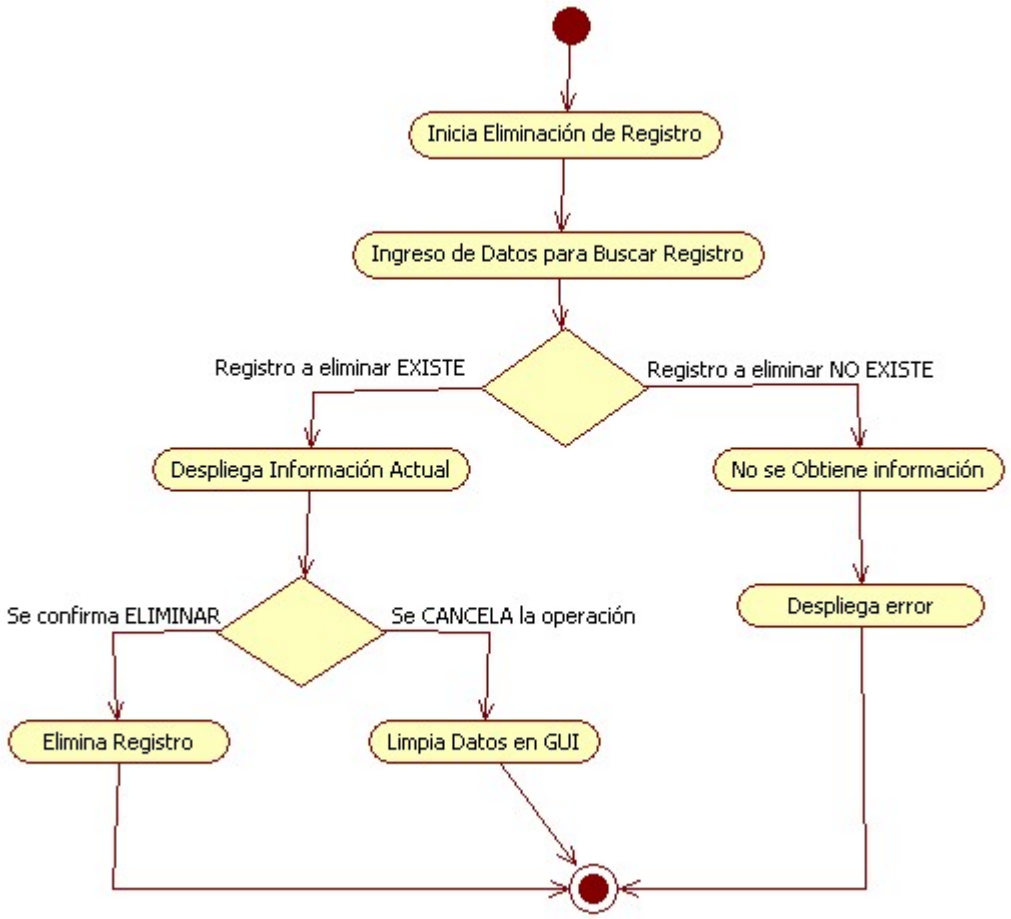
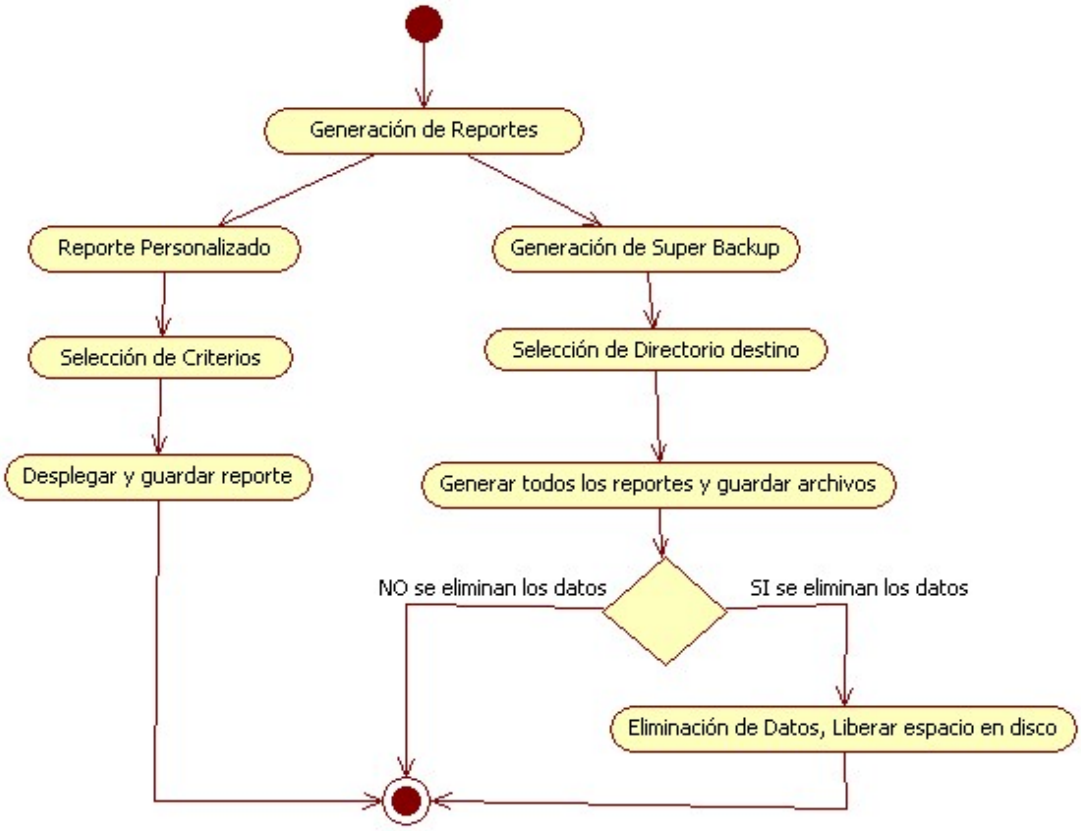


Ilustración 47. Diagrama de Actividades de la Generación de Reportes/SuperBackup



ANEXO E: TARJETAS CRC

A continuación se encuentran las Tarjetas de clase, responsabilidad, colaboración (CRC cards) del Componente Servidor. Se hace la aclaración que a estas tarjetas tienen un componente adicional al que XP habla, se le adiciono los atributos de cada clase, que son diferenciados por medio de un guion, para tener una mejor facilidad al momento de asignar responsabilidades.

Ilustración 48. CRC Clase Cliente

Clss_Cliente	
-hayIp -ipAddr enviarMensaje	

Ilustración 49. CRC Clase Log

Clss_Log	
escribir	

Ilustración 50. CRC Clase Servidor

Clss_Servidor	
-data -log -salir -clientes iniciar parsear conectar enviarMensaje desconectar sacarDeBD atendido lineaPendiente lineaQuitarPendiente broadcoast	Clss_Log Clss_Cliente

Ilustración 51. CRC Formulario Configuración

Frm_Configuracion	
-frmPropiedadesAbierto	Frm_Principal
-padre	Frm_Propiedades
frm_Configuracion_Load	
frm_Configuracion_Unload	
btn_Configurar_Llamadas_Click	
btn_Cancelar_Click	
btn_Aceptar_Click	
verificarPuerto	

Ilustración 52. CRC Formulario Propiedades

Frm_Propiedades	
-principal	Frm_Principal
-padre	Frm_Configuracion
-nombre	
frm_Propiedades_Load	
btn_Predeterminados_Click	
btn_Aceptar_Click	
btn_Cancelar_Click	
btn_Aplicar_Click	
cmbx_Habilitar_Aplicar_IndexChange	
cargarConfiguracion	
predeterminados	
guardarConfiguracion	

Ilustración 53. CRC Modulo Variables

mdl_Variables	
-listaHilos	
-tiempo	
-DATABASE	
-SERVER	
-USUARIO	
-PASSWORD	

Ilustración 54. CRC Formulario Principal

Frm_Principal	
-hiloServidor	Clss_Servidor
-delegadoPuertoSerial	Clss_Log
-servidor	Clss_Cliente
-salirCompletamente	Frm_Configuracion
-log	
-iconoBarraTareas	
-menuBarraTareas	
-puertoSerial	
frm_Principal_Load	
btn_Iniciar_Servidor_Click	
btn_Detener_Servidor_Click	
mnItm_Configuracion_Click	
mnItm_Mostrar_Click	
mnItm_Ocultar_Click	
mnItm_Cerrar_Click	
srlprt_Puerto_Serial_DataReceived	
cargarConfiguracionPuerto	
ejecutarHiloServidor	
notificarIngreso	
abrirPuerto	
ocultar	
informarSalida	
guardarLinea	
soloNumeros	
borrarTelefonoEnIdentificador	
borrarTelefonoEnBD	
informar	
conectarBaseDeDatos	
frm_Principal_FormClosing	
WndProc	

Tarjetas de clase, responsabilidad, colaboración (CRC cards) del Componente Administrador.

Ilustración 55. CRC Formulario Cambio Clave

Frm_Cambio_Clave	
btn_Aceptar_Click btn_Cancelar_Click	

Ilustración 56. CRC Formulario Editar Registro

Frm_Editar_Registro	
-idHistorial -usuario frm_Editar_Registro_Load cargarUsuarios btn_Editar_Direccion_Click btn_Editar_Movil_Click btn_Aceptar_Click btn_Cancelar_Click	

Ilustración 57. CRC Formulario Login

Frm_Login	
-frmLoginExpandida frm_Login_Load btn_Aceptar_Click btn_Cancelar_Click btn_Opciones_Click	Frm_Principal

Ilustración 58. CRC Formulario Súper Backup

Frm_Super_Backup	
btn_Realizar_Super_Backup borrarRegistros btn_Examinar_Click	Frm_Reportes

Ilustración 59. CRC Formulario Reportes

Frm_Reportes	
generarReporte_LlamadasPerdidas_Cliente generarReporte_LlamadasPerdidas_Operador generarReporte_LlamadasPerdidas_Todos generarReporte_LlamadasEliminadas_Cliente generarReporte_LlamadasEliminadas_Operador generarReporte_LlamadasEliminadas_Razon generarReporte_LlamadasEliminadas_Todos generarReporte_Rendimiento_ServicioCompleto generarReporte_Rendimiento_Operador organizarReporte organizarReporteRendimiento actualizarParametros borrarRendimiento llenarFechas llenarAtendidos llenarBorrados llenarPerdidas exportar	

Ilustración 60. CRC Modulo Variables

mdl_Variables	
-DATABASE -SERVER -USERNAME -PASSWORD -nombreUsuario -STATUS_PRIVILEGIO -OPCION_USUARIO -COD_usuario	

Ilustración 61. CRC Formulario Principal

Frm_Principal	
-seguridadLimpiarDtgRegistrosDatos	Frm_Cambio_Clave
-dtgRegistrosDatosVacio	Frm_Editar_Registro
frm_Principal_Load	Frm_Reportes
ubicar	Frm_Super_Backup
frm_Principal_FormClosing	
ubicarTabUsuarios	
grupoOpcionesUsuario	
grupoDatosUsuario	
cmbx_Usuarios_Opciones_SelectedIndexChanged	
prepararNuevoUsuario	
limpiarGrpbxUsuariosDatos	
prepararEdicionUsuario	
prepararEliminacionUsuario	
estadoInicialUsuario	
txt_Usuarios_Cedula_Buscar_KeyPress	
btn_Usuarios_Buscar_Click	
spltcntnr_Usuarios_SplitterMoved	
cmbx_Usuarios_Privilegio_SelectedIndexChanged	
btn_Usuarios_Cambiar_Clave_Click	
txt_Usuarios_Verificacion_Clave_TextChanged	
btn_Usuarios_Guardar_Click	
validarExistente	
ubicarTabMoviles	
grupoOpcionesMoviles	
grupoDatosMoviles	
txt_Moviles_Codigo_H_Buscar_KeyPress	
cmbx_Moviles_Opciones_SelectedIndexChanged	
prepararNuevoMovil	
limpiarGrpbxMovilesDatos	
prepararEdicionMovil	
prepararEliminacionMovil	
estadoInicialMovil	
btn_Moviles_Buscar_Click	
spltcntnr_Moviles_SplitterMoved	
btn_Moviles_Guardar_Click	
cargarDtgRegistrosDatos	
ubicarTabRegistros	
grupoOpcionesRegistros	
grupoDatosRegistros	
cmbx_Registros_Opciones_SelectedIndexChanged	

estadoInicialRegistro	
mostrarGrupoRegistros	
limpiarDtgRegistrosDatos	
prepararEdicionRegistro	
txt_Registros_Telefono_Buscar_KeyPress	
chkbx_Registros_Fecha_CheckedChanged	
btn_Registros_Buscar_Click	
buscarTelefono	
obtenerUsuario	
mostarRegistros	
spltcntnr_Registros_SplitterMoved	
dtg_Registros_Datos_RowEnter	
btn_Registros_Registrar_Click	
agregarRegistros	
obtenerCodigo	
eliminarRegistro	
ubicarTabSanciones	
grupoOpcionesSanciones	
grupoDatosSanciones	
cmbx_Sanciones_Opciones_SelectedIndexChanged	
prepararNuevaSancion	
prepararEdicionSancion	
prepararEliminacionSancion	
estadoInicialSancion	
txt_Sanciones_Movil_H_Buscar_KeyPress	
cmbx_Sanciones_Jornada_Buscar_SelectedIndexChanged	
btn_Sanciones_Buscar_Click	
spltcntnr_Sanciones_SplitterMoved	
btn_Sanciones_Aceptar_Click	
limpiarGrpbxSancionesDatos	
cmbx_Reportes_Fuente_SelectedIndexChanged	
cmbx_Reportes_Objeto_SelectedIndexChanged	
txt_Reportes_Objeto_TextChanged	
cmbx_Reportes_Objeto_Razon_SelectedIndexChanged	
cmbx_Reportes_Rango_SelectedIndexChanged	
cmbx_Reportes_Rango_EnabledChanged	
btn_Reportes_Consultar_Click	
btn_Reportes_Super_Backup_Click	
tmr_Hora_Tick	
tbcntrl_Principal_SelectedIndexChanged	

Tarjetas de clase, responsabilidad, colaboración (CRC cards) del Componente Identificador

Ilustración 62. CRC Clase Cliente

Clss_Cliente	
-hayServidor	Clss_Log
-ipAddr	
-archivo	
Main	

Ilustración 63. CRC Clase Log

Clss_Log	
escribir	

Ilustración 64. CRC Clase Servidor

Clss_Servidor	
-data	Clss_Log
-salir	Frm_Principal
-archivo	
-hayServidor	
-principal	
Main	
parsear	
conectar	
desconectar	
atendio	
llamada	
lineaPendiente	
lineaQuitarPendiente	

Ilustración 65. CRC Formulario Acerca De

Frm_Acerca_De	
-padre	Frm_Principal
frm_Acerca_De_Load	
frm_Acerca_De_FormClosing	
btn_OK_Click	

Ilustración 66. CRC Formulario Configuración Puerto

Frm_Configuracion_Puerto	
-frmPropiedadesAbierta	Clss_Log
-padre	Frm_Principal
-archivo	Frm_Propiedades
frn_Configuracion_Puerto_Load	
btn_Configurar_Click	
btn_Cancelar_Click	
btn_Aceptar_Click	
verificarPuerto	
desconectar	
atendio	
llamada	
lineaPendiente	
lineaQuitarPendiente	

Ilustración 67. CRC Formulario Dialogo Asignar

Frm_Dialogo_Asignar	
-fondo	Frm_Principal
-direccion	
-nuevo	
-linea	
-principal	
frm_Dialogo_Asignar_Load	
btn_Cancelar_Click	
btn_Editar_Direccion_Click	
btn_Aceptar_Click	
frm_Dialogo_Asignar_FormClosing	
ingresarNuevo	
modificarDireccion	
cambiarRadioOperador	
guardarHistorial	
pasarHistorial	

Ilustración 68. CRC Formulario Dialogo Eliminación Manual

Frm_Dialogo_Eliminacion_Manual	
-padre	Frm_Principal
frm_Dialogo_Eliminacion_Manual_Load	
cmbx_Razon_Motivo_SelectedIndexChanged	
btn_Atras_Click	
btn_Eliminar_Click	
guardarHistorialBorrados	

Ilustración 69. CRC Formulario Emergencias

Frm_Emergencias	
-padre	Frm_Principal
-dlgtMostrar	
-dlgtBorrarLinea	
-dtgEmergenciasVacio	
frm_Emergencias_Load	
frm_Emergencias_FormClosing	
mnItm_Eliminar_Click	
btn_Agregar_Click	
dtg_Emergencias_UserDeletedRow	
txt_H_TextChanged	
frm_Emergencias_Shown	
frm_Emergencias_DeletingRow	
mostrar	
actualizarColoreadoEmergencias	
colorearEmergencia	
borrar	
registrarEmergencia	

Ilustración 70. CRC Formulario Login

Frm_Login	
-frmLoginExpandida	Frm_Principal
-hiloEscucharServidor	
frm_Login_Load	
btn_Aceptar_Click	
btn_Cancelar_Click	
btn_Opciones_Click	
cargarIp	
Ingersarme	
eliminarSanciones	
contarServiciosPrestados	
ThreadTask	
notificarServidor	

Ilustración 71. CRC Formulario Número Líneas

Frm_Numero_Lineas	
-padre	Frm_Principal
btn_Cancelar_Click	
txt_Numero_Lineas_TextChanged	
btn_Aceptar_Click	

Ilustración 72. CRC Formulario Reporte

Frm_Reporte	
-ReporteServiciosPrestados1	
-ReporteServiciosPrestadosTelefono1	
-ReporteMayorServiciosMoviles1	
-ReporteServiciosPrestadosUnidades1	
-ReporteMayorServiciosOperador1	
-ReporteMayorServiciosTelefono1	
-ReporteMayorServiciosTelefono2	
-ReporteMayorServiciosUnidades1	
-crystlRprtVwr_Reportes	

Ilustración 73. CRC Formulario Propiedades

Frm_Propiedades	
-frmPrincipal	Frm_Principal
-frmConfiguracionPuerto	Frm_Configuracion_Puerto
-nombreArchivo	
frm_Propiedades_Load	
cargarConfiguracion	
btn_Restaurar_Predeterminados_Click	
restaurarPredeterminados	
btn_Aceptar_Click	
guardarConfiguracion	
btn_Cancelar_Click	
btn_Aplicar_Click	
habilitarAplicar	

Ilustración 74. CRC Formulario Principal

Frm_Principal	
-mensajeTiempo	Frm_Emergencias
-dtgMovilVacio	Frm_Dialogo_Asignar
-dtgHistorialVacio	Frm_Configuracion_Puerto
-frmAcercaDeAbierto	Frm_Acerca_De
-frmDialogoAsignarAbierto	Clss_Cliente
-frmEmergenciaAbierto	Frm_Dialogo_Eliminacion_Manual
-estaAsignando	Frm_Emergencias
-cedula	Frm_Reporte
-nombreUsuario	Frm_Numero_Lineas
-serviciosPrestados	
-frmEmergencia	
-emergencias	
-colorEmergencia	
-horaEmergencia	
-descripcionEmergencia	
-anteriorEmergencia	
-seguridadModificarLineas	
-seguridadMostrar	
-seguridadMostrarMoviles	
-seguridadBorrarLinea	

-seguridadBorrarLineaMovil -seguridadActualizarDtgPrincipal -seguridadActualizarDtgMoviles -frmDialogoAsignar -restaurarDatosDtgHistorial -histLinea -histDireccion -histTelefono -histHora -histFecha -histMovil frm_Principal_Load ubicar etiquetar abrirPuerto dibujarLineas frm_Principal_FormClosing borrarConectados notificarSalidaAlServidor tmr_Fecha_Hora_Tick tmr_Emergencias_Tick tmr_Linea_Tick conectarBaseDeDatos mnItm_Log_Emergencias_Click mnItm_Autores_Click tbcntrl_Principal_SelectedIndexChanged dtg_Movil_CellClick dtg_Llamadas_CellDoubleClick eliminarPendientes avisarAsignacion txt_Movil_MouseClick txt_Movil_TextChanged cntxtmnstrp_Llamadas_Opening mnItm_Pendiente_Click avisarPendiente avisarQuitarPendiente mnItm_Borrar_Click btn_Asignar_Click validarMovil guardarHistorial pasarHistorial		

sumarServicios	
btn_Emergencia_Click	
txt_Buscar_Historial_MouseClick	
txt_Buscar_Historial_TextChanged	
btn_Buscar_Historial_Click	
cmbx_Objeto_SelectedIndexChanged	
chkbx_Criterio_Consulta_CheckedChanged	
cmbx_Rango_SelectedIndexChanged	
btn_Consultar_Click	
validarConsulta	
actualizarParametros	
cmbx_Moviles_SelectedIndexChanged	
txt_Buscar_Moviles_TextChanged	
btn_Buscar_Moviles_Click	
mostrarPlanillas	
btn_Adicionar_Planillas_Click	
srlprt_Llamadas_DataReceived	
registrarEmergencia	
borrarMovilDato	
movilReportado	
obtenerRadioOperador	
mostarMoviles	
actualizarCurrentRowMoviles	
horario	
colorearSancionado	
mostrar	
actualizarColoreadoMoviles	
cargarPendientes	
buscarDireccion	
borrar	
borrarMovil	
borrarMovil	
cargarConfiguracionPuerto	
mnItm_Ocultar_Informacion_De_Sanciones_Click	
mnItm_Numero_De_Lineas_Click	
modificarNumeroLineas	
mnItm_Registrar_Servicio_Click	

ANEXO F: MANUAL DE USUARIO

A continuación se encuentran los manuales de usuario de los tres componentes de software resultantes del proyecto.

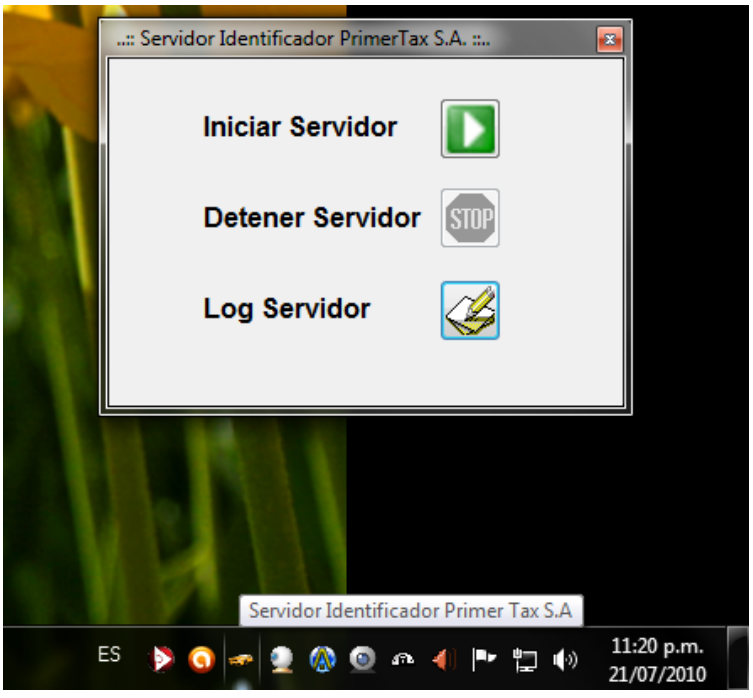
SERVIDOR

Este componente se encarga de crear un puente de comunicación para todo el sistema y principalmente entre los identificadores. Su función principal es la de escuchar por el puerto serial los mensajes que le envía el dispositivo electrónico identificador de llamadas, y comunica a los identificadores la presencia del nuevo número telefónico.

El servidor debe ser instalado en el computador que tenga la conexión, por medio del puerto serial, con el dispositivo electrónico identificador de llamadas. Además en el mismo computador debe estar la base de datos del sistema.

El servidor al iniciar tiene la interface que se muestra en la ilustración 75.

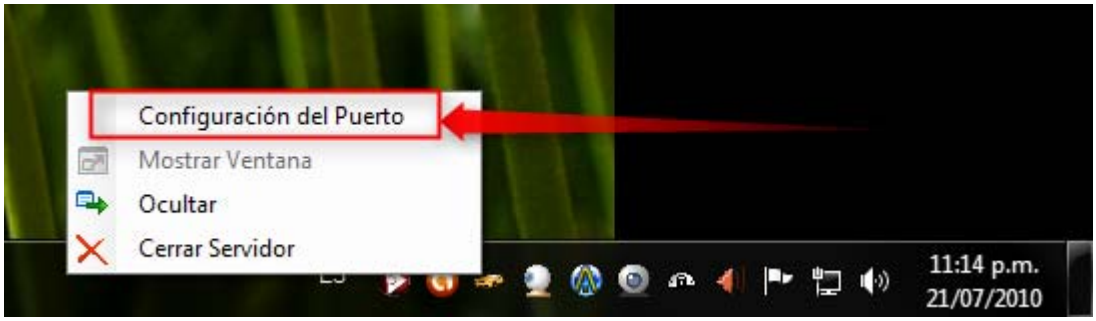
Ilustración 75. Interfaz del Servidor



La interfaz del servidor cuenta con tres botones principales para el control del servicio de informar la llegada de llamadas entrantes. El primero botón es para iniciar el servicio (Iniciar Servidor), el segundo es para detenerlo (Detener Servidor) y el tercero es para realizar auditoria de accesos al servidor por medio de un Log (Log Servidor).

Cuando inicia su ejecución, por defecto es iniciado el servidor, además también se crea un icono en la barra de tareas al lado derecho, al cual al darle clic derecho muestra un menú con el cual podemos configurar el puerto serial, como lo muestra la Ilustración 76, usando la misma forma estándar que usa Windows la configuración de éstos puertos.

Ilustración 76. Ítem para configurar el puerto serial



Desde el mismo menú podemos mostrar u ocultar la ventana como se muestra en las Ilustraciones 77 y 78. Se hace aclaración que el servidor no puede ser cerrado por medio de la X (equis) roja de la ventana para evitar errores de cerrar el servidor por error, en vez de cerrarse se oculta en la barra de tareas y continua con su funcionamiento normal. Se recomienda que se mantenga oculta.

Ilustración 77. Ítem para ocultar la ventana servidor.

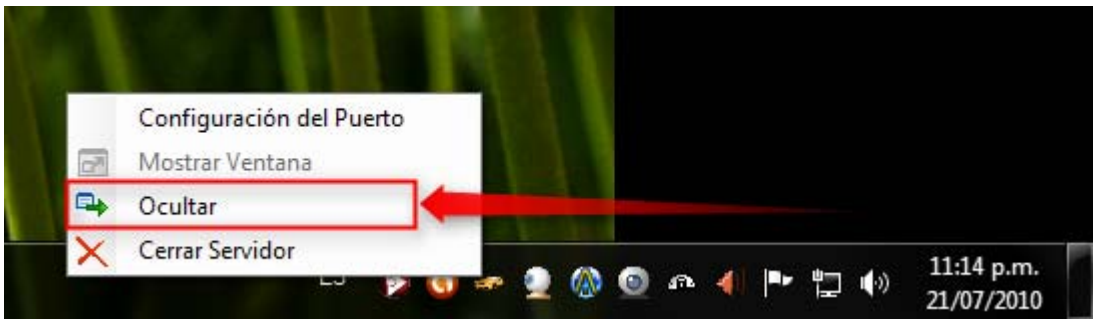
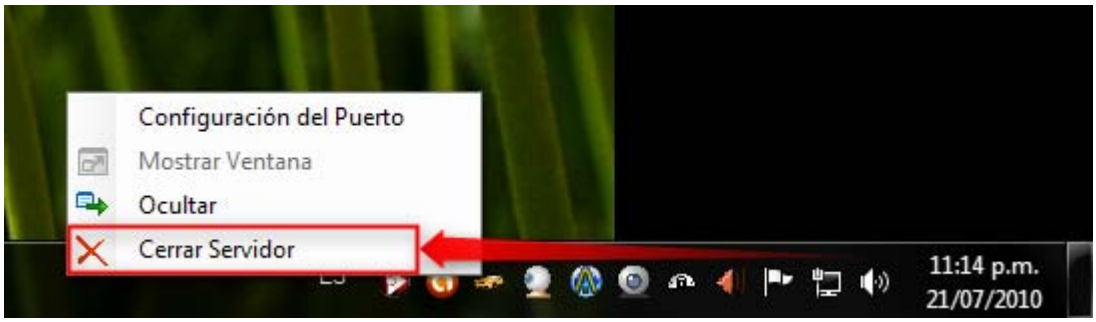


Ilustración 78. Ítem para mostrar la ventana servidor.

Por último se encuentra la opción de cerrar el servidor como lo muestra la Ilustración 79. Se debe tener en cuenta que sin el servidor el sistema no estará funcionando correctamente y que además no identificará ninguna llamada entrante.

Ilustración 79. Ítem para cerrar el servidor.



Se recomienda que la manipulación del servidor este a cargo de solamente una persona, la cual tiene la responsabilidad de mantener el servicio del servidor siempre disponible.

ADMINISTRADOR

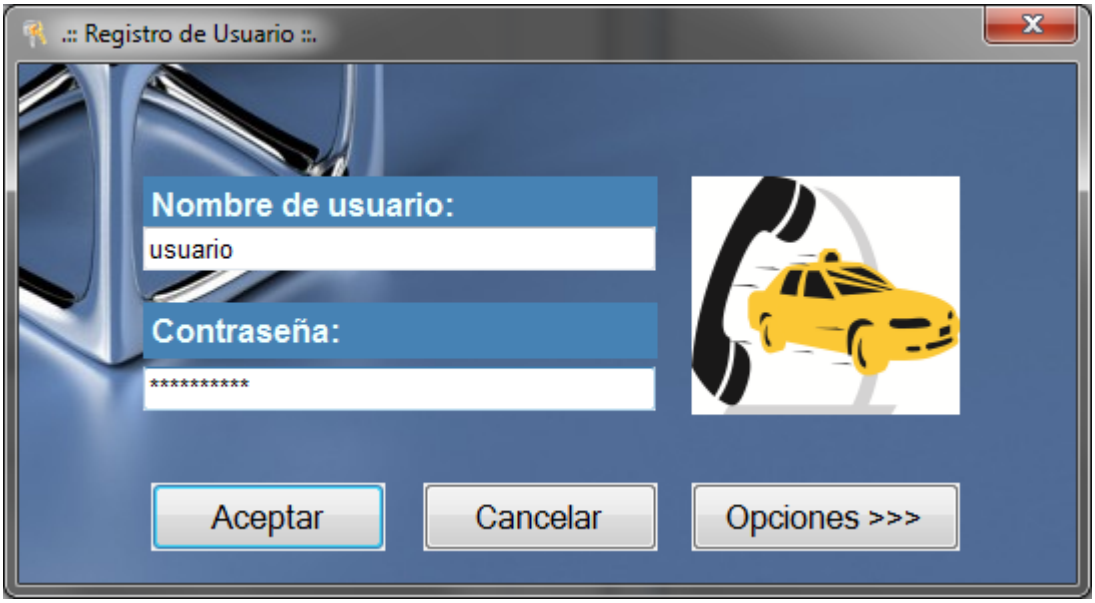
Este componente se encarga de la creación, modificación y eliminación de Usuarios, Móviles Registros, Sanciones y Reportes del sistema.

El servidor puede estar instalado en cualquier computador que tenga acceso por medio de una red al servidor.

La administración del sistema debe ser asignada a pocas personas, para minimizar el ingreso de datos erróneos y canalizar los posibles problemas de inconsistencias a un grupo pequeño de usuarios.

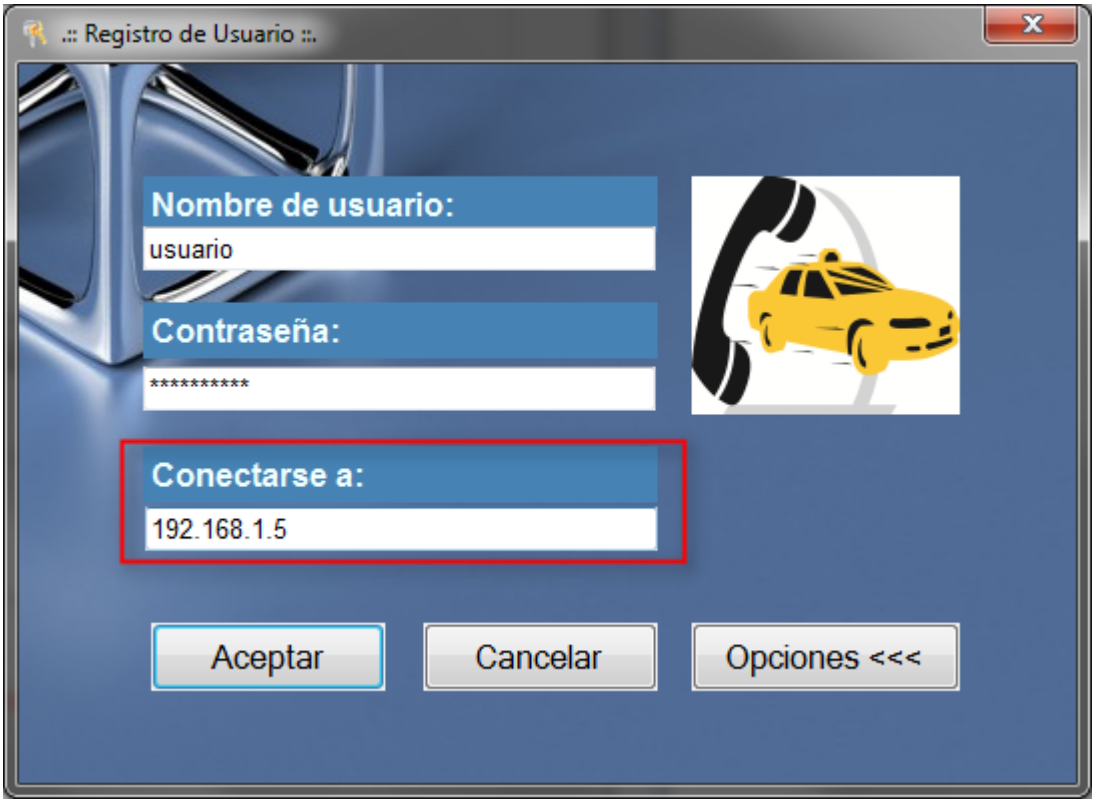
El administrador al iniciar muestra una ventana de validación para su acceso, como se muestra en la ilustración 80.

Ilustración 80. Validación para el Administrador



En su primera ejecución se debe ingresar el nombre de red de la máquina en donde se encuentra para que pueda conectarse a la base de datos, como se muestra en la ilustración 81. Recuerde revisar si la base de datos es accesible y correctamente configurada.

Ilustración 81. Configurar el lugar de la base de datos

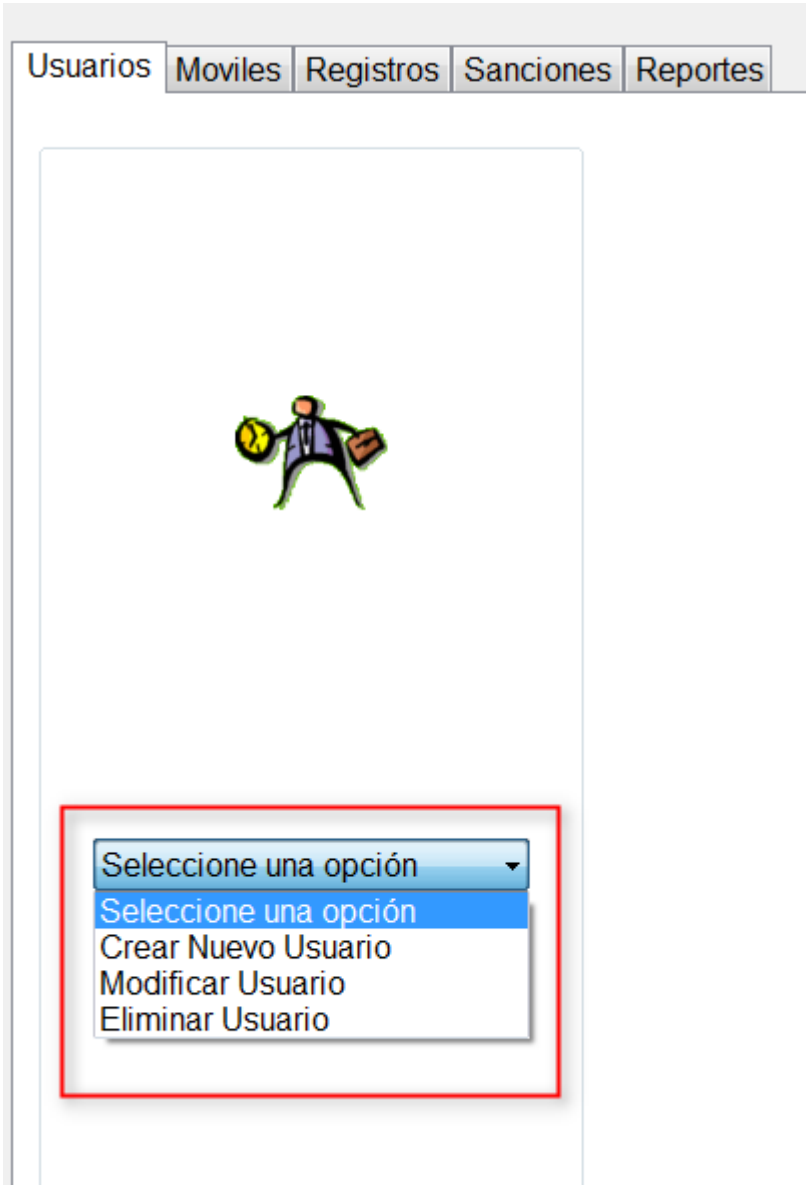


La interfaz de Administrador se encuentra dividida por secciones, usando la distribución de pestañas, dentro de las cuales se realizan todas las acciones que el sistema permite con respecto a cada sección. Las secciones son: Usuarios, Móviles, Registros, Sanciones y Reportes, como lo muestra la Ilustración 82.

La sección de Usuarios permite la administración de usuarios que accederán a los componentes Administrador e Identificador, asignándoles la información necesaria para ello.

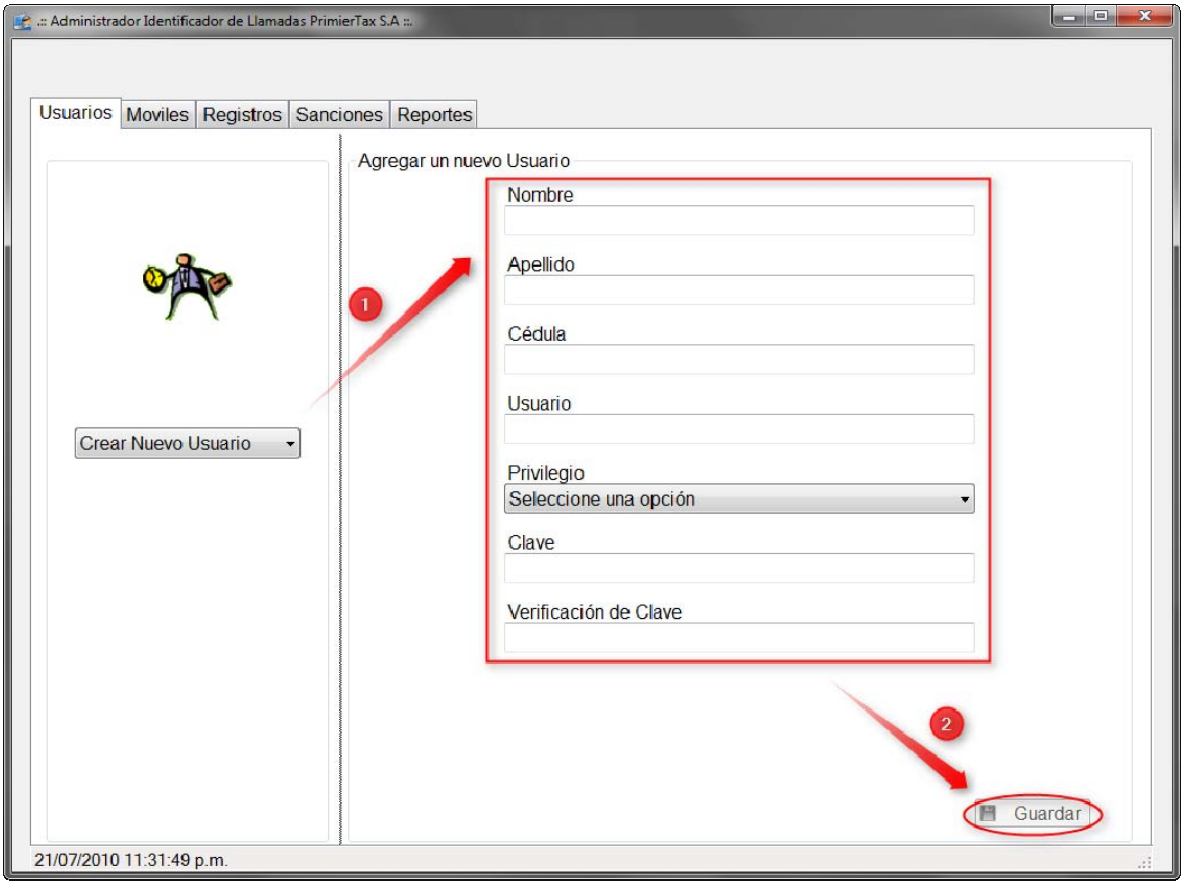
En la pestaña Usuarios, en la parte izquierda se encuentra el menú de opciones posibles: Crear Nuevo Usuario, Modificar Usuario y Eliminar Usuario, los cuales son para el fin que cada nombre expresa, como se muestra en la Ilustración 82.

Ilustración 82. Pestaña Usuarios, Menú de Opciones



Para crear un usuario se da clic en la opción “Crear Nuevo Usuario” lo cual genera que se cree un formulario con la información necesaria para su creación: Nombre, Apellido, Cédula, Usuario, Privilegio, Clave y Verificación de la Clave, como lo muestra la Ilustración 83. Por último se hace clic en el botón guardar para que las modificaciones queden en la base de datos y sea vistas por el sistema.

Ilustración 83. Pestaña Usuarios, Crear Nuevo Usuario



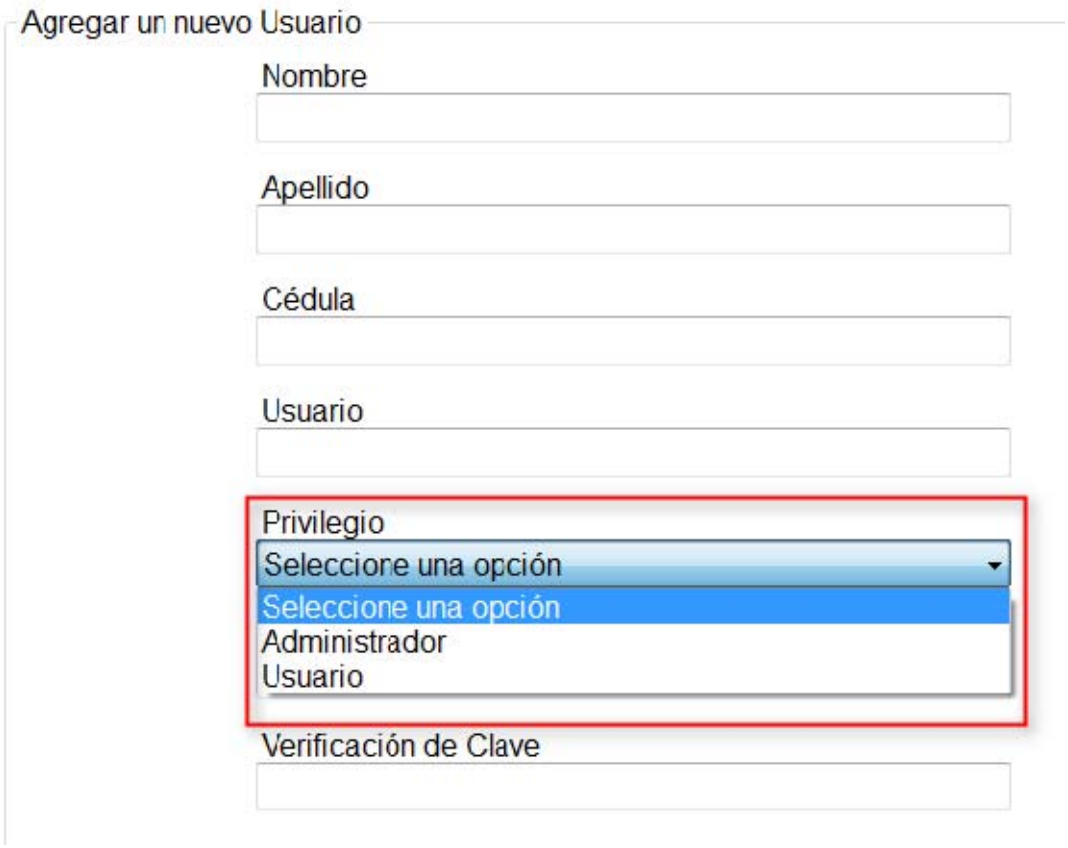
Los campos Nombre, Apellido y Cédula deben ser llenados con los datos reales de la persona al que se está ingresando.

El campo Usuario es el usuario que va a ser usado para entrar al sistema junto con el campo Clave.

El campo Privilegio es para diferenciar a que tiene acceso el usuario, como lo muestra la Ilustración 84, los tipos de privilegios son:

- Administrador: Tiene acceso tanto al componente Identificador y al componente administrador.
- Usuario: Solo tiene acceso al componente Identificador.

Ilustración 84. Pestaña Usuarios, Privilegios



Para realizar a una modificación o eliminación de un usuario, Ilustraciones 85 y 86 respectivamente, se debe buscar al usuario digitando la cedula y dando clic en el botón Buscar, lo cual muestra el formulario con la información del usuario consultado. Por último se hace el cambio que se necesite, sea modificar los datos o eliminar por completo el usuario del sistema.

Ilustración 85. Pestaña Usuarios, Modificar Usuario

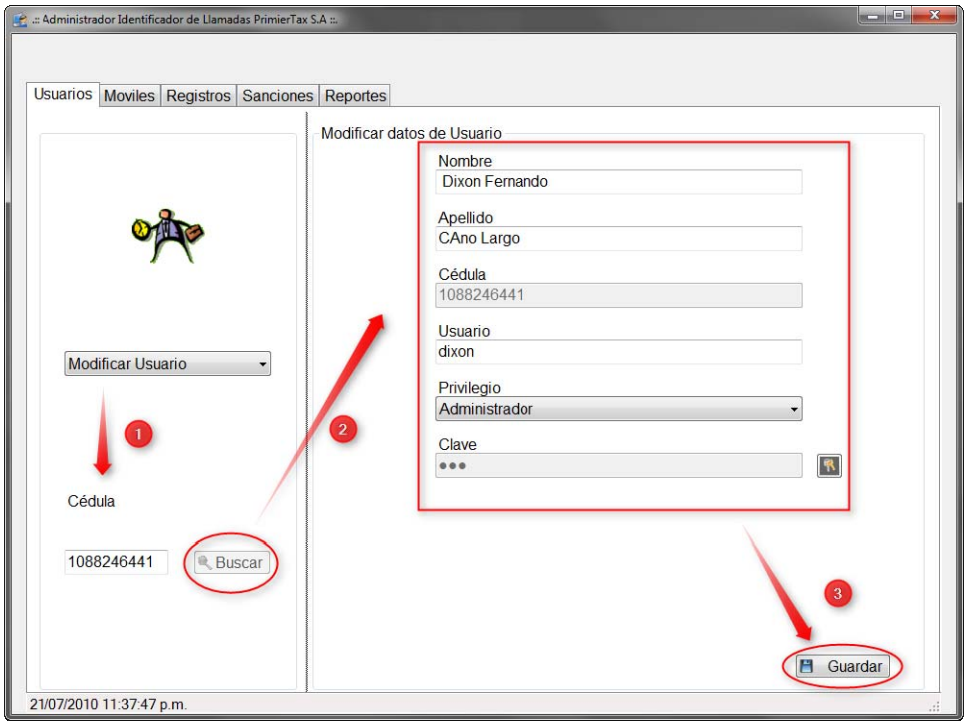
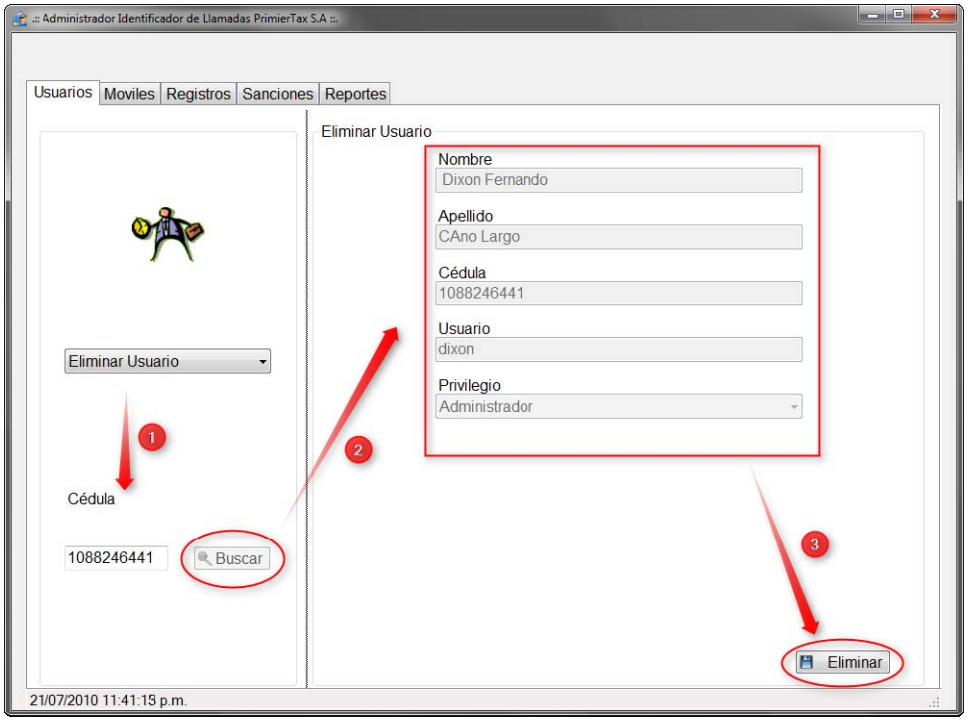


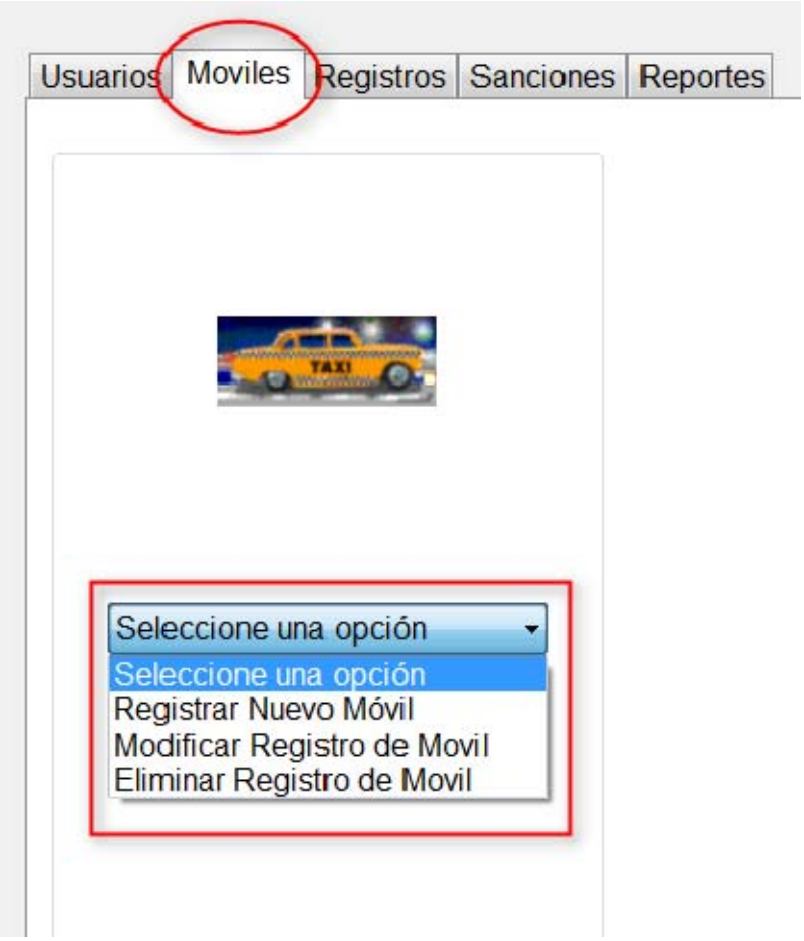
Ilustración 86. Pestaña Usuarios, Eliminar Usuario



La sección de Móviles permite la administración de móviles o taxis, los cuales serán identificados por el sistema y así poder ser asignados a un servicio despachado desde el componente Identificador.

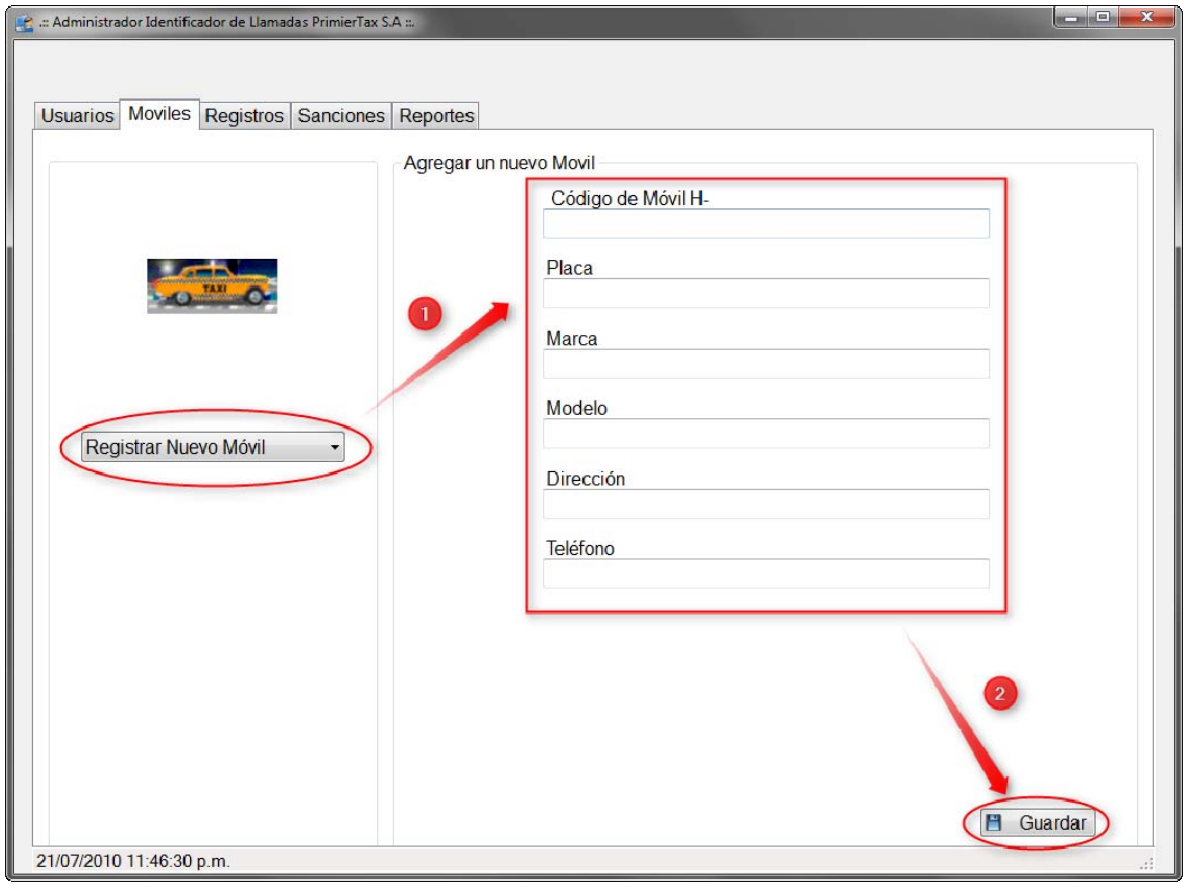
En la pestaña Móviles, en la parte izquierda se encuentra el menú de opciones posibles: Registrar Nuevo Móvil, Modificar Registro de Móvil y Eliminar Registro de Móvil, los cuales son para el fin que cada nombre expresa, como se muestra en la Ilustración 87.

Ilustración 87. Pestaña Móviles, Menú de Opciones



Para registrar un móvil se da clic en la opción “Registrar Nuevo Móvil” lo cual genera que se cree un formulario con la información necesaria para su registro: Código de Móvil también llamado Número Interno o Lateral, Placa, Marca, Modelo y a manera de información: Dirección y Teléfono del Propietario, como lo muestra la Ilustración 88. Por último se hace clic en el botón guardar para que las modificaciones queden en la base de datos y sea vistas por el sistema.

Ilustración 88. Pestaña Móviles, Registrar Nuevo Usuario



Para realizar a una modificación o eliminación de un registro de móvil, Ilustraciones 89 y 90 respectivamente, se debe buscar al móvil digitando el código del móvil (lateral) y dando clic en el botón Buscar, lo cual muestra el formulario con la información del Móvil consultado. Por último se hace el cambio que se necesite, sea modificar los datos o eliminar por completo el móvil del sistema.

Ilustración 89. Pestaña Usuarios, Modificar Usuario

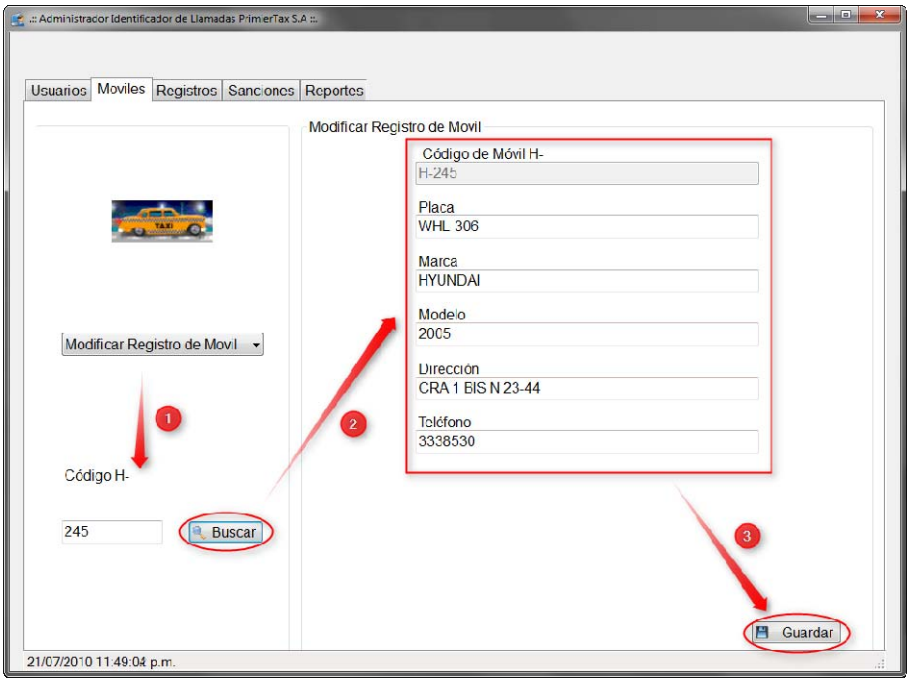
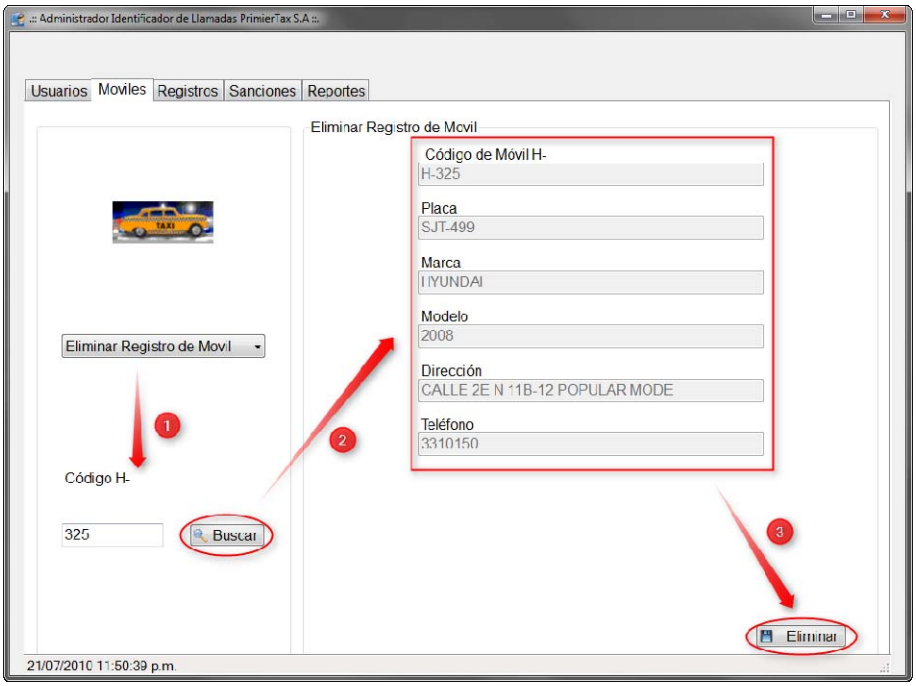


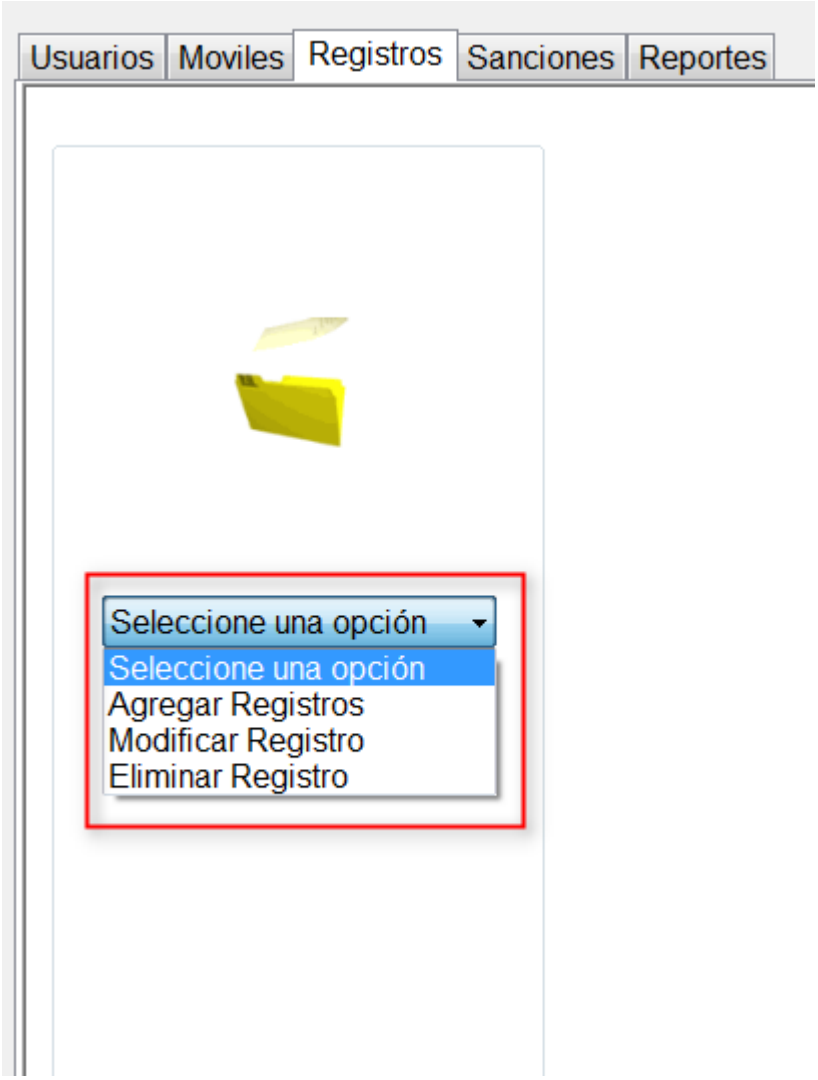
Ilustración 90. Pestaña Usuarios, Eliminar Usuario



La sección de Registros permite la administración del registro de servicios prestados. Su uso es para cuando se necesita adicionar registros que no pudieron ser registrados por el componente Identificador por cualquier inconveniente generado, también para modificar y eliminar datos erróneos. Se recomienda el mantenimiento de los registros de los servicios prestados para tener una mejor estadística más exacta.

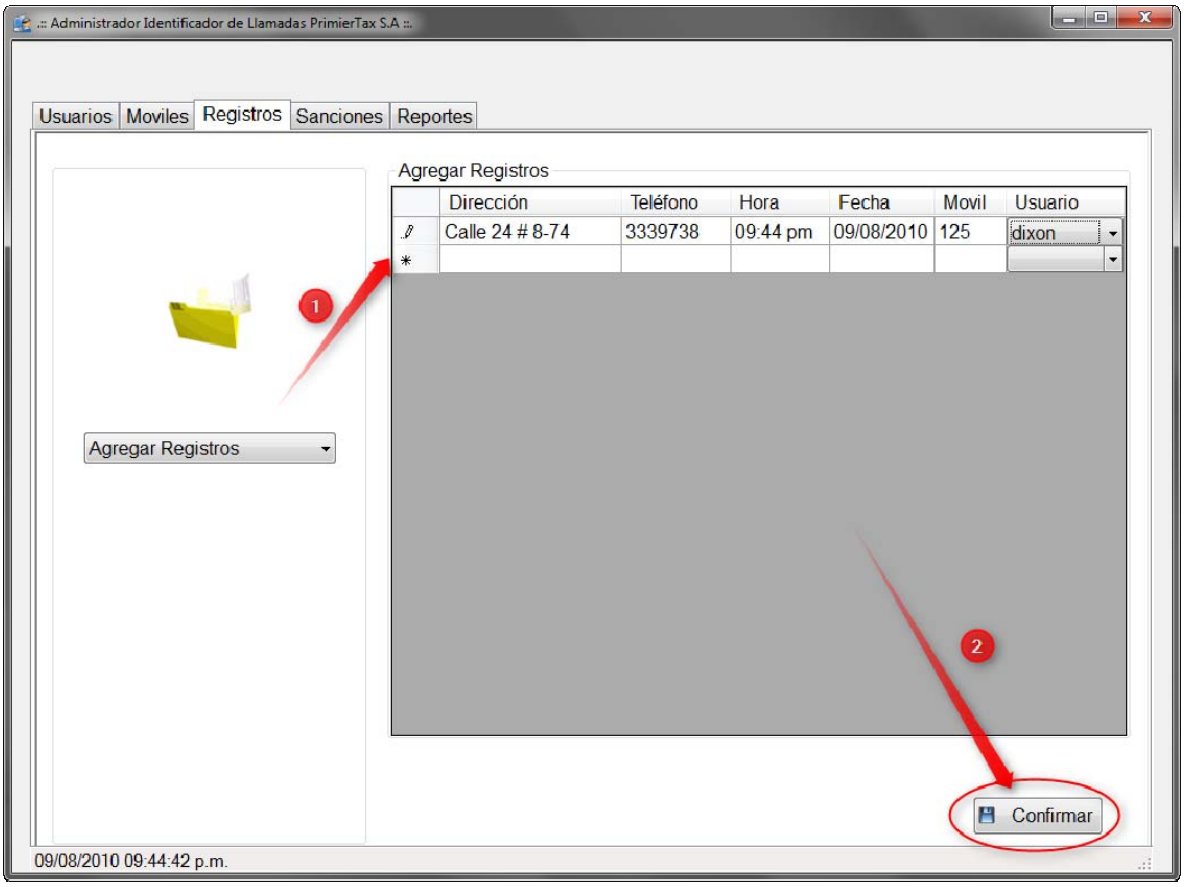
En la pestaña Registros, en la parte izquierda se encuentra el menú de opciones posibles: Agregar Registros, Modificar Registros y Eliminar Registro, los cuales son para el fin que cada nombre expresa, como se muestra en la Ilustración 91.

Ilustración 91. Pestaña Registros, Menú de Opciones



Para adicionar registros de servicios prestados cuando el sistema estuvo fuera de servicio, se da clic en la opción “Agregar Registros” lo cual genera que se cree una tabla con la información necesaria para su registro: Dirección, Teléfono, Hora, Fecha, Móvil y Usuario, como lo muestra la Ilustración 92. Por último se hace clic en el botón guardar para que las modificaciones queden en la base de datos y sean válidas para el sistema. El campo Usuario es una lista desplegable con los nombres de los posibles usuarios del sistema.

Ilustración 92. Pestaña Registros, Agregar Registros



Para realizar a una modificación o eliminación de un registro de servicio prestado, Ilustraciones 93 y 94 respectivamente, se debe buscar por el teléfono al cual fue prestado el servicio y además se da la posibilidad de buscarlo en una fecha específica colocando el cuadro de chequeo activo (con el chulo) y si es desactivado se busca todos los registros que tengan el teléfono y dando clic en el botón Buscar, lo cual muestra la tabla con la información de los servicios prestados al teléfono consultado. Por último se hace el cambio que se necesite, sea modificar los datos o eliminar por completo el móvil del sistema.

Ilustración 93. Pestaña Registros, Modificar Registro

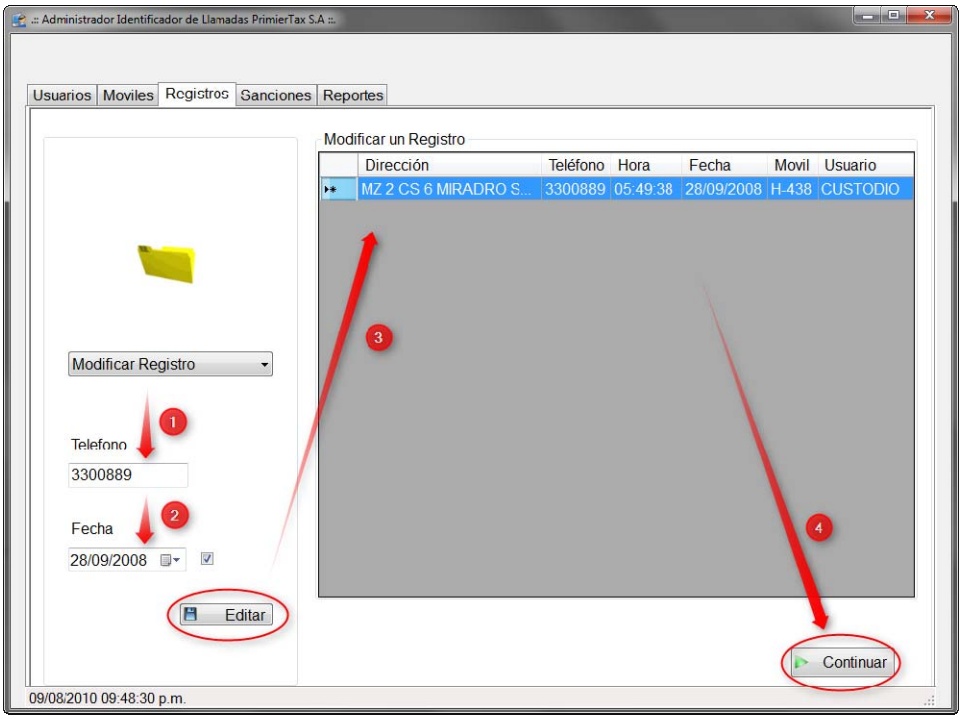
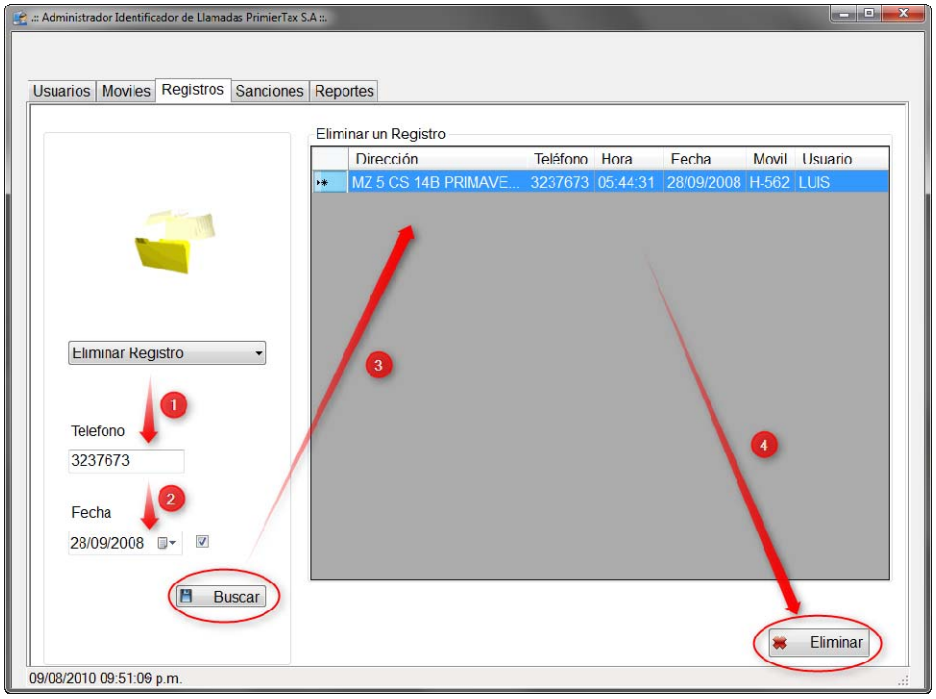


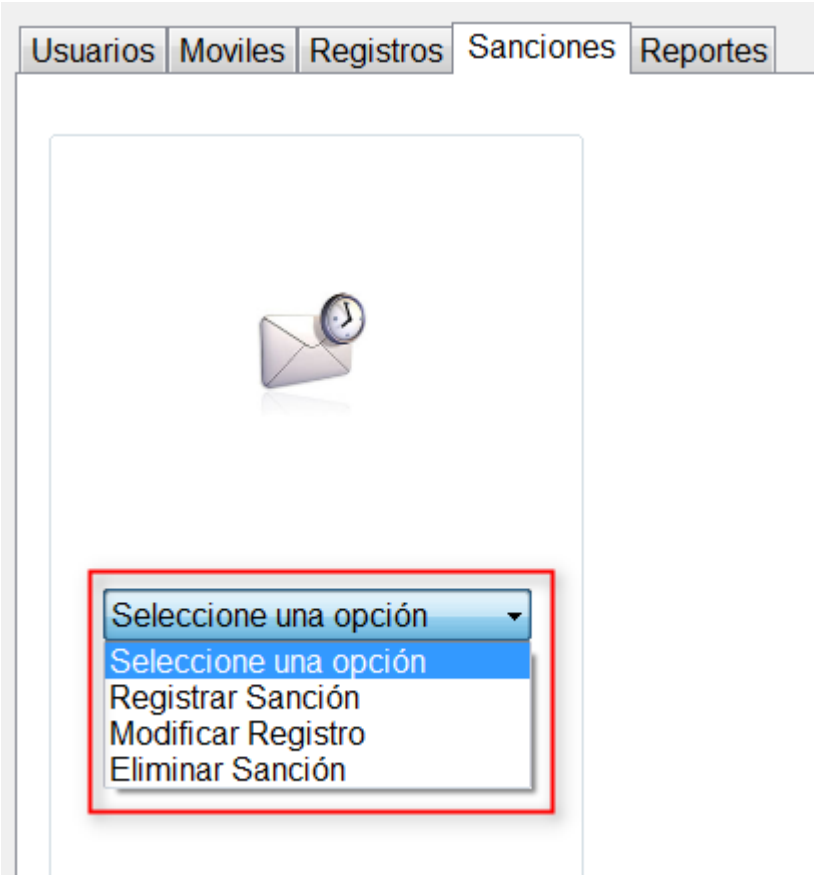
Ilustración 94. Pestaña Registros, Eliminar Registro



La sección de Sanciones permite la administración de las sanciones impuestas a los móviles para que al momento de reportarse para un servicio sea diferenciado para que el operador del sistema pueda saber que a ese móvil no se le puede asignar el servicio.

En la pestaña Sanciones, en la parte izquierda se encuentra el menú de opciones posibles: Registrar Sanción, Modificar Registro y Eliminar Sanción, los cuales son para el fin que cada nombre expresa, como se muestra en la Ilustración 95.

Ilustración 95. Pestaña Sanciones, Menú de Opciones

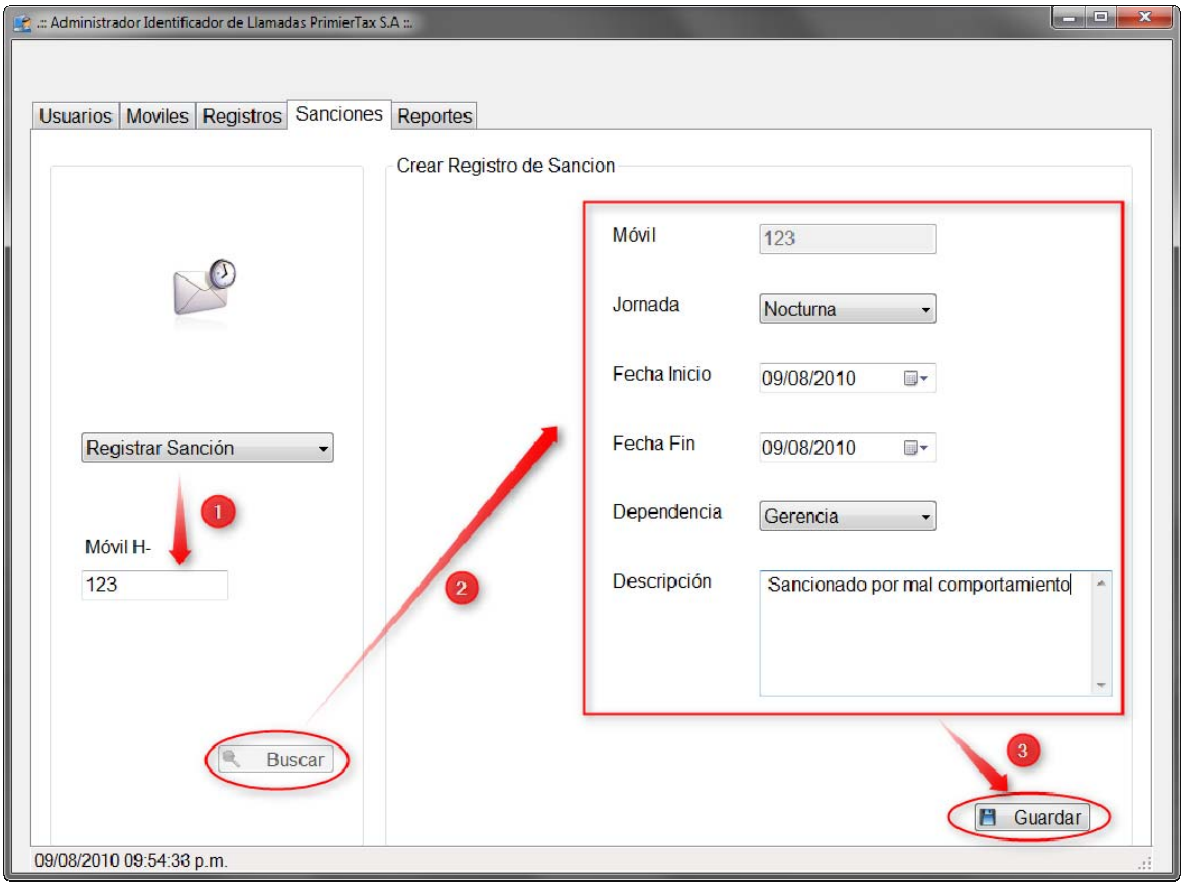


Para adicionar una sanción a un móvil se da clic en la opción “Registrar Sanción” lo cual genera hace que se muestre el campo Móvil para ingresar el número del lateral al cual se le va a registrar la sanción, y luego se da clic en el botón buscar para que se cree un formulario con la información necesaria para su registro: Móvil o número de lateral, Jornada, Fecha Inicio, Fecha Fin, Dependencia y Descripción, como lo muestra la Ilustración 96. Por último se hace clic en el botón guardar para que las modificaciones queden en la base de datos y sean válidas para el sistema.

El campo Jornada es una lista desplegable con las opciones: Diurna y Nocturna, que especifica la jornada en la cual el móvil se encontrará sancionado.

El campo Dependencia es una lista desplegable con las opciones: Gerencia, Junta Vigilancia y Oficina. Este campo representa la dependencia que le ha levantado la sanción a dicho móvil, para que en el componente Identificador, cuando el móvil se reporte, no solo sea identificado como sancionado sino que también se pueda saber por qué dependencia fue sancionado para que se le sea informado al móvil a donde se debe dirigir

Ilustración 96. Pestaña Sanciones, Registrar Sanción



Para realizar a una modificación o eliminación de una sanción, Ilustraciones 97 y 98 respectivamente, se debe buscar por medio del número del lateral y la jornada de sanción, lo cual, si encuentra la sanción, muestra un formulario con la información de la sanción. Por último se hace el cambio que se necesite, sea modificar los datos o eliminar por completo la sanción del sistema.

Ilustración 97. Pestaña Sanciones, Modificar Registro

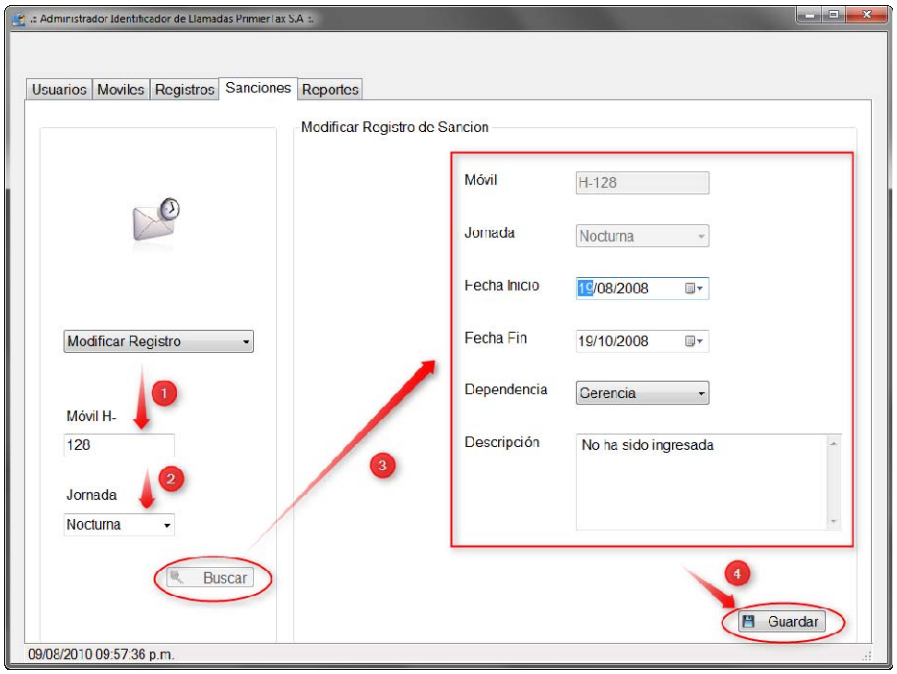
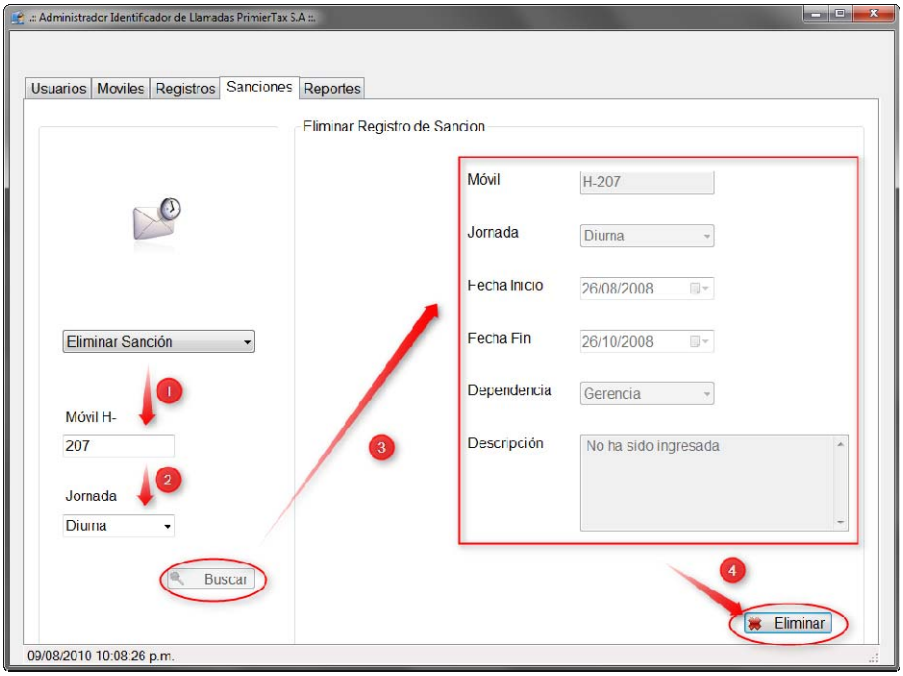


Ilustración 98. Pestaña Sanciones, Eliminar Sanción



La sección de Reportes permite la generación de reportes dinámicos mediante la selección de criterios de filtro.

Como se muestra en la Ilustración 99, los pasos para la generación de un reporte son:

1. Seleccionar la fuente del reporte: Los cuales pueden ser:
    a. Registro Llamadas Perdidas
    b. Registro Llamadas Eliminadas
    c. Rendimiento

2. Seleccionar el objeto del reporte: El cual depende de cada fuente del reporte:
    a. Para “Registro Llamadas Perdidas”:
        - Cliente: Se debe ingresar el teléfono del cliente.
        - Operador: Se debe ingresar la cédula del operador.
        - Todos
    b. Para “Registro Llamadas Eliminadas”:
        - Cliente: Se debe ingresar el teléfono del cliente.
        - Operador: Se debe ingresar la cédula del operador.
        - Razón: Se debe seleccionar una de las siguientes:
            - Llamada de servicio al cliente.
            - Llamada equivocada.
            - Cliente Ausente.
            - Llamada de mensaje.
            - Otro...
        - Todos
    c. Para “Rendimiento”:
        - Servicio Completo
        - Operador: Se debe ingresar la cédula del operador

3. Seleccionar el rango de fechas a consultar.

4. Dar clic en el botón consultar.

El reporte de “Registro Llamadas Perdidas” muestra las llamadas que los operadores no atendieron después de los 3 minutos de espera para cada llamada, lo cual hace que dicha llamada se considere como “Perdida”. Este reporte se puede generar para saber las la información de un cliente en específico, o para un operador y/o para revisar todas las llamadas perdidas.

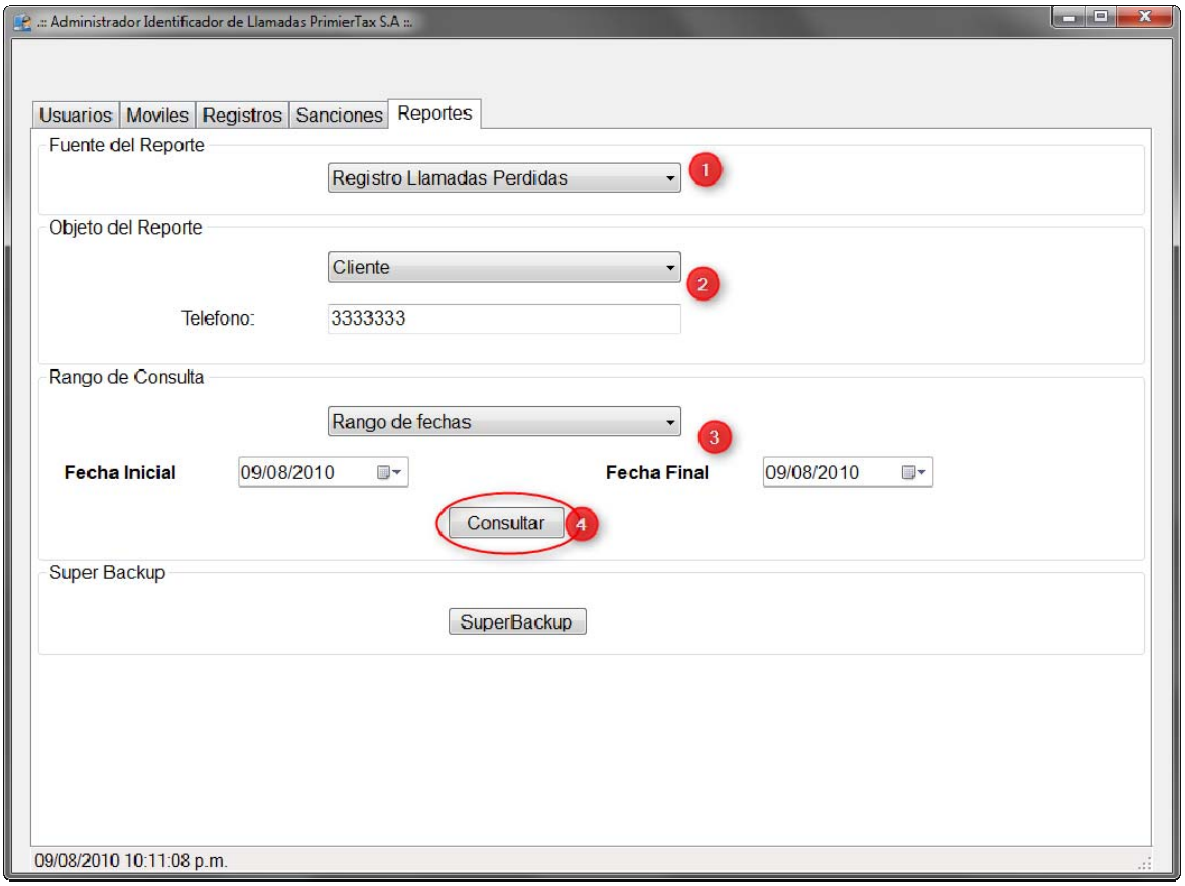
El reporte de “Registro Llamadas Eliminadas” muestra las llamadas que fueron eliminadas manualmente por los operadores, las cuales deben tener su correspondiente razón por la cual fue eliminada. Este reporte puede ser generado

por Cliente para conocer cuáles y cuantas llamadas han sido borradas para él, por Operador para saber cuáles y cuantas llamadas ha borrado un operador en específico, por Razón para saber cuáles y cuantas llamadas han sido borradas por una determinada razón y por Todos que saca el reporte completo de todas las eliminadas.

El reporte de “Rendimiento” genera un reporte consolidado con la información de las llamadas perdidas, las eliminadas y las atendidas. Este reporte se puede realizar por Servicio Completo para conocer en la totalidad del sistema cuales y cuantas llamadas han sido perdidas, eliminadas y atendidas dentro de un mismo reporte. También puede ser realizado por Operador para saber la misma información pero sólo filtrada a un solo operador del sistema.

Estos reportes son acotados por un rango de fechas para facilitar su estudio posterior.

Ilustración 99. Pestaña Reportes, Proceso de creación de un reporte.



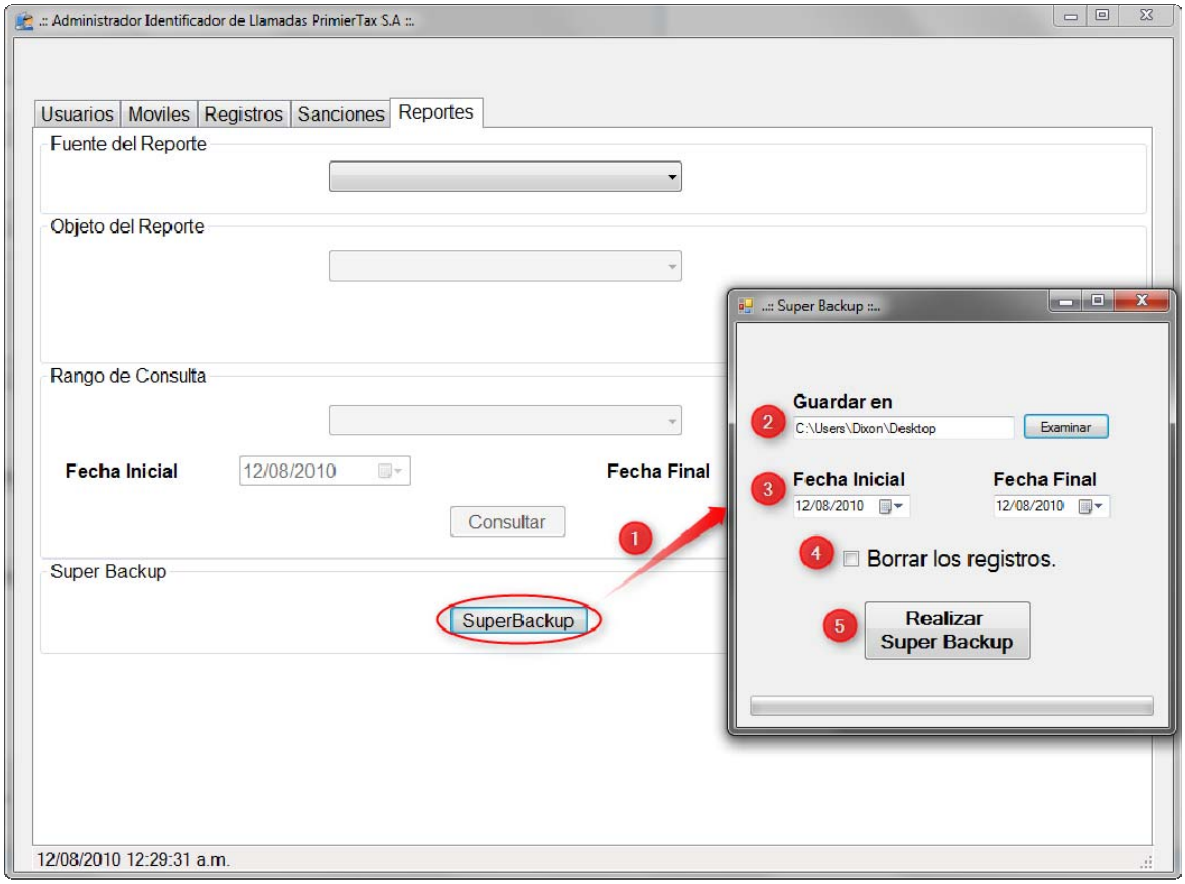
Dentro de la sección de Reportes se encuentra la opción para generar el Super Backup, el cual se encarga de generar todos los reportes posibles dentro del sistema y son guardados en el lugar que especifique el usuario.

Como se muestra en la Ilustración 100, los pasos para realizar el super backup son:

1. Dar clic en el botón “SuperBackup”.
2. En la ventana que sale se debe seleccionar el lugar en donde se van a guardar todos los reportes.
3. Se selecciona el rango de fechas a los cuales se van a sacar los datos para los reportes.
4. Se elige si se desea borrar los registros que se le van a hacer el backup.
5. Se da clic en el botón “Realizar Super Backup” para que comience el proceso.

Cuando el proceso finalice se tendrán todos los reportes dentro de la carpeta especificada.

Ilustración 100. Pestaña Reportes, SuperBackup.



IDENTIFICADOR

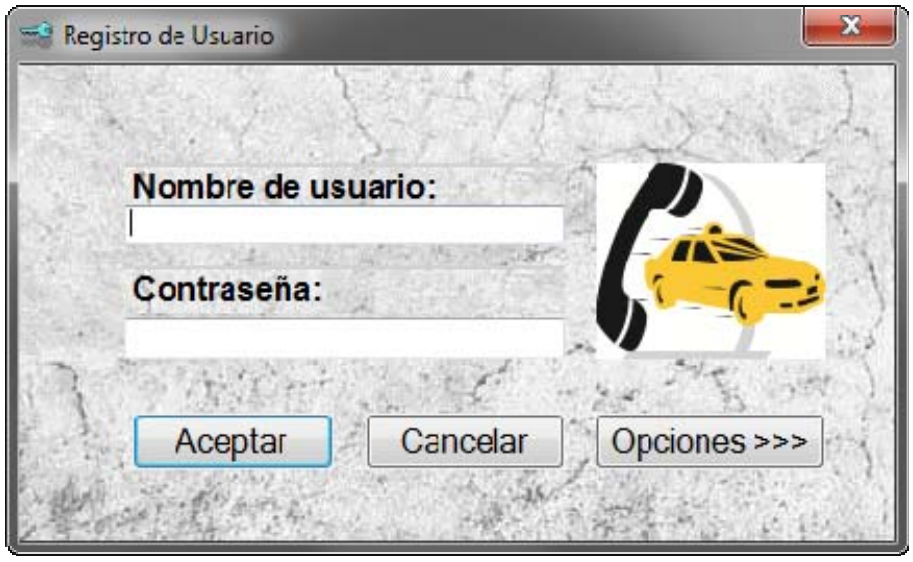
Este componente se encarga de interacción con el operador, ayudándole a realizar el préstamo del servicio de taxis.

El componente se conecta por medio del puerto serial con el dispositivo electrónico identificador de móviles o taxis, de esta forma, cuando un móvil se reporte ante una llamada del operador para prestar un servicio, el Identificador podrá identificarlo y mostrarlo al operador en el orden en que se reportó con respecto a los demás móviles, para ayudar al operador a tomar decisiones de a quien asignarle el servicio solicitado.

Permite la asignación de un servicio con uno o más móviles y su posterior consulta.

El Identificador al iniciar muestra una ventana de validación para su acceso, como se muestra en la ilustración 80.

Ilustración 101. Validación para el Identificador



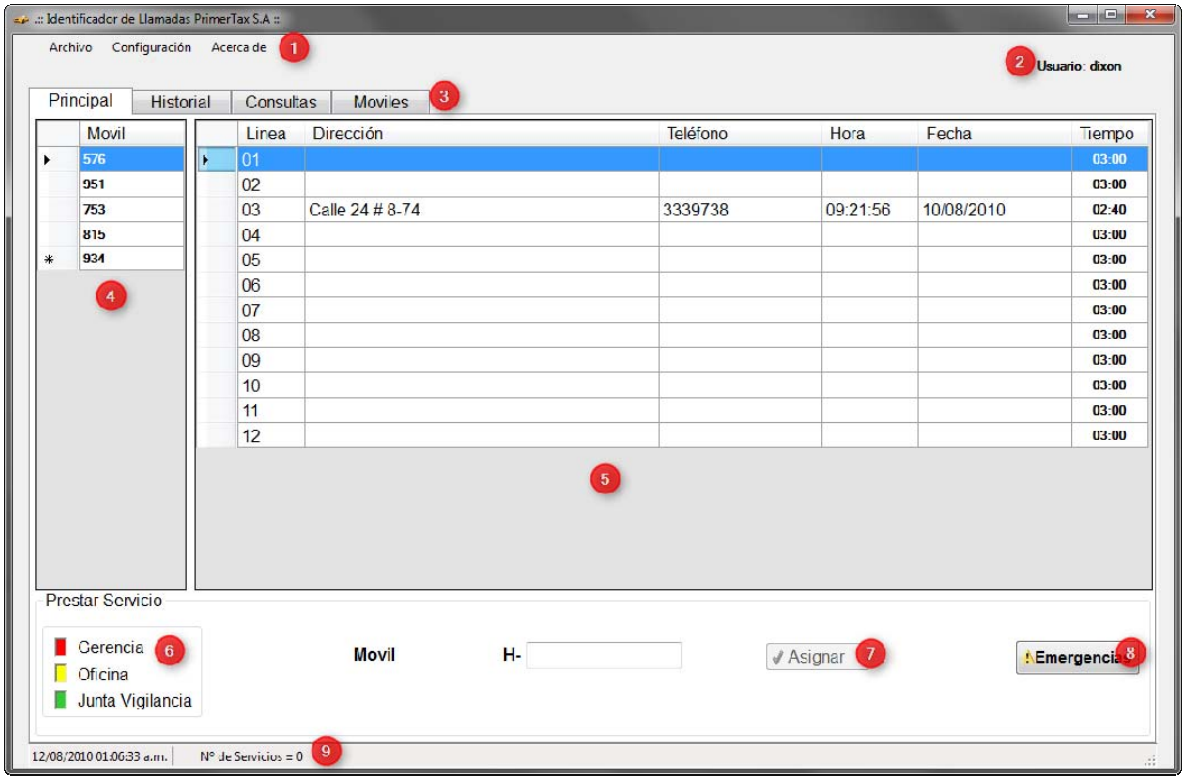
En su primera ejecución se debe ingresar el nombre de red de la máquina en donde se encuentra para que pueda conectarse a la base de datos y al servidor, como se muestra en la ilustración 102. Recuerde revisar si la base de datos es accesible y está correctamente configurada.

Ilustración 102. Configurar el lugar de la base de datos

Como se muestra en la Ilustración 103, la interfaz de Identificador se encuentra dividida de la siguiente forma:

A. Menú superior.
B. Identificación de usuario.
C. Sección central con división por pestañas.
D. Dentro de la pestaña Principal el listado de los móviles reportados.
E. Dentro de la pestaña Principal el listado de las líneas telefónicas con los campos para la información referente a cada llamada.
F. Dentro de la pestaña Principal Información de los colores de cada sanción para los móviles.
G. Dentro de la pestaña Principal asignación de servicios.
H. Dentro de la pestaña Principal Emergencias activas.
I. Número de servicios prestados por el usuario durante ese día.

Ilustración 103. Distribución de la interfaz del Identificador



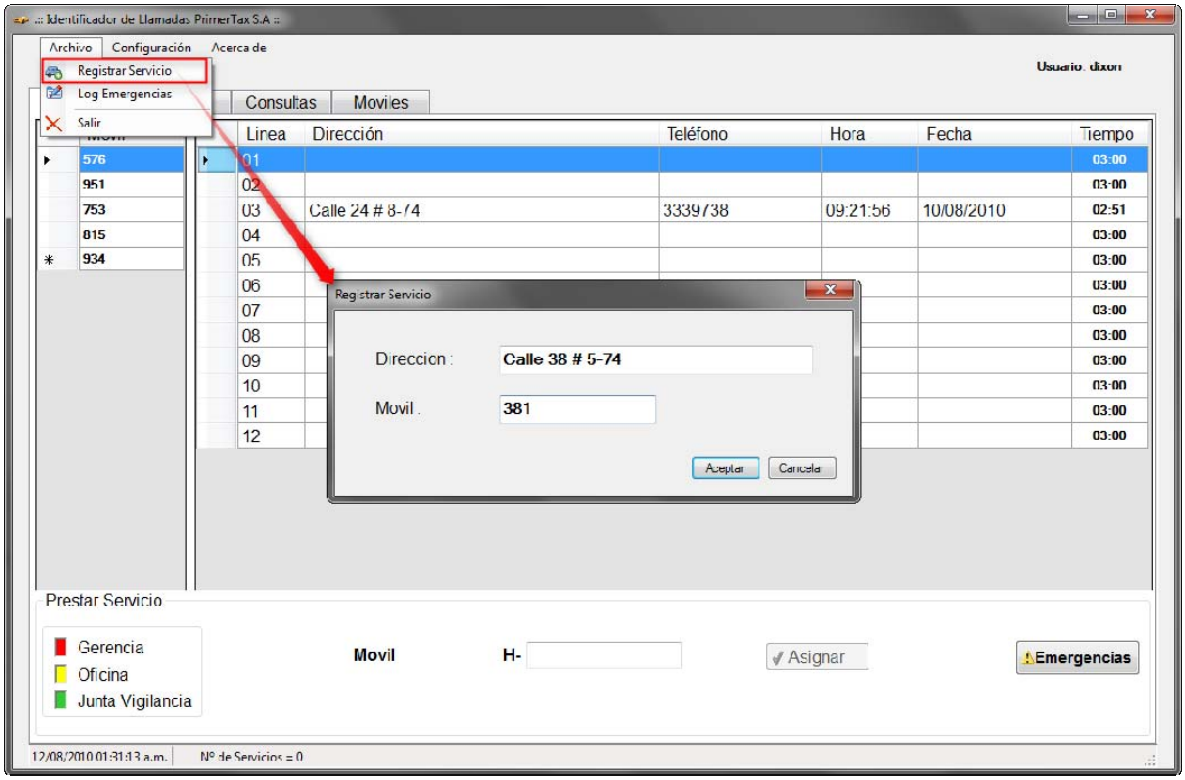
Como se muestra en la Ilustración 103, el menú superior contiene tres bloques de opciones:

a) El primero de ellos es Archivo, dentro del cual contiene las siguientes opciones:

- Registrar Servicio: Existen casos especiales donde el servicio no es pedido por medio del teléfono o también debido a fallas electrónicas el dispositivo no identifique la llamada, para estos y otros casos similares donde el registro de la asignación del servicio no se pueda dar mediante la interfaz normal, se usa esta para no dejar ningún servicio sin ser registrado dentro del sistema.

  Como se muestra en la Ilustración 104, los datos a pedir del servicio son la dirección y el móvil ligados al servicio prestado.

Ilustración 104. Registrar Servicio

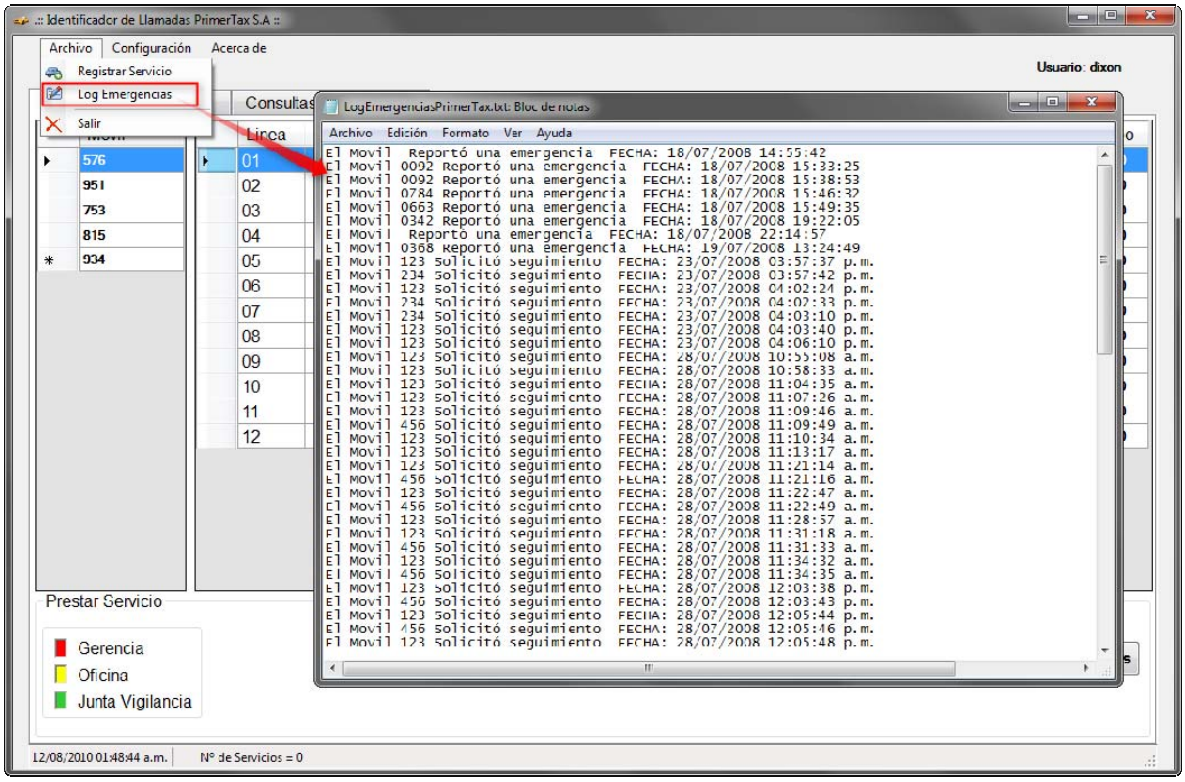


- Log de emergencias: Se mantiene un registro histórico de todas las emergencias reportadas por un móvil y además los seguimientos a los móviles cuando se dirigen a lugares de alta peligrosidad. Para este fin es el Log de Emergencias, el cual se realiza en un archivo plano para su facilitar su manejo.

  Como se muestra en la Ilustración 105, el archivo dice qué móvil reportó la emergencia, la fecha y la hora, además permite que se le adicione texto para anotaciones del operador con respecto a alguna emergencia.

  Se hace la aclaración de que en este Log se encuentran todas las emergencias registradas por el sistema, a diferencia de la ventana de emergencias que mantiene en pantalla las emergencias vigentes.

Ilustración 105. Log de Emergencias



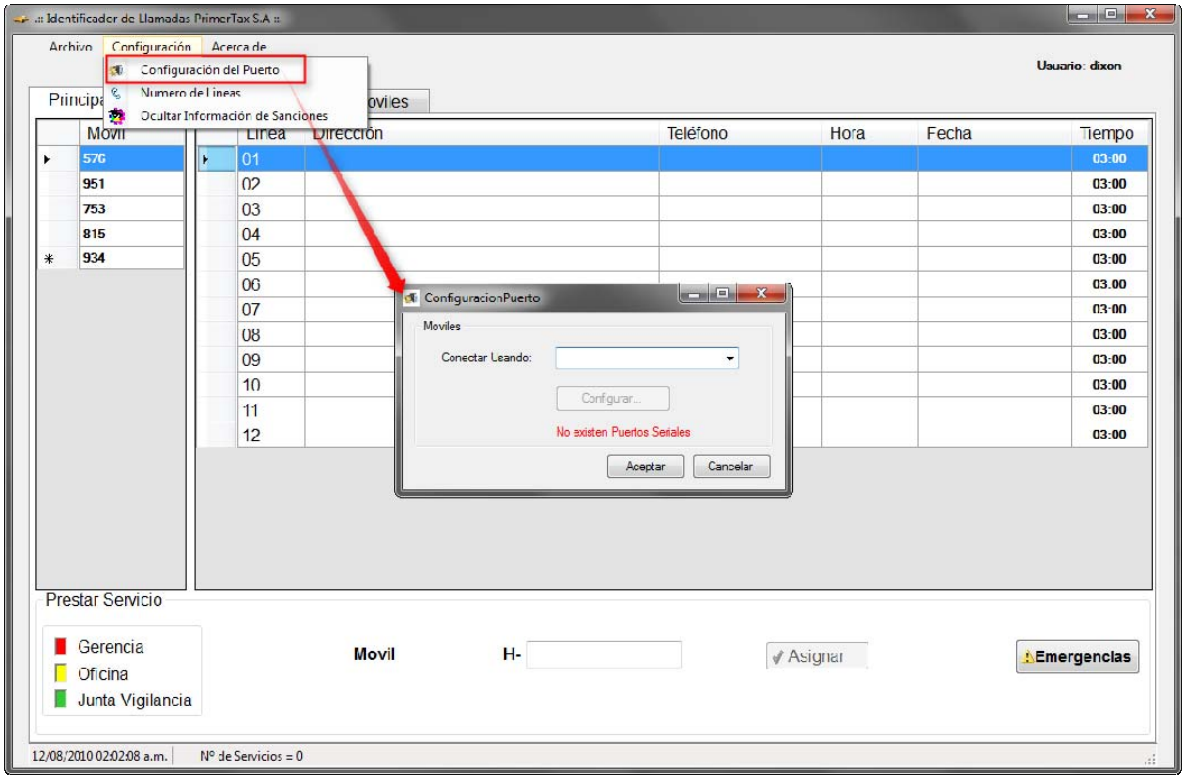
- Salir: Permite salir de la aplicación correctamente.

b) El segundo es Configuración, dentro del cual contiene las siguientes opciones:

- Configuración Puerto: Permite la configuración del puerto serial con el cual se va a tener la comunicación con el dispositivo electrónico identificador de móviles.

  Como se muestra en la Ilustración 106, cuando no existen puertos seriales habilitados, el sistema mostrará el mensaje de “No existen Puertos Seriales” y le permite al Identificador continuar funcionando aunque no identificará a los móviles de manera automática, pero puede ser suplantada por la asignación del móvil por medio del teclado.

Ilustración 106. Configuración del puerto serial



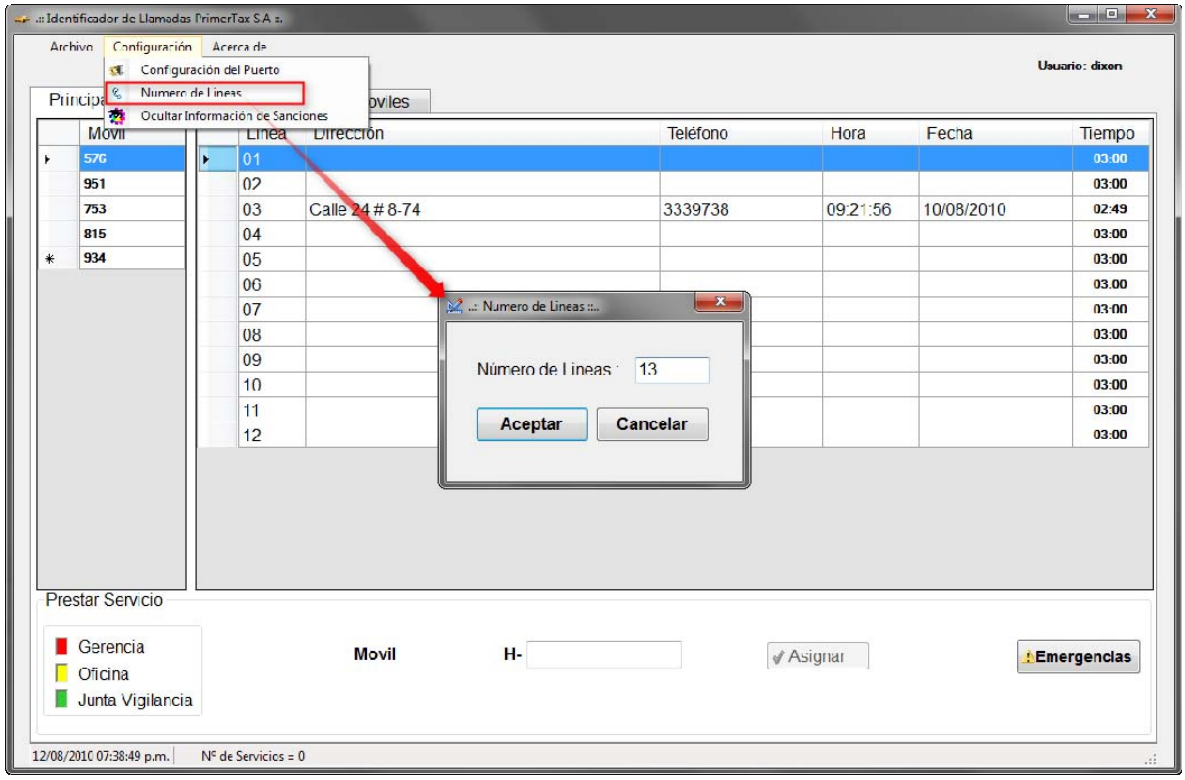
- Número de Líneas: Permite definir el número de líneas que el sistema va a identificar. Se usa cuando se expanda o se contraiga la capacidad de los dispositivos electrónicos de identificación de llamadas, y/o cuando se necesite definir el número de líneas habilitadas para funcionar.

  Como se muestra en la Ilustración 107, solo se debe ingresar el número de líneas y el sistema ya estará listo para aceptar dicho número sin necesidad de reiniciar el sistema.

  Se aclara que el Identificador no se encarga de la comunicación directa con el dispositivo electrónico de identificación de llamadas, quien es el que proporciona el número de la línea por donde está entrando la llamada. El encargado de dicha comunicación es el Servidor.

  El dispositivo electrónico identificador de llamadas debe estar correctamente configurado en su numeración de las líneas, pues de ello depende la consistencia en los datos mostrados al operador en el Identificador.

Ilustración 107. Número de Líneas



- Ocultar/Mostrar Información de Sanciones: Como se muestra en las Ilustraciones 108 y 109, permite ocultar y/o mostrar la Información de explicación de los colores de las sanciones.

  Se recomienda mantenerlo visible mientras se aprende a diferenciar cada tipo de sanción por medio de su color, pues es importante conocer la dependencia que ha generado la sanción para poder informar al conductor hacia donde se debe dirigir para tramitar la eliminación de la misma.

  La identificación por colores permite que al momento de reportarse un móvil para un servicio y el sistema lo identifique de forma automática, sea catalogado como un móvil con por lo menos un problema dentro de la empresa y así poder ejecutar los lineamientos de la empresa de acuerdo a el tipo de sanción.

Ilustración 108. Ocultar Información de Sanciones

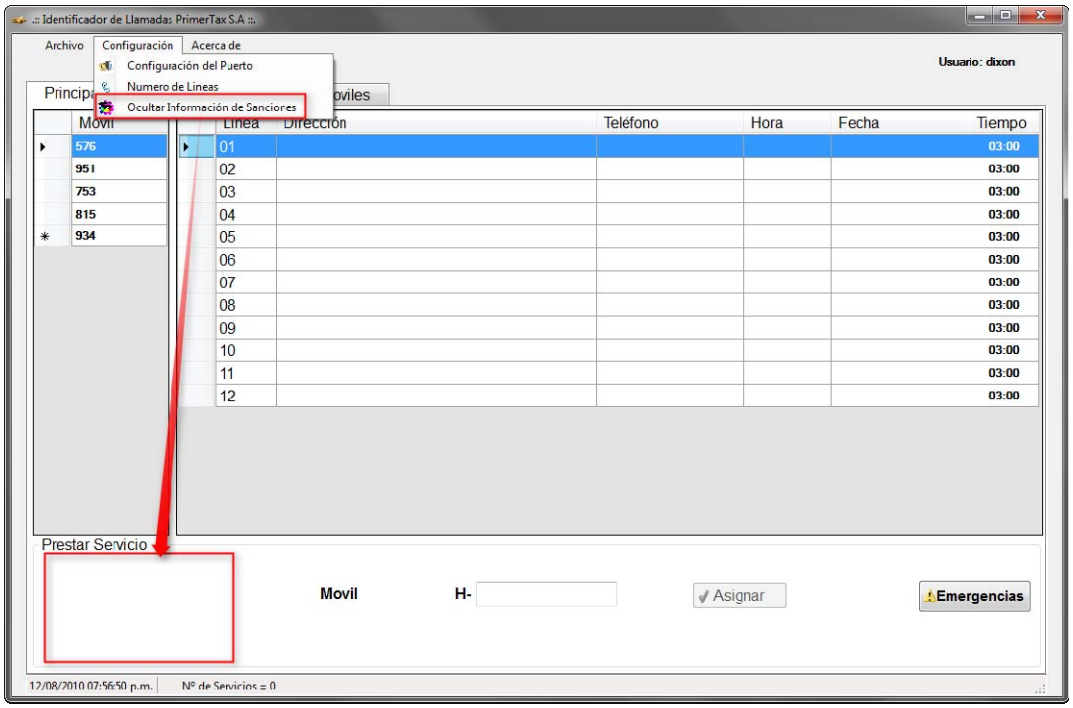
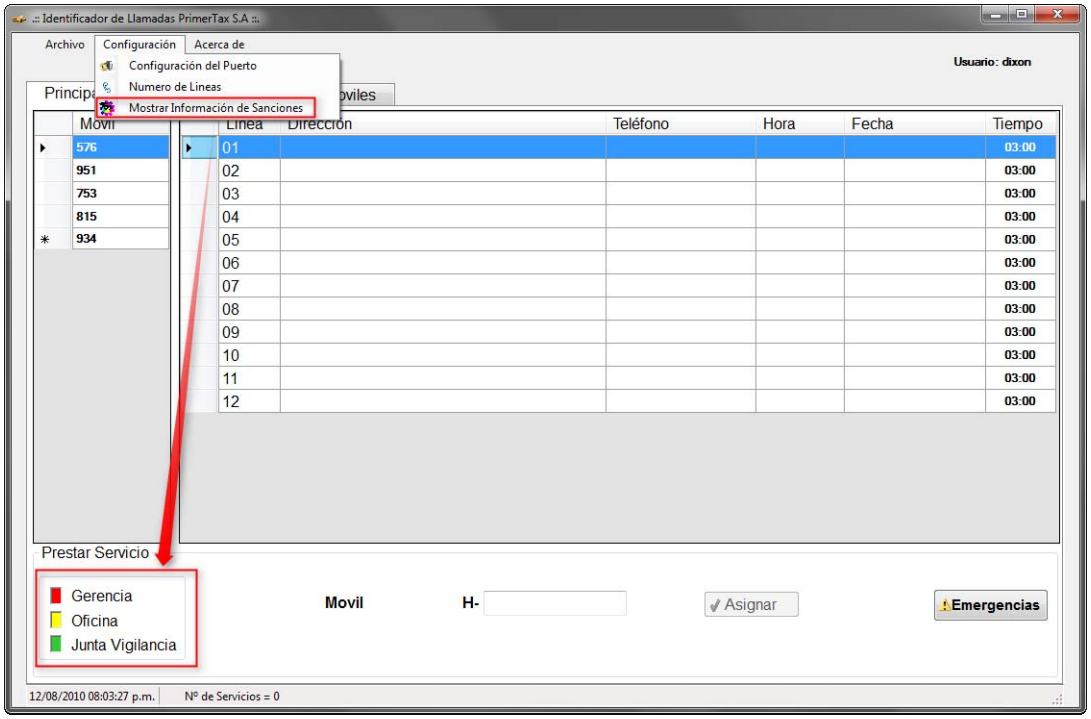


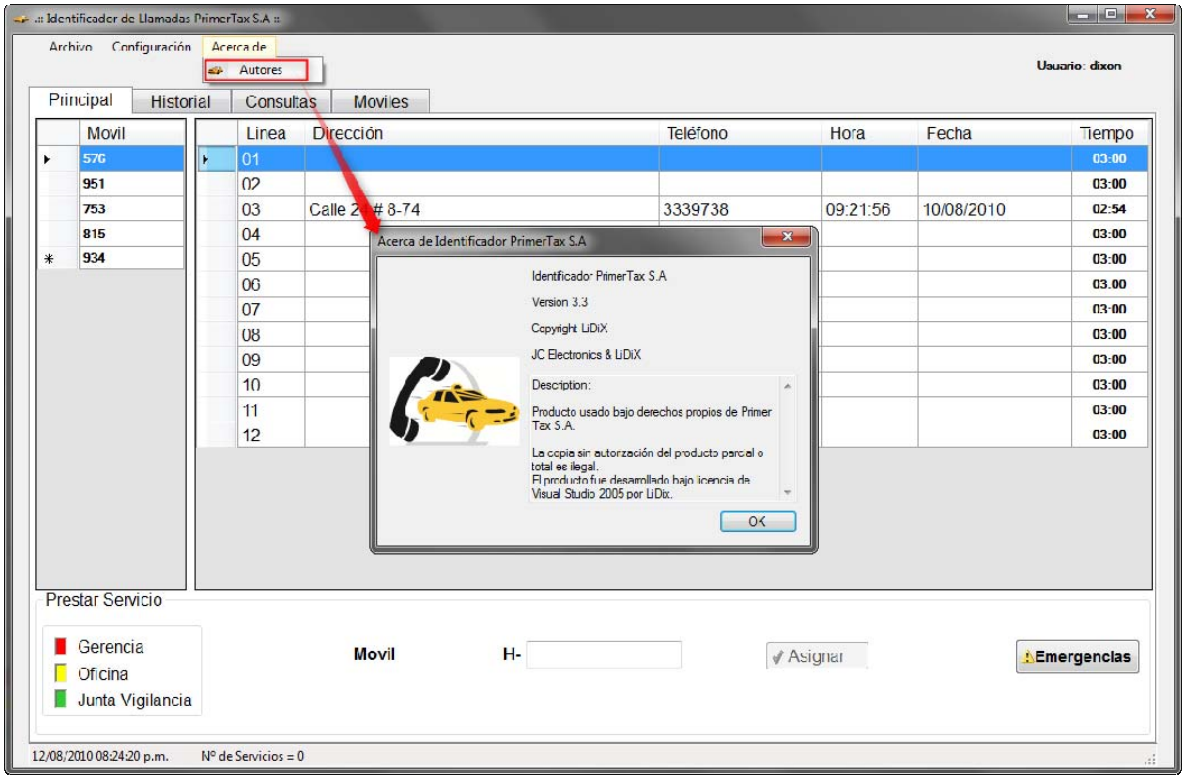
Ilustración 109. Mostrar Información de Sanciones



c) El tercero es Acerca de, en el cual solo hay una opción:

- Autores: Como lo muestra la Ilustración 110, se encuentra las información de los creadores del programa, la versión actual del programa y una pequeña descripción de la licencia.

Ilustración 110. Acerca de los autores



Proceso para la asignación de un servicio

Como se muestra en la Ilustración 111, los pasos para realizar una asignación de un servicio son:

1. Dar clic en los móviles a los que se les va a asignar el servicio.
2. Dar clic en la línea a la cual se le va a prestar el servicio.
3. Verificar que estén todos los móviles y si es necesario editar el campo de móvil, dentro de este campo pueden ir varios móviles separados con un símbolo de mas (+).
4. Dar clic en el botón asignar
5. Revisar en el historial en el historial si ha quedado la reserva guardada correctamente, como lo muestra la Ilustración 112. Este paso es opcional.

Ilustración 111. Proceso para la asignación de un servicio

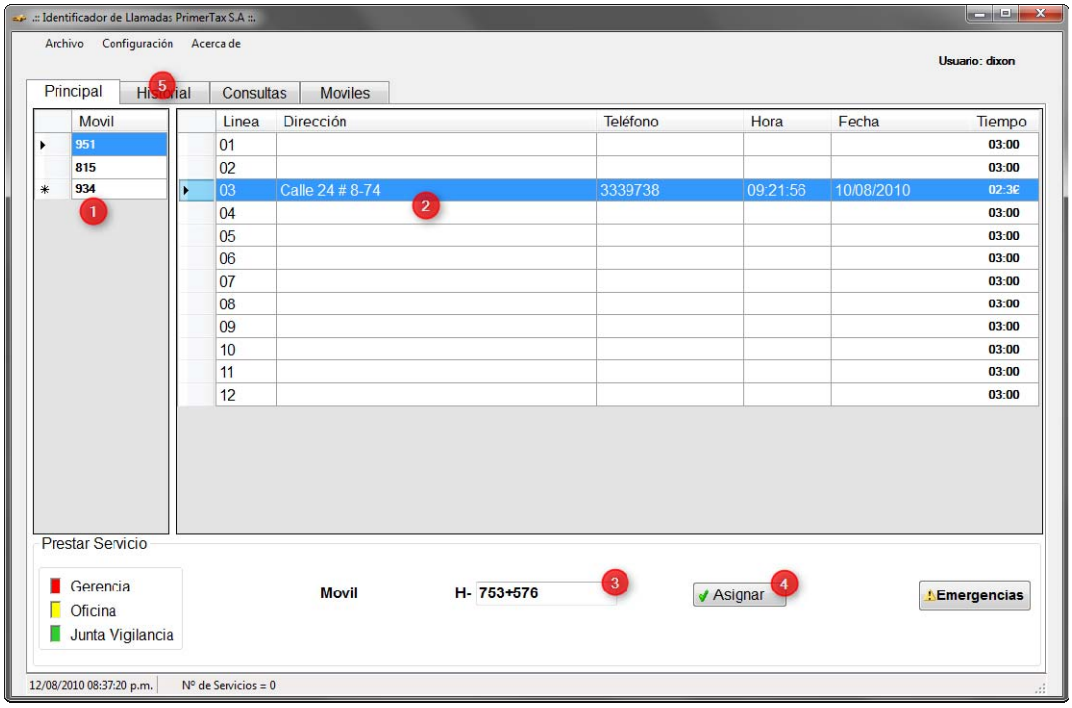
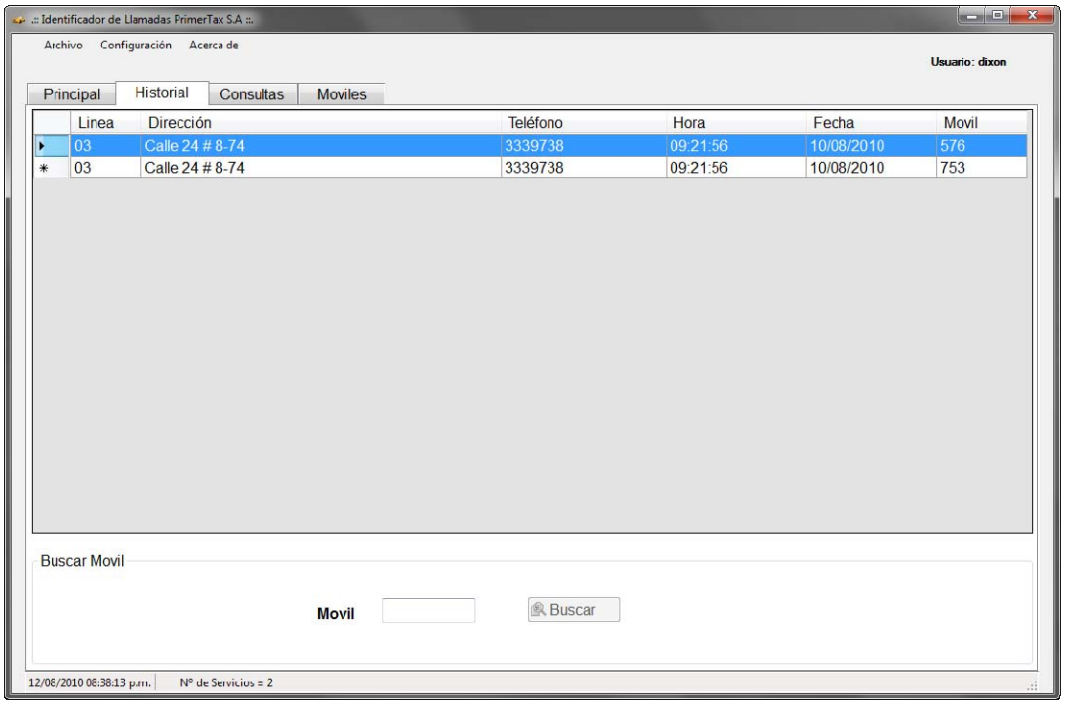


Ilustración 112. Historial de servicios prestados

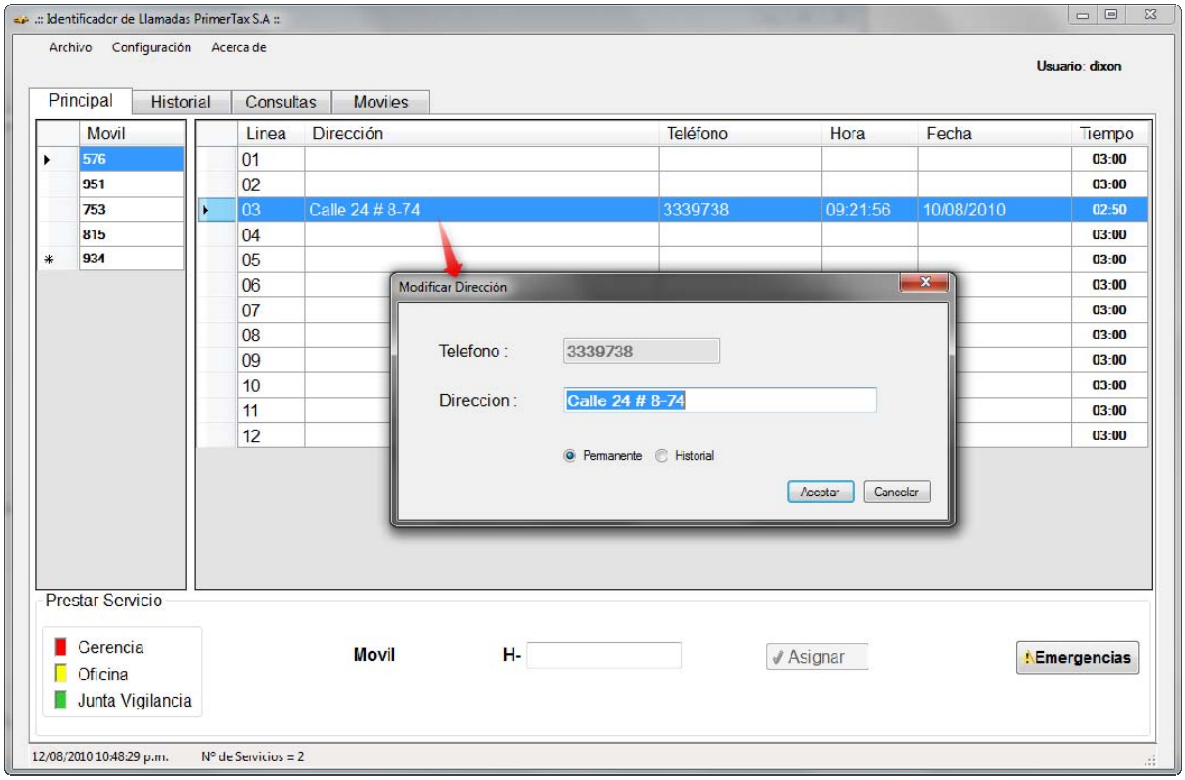


Como se muestra en la Ilustración 112, se puede ver un historial de los servicios prestados por el usuario desde que ha iniciado la sesión hasta el momento. La información mostrada es: Línea, Dirección, Teléfono, Hora, Fecha y Móvil.

Si se desea filtrar esta información para encontrar un servicio con mayor facilidad, se puede realizar por medio del móvil al cual le fue asignado el servicio. Al borrar el campo de Móvil se vuelve a tener toda la información del historial.

Como se muestra en la Ilustración 113, al dar doble clic en una línea con información de una llamada entrante, sale una ventana que da la posibilidad de cambiar la dirección asociada a ese número telefónico. Este cambio puede ser tomado de forma permanente, para que la siguientes oportunidades en que ese teléfono entre al sistema quede con la nueva dirección, o también puede ser solo para el Historial, para que solo sea un cambio de dirección para ese servicio y la próxima vez que ese teléfono entre al sistema saldrá la dirección anterior.

Ilustración 113. Modificar dirección

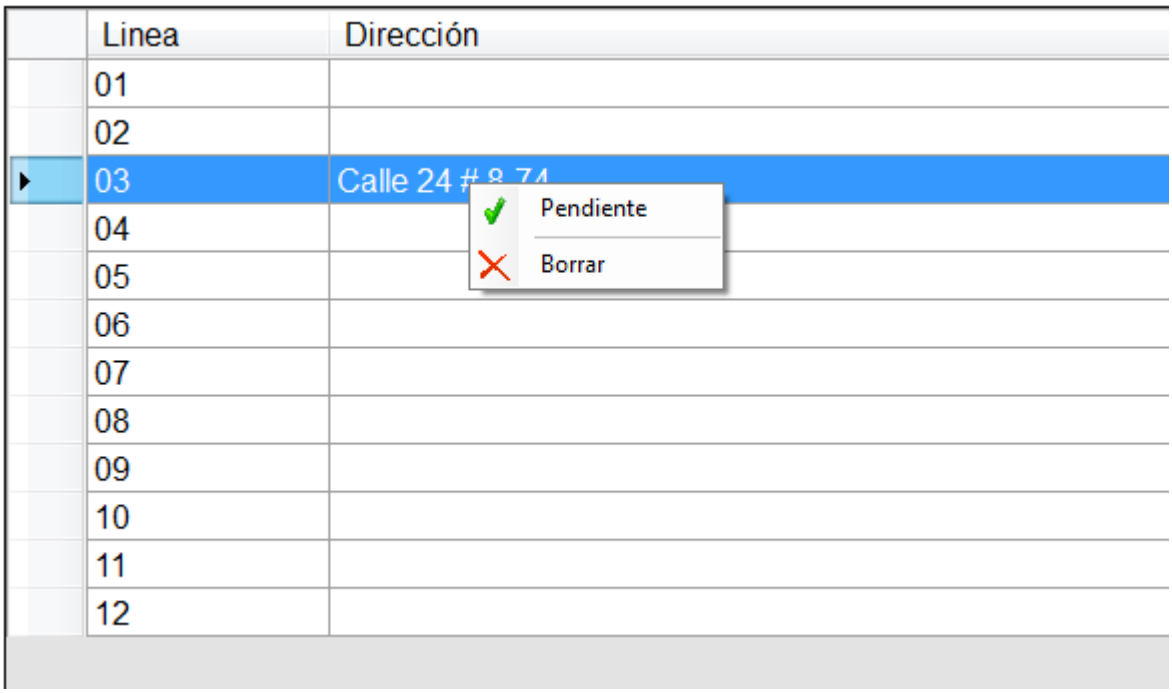


Como se muestra en la Ilustración 114, cuando se da clic derecho en una línea con información de una llamada entrante, sale un menú el cual tiene dos opciones:

- Pendiente: Significa que la línea ya ha sido contestada y se encuentra en proceso de atención. Cuando una línea es puesta como pendiente, para el usuario que hizo esa acción le sale la línea de color verde, pero si fue otro usuario el que lo realizó sale de color gris para que se pueda diferenciar entre las demás.

- Borrar: Permite eliminar la línea, para lo cual se debe seleccionar la Razón/Motivo, los cuales pueden ser:
    - Llamada de servicio al cliente.
    - Llamada equivocada.
    - Cliente Ausente
    - Llamada de mensaje
    - Otro: Para este se debe especificar cuál es la razón.

Ilustración 114. Menú clic derecho



En la sección de Consultas se pueden generar reportes que son alimentados con la información de los servicios prestados por medio del sistema.

Como se muestra en la Ilustración 115, los pasos para realizar una consulta son los siguientes:

1. Seleccionar el Objeto de consulta: Los cuales pueden ser:
    a. Cliente: Se puede especificar el número de teléfono del cliente.
    b. Móvil: Se puede especificar el número del lateral del móvil.
    c. Operador: Se puede especificar el número de cédula del operador.
    d. Unidades: Se puede especificar el código de la unidad.
    e. Sanciones: No se necesita especificar más nada y se da en el botón consultar para generar el reporte.

2. Seleccionar el Criterio de Consulta: Se usa en caso de no querer especificar de los puntos a. a d. del anterior punto, sino que se desea una consulta generalizada, se da clic en la opción de usar criterio y permite la selección de uno de los siguientes criterios para realizar el reporte:
    a. Mayor servicios prestados: Da un reporte con el orden de mayor a menor número de servicios fueron realizados.
    b. Todo: Reporte de todos los datos de la base de datos, sin categorizarlos por ningún criterio.

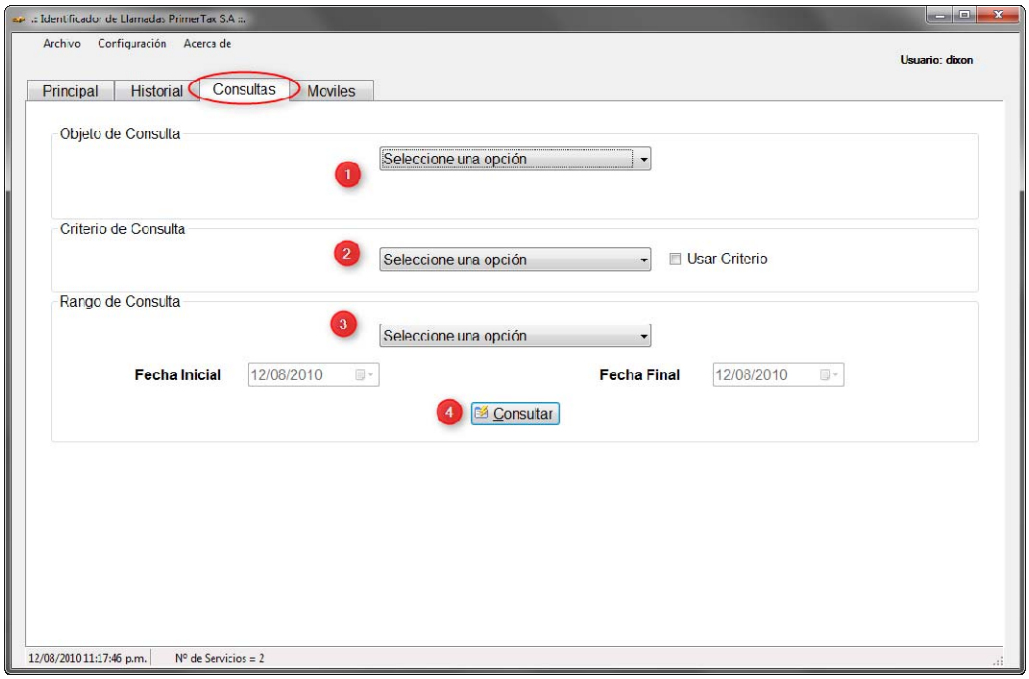
3. Seleccionar el Rango de Consulta: El cual da un límite para poder realizar el reporte y facilitar la lectura del mismo.

4. Se da clic en el botón consultar

Se recomienda sacar los reportes mensualmente, para comprobar los datos del sistema.

Se aclara que si se proporciona un rango de fechas muy grande, el reporte demorará en reunir todos los datos y organizarlos para ser mostrados.

Ilustración 115. Proceso para realizar consultas



En la sección de Móviles se da la opción de consultar la información de cada uno de los móviles, la cual es:

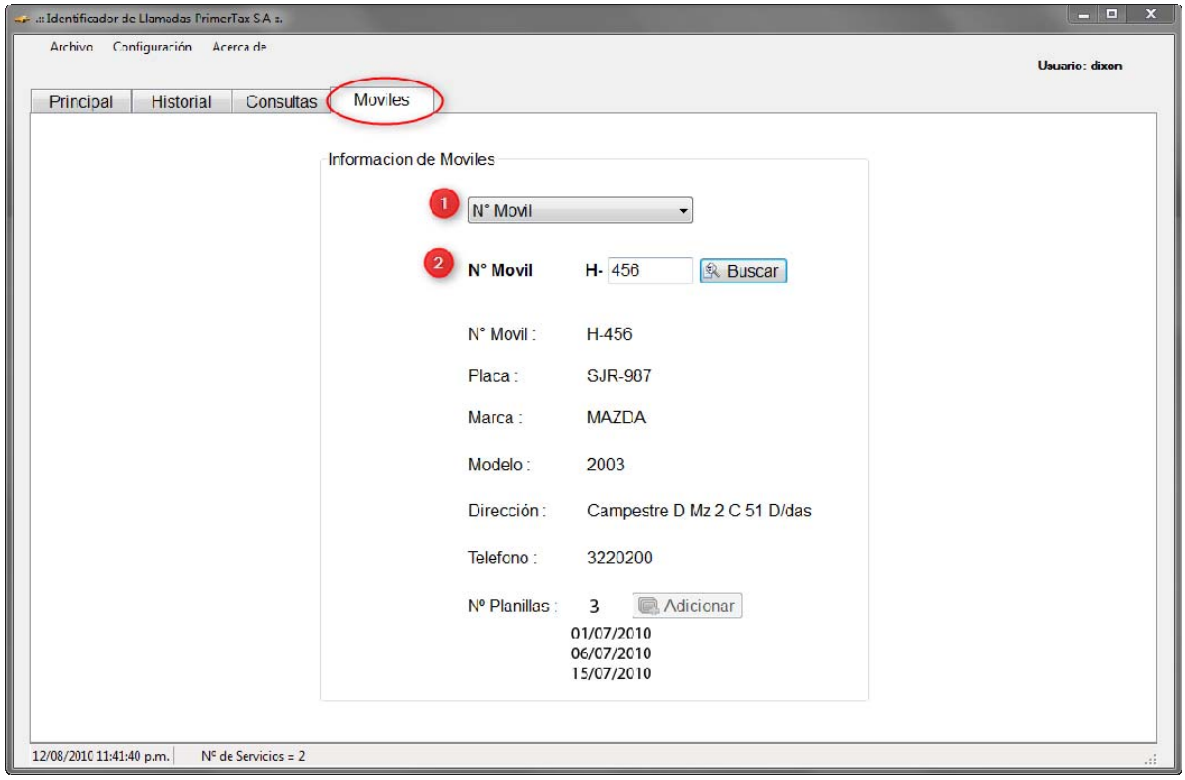
- N° Móvil.
- Placa.
- Marca.
- Modelo.
- Dirección.
- Teléfono.
- N° Planillas

El número máximo de planillas permitidas por móvil es 3, lo cual el botón de adicionar solo permitirá agregar hasta dicho límite. Cada planilla se le es registrado la fecha de cuando fue asignada.

Como se muestra en la Ilustración 116, se puede consultar la información relacionada con un móvil por medio de:

- Número de lateral (N° de móvil)
- Placa

Ilustración 116. Consultar Información de Móvil.



ANEXO G: ARQUITECTURA DEL SISTEMA

El sistema “Identificador de Llamadas PrimerTax S.A.” tiene como requerimiento el funcionamiento en varios computadores, pero con la restricción que solo uno de ellos se conecte con los dispositivos electrónicos, encargados de la identificación del número telefónico y enviarlo al computador para ser procesado. Para cumplir con dicho requerimiento, se plantea un sistema basado en la arquitectura Cliente/Servidor para poder intercomunicar a los componentes ya sea dentro de un solo computador o en varios computadores.

Otro requerimiento importante a tener en cuenta para la definición de la arquitectura es la comunicación con los dispositivos electrónicos, los cuales usan el puerto serial para informar los eventos relacionados con las líneas telefónicas. Debido a esto, se escoge como lenguaje de programación Visual Basic, por su facilidad de comunicación con el puerto serial. La versión escogida para dicho lenguaje es express.

DIAGRAMA DE PAQUETES

Servidor

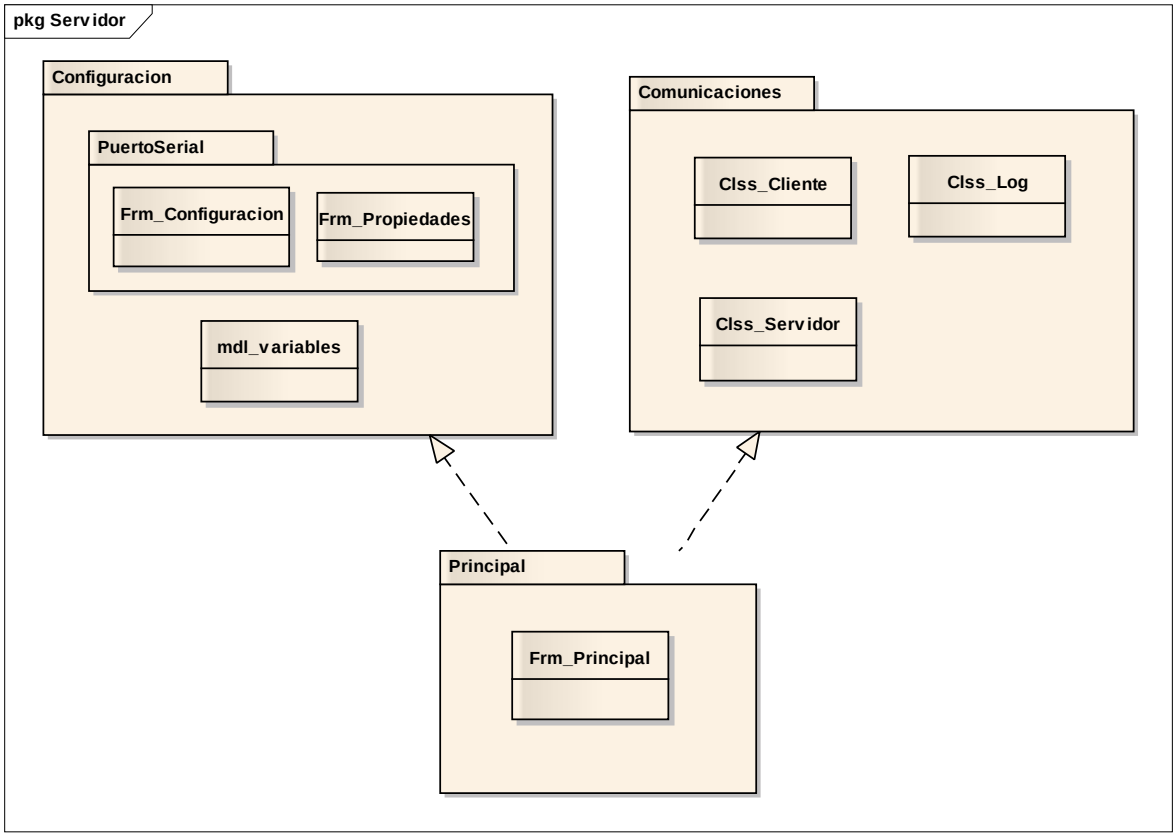
El componente servidor tiene las siguientes clases:

- Clss_Cliente
- Clss_Log
- Clss_Servidor
- Frm_Configuración
- Frm_Propiedades
- mdl_variables
- Frm_Principal

La Ilustración 117 muestra el diagrama de paquetes de este componente. Se tienen tres paquetes principales, los cuales son:

- Configuración: Es el encargado de todas las configuraciones del componente, en este caso tiene a la configuración del puerto serial y del sistema.
- Comunicaciones: Es el encargado de todo lo relacionado con comunicaciones ya sea para la comunicación Cliente/Servidor o para guardar el log del sistema.
- Principal: Es quien se encarga de usar los dos paquetes anteriores para su correcto funcionamiento.

Ilustración 117. Diagrama de Paquetes - Servidor.

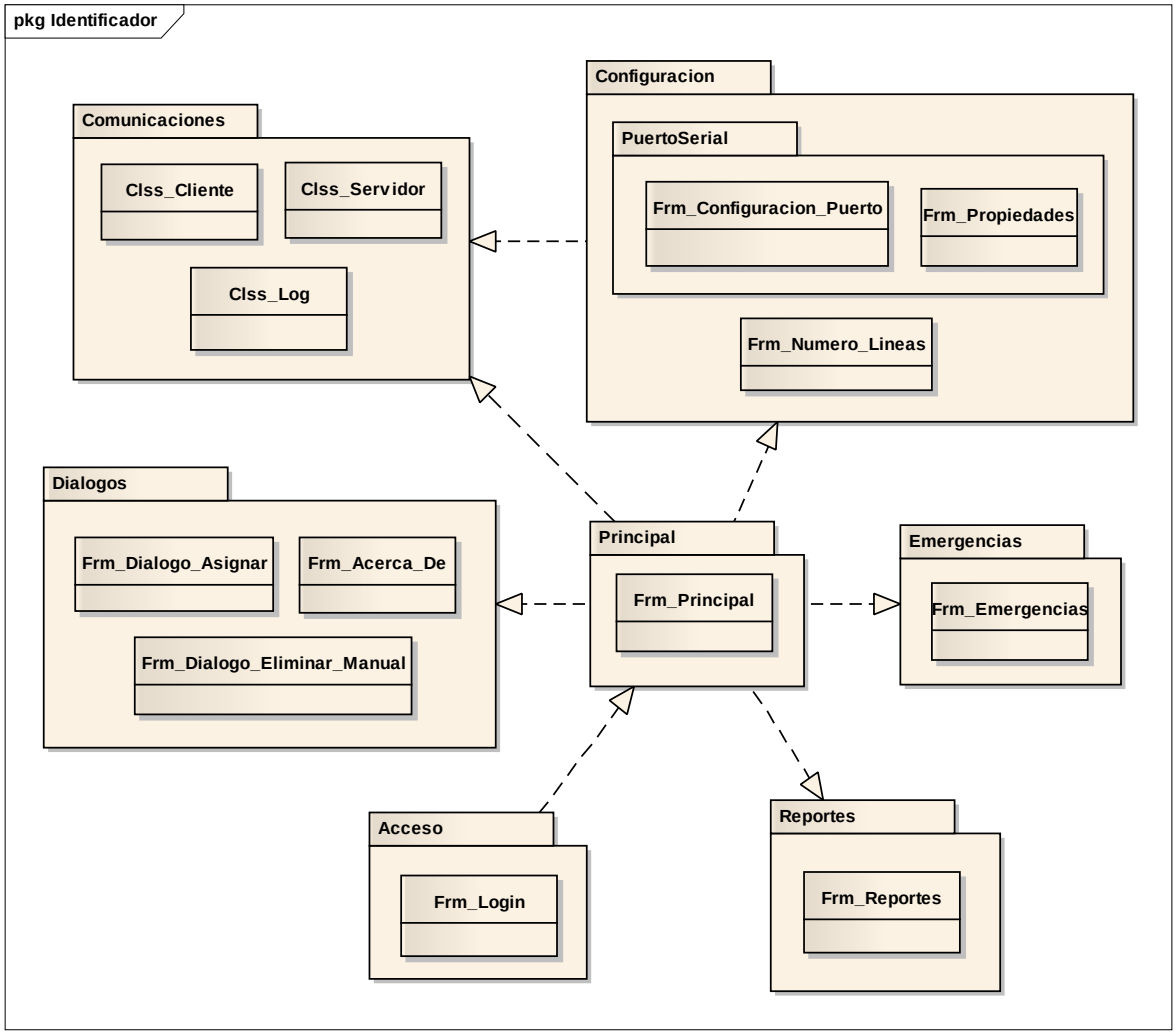


Identificador

El componente identificador tiene las siguientes clases:

- Clss_Cliente
- Clss_Log
- Clss_Servidor
- Frm_Acerca_De
- Frm_Configuracion_Puerto
- Frm_Dialogo_Asignar
- Frm_Dialogo_Eliminacion_Manual
- Frm_Emergencias
- Frm_Login
- Frm_Numero_Lineas
- Frm_Principal
- Frm_Propiedades
- Frm_Reportes

Ilustración 118. Diagrama de Paquetes - Identificador.



La Ilustración 118 muestra el diagrama de paquetes de este componente. Se tienen 7 paquetes principales, los cuales son:

- Configuración: Es el encargado de todas las configuraciones del componente, en este caso tiene a la configuración del puerto serial y del número de líneas con las cuales se va a trabajar.
- Emergencias: Se encarga de manipular las emergencias reportadas por los radios dentro de los móviles, y además se encarga de los seguimientos que los móviles piden al momento de entrar en una zona de alto riesgo.
- Reportes: Se encarga de mostrar todos los reportes que el identificador puede generar. En este paquete se encuentran todos los reportes que se pueden generar desde el Principal.
- Acceso: Se encarga de realizar las validaciones necesarias para comenzar a usar el sistema, estas validaciones son primordialmente el usuario y su contraseña, luego elimina las sanciones que ya han sido cumplidas para

que en esta nueva sesión no sean tomadas en cuenta. Y en este paquete se deben consignar todas las acciones necesarias antes de entrar al sistema.

- Dialogos: Se encarga de contener todas aquellas clases que se usan para la interacción con el usuario en cuando a mostrar cuadros de información o peticiones de pocos campos. En este caso tiene a su cargo la clase de Frm_Dialogo_Asignar que es un cuadro que sirve para pedir el teléfono y la dirección para asignar un servicio sin necesidad de que este sea identificado automáticamente por el sistema. También tiene la clase Frm_Dialogo_Eliminar_Manual que se encarga de pedirle al usuario la razón/motivo por el cual se está eliminando una llamada. Y por ultimo tiene la clase Frm_Acerca_De que es un cuadro de información acerca de los creadores del programa.
- Comunicaciones: Es el encargado de todo lo relacionado con comunicaciones ya sea para la comunicación Cliente/Servidor o para guardar el log del sistema.
- Principal: Es quien se encarga de usar los cinco paquetes anteriores, menos el de Acceso, para su correcto funcionamiento.

Administrador

El componente Administrador tiene las clases:

- Frm_Cambio_Clave
- Frm_Editar_Registro
- Frm_Login
- Frm_Principal
- Frm_Reportes
- Frm_Super_Backup
- mdl_variables

La Ilustración 119 muestra el diagrama de paquetes de este componente. Se tienen 5 paquetes, los cuales son:

- Reportes: Se encarga de mostrar todos los reportes que el identificador puede generar. En este paquete se encuentran todos los reportes de sistema. Además se encarga del Super Backup, que significa generar todos los reportes posibles del sistema, tanto del componente Identificador como del Administrador.
- Acceso: Se encarga de realizar las validaciones necesarias para comenzar a usar el sistema, en este caso solo se valida el usuario y su contraseña.
- Dialogos: Se encarga de contener todas aquellas clases que se usan para la interacción con el usuario en cuando a mostrar cuadros de información o peticiones de pocos campos. En este caso tiene a su cargo la clase Frm_Cambio_Clave que sirve para pedirle al usuario que cambie la clave

con una pequeña comprobación de la misma. Además tiene la clase Frm_Editar_Registro que sirve para que el usuario pueda editar un registro mostrándole los datos necesarios para ello.

- Configuración: Es el encargado de todas las configuraciones del componente, en este caso tiene las variables de configuración del sistema.
- Principal: Es quien se encarga de usar los tres paquetes anteriores, menos el de Acceso, para su correcto funcionamiento.

Ilustración 119. Diagrama de Paquetes - Administrador.

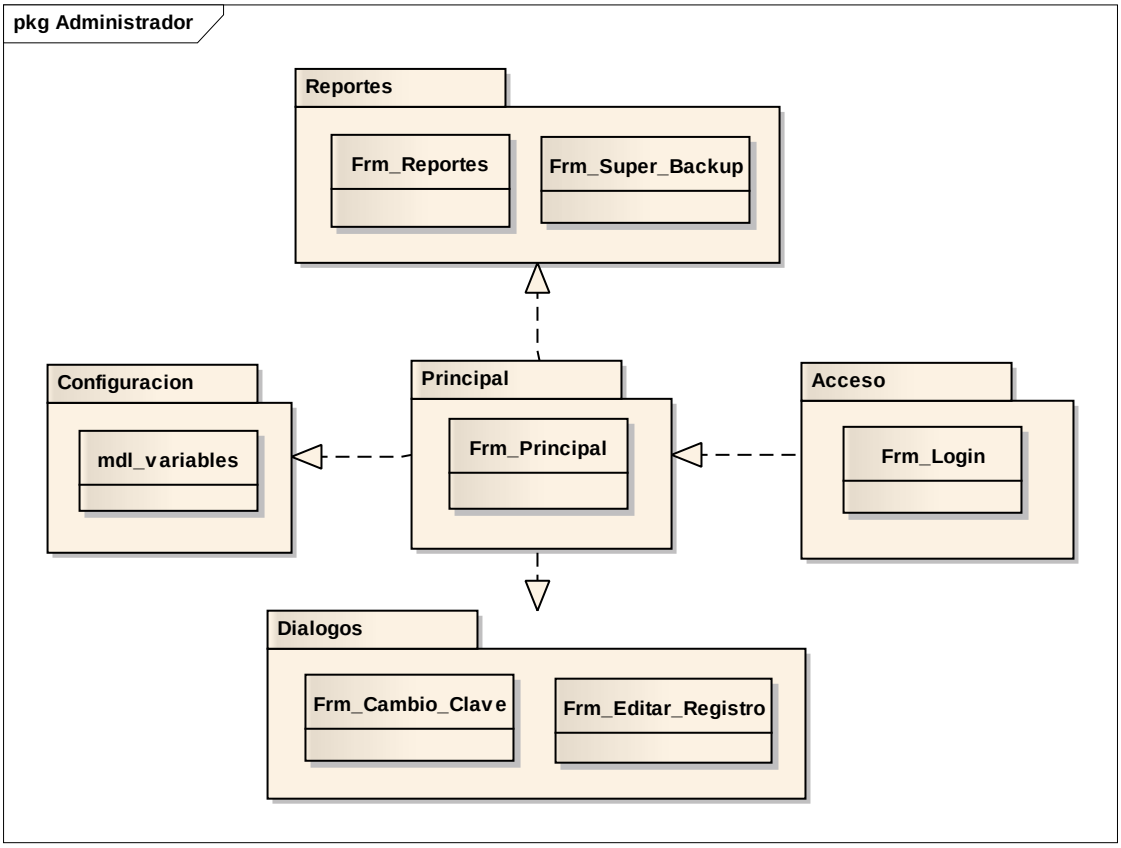


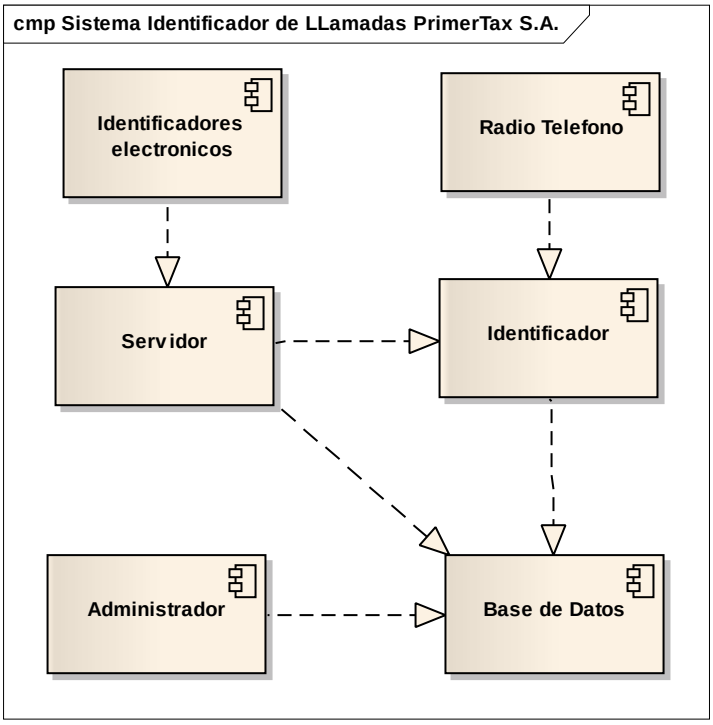
DIAGRAMA DE COMPONENTES

El diagrama de componentes de la Ilustración 120, muestra la como se conectan los componentes. El servidor recibe por medio del puerto serial la información que los dispositivos electrónicos le proporcionan con respecto a una llamada. La empresa Primertax cuenta con dos identificadores electrónicos de llamadas los cuales pueden manejar 8 líneas telefónicas cada uno, estos son conectados entre sí para dar una única salida de hasta 16 líneas telefónicas, de las cuales solo son usadas 12 por mandato de la misma empresa.

El componente Servidor procesa las líneas e informa al componente identificador que existe una nueva llamada. El servidor almancena la información de la nueva llamada y de todos los identificadores conectados a él, dentro de la base de datos para evitar perdida de información en caso de una caída del componente servidor.

El componente Identificador consulta en la base de datos la información de la nueva llamada y la muestra al usuario. Este componente también se encarga de almacenar la información relacionada con el servicio, que seria la información de la llamada y además la información del móvil el cual se le fue asignado dicho servicio.

Ilustración 120. Diagrama de Componentes.



El componente Radio Teléfono es otro dispositivo electrónico que le suministra al

componente Identificador el identificador del móvil que se ha reportado por este medio. De esta forma el sistema se vuelve más automático y más usable para los usuarios del componente Identificador.

El componente Administrador se usa para adicionar, modificar y eliminar información de la base de datos. Dicha información es la de los usuarios del sistema, los móviles que se pueden reportar, los registros de los servicios realizados por medio del Identificador, las sanciones que se le pueden adicionar a un móvil, y los reportes administrativos.

El componente Base de Datos es una donde se almacenarán los datos de todo el sistema, el cual se implementara en mysql gracias a su licencia de libre uso.

DIAGRAMA DE DESPLIEGUE

La Ilustración 121 muestra el diagrama de despliegue del sistema. Se tiene que el primer computador actúa como una especie de servidor, donde se encuentra el componente Identificador, el cual se conecta al radio teléfono por medio del puerto serial, pero esta conexión debe ser por medio de un cable de conversión de puerto serial a USB, pues el puerto serial estará ocupado por los Identificadores Electrónicos.

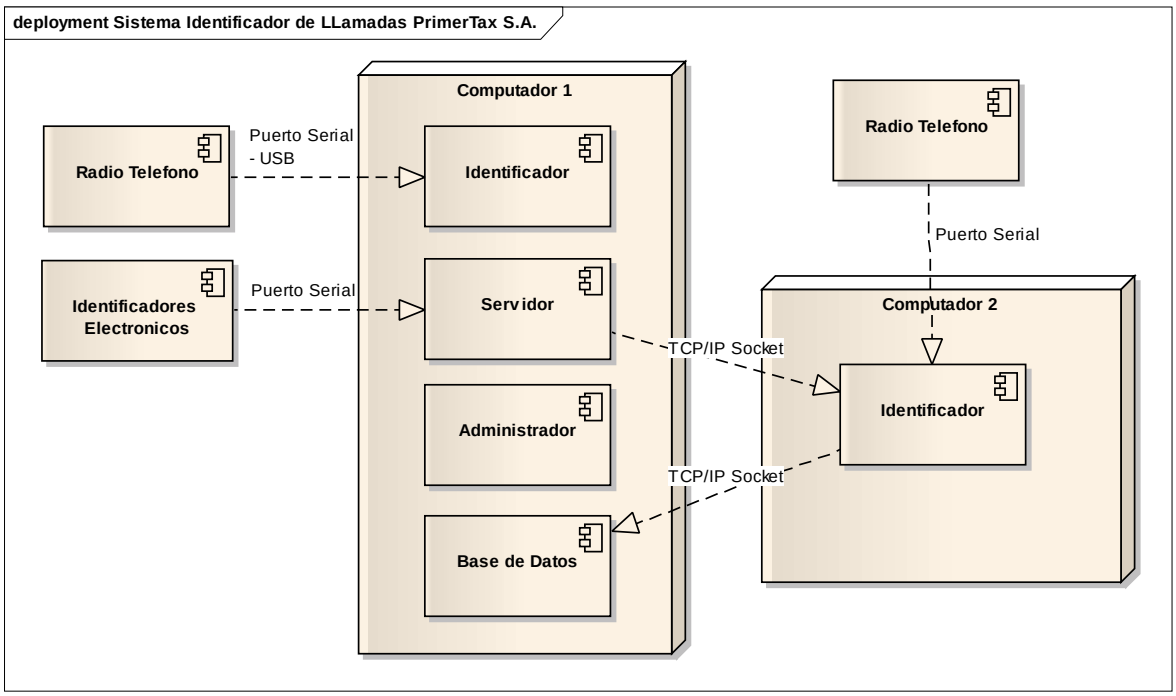
También se tiene el componente Servidor, el cual se conecta directamente por el puerto serial con el componente Identificadores Electrónicos.

Al actuar como servidor, se adiciona allí también el componente de Administrador para unificar la administración del sistema.

El componente de Base de Datos se encuentra en el mismo computador por lo cual el computador tiene que convertirse en servidor por medio del programa Apache o un programa que cumpla la misma función, para que los demás computadores puedan acceder a este componente. Se recomienda el uso del programa wampserver por su facilidad de instalación del apache, mysql y php, y además por su licencia de libre uso.

En el computador 2 solo de tiene al componente Identificador en cual está conectado al componente Radio Telefono por medio del puerto serial directamente. También está conectado computador 1 para tener comunicación con el componente Servidor por medio de una conexión de Ethernet TCP/IP usando la tecnología de Sockets propia de la arquitectura Cliente/Servidor, y también para conectarse con la base de datos por medio de Ethernet TCP/IP Sockets.

Ilustración 121. Diagrama de Despliegue.


deployment Sistema Identificador de LLamadas PrimerTax S.A.
Computador 1
Radio Telefono
Puerto Serial
- USB
Identificador
Identificadores
Electronicos
Puerto Serial
Servidor
Administrador
Base de Datos
TCP/IP Socket
TCP/IP Socket
Radio Telefono
Puerto Serial
Computador 2
Identificador