

# **ESCUELA POLITÉCNICA NACIONAL**

## **FACULTAD DE INGENIERÍA DE SISTEMAS**

**DESARROLLO DE UN SISTEMA WEB PARA EL MONITOREO DE  
TRANSACCIONES DE APLICACIONES EN UN ENTORNO DE  
INTRANET.**

**PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN  
SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

**Cano Sánchez Marcelo Javier**

[marcel\\_knoo@hotmail.com](mailto:marcel_knoo@hotmail.com)

**Samaniego Vizcaíno Lenin Guillermo**

[lgvsamaniego@hotmail.com](mailto:lgvsamaniego@hotmail.com)

**Director: Ing. Bolívar Palán Tamayo MSc.**

[bolivar.palan@epn.edu.ec](mailto:bolivar.palan@epn.edu.ec)

**Quito, Noviembre de 2015**

## DECLARACIÓN

Nosotros, Marcelo Javier Cano Sánchez y Lenin Guillermo Samaniego Vizcaíno, declaramos estando bajo juramento que todo el trabajo que se describe aquí es totalmente de nuestra autoría, además el mismo no ha sido anteriormente presentado en ningún grado o presentación profesional. Nosotros hemos consultado las referencias bibliográficas que se encuentran en este documento.

Dicho esto se concede nuestros derechos de propiedad intelectual correspondiente a este trabajo que La Escuela Politécnica Nacional tiene establecido por ley de Propiedad Intelectual, y por las normativas institucionales vigentes en su reglamento.

---

Marcelo Cano Sánchez

---

Lenin Samaniego Vizcaíno

## CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Marcelo Javier Cano Sánchez y Lenin Guillermo Samaniego Vizcaíno, bajo mi supervisión.

---

**Msc. Ing. Bolívar Palán Tamayo**

Director de Proyecto

## Agradecimiento

Agradezco a mi Padre Dios que me dio la vida y salud para poder cumplir todas las metas que he obtenido en esta vida hasta este momento.

A mis padres que me ayudaron a ser una mejor persona enseñándome con el mejor de los métodos que es el ejemplo.

A mi hermana que siempre daba todo de su vida para apoyarme cuando más lo necesitaba dándome un motivo para no rendirme.

A mis compañeros que me ayudaron a ser mejor profesional.

Al Msc. Ing. Bolívar Palán que me apoyo en cada momento tanto como estudiante como tesista.

**Marcelo Cano Sánchez**

## **Dedicatoria**

La presente Tesis se la agradezco a Dios el cual me regalo sabiduría para alcanzar esta meta.

A mis padres que siempre me dieron una total confianza para convertirme en un profesional y una persona de la cual ellos estarían complacidos.

A mi hermana que me apoyo más que ninguna otra persona.

**Marcelo Cano Sánchez**

## **Agradecimiento**

A mi madre y hermanas por brindarme el apoyo necesario para cumplir mis metas, además que gracias a sus enseñanzas he logrado ser una mejor persona.

A mis amigos los cuales me brindaron horas de diversión, ya que gracias a ellos el ambiente estresante de la universidad, pasó desapercibido.

A mis profesores por ser una guía en mi camino estudiantil, ya que gracias a ellos he logrado formarme como un futuro profesional con principios éticos y morales.

Al Ing. Bolívar Palan quien me ha guiado en la elaboración del proyecto de titulación.

**Lenin Samaniego Vizcaíno**

## **Dedicatoria**

A mi familia Suli Vizcaíno, Elisa Samaniego y Lizeth Samaniego los cuales me han brindado un apoyo moral y económico.

A mi tutor Bolívar Palan quien nos a Guido a mí y a Marcelo Cano en la elaboración del trabajo.

A amigos por brindarme horas de diversión y compañía.

A mis profesores por brindarme generosamente sus conocimientos, pues a través de sus enseñanzas he logrado ser mejor cada día de mi vida.

**Lenin Samaniego Vizcaíno**

## ÍNDICE DE CONTENIDO

<b>RESUMEN .....</b>	<b>I</b>
<b>CAPÍTULO 1 .....</b>	<b>2</b>
<b>1 DESCRIPCIÓN DEL PROBLEMA.....</b>	<b>2</b>
<b>1.1 ANALIZAR TRANSACCIONES DE APLICATIVOS EN UNA INTRANET .....</b>	<b>2</b>
<b>1.2 SITUACIÓN ACTUAL DE LAS ORGANIZACIONES EN EL AMBITO DEL RENDIMIENTO DE SUS APLICACIONES.....</b>	<b>3</b>
<b>1.3 JUSTIFICACIÓN DE LA METODOLOGÍA A UTILIZAR .....</b>	<b>4</b>
<b>1.3.1 METODOLOGÍA SCRUM.....</b>	<b>6</b>
<b>1.3.2 RAZONES PARA UTILIZAR SCRUM .....</b>	<b>7</b>
<b>1.4 JUSTIFICACIÓN DE LAS HERRAMIENTAS A UTILIZAR .....</b>	<b>8</b>
<b>1.4.1 PLATAFORMA MICROSOFT .NET.....</b>	<b>9</b>
<b>1.4.2 ASP.NET .....</b>	<b>10</b>
<b>1.4.3 MODELO MVC .....</b>	<b>12</b>
<b>1.4.4 MOTOR DE BASE DE DATOS SQL SERVER .....</b>	<b>14</b>
<b>1.4.5 LENGUAJE DE PROGRAMACIÓN C#.....</b>	<b>15</b>
<b>CAPÍTULO 2 .....</b>	<b>16</b>
<b>2 DESARROLLO DEL SISTEMA UTILIZANDO SCRUM.....</b>	<b>16</b>
<b>2.1 ESPECIFICACIÓN DE REQUISITOS – HISTORIAS DE USUARIO (PRODUCT BACKLOG) .....</b>	<b>16</b>
<b>2.1.1 DEFINICIÓN DE PERFILES Y ROLES DEL SISTEMA .....</b>	<b>16</b>
<b>2.1.2 HISTORIAS DE USUARIO .....</b>	<b>17</b>
<b>2.1.3 PILA DEL PRODUCTO INICIAL (Product Backlog) .....</b>	<b>20</b>
<b>2.2 PLANIFICACIÓN DE LOS SPRINTS (SPRINTS BACKLOG) .....</b>	<b>21</b>
<b>2.2.1 PLANIFICACIÓN DEL SPRINT CERO .....</b>	<b>22</b>
<b>2.2.2 PLANIFICACIÓN DEL PRIMER SPRINT .....</b>	<b>24</b>
<b>2.2.3 PLANIFICACIÓN DEL SEGUNDO SPRINT .....</b>	<b>26</b>
<b>2.2.4 PLANIFICACIÓN DEL TERCER SPRINT .....</b>	<b>28</b>
<b>2.2.5 PLANIFICACIÓN DEL CUARTO SPRINT .....</b>	<b>30</b>

2.3	EJECUCIÓN DE LOS SPRINTS .....	32
2.3.1	EJECUCIÓN DEL SPRINT CERO .....	32
2.3.2	EJECUCIÓN DEL PRIMER SPRINT .....	37
2.3.3	EJECUCIÓN DEL SEGUNDO SPRINT .....	42
2.3.4	EJECUCIÓN DEL TERCER SPRINT .....	45
2.3.5	EJECUCIÓN DEL CUARTO SPRINT .....	47
2.3.6	PRUEBAS UNITARIAS .....	49
2.4	ENTREGA DE LOS SPRINTS .....	85
2.4.1	PRESENTACIÓN DEL SPRINT CERO .....	85
2.4.2	PRESENTACIÓN DEL PRIMER SPRINT .....	86
2.4.3	PRESENTACIÓN DEL SEGUNDO SPRINT .....	88
2.4.4	PRESENTACIÓN DEL TERCER SPRINT .....	89
2.4.5	PRESENTACIÓN DEL CUARTO SPRINT .....	90
2.4.6	REUNIONES DE REVISIÓN .....	91
2.4.7	REUNIONES DE RETROSPECTIVA .....	93
CAPÍTULO 3 .....	97	
3	EVALUACIÓN DEL SISTEMA EN EL CASO DE ESTUDIO .....	97
3.1	RECOPILACIÓN DE DATOS DE MONITOREO .....	97
3.1.1	SISTEMA A MONITOREAR .....	97
3.1.2	DATOS A RECOPILAR .....	99
3.2	INSTALACIÓN DEL SISTEMA .....	102
3.2.1	CONFIGURACIÓN DEL ENTORNO .....	103
3.2.2	COMPILACION DE LIZARD .....	104
3.2.3	PUESTA EN MARCHA .....	107
3.2.4	CAPTURA DE DATOS .....	109
3.3	PRUEBAS DEL SISTEMA DE MONITOREO .....	115
3.3.1	PRUEBAS DE ACEPTACIÓN .....	115
3.3.2	PRUEBAS DE RESISTENCIA .....	118
3.3.3	PRUEBAS DE RENDIMIENTO .....	121
3.4	ANALISIS DE RESULTADOS .....	125
3.4.1	CRITERIOS PARA A VALORACIÓN .....	125
3.4.2	ANALISIS DE ACEPTACIÓN .....	126

<b>3.4.3</b>	<b>ANALISIS DE RESISTENCIA.....</b>	<b>129</b>
<b>3.4.4</b>	<b>ANALISIS DE RENDIMIENTO .....</b>	<b>130</b>
<b>3.4.5</b>	<b>ANALISIS DE FUNCIONALIDAD .....</b>	<b>133</b>
<b>3.4.6</b>	<b>ANALISIS DE USABILIDAD.....</b>	<b>135</b>
<b>3.4.7</b>	<b>RESUMEN DEL ANÁLISIS DE EVALUACIÓN .....</b>	<b>136</b>
<b>CAPÍTULO 4.....</b>		<b>137</b>
<b>4</b>	<b>CONCLUSIONES Y RECOMENDACIONES .....</b>	<b>137</b>
<b>4.1</b>	<b>CONCLUSIONES .....</b>	<b>137</b>
<b>4.2</b>	<b>RECOMENDACIONES .....</b>	<b>138</b>
<b>5</b>	<b>BIBLIOGRAFÍA .....</b>	<b>140</b>
<b>6</b>	<b>GLOSARIO DE TÉRMINOS .....</b>	<b>142</b>
<b>7</b>	<b>ANEXOS .....</b>	<b>144</b>

## LISTA DE FIGURAS

Figura 1.1 Marco de trabajo de ASP.NET.....	10
Figura 1.2 Arquitectura ASP.Net .....	11
Figura 1.3 Procesamiento de páginas web en ASP.NET .....	12
Figura 1.4 Arquitectura MVC 2 Para la WEB.....	14
Figura 1.5: Arquitectura SQL Server .....	15
Figura 2.1 Planificación de los Sprints.....	21
Figura 2.2 Tareas del Sprint Cero.....	23
Figura 2.3 Esfuerzo realizado en el Sprint Cero.....	23
Figura 2.4 Avances de tareas del Sprint Cero .....	23
Figura 2.5 Tareas del Primer Sprint.....	25
Figura 2.6 Esfuerzo realizado en el Primer Sprint.....	25
Figura 2.7 Avances de tareas del Primer Sprint .....	26
Figura 2.8 Tareas del Segundo Sprint .....	27
Figura 2.9 Esfuerzo realizado en el Segundo Sprint .....	27
Figura 2.10 Avances de tareas del Segundo Sprint .....	28
Figura 2.11 Tareas del Tercer Sprint.....	29
Figura 2.12 Esfuerzo realizado en el Tercer Sprint .....	30

Figura 2.13 Avances de tareas del Tercer Sprint.....	30
Figura 2.14 Tareas del Cuarto Sprint.....	31
Figura 2.15 Esfuerzo realizado en el Cuarto Sprint.....	32
Figura 2.16 Avances de tareas del Cuarto Sprint .....	32
Figura 2.17 Arquitectura de la Solución.....	33
Figura 2.18 Modelo Entidad-Relación de la base de datos .....	33
Figura 2.19 Diagrama modelo físico de la base de datos .....	34
Figura 2.20 Generación de la BDD .....	36
Figura 2.21 Procedimiento Almacenado Insertar Usuario .....	36
Figura 2.22 Procedimiento Almacenado Acceso Usuario.....	37
Figura 2.23 Cadena de conexión .....	38
Figura 2.24 Cadena de conexión del proyecto .....	38
Figura 2.25 Clase LINQ DataLizardBDD.dbml.....	38
Figura 2.26 Mapeo de Tabla y Procedimientos Almacenados de Usuario.....	39
Figura 2.27 Prototipo Página USUARIO .....	40
Figura 2.28 Mapeo Tabla y Procedimientos Almacenados UBICACION.....	40
Figura 2.29 Prototipo Página UBICACION.....	41
Figura 2.30 Cambio de tareas planificadas primer Sprint .....	42
Figura 2.31 Mapeo de Tabla y Procedimientos Almacenados de Configuración .....	43
Figura 2.32 Prototipo Página Configuración .....	44
Figura 2.33 Cambio de tareas planificadas segundo Sprint.....	44
Figura 2.34 Mapeo de Tabla y SPs correspondientes a CONFIGURACION.....	45
Figura 2.35 Prototipo Página Transacciones .....	46
Figura 2.36 Cambio de tareas planificadas tercer Sprint.....	47
Figura 2.37 Prototipo Página Reportes.....	48
Figura 2.38 Pagina de registro del Sistema .....	50
Figura 2.39 Crear nuevo Usuario del Sistema .....	51
Figura 2.40 Página de inicio del Sistema .....	51
Figura 2.41 Ingreso al sistema de un usuario registrado .....	52
Figura 2.42 Ingreso a la aplicación por parte de un usuario registrado.....	53
Figura 2.43 Inicio de aplicación .....	53
Figura 2.44 Ingreso al módulo de ubicación .....	54

Figura 2.45 Ingreso de datos de ubicación .....	55
Figura 2.46 Registro de ubicación.....	55
Figura 2.47 Ingreso al módulo de ubicación .....	56
Figura 2.48 Agregar una sub-ubicación .....	56
Figura 2.49 Ingreso de datos de sub-ubicación .....	57
Figura 2.50 Registro de sub-ubicación .....	57
Figura 2.51 Ingreso al módulo de ubicación .....	58
Figura 2.52 Edición de ubicación .....	58
Figura 2.53 Edición de datos de ubicación.....	59
Figura 2.54 Actualización de datos de ubicación .....	59
Figura 2.55 Ubicación Actualizada .....	59
Figura 2.56 Ingreso al módulo ubicación .....	60
Figura 2.57 Eliminación de ubicación.....	61
Figura 2.58 Ubicación eliminada.....	61
Figura 2.59 Ingreso al módulo de configuración.....	62
Figura 2.60 Ingreso de datos de nueva configuración .....	62
Figura 2.61 Registro de configuración .....	63
Figura 2.62 Ingreso al módulo de configuración.....	64
Figura 2.63 Edición de configuración .....	64
Figura 2.64 Ingreso de Datos de configuración.....	64
Figura 2.65 Edición de datos de configuración .....	65
Figura 2.66 Registro de edición de configuración.....	65
Figura 2.67 Ingreso al módulo configuración .....	66
Figura 2.68 Selección de ubicación a ser eliminada.....	66
Figura 2.69 Confirmación de eliminación de configuración .....	66
Figura 2.70 Acceso al módulo de configuración .....	67
Figura 2.71 Selección de configuración .....	68
Figura 2.72 Transacciones tipo Históricas .....	68
Figura 2.73 Selección de ubicación para transacciones .....	68
Figura 2.74 Selección de ubicación para Históricos.....	69
Figura 2.75 Generación de gráficos transaccionales .....	69
Figura 2.76 Generación de gráficos Históricos .....	70

Figura 2.77 Ingreso al módulo transacciones .....	71
Figura 2.78 Selección de ubicación para gráficas en tiempo real .....	71
Figura 2.79 Generación de gráfica en tiempo real.....	71
Figura 2.80 Ingreso al módulo de configuración.....	72
Figura 2.81 Selección de configuración de transacciones.....	73
Figura 2.82 Selección de configuración para reportes históricos.....	73
Figura 2.83 Selección de ubicación para reportes transaccionales.....	73
Figura 2.84 Selección de ubicación para reportes históricos.....	74
Figura 2.85 Generación de reportes de transacción.....	74
Figura 2.86 Generación de reportes históricos .....	75
Figura 2.87 Reporte por transacción .....	75
Figura 2.88 Reporte histórico .....	76
Figura 2.89 Ingreso al módulo de configuración.....	77
Figura 2.90 Selección de configuración .....	77
Figura 2.91 Selección de configuración de históricos.....	78
Figura 2.92 Selección de ubicación de reportes transaccionales.....	78
Figura 2.93 Selección de ubicaciones históricas .....	78
Figura 2.94 Generación de reporte transaccional .....	79
Figura 2.95 Generación de reporte histórico .....	79
Figura 2.96 Reporte de transacciones .....	80
Figura 2.97 Reportes históricos .....	80
Figura 2.98 Ingreso al módulo transacciones .....	81
Figura 2.99 Selección de ubicación.....	82
Figura 2.100 Generación de reporte en tiempo real .....	82
Figura 2.101 Reporte en tiempo real de clientes .....	82
Figura 2.102 Acceso a un cliente por parte del súper administrador .....	83
Figura 2.103 Referenciación de la DLL del proyecto .....	84
Figura 2.104 Método de captura de datos .....	84
Figura 2.105 Cumplimiento de las Tareas Sprint Cero .....	85
Figura 2.106 Esquema BDD SQL Server.....	85
<b>Figura 2.107 Página Ingreso Usuarios .....</b>	<b>86</b>
Figura 2.108 Cumplimiento Tareas Primer Sprint .....	86

Figura 2.109 Página Registro de Usuarios .....	87
Figura 2.110 Página Ingreso Usuarios.....	87
Figura 2.111 Página ASPX, Módulo UBICACIÓN.....	87
Figura 2.112 Cumplimiento Tareas Segundo Sprint .....	88
Figura 2.113 Página ASPX, Módulo Configuración.....	88
Figura 2.114 Cumplimiento Tareas Tercer Sprint.....	89
Figura 2.115 Página ASPX, Módulo Transacciones .....	89
Figura 2.116 Cumplimiento Tareas Cuarto Sprint .....	90
Figura 2.117 Pagina de Reportes.....	90
Figura 3.1 Acceso al sistema SIREC-Q .....	98
Figura 3.2 SIREC-Q del Municipio de Quito.....	98
Figura 3.3 Selección de registro a editar .....	99
Figura 3.4 Compilación de la Solución .....	105
Figura 3.5 Generación de DLL de la Solución .....	105
Figura 3.6 Publicación del Proyecto .....	105
Figura 3.7 Precompilado de la Solución.....	106
Figura 3.8 Carpeta Instalación LIZARD .....	106
Figura 3.9 SRIFT para generar la BDD.....	107
Figura 3.10 Infraestructura de la base de datos de la Solución .....	107
Figura 3.11 Servidor Web IIS.....	108
Figura 3.12 Sistema corriendo desde el servidor.....	108
Figura 3.13 Lizard.dll .....	109
Figura 3.14 Referencia DLL “Lizard.dll” .....	109
Figura 3.15 “Captura Datos” .....	110
Figura 3.16 Adición de Cadena de Conexión.....	110
Figura 3.17 Modulo Ubicación Aplicativo Lizard .....	111
Figura 3.18 Ubicación DMI Prueba SIRECQ .....	112
Figura 3.19 Modulo Configuración .....	112
Figura 3.20 Registro de configuraciones.....	112
Figura 3.21 Configuraciones Registradas.....	113
Figura 3.22 Aplicación de la DMI.....	113
Figura 3.23 Sistema SIREC-Q.....	114

Figura 3.24 Sistema SIREC-Q Edición de Lotes .....	114
Figura 3.25 Confirmación de Edición .....	114
Figura 3.26 Transacciones registradas en la Base de datos LIZARDBDD .....	115
Figura 3.27 Grafico de transacciones del SIRECQ .....	116
Figura 3.28 Reporte de las transacciones del SIREC-Q .....	116
Figura 3.29 Reporte de las transacciones en tiempo real del SIREC-Q .....	117
Figura 3.30 Grafico Histórico de las Transacciones del SIRECQ .....	117
Figura 3.31 Reporte Histórico del SIRECQ .....	118
Figura 3.32 Configuraciones para pruebas de resistencia .....	118
Figura 3.33 Grafico de Prueba con 30000 Transacciones .....	119
Figura 3.34 Reporte con 30000 Transacciones .....	120
Figura 3.35 Grafico de Prueba con 1000000 de Transacciones .....	121
Figura 3.36 Reporte Histórico con 1000000 de transacciones .....	121
Figura 3.37 Configuraciones para pruebas de resistencia .....	122
Figura 3.38 Grafico de Prueba con 30000 Transacciones .....	123
Figura 3.39 Reporte con 30000 Transacciones .....	123
Figura 3.40 Grafico de Prueba con 1000000 Transacciones .....	124
Figura 3.41 Reporte con 1000000 Transacciones .....	125
Figura 3.42 Generación de Gráfica por Rangos .....	127
Figura 3.43 Generación de Grafica en tiempo real.....	127
Figura 3.44 Generación de Gráfica Histórica.....	128
Figura 3.45 Ponderado de Aceptación del Sistema .....	129
Figura 3.46 Ponderación de Resistencia a la Carga.....	133
Figura 3.47 Gráfico porcentual funcionalidad.....	134
Figura 3.48 Ponderación final .....	136

## LISTA DE TABLAS

Tabla 1.1 Comparación entre metodologías tradicionales y ágiles .....	5
Tabla 2.1 Historias de Usuario .....	18
Tabla 2.2 Product Backlog .....	20
Tabla 2.3 Selección de tareas del Sprint Cero .....	22
Tabla 2.4 Selección de tareas del Primer Sprint .....	24
Tabla 2.5 Selección de tareas del Segundo Sprint .....	26
Tabla 2.6 Selección de tareas del Tercer Sprint .....	29
Tabla 2.7 Selección de tareas del Cuarto Sprint .....	31
Tabla 2.8 Entidades y Atributos del modelo físico de la base de datos .....	34
Tabla 2.9 Dominio de Atributos de la BDD .....	35
Tabla 2.10 Estado de aceptación del Primer Sprint .....	91
Tabla 2.11 Estado de aceptación del Segundo Sprint .....	92
Tabla 2.12 Estado de aceptación del Tercer Sprint .....	92
Tabla 2.13 Estado de aceptación del Cuarto Sprint .....	93
Tabla 2.14 Reunión de Retrospectiva Primer Sprint .....	94
Tabla 2.15 Reunión de Retrospectiva Segundo Sprint .....	94
Tabla 2.16 Reunión de Retrospectiva Tercer Sprint .....	95
Tabla 2.17 Reunión de Retrospectiva Cuarto Sprint .....	96
Tabla 3.1 Resultados prueba de resistencia módulo reportes .....	119
Tabla 3.2 Resultados prueba de resistencia módulo histórico .....	120
Tabla 3.3 Pruebas de rendimiento de reportes .....	122
Tabla 3.4 Pruebas de rendimiento de reportes .....	124
Tabla 3.5 Valoración de Aceptabilidad .....	125
Tabla 3.6 Transacciones Pruebas Aceptación .....	126
Tabla 3.7 Valoración de Aceptación .....	128
Tabla 3.8 Valoración de Resistencia .....	130
Tabla 3.9 Pruebas de rendimiento de reportes .....	131
Tabla 3.10 Pruebas de rendimiento de reportes .....	132

Tabla 3.11 Valoración de Rendimiento .....	133
Tabla 3.12 Análisis de funcionalidad .....	133
Tabla 3.13 Análisis de usabilidad.....	135
Tabla 3.14 Resultados de aceptación.....	136

## RESUMEN

Este proyecto de titulación tiene como finalidad desarrollar un sistema web para la gestión del rendimiento de aplicaciones informáticas en un entorno de intranet. Primeramente se realizó un análisis de la situación actual del Distrito Metropolitano de Informática, mediante entrevistas a los jefes designados de las áreas relacionadas directamente con el caso de estudio de este proyecto que abarcó: el área de redes, el departamento de ingeniería de soluciones y el departamento de catastros.

En base a los requerimientos del proyecto y al análisis desarrollado; se decidió seleccionar como metodología para la elaboración del proyecto a SCRUM. Después de esto se elaboraron las historias de usuario, las cuales pasaron a formar el "Product Backlog". Para conseguir el producto final se implementó cinco sprints de alrededor de 15 días cada uno; al momento de finalizar cada sprint se realizaron pruebas funcionales en conjunto con el Product Owner para cumplir los requisitos implementados.

Para la elaboración del producto se ha utilizado herramientas como lo son: SQL Server el cual sirve como motor de base de datos, Visual Studio como entorno de desarrollo e Internet Information Server como servidor web. Cumplimiento con estas herramientas la planificación prevista para el desarrollo del producto final.

Como caso de estudio, el producto final fue instalado y probado por los usuarios del Distrito Metropolitano de Informática, concluyendo que el sistema tiene una aceptabilidad del 92.25% y un resultado de funcionalidad del 93.57%; adicionalmente opinaron que el producto tiene un menú intuitivo y de fácil acceso a los contenidos que presenta actividades del sistema.

# CAPÍTULO 1

## 1 DESCRIPCIÓN DEL PROBLEMA

### 1.1 ANALIZAR TRANSACCIONES DE APLICATIVOS EN UNA INTRANET

Una intranet permite el acceso a usuarios identificados dentro de la empresa; permitiendo una interrelación en la cual los usuarios y recursos puedan relacionarse entre sí. Las organizaciones de esta manera entregan a su personal recursos de forma eficiente; por tanto a través del uso de un intranet, se facilita la instalación y puesta en marcha del software institucional facilitando el trabajo del área de TI.

Otra de los beneficios que ofrecen las intranets a las organizaciones es la seguridad y privacidad de los diversos medios de comunicación en la empresa, lo cual logran con el uso de servidores de seguridad y tecnologías de cifrado que permite proteger la confidencialidad y seguridad de la información de posibles ataques externos. A diferencia de ambientes de trabajo abiertos como internet, con una intranet el personal no debe preocuparse de las limitaciones en el ancho de banda y accesibilidad para obtener herramientas y documentos electrónicos.

Sin embargo, existen diversos factores que pueden ser negativos en el uso de una intranet empresarial; como la falta de información actualizada y uso de herramientas mal gestionadas. También se tiene factores externos como lo son agentes mal intencionados que generan un riesgo para la seguridad de la información.

En síntesis una intranet empresarial permite generar un rendimiento favorable, ya que facilita la comunicación entre diversas áreas, además favorece la

disponibilidad de recursos y ofrece uso exclusivo solo a usuarios definidos dentro de la organización.

Las aplicaciones web o también llamadas en este caso de estudio como clientes, dentro de una intranet generan logs de sus transacciones los cuales resultan ser muy numerosos en cantidad, estos se almacenan en los servidores de la institución; por tanto su análisis resulta ser una tarea pesada por la proporción que pueden llegar a tener. Además tomando en cuenta la cantidad de aplicaciones informáticas que se presentan en el Distrito Metropolitano de Informática también llamado DMI, se tiene al final un gran lote de transacciones registradas las cuales son difíciles de analizar; por consiguiente es en este entorno donde se evidencia la necesidad de una nueva forma de análisis de aplicaciones para una buena gestión de productos que son propiedad de la DMI.

Partiendo de esto, las aplicaciones web en cada transacción que realizan brindan cierta información que puede ser tomada y aprovechada para un correcto análisis; información del cliente como: hora, fecha, IP, el nombre de máquina, etc. Esta información es la base para este caso de estudio. Capturando todos estos eventos y procesándolos se tiene datos actuales e históricos del trabajo del cliente. Con lo cual se tiene un conjunto amplio de información; es en este momento donde se implementa la solución propuesta y el método de captura de datos se lo realiza en transacciones específicas de cada uno de los clientes a ser analizados.

## **1.2 SITUACIÓN ACTUAL DE LAS ORGANIZACIONES EN EL AMBITO DEL RENDIMIENTO DE SUS APLICACIONES.**

Basados en las entrevistas a los jefes designados de las áreas relacionadas directamente con el caso de estudio [1], se tiene que en la DMI hoy en día no cuenta con algún medio de monitoreo aplicado a sus aplicaciones informáticas, además que la monitorización del rendimiento de los aplicativos se ha convertido en una necesidad, ya que con el paso del tiempo y los cambios sufridos a nivel de

código por nuevos requerimientos de los sistemas informáticos; se hace evidente un medio para monitorear el rendimiento de estos clientes. Actualmente la DMI necesita que sus aplicaciones tengan un tiempo de respuesta óptimo y que no se produzcan caídas de sistemas por la sobrecarga llegando a ser confiables.

Para medir el rendimiento de sus aplicaciones informáticas la DMI ha recurrido a una gestión tradicional para la medición de rendimiento de sus aplicaciones. A continuación se muestran unos ejemplos:

- Dar una calificación al monitoreo de la experiencia con el usuario final.
- Medir el rendimiento de los elementos de hardware como el de software que estén relacionados con el rendimiento del aplicativo informático.
- El análisis de datos recopilados en los servidores generados por las aplicaciones ayudan a encontrar patrones de errores, áreas de conflictos y definir posibles problemas [1].
- Las reiteradas llamadas telefónicas, mensajes instantáneos y diálogos con el usuario son las principales formas de identificar falencias en los aplicativos web, dando así una noción del rendimiento de cada aplicativo.

Con estos métodos tradicionales se evidencia la necesidad de un servicio que proporcione una visión más amplia del rendimiento de modo que vaya más allá del centro de datos de una empresa y ofrezca métricas en tiempo casi real e independiente de cada aplicativo de la organización.

### **1.3 JUSTIFICACIÓN DE LA METODOLOGÍA A UTILIZAR**

En el mundo de hoy en día no se puede manejar el desarrollo de software con metodologías tradicionales o previsibles, ya que no están preparadas para cambios no planificados. Usar diferentes tipos de metodologías de desarrollo de software se ha convertido en una decisión importante para lograr la calidad en el proceso que implican las actividades de una empresa.

Actualmente en el mercado se encuentran diferentes tipos de metodologías tradicionales y ágiles, se describe en la Tabla 1.1 se diferencia las principales características de estas metodologías.

**Tabla 1.1 Comparación entre metodologías tradicionales y ágiles**

Metodologías tradicionales	Metodologías ágiles
Se basan en normas y estándares del entorno de desarrollo	Se basan en heurísticas de prácticas de generación de código
El proceso es muy controlado al contar con demasiadas normas	Procesos menos rigurosos al tener pocas normas
Poca resistencia a los repentinos cambios	Listo para cambios durante el proyecto
Contrato establecido	Contrato muy flexible
Grupos grandes de personas	Grupos pequeños de personas
El cliente puede interactuar con los desarrolladores por medio de reuniones	El cliente es parte del equipo de desarrollo
Mayor énfasis en la arquitectura de software	Menor énfasis hacia la arquitectura de software

Fuente: Adaptado de [www.lecciones-aprendidas.com](http://www.lecciones-aprendidas.com). Modificado por los Autores

Basándose en esta información se determina que lo más viable es seguir los lineamientos que brindan las metodologías ágiles, ya que sus características contienen facilidades y comodidades en la elaboración de este proyecto.

La metodología ágil a seguir para el desarrollo de este proyecto de titulación es el conocido marco de trabajo ágil SCRUM. Ya que esta metodología está alineada perfectamente en tiempo efectivo con las necesidades del cliente y el equipo de trabajo, también al cliente se lo considera como parte del equipo del proyecto; además cuenta con la flexibilidad y adaptación a repentinos cambios reduciendo errores y controlando la calidad del producto en sus diferentes entregas hasta el producto final, satisfaciendo las expectativas del cliente por medio de resultados anticipados de forma continua.

### 1.3.1 METODOLOGÍA SCRUM

La metodología Scrum sirve para gestionar proyectos, además se basa en dos principios fundamentales:

- Desarrollo rápido y cambiante.
- Ciclos de desarrollo cortos [2].

Scrum evita la generación excesiva de documentación, por lo cual se puede iniciar el proyecto prescindiendo de esta. La documentación existente en SCRUM está orientada a la obtención de un producto visible en el cual el cliente pueda evaluar los avances del proyecto conforme se van cumpliendo los requerimientos en su totalidad [3]. En general en esta metodología existen dos elementos primordiales:

- Actores: personas las cuales cumplen con el proceso de desarrollo.
- Acciones: las tareas a realizarse con las que cuenta cada iteración (SPRINT).

También en esta metodología se define 3 actores:

- El propietario del producto (Product Owner), el cual define las prioridades del proyecto o producto.
- El administrador Scrum (Scrum Master), el cual asegura el correcto seguimiento de la metodología por parte del equipo.
- El equipo de Scrum (Scrum team) los cuales se encargan de implementar las funcionalidades del propietario en el producto.

Las acciones básicas de la metodología Scrum son:

- Pila del producto (Product Backlog): es una lista de tareas, funcionalidades o requerimientos definidos por el propietario del producto el cual se encargada de mantener esta lista y marcar las prioridades del proyecto [4].

- Pila del Sprint (Sprint Backlog): es un listado de tareas obtenido de la pila del producto. Estas tareas deben ser realizadas en un periodo de 2 a 4 semanas. Cuando ya se ha iniciado la pila del Sprint, no puede ser alterada o modificada. Se debe esperar al final del desarrollo del Sprint actual y luego modificar la tarea deseada, y pasaría a ser parte de una nueva pila del Sprint [5].
- La reunión diaria de SCRUM (Daily SCRUM Meeting), es una tarea realizada diariamente mientras dura el desarrollo de la pila Sprint, esta reunión es de máximo 30 minutos. Los temas principales temas a tratar son:
  - Tareas que se han realizado desde la anterior reunión.
  - Tareas a realizar en ese día.
  - Problemas u obstáculos para realizar alguna tarea determinada.

Al final de cada SPRINT se realiza una reunión para realizar un análisis del cumplimiento de los objetivos.

Al final de la culminación del último SPRINT se tiene una reunión realizando un análisis para saber si se cumplieron los objetivos iniciales que se fijaron, esto se lo realiza con el propietario del producto. No obstante solo se marcarán los aspectos positivos sino también los negativos que sucedieron durante todos los SPRINT's, en este punto se debe tener un documento el cual pueda ser evaluado por el propietario del producto [3].

Un proyecto se da por concluido cuando todos los SPRINT's se han finalizado y no queda ningún requerimiento por cumplir.

### **1.3.2 RAZONES PARA UTILIZAR SCRUM**

Existen muchas metodologías orientadas al desarrollo de aplicaciones web sin embargo se ha seleccionado SCRUM por las siguientes razones:

- La web es muy cambiante a pasos muy rápidos por lo que se necesita de un desarrollo ágil para que las aplicaciones no queden en la obsolescencia incluso antes de que se ponga a producción. SCRUM plantea un desarrollo altamente veloz por lo que soluciona este inconveniente.
- Los proyectos muchas veces son informales, por lo que a veces la documentación queda incompleta. SCRUM deja como una posibilidad hacer la documentación lo cual beneficia a un proyecto altamente veloz.
- Al desarrollar aplicaciones se tiene un equipo heterogéneo como son programadores, diseñadores, etc. Al tener una gran cantidad de reuniones planteadas por la metodología mejora la comunicación entre los mismos.
- SCRUM permite realizar SPRINTs en el que se definen tareas lo que lo hace iterativo, esto ayuda ya que un aplicación no deba esperar a estar desarrollada completamente para ponerla a producción más bien se la puede desarrollar por partes (Sprint) e ir actualizándola a medida que se realicen más iteraciones.

#### **1.4 JUSTIFICACIÓN DE LAS HERRAMIENTAS A UTILIZAR**

Se ha escogido como pilar para la construcción del sistema, herramientas propietarias de Microsoft ya que por su documentación, disponibilidad, fácil uso y lenguaje de programación orientados a objetos hacen a estas herramientas muy atractivas para su selección. Además para su correcto uso se requiere de una persona (súper administrador) con capacidades de configuración adaptables a estas tecnologías [6].

Por este motivo, las plataformas, arquitecturas y herramientas utilizadas para la construcción de este sistema informático serán las siguientes:

- Plataforma: ASP.Net
- Motores de Base de Datos: SQL Server 2008 o superior
- Lenguaje: C#

Se describe cada una de las herramientas listadas:

### 1.4.1 PLATAFORMA MICROSOFT .NET

Es una plataforma de desarrollo la cual integra un conjunto de herramientas, lenguajes de programación, un entorno de desarrollo visual y múltiples tecnologías entre las cuales se tiene ASP.NET, ADO.NET, LINQ, WPF, Silverlight, etc. Además que esta herramienta nos brinda un entorno de desarrollo con muchas tecnologías [7] y está dirigida a la creación de sistemas informáticos web, escritorio, consola y servicios web, es una herramienta con mucha documentación para su correcta utilización.

#### 1.4.1.1 Ventajas

**Fácil Uso:** Las clases que maneja esta plataforma tienen una sintaxis sencilla y completa además de un conjunto de métodos los cuales facilitan el trabajo del programador [8].

**Multilenguaje:** Soporta múltiples lenguajes, en los cuales además de ofrecer independencia, ofrece integración entre ellos [9].

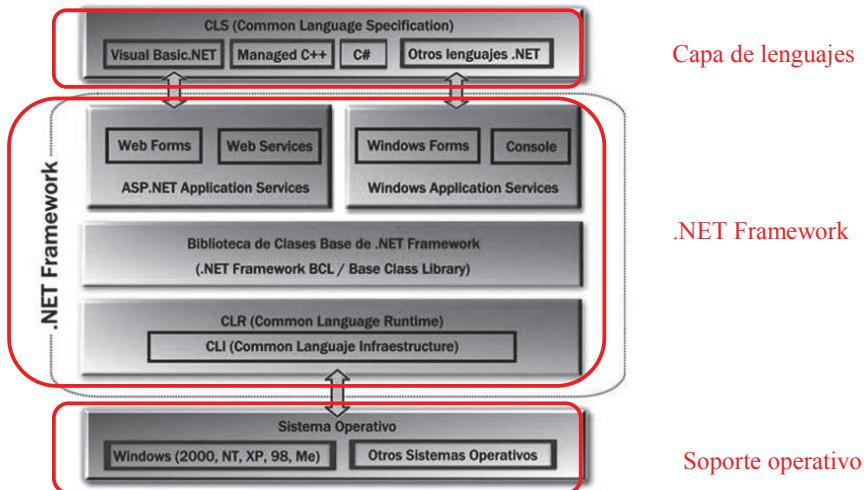
**Orientado a objetos:** Esto se da ya que admite los conceptos de encapsulamiento, herencia y polimorfismo. Además del uso de firmas de métodos encapsulados, descriptores de acceso, atributos de tiempo de ejecución, comentarios en línea y LINQ.

**Interoperabilidad:** Ya que sus aplicaciones son compatibles con dispositivos de escritorio así como con dispositivos móviles, además de ser compatibles con cualquier navegador.

#### 1.4.1.2 Arquitectura

Se muestra el Framework para aplicaciones web de Microsoft, usado para el desarrollo de la aplicación en la Figura 1.1.

**Figura 1.1 Marco de trabajo de ASP.NET**



Fuente: Ramírez Felipe, Aprenda practicando ASP.NET usando visual studio 2012, 2012, pág. 9  
 Modificado: Por los Autores

Lo que caracteriza a este Framework es la capa de servicios y recursos de .Net la cual brinda una intercomunicación entre programas, la biblioteca de clases que ofrece funcionalidad intrínseca en el lenguaje con sus más de 6000 clases [10], se tiene también el Motor de ejecución el cual hace de compilador, encargándose en resumen de todo lo relacionado a la administración en tiempo de ejecución de aplicaciones.

#### 1.4.2 ASP.NET

Es la parte dentro del Framework .Net que permite el desarrollo y la ejecución de aplicaciones y servicios Web [6] esto se ejecuta del lado del servidor y también proporciona un modelo de desarrollo web unificado necesario para crear aplicaciones web.

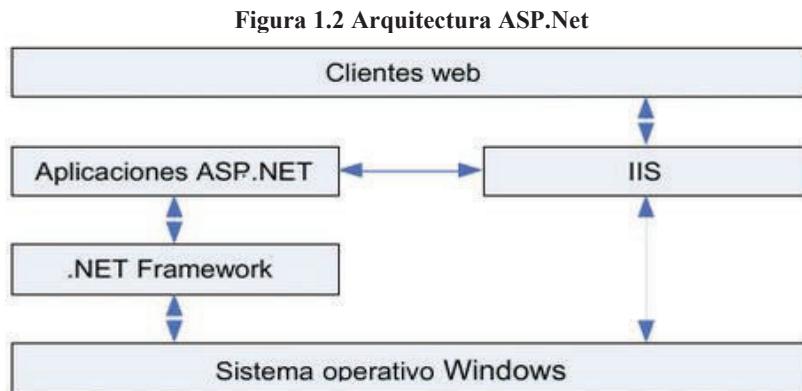
#### 1.4.2.1 Ventajas

ASP.Net presenta una cantidad considerable de cualidades las cuales son resumidas en:

- Permite la separación de la capa de presentación de la lógica de negocio.
- La compilación de páginas se guarda siempre para ser reutilizada provocando un mayor rendimiento.
- Permite la actualización de ficheros que conforman la aplicación [12].

#### 1.4.2.2 Arquitectura ASP.NET

Cuando una página ASP.NET implementa código correspondiente a la lógica de negocio [11], lo ejecutara a través del Framework de .Net y utilizara Internet Information Services (IIS) para proporcionar el resultado como código HTML al cliente web como se muestra en la Figura 1.2.



Fuente: Ceballos Francisco, Enciclopedia de Microsoft Visual C#: interfaces gráficas y aplicaciones para Internet con Windows Forms y ASP.NET, 2012, pag 743

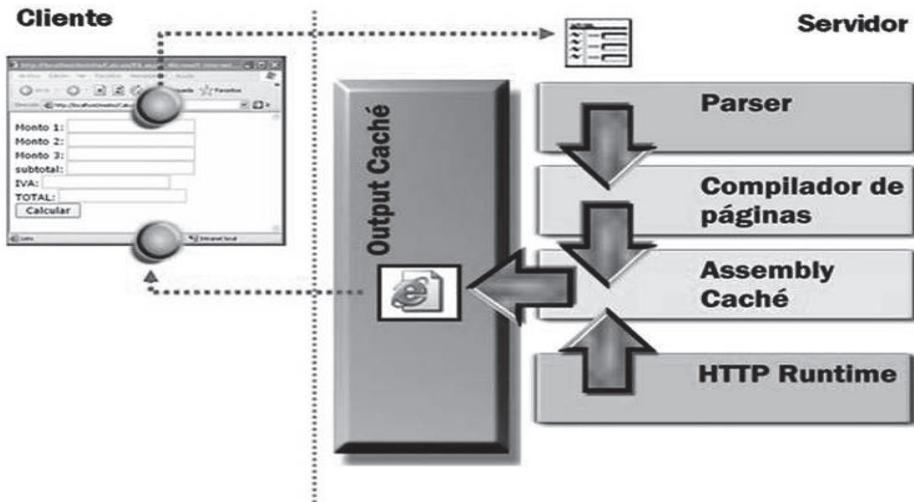
#### 1.4.2.3 Procesamiento de Páginas Web en ASP.NET

ASP.NET procesa las páginas de la siguiente manera:

1. Inicialmente se hace una petición al servidor el cual verifica que la página ASPX no posea errores de sintaxis, semántica o posea errores de referencia. Para esto se usa un analizador de código [12].
2. Si el programa no posee errores de compilado se genera un ensamblado el cual produce código HTML, este código puede ser interpretado por el explorador web del cliente.
3. El ensamblado es guardado en el (Assembly Cache).
4. Luego se solicita al HTTP Runtime la ejecución de clases y generación de código HTML, el cual es compatible con el explorador web del solicitante.
5. El contenido web es guardado en el cache de salida, el cual almacena los contenidos HTML, scripts del cliente y los datos de salida.
6. El contenido web es enviado al cliente.

Si se vuelve a solicitar el contenido web, el proceso se realiza a partir del cache de salida, si no existe el contenido, se ejecuta nuevamente clase y la generación del contenido web.

**Figura 1.3 Procesamiento de páginas web en ASP.NET**



Fuente: Felipe Ramírez, Practicando ASP.NET Usando Visual Studio, 2013, pág. 12

#### 1.4.3 MODELO MVC

El modelo de arquitectura Model-View-Controller (MVC) separa una aplicación en tres componentes principales: el modelo, la vista y el controlador.

El marco de ASP.NET MVC proporciona una alternativa al modelo de formularios Web Forms de ASP.NET para crear aplicaciones web.

El marco de ASP.NET MVC es un marco de presentación de poca complejidad y fácil de comprobar que [13] se integra con las características de ASP.NET existentes, como páginas maestras y la autenticación basada en pertenencia.

#### **1.4.3.1 Ventajas**

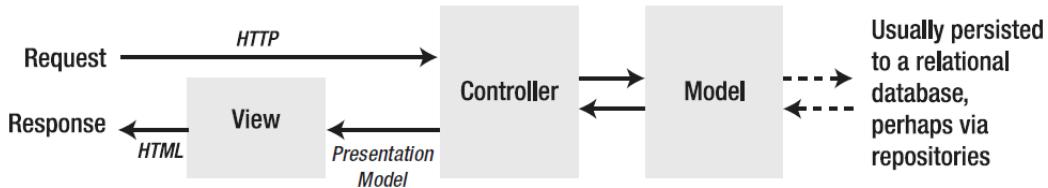
Las principales ventajas del modelo MVC son las siguientes:

- Ayuda de interfaz de usuario con estructura automática y plantillas personalizadas.
- La validación del modelo es basada en atributos tanto en el cliente y el servidor.
- Fuertemente ayudado por la escritura HTML.
- Múltiples herramientas en Visual Studio.

#### **1.4.3.2 Arquitectura MVC**

En esta arquitectura, las solicitudes se dirigen a una clase controlador, que procesa la entrada del usuario y funciona con el modelo de dominio para manejar la petición. Mientras que la capa de modelo tiene la lógica de dominio al acceso de datos, los controladores tienen lógica de la aplicación, como la navegación a través de un proceso para devolver los resultados a la vista. Cuando sea el momento para producir una interfaz [14] visible para el usuario, el controlador prepara datos para ser mostrados, selecciona un punto de vista, y lo deja disponible para su utilización como se muestra en la Figura 1.4.

**Figura 1.4 Arquitectura MVC 2 Para la WEB**



Fuente: Steven Sanderson, Pro ASP.NET MVC 2 Framework, 2010, pag 47

#### 1.4.4 MOTOR DE BASE DE DATOS SQL SERVER

El motor de acceso a bases de datos también es un requerimiento para la creación de la aplicación, ya que cuenta con un motor de base de datos el cual es propietario de Microsoft [15] y se adapta perfectamente con Visual Studio.

Las bases de datos relacionales nos brindan la integridad [16] de información necesaria, además de ser un perfecto complemento para desarrollar aplicaciones con Visual Studio.

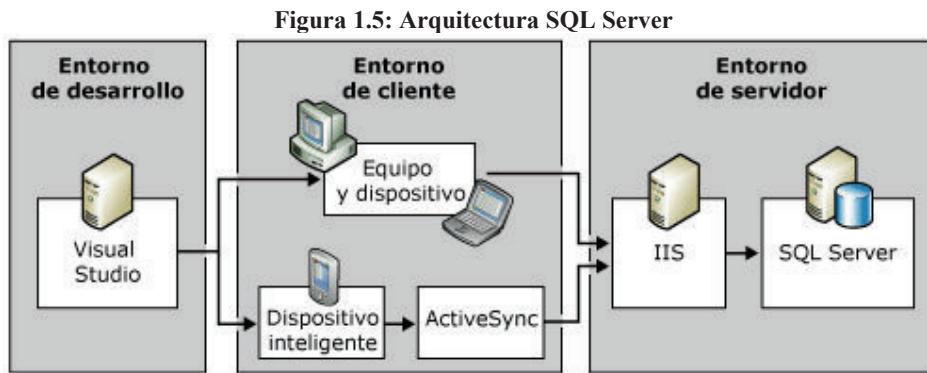
##### 1.4.4.1 Ventajas

Entre las principales ventajas que ofrece el motor de bases de datos SQL tenemos:

- Permite la creación de bases de datos relacionales.
  - Procesamiento de transacciones en línea.
  - Permite crear objetos de base de datos como son procedimientos almacenados, vistas e índices para administrar, ver y proteger los datos.
- [17]

##### 1.4.4.2 Arquitectura Motor de base de datos SQL Server

En la arquitectura de SQL Server, el entorno de cliente se compone de uno o varios equipos y dispositivos compatibles, en los que se implementa la aplicación que tiene a SQL Server como entorno de servidor de base de datos [18]. Se presenta la arquitectura de SQL Server.



Fuente: Tomado de [technet.microsoft.com](http://technet.microsoft.com). Modificado por los autores

#### 1.4.5 LENGUAJE DE PROGRAMACIÓN C#

Microsoft C# es un lenguaje de programación diseñado para crear un amplio número de aplicaciones empresariales que se ejecutan bajo la arquitectura de .NET Framework. Es sencillo, moderno, proporciona seguridades y está orientado a objetos. El código creado mediante C# se compila como código administrado, lo cual significa que se beneficia de los servicios de Common Language Runtime [19]. Estos servicios incluyen interoperabilidad entre lenguajes, recolección de elementos no utilizados, mejora la seguridad y compatibilidad entre versiones.

##### 1.4.5.1 Ventajas

Este lenguaje de programación muestra las principales ventajas.

- Gran robustez, gracias a la recolección de elementos no utilizados en cuanto a liberación de memoria.
- Seguridad implementada por medio de mecanismos de confianza intrínsecos del código.
- Plena compatibilidad con conceptos de metadatos extensibles.
- Language-Integrated Query (LINQ) que proporciona funciones de consulta integradas en una gran variedad de orígenes de datos. [19]

## CAPÍTULO 2

### 2 DESARROLLO DEL SISTEMA UTILIZANDO SCRUM

#### 2.1 ESPECIFICACIÓN DE REQUISITOS – HISTORIAS DE USUARIO (PRODUCT BACKLOG)

Para definir las historias de usuario se tiene como requisito la definición de perfiles de roles del sistema, ya que en base a ellos se define las actividades que cada historia de usuario debe cumplir.

##### 2.1.1 DEFINICIÓN DE PERFILES Y ROLES DEL SISTEMA

###### **Súper Administrador**

Tiene acceso al código del sistema así como a la base de datos física, esta persona es la encargada de realizar las respectivas configuraciones para el correcto funcionamiento del sistema. Este usuario es el encargado de la instalación y puesta en marcha de la DLL del sistema, por medio de la cual se realiza la captura de transacciones del Cliente.

###### **Administrador**

Esta persona tiene acceso a todas las funcionalidades del sistema:

1. Gestión de las ubicaciones.
2. Gestión de las configuraciones.
3. Acceso al módulo de Transacciones y Transacciones Históricas.
4. Generar reportes de Transacciones y Transacciones Históricas.

### **Cliente**

Son aquellos que generan transacciones las cuales funcionan bajo aplicativos.

Estas personas no tienen acceso al sistema. Además cada transacción que ellos generen es capturada por el aplicativo.

#### **2.1.2 HISTORIAS DE USUARIO**

Con el fin de capturar los requerimientos del sistema se pretende usar historias de usuario, las mismas que representan los deseos y funcionalidades del propietario del sistema.

Por medio de las reuniones diarias establecidas entre el Product Owner y los involucrados en el desarrollo del sistema, se han establecido las siguientes historias de usuario para la elaboración del sistema de monitoreo transaccional:

Tabla 2.1 Historias de Usuario

Identificador (ID) de la historia	Rol	Enunciado de la Historia		Criterio de Aceptación (Título)	Contexto	Criterios de Aceptación	
		Característica / Funcionalidad	Razón / Resultado			Evento	Resultado / Comportamiento esperado
HPS001	Como un Administrador	Necesito Un login	Con la finalidad de Registrar configuraciones personalizadas.	1 Registrar un nuevo usuario	En caso que El usuario no exista	cuando Necesita registrar un nuevo usuario	Registro exitoso del usuario
			Ingresar como usuario existente	2 Ingresar como usuario existente	En caso que El usuario desee acceder a la aplicación	cuando Necesita acceder al sistema	Acceso a los modulos del sistema
HPS002	Como un Administrador	Necesito Administrar Ubicaciones	Con la finalidad de Gestionar ubicaciones	1 Registrar una nueva ubicación.	En caso que No exista una ubicación	cuando Necesita obtener una nueva ubicación	Registro de nueva ubicación exitosa
				2 Registrar sub ubicaciones.	En caso que Exista una ubicación sin sub ubicaciones	cuando Necesite obtener sub ubicaciones	Registro de sub ubicación exitosa
				3 Edición de ubicaciones.	En caso que Requiera cambiar ubicaciones	cuando Necesite actualizar ubicaciones / sub ubicaciones	Edición de ubicación / sub ubicación exitosa
				4 Eliminar ubicaciones.	En caso que Existan ubicaciones / sub ubicaciones	cuando Necesite eliminar ubicaciones	Eliminación de ubicación exitosa

Identificador (ID) de la HPS	Rol	Característica / Funcionalidad	Enunciado de la Historia		Criterio de Aceptación (Título)	Contexto	Criterios de Aceptación	
			Razón / Resultado	Número o (#) de Escena			Evento	Resultado / Comportamiento esperado
HPS003	Como un Administrador	Necesito Administrar Configuraciones	Con la finalidad de Gestionar una configuración	1 Registrar una nueva configuración	En caso que No exista una configuración	cuando Necesita una configuracion personalizada	Registro de nueva configuración existosa	
				2 Edición de configuraciones	En caso que Requiera cambiar configuraciones	cuando Necesite editar configuraciones	Actualización de datos exitosa	
				3 Eliminación de configuraciones	En Caso que Exista una configuración	cuando Necesite eliminar ubicaciones	Eliminación de configuración exitosa	
HPS004	Como un Administrador	Necesito Generar Gráficas estadísticas	Con la finalidad de Medir el rendimiento de las aplicaciones	1 Generar gráficas de los clientes	En caso que Exista una configuración correcta	cuando Necesite analizar clientes en tiempo real	Creadión de gráficos por clientes	Creadión de gráfica exitosa
				2 Generar gráficas en tiempo real	En caso que Exista un cliente que genere eventos			
HPS005	Como un Administrador	Necesito Generar Reportes Estadísticos	Con la finalidad de Evaluar la disponibilidad y rendimiento de sistemas clientes	1 Generar Reportes globales de los clientes	En caso que Existas clientes configurados	cuando Necesita evaluar todos los clientes	Creadión de reportes globales Exitosa	
				2 Generar Reportes por cliente	En caso que Exista un cliente configurado	cuando Necesita evaluar un cliente específico		
				3 Generar Reportes por tiempo real	En caso que Existan clientes realizando eventos	cuando Necesita evaluar clientes del día actual		
HPS006	Como un Super Administrador	Necesito Configurar los clientes	Con la finalidad de Capturar Transaccion	1 Referenciar la DLL en el cliente.	En caso que exista un cliente.	cuando Necesita monitorear un cliente	Referencia correcta de la DLL	
				2 Instanciar metodo de captura de tranciones en el cliente	En caso de usar la DLL	cuando Necesita evaluar capturar transacciones	Customización del sistema exitosa	

Fuente: Adaptado de [www.navegapolis.net](http://www.navegapolis.net). Modificado por los Autores

### 2.1.3 PILA DEL PRODUCTO INICIAL (Product Backlog)

En base a las historias de usuario planteadas anteriormente se procede a realizar la pila del producto que es el conjunto de todas las características que darán funcionalidad a nuestro sistema. Ver Tabla 2.2.

**Tabla 2.2 Product Backlog**

Nº	Tarea	Complejidad	Prioridad	Historia de Usuario
1	Implementación de la Arquitectura de la solución	3	4	HPS001
2	Diseño de Diagrama Entidad-Relación	4	4	HPS001
3	Generación del modelo físico	3	4	HPS001
4	Generación de scripts SQL	2	3	HPS001
5	Implementación de procedimientos almacenados para control de datos	3	4	HPS001
6	Control de Calidad	4	5	HPS001
7	Afinación de la Base de Datos	5	4	HPS001
8	Definición de roles y perfiles del sistema	3	4	HPS001
9	Mapeo de las tablas y procedimientos almacenados de Usuarios	4	3	HPS002
10	Implementación de los servicios de Usuarios	4	4	HPS002
11	Implementación de controles para la gestión de Usuarios	3	3	HPS002
12	Implementación del formulario de ingreso	4	3	HPS002
13	Implementación del formulario registro de usuarios	3	4	HPS002
14	Mapeo de las tablas y procedimientos almacenados de Ubicación	2	3	HPS002
15	Implementación de los servicios de Ubicación	3	4	HPS002
16	Implementación de clases de acceso para la gestión de Ubicación	4	5	HPS002
17	Implementación del formulario de Ubicación	5	4	HPS002
18	Control de calidad de los formularios, servicios y controles	4	3	HPS002
19	Afinamiento de formularios, servicios y controles	3	4	HPS002
20	Implementación de pruebas	2	3	HPS002
21	Mapeo de las tablas y procedimientos almacenados de Configuración	3	4	HPS003
22	Implementación de los servicios de Configuración	4	5	HPS003
23	Implementación de clases de acceso para la gestión de Configuración	5	4	HPS003
24	Implementación del formulario de Configuración	3	4	HPS003
25	control de calidad del formulario de Configuración	4	3	HPS003
26	Afinamiento página de Configuración	4	4	HPS003
27	Implementación de pruebas	3	3	HPS003
28	Mapeo de procedimientos almacenados de Gráficas	4	3	HPS004
29	Implementación de Servicios y controles para la captura de transacciones	5	4	HPS004
30	Implementación del control web de gráficos Transacciones	3	4	HPS004

Nº	Tarea	Complejidad	Prioridad	Historia de Usuario
31	Implementación del formulario de Gráficas Transaccionales	3	4	HPS004
32	Implementación del control web de gráficos Históricos	4	4	HPS004
33	Implementación del formulario de Gráficas Históricas	4	4	HPS004
34	Control de calidad página de Gráficas	4	3	HPS004
35	Afinamiento página de Gráficas	3	4	HPS004
36	Implementación de pruebas	2	3	HPS004
37	Implementación reporte genérico de Transacción	3	4	HPS005
38	Implementación de formulario de reporte de Transacción	4	5	HPS005
39	Implementación reporte genérico Históricos	5	4	HPS005
40	Implementación de formulario de reporte Histórico	3	4	HPS005
41	Control de calidad página de Reportes	4	3	HPS005
42	Afinamiento página de Reportes	4	4	HPS005
43	Customización del Sistema	3	5	HPS005
44	Implementación de pruebas	3	3	HPS005

Elaborado por los Autores

## 2.2 PLANIFICACIÓN DE LOS SPRINTS (SPRINTS BACKLOG)

En la Figura 2.1 se observa la numeración de los Sprints y fechas establecidas para la realización del desarrollo del sistema mediante jornadas laborables de seis horas. Se presenta las tareas del Product Backlog con horas de esfuerzo para realizarlas y el avance de estas tareas.

**Figura 2.1 Planificación de los Sprints**

Proyecto			
Nombre del proyecto			
Nº de sprint	Inicio	Días	Jornada
0	25-jun.-15	14	6
1	15-jul.-15	19	6
2	11-agosto-15	16	6
3	2-sep.-15	19	6
4	29-jun.-15	16	6

TIPOS	TAREAS		EQUIPO	FESTIVOS
		ESTADOS		
Análisis	Pendiente	Lenin		
Codificación	En curso	Marcelo		
Prototipado	Terminada	Lenin, Marcelo		
Pruebas	Eliminada	William		
Reunión		Bolívar		
Documentación				

Fuente: Adaptado de [www.navegapolis.net](http://www.navegapolis.net). Modificado por los Autores

### 2.2.1 PLANIFICACIÓN DEL SPRINT CERO

Para la planificación del Sprint Cero se toma las tareas de la 1 a la 8 del Product Backlog, estas actividades tienen que ver con la creación de la base de datos, la creación de los procedimientos almacenados para la administración de las entidades utilizadas y todas las tareas que conllevan a la construcción del sistema.

Este Sprint inicia el 25 de junio del 2015 y finaliza el 14 de Julio del 2015, el cual tiene una duración de 14 días laborables.

La Tabla 2.3 muestra las tareas indicadas para este Sprint así como el tiempo requerido para por tarea. El Sprint muestra una duración de 120 horas.

**Tabla 2.3 Selección de tareas del Sprint Cero**

No	Tarea	Tiempo Estimado
1	Implementación de la Arquitectura de la solución	12
2	Diseño de Diagrama Entidad-Relación	12
3	Generación del modelo físico	6
4	Generación de scripts SQL	6
5	Implementación de procedimientos almacenados para control de datos	48
6	Control de Calidad	12
7	Afinación de la Base de Datos	12
8	Definición de roles y perfiles del sistema	12
<b>Total</b>		<b>120</b>

**Elaborado por los Autores**

En esta Figura 2.2 se puede verificar el listado de tareas indicadas, el establecimiento del tiempo asignado, sus estados y los responsables de estas tareas.

En la Figura 2.3 se puede observar el avance y el debido esfuerzo de las tareas necesarias para entregar el Sprint Cero, se define al esfuerzo como el trabajo individual o colectivo de los recursos sobreentendiendo a personas necesarias para la finalización de las tareas.

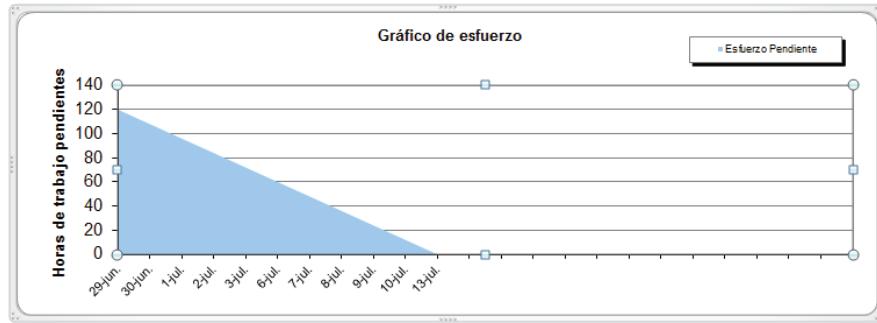
Figura 2.2 Tareas del Sprint Cero									
SPRINT	INICIO	DURACIÓN							
0	29-jun.-15	10							
Tareas pendientes	8	7							
Horas de trabajo pendientes	120	108							
L	M	X	J	V	L	M	X	J	V
29-jun.	30-jun.	1-jul.	2-jul.	3-jul.	6-jul.	7-jul.	8-jul.	9-jul.	10-jul.

**PILA DEL SPRINT**

Backlog	Tarea	Tipo	Estado	Responsable	
1	Implementación de la Arquitectura d	Análisis	Terminada	Lenin, Marcel	12
2	Diseño de Diagrama Entidad-Relació	Prototipado	Terminada	Lenin, Marcel	12
3	Generación del modelo físico	Codificación	Terminada	Lenin, Marcel	6
4	Generación de scripts SQL	Codificación	Terminada	Lenin, Marcel	6
5	Implementación de procedimientos a	Codificación	Terminada	Lenin, Marcel	48
6	Control de Calidad	Pruebas	Terminada	Lenin, Marcel	12
7	Afinación de la Base de Datos	Codificación	Terminada	Lenin, Marcel	12
8	Definición de roles y perfiles del s	Documentación	Terminada	Lenin, Marcel	12

Elaborado por los Autores

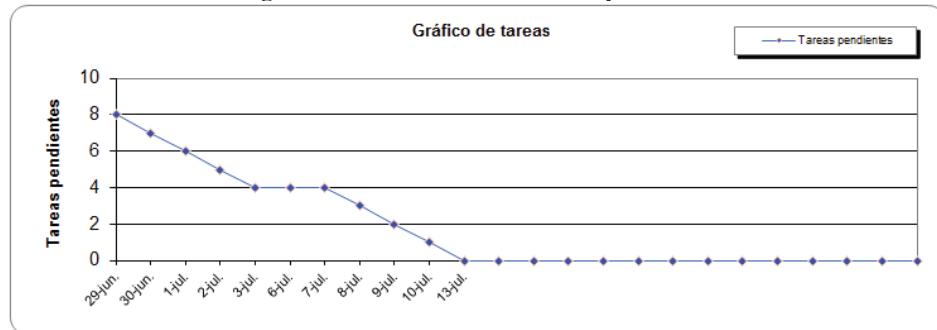
**Figura 2.3 Esfuerzo realizado en el Sprint Cero**



Elaborado por los Autores

En la Figura 2.4 se presenta el progreso de las tareas en los intervalos de tiempo, en el cual se presenta una fecha de inicio y una fecha final de las diferentes tareas del Sprint Cero a ser desarrolladas por el equipo de trabajo.

**Figura 2.4 Avances de tareas del Sprint Cero**



Elaborado por los Autores

Para ver la ejecución de este Sprint conforme a su planificación ver la sección [2.3.1](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

### **2.2.2 PLANIFICACIÓN DEL PRIMER SPRINT**

Para la planificación del Primer Sprint se toma las tareas de la 9 a la 20 del Product Backlog, estas actividades tienen que ver con la creación de páginas y formularios tanto para la entidad usuarios como para ubicación, el mapeo de los procedimientos almacenados para la administración de usuarios y ubicación.

Este Sprint inicia el 15 de julio del 2015 y finaliza el 7 de agosto del 2015, mostrando una duración de 19 días laborables.

La Tabla 2.4 muestra las tareas indicadas para este Sprint así como el tiempo requerido para por tarea. El Sprint muestra una duración de 180 horas.

**Tabla 2.4 Selección de tareas del Primer Sprint**

No	Tarea	Tiempo Estimado
9	Mapeo de las tablas y procedimientos almacenados de Usuarios	3
10	Implementación de los servicios de Usuarios	12
11	Implementación de controles para la gestión de Usuarios	6
12	Implementación del formulario de ingreso	24
13	Implementación del formulario registro de usuarios	24
14	Mapeo de las tablas y procedimientos almacenados de Ubicación	3
15	Implementación de los servicios de Ubicación	12
16	Implementación de clases de acceso para la gestión de Ubicación	12
17	Implementación del formulario de Ubicación	36
18	Control de calidad de los formularios, servicios y controles	12
19	Afinamiento de formularios, servicios y controles	24
20	Implementación de pruebas	12
<b>Total</b>		<b>180</b>

Elaborado por los Autores

En esta Figura 2.5 se puede verificar el listado de tareas realizadas, el establecimiento del tiempo, sus estados y los responsables de estas tareas.

Figura 2.5 Tareas del Primer Sprint																					
SPRINT	INICIO	DURACIÓN																			
1	17-jul-15	15																			
Tareas pendientes	V 17-jul.	L 18-jul.	M 19-jul.	X 20-jul.	J 21-jul.	V 22-jul.	L 23-jul.	M 24-jul.	X 25-jul.	J 26-jul.	V 27-jul.	L 28-jul.	M 29-jul.	X 30-jul.	J 31-jul.	V 1-agosto.	L 2-agosto.	M 3-agosto.	X 4-agosto.	J 5-agosto.	
Horas de trabajo pendientes	12	11	9	22	9	8	8	6	5	4	4	4	3	2	2	1					
	180	168	156	144	132	120	108	96	84	72	60	48	36	24	12						

Elaborado por los Autores

En la Figura 2.6 se puede observar el avance y el debido esfuerzo de las tareas necesarias para entregar el Primer Sprint, se define al esfuerzo como el trabajo individual o colectivo de los recursos sobreentendiendo a personas necesarias para la finalización de las tareas.

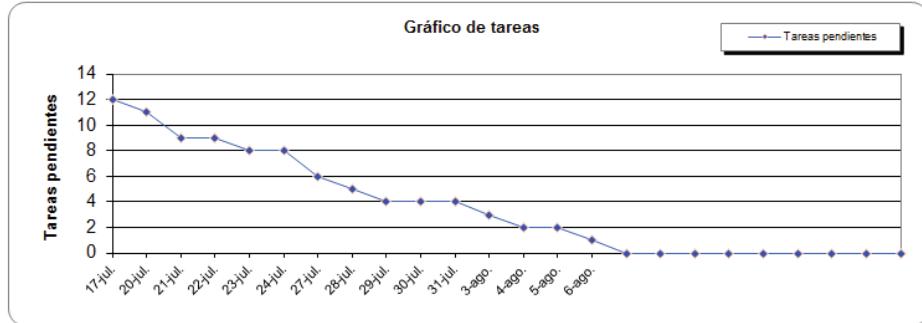
Figura 2.6 Esfuerzo realizado en el Primer Sprint



Elaborado por los Autores

En la Figura 2.7 se presenta el progreso de las tareas en los intervalos de tiempo, en el cual se presenta una fecha de inicio y una fecha final de las diferentes tareas del Primer Sprint a ser desarrolladas por el equipo de trabajo.

Para ver la ejecución de este Sprint conforme a su planificación ver la sección [2.3.2](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

**Figura 2.7 Avances de tareas del Primer Sprint**

Elaborado por los Autores

### 2.2.3 PLANIFICACIÓN DEL SEGUNDO SPRINT

Para la planificación del Segundo Sprint se toma las tareas de la 21 a la 27 del Product Backlog, estas actividades tienen que ver con la creación del formulario de la página de Configuración, el mapeo de los procedimientos almacenados para la administración de Configuraciones.

Este Sprint inicia el 11 de agosto del 2015 y finaliza el 1 de septiembre del 2015, mostrando una duración de 16 días laborables.

La Tabla 2.5 muestra las tareas indicadas para este Sprint así como el tiempo requerido para por tarea. El Sprint muestra una duración de 144 horas.

**Tabla 2.5 Selección de tareas del Segundo Sprint**

No	Tarea	Tiempo Estimado
21	Mapeo de las tablas y procedimientos almacenados de Configuración	6
22	Implementación de los servicios de Configuración	18
23	Implementación de clases de acceso para la gestión de Configuración	12
24	Implementación del formulario de Configuración	60
25	control de calidad del formulario de Configuración	12
26	Afinamiento página de Configuración	24
27	Implementación de pruebas	12
<b>Total</b>		<b>144</b>

Elaborado por los Autores

En esta Figura 2.8 se puede verificar el listado de tareas indicadas, el establecimiento del tiempo, sus estados y los responsables de las mismas.

**Figura 2.8 Tareas del Segundo Sprint**

SPRINT		INICIO	DURACIÓN												
2		13-agosto-15	12	J	V	L	M	X	J	V	L	M	X	J	
				13-agosto.	7	6	5	4	4	4	4	3	3	2	1
				14-agosto.	144	132	120	96	84	72	60	48	36	24	12
				15-agosto.											
				16-agosto.											
				17-agosto.											
				18-agosto.											
				19-agosto.											
				20-agosto.											
				21-agosto.											
				22-agosto.											
				23-agosto.											
				24-agosto.											
				25-agosto.											
				26-agosto.											
				27-agosto.											

Elaborado por los Autores

En la Figura 2.9 se puede observar el avance y el debido esfuerzo de las tareas necesarias para entregar el Segundo Sprint, se define al esfuerzo como el trabajo individual o colectivo de los recursos sobreentendiendo a personas necesarias para la finalización de las tareas.

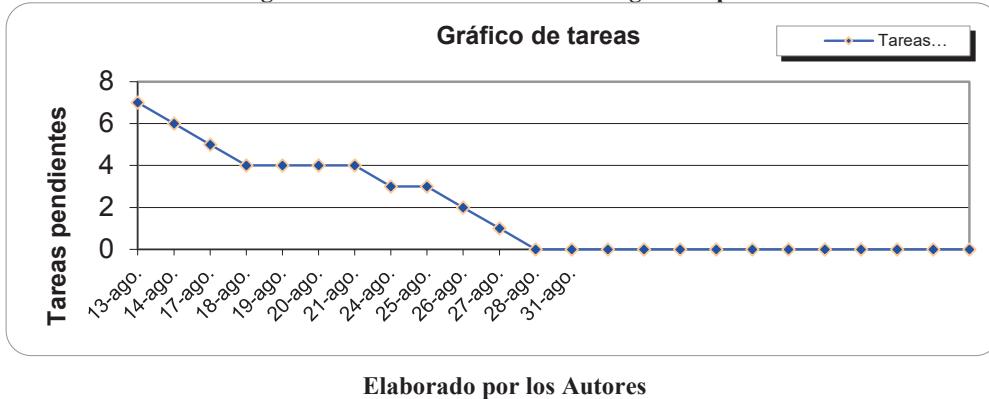
**Figura 2.9 Esfuerzo realizado en el Segundo Sprint**



Elaborado por los Autores

En la Figura 2.10 se presenta el progreso de las tareas en los intervalos de tiempo, en el cual se presenta una fecha de inicio y una fecha final de las diferentes tareas del Segundo Sprint a ser desarrolladas por el equipo de trabajo.

**Figura 2.10 Avances de tareas del Segundo Sprint**



Para ver la ejecución de este Sprint conforme a su planificación ver la sección [2.3.3](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

#### 2.2.4 PLANIFICACIÓN DEL TERCER SPRINT

Para la planificación del Tercer Sprint se toma las tareas de la 28 a la 36 del Product Backlog, estas actividades tienen con la generación de Gráficas para los clientes, el mapeo de los procedimientos almacenados para la generación de Gráficas y todas la tareas que conlleva la creación de los formularios y páginas de Graficas Transaccionales e Históricas. Este Sprint inicia el 2 de septiembre del 2015 y finaliza el 28 de septiembre del 2015, mostrando una duración de 19 días laborables.

La Tabla 2.6 muestra las tareas indicadas para este Sprint así como el tiempo requerido para por tarea. El Sprint muestra una duración de 180 horas.

En esta Figura 2.11 se puede verificar el listado de tareas indicadas, el establecimiento del tiempo, sus estados y los responsables de las mismas.

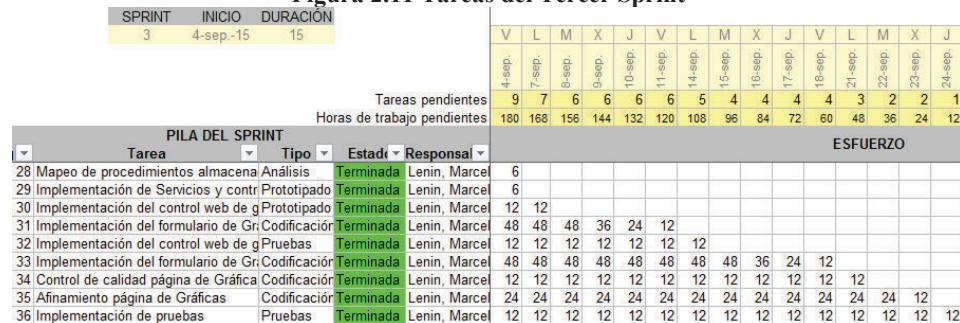
**Tabla 2.6 Selección de tareas del Tercer Sprint**

No	Tarea	Complejidad
28	Mapeo de procedimientos almacenados de Gráficas	6
29	Implementación de Servicios y controles para la captura de transacciones	6
30	Implementación del control web de gráficos Transacciones	12
31	Implementación del formulario de Gráficas Transaccionales	48
32	Implementación del control web de gráficos Históricos	12
33	Implementación del formulario de Gráficas Históricas	48
34	Control de calidad página de Gráficas	12
35	Afinamiento página de Gráficas	24
36	Implementación de pruebas	12
	<b>Total</b>	180

---

Elaborado por los Autores

## **Figura 2.11 Tareas del Tercer Sprint**

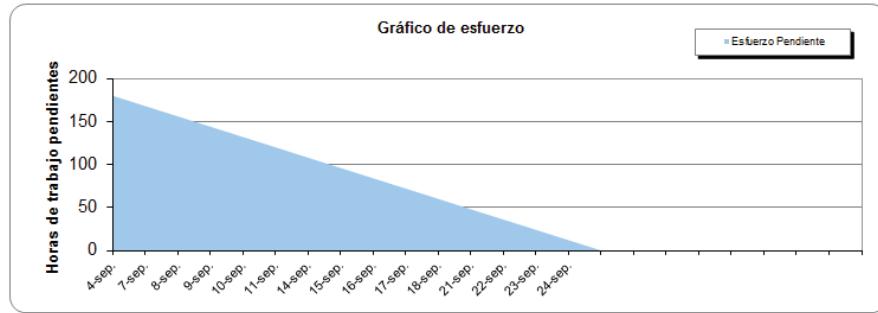


Elaborado por los Autores

En la Figura 2.12 se puede observar el avance y el debido esfuerzo de las tareas necesarias para entregar el Tercer Sprint, se define al esfuerzo como el trabajo individual o colectivo de los recursos sobreentendiéndose a personas necesarias para la finalización de las tareas.

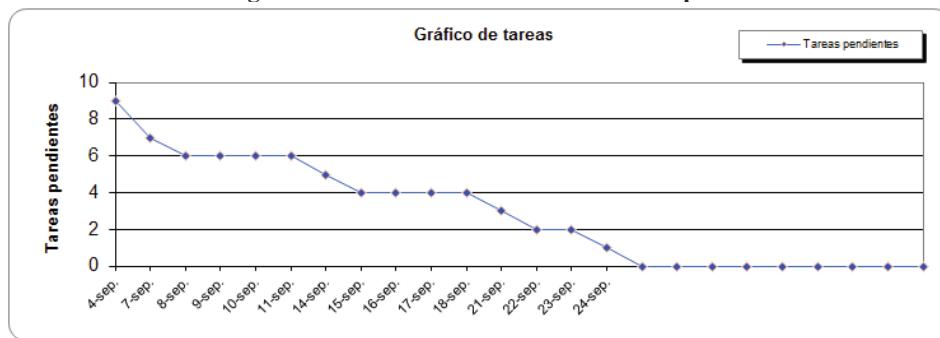
En la Figura 2.13 se presenta el progreso de las tareas en los intervalos de tiempo, en el cual se presenta una fecha de inicio y una fecha final de las diferentes tareas del Tercer Sprint a ser desarrolladas por el equipo de trabajo.

**Figura 2.12 Esfuerzo realizado en el Tercer Sprint**



Elaborado por los Autores

**Figura 2.13 Avances de tareas del Tercer Sprint**



Elaborado por los Autores

Para ver la ejecución de este Sprint conforme a su planificación ver la sección [2.3.4](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

## 2.2.5 PLANIFICACIÓN DEL CUARTO SPRINT

Para la planificación del Cuarto Sprint se toma las tareas de la 36 a la 44 del Product Backlog, estas actividades tienen que ver con el mapeo de los procedimientos almacenados para la generación de Reportes y todas la tareas que conlleva la creación Reportes Transaccionales e Históricos. Además se presenta los afinamientos finales de la aplicación con su puesta en marcha en el servidor local como carga de datos de inicio y pruebas finales.

Este Sprint inicia el 29 de septiembre del 2015 y finaliza el 20 de octubre del 2015, mostrando una duración de 16 días. La Tabla 2.7 muestra las tareas indicadas para este Sprint así como el tiempo requerido para por tarea. El Sprint muestra una duración de 144 horas.

**Tabla 2.7 Selección de tareas del Cuarto Sprint**

No	Tarea	Complejidad
37	Implementación reporte genérico de Transacción	36
38	Implementación de formulario de reporte de Transacción	12
39	Implementación reporte genérico Históricos	24
40	Implementación de formulario de reporte Histórico	12
41	Control de calidad página de Reportes	12
42	Afinamiento página de Reportes	24
43	Customización del Sistema	12
44	Implementación de pruebas	12
<b>Total</b>		<b>144</b>

Elaborado por los Autores

En esta Figura 2.14 se puede verificar el listado de tareas indicadas, el establecimiento del tiempo, sus estados y los responsables de las mismas.



Elaborado por los Autores

En la Figura 2.15 se puede observar el avance y el debido esfuerzo de las tareas necesarias para entregar el Cuarto Sprint, se define al esfuerzo como el trabajo individual o colectivo de los recursos sobreentendiendo a personas necesarias para la finalización de las tareas.

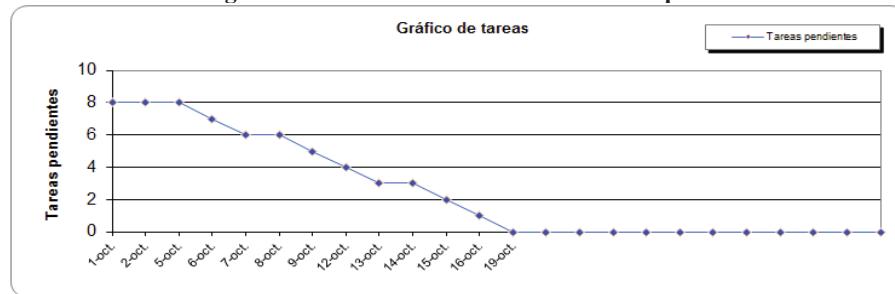
**Figura 2.15 Esfuerzo realizado en el Cuarto Sprint**



Elaborado por los Autores

En la Figura 2.16 se presenta el progreso de las tareas en los intervalos de tiempo, en el cual se presenta una fecha de inicio y una fecha final de las diferentes tareas del Cuarto Sprint a ser desarrolladas por el equipo de trabajo.

**Figura 2.16 Avances de tareas del Cuarto Sprint**



Elaborado por los Autores

Para ver la ejecución de este Sprint conforme a su planificación ver la sección [2.3.5](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

## 2.3 EJECUCIÓN DE LOS SPRINTS

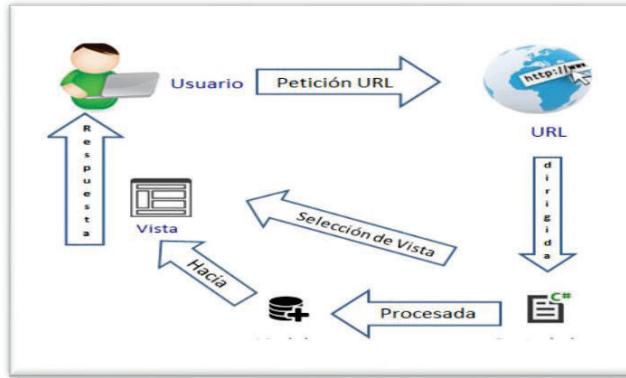
### 2.3.1 EJECUCIÓN DEL SPRINT CERO

#### 2.3.1.1 Arquitectura de la Solución

Para la realización de este proyecto se utiliza la arquitectura MVC (Modelo – Vista - Controlador), la cual además de facilitar la organización del proyecto permite la

integración de las herramientas a utilizar. En la Figura 2.17 se muestra la arquitectura MVC.

**Figura 2.17 Arquitectura de la Solución**

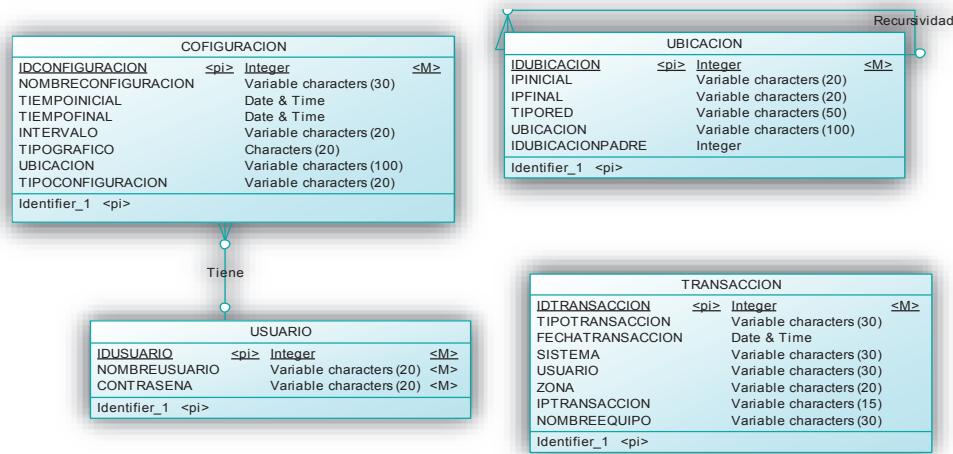


Elaborado por los autores

### 2.3.1.2 Diseño lógico de la base de datos

Una vez definida la arquitectura a usar, se procede con el diseño del modelo entidad relación de la base de datos, el cual es el punto de partida del proyecto ya que contiene la estructura de la funcionalidad global de la aplicación, ver modelo físico y lógico de la base de datos de la solución en la Figura 2.18.

**Figura 2.18 Modelo Entidad-Relación de la base de datos**

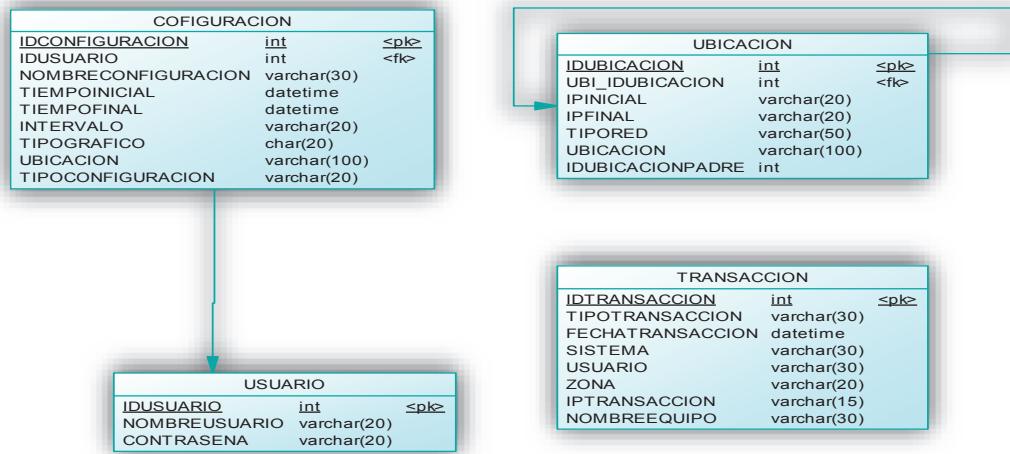


Elaborado por los autores

### 2.3.1.3 Diseño físico de la base de datos

Partiendo del modelo entidad relación se procede con la generación el modelo físico de la base de datos de la solución en la Figura 2.19.

**Figura 2.19 Diagrama modelo físico de la base de datos**



Elaborado por los autores

En la Tabla 2.8 se describe cada uno de los atributos y entidades del modelo físico.

**Tabla 2.8 Entidades y Atributos del modelo físico de la base de datos**

Nombre de Tabla	Atributos	Tipo	Almacenamiento
CONFIGURACION	IDCONFIGURACION (Clave Primaria)	Int	4 bytes
	IDUSUARIO (Clave Foránea)	Int	4 bytes
	NOMBRECONFIGURACION	Varchar (30)	-
	TIEMPOINICIAL	Datetime	8 bytes
	TIEMPOFINAL	Datetime	8 bytes
	INTERVALO	Varchar (20)	-
	TIPOGRAFICO	Varchar (20)	-
	UBICACION	Varchar (30)	-
USUARIO	TIPOCONFIGURACION	Varchar (20)	-
	IDUSUARIO (Clave Primaria)	Int	4 bytes
	NOMBREUSUARIO	Varchar (20)	-
	CONTRASENA	Varchar (20)	-
	IDUBICACION (Clave Primaria)	Int	4 bytes

UBICACION	IPINICIAL	Varchar (20)	-
	IPFINAL	Varchar (20)	-
	TIPORED	Varchar (50)	-
	UBICACIÓN	Varchar (100)	-
	IDUBICACIONPADRE(Clave Foránea)	Int	4 bytes
TRANSACCION	IDTRANSACCION (Clave Primaria)	Bigint	8 bytes
	TIPOTRANSACCION	Varchar (30)	-
	FECHATRANSACCION	Datetime	
	SISTEMA	Varchar (30)	-
	USUARIO	Varchar (30)	-
	ZONA	Varchar (20)	-
	IPTRANSACCION	Varchar (15)	-
	NOMBREEQUIPO	Varchar (30)	-

**Elaborado por los autores**

En la siguiente Tabla 2.9 se lista el dominio de los tipos de atributos del sistema.

**Tabla 2.9 Dominio de Atributos de la BDD**

	Datetime
Intervalo de fechas	Del 1 de enero de 1753 hasta el 31 de diciembre de 9999
Intervalo de horas	De 00:00:00 a 23:59:59.997
	Int
Intervalo	De 1 a 2^31-1 (2.147.483.647)
	Bigint
Intervalo	De 1 a 2^63-1 (9.223.372.036.854.775.807)
	Varchar
Intervalo	1 a 8000

**Elaborado por los autores**

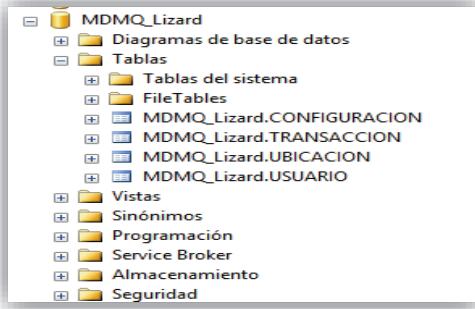
Nota: Para la elaboración de la base de datos del sistema se utilizó el motor de base de datos SQL Server 2008 Service Pack 1.

#### **2.3.1.4 Generación del Script para la creación de la base de datos.**

Una vez que se tiene el modelo físico de la Base de Datos se procede con la generación del Script, el cual permite la creación de la BDD. Incluido en Anexo Codificación.

Al terminar esto se ejecuta el Script, generando así la base de datos. Figura 2.20.

Figura 2.20 Generación de la BDD



Elaborado por los Autores

### 2.3.1.5 IMPLEMENTACIÓN DE PROCEDIMIENTOS ALMACENADOS PARA CONTROL DE DATOS

Partiendo de la base de datos se procede con la implementación de los procedimientos almacenados los cuales son de gran importancia ya que almacenan prácticamente toda la funcionalidad de nuestro aplicativo. Estos procedimientos almacenados contienen desde la inserción de un usuario en la base de datos hasta la generación de Reportes y Gráficos estadísticos lo cual es la meta del aplicativo. Los procedimientos almacenados para la inserción y acceso de un usuario ver figura 2.21 y 2.22, para ver la lista completa de procedimientos almacenados diríjase a anexo Codificación.

Figura 2.21 Procedimiento Almacenado Insertar Usuario

```

/*
----- Procedimiento
----- InsertarDatosUsuario
----- Procedimiento almacenado para insertar usuarios
----- use MDMQ_Lizard
go
If exists
(
SELECT * FROM sys.objects WHERE type = 'P' AND name = 'insertarDatosUsuario'
)
drop procedure MDMQ_Lizard.insertarDatosUsuario
go
Create proc MDMQ_Lizard.insertarDatosUsuario
(
    @NOMBREUSUARIO          varchar(20),
    @CONTRACENA              varchar(20)
)
as
If not exists
Select * from MDMQ_Lizard.USUARIO
where NOMBREUSUARIO=@NOMBREUSUARIO
begin
    insert into MDMQ_Lizard.USUARIO values (@NOMBREUSUARIO,@CONTRACENA)
end
go

```

Elaborado por los Autores

Implementación del procedimiento almacenado para realiza el ingreso de un usuario. Figura 2.22.

**Figura 2.22 Procedimiento Almacenado Acceso Usuario.**

```

/*
=====
Procedimiento
=====
logearUsuario
=====*/
--Procedimiento para checar logear un usuario
use MDMQ_Lizard
go
if exists
(
Select * from sys.objects where type = 'P' AND name='logearUsuario'
)
drop procedure MDMQ_Lizard.logearUsuario
go
create PROCEDURE MDMQ_Lizard.logearUsuario
(
    @NOMBREUSUARIO      varchar(20),
    @CONTRACENA         varchar(20)
)
AS
SELECT * FROM MDMQ_Lizard.USUARIO WHERE NOMBREUSUARIO=@NOMBREUSUARIO AND CONTRACENA=@CONTRACENA
go

```

Elaborado por los Autores

### 2.3.1.6 Sprint Cero Review

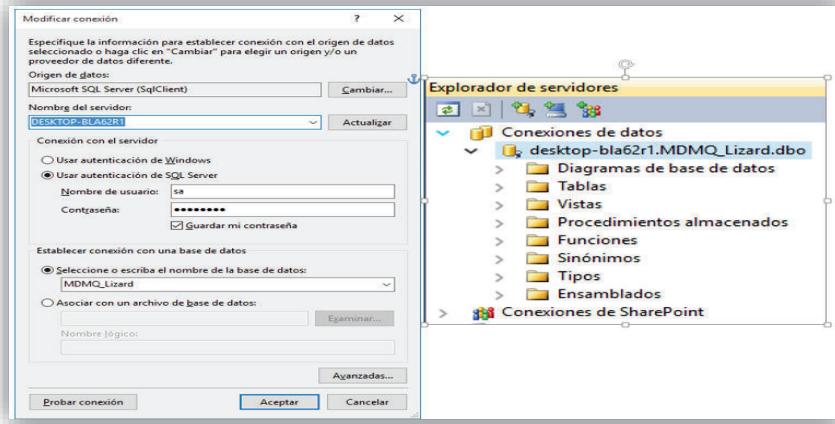
En esta iteración se realizaron todas las tareas técnicas necesarias para el arranque del proyecto, por lo tanto el equipo de SCRUM analiza cómo proceder en el siguiente Sprint para seguir completando las tareas correspondientes en el tiempo necesario.

Para ver la entrega de este Sprint con sus tareas completas en el plazo establecido revisar la sección [2.4.1](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

### 2.3.2 EJECUCIÓN DEL PRIMER SPRINT

En este Sprint inicia con la creación de una cadena de conexión la cual realizará el enlace entre la aplicación y la base de datos. Figura 2.23.

**Figura 2.23 Cadena de conexión**



Elaborado por los Autores

También se presenta la cadena de conexión del proyecto destacando que se hace referencia al servidor donde se está ejecutando la aplicación.

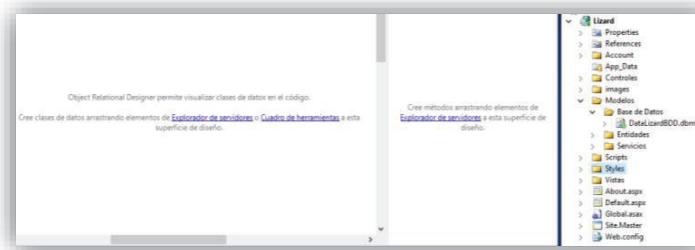
**Figura 2.24 Cadena de conexión del proyecto**

```
<connectionStrings>
  <add name="MDMQ_LizardConnectionString" connectionString="Data Source=172.22.16.73\SQLMDMQ2008;
    Initial Catalog=MDMQ_Lizard;Persist Security Info=True;User ID=migracion;Password=migra$2k12"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

Elaborado por los Autores

Partiendo de la cadena de conexión se procede con la creación de una clase LINQ, la cual facilitara el mapeo de tablas y procedimientos almacenados los cuales realizan las funcionalidades del aplicativo. Figura 2.25.

**Figura 2.25 Clase LINQ DataLizardBDD.dbml**

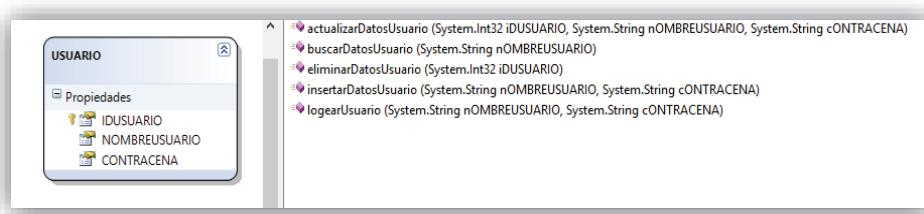


Elaborado por los Autores

### 2.3.2.1 CAPA MODELO USUARIO

En esta capa procedemos con el mapeo de la tabla y procedimientos almacenados del usuario con el objeto de instanciarlos como entidades y métodos. Para esto se usó nuestra clase LinQ “DataLizardBDD.dbml” la cual permite dicho proceso. Figura 2.26.

**Figura 2.26 Mapeo de Tabla y Procedimientos Almacenados de Usuario**



Elaborado por los Autores

A partir de esto se implementa la clase *ServicioUsuario.cs* en la cual se personaliza el tipo de retorno de datos de los procedimientos almacenados de USUARIO. El código respectivo se presenta en el anexo [Codificación](#).

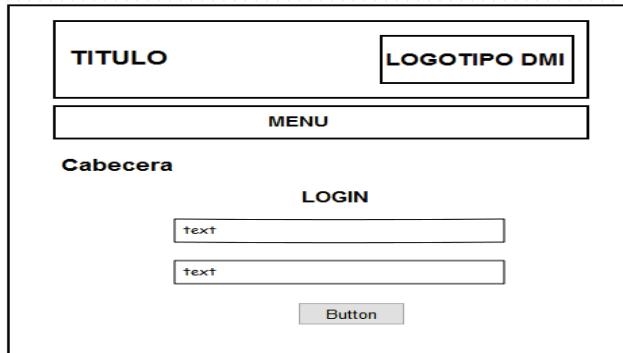
### 2.3.2.2 CAPA CONTROL USUARIO

En esta capa implementamos la clase *ControladorUsuario.cs* la cual permite el acceso a los métodos definidos en la clase *ServicioUsuario.cs*, esto se hace con la finalidad de facilitar la interrelación entre la capa de vista que contiene las interfaces Web las cuales interactúan con el usuario y la capa de modelo.

### 2.3.2.3 CAPA VISTA USUARIO

En esta capa inicia con la satisfacción de los requerimientos del cliente los cuales están definidos en el Product Backlog. Partiendo de esto se diseña e implementa la interfaz que interactuará con el cliente, la cual tendrá varias funciones en las cuales intervienen la capa de modelo y control. Para esto se realizó un prototipo de la primera interfaz web. Figura 2.27.

**Figura 2.27 Prototipo Página USUARIO**



Elaborado por los Autores

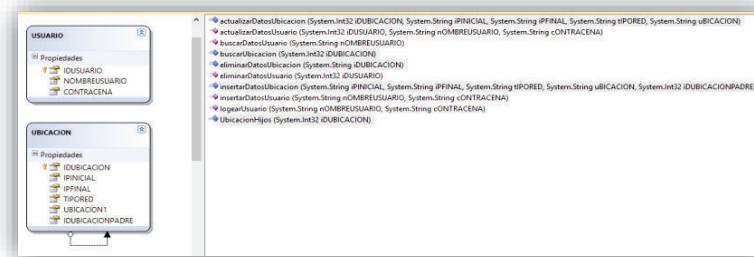
A partir de este prototipo se implementa las clases *LoginLizard.aspx* y *RegistroUsuario.aspx*, las cuales permite el acceso y registró de un usuario que interactúa directamente con el cliente.

De la misma manera que se implementa el *LoginLizard.aspx* y se implementa la clase *RegistroUsuario.aspx* ver anexo **Codificación**.

#### 2.3.2.4 Capa Modelo Ubicación

En esta capa se inició realizando el mapeo de la tabla y procedimientos almacenados correspondientes a UBICACIÓN para lo cual se usó la clase LinQ “DataLizardBDD.dbml”. Figura 2.28.

**Figura 2.28 Mapeo Tabla y Procedimientos Almacenados UBICACION**



Elaborado por los Autores

A partir de esto se implementa la clase *ServicioUbicación.cs* en la cual personalizamos el tipo de retorno de nuestros procedimientos almacenados de UBICACION. El siguiente código muestra de una manera detallada.

### 2.3.2.5 CAPA CONTROL UBICACIÓN

En esta capa implementamos la clase *ControladorUbicación.cs* la cual permite el acceso a los métodos definidos en la clase *ServicioUbicación.cs*, esto se hace con la finalidad de facilitar la interrelación entre la capa de vista que contiene las interfaces Web las cuales interactúan con el usuario y la capa de modelo.

### 2.3.2.6 CAPA VISTA UBICACIÓN

En esta capa inicia con la satisfacción de los requerimientos del cliente los cuales están definidos en el Product Backlog en el Primer Sprint. Partiendo de esto se diseña e implementa la interfaz que interactuara con el cliente, la cual tendrá varias funciones en las cuales intervienen la capa de modelo y control. Para esto se realizó un prototipo de nuestra primera interfaz web. Figura 2.29.

Figura 2.29 Prototipo Página UBICACION

	UBICACION	IP INICIAL	IP FINAL
<input type="checkbox"/>	Texto	Texto	Texto
<input type="checkbox"/>	Texto	Texto	Texto

Elaborado por los Autores

A partir de este prototipo se implementa la clase *Ubicacion.aspx* la cuales permite la administración de las ubicaciones y sub-ubicaciones generadas por el cliente.

### 2.3.2.7 Primer Sprint Review

En esta iteración la tarea de creación del formulario de Ubicación llevó más tiempo en su implementación, así que el equipo de trabajo tuvo que aumentar el tiempo de consecución de esta tarea y disminuyendo el tiempo de afinamiento de este formulario logrando un cambio en horarios de tareas planificadas como se presenta en la Figura 2.30.

**Figura 2.30 Cambio de tareas planificadas primer Sprint**



Elaborado por los Autores

Donde se puede observar una ligera variación en tareas del 4 de agosto y una disminución de tareas para el 6 de agosto. También un extensión de las tareas que empiezan el 29 de julio.

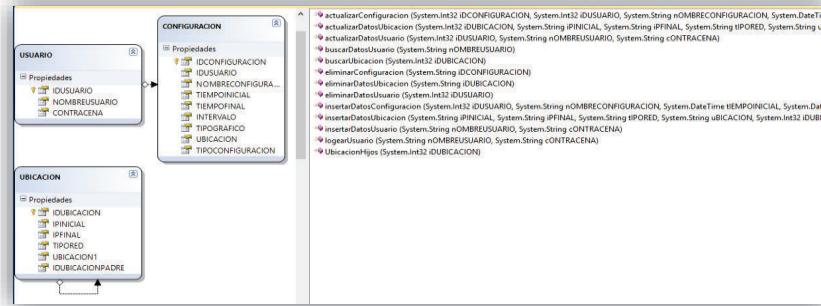
Para ver la entrega de este Sprint con sus tareas completas en el plazo establecido revisar la sección [2.4.2](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

### 2.3.3 EJECUCIÓN DEL SEGUNDO SPRINT

#### 2.3.3.1 Capa Modelo Configuración

En esta capa se inició realizando el mapeo de la tabla y procedimientos almacenados correspondientes a CONFIGURACION para lo cual se usó la clase LINQ anteriormente definida. Figura 2.31.

Figura 2.31 Mapeo de Tabla y Procedimientos Almacenados de Configuración



Elaborado por los Autores

A partir de esto se implementa la clase *ServicioConfiguración.cs* en la cual se personaliza el tipo de retorno de nuestros procedimientos almacenados de CONFIGURACION. El siguiente código muestra de una manera detallada la clase *ServicioConfiguración.cs* incluida en el anexo [Codificación](#).

### 2.3.3.2 CAPA CONTROL CONFIGURACIÓN

En esta capa se implementa la clase *ControladorConfiguracion.cs* la cual permite el acceso a los métodos definidos en la clase *ServicioConfiguracion.cs*, esto se hace con la finalidad de facilitar la interrelación entre la capa de vista que contiene las interfaces Web las cuales interactúan con el usuario y la capa de modelo.

### 2.3.3.3 Capa Vista Configuración

En esta capa inicia con la satisfacción de los requerimientos del cliente los cuales están definidos en el Product Backlog del segundo sprint. Partiendo de esto se diseña e implementa la interfaz que interactuara con el cliente, la cual tendrá varias funciones en las cuales intervienen la capa de modelo y control. Para esto se realizó un prototipo de nuestra primera interfaz web. Figura 2.32.

**Figura 2.32 Prototipo Página Configuración**

	UBICACION	IP INICIAL	IP FINAL
<input type="checkbox"/>	Texto	Texto	Texto
<input checked="" type="checkbox"/>	Text	Text	Text

text  
text  
text

Button

Elaborado por los Autores

A partir de este prototipo se implementa la clase Configuración.aspx la cual permite la administración de las configuraciones generadas por el cliente. Para ver todo el código ver anexo [Codificación](#).

#### 2.3.3.4 Segundo Sprint Review

En esta iteración se realizaron la implementación del formulario de Configuración el cual en la práctica real llevó más tiempo realizarlo produciendo un ligero cambio en el gráfico de tareas realizadas dando así un ligero desnivel en una sección del gráfico para realizar la tarea extra y otro desnivel para equilibrar la anterior tarea de acuerdo al tiempo de estimación de planificación. Ver Figura 2.33.

**Figura 2.33 Cambio de tareas planificadas segundo Sprint**



Elaborado por los Autores

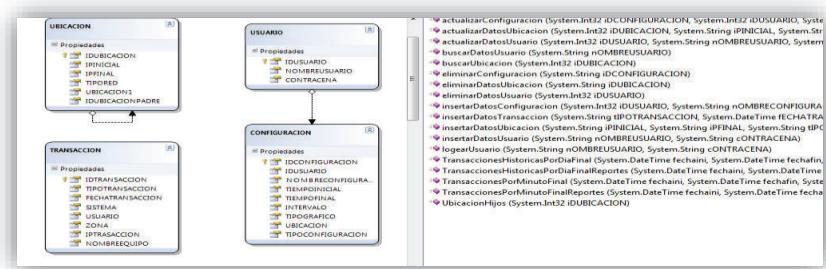
Para ver la entrega de este Sprint con sus tareas completas en el plazo establecido revisar la sección [2.4.3](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

### 2.3.4 EJECUCIÓN DEL TERCER SPRINT

#### 2.3.4.1 Capa Modelo Transacción

En esta capa se inició realizando el mapeo de la tabla y SPs correspondientes a TRANSACCION para lo cual se usó la clase LinQ anteriormente definida, ver Figura 2.34.

Figura 2.34 Mapeo de Tabla y SPs correspondientes a CONFIGURACION



Elaborado por los Autores

A partir de esto se implementa la clase *ServicioTransaccion.cs* la cual inserta una transacción de acuerdo a su IP y en base a esto la relaciona con una UBICACION y los procedimientos almacenados de TRANSACCION.

Básicamente esta clase pasa desapercibida en la capa de vista ya que es usada como una librería para el rol de cliente definido en la sección “roles y responsabilidades”, La clase del servicio *ServicioTransaccion.cs* se encuentra en el anexo Codificación.

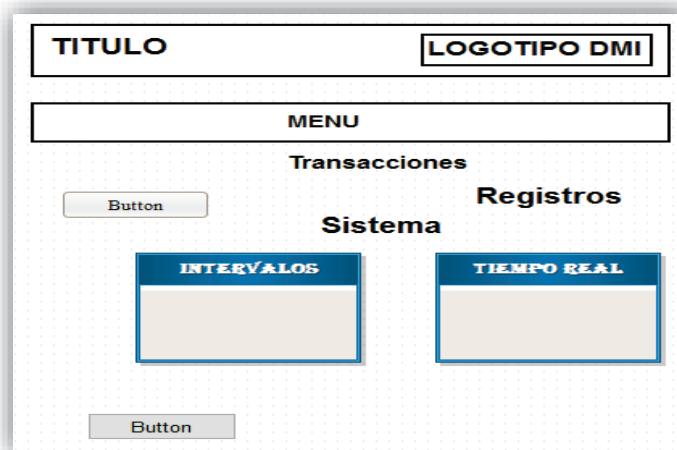
#### 2.3.4.2 Capa Control Transacción

En esta capa se implementa la clase *ControladorTransaccion.cs* la cual permite el acceso a los métodos definidos en la clase *ServicioTransaccion.cs* con el fin de la captura de transacciones por parte de los clientes.

### 2.3.4.3 Capa Vista Transacción

En esta capa inicia con la satisfacción de los requerimientos del cliente los cuales están definidos en el Product Backlog en el tercer sprint. Partiendo de esto se diseña e implementa la interfaz que interactuara con el cliente, la cual tendrá varias funciones en las cuales intervienen la capa de modelo y control. No obstante en esta capa se usó varios SLQDataSource los cuales permiten el acceso a los procedimientos almacenados sin necesidad de pasar por la clase LINQ. Figura 2.35.

Figura 2.35 Prototipo Página Transacciones



Elaborado por los Autores

A partir de este prototipo se implementaron dos clases principales *ReportesTransacción.aspx* y *ReportesHistóricos.aspx*.

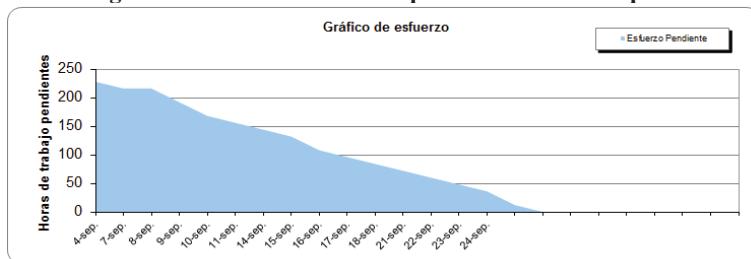
Para la generación de una clase principal de tipo grafica se realiza el uso de un webUserControl. En el cual se define de forma genérica el formato de la gráfica. La clase principal tiene por función albergar una lista con las n-graficas correspondientes a n-sistemas. Todo esto se encuentra dentro de la clase *ReportesTransacción.aspx* y *WebUserReportes.ascx*.

Para ver el código del resto de formularios que corresponden a *ReportesHistóricos.aspx* y *WebUserHistóricos.ascx* en el anexo [Codificación](#).

#### 2.3.4.4 Tercer Sprint Review

Para esta iteración las tareas de integración de formularios tanto para transacciones como para históricos han llevado más tiempo para su realización que el estimado en la planificación del segundo Sprint. Ver Figura 2.36.

**Figura 2.36 Cambio de tareas planificadas tercer Sprint**



Elaborado por los Autores

Para ver la entrega de este Sprint con sus tareas completas en el plazo establecido revisar la sección [2.4.4](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

#### 2.3.5 EJECUCIÓN DEL CUARTO SPRINT

##### 2.3.5.1 Implementación Reportes

Este punto inicia con la satisfacción de los requerimientos del cliente, los cuales están definidos en el Product Backlog en el cuarto sprint. Partiendo del esto se diseña e implementa la interfaz que interactuara con el cliente. Sin embargo para la generación de reportes no es necesario el uso de las demás capas del sistema, ya que su funcionamiento proviene del uso de varios *SLQDataSource* los cuales permiten el acceso a los procedimientos almacenados sin necesidad de pasar por el resto de capas. En la Figura 2.37 se puede apreciar un prototipo de un reporte.

Figura 2.37 Prototipo Página Reportes



Elaborado por los Autores

A partir de este prototipo se implementan varias clases .aspx las cuales tienen por función capturar los datos para la generación de reportes, entonces se mencionan las clases .aspx utilizadas para este fin: *ReportesPorSistema.aspx*, *ReportesPorSistemaReal.aspx*, *ReportesTransaccionGeneral.aspx*, *ReportesHistoricosPorSistema.aspx* y *TransaccionesHistoricasGeneral.aspx*. Las clases .aspx realizan una llamada a clases .cs las cuales contienen un prototipo de los reportes, con ayuda de los datos solicitados se realiza la generación de reportes del sistema estas son: *TReportesHistoricosPorSistema.cs*, *TReporteTransaccionesHistoricasGeneral.cs*, *TReportesPorSistema.cs* y *TReportesPorSistemaGeneral.cs*.

Para la generación del primer reporte, el cual filtra las transacciones por fecha inicial, fecha final, intervalo, sistema y ubicación se hizo el uso de dos clases. La primera clase principal *ReportesPorSistema.aspx* la cual es encargada de enviar parámetros para la generación del reporte, y la segunda la cual contiene el formato genérico de un reporte *TReportesPorSistema.cs*.

La página de reportes permite al usuario generar n-gráficas por n-sistemas clientes. Para encontrar todos las clases de tipo reporte ver anexo [Codificación](#).

### 2.3.5.2 Cuarto Sprint Review

En esta iteración las tareas finales se realizaron con todo éxito sin demora ni variación de las tareas planificadas además de la aceptación del cliente para el producto final entregado ha sido positiva y el resultado fue aceptado.

Para ver la entrega de este Sprint con sus tareas completas en el plazo establecido revisar la sección [2.4.5](#). Donde se expondrán las tareas realizadas correspondientes a esta iteración.

## 2.3.6 PRUEBAS UNITARIAS

Se procede a realizar la ejecución del proyecto para esto se define casos de prueba que sirven para verificar si el sistema cumple con los objetivos deseados.

Con la metodología SCRUM la forma para realizar las pruebas, es verificando el cumplimiento de las historias de usuario.

### 2.3.6.1 Formato Casos De Prueba

Para realizar la simulación vamos a utilizar el siguiente formato:

**Número:** secuencial de la historia del caso de simulación (prueba).

**Historia de Usuario:** nombre y número de la historia de usuario de lo que se quiere simular (probar).

**Nombre caso de simulación:** Nombre de la tarea o requisito funcional a ser probado.

**Descripción:** breve descripción de lo que se desea simular.

**Prerrequisitos:** condiciones previas que deben existir antes de realizar la prueba.

**Entrada/Pasos de Ejecución:** pasos y entradas necesarias para realizar la prueba.

**Resultado esperado:** resultado que se espera luego de realizar la prueba.

**Resultado Simulación:** resultado real obtenido luego de haberse realizado la prueba.

### CASO DE PRUEBA #1 “Registro de Cliente en el Sistema”

<b>Caso de Prueba</b>	
<b>Código:</b> 1	<b>Historia de Usuario:</b> 1 – Acceso de Usuarios
<b>Nombre Caso de Simulación:</b>	
Registrar un nuevo usuario	
<b>Descripción:</b>	
Se procede a realizar la prueba del proceso de registro del usuario al sistema para su posterior ingreso.	
<b>Prerrequisitos:</b>	
Ninguno	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>1. El usuario ingresa al sistema y selecciona la opción registrarse</li> <li>2. Ingresar su nombre de usuario, contraseña y confirmación de contraseña</li> <li>3. La aplicación redirecciona a la página de inicio de sesión</li> </ol>	
<b>Resultado Esperado:</b>	
<ul style="list-style-type: none"> <li>• El usuario puede acceder a la aplicación</li> <li>• La información de acceso de usuario ha sido registrados en la base de datos</li> </ul>	
<b>Resultado de la Prueba:</b> Muy Bueno	
El usuario ha sido registrado en la base de datos del sistema.	
El usuario ya puede hacer uso de su nombre y contraseña para ingresar al sistema.	
El sistema no se vincula al directorio activo de usuarios de la organización.	
El sistema carece de un módulo para la gestión de usuarios.	

#### Demostración:

1. El usuario ingresa al sistema y selecciona la opción registrarse.

**Figura 2.38 Pagina de registro del Sistema**

**Creado por los Autores**

2. Ingresa su nombre de usuario, contraseña y confirmación de contraseña.

**Figura 2.39 Crear nuevo Usuario del Sistema**

**Crear una nueva cuenta**

Use el formulario siguiente para crear una cuenta nueva.

Información de cuenta

Nombre de usuario:

Contraseña:

Confirmar contraseña:

**REGISTRAR USUARIO**

**REGRESAR A LA LISTA**

Creado por los Autores

3. La aplicación redirecciona a la página de inicio de sesión

**Figura 2.40 Página de inicio del Sistema**

**Lizard**

**QUITO**  
DIRECCIÓN METROPOLITANA  
DE INFORMACIÓN

Ingresar

Página principal Acerca de

**Iniciar sesión**

Especifique su nombre de usuario y contraseña. [Registrarse](#) si no tiene una cuenta.

Información de cuenta

Nombre de usuario:

Contraseña:

Mantenerme conectado

**INICIAR SESIÓN**

Creado por los Autores

## CASO DE PRUEBA #2 “Ingresar como usuario existente”

<b>Caso de Prueba</b>	
<b>Código:</b> 2	<b>Historia de Usuario:</b> 1 – Acceso de Usuarios
<b>Nombre Caso de Simulación:</b> Ingresar como usuario existente	

<b>Descripción:</b>
Se procede a realizar la prueba del acceso al sistema por parte de un usuario.
<b>Prerrequisitos:</b>
El usuario debe estar registrado en el sistema.
<b>Entrada/Pasos de ejecución:</b>
<ol style="list-style-type: none"> <li>1. El usuario accede al sistema.</li> <li>2. Ingresá su nombre de usuario, contraseña.</li> <li>3. El sistema redirecciona a la página principal del proyecto.</li> </ol>
<b>Resultado Esperado:</b>
<ul style="list-style-type: none"> <li>• El usuario accede al sistema como usuario registrado.</li> </ul>
<b>Resultado de la Prueba:</b> Excelente
El usuario tiene acceso a los módulos del sistema.

#### Demostración:

1. El usuario accede al sistema.

**Figura 2.41 Ingreso al sistema de un usuario registrado**

Registrarse si no tiene una cuenta.' The main form area is titled 'Información de cuenta' and contains fields for 'Nombre de usuario:' and 'Contraseña:', both with placeholder text. There is also a checked checkbox for 'Mantenerme conectado'. At the bottom of the form is a blue 'INICIAR SESIÓN' button."/>

**Creado por los Autores**

2. Ingresá su nombre de usuario, contraseña.

**Figura 2.42 Ingreso a la aplicación por parte de un usuario registrado**

**Creado por los Autores**

3. El sistema redirecciona a la página principal del proyecto.

**Figura 2.43 Inicio de aplicación**

**Creado por los Autores**

### **CASO DE PRUEBA #3 “Registrar una nueva Ubicación”**

<b>Caso de Prueba</b>	
<b>Código:</b> 3	<b>Historia de Usuario:</b> 2 – Administrar Ubicaciones
<b>Nombre Caso de Simulación:</b>	

Registrar una nueva ubicación
<b>Descripción:</b>
Se procede a realizar la prueba para registrar una ubicación al sistema para su posterior uso.
<b>Prerrequisitos:</b>
<ul style="list-style-type: none"> <li>• Haber accedido al sistema como un usuario registrado.</li> </ul>
<b>Entrada/Pasos de ejecución:</b>
<ol style="list-style-type: none"> <li>1. El usuario ingresa al sistema y accede al módulo ubicación.</li> <li>2. Ingresa el nombre y un rango de IP.</li> <li>3. Finalmente damos click en el botón “Registrar Ubicación”.</li> </ol>
<b>Resultado Esperado:</b>
<ul style="list-style-type: none"> <li>• La información de la ubicación ha sido registrada en la base de datos.</li> </ul>
<b>Resultado de la Prueba:</b> Muy buena
La ubicación ha sido registrada en la base de datos del sistema.
Los usuarios pueden hacer uso de la nueva ubicación
Las ubicaciones debes ser registradas por el usuario de forma manual

1. El usuario ingresa al sistema y accede al módulo ubicación

Figura 2.44 Ingreso al módulo de ubicación

Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.168.1.1	192.168.1.255	
3	DMI	127.0.0.1	127.0.0.1	

Ubicación: \_\_\_\_\_  
 Ip Inicial: \_\_\_\_\_  
 Ip Final: \_\_\_\_\_

**REGISTRAR UBICACIÓN**

Creado por los Autores

2. Ingresa el nombre y un rango de IP

Figura 2.45 Ingreso de datos de ubicación

The screenshot shows a table with columns: Id, Ubicación, Ip Inicial, and Ip Final. There are three rows: Quito (Id 2), DMI (Id 3), and a new row being added (Id 4). The new row has the following values: Ubicación: DMI\_Redes, Ip Inicial: 172.20.47.0, and Ip Final: 172.20.47.255. Below the table is a form with fields: Ubicación: DMI\_Redes, Ip Inicial: 172.20.47.0, and Ip Final: 172.20.47.255. A red arrow points to a blue 'REGISTRAR UBICACIÓN' button.

Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.168.1.1	192.168.1.255	
3	DMI	127.0.0.1	127.0.0.1	
4	DMI_Redes	172.20.47.0	172.20.47.255	

REGISTRAR UBICACIÓN

Creado por los Autores

3. Finalmente damos click en el botón “Registrar Ubicación”.

Figura 2.46 Registro de ubicación

The screenshot shows the same table as Figure 2.45, but now with four rows. The fourth row (Id 4) has been successfully added with the values: Ubicación: DMI\_Redes, Ip Inicial: 172.20.47.0, and Ip Final: 172.20.47.255. A red box highlights this newly added row.

Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.168.1.1	192.168.1.255	
3	DMI	127.0.0.1	127.0.0.1	
4	DMI_Redes	172.20.47.0	172.20.47.255	

Creado por los Autores

#### CASO DE PRUEBA #4 “Registrar sub ubicaciones”

Caso de Prueba	
Código: 4	Historia de Usuario: 2 – Administrar Ubicaciones
<b>Nombre Caso de Simulación:</b>	
Registrar sub ubicaciones	
<b>Descripción:</b>	
Se procede a realizar la prueba para registrar una sub-ubicación al sistema para su posterior uso.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>Haber accedido al sistema como un usuario registrado.</li> <li>Tener por lo menos una ubicación registrada.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>El usuario accede al módulo ubicación.</li> <li>Se selecciona una ubicación a la cual se le añadirá una sub-ubicación.</li> <li>Ingresar el nombre y un rango de IP.</li> <li>Finalmente hacemos click en el botón “Registrar Sub-Ubicación”.</li> </ol>	
<b>Resultado Esperado:</b>	

1. Los usuarios pueden hacer uso de la nueva sub-ubicación. 2. La información de la sub-ubicación ha sido registrada en la base de datos.
<b>Resultado de la Prueba:</b> Muy Buena  La sub-ubicación ha sido registrada en la base de datos del sistema. Los usuarios pueden hacer uso de la nueva sub-ubicación.

Demostración:

1. El usuario accede al módulo ubicación.

**Figura 2.47 Ingreso al módulo de ubicación**



2. Se selecciona una ubicación a la cual se le añadirá una sub-ubicación.

**Figura 2.48 Agregar una sub-ubicación**

The screenshot shows the 'Ubicación' section of the Lizard application. It displays a table with columns: Id, Ubicación, Ip Inicial, Ip Final, and actions (represented by icons for add, edit, and delete). There are three rows in the table. At the bottom, there is a page navigation bar with arrows, a page size selector set to 10, and the text 'Creado por los Autores'.

Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.168.1.1	192.168.1.255	
3	DMI	127.0.0.1	127.0.0.1	
5	DMI_Netwes			

3. Ingresa el nombre y un rango de IP.

**Figura 2.49 Ingreso de datos de sub-ubicación**

ID	Ubicación	IP Inicial	IP Final	
2	Quito	192.168.1.1	192.168.1.255	
3	DMI	127.0.0.1	127.0.0.1	
5	DMI_Redes			
<input style="width: 100%;" type="button" value="REGISTRAR SUB-UBICACIÓN"/>				

Ubicación: DMI Redes Planta baja  
 IP Inicial: 172.20.47.10  
 IP Final: 172.20.47.50

Creado por los Autores

- Finalmente hacemos click en el botón “Registrar Sub-Ubicación”.

**Figura 2.50 Registro de sub-ubicación**

ID	Ubicación	IP Inicial	IP Final	
2	Quito	192.168.1.1	192.168.1.255	
3	DMI	127.0.0.1	127.0.0.1	
5	DMI_Redes			
6	DMI Redes Planta baja	172.20.47.10	172.20.47.50	
<input style="width: 100%;" type="button" value="REGISTRAR SUB-UBICACIÓN"/>				

Ubicación: DMI Redes Planta baja  
 IP Inicial: 172.20.47.10  
 IP Final: 172.20.47.50

Creado por los Autores

## CASO DE PRUEBA #5 “Edición de Ubicaciones”

Caso de Prueba	
Código: 5	Historia de Usuario: 2 – Administrar Ubicaciones
<b>Nombre Caso de Simulación:</b>	
Edición de ubicaciones	
<b>Descripción:</b>	
Se procede a realizar la prueba para editar una ubicación o sub-ubicación existente en el sistema.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>Tener por lo menos una ubicación registrada.</li> </ul>	

**Entrada/Pasos de ejecución:**

1. El usuario accede al módulo ubicación.
2. Se selecciona la ubicación a ser editada.
3. Se pueden editar los campos nombre y rango de IP.
4. Finalmente damos click en el botón “Actualizar”.

**Resultado Esperado:**

- La información de la ubicación ha sido actualizada en la base de datos.

**Resultado de la Prueba:** Excelente

La ubicación ha sido actualizada en la base de datos del sistema.

Los usuarios pueden hacer uso de la ubicación.

**Demostración:**

1. El usuario accede al módulo ubicación.

**Figura 2.51 Ingreso al módulo de ubicación**



2. Se selecciona la ubicación a ser editada.

**Figura 2.52 Edición de ubicación**

The screenshot shows a table listing locations. The columns are labeled 'Id', 'Ubicación', 'Ip Inicial', 'Ip Final', and four icons for 'Nuevo', 'Editar', 'Borrar', and 'Historico'. The data in the table is as follows:

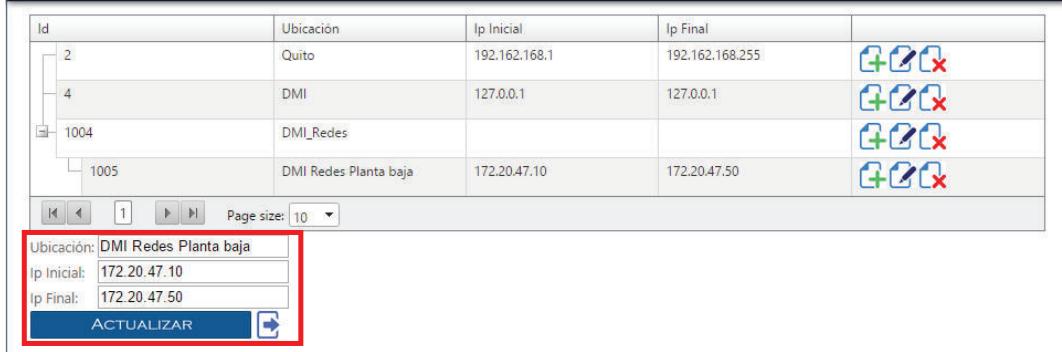
Id	Ubicación	Ip Inicial	Ip Final			
2	Quito	192.162.168.1	192.162.168.255			
4	DMI	127.0.0.1	127.0.0.1			
1004	DMI_Network					
1005	DMI Redes Planta baja	172.20.47.10	172.20.47.50			

Below the table, there is a page navigation bar with arrows and a 'Page size: 10' dropdown.

Creado por los Autores

3. Se pueden editar los campos nombre y rango de IP.

**Figura 2.53 Edición de datos de ubicación**



Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.162.168.1	192.162.168.255	
4	DMI	127.0.0.1	127.0.0.1	
1004	DMI_Redes			
1005	DMI Redes Planta baja	172.20.47.10	172.20.47.50	

Page size: 10

Ubicación: DMI Redes Planta baja  
 Ip Inicial: 172.20.47.10  
 Ip Final: 172.20.47.50

ACTUALIZAR

Creado por los Autores

**Figura 2.54 Actualización de datos de ubicación**



Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.162.168.1	192.162.168.255	
4	DMI	127.0.0.1	127.0.0.1	
1004	DMI_Redes			
1005	DMI Redes Planta baja	172.20.47.10	172.20.47.50	

Page size: 10

Ubicación: DMI Redes Planta baja  
 Ip Inicial: 172.20.47.1  
 Ip Final: 172.20.47.255

ACTUALIZAR

Creado por los Autores

4. Finalmente damos click en el botón “Actualizar”.

**Figura 2.55 Ubicación Actualizada**



Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.162.168.1	192.162.168.255	
4	DMI	127.0.0.1	127.0.0.1	
1004	DMI_Redes			
1005	DMI Redes Planta baja	172.20.47.1	172.20.47.255	

Page size: 10

Creado por los Autores

## CASO DE PRUEBA #6 “Eliminar Ubicaciones”

<b>Caso de Prueba</b>	
<b>Código:</b> 6	<b>Historia de Usuario:</b> 2 – Administrar Ubicaciones
<b>Nombre Caso de Simulación:</b>	
Eliminar Ubicaciones	
<b>Descripción:</b>	
Se procede a realizar la prueba para eliminar una ubicación o sub-ubicación existente en el sistema.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>• Tener por lo menos una ubicación registrada.</li> <li>• La Ubicación a ser eliminada no tenga descendencia.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>1. El usuario accede al módulo ubicación.</li> <li>2. Se selecciona una ubicación la cual se eliminara.</li> <li>3. Finalmente damos click en el botón de eliminación de ubicaciones.</li> </ol>	
<b>Resultado Esperado:</b>	
<ul style="list-style-type: none"> <li>• La ubicación será eliminada del sistema</li> <li>• La información de la ubicación ha sido eliminada de la base de datos.</li> </ul>	
<b>Resultado de la Prueba:</b> Muy Buena	
La ubicación ha sido eliminada de la base de datos del sistema.	
Se requirió incluir mensajes de alerta para la eliminación de ubicaciones.	

#### Demostración:

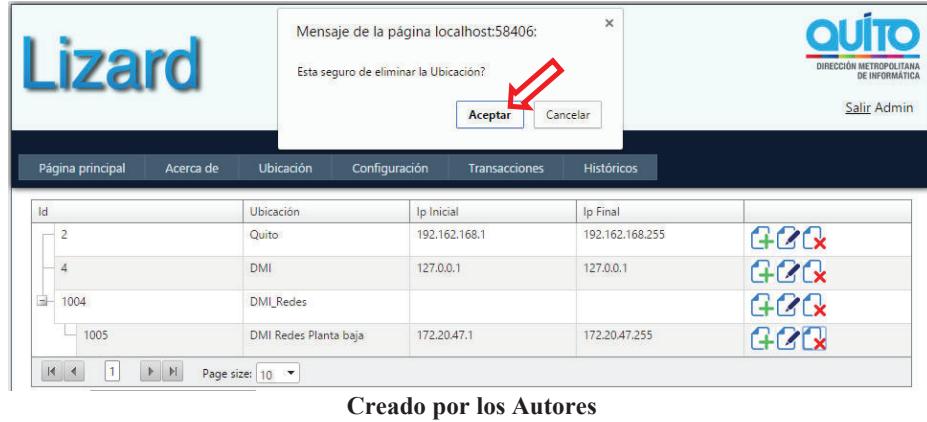
1. El usuario accede al módulo ubicación.

**Figura 2.56 Ingreso al módulo ubicación**



2. Se selecciona una ubicación la cual se eliminara.

Figura 2.57 Eliminación de ubicación



Creado por los Autores

- Finalmente damos click en el botón de eliminar de ubicaciones y confirmamos la acción.

Figura 2.58 Ubicación eliminada

Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.162.168.1	192.162.168.255	
4	DMI	127.0.0.1	127.0.0.1	
1004	DMI_Redes			

Creado por los Autores

#### CASO DE PRUEBA #7 “Registrar una nueva configuración”

Caso de Prueba de Aceptación	
<b>Código:</b> 7	<b>Historia de Usuario:</b> 3 – Administrar Configuraciones
<b>Nombre Caso de Simulación:</b>	
Registrar una nueva configuración	
<b>Descripción:</b>	
Se procede a realizar la prueba del proceso de registro de una configuración para el usuario.	
<b>Prerrequisitos:</b>	
El usuario de acceder al módulo de configuración.	

<b>Entrada/Pasos de ejecución:</b>
1. El usuario accede al módulo de configuración.
2. Ingresá una nueva configuración seleccionando el tipo de configuración.
3. Finalmente se da click en el botón “Registrar Configuración”.
<b>Resultado Esperado:</b>
La información de configuración es actualizada en la base de datos y la nueva información puede ser vista.
<b>Resultado de la Prueba:</b> Muy Buena Se pudo realizar la creación de la configuración. Al desplegar la nueva información, ésta ya reflejó los cambios realizados. Se agregó para la correcta prueba validaciones necesarias especialmente con el ingreso de fechas apropiadas para el sistema.

### Demostración:

1. El usuario accede al módulo de configuración.

Figura 2.59 Ingreso al módulo de configuración



2. Ingresá una nueva configuración seleccionando el tipo de configuración.

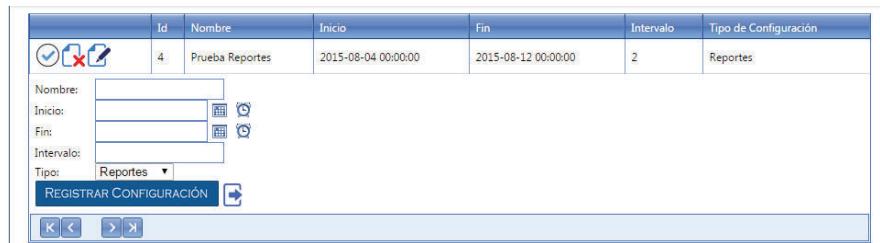
Figura 2.60 Ingreso de datos de nueva configuración

The screenshot shows the configuration form within the Lizard application. The form has columns for Id, Nombre, Inicio, Fin, Intervalo, and Tipo de Configuración. The 'Nombre' field contains 'Prueba Reportes'. The 'Inicio' field shows '2015-08-04 00:00:00'. The 'Fin' field shows '2015-08-12 00:00:00'. The 'Intervalo' field is set to '2'. The 'Tipo' dropdown is set to 'Reportes'. A red arrow points to the 'REGISTRAR CONFIGURACIÓN' button at the bottom of the form. Below the form is a footer with navigation icons.

Creado por los Autores

3. Finalmente se da click en el botón “Registrar Configuración”.

**Figura 2.61 Registro de configuración**



Creado por los Autores

### CASO DE PRUEBA #8 “Edición de Configuraciones”

<b>Caso de Prueba</b>	
<b>Código:</b> 8	<b>Historia de Usuario:</b> 2 – Administrar Configuraciones
<b>Nombre Caso de Simulación:</b> Edición de configuraciones	
<b>Descripción:</b> Se procede a realizar la prueba para editar una configuración existente en el sistema.	
<b>Prerrequisitos:</b> <ul style="list-style-type: none"> <li>Tener por lo menos una configuración registrada.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b> <ol style="list-style-type: none"> <li>1. El usuario accede al módulo configuración.</li> <li>2. Se selecciona la configuración a ser editada.</li> <li>3. Se pueden editar los campos nombre, las fechas, el intervalo de mediciones y el tipo de reporte.</li> <li>4. Finalmente damos click en el botón “Actualizar”.</li> </ol>	
<b>Resultado Esperado:</b> <ul style="list-style-type: none"> <li>La información de la configuración ha sido actualizada en la base de datos.</li> </ul>	
<b>Resultado de la Prueba:</b> Excelente La configuración ha sido actualizada en la base de datos del sistema. El usuario puede hacer uso de la configuración.	

Demostración:

- El usuario accede al módulo configuración.

Figura 2.62 Ingreso al módulo de configuración



- Se selecciona la configuración a ser editada.

Figura 2.63 Edición de configuración

This screenshot shows the configuration editing screen. It features a table with a single row containing configuration details. Below the table is a form with fields for 'Nombre', 'Inicio', 'Fin', 'Intervalo', and 'Tipo'. The 'Tipo' field is set to 'Reportes'. At the bottom of the form is a blue button labeled 'REGISTRAR CONFIGURACIÓN' with a red arrow pointing to it. Navigation buttons for the table are also visible at the bottom.

Creado por los Autores

- Se pueden editar los campos nombre, las fechas, el intervalo de mediciones y el tipo de reporte.

Figura 2.64 Ingreso de Datos de configuración

This screenshot shows the configuration data entry screen. It displays a table with a single row of configuration data. The 'Id' field is highlighted with a red box. Below the table is a form with fields for 'Nombre', 'Inicio', 'Fin', 'Intervalo', and 'Tipo'. The 'Tipo' field is set to 'Reportes'. At the bottom of the form is a blue button labeled 'ACTUALIZAR' with a red arrow pointing to it. Navigation buttons for the table are also visible at the bottom.

Creado por los Autores

**Figura 2.65 Edición de datos de configuración**

Página principal	Acerca de	Ubicación	Configuración	Transacciones	Históricos													
			<table border="1"> <tr> <th></th> <th>Id</th> <th>Nombre</th> <th>Inicio</th> <th>Fin</th> <th>Intervalo</th> <th>Tipo de Configuración</th> </tr> <tr> <td></td> <td>4</td> <td>Prueba Reportes</td> <td>2015-08-05 00:00:00</td> <td>2015-08-12 00:00:00</td> <td>2</td> <td>Reportes</td> </tr> </table>		Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración		4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes	
	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración												
	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes												

Creado por los Autores

- Finalmente damos click en el botón “Actualizar”.

**Figura 2.66 Registro de edición de configuración**

Página principal	Acerca de	Ubicación	Configuración	Transacciones	Históricos								
			<table border="1"> <tr> <td><input checked="" type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td>4</td> <td>Prueba Reportes</td> <td>2015-08-05 00:00:00</td> <td>2015-08-12 00:00:00</td> <td>2</td> <td>Reportes</td> </tr> </table>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes					

Creado por los Autores

#### CASO DE PRUEBA #9 “Eliminación de Configuraciones”

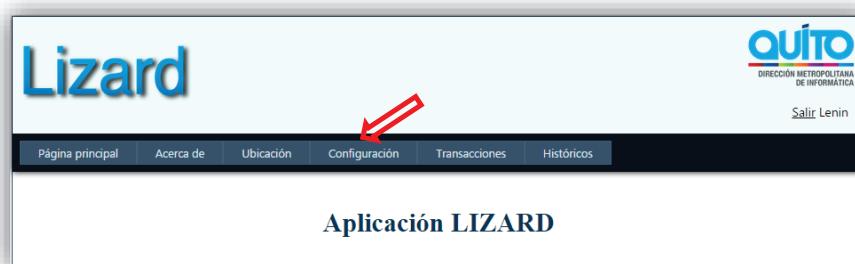
<b>Caso de Prueba</b>	
<b>Código:</b> 9	<b>Historia de Usuario:</b> 2 – Administrar Configuraciones
<b>Nombre Caso de Simulación:</b>	
Eliminación de Configuraciones	
<b>Descripción:</b>	
Se procede a realizar la prueba para eliminar una configuración existente en el sistema.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>Tener por lo menos una configuración registrada.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>El usuario accede al módulo configuración.</li> <li>Se selecciona una configuración la cual se eliminará.</li> </ol>	

3. Finalmente damos click en el botón de eliminación de configuraciones.
<b>Resultado Esperado:</b>
<ul style="list-style-type: none"> <li>• La configuración será eliminada del sistema</li> <li>• La información de la configuración ha sido eliminada de la base de datos.</li> </ul>
<b>Resultado de la Prueba:</b> Muy Buena La configuración ha sido eliminada de la base de datos del sistema. Se agregó mensajes de confirmación o de rechazo de la eliminación de configuraciones.

Demostración:

1. El usuario accede al módulo configuración.

Figura 2.67 Ingreso al módulo configuración



Creado por los Autores

2. Se selecciona una configuración la cual se eliminará.

Figura 2.68 Selección de ubicación a ser eliminada

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
<input checked="" type="checkbox"/>	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
<input checked="" type="checkbox"/>	5	Reporte Historico	2014-07-05 08:00:00	2015-08-22 00:00:00	1	Históricos
<input checked="" type="checkbox"/>	6	Prueba Eliminar	2015-07-09 00:00:00	2015-07-17 00:00:00	2	Reportes

Creado por los Autores

3. Finalmente damos click en el botón de eliminación de configuraciones.

Figura 2.69 Confirmación de eliminación de configuración



Creado por los Autores

## CASO DE PRUEBA #10 “Generar gráficas de clientes”

<b>Caso de Prueba</b>	
<b>Código:</b> 10	<b>Historia de Usuario:</b> 4 – Generar Gráficas Estadísticas
<b>Nombre Caso de Simulación:</b>	
Generar gráficas de clientes	
<b>Descripción:</b>	
Se procede a realizar la prueba para la generación de gráficas estadísticas de los clientes o aplicativos monitoreados por el sistema el sistema.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>• Tener por lo menos una ubicación registrada.</li> <li>• Tener Transacciones pertenecientes a una ubicación.</li> <li>• Tener una configuración registrada la cual tenga parámetros de filtración válidos.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>1. El usuario accede al módulo configuración.</li> <li>2. El usuario selecciona la configuración correspondiente, a través de la cual desea la generación de gráficas dependiendo si son gráficas históricas o de transacciones.</li> <li>3. El usuario selecciona la ubicación.</li> <li>4. El sistema redirecciona a la página que contiene el gráfico.</li> </ol>	
<b>Resultado Esperado:</b>	
<ul style="list-style-type: none"> <li>• La obtención de gráficos con los respectivos parámetros de búsqueda ingresados.</li> </ul>	
<b>Resultado de la Prueba:</b> Excelente	
El gráfico ha sido generado.	

Comprobación:

1. El usuario accede al módulo configuración.

**Figura 2.70 Acceso al módulo de configuración**



2. El usuario selecciona la configuración correspondiente, a través de la cual desea la generación de gráficas dependiendo si son gráficas históricas o de transacciones.

### Transacciones

**Figura 2.71 Selección de configuración**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
	5	Reporte Histórico	2014-07-05 08:00:00	2015-08-22 00:00:00	1	Históricos

Creado por los Autores

### Históricos

**Figura 2.72 Transacciones tipo Históricas**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
	2	Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
	3	Reporte Histórico	2015-08-04 00:00:00	2015-08-22 00:00:00	1	Históricos

Creado por los Autores

3. El usuario selecciona la ubicación.

### Transacciones

**Figura 2.73 Selección de ubicación para transacciones**

Página principal Acerca de Ubicación Configuración Transacciones Históricos

GENERAR REPORTE

Parametros de Busqueda:  
Fecha Inicial: 2015-08-05 00:00:00  
Fecha Final: 2015-08-12 00:00:00

Selecciona la Ubicación:

LUAE

GENERAR REPORTE

Por Intervalo

Transacciones por minuto

TIEMPO REAL

GENERAR REPORTE

Tiempo Real

Transacciones por minuto

Creado por los Autores

## Históricos

**Figura 2.74 Selección de ubicación para Históricos**

**Creado por los Autores**

4. El sistema redirecciona a la página que contiene el gráfico.

## Transacciones

**Figura 2.75 Generación de gráficos transaccionales**



**Creado por los Autores**

## Históricos

**Figura 2.76 Generación de gráficos Históricos**



Creado por los Autores

### **CASO DE PRUEBA #11 “Generar gráficas en tiempo real”**

<b>Caso de Prueba</b>	
<b>Código:</b> 11	<b>Historia de Usuario:</b> 5 – Generar Gráficas Estadísticas
<b>Nombre Caso de Simulación:</b> Generar gráficas en tiempo real	
<b>Descripción:</b> Se procede a realizar la prueba para la generación de gráficas por clientes en tiempo real, los cuales están monitoreados por el aplicativo.	
<b>Prerrequisitos:</b> <ul style="list-style-type: none"> <li>• Tener por lo menos una ubicación registrada.</li> <li>• Que el cliente este generando tracciones.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b> <ol style="list-style-type: none"> <li>1. El usuario accede al módulo Transacciones.</li> <li>2. El usuario selecciona la ubicación.</li> <li>3. El sistema genera una gráfica en tiempo real.</li> </ol>	
<b>Resultado Esperado:</b> <ol style="list-style-type: none"> <li>1. La obtención de gráficas la cual toma datos del día actual.</li> </ol>	
<b>Resultado de la Prueba:</b> Muy Bueno  El gráfico ha sido generado con algunos problemas con el grafico de transacciones en tiempo real. En las figuras se muestra el gráfico sin errores.	

#### **Demostración:**

1. El usuario accede al módulo Transacciones.

Figura 2.77 Ingreso al módulo transacciones



2. El usuario selecciona la ubicación.

Figura 2.78 Selección de ubicación para gráficas en tiempo real



3. El sistema genera una gráfica en tiempo real.

Figura 2.79 Generación de gráfica en tiempo real



## CASO DE PRUEBA #12 “Generar reportes globales de todos los clientes”

<b>Caso de Prueba</b>	
<b>Código:</b> 12	<b>Historia de Usuario:</b> 5 – Generar Reportes Estadísticos
<b>Nombre Caso de Simulación:</b>	
Generar reportes globales de todos los clientes	
<b>Descripción:</b>	
Se procede a realizar la prueba para la generación de reportes globales de los clientes o aplicativos monitoreados por el sistema el sistema.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>• Tener por lo menos una ubicación registrada.</li> <li>• Tener Transacciones pertenecientes a una ubicación.</li> <li>• Tener una configuración registrada la cual tenga parámetros de filtración válidos.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>1. El usuario ingresa al sistema y accede al módulo configuración.</li> <li>2. El usuario selección la configuración correspondiente, a través de la cual desea la generación de reportes. En este punto es muy importante ya que puede obtener reportes históricos o de transacciones.</li> <li>3. El usuario selecciona la ubicación.</li> <li>4. El usuario selecciona generar reporte.</li> <li>5. Se redirección a la página que contiene el reporte.</li> </ol>	
<b>Resultado Esperado:</b>	
<ul style="list-style-type: none"> <li>• La obtención de reportes con los respectivos parámetros de búsqueda ingresados.</li> </ul>	
<b>Resultado de la Prueba:</b> Excelente	
El reporte ha sido obtenido.	

### Demostración:

1. El usuario ingresa al sistema y accede al módulo configuración.

Figura 2.80 Ingreso al módulo de configuración



2. El usuario selecciona la configuración correspondiente, a través de la cual desea la generación de reportes.

### Transacciones

**Figura 2.81 Selección de configuración de transacciones**

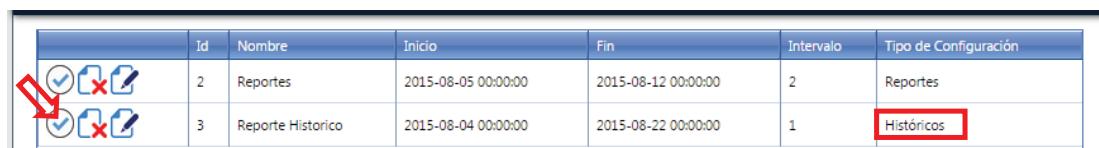


	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
	5	Reporte Historico	2014-07-05 08:00:00	2015-08-22 00:00:00	1	Históricos

**Creado por los Autores**

### Históricos

**Figura 2.82 Selección de configuración para reportes históricos**



	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
	2	Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
	3	Reporte Historico	2015-08-04 00:00:00	2015-08-22 00:00:00	1	Históricos

3. El usuario selecciona la ubicación.

### Transacciones

**Figura 2.83 Selección de ubicación para reportes transaccionales**



Página principal Acerca de Ubicación Configuración Transacciones Históricos

PARAMETROS DE BUSQUEDA:  
Fecha Inicial: 2015-08-05 00:00:00  
Fecha Final: 2015-08-12 00:00:00

Selección la Ubicación:

LUAE

GENERAR REPORTE

Por Intervalo

Transacciones por minuto

GENERAR REPORTE

Tiempo Real

Transacciones por minuto

**Creado por los Autores**

### Históricos

Figura 2.84 Selección de ubicación para reportes históricos



4. El usuario selecciona generar reporte.

Transacciones

Figura 2.85 Generación de reportes de transacción



Históricos

**Figura 2.86 Generación de reportes históricos**



Creado por los Autores

5. Se redirección a la página que contiene el reporte.

### Transacciones

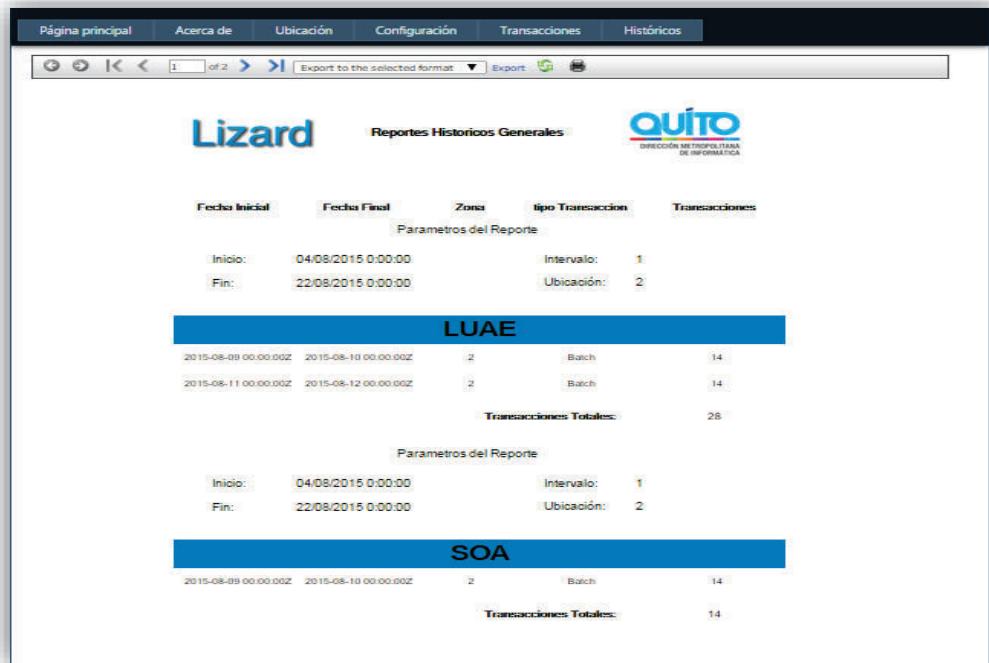
**Figura 2.87 Reporte por transacción**



Elaborado por los Autores

### Históricos

**Figura 2.88 Reporte histórico**



**Elaborado por los Autores**

### CASO DE PRUEBA #13 “Generar reportes por cliente”

<b>Caso de Prueba</b>	
<b>Código:</b> 13	<b>Historia de Usuario:</b> 5 – Generar Reportes Estadísticos
<b>Nombre Caso de Simulación:</b>	
Generar reportes por cliente	
<b>Descripción:</b>	
Se procede a realizar la prueba para la generación de reportes por clientes o aplicativos monitoreados por el sistema el sistema.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>• Tener por lo menos una ubicación registrada.</li> <li>• Tener Transacciones pertenecientes a una ubicación.</li> <li>• Tener una configuración registrada la cual tenga parámetros de filtración válidos.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>1. El usuario ingresa al sistema y accede al módulo configuración.</li> <li>2. El usuario selecciona la configuración correspondiente, a través de la cual desea la generación de reportes. En este punto es muy importante ya que puede obtener reportes históricos o de transacciones.</li> </ol>	

3. El usuario selecciona la ubicación. 4. El usuario selecciona generar reporte. 5. Se redirección a la página que contiene el reporte.
<b>Resultado Esperado:</b>
1. La obtención de reportes con los respectivos parámetros de búsqueda ingresados.

### Demostración:

1. El usuario ingresa al sistema y accede al módulo configuración.

**Figura 2.89 Ingreso al módulo de configuración**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
<input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
<input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	5	Reporte Histórico	2014-07-05 08:00:00	2015-08-22 00:00:00	1	Históricos

2. El usuario selección la configuración correspondiente, a través de la cual desea la generación de reportes.

### Transacciones

**Figura 2.90 Selección de configuración**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
<input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	4	Prueba Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
<input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	5	Reporte Histórico	2014-07-05 08:00:00	2015-08-22 00:00:00	1	Históricos

Creado por los Autores

### Históricos

**Figura 2.91 Selección de configuración de históricos**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
<input checked="" type="checkbox"/>	2	Reportes	2015-08-05 00:00:00	2015-08-12 00:00:00	2	Reportes
<input checked="" type="checkbox"/>	3	Reporte Historico	2015-08-04 00:00:00	2015-08-22 00:00:00	1	Históricos

**Creado por los Autores**

3. El usuario selecciona la ubicación.

### Transacciones

**Figura 2.92 Selección de ubicación de reportes transaccionales**

Página principal Acerca de Ubicación Configuración Transacciones Históricos

Parametros de Busqueda:  
Fecha Inicial: 2015-08-05 00:00:00  
Fecha Final: 2015-08-12 00:00:00

Seleccione la Ubicación:

GENERAL REPORTE

GENERAL REPORTE

GENERAL REPORTE

GENERAL REPORTE

Transacciones por minuto

Transacciones por minuto

LUAE

Tiempo Real

Por Intervalo

Creado por los Autores

Históricos

**Figura 2.93 Selección de ubicaciones históricas**

Página principal Acerca de Ubicación Configuración Transacciones Históricos

Parametros de Busqueda:  
Fecha Inicial: 2015-08-04 00:00:00  
Fecha Final: 2015-08-22 00:00:00

Seleccione la Ubicación:

GENERAL REPORTE H.

GENERAL REPORTE H.

GENERAL REPORTE H.

GENERAL REPORTE H.

Transacciones por día

Transacciones por día

LUAE

Reinas

Quito

DMI

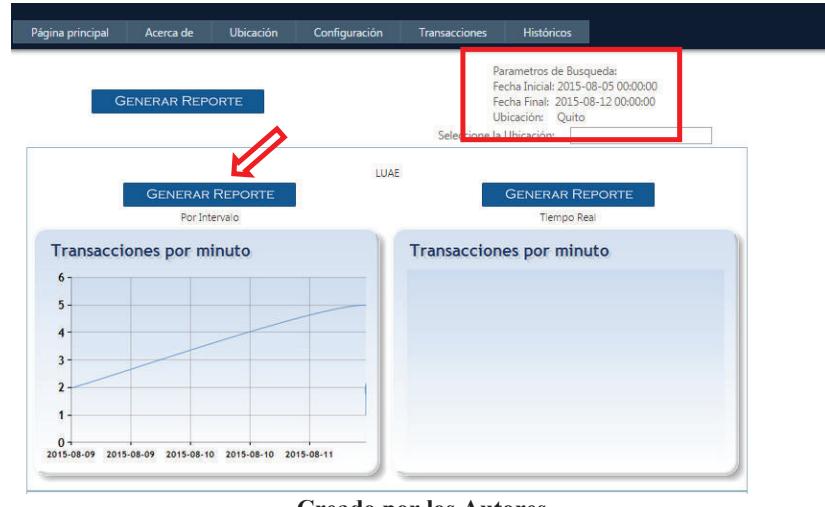
DMI Redes

Creado por los Autores

4. El usuario selecciona generar reporte.

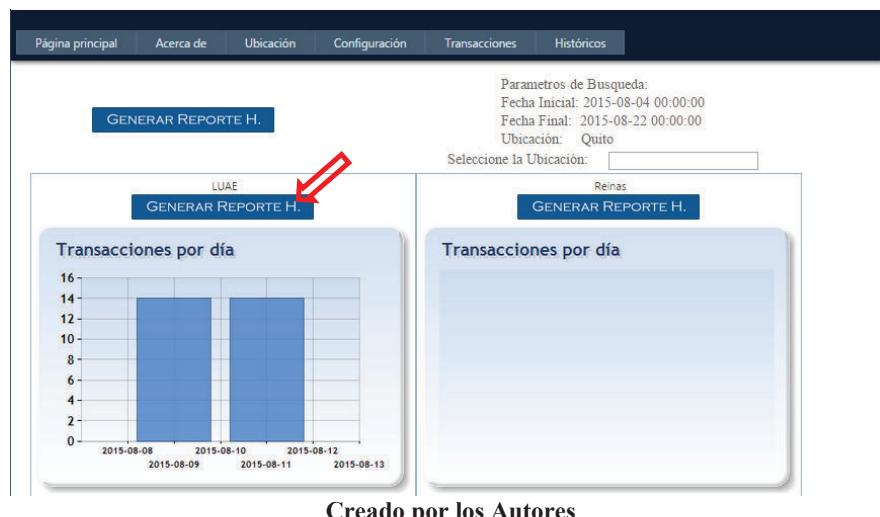
## Transacciones

**Figura 2.94 Generación de reporte transaccional**



## Históricos

**Figura 2.95 Generación de reporte histórico**



5. Se redirecciona a la página que contiene el reporte.

## Transacciones

**Figura 2.96 Reporte de transacciones**

The screenshot shows a web-based application titled "Lizard" with a header "Reportes Por Sistema" and the QUITO logo. The main content area displays a table of transaction data for the system LUAE. The table has columns: Fecha Inicial, Fecha Final, Zona, tipo Transaccion, and Transacciones. The parameters for the report are set to: Inicio: 05/08/2015 00:00, Fin: 12/08/2015 00:00, Ubicación: 2, and Intervalo: 2. The table data shows four entries, each with a timestamp from 2015-08-09 00:00:00Z to 2015-08-09 00:08:00Z, a zone value of 2, a type of "Batch", and a count of 2 transactions.

Fecha Inicial	Fecha Final	Zona	tipo Transaccion	Transacciones
2015-08-09 00:00:00Z	2015-08-09 00:02:00Z	2	Batch	2
2015-08-09 00:02:00Z	2015-08-09 00:04:00Z	2	Batch	2
2015-08-09 00:04:00Z	2015-08-09 00:06:00Z	2	Batch	2
2015-08-09 00:06:00Z	2015-08-09 00:08:00Z	2	Batch	2

**Creado por los Autores**

## Históricos

**Figura 2.97 Reportes históricos**

This screenshot shows a historical report for the system LUAE. The parameters are set to: Inicio: 04/08/2015 00:00, Fin: 22/08/2015 00:00, Sistema: LUAE, and Intervalo: 1. The table data shows two entries: one from 2015-08-09 00:00:00Z to 2015-08-10 00:00:00Z with 14 transactions, and another from 2015-08-11 00:00:00Z to 2015-08-12 00:00:00Z with 14 transactions. A summary row at the bottom indicates a total of 28 transactions. The footer of the page shows the date "jueves, 20 de agosto de 2015 12:41".

Fecha Inicial	Fecha Final	Zona	tipo Transaccion	Transacciones
2015-08-09 00:00:00Z	2015-08-10 00:00:00Z	2	Batch	14
2015-08-11 00:00:00Z	2015-08-12 00:00:00Z	2	Batch	14

Transacciones Totales: 28

**Creado por los Autores**

## CASO DE PRUEBA #14 “Generar reportes en tiempo real”

<b>Caso de Prueba</b>	
<b>Código:</b> 14	<b>Historia de Usuario:</b> 5 – Generar Reportes Estadísticos
<b>Nombre Caso de Simulación:</b>	

Generar reportes en tiempo real
<b>Descripción:</b> Se procede a realizar la prueba para la generación de reportes por clientes o aplicativos en tiempo real, los cuales están monitoreados por el aplicativo.
<b>Prerrequisitos:</b> <ul style="list-style-type: none"> <li>• Tener por lo menos una ubicación registrada.</li> <li>• Que el cliente este generando transacciones.</li> </ul>
<b>Entrada/Pasos de ejecución:</b> <ol style="list-style-type: none"> <li>1. El usuario accede al módulo Transacciones.</li> <li>2. El usuario selecciona la ubicación.</li> <li>3. El usuario selecciona generar reporte.</li> <li>4. Se redirección a la página que contiene el reporte.</li> </ol>
<b>Resultado Esperado:</b> <ul style="list-style-type: none"> <li>• La obtención de reportes el cual toma datos del día actual.</li> </ul>
<b>Resultado de la Prueba:</b> Muy Bueno  El reporte ha sido obtenido.  Para todos los reportes se omitió un gráfico para ver el resultado en forma visual.

#### Demostración:

1. El usuario accede al módulo Transacciones.

Figura 2.98 Ingreso al módulo transacciones



Creado por los Autores

2. El usuario selecciona la ubicación.

Figura 2.99 Selección de ubicación



3. El usuario selecciona generar reporte.

Figura 2.100 Generación de reporte en tiempo real



4. Se redirección a la página que contiene el reporte.

Figura 2.101 Reporte en tiempo real de clientes



Creado por los Autores

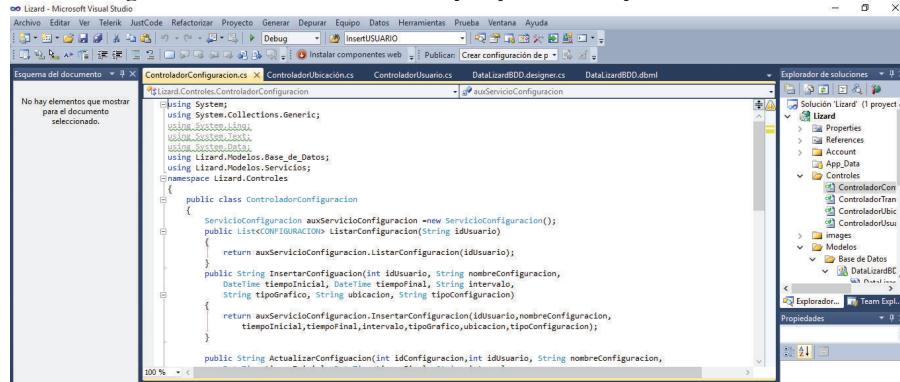
## CASO DE PRUEBA #15 “Customizar el Sistema”

<b>Caso de Prueba</b>	
<b>Código:</b> 15	<b>Historia de Usuario:</b> 6 – Configurar los Clientes
<b>Nombre Caso de Simulación:</b>	
Customizar el Sistema	
<b>Descripción:</b>	
Se procede a realizar la prueba de la configuración requerida para el funcionamiento y puesta en marcha del aplicativo monitoreando diversos clientes.	
<b>Prerrequisitos:</b>	
<ul style="list-style-type: none"> <li>• Tener el compilado del aplicativo es decir la DLL del proyecto.</li> <li>• Tener permisos de super administrador para la configuración de clientes.</li> </ul>	
<b>Entrada/Pasos de ejecución:</b>	
<ol style="list-style-type: none"> <li>1. El super administrador accede al cliente.</li> <li>2. El super administrador referencia la DLL del proyecto.</li> <li>3. El super administrador llama al método de captura de datos.</li> </ol>	
<b>Resultado Esperado:</b>	
<ul style="list-style-type: none"> <li>• El cliente realiza eventos monitoreados por el proyecto.</li> </ul>	
<b>Resultado de la Prueba:</b> Excelente	
Las transacciones del cliente se almacenan en la base de datos.	

Demostración:

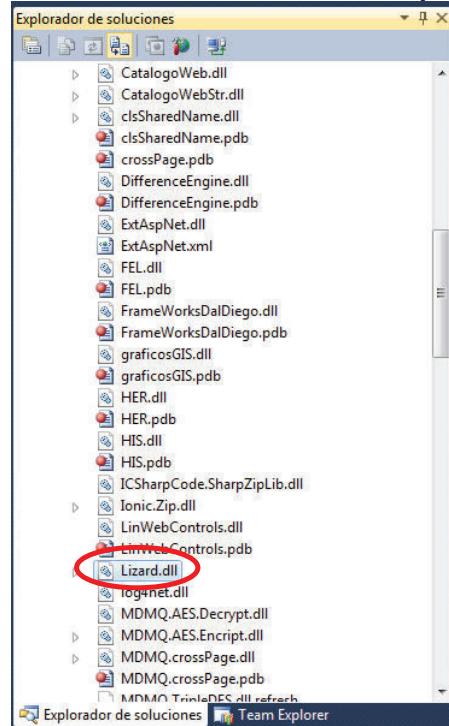
1. El super administrador accede al cliente.

Figura 2.102 Acceso a un cliente por parte del super administrador



Elaborado por los autores

2. El super administrador referencia la DLL del proyecto.

**Figura 2.103 Referenciación de la DLL del proyecto**

Elaborado por los autores

3. El súper administrador llama al método de captura de datos.

**Figura 2.104 Método de captura de datos**

```

TRANSACCION auxTransaccion = new TRANSACCION();
ControladorTransaccion cT = new ControladorTransaccion();
String dirIp = Request.ServerVariables["REMOTE_ADDR"];
string[] computer_name = System.Net.Dns.GetHostEntry(
(Request.ServerVariables["remote_host"]).HostName.Split(new Char[] { '.' }));
String nombreEquipo = computer_name[0].ToString();
String sistema = "Nombre del Sistema";
String tipoTransaccion = "Transacciones que se realizan";
String usuario = "Nombre del Usuario";
auxTransaccion.NOMBREQUIPO = nombreEquipo;
auxTransaccion.IPTRASACCION = dirIp;
auxTransaccion.TIOTRANSACCION = tipoTransaccion;
auxTransaccion.SISTEMA = sistema;
auxTransaccion.USUARIO = usuario;
cT.InsertarTransaccion(auxTransaccion);
    
```

Elaborado por los autores

## 2.4 ENTREGA DE LOS SPRINTS

### 2.4.1 PRESENTACIÓN DEL SPRINT CERO

Esta iteración se desarrolló con normalidad logrando el cumplimiento de las tareas asignadas en el tiempo y fechas establecidas. Se puede apreciar en la Figura 2.105 el estado terminado de las tareas propuestas.

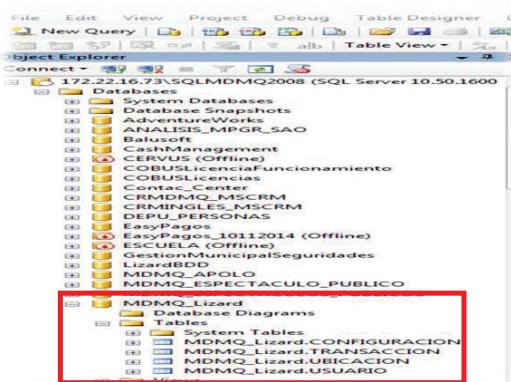
Figura 2.105 Cumplimiento de las Tareas Sprint Cero

Tarea	Tipo	Estado	Responsal
Implementacion de la Arquitectura d	Análisis	Terminada	Lenin, Marcel
Diseño de Diagrama Entidad-Relació	Prototipado	Terminada	Lenin, Marcel
Generación del modelo físico	Codificación	Terminada	Lenin, Marcel
Generación de scripts SQL	Codificación	Terminada	Lenin, Marcel
Implementación de procedimientos	Codificación	Terminada	Lenin, Marcel
Control de Calidad	Pruebas	Terminada	Lenin, Marcel
Afinación de la Base de Datos	Codificación	Terminada	Lenin, Marcel
Definición de roles y perfiles del s	Documentación	Terminada	Lenin, Marcel

Elaborado por los Autores

De esta iteración se obtiene la generación de Scripts de la base de datos y procedimientos almacenados, los cuales posteriormente son utilizados para la generación de los mismos con en la herramienta SQL Server. Ver Figura 2.106.

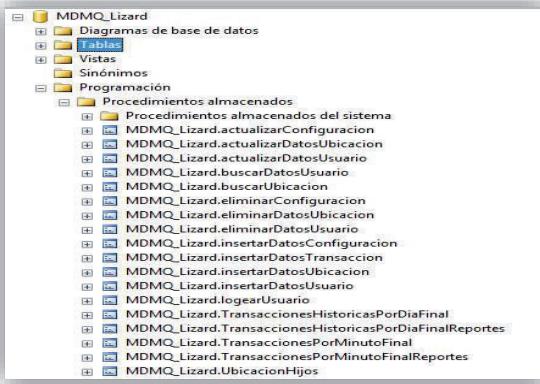
Figura 2.106 Esquema BDD SQL Server



Fuente: Tomado del Servidor del Municipio de Quito

También en la Figura 2.107 se puede apreciar los procedimientos almacenados ya generados en la herramienta mencionada.

**Figura 2.107 Página Ingreso Usuarios**



Elaborado por los Autores

#### 2.4.2 PRESENTACIÓN DEL PRIMER SPRINT

Este sprint se desarrolló con normalidad cumpliendo las tareas especificadas y en el tiempo determinado. Como se puede apreciar en la Figura 2.108.

**Figura 2.108 Cumplimiento Tareas Primer Sprint**

Tarea	Tipo	Estado	Responsable
Mapeo de las tablas y procedimientos	Análisis	Terminada	Lenin, Marcel
Implementación de los servicios de	Prototipado	Terminada	Lenin, Marcel
Implementación de controles para la	Prototipado	Terminada	Lenin, Marcel
Implementación del formulario de ing	Codificación	Terminada	Lenin, Marcel
Implementación del formulario regis	Pruebas	Terminada	Lenin, Marcel
Mapeo de las tablas y procedimientos	Codificación	Terminada	Lenin, Marcel
Implementación de los servicios de	Codificación	Terminada	Lenin, Marcel
Implementación de clases de acceso	Codificación	Terminada	Lenin, Marcel
Implementación del formulario de Ub	Codificación	Terminada	Lenin, Marcel
Control de calidad de los formularios	Codificación	Terminada	Lenin, Marcel
Afinamiento de formularios, servicios	Pruebas	Terminada	Lenin, Marcel
Implementación de pruebas	Pruebas	Terminada	Lenin, Marcel

Elaborado por los Autores

Entre las tareas realizadas en esta iteración se tiene la generación de tres formularios los cuales realizar el registro de usuarios, el ingreso al sistema y la gestión de ubicaciones. En seguida se presenta la página de registro de nuevos usuarios al sistema. Figura 2.109.

**Figura 2.109 Página Registro de Usuarios**

Crear una nueva cuenta

Use el formulario siguiente para crear una cuenta nueva.

Información de cuenta	
Nombre de usuario:	<input type="text" value="Lenin"/>
Contraseña:	<input type="password" value="..."/>
Confirmar contraseña:	<input type="password" value="..."/>
<input type="button" value="REGISTRAR USUARIO"/>	

Elaborado por los Autores

También se presenta la página que contiene el formulario del ingreso de usuarios registrados en la base de datos del sistema. Figura 2.110.

**Figura 2.110 Página Ingreso Usuarios**

Iniciar sesión

Especifique su nombre de usuario y contraseña. [Registrarse](#) si no tiene una cuenta.

Información de cuenta	
Nombre de usuario:	<input type="text"/>
Contraseña:	<input type="password"/>
<input type="checkbox"/> Mantenerme conectado	
<input type="button" value="INICIAR SESIÓN"/>	

Elaborado por los Autores

Como parte de la presentación de los entregables funcionales, se presenta el módulo de Ubicación la cual se muestra en la Figura 2.111.

**Figura 2.111 Página ASPX, Módulo UBICACIÓN**

Página principal Acerca de Ubicación Configuración Transacciones Históricos

ID	Ubicación	IP Inicial	IP Final	Acciones
2	Quito	192.168.1.1	192.168.1.255	
3	DMI	127.0.0.1	127.0.0.1	
5	DMI Redes			
6	DMI Redes Planta baja	172.20.47.10	172.20.47.50	

Ubicación: DMI Redes Planta baja  
IP Inicial: 172.20.47.10  
IP Final: 172.20.47.50

Elaborado por los Autores

Para ver el estado de aceptación de las tareas de este Sprint por parte del cliente ver la sección [Reuniones de Revisión](#).

#### 2.4.3 PRESENTACIÓN DEL SEGUNDO SPRINT

Como se aprecia en la Figura 2.112 las tareas del segundo sprint se realizaron con normalidad y en las fechas establecidas.

**Figura 2.112 Cumplimiento Tareas Segundo Sprint**

PILA DEL SPRINT				
Tarea	Tipo	Estado	Responsal	
Mapeo de las tablas y procedimientos	Análisis	Terminada	Lenin, Marcel	
Implementación de los servicios de	Prototipado	Terminada	Lenin, Marcel	
Implementación de clases de acceso	Prototipado	Terminada	Lenin, Marcel	
Implementación del formulario de Co	Codificación	Terminada	Lenin, Marcel	
control de calidad del formulario de C	Pruebas	Terminada	Lenin, Marcel	
Afinamiento página de Configuración	Codificación	Terminada	Lenin, Marcel	
Implementación de pruebas	Codificación	Terminada	Lenin, Marcel	

Elaborado por los Autores

En este Sprint de acuerdo a la planificación se presenta en la Figura 2.113 la creación del módulo de CONFIGURACION para el análisis personalizado de monitoreo de transacciones.

**Figura 2.113 Página ASPX, Módulo Configuración**

Página principal	Acerca de	Ubicación	Configuración	Transacciones	Históricos

Nombre:

Inicio:

Fin:

Intervalo:

Tipo:  Reportes

**REGISTRAR CONFIGURACIÓN**

Elaborado por los Autores

Para ver el estado de aceptación de las tareas de este Sprint por parte del cliente ver la sección [Reuniones de Revisión](#).

#### 2.4.4 PRESENTACIÓN DEL TERCER SPRINT

Para la presentación de esta iteración no hubo contratiempos, las tareas fueron realizadas correctamente y en el tiempo establecido como se muestra en la Figura 2.114.

Figura 2.114 Cumplimiento Tareas Tercer Sprint

Tarea	Tipo	Estado	Responsal
Mapeo de procedimientos almacena	Análisis	Terminada	Lenin, Marcel
Implementación del control web de g	Prototipado	Terminada	Lenin, Marcel
Implementación del formulario de Gr	Prototipado	Terminada	Lenin, Marcel
Implementación del control web de g	Codificación	Terminada	Lenin, Marcel
Implementación del formulario de Gr	Pruebas	Terminada	Lenin, Marcel
Control de calidad página de Gráfica	Codificación	Terminada	Lenin, Marcel
Afinamiento página de Gráficas	Codificación	Terminada	Lenin, Marcel
Implementación de pruebas	Codificación	Terminada	Lenin, Marcel

Elaborado por los Autores

Entre las tareas realizadas para este sprint destaca la creación del método de obtención de datos, utilizado en los clientes a través de una DLL la cual captura las transacciones, partiendo de esto se implementó el módulo Transacciones el cual es encargado de generar gráficas de transacciones de los clientes. En la Figura 2.115 se muestra el módulo de transacciones.

Para ver el estado de aceptación de las tareas de este Sprint por parte del cliente ver la sección [Reuniones de Revisión](#).

Figura 2.115 Página ASPX, Módulo Transacciones



Elaborado por los Autores

#### 2.4.5 PRESENTACIÓN DEL CUARTO SPRINT

Esta iteración final se realizó con normalidad completando las tareas y respetando las fechas establecidas en la planificación para su respectiva entrega. En la Figura 2.116 se muestra la consecución de las tareas de este sprint.

**Figura 2.116 Cumplimiento Tareas Cuarto Sprint**

PILA DEL SPRINT			
Tarea	Tipo	Estado	Responsal
Implementación reporte genérico de	Análisis	Terminada	Lenin, Marcel
Implementación de formulario de rep	Prototipado	Terminada	Lenin, Marcel
Implementación reporte genérico His	Prototipado	Terminada	Lenin, Marcel
Implementación de formulario de rep	Codificación	Terminada	Lenin, Marcel
Control de calidad página de Reporte	Codificación	Terminada	Lenin, Marcel
Afinamiento página de Reportes	Pruebas	Terminada	Lenin, Marcel
Implementación de pruebas	Codificación	Terminada	Lenin, Marcel

Elaborado por los Autores

En este Sprint se presentó los formularios para la generación de reportes los cuales se generan a partir del módulo Gráficas Transaccionales e Históricas. La finalidad de los reportes es mostrar de una manera más detallada el rendimiento de trabajo de cada cliente.

Como se puede ver en la Figura 2.117 se muestra la generación de reportes a partir de la tabla Transacciones de la base de datos.

**Figura 2.117 Pagina de Reportes**



Elaborado por los Autores

Para ver el estado de aceptación de las tareas de este Sprint por parte del cliente ver la sección [Reuniones de Revisión](#).

#### 2.4.6 REUNIONES DE REVISIÓN

En las reuniones de revisión se analizan las tareas propuestas en las historias de usuario, las cuales son presentadas al cliente y conjuntamente con el equipo de desarrollo se determina el estado de aceptación de cada tarea por iteración. Dando así los siguientes estados: aceptado, rechazado y pendiente.

*Nota: Debido a que las tareas del Sprint Cero no tiene requisitos funcionales, este Sprint no es tomado en cuenta en la revisión.*

Como se puede ver se presenta la aceptación del cliente por cada iteración.

##### 2.4.6.1 Aceptación Del Primer Sprint

En la tabla 2.10 se muestra el estado de aceptación del cliente de las tareas correspondiente a este Sprint. Como todas las tareas se aceptaron sin ningún problema no fue necesario que alguna de estas tareas pase al siguiente Sprint.

**Tabla 2.10 Estado de aceptación del Primer Sprint**

No	Tarea	Estimado de Aceptación
9	Mapeo de las tablas y procedimientos almacenados de Usuarios	Aceptado
10	Implementación de los servicios de Usuarios	Aceptado
11	Implementación de controles para la gestión de Usuarios	Aceptado
12	Implementación del formulario de ingreso	Aceptado
13	Implementación del formulario registro de usuarios	Aceptado
14	Mapeo de las tablas y procedimientos almacenados de Ubicación	Aceptado
15	Implementación de los servicios de Ubicación	Aceptado
16	Implementación de clases de acceso para la gestión de Ubicación	Aceptado
17	Implementación del formulario de Ubicación	Aceptado
18	Control de calidad de los formularios, servicios y controles	Aceptado
19	Afinamiento de formularios, servicios y controles	Aceptado
20	Implementación de pruebas	Aceptado

Elaborado por los Autores

##### 2.4.6.2 Aceptación Del Segundo Sprint

En la tabla 2.11 se muestra el estado de aceptación del cliente de las tareas correspondiente a este Sprint. Como todas las tareas se aceptaron sin ningún problema no fue necesario que alguna de estas tareas pase al siguiente Sprint.

**Tabla 2.11 Estado de aceptación del Segundo Sprint**

No	Tarea	Estado de Aceptación
21	Mapeo de las tablas y procedimientos almacenados de Configuración	Aceptado
22	Implementación de los servicios de Configuración	Aceptado
23	Implementación de clases de acceso para la gestión de Configuración	Aceptado
24	Implementación del formulario de Configuración	Aceptado
25	control de calidad del formulario de Configuración	Aceptado
26	Afinamiento página de Configuración	Aceptado
27	Implementación de pruebas	Aceptado

**Elaborado por los Autores**

#### **2.4.6.3 Aceptación Del Tercer Sprint**

En la tabla 2.12 se muestra el estado de aceptación del cliente de las tareas correspondiente a este Sprint. Como todas las tareas se aceptaron sin ningún problema no fue necesario que alguna de estas tareas pase al siguiente Sprint.

**Tabla 2.12 Estado de aceptación del Tercer Sprint**

No	Tarea	Estado de Aceptación
28	Mapeo de procedimientos almacenados de Gráficas	Aceptado
29	Implementación de Servicios y controles para la captura de transacciones	Aceptado
30	Implementación del control web de gráficos Transacciones	Aceptado
31	Implementación del formulario de Gráficas Transaccionales	Aceptado
32	Implementación del control web de gráficos Históricos	Aceptado
33	Implementación del formulario de Gráficas Históricas	Aceptado
34	Control de calidad página de Gráficas	Aceptado
35	Afinamiento página de Gráficas	Aceptado
36	Implementación de pruebas	Aceptado

**Elaborado por los Autores**

#### 2.4.6.4 Aceptación Del Cuarto Sprint

En la tabla 2.13 se muestra el estado de aceptación del cliente de las tareas correspondiente a este Sprint. Como todas las tareas se aceptaron sin ningún problema no fue necesario que alguna de estas tareas pase al siguiente Sprint.

**Tabla 2.13 Estado de aceptación del Cuarto Sprint**

No	Tarea	Estado de Aceptación
37	Implementación reporte genérico de Transacción	Aceptado
38	Implementación de formulario de reporte de Transacción	Aceptado
39	Implementación reporte genérico Históricos	Aceptado
40	Implementación de formulario de reporte Histórico	Aceptado
41	Control de calidad página de Reportes	Aceptado
42	Afinamiento página de Reportes	Aceptado
43	Customización del Sistema	Aceptado
44	Implementación de pruebas	Aceptado

Elaborado por los Autores

#### 2.4.7 REUNIONES DE RETROSPECTIVA

En cuanto a las reuniones de retrospectiva que se realizan a final de cada sprint en la cual participaron todos los miembros del equipo con el fin de evaluar y mejorar la productividad del equipo y la calidad del producto presentado en cada iteración en las cuales se destacaron los siguientes puntos para cada Sprint.

##### 2.4.7.1 Retrospectiva del Primer Sprint

###### Información de la empresa y proyecto:

Empresa / Organización	Distrito Metropolitano de Informática
Proyecto	LIZARDBDD

###### Información de la reunión:

Lugar	Municipio de Quito
Fecha	10/08/2015

Número de iteración / sprint	Primera
Personas convocadas a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez
Personas que asistieron a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez

## Reunión retrospectiva

Tabla 2.14 Reunión de Retrospectiva Primer Sprint

¿Qué salió bien en la iteración? (aciertos)	¿Qué no salió bien en la iteración? (errores)	¿Qué mejoras vamos a implementar en la próxima iteración?
Correcta Planificación y ejecución del acceso de usuarios y registro tanto de ubicación.	Vinculación de la aplicación con el Directorio Activo del DMI.	Permisos necesarios para cualquier acción a realizarse.

Fuente: Tomado de [www.pmoinformatica.com](http://www.pmoinformatica.com) y modificado por los autores

### 2.4.7.2 Retrospectiva del Segundo Sprint

#### Información de la empresa y proyecto:

Empresa / Organización	Distrito Metropolitano de Informática
Proyecto	LIZARDBDD

#### Información de la reunión:

Lugar	Municipio de Quito
Fecha	01/09/2015
Número de iteración / sprint	Segundo
Personas convocadas a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez
Personas que asistieron a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez

## Reunión retrospectiva

Tabla 2.15 Reunión de Retrospectiva Segundo Sprint

¿Qué salió bien en la iteración? (aciertos)	¿Qué no salió bien en la iteración? (errores)	¿Qué mejoras vamos a implementar en la próxima iteración?
Correcta implementación de	Ninguna	Interfaz más amigable con el usuario.

¿Qué salió bien en la iteración? (aciertos)	¿Qué no salió bien en la iteración? (errores)	¿Qué mejoras vamos a implementar en la próxima iteración?
configuraciones personalizadas por usuario		

Fuente: Tomado de [www.pmoinformatica.com](http://www.pmoinformatica.com) y modificado por los autores

#### 2.4.7.3 Retrospectiva del Tercer Sprint

##### Información de la empresa y proyecto:

Empresa / Organización	Distrito Metropolitano de Informática
Proyecto	LIZARDBDD

##### Información de la reunión:

Lugar	Municipio de Quito
Fecha	28/09/2015
Número de iteración / sprint	Tercero
Personas convocadas a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez
Personas que asistieron a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez

#### Reunión retrospectiva

Tabla 2.16 Reunión de Retrospectiva Tercer Sprint

¿Qué salió bien en la iteración? (aciertos)	¿Qué no salió bien en la iteración? (errores)	¿Qué mejoras vamos a implementar en la próxima iteración?
Correcta generación de gráficas de transacciones por clientes	Se requirió un afinamiento de la generación de gráficas transaccionales	Las tareas de la iteración debieron ser reajustadas

Fuente: Tomado de [www.pmoinformatica.com](http://www.pmoinformatica.com) y modificado por los autores

#### 2.4.7.4 Retrospectiva del Cuarto Sprint

##### Información de la empresa y proyecto:

Empresa / Organización	Distrito Metropolitano de Informática
Proyecto	LIZARDBDD

### Información de la reunión:

Lugar	Municipio de Quito
Fecha	20/10/2015
Número de iteración / sprint	Cuarto
Personas convocadas a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez
Personas que asistieron a la reunión	Lenin Guillermo Samaniego Vizcaíno Marcelo Javier Cano Sánchez

### Reunión Retrospectiva

Tabla 2.17 Reunión de Retrospectiva Cuarto Sprint

¿Qué salió bien en la iteración? (aciertos)	¿Qué no salió bien en la iteración? (errores)	¿Qué mejoras vamos a implementar en la próxima iteración?
Correcta generación de reportes por cliente	Implementación de reportes en tiempo real	-----

Fuente: Tomado de [www.pmoinformatica.com](http://www.pmoinformatica.com) y modificado por los autores

## CAPÍTULO 3

### 3 EVALUACIÓN DEL SISTEMA EN EL CASO DE ESTUDIO

#### 3.1 RECOPILACIÓN DE DATOS DE MONITOREO

##### 3.1.1 SISTEMA A MONITOREAR

**Caso de aplicación**

Municipio del Distrito Metropolitano de Quito (MDMQ)

**Aplicativo a monitorear**

Sistema Integrado de Registro Catastral de Quito (SIREC-Q)

**Dirección encargada**

Dirección metropolitana de Catastro

**Responsable**

Geovanny Baño

**Objetivo del sistema (SIREC-Q)**

Generación de avalúos para terrenos y construcciones en las 8 administraciones zonales del Distrito Metropolitano de Quito.

**Tipos de Usuarios**

Personal autorizados miembros del área de Catastros.

**Descripción del caso (SIREC-Q)**

EL sistema SIREC-Q, encargado de la generación de avalúos para terrenos y construcciones, este sistema realiza tres funciones: Trámites, Solicitudes y Actualización de Información de Catastros, siendo esta última la función más utilizada.

SIREC-Q es un sistema usado por 200 usuarios los cuales atienden a un promedio de mil reclamos por día. Entendiéndose como reclamos a la actualización de datos, propietarios, terrenos y construcciones.

La siguiente ilustración muestra el acceso al sistema. Ver figura 3.1

**Figura 3.1 Acceso al sistema SIREC-Q**



Fuente: Sistema SIREC-Q

Una vez accedido al aplicativo se tiene el formulario de inicio. Ver figura 3.2.

**Figura 3.2 SIREC-Q del Municipio de Quito**



Fuente: Tomado del Distrito Metropolitano de Informática

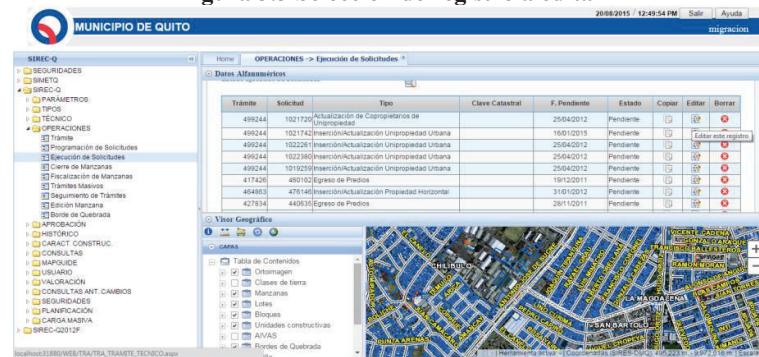
### 3.1.2 DATOS A RECOLPILAR

Para la recopilación de datos es necesario tener preparado el sistema a monitorear “SIREC-Q” y el aplicativo que monitorea “Lizard”. Para esto se realiza un diseño en que detalla el método de captura de transacciones el cual se instancia en el aplicativo “SIREC-Q”, además de la configuración del aplicativo “Lizard” para la obtención de gráficos y reportes.

El sistema SIREC-Q realiza tres tipos de operaciones: Administración de Trámites, Gestión de Solicitudes y Actualización de Información de Catastros. A partir de esto se tiene que clasificar a una transacción como una de estas operaciones.

En este caso práctico se trabajó con la actualización de solicitudes en el cual se define una transacción de tipo “Actualización”. Ver Figura 3.3.

**Figura 3.3 Selección de registro a editar**



**Fuente:** Sistema SIREC-Q

Una vez que se definió el tipo de transacción y el nombre del sistema monitoreado se procede al análisis del método de captura de transacciones.

#### 3.1.2.1 Método De Captura De Transacciones

Este método realiza la función de capturar una transacción, para lo es necesario enviar un elemento de tipo transacción en el cual se define los siguientes

parámetros: dirección IP, nombre del computador, sistema, tipo de transacción, usuario, fecha y hora.

Dirección IP, este parámetro se autogenera a través de una librería propia de las páginas ASPX, para la obtención de una dirección IP se utilizará la siguiente línea.

```
String dirIp = Request.ServerVariables["REMOTE_ADDR"];
```

Nombre del Computador, este parámetro se genera a partir de una librería propia de las páginas ASPX, para la obtención del nombre del computador se utilizará la siguiente línea.

```
string[] computer_name = System.Net.Dns.GetHostEntry(Request.ServerVariables["remote_host"]).HostName.Split(new Char[] { '.' });
String nombreEquipo = computer_name[0].ToString();
```

Sistema, este parámetro es un String el cual es definido por el super administrador, en este caso se utilizó “SIRECQ”.

```
String sistema = "SIRECQ";
```

Tipo de transacción, para este parámetro se usó el tipo de transacción definido anteriormente que en este caso es de Actualización.

```
String tipoTransaccion = "Actualización";
```

Usuario, este parámetro es un String, en esta versión del aplicativo no realiza ninguna función, no obstante en versiones posteriores el super administrador podrá capturar el nombre del usuario que está generando transacciones. En este caso se envió a SIRECQ como usuario predefinido.

```
auxTransaccion.USUARIO = usuario;
```

Para el cálculo de los últimos parámetros como hora, fecha y asignación de zona, la DLL “Lizard.dll” realiza operaciones en background encargadas de la generación de estos parámetros.

### **3.1.2.2 Diseño De La Configuración De Lizard**

Como punto de partida en el diseño de una configuración optima se necesita establecer tres parámetros: una ubicación, una configuración para el módulo de reportes y una configuración para el modulo Histórico.

#### **Ubicación**

Como punto de partida se tiene el siguiente escenario: se tuvo el acceso a una copia local del aplicativo SIRECQ en nuestro equipo, lo cual significa que las IPs capturadas van a ser IPs locales es decir localhost. Por lo cual se diseñó la siguiente ubicación.

Nombre de la ubicación: DMI Prueba SIREC-Q

IP inicial: 127.0.0.1

IP final: 127.0.0.1

#### **Configuración para el módulo de reportes**

Para el correcto uso de esta configuración se estableció un rango de fechas a partir de las cuales se genera la obtención de gráficas y reportes, además se asignó un intervalo de agrupación de transacciones de un minuto esto significa que cada minuto en el rango de fechas definidas se va a contabilizar el número de transacciones por minuto y como resultado graficarlo. Para esto se diseñó la siguiente configuración.

Nombre: Prueba

Fecha Inicio: 2015-10-01 00:00:00 a. m.

Fecha Fin: 2015-10-06 00:00:00 a. m.

Tipo: Reportes

### **Configuración para el módulo de históricos**

Del mismo modo que en el módulo de reporte se estableció un rango de fechas y un intervalo, la diferencia entre las dos configuraciones es el intervalo que en este caso es un día. Lo cual quiere decir que las transacciones se agruparan por día generando un gráfico. Para esto se diseñó la siguiente configuración.

Nombre: Prueba Históricos

Fecha Inicio: 2015-10-01 00:00:00 a. m.

Fecha Fin: 2015-10-06 00:00:00 a. m.

Tipo: Históricos

## **3.2 INSTALACIÓN DEL SISTEMA**

Este sistema es una aplicación web por lo cual no es instalable, sin embargo para su funcionamiento es necesaria la instanciación de una DLL la cual contiene un método que realiza la captura de transacciones. Para el correcto funcionamiento de este método es necesario un script en el cual se realiza el paso de parámetros del cliente.

La instanciación del script debe ser realizada por una persona que conozca la estructura del cliente, ya que el script debe ser insertado en un evento específico, debido a esto no es posible crear un instalador automático que inserte el script en cada uno de los sistemas clientes. Esto se debe a que cada cliente tiene una estructura diferente; no obstante este método realiza llamadas a funciones propias de las páginas aspx por lo cual no es posible su compactación en métodos del lenguaje utilizado (.cs). A continuación se procede a explicar los pasos para la puesta en marcha del sistema.

### 3.2.1 CONFIGURACIÓN DEL ENTORNO

Procedemos a realizar el levamiento de ambiente de prueba, esta simulación nos permitirá determinar si el sistema está en condiciones de ser implementado en producción.

#### 3.2.1.1 Hardware Utilizado En La Simulación

##### **Servidor 1**

Este servidor de aplicaciones contendrá el proyecto, al cual los clientes enviaran solicitudes para ser atendidas

###### *Características*

IP: 172.22.4.31

RAM: 8 GB

PROCESADOR: Intel Core I5 3.07GHz

DISCO: 500GB

##### **Servidor 2**

Este servidor de base de datos del proyecto, el cual están todos los datos del proyecto

###### *Características*

IP: 172.22.16.73

RAM: 32 GB

PROCESADOR: Intel Core I5 3.07GHz

DISCO: 500GB

##### **Cliente**

Este cliente es cualquier computadora que tenga conexión física con la intranet en el entorno de trabajo.

###### *Características*

IP: 172.22.48.23

RAM: 32 GB

PROCESADOR: Intel Core I5 3.07GHz

DISCO: 250GB

TARJETA ETHERNET: Realtek PCIe FE Family Controller

### **3.2.1.2 Software Utilizado En La Simulación**

Servidor 1

*Características*

Sistema Operativo: Windows 7 Professional Service Pack 1

Servidor Web: Microsoft IIS

SDK: .Net Framework 4.0

Servidor 2

*Características*

Sistema Operativo: Windows 7 Professional Service Pack 1

Servidor Web: Microsoft IIS

SDK: .Net Framework 4.0

Cliente

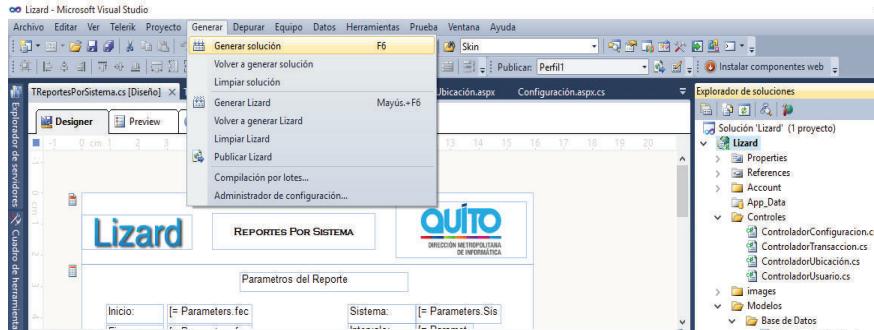
*Características*

NAVEGADOR WEB: Mozilla Firefox 14.0.1 y Google Chrome 46.0.1 o superior

(Recomendados)

### **3.2.2 COMPILACION DE LIZARD**

Como punto de partida para la instalación del aplicativo se necesita una DLL a través de la cual se realiza la captura de transacciones. Para obtener la DLL fue necesario compilar el proyecto y generar una solución. Ver Figura 3.4.

**Figura 3.4 Compilación de la Solución****Elaborado por los Autores**

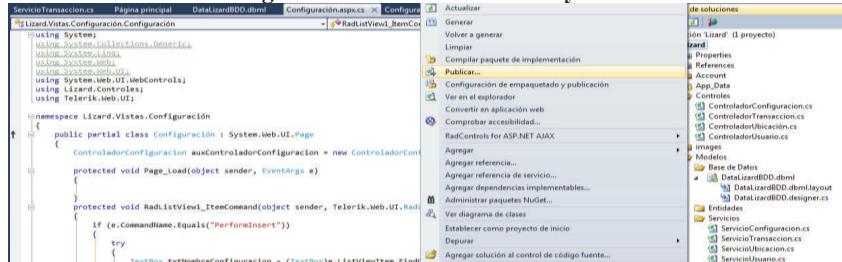
Una vez generada la solución del proyecto se genera una DLL la cual es compatible con la mayoría de proyectos desarrollados en visual estudio. Ver Figura 3.5.

**Figura 3.5 Generación de DLL de la Solución**

Nombre	Fecha de modificación	Tipo	Tamaño
AjaxControlToolkit.dll	5/7/2015 10:30	Extensión de la apl...	5.027 KB
AjaxMin.dll	23/2/2015 9:18	Extensión de la apl...	424 KB
Lizard.dll	2/9/2015 13:09	Extensión de la apl...	191 KB
Lizard	2/9/2015 13:09	Program Debug D...	254 KB
Telerik.Web.UI.dll	8/11/2010 15:31	Extensión de la apl...	18.523 KB

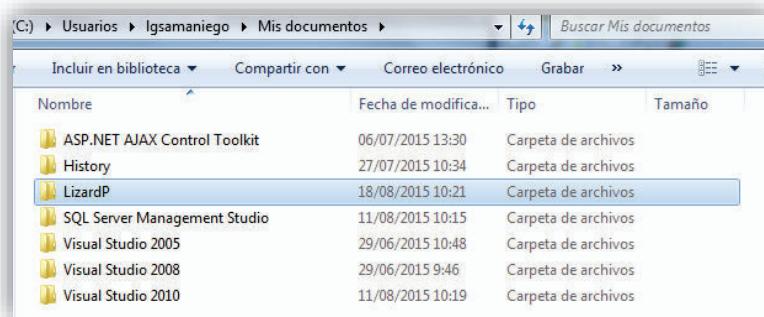
**Elaborado por los Autores**

Una vez que se tiene la DLL se procede con la generación del precompilado del proyecto para lo cual se ingresa al aplicativo y se lo publica. Ver Figura 3.6.

**Figura 3.6 Publicación del Proyecto****Elaborado por los Autores**

En el proceso de publicación se debe señalar la dirección en la que se guarde el precompilado del proyecto, aquí se debe tener muy en cuenta esta dirección, ya que el precompilado del proyecto es el que se publica en el servidor. Figura 3.7.

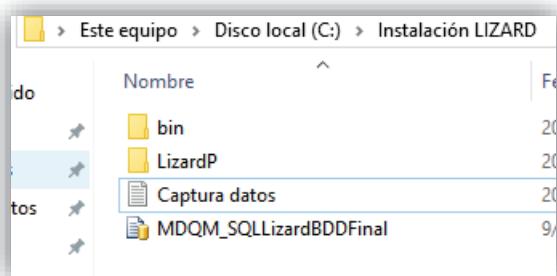
**Figura 3.7 Precompilado de la Solución**



**Elaborado por los autores**

Una vez que se obtiene la DLL para la captura de transacciones y el precompilado del proyecto, se procede a copiarlos en la carpeta “Instalación LIZARD”, además de esto a esta carpeta se adjunta el SCRIPT de la generación de la BDD y un blog de notas el cual contiene un SCRIPT para la captura de transacciones. Ver figura 3.8.

**Figura 3.8 Carpeta Instalación LIZARD**



**Elaborado por los Autores**

En esta carpeta se tiene todo lo necesario para la instalación del sistema a partir de este punto se procede con la configuración del entorno y la puesta en marcha del sistema.

### 3.2.3 PUESTA EN MARCHA

Para la puesta en marcha del sistema se parte con el uso del script “MDQM\_SQLLizardBDDFinal.sql” para la generación de la base de datos. Este SCRIPT se encuentra en la carpeta “Instalación LIZARD”, además deberá ser ejecutado en el servidor de base de datos el cual deberá poseer SQL Server 2008 o versiones posteriores. Ver figura 3.9.

Figura 3.9 SCRIPt para generar la BDD

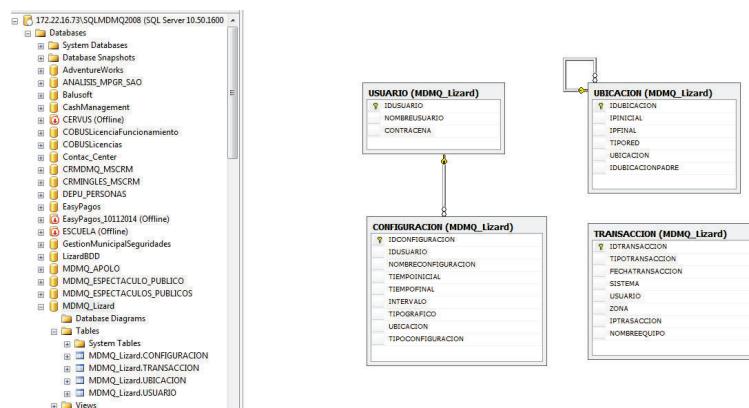
```
--Consulta para la creación de la BDD con este script se crea la base
-- de datos ocupada por el sistema
/*
 * DBMS name: Microsoft SQL Server 2008
 * Created on: 28/10/2014 9:27:15
 */
/*
drop database MDQM_Lizard
go
create database MDQM_Lizard
go
use MDQM_Lizard
GO
set dateformat ymd
go
```

Elaborado por los Autores

**Nota:** la generación de este SCRIPt solo se la hace en caso que no exista la Base de Datos del aplicativo “LIZARD”.

Luego de esto nuestra base de datos tendrá el esquema el cual será usado para el funcionamiento del sistema. Ver figura 3.10.

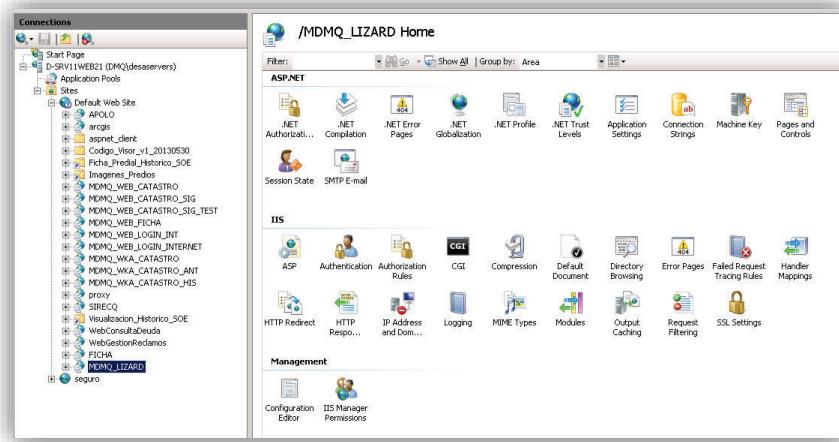
Figura 3.10 Infraestructura de la base de datos de la Solución



Elaborado por los autores

Una vez que se tiene la base de datos lista se procede con la publicación del precompilado en el servidor web. Para esto se utiliza el precompilado obtenido anteriormente, el cual se debe publicar en el servidor web IIS. Ver figura 3.11.

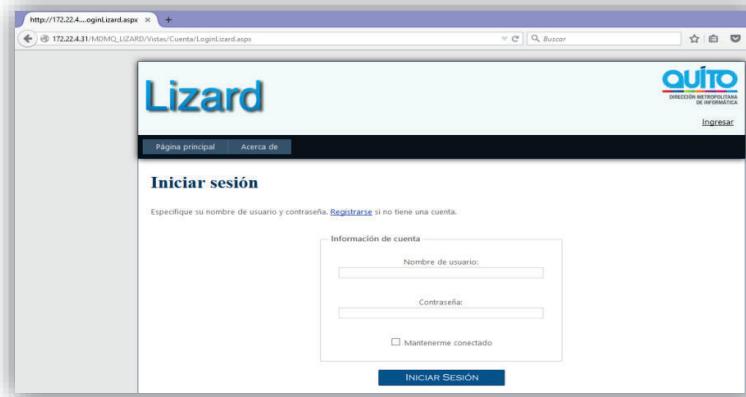
**Figura 3.11 Servidor Web IIS**



Fuente: Tomado y modificado por los autores

Para finalizar se comprueba que el aplicativo web este corriendo bajo la IP del servidor web. Ver Figura 3.12.

**Figura 3.12 Sistema corriendo desde el servidor**



Fuente Servidor Web

### 3.2.4 CAPTURA DE DATOS

Para realizar la captura de datos es necesario el uso de la DLL obtenida en el aplicativo la cual se almaceno en la carpeta “Instalación LIZARD”. Esta DLL es compatible con la mayoría de proyectos desarrollados en visual estudio.

Ver Figura 3.13.

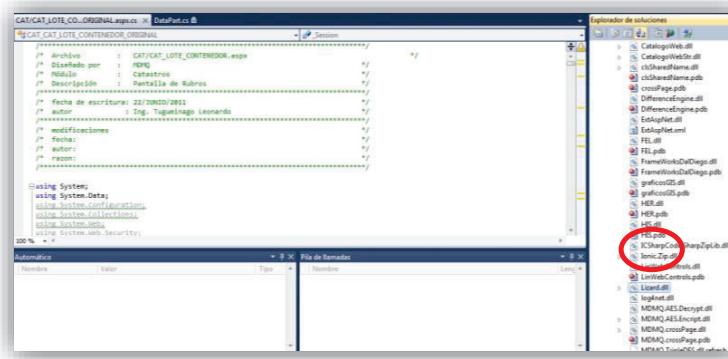
**Figura 3.13 Lizard.dll**

e equipo > Disco local (C:) > Instalación LIZARD > bin			
Nombre	Fecha de modifica...	Tipo	Tamaño
AjaxControlToolkit.dll	5/7/2015 10:30	Extensión de la apl...	5.027 KB
AjaxMin.dll	23/2/2015 9:18	Extensión de la apl...	424 KB
Lizard.dll	20/9/2015 17:46	Extensión de la apl...	191 KB
Lizard	20/9/2015 17:46	Program Debug D...	228 KB
Telerik.Web.UI.dll	8/11/2010 15:31	Extensión de la apl...	18.523 KB

Elaborado por los Autores

Partiendo de esta DLL procede a instanciarla en el cliente “SIREC Q”, para lo cual fue necesario el acceso al proyecto DMQ\_SIRECQ. Al cual se referencia la DLL “Lizard.dll”. Figura 3.14.

**Figura 3.14 Referencia DLL “Lizard.dll”**



**Fuente proyecto SIRECQ**

Una vez referenciada la DLL se procede con la adición de la cadena de conexión para lo cual se utiliza el editor de texto para mostrar el archivo “Captura Datos.txt” el cual se encuentra adjunto en la carpeta “Instalación LIZARD”. Ver figura 3.15.

**Figura 3.15 “Captura Datos”**

```

Captura datos: Bloc de notas
Archivo Edición Formato Ver Ayuda
//Cadena de Conexión
<add name="MDMQ_LizardConnectionString" connectionString="Data Source=DESKTOP-BLA62R1;
Initial Catalog=MDMQ_Lizard;Persist Security Info=True;User ID=sa;Password=nada1234"
providerName="System.Data.SqlClient" />

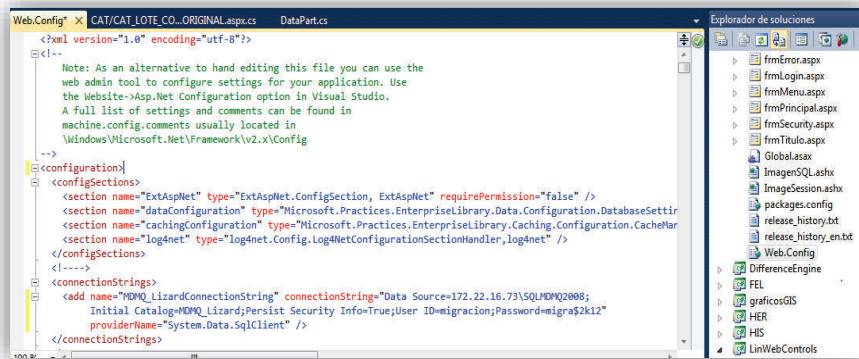
//Método de captura de Transacciones
ControladorTransaccion auxTransaccion = new ControladorTransaccion();
string dirIp = Request.ServerVariables["REMOTE_ADDR"];
string[] computer_name = System.Net.Dns.GetHostEntry(Request.ServerVariables["remote_host"]).HostName.Split(new Char[] { '.' });
String sistema = computer_name[0];
String tipoTransaccion = "SIRECQ";
String nombreEquipo = computer_name[0].ToString();
String auxTransaccion = "SIRECQ";
String usuario = "SIRECQ";
auxTransaccion.NOMBREQUIPO = nombreEquipo;
auxTransaccion.IPTRANSACCION = dirIp;
auxTransaccion.TIPOTRANSACCION = tipoTransaccion;
auxTransaccion.SISTEMA = sistema;
auxTransaccion.USUARIO = usuario;
ct.InsertarTransaccion(auxTransaccion);

```

Elaborado por los Autores

Para adjuntar la cadena de conexión se accedió al web.config del proyecto “DMQ\_SIRECQ” y se añadió la cadena de conexión. Ver Figura 3.16.

**Figura 3.16 Adición de Cadena de Conexión**



Fuente Aplicativo SIREQ

Como paso final para la captura de transacciones se hace la llamada al método de captura, al cual se pasa los parámetros que determinan una transacción. Este método se añadió al evento que permite actualizar una programación de solicitudes.

```

protected void btnLOTE_Aceptar_Click(object sender, EventArgs e)
{
    // Se comprueba el valor del campo oculto HF_EDICION_GRAFICA
    // Si almacena "true", se puede continuar con el proceso de guardado de los datos alfanuméricos
    // Si almacena "false", no se continúa
    if (HF_EDICION_GRAFICA.Value.ToString() == "true")
    {

```

```

HFValidar_Est.Value = string.Empty;
cargarProvinvia();
cargarCanton();
}

#region LIZARD

TRANSACCION auxTransaccion = new TRANSACCION();
ControladorTransaccion cT = new ControladorTransaccion();
String dirIp = Request.ServerVariables["REMOTE_ADDR"];
string[] computer_name
System.Net.Dns.GetHostEntry(Request.ServerVariables["remote_host"]).HostName.Split(new Char[] { '.' });
String nombreEquipo = computer_name[0].ToString();
String sistema = "SIREQ";
String tipoTransaccion = "Lotes";
String usuario = "SIREQ";
auxTransaccion.NOMBREQUIPO = nombreEquipo;
auxTransaccion.IPTRASACCION = dirIp;
auxTransaccion.TIPOTRANSACCION = tipoTransaccion;
auxTransaccion.SISTEMA = sistema;
auxTransaccion.USUARIO = usuario;
cT.InsertarTransaccion(auxTransaccion);

#endregion
}

```

Una vez que se tiene configurado el cliente es necesario configurar el aplicativo de manera que capture y grafique las transacciones del cliente, para lo cual se accede al aplicativo e ingresa al módulo de ubicación. Ver figura 3.17.

**Figura 3.17 Modulo Ubicación Aplicativo Lizard**

Id	Ubicación	Ip Inicial	Ip Final	
2	Quito	192.162.168.1	192.162.168.255	
4	DMI			
1004	DMI.Redes			
2004	Prueba Manual de Usuario			

Ubicación: \_\_\_\_\_  
 Ip inicial: \_\_\_\_\_  
 Ip final: \_\_\_\_\_  
 REGISTRAR UBICACIÓN

**Fuente Aplicativo Lizard**

En este módulo es necesario definir una ubicación, la cual permite la captura de transacciones en base a un rango de IPs. En este caso se ha puesto la IP local, esto se debe a que el aplicativo Sirec-Q solo se encuentra funcionando en la máquina de manera local ya que maneja información delicada por lo que solo se tuvo acceso a una copia de este. Ver Figura 3.18.

**Figura 3.18 Ubicación DMI Prueba SIRECQ**

Página principal	Acerca de	Ubicación	Configuración	Transacciones	Históricos
2	Ubicación	Quito	Ip Inicial	192.162.168.1	Ip Final
4	DMI Prueba SIRECQ		127.0.0.1		127.0.0.1
1004	DMI_Redes				
2004	Prueba Manual de Usuario				

Ubicación: DMI Prueba SIRECQ  
Ip Inicial: 127.0.0.1  
Ip Final: 127.0.0.1  
REGISTRAR UBICACIÓN

**Elaborado por los Autores**

Una vez que se tiene lista la ubicación se accede al módulo Configuración. Ver Figura 3.19.

**Figura 3.19 Modulo Configuración**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
<input checked="" type="checkbox"/>	10	Prueba	2015-10-01 00:00:00 a. m.	2015-10-06 00:00:00 a. m.	1	Reportes
<input checked="" type="checkbox"/>	11	Prueba históricos	2015-10-01 00:00:00 a. m.	2015-10-06 00:00:00 a. m.	1	Históricos

**Fuente Aplicativo Lizard**

En este módulo se añade dos configuraciones con parámetros de filtración válidos las cuales permitan el uso de los módulos Transacciones e Históricos. Ver Figura 3.20.

**Figura 3.20 Registro de configuraciones**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
<input checked="" type="checkbox"/>	10	Prueba	2015-10-01 00:00:00 a. m.	2015-10-06 00:00:00 a. m.	1	Reportes
<input checked="" type="checkbox"/>	11	Prueba históricos	2015-10-01 00:00:00 a. m.	2015-10-06 00:00:00 a. m.	1	Históricos
<input checked="" type="checkbox"/>	1006	Reporte Prueba SIRECQ	2015-08-20 00:00:00 a. m.	2015-08-22 00:00:00 a. m.	1	Reportes
<input checked="" type="checkbox"/>	1007	Reporte Prueba SIRECQ	2015-08-20 00:00:00 a. m.	2015-08-22 00:00:00 a. m.	1	Históricos

Nombre:   
 Inicio:    
 Fin:    
 Intervalo:   
 Tipo: Reportes   
 REGISTRAR CONFIGURACIÓN

**Elaborado por los Autores**

Las configuraciones registradas permiten la generación de gráficos y reportes. Ver Figura 3.21.

**Figura 3.21 Configuraciones Registradas**

	10	Prueba	2015-10-01 00:00:00 a. m.	2015-10-06 00:00:00 a. m.	1	Reportes
	11	Prueba historicos	2015-10-01 00:00:00 a. m.	2015-10-06 00:00:00 a. m.	1	Historicos

**Elaborado por los Autores**

Con el método de captura de datos y el aplicativo configurado se procede a realizar la captura de transacciones para lo cual se ingresa como un cliente y se ejecuta el método de captura, el cual está alojado él en evento de actualizar programación de solicitudes.

La aplicación SIREC-Q alojada en el Distrito Metropolitano de Informática contiene el llamado al evento para la captura de datos de monitoreo. Ver Figura 3.22.

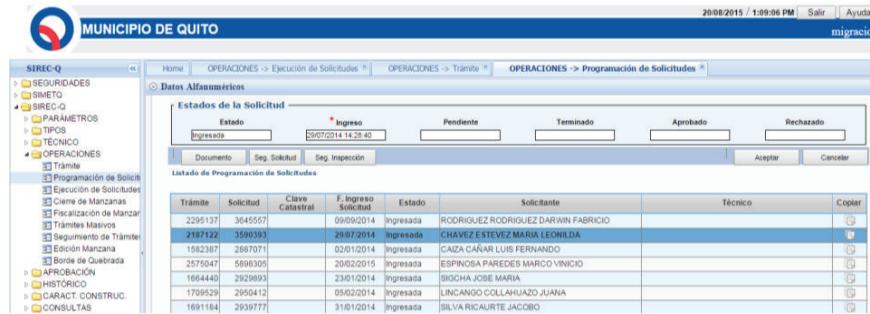
**Figura 3.22 Aplicación de la DMI**



**Fuente:** Distrito Metropolitano de Informática

Primero se ingresa hacia la programación de soluciones para modificar un lote registrado en esta aplicación. Ver Figura 3.23.

Figura 3.23 Sistema SIREC-Q



Fuente: Distrito Metropolitano de Informática

Luego se selecciona el registro se procede a editarlo. Ver Figura 3.24.

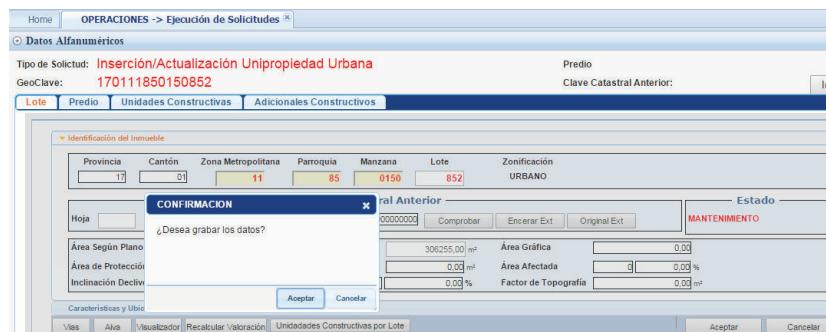
Figura 3.24 Sistema SIREC-Q Edición de Lotes



Fuente: Aplicación SIRECQ

Una vez editados los datos se procede a confirmar la edición, en ese momento se produce el evento de llamado a nuestra DLL, el cual realiza la captura de datos. Ver Figura 3.25.

Figura 3.25 Confirmación de Edición



Fuente: Aplicación SIRECQ

Este proceso se realizó varias veces, y para verificar que los datos hayan sido capturados se realiza un Select en la base de datos. Ver Figura 3.26.

**Figura 3.26 Transacciones registradas en la Base de datos LIZARDBDD**

IDTRANSACTION	TIPOTRANSACTION	FECHATRANSACTION	SISTEMA	USUARIO	ZONA	IPTRANSACTION	NOMBREEQUIPO	
22	22	Actualización	2015-10-05 09:18:15.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
23	23	Actualización	2015-10-05 09:18:50.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
24	24	Actualización	2015-10-05 09:38:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
25	25	Actualización	2015-10-05 10:00:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
26	26	Actualización	2015-10-05 11:08:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
27	27	Actualización	2015-10-05 11:15:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
28	28	Actualización	2015-10-05 11:15:22.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
29	29	Actualización	2015-10-05 11:15:44.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
30	30	Actualización	2015-10-05 11:15:56.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
31	31	Actualización	2015-10-05 12:09:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
32	32	Actualización	2015-10-05 12:18:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
33	33	Actualización	2015-10-05 13:08:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
34	34	Actualización	2015-10-05 13:38:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006
35	35	Actualización	2015-10-05 13:58:10.123	SIRECQ	SIRECQ	4	127.0.0.1	PC11INGS006

Elaborado por los autores

Cabe recalcar que la DLL es muy versátil ya que puede ser instanciada en cualquier evento de cualquier cliente.

### 3.3 PRUEBAS DEL SISTEMA DE MONITOREO

Luego de realizar la instalación del sistema se procede a realizar u conjunto de pruebas las cuales sirven para verificar la tolerancia del sistema en cuanto a su carga, el tiempo de ejecución que el aplicativo tendrá cuando maneje volúmenes grandes de información y la aceptación del cliente.

#### 3.3.1 PRUEBAS DE ACEPTACIÓN

Para estas pruebas se parte de la instalación del sistema en la cual se realizaba todos los pasos previos para la captura de transacciones del aplicativo SIRECQ, partiendo de esto se probó los módulos que determinan la funcionalidad del sistema.

**Nota:** La toma de las pruebas se la realizo en 5 días a partir de los cuales se capturaron 35 transacciones. Para ver la configuración del aplicativo y captura de transacciones de SIRECQ ver sección [Instalación del sistema](#).

### 3.3.1.1 Módulo Reporte

En este módulo a partir de la captura de datos se obtuvo la generación de gráficas, las cuales determinan el funcionamiento del aplicativo SIRECQ. Ver Figura 3.27.

**Figura 3.27 Grafico de transacciones del SIRECQ**



Fuente Aplicativo Lizard

Partiendo del grafico transacciones por minuto se obtuvo el siguiente reporte. Ver Figura 3.28.

**Figura 3.28 Reporte de las transacciones del SIREC-Q**

The figure shows a screenshot of a report titled 'Lizard' under 'REPORTES POR SISTEMA'. The report is for the 'SIRECQ' system. It displays a table of transaction details with columns: FECHA INICIAL, FECHA FINAL, ZONA, TIPO TRANSAKCION, and TRANSACCIONES. The data is as follows:

FECHA INICIAL	FECHA FINAL	ZONA	TIPO TRANSAKCION	TRANSACCIONES
2015-10-01 00:00:00Z	2015-10-01 00:00:00Z	4	ACTUALIZACIÓN	1
2015-10-01 00:00:00Z	2015-10-01 00:00:00Z	4	ACTUALIZACIÓN	2
2015-10-01 09:00:00Z	2015-10-01 09:00:00Z	4	ACTUALIZACIÓN	1
2015-10-01 10:00:00Z	2015-10-01 10:00:00Z	4	ACTUALIZACIÓN	1
2015-10-02 08:00:00Z	2015-10-02 08:00:00Z	4	ACTUALIZACIÓN	1
2015-10-02 09:00:00Z	2015-10-02 09:00:00Z	4	ACTUALIZACIÓN	1

Fuente Aplicativo Lizard

Partiendo del grafico transacciones por minuto real se obtuvo el siguiente reporte. Ver figura 3.29.

**Figura 3.29 Reporte de las transacciones en tiempo real del SIREC-Q**

REPORTES POR SISTEMA

Sistema: SIRECQ

Parametros del Reporte

Inicio:	5/10/2015 0:00:00	Sistema:	SIRECQ	
Fin:	5/10/2015 17:19:51	Intervalo:	1	
<b>SIRECQ</b>				
FECHA INICIAL	FECHA FINAL	ZONA	TIPO TRANSACCION	TRANSACCIONES
2015-005 09:18:00Z	2015-005 09:19:00Z	4	ACTUALIZACIÓN	3
2015-005 09:38:00Z	2015-005 09:39:00Z	4	ACTUALIZACIÓN	1
2015-005 10:00:00Z	2015-005 10:01:00Z	4	ACTUALIZACIÓN	1
2015-005 11:00:00Z	2015-005 11:01:00Z	4	ACTUALIZACIÓN	1
2015-005 11:15:00Z	2015-005 11:16:00Z	4	ACTUALIZACIÓN	2
2015-005 12:00:00Z	2015-005 12:01:00Z	4	ACTUALIZACIÓN	1
2015-005 12:18:00Z	2015-005 12:19:00Z	4	ACTUALIZACIÓN	1
2015-005 13:00:00Z	2015-005 13:01:00Z	4	ACTUALIZACIÓN	1
2015-005 13:38:00Z	2015-005 13:39:00Z	4	ACTUALIZACIÓN	1
2015-005 13:58:00Z	2015-005 13:59:00Z	4	ACTUALIZACIÓN	1
TRANSACCIONES TOTALES:				15

Fuente Aplicativo Lizard

### 3.3.1.2 Módulo Histórico

En este módulo a partir de la captura de datos se obtuvo la generación del siguiente gráfico, el cual determinó el funcionamiento del aplicativo SIREC-Q en dos días de trabajo. Ver Figura 3.30.

**Figura 3.30 Grafico Histórico de las Transacciones del SIRECQ**



Fuente Aplicativo Lizard

Seguido del grafico transacciones por día se obtuvo el siguiente reporte. Ver Figura 3.31.

**Figura 3.31 Reporte Histórico del SIRECQ**

Fuente Aplicativo Lizard

### 3.3.2 PRUEBAS DE RESISTENCIA

Estas pruebas sirven para determinar la carga que soporta el sistema, por lo cual se tomó el número de transacciones que soporta cada módulo, en este punto se realizó un análisis en base a los módulos que determinan la funcionalidad del sistema que son “Históricos” y “Reportes”. Para esto se establecieron una configuración por modulo. Ver figura 3.32.

**Figura 3.32 Configuraciones para pruebas de resistencia**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
	10	Prueba	2015-09-23 00:00:00 a. m.	2015-09-24 00:00:00 a. m.	1	Reportes
	11	Prueba históricos	2015-09-23 00:00:00 a. m.	2015-09-24 00:00:00 a. m.	1	Históricos

Fuente Aplicativo Lizard

#### 3.3.2.1 Módulo Reporte

Para este módulo se realizó un conjunto de pruebas en las cuales se comprueba la funcionalidad del sistema. La siguiente tabla muestra lo resultados de funcionamiento obtenido en base al número de transacciones.

**Tabla 3.1 Resultados prueba de resistencia módulo reportes  
Módulo (Reportes)**

<b>Transacciones</b>	
<b>100</b>	Funciona
<b>200</b>	Funciona
<b>300</b>	Funciona
<b>1000</b>	Funciona
<b>2000</b>	Funciona
<b>3000</b>	Funciona
<b>10000</b>	Funciona
<b>20000</b>	Funciona
<b>30000</b>	Funciona
<b>100000</b>	No Funciona
<b>200000</b>	No Funciona
<b>300000</b>	No Funciona
<b>1000000</b>	No Funciona

Elaborado por los autores

Como se puede observar la carga máxima soportada en el módulo de reporte es de 30000 transacciones. Además los resultados obtenidos por el modulo fueron los siguientes. Ver figura 3.33.

**Figura 3.33 Grafico de Prueba con 30000 Transacciones**



Elaborado por los Autores

En la siguiente figura se puede observar la generación de reportes en la cual se tiene el flujo de tracciones las cuales fueron insertadas para esta prueba. Ver figura 3.34.

Figura 3.34 Reporte con 30000 Transacciones



Elaborado por los Autores

### 3.3.2.2 Módulo Histórico

Para este módulo se realizó un conjunto de pruebas en las cuales se comprueba la funcionalidad del sistema. La siguiente tabla muestra los resultados de funcionamiento obtenido en base al número de transacciones. Ver Tabla 3.2.

Tabla 3.2 Resultados prueba de resistencia módulo histórico

#### Módulo (Histórico)

Transacciones	
<b>100</b>	Funciona
<b>200</b>	Funciona
<b>300</b>	Funciona
<b>1000</b>	Funciona
<b>2000</b>	Funciona
<b>3000</b>	Funciona
<b>10000</b>	Funciona
<b>20000</b>	Funciona
<b>30000</b>	Funciona
<b>100000</b>	Funciona
<b>200000</b>	Funciona
<b>300000</b>	Funciona
<b>1000000</b>	Funciona

Elaborado por los autores

En este módulo no se pudo determinar una carga máxima ya que su funcionamiento siempre fue óptimo. Además los resultados obtenidos por el modulo fueron los siguientes. Ver Figura 3.35.

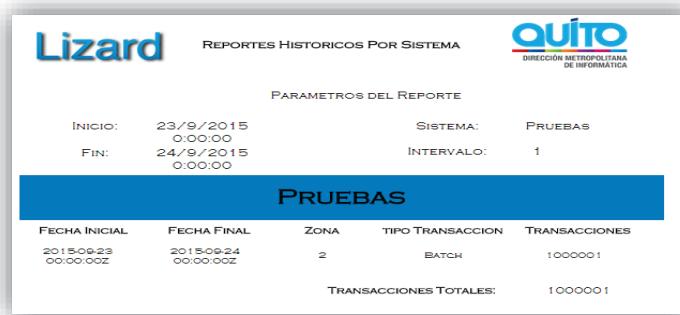
**Figura 3.35 Grafico de Prueba con 1000000 de Transacciones**



Elaborado por los Autores

En la siguiente figura se puede observar la generación de reportes en la cual se tiene el flujo de tracciones las cuales fueron insertadas para esta prueba. Ver Figura 3.36.

**Figura 3.36 Reporte Histórico con 1000000 de transacciones**



Fuente Aplicativo Lizard

### 3.3.3 PRUEBAS DE RENDIMIENTO

Estas pruebas sirven para determinar la carga que soporta el sistema en función del tiempo de respuesta, por lo cual se tomó el número de transacciones que soporta cada módulo en base a su tiempo de respuesta, en este punto se realizó un análisis en base a los módulos que determinan la funcionalidad del sistema que son “Históricos” y “Reportes”. Para esto se establecieron una configuración por modulo. Ver figura 3.37.

**Figura 3.37 Configuraciones para pruebas de resistencia**

	Id	Nombre	Inicio	Fin	Intervalo	Tipo de Configuración
	10	Prueba	2015-09-23 00:00:00 a.m.	2015-09-24 00:00:00 a.m.	1	Reportes
	11	Prueba históricos	2015-09-23 00:00:00 a.m.	2015-09-24 00:00:00 a.m.	1	Históricos

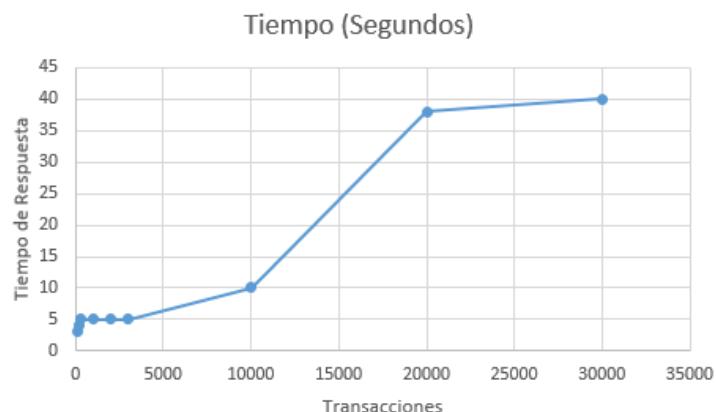
Fuente Aplicativo Lizard

### 3.3.3.1 Módulo Reporte

Para este módulo se realizó un conjunto de pruebas en las cuales se comprueba la funcionalidad del sistema. La siguiente tabla muestra los resultados de funcionamiento obtenido en base al número de transacciones y tiempo de respuesta. Ver Tabla 3.3.

**Tabla 3.3 Pruebas de rendimiento de reportes**

<b>Modulo (Reportes)</b>	
Transacciones	Tiempo (Segundos)
100	3
200	4
300	5
1000	5
2000	5
3000	5
10000	10
20000	38
30000	40
100000	No Funciona
200000	No Funciona
300000	No Funciona
1000000	No Funciona



Elaborado por los autores

Como se puede observar a medida que se tiene más carga el tiempo de respuesta aumenta llegando así a un límite de 30000 transacciones a partir de la cual el modulo deja de funcionar. Los resultados obtenidos por el modulo fueron los siguientes. Ver figura 3.38.

**Figura 3.38 Grafico de Prueba con 30000 Transacciones**



Fuente Aplicativo Lizard

En la siguiente figura se puede observar la generación de reportes en la cual se tiene el flujo de tracciones las cuales fueron insertadas para esta prueba. Ver Figura 3.39.

**Figura 3.39 Reporte con 30000 Transacciones**

Lizard		REPORTES POR SISTEMA		QUITO DIRECCIÓN METROPOLITANA DE INFORMÁTICA	
Parametros del Reporte					
Inicio:	23/9/2015 0:00:00	Sistema:	Rendimiento		
Fin:	24/9/2015 0:00:00	Intervalo:	1		
<b>RENDIMIENTO</b>					
FECHA INICIAL	FECHA FINAL	ZONA	TIPO TRANSACCION	TRANSACCIONES	
2015-09-23 14:18:00Z	2015-09-23 14:19:00Z	2	BATCH	1	
2015-09-23 14:19:00Z	2015-09-23 14:20:00Z	2	BATCH	10000	
2015-09-23 14:20:00Z	2015-09-23 14:21:00Z	2	BATCH	10000	
2015-09-23 14:21:00Z	2015-09-23 14:25:00Z	2	BATCH	10000	
TRANSACCIONES TOTALES:					30000

Fuente Aplicativo Lizard

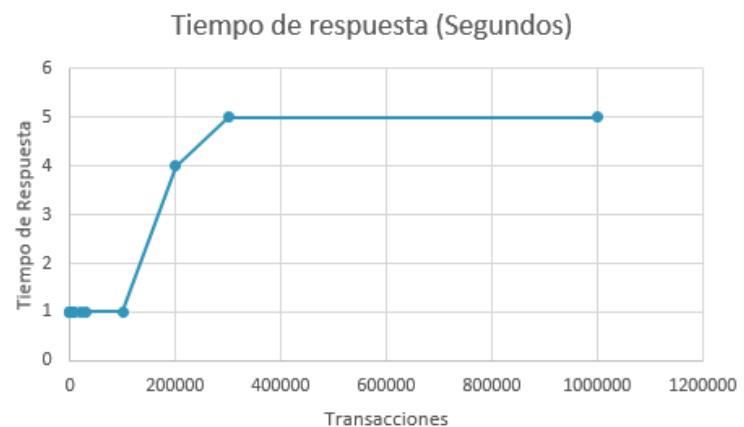
### 3.3.3.2 MÓDULO HISTÓRICO

Para este módulo se realizó un conjunto de pruebas en las cuales se comprueba la funcionalidad del sistema. La siguiente tabla muestra los resultados

de funcionamiento obtenido en base al número de transacciones y tiempo de respuesta. Ver Tabla 3.4.

**Tabla 3.4 Pruebas de rendimiento de reportes**

<b>Modulo (Reportes)</b>	
<b>Transacciones</b>	Tiempo (Segundos)
<b>100</b>	1
<b>200</b>	1
<b>300</b>	1
<b>1000</b>	1
<b>2000</b>	1
<b>3000</b>	1
<b>10000</b>	1
<b>20000</b>	1
<b>30000</b>	1
<b>100000</b>	1
<b>200000</b>	4
<b>300000</b>	5
<b>1000000</b>	5



Elaborado por los autores

En este módulo no se pudo determinar una carga máxima ya que su funcionamiento siempre fue óptimo. Además los resultados obtenidos por el modulo fueron los siguientes. Ver figura 3.40.

**Figura 3.40 Grafico de Prueba con 1000000 Transacciones**



Elaborado por los Autores

En la siguiente figura se puede observar la generación de reportes en la cual se tiene el flujo de tracciones las cuales fueron insertadas para esta prueba. Ver figura 3.41.

**Figura 3.41 Reporte con 1000000 Transacciones**



Elaborado por los Autores

## 3.4 ANALISIS DE RESULTADOS

Luego de realizar las pruebas correspondientes, se procedemos a analizar los resultados obtenidos teniendo también una serie de criterios de valoración para cada una de las pruebas realizadas.

### 3.4.1 CRITERIOS PARA A VALORACIÓN

Para poder realizar la evaluación se debe establecer los rangos en los cuales estarán los porcentajes para poder realizar el análisis de resultados con respecto a criterios definidos. Tabla 3.5.

**Tabla 3.5 Valoración de Aceptabilidad**

Escala	Valor	Nivel de Aceptación
0% - 50%	Malo	Inaceptable

<b>51% - 79%</b>	Regular	Inaceptable
<b>80% - 89%</b>	Bueno	Aceptable
<b>90% - 100%</b>	Excelente	Optimo

**Elaborado por los Autores**

Basándose en esta valoración para que un sistema sea aceptado por el cliente debe tener sus aspectos a valorar en las diferentes pruebas como bueno y excelente representando al estado aceptable y óptimo para su entrega.

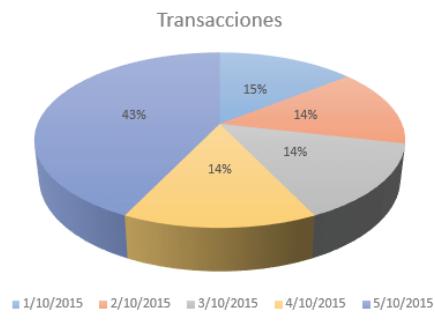
### 3.4.2 ANALISIS DE ACEPTACIÓN

En base a los resultados obtenidos en las pruebas de aceptación se pudo analizar los módulos de Reportes e Históricos. Como parámetro inicial se tiene una muestra de 35 transacciones capturadas en 5 días. Como se puede apreciar en la Tabla 3.6.

**Tabla 3.6 Transacciones Pruebas Aceptación**

<b>Fecha</b>	<b>Transacciones</b>
1/10/2015	5
2/10/2015	5
3/10/2015	5
4/10/2015	5
5/10/2015	15

**Elaborado por los Autores**



### Modulo Reportes

En este módulo se calificó dos funcionalidades en los cuales se puede ver y analizar el funcionamiento del sistema. La primera funcionalidad es la encargada de la generación de gráficas en un rango de tiempo mientras que la segunda es la generación de graficas en tiempo real. Entonces se analizara la primera funcionalidad. Ver figura 3.42.

**Figura 3.42 Generación de Gráfica por Rangos**



Elaborado por los Autores

Con ayuda de este grafico 3.42 el administrador puede determinar el flujo de carga del sistema en el rango de tiempo establecido, no obstante para este caso el sistema tuvo una carga mayor en el último día mientras que en los días posteriores tubo una carga de una transacción, esto se debe a que el intervalo de la toma de transacciones está establecido en 1 minuto, por lo cual el número máximo de transacciones en el primer día fue de 2, el segundo, tercero y cuarto día fue de una mientras que el último día fue de 4 Transacciones. El siguiente grafico muestra la generación de graficas en tiempo real. Ver Figura 3.43.

**Figura 3.43 Generación de Grafica en tiempo real**



Elaborado por los Autores

Como se puede observar este gráfico 3.43 muestra de una manera más detallada el día actual de monitoreo, en resumen esta funcionalidad permite al administrador identificar las horas pico y de uso del sistema, también permite observar cuanta carga tiene el sistema, dando como resultado la visualización del flujo de carga en mediciones que simulan el tiempo real.

## Modulo Histórico

En este módulo se calificó la generación de graficas en intervalos un intervalo de un día. Tomando en cuenta esto se obtuvo el siguiente resultado. Ver Figura 3.44.

**Figura 3.44 Generación de Gráfica Histórica**



Elaborado por los Autores

Como se puede observar este tipo de grafico sirve para que el administrador observe el flujo de carga del sistema por lapsos de días, en resumen esta funcionalidad permite ver fechas son las que tiene mayor carga, además que permite observar el funcionamiento del sistema en grandes intervalos de tiempo.

Dado la valoración para la aceptabilidad del Cliente en cuanto al sistema de monitoreo se determinó los siguientes valores tanto para el módulo de Reportes como para el módulo Históricos antes descritos.

**Tabla 3.7 Valoración de Aceptación**

Aspecto	Módulo Reporte	Módulo Histórico
Diseño	Bueno	Bueno
Eficiencia	Excelente	Excelente
Utilidad	Bueno	Excelente
Manejo	Excelente	Excelente

Elaborado por los Autores

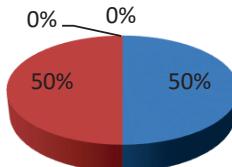
Como se puede apreciar tanto el módulo de Reporte como Histórico presentan una aprobación óptima del 62% y aceptable del 38% y estados inaceptables del

0% con estos aspectos presentados al cliente. Determinando así que el sistema realiza su función de monitoreo de clientes de forma aceptable.

**Figura 3.45 Ponderado de Aceptación del Sistema**

### Ponderado de Aceptación

■ excelente ■ bueno ■ regular ■ malo



Elaborado por los Autores

Además de la generación de graficas cada módulo tiene una funcionalidad adicional de generar reportes, no obstante esta funcionalidad no está tomada en cuenta ya que su objetivo es prácticamente el mismo que la generación de gráficas, cambiando únicamente la forma de apreciación de las transacciones de una manera más detallada. Debido a los criterios de valoración antes propuestos para el sistema se deduce que el ponderado final de la aplicación para su aceptación es del 91.25%.

### 3.4.3 ANALISIS DE RESISTENCIA

Para este análisis se realizó un conjunto de pruebas definidas en la sección pruebas de resistencia a partir de esta pruebas se pudo determinar la carga de transacciones que soportan los módulos de reportes e históricos.

#### Modulo Reporte

En este módulo se tuvo una carga máxima soportada de 30000 transacciones, con lo cual determinamos que este módulo, no tiene un gran alcance, ya que en empresas como Google y Facebook los usuarios generan millones de

transacciones. A partir de este resultado se pudo determinar que la usabilidad del aplicativo es para empresas de pequeño tamaño; lo cual se adapta de excelente forma a este caso de estudio ya que el ambiente es de una intranet.

Nota: para aumentar la resistencia del aplicativo se debe mejorar los algoritmos de funcionalidad, lo cual se puede realizar en versiones posteriores del aplicativo.

### **Módulo Histórico**

En este módulo se tuvo una carga máxima soportada de 1000000 de transacciones, con lo cual se determina que este módulo tiene un alcance mayor que el módulo de reportes, esto se debe a que este módulo tiene un algoritmo más simple pero a su vez la funcionalidad de este módulo es inferior al módulo de reportes. A partir de esto determinamos que la carga que soportada en este módulo es amplia.

Para la evaluación de las pruebas de resistencia se toma como un criterio de prueba a la carga que se realizó al sistema. Tabla 3.8.

<b>Tabla 3.8 Valoración de Resistencia</b>		
<b>Aspecto</b>	<b>Módulo Reporte</b>	<b>Módulo Histórico</b>
<b>Carga</b>	Bueno	Excelente
Elaborado por los Autores		

Como ya se analizó el módulo de Reporte soporta menos carga que el módulo Históricos entonces como ya está establecido de acuerdo a la valoración el módulo Reporte es aceptable y el módulo Histórico es óptimo para la evaluación en la resistencia del sistema.

#### **3.4.4 ANALISIS DE RENDIMIENTO**

Para este análisis se realizó un conjunto de pruebas definidas en la sección **pruebas de rendimiento** a partir de las cuales se pude determinar el tiempo de

respuesta por número de transacciones realizadas. En este análisis se utilizará los módulos de Reporte e Históricos.

### Modulo Reporte

Para el análisis de resultados se utiliza la siguiente tabla en la cual se puede visualizar el tiempo de respuesta por n transacciones. Ver Tabla 3.9.

**Tabla 3.9 Pruebas de rendimiento de reportes**

<b>Modulo (Reportes)</b>	
<b>Transacciones</b>	<b>Tiempo (Segundos)</b>
100	3
200	4
300	5
1000	5
2000	5
3000	5
10000	10
20000	38
30000	40
100000	No Funciona
200000	No Funciona
300000	No Funciona
1000000	No Funciona



Elaborado por los autores

A partir de la tabla anterior se determina que mientras mayor número de transacciones son procesadas por el modulo, su tiempo de respuesta aumenta llegando a un límite de 30000 con 40 segundos a partir del cual el modulo deja de funcionar. Con los datos ilustrados en la tabla se puede ver que el módulo de reporte tiene un tiempo de respuesta aceptable tomando en cuenta que el aplicativo es para el uso de empresas de pequeño tamaño, caso de estudio de una intranet.

## Módulo Histórico

Para el análisis de los resultados utilizaremos la siguiente tabla en la cual se puede visualizar el tiempo de respuesta por n transacciones. Ver Tabla 3.10.

**Tabla 3.10 Pruebas de rendimiento de reportes**

<b>Modulo (Reportes)</b>	
<b>Transacciones</b>	<b>Tiempo (Segundos)</b>
<b>100</b>	<b>1</b>
<b>200</b>	<b>1</b>
<b>300</b>	<b>1</b>
<b>1000</b>	<b>1</b>
<b>2000</b>	<b>1</b>
<b>3000</b>	<b>1</b>
<b>10000</b>	<b>1</b>
<b>20000</b>	<b>1</b>
<b>30000</b>	<b>1</b>
<b>100000</b>	<b>1</b>
<b>200000</b>	<b>4</b>
<b>300000</b>	<b>5</b>
<b>1000000</b>	<b>5</b>



Elaborado por los autores

A partir del gráfico y la tabla determinamos que mientras mayor número de transacciones son procesadas por el módulo su tiempo de respuesta aumenta. Con los datos ilustrados en la tabla se puede ver que el módulo de históricos tiene un tiempo de respuesta óptimo tomando en cuenta que el módulo tiene un alcance mucho mayor al de 5 segundos con un 1000000 de transacciones utilizadas en la prueba.

Dado la valoración para el rendimiento y aprobación del mismo por parte del cliente en cuanto al sistema de monitoreo se determinó los siguientes valores tanto para el módulo de Reportes como para el módulo Históricos antes descritos.

**Tabla 3.11 Valoración de Rendimiento**

Aspecto	Módulo Reporte	Módulo Histórico
Carga	Bueno	Excelente
Tiempo de Respuesta	Bueno	Excelente

Elaborado por los Autores

El sistema es evaluado tanto en la carga como en tiempo de respuesta en el cual siendo Excelente un tiempo de respuesta muy pequeño. Para el análisis de estas valoraciones se puede observar que tiene 50% de alcance óptimo y 50% de estado aceptable para el rendimiento y carga es decir es que el sistema tiene un nivel de resistencia del 90% a la carga y resistencia aplicada. Figura 3.46.

**Figura 3.46 Ponderación de Resistencia a la Carga**

Elaborado por los Autores

### 3.4.5 ANALISIS DE FUNCIONALIDAD

Conforme a los resultados de las prueba de simulación se procede a realizar la siguiente tabla, en esta se cataloga los resultados de las simulaciones en Excelente, Bueno y Malo. Ver Tabla 3.12.

**Tabla 3.12 Análisis de funcionalidad**

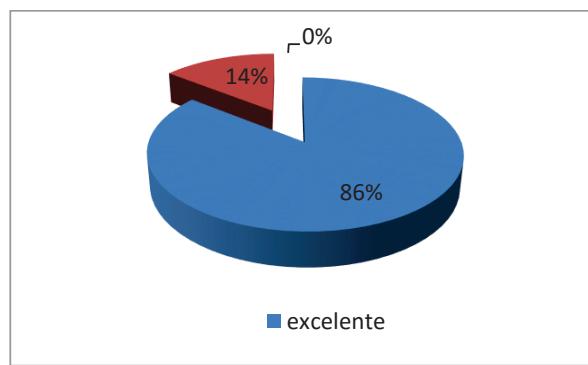
Tareas	Excelente	Bueno	Regular	Malo
Registrar nuevo usuario	X			
Ingresar usuario	X			
Registrar ubicación	X			

Registrar sub-ubicación	X
Edición ubicaciones	X
Eliminar ubicaciones	X
Registrar configuración	X
Edición de configuración	X
Eliminación de configuración	X
Generación de gráficas	X
Generación de gráficas en tiempo real	X
Generación de reportes globales	X
Generación de reportes por cliente	X
Generación de reportes por tiempo real	X

**Elaborado por los autores**

De un total de 14 tareas, 12 estuvieron excelentes y 2 tareas tuvieron el resultado de bueno, realizando un ponderado. Ver Figura 3.47.

**Figura 3.47 Gráfico porcentual funcionalidad**



**Elaborado por los Autores**

Como se aprecia en el gráfico el 86% de las funcionalidades estuvieron excelentes, mientras que el 14% funcionaron bueno y un 0% malo y regular. El 14%

se debe a que la funcionalidad "generación de gráficas" genera n-gráficas por n-aplicativos pero si una gráfica no cuenta con valores, esta se presenta en blanco. Y en la funcionalidad "generación de reportes globales" no se tiene un gráfico que contenga todos los clientes para su apreciación visual.

Debido a los criterios de valoración con su escala de porcentajes se tiene que el sistema tiene una Funcionalidad del 93,57%.

#### **3.4.6 ANALISIS DE USABILIDAD**

Se ha definido la usabilidad del proyecto con parámetros de como son: aceptación, tiempo de ejecución, facilidad de uso, errores y visualización. Para el tiempo de ejecución se toma una escala de alta media y baja.

En la tabla se muestra la aceptación de los usuarios, definiendo los valores bueno, muy bueno y malo para los módulos del Sistema. Ver Tabla 3.13.

<b>Tabla 3.13 Análisis de usabilidad</b>	
<b>Módulo</b>	<b>Tiempo de ejecución</b>
Ubicación	Excelente
Configuración	Excelente
Transacciones	Bueno
Históricos	Bueno

Elaborado por los Autores

Los errores producidos en el módulo de transacciones, se dan en los gráficos esto ocurre cuando los intervalos de tiempo son muy extensos esto produce un prolongado tiempo espera para realizar la consulta en la base de datos, lo cual produce errores en el tiempo de ejecución.

La visualización es un concepto fundamental ya que esta atrae la atención de los usuarios, de esta forma influye directamente en la aceptación del sistema, los resultados del análisis efectuado acerca de este parámetro produjo resultados mostrados en la Tabla 3.14.

**Tabla 3.14 Resultados de aceptación**

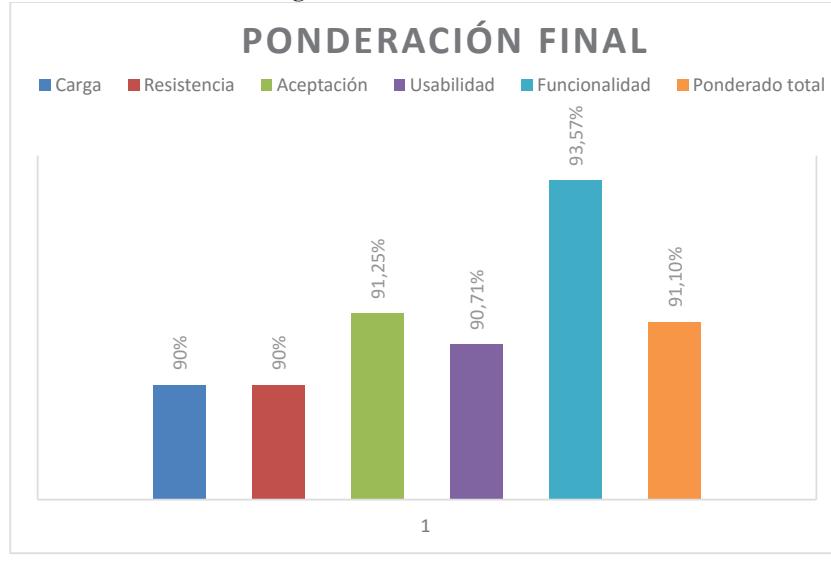
<b>Parámetro</b>	<b>Aceptación</b>
<b>Interfaces</b>	Excelente
<b>Facilidad de operación</b>	Excelente
<b>Gama de colores</b>	Buena

Elaborado por los Autores

De acuerdo a los criterios de valoración antes propuestos la usabilidad ponderada de estos módulos es del 90,71% en total.

### 3.4.7 RESUMEN DEL ANÁLISIS DE EVALUACIÓN

Teniendo los resultados para cada una de las métricas de pruebas para la calidad de la aplicación Lizard, se procede a la presentación de la ponderación final de las métricas globales los cuales en total tiene un resultado óptimo para su criterio de calidad. Como se puede ver en la Figura 3.48 se presenta ponderados finales del análisis de las pruebas del sistema.

**Figura 3.48 Ponderación final**

Elaborado por los Autores

Como se puede observar el ponderado final supero el 90% es decir el sistema tiene una calidad de presentación y funcionalidad óptima para el cliente.

## CAPÍTULO 4

### 4 CONCLUSIONES Y RECOMENDACIONES

Al finalizar el proyecto de titulación se puede indicar que el objetivo general del proyecto que era “Desarrollar un Sistema Web para el monitoreo de transacciones realizadas por otros sistemas web en un entorno de intranet” se ha cumplido en su totalidad con la entrega de un producto de software que se pone a disposición de la comunidad ecuatoriana y de la experiencia de desarrollo se puede extraer las siguientes conclusiones y recomendaciones.

#### 4.1 CONCLUSIONES

1. Basándose las entrevistas realizadas hacia los involucrados en el requerimiento del actual proyecto, se puede concluir que una intranet corporativa es exitosa si dispone de tres factores fundamentales como son: la infraestructura técnica, el contenido y su manejo. Estos factores son primordiales para que las aplicaciones que hacen uso de esta intranet tengan un rendimiento adecuado para el manejo de información teniendo como resultado transacciones exitosas.
2. Mediante las entrevistas realizadas a jefes de diferentes áreas de la DMI se concluye que esta carece de una herramienta que les permita monitorear sus aplicaciones web, la falta de esta herramienta produce diversos problemas con respecto al rendimiento laboral.
3. El uso de la metodología Scrum para el análisis, diseño y desarrollo del sistema, permitió obtener un producto de software que facilita la monitorización dentro de la DMI; se lo concluyó dentro de los plazos previstos y cubriendo todas las necesidades que exige el Distrito Metropolitano de Informática

4. El sistema web para el monitoreo de aplicaciones en una intranet puede agrupar las transacciones realizadas por zona y aplicación, facilitando al administrador realizar búsquedas personalizadas de la actividad de cada sistema cliente. Teniendo como resultado gráficos estadísticos y reportes puntuales de la actividad detallada de cada aplicación.
5. Basados en las pruebas realizadas con el algoritmo de generación de transacciones, se concluye que el módulo de reportes que genera gráficas por intervalo y en tiempo real soporta una carga de 30000 transacciones dentro de un intervalo de cada 5 minutos, por lo cual pasado este límite es innecesario la continuación de pruebas de rendimiento.
6. En el desarrollo del proyecto se logró seguir con las estrategias establecidas en la planificación existente en cada sprint, cumpliendo las tareas en el tiempo establecido y los retrasos generados no fueron significativos para la presentación de cada uno de los entregables.
7. Los resultados de las encuestas realizadas reflejan que el sistema tiene una aceptabilidad del 92.25% y un resultado de funcionalidad del 93.57%. Considerando al sistema altamente operacional en producción.

## 4.2 RECOMENDACIONES

1. En el caso de estudio, la intranet de la DMI cuenta con aplicaciones que generan miles de transacciones diarias que siendo mal gestionadas pueden poner en riesgo el normal funcionamiento de la misma, afectando directamente el rendimiento laboral.

2. La aplicación web de monitoreo es un prototipo por lo que se recomienda al momento al momento de implantar la aplicación en un ambiente de producción se utilice una versión estable.
3. El aplicativo registra transacciones a través de un método de captura, se recomienda instanciar el método en procesos críticos de los aplicativos monitoreados, los cuales determinen funciones específicas que requieran ser analizadas. Además para implantar este método en varios clientes es recomendable instanciarlo en eventos concurrentes.
4. Ya que Distrito Metropolitano de Informática maneja sus usuarios a través del directorio activo es recomendable enlazar el acceso al sistema por medio de este sistema, aumentando la seguridad y transparencia del aplicativo.
5. Con los resultados obtenidos en las encuestas realizadas se recomienda que el sistema pase a producción esperando la aprobación del jefe del área de soluciones del Distrito Metropolitano de Informática.

## 5 BIBLIOGRAFÍA

- [1] M. Naranjo y P. Ituralde, Interviewees, *Análisis de situación de rendimiento de aplicaciones DMI*. [Entrevista]. 15 Marzo 2015.
- [2] P. Juan, in Flexibilidad con Scrum, 2008.
- [3] SCRUMstudy™, A Guide to the SCRUM BODY OF KNOWLEDGE, New York, 2013.
- [4] K. Schwaber, Agile Project Management with Scrum, O'Reilly Media, Inc, 2009.
- [5] H. Kniberg, «Scrum y XP desde las Trincheras,» 2007. [En línea]. Available: <http://goo.gl/SiF56t>. [Último acceso: 2015].
- [6] À. R. G. ( . J. C. V. y. D. G. J. Jordi Conesa Caralt, Introducción a .NET, II ed., Barcelona: UOC, 2010, pp. 13-16.
- [7] L. Koskela, Application of the new production philosophy to construction, Finland: VTT Building Technology, 1992, p. 13.
- [8] F. Ramírez, «Aprenda Practicando ASP.NET Usando Visual Studio 2012,» de *Aprenda Practicando ASP.NET Usando Visual Studio 2012*, Mexico, Alfaomega, 2013, pp. 8-14, 213-215.
- [9] F. J. C. J. C. Fernando Berzal, Desarrollo Profesional de Aplicaciones Web con ASP.Net, Primera ed., 2005.
- [10] F. J. Ceballos Sierra, Enciclopedia de Microsoft Visual C#: interfaces gráficas y aplicaciones para Internet con Windows Forms y ASP.NET, Cuarta ed., RA-MA Editorial, 2012.
- [11] Microsoft, «msdn.microsoft.com,» [En línea]. Available: <https://msdn.microsoft.com/es-es/library/z1zx9t92.aspx>. [Último acceso: 26 Mayo 2015].
- [12] W. Penberthy, Developing ASP.NET MVC 4 Web Applications, Microsoft Press, 2013.
- [13] Microsoft, [En línea]. Available: [https://msdn.microsoft.com/es-es/library/dd381412\(v=vs.100\).aspx](https://msdn.microsoft.com/es-es/library/dd381412(v=vs.100).aspx).

- [14] S. Sanderson, Pro ASP.Net MVC 2 Framework, Segunda ed., 2010.
- [15] M. M. J. I. A. M. J. A. G. Quintana, «APRENDE SQL,» de *APRENDE SQL* , Universitat Jaume I, 2010, pp. 8-11.
- [16] A. Hofacker, Rapid lean construction - quality rating model, Manchester: s.n., 2008.
- [17] A. O. y. R. Sheldon, «Fundamentos de SQL,» de *Fundamentos de SQL*, Santa Fe, MC Graw Hill, 2010, pp. 4-13.
- [18] M. Otey, «Innovaciones en Microsoft SQL Server 2008,» de *Innovaciones en Microsoft SQL Server 2008*, Mexico, Mc Graw Hill, 2010, pp. 3-60.
- [19] Microsoft, 2011. [En línea]. Available: [https://msdn.microsoft.com/es-es/library/aa287558\(v=vs.71\).aspx](https://msdn.microsoft.com/es-es/library/aa287558(v=vs.71).aspx).

## 6 GLOSARIO DE TÉRMINOS

**DMQ:** Distrito Metropolitano de Quito

**DMI:** Distrito Metropolitano de Informática

**SQL Server:** Es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional.

**Visual Studio 2010:** Entorno de desarrollo integrado para sistemas Windows, soporta múltiples lenguajes de programación como C++, Visual C#, Visual J# y Visual Basic .Net.

**ASP.NET:** Es un framework para aplicaciones web comercializado y desarrollado por Microsoft. Usado por programadores para la construcción de sitios web dinámicos, aplicaciones web y servicios webXML.

**Framework:** Estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, con base a la cual otro proyecto de software puede ser más fácilmente organizado y desarrollado. Incluye soporte de programas, bibliotecas y un lenguaje interpretado, entre otras herramientas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

**IIS:** Servidor Web que ofrece una infraestructura de gran fiabilidad, capacidad de manejo y escalabilidad para aplicaciones Web sobre todas las versiones de Windows Server 2003.

**DLL:** Es un componente creada en Visual Basic, formada por una o varias clases las cuales contienen un conjunto de métodos que son los encargados de realizar las distintas funcionalidades que posee la DLL.

**Scripts:** Lista de comandos que pueden ser ejecutados sin la intervención de un usuario.

**Transacción:** Interacción con una estructura de datos, está compuesta por varios procesos que se han de aplicar uno después del otro.

## 7 ANEXOS

Los siguientes anexos que se muestran a continuación se encuentran en el disco adjunto:

**ANEXO 1 – CODIFICACION.**

**ANEXO 2 – MANUAL DE USUARIO.**