



CARRERA DE ANÁLISIS DE SISTEMAS

**AUTOMATIZACIÓN DEL PROCESO DE EJECUCIÓN DE PRUEBAS
AUTOMÁTICAS UNITARIAS DE APLICATIVOS PARA LA EMPRESA
BIOLOGICSOFT UBICADA EN LA CIUDAD DE QUITO.**

**Trabajo de Titulación previo la obtención del título de Tecnólogo en
Análisis de Sistemas**

AUTOR: Caguano Millingalle Lucía Janeth

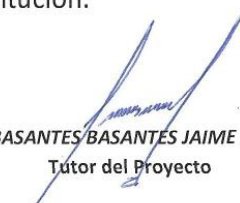
DIRECTOR: Ing. Basantes Basantes Jaime Neptalí

Quito, 2018

ACTA DE APROBACIÓN DEL PROYECTO DE GRADO

Quito, 14 de Mayo de 2018.

El equipo asesor del Trabajo de Titulación del Sr. (Srta.) (Sra.) **CAGUANO MILLINGALLE LUCÍA JANETH** de la Carrera de Análisis de Sistemas cuyo tema de investigación fue: **"AUTOMATIZACIÓN DEL PROCESO DE EJECUCIÓN DE PRUEBAS AUTOMÁTICAS UNITARIAS DE APLICATIVOS PARA LA EMPRESA BIOLOGICSOFT UBICADA EN LA CIUDAD DE QUITO."** una vez considerados los objetivos del estudio, coherencia entre los temas y metodologías desarrolladas; adecuación de la redacción, sintaxis, ortografía y puntuación con las normas vigentes sobre la presentación del escrito, resuelve: **APROBAR** el proyecto de grado, certificando que cumple con todos los requisitos exigidos por la Institución.


BASANTES/BASANTES JAIME NEPTALÍ
Tutor del Proyecto
GARZÓN JACOME ELSA PATRICIA
Lector del Proyecto
HEREDIA MAYORGA HUGO PATRICIO.
Director de Carrera
CISNEROS VITERI GALO FERNANDO
Coordinador Unidad de Titulación**CAMPUS 1 - MATRIZ**
Av. de la Prensa N45-268 y Logroño
Teléfono: 2255460 / 2269900
E-mail: instituto@cordillera.edu.ec
Pág. Web: www.cordillera.edu.ec
Quito - Ecuador**CAMPUS 2 - LOGROÑO**
Calle Logroño Oe 2-84 y
Av. de la Prensa (esq.)
Edif. Cordillera
Telfs.: 2430443 / Fax: 2433649**CAMPUS 3 - BRACAMOROS**
Bracamoros N15 - 163
y Yacuambí (esq.)
Telf.: 2262041**CAMPUS 4 - BRASIL**
Av. Brasil N46-45 y
Zamora
Telf.: 2246036**CAMPUS 5 - YACUAMBÍ**
Yacuambí
Oe2-36 y
Bracamoros.
Telf: 2249994

DECLARATORIA

Yo, Lucía Janeth Caguano Millingalle, declaro bajo juramento que la investigación es absolutamente original, autentica, es de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes. Las ideas, doctrinas, resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad



Lucia Janeth Caguano Millingalle

C.C: 1720385374

LICENCIA DE USO NO COMERCIAL

Yo, Lucía Janeth Caguano Millingalle portador de la cédula de ciudadanía signada con el No. 172038537-4 de conformidad con lo establecido en el Artículo 110 del Código de Economía Social de los Conocimientos, la Creatividad y la Innovación (INGENIOS) que dice: “En el caso de las obras creadas en centros educativos, universidades, escuelas politécnicas, institutos superiores técnicos, tecnológicos, pedagógicos, de artes y los conservatorios superiores, e institutos públicos de investigación como resultado de su actividad académica o de investigación tales como trabajos de titulación, proyectos de investigación o innovación, artículos académicos, u otros análogos, sin perjuicio de que pueda existir relación de dependencia, la titularidad de los derechos patrimoniales corresponderá a los autores. Sin embargo, el establecimiento tendrá una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra con fines académicos. Sin perjuicio de los derechos reconocidos en el párrafo precedente, el establecimiento podrá realizar un uso comercial de la obra previa autorización a los titulares y notificación a los autores en caso de que se traten de distintas personas. En cuyo caso corresponderá a los autores un porcentaje no inferior al cuarenta por ciento de los beneficios económicos resultantes de esta explotación. El mismo beneficio se aplicará a los autores que hayan transferido sus derechos a instituciones de educación superior o centros educativos.”, otorgo licencia gratuita, intransferible y no exclusiva para el uso no comercial del proyecto denominado **AUTOMATIZACIÓN DEL PROCESO DE EJECUCIÓN DE PRUEBAS AUTOMÁTICAS UNITARIAS DE APLICATIVOS PARA LA EMPRESA BIOLOGICSOFT UBICADA EN LA CIUDAD DE QUITO.**

Con fines académicos al Instituto Tecnológico Superior Cordillera.



Lucia Janeth Caguano Millingalle

C.C: 172038537-4

Quito, 14/06/2018

AGRADECIMIENTO

Quiero agradecer a mis padres y mi familia por todo el apoyo incondicional que me han brindado en este proceso de crecimiento profesional.

Al Instituto Tecnológico Superior “Cordillera” por haberme permitido estudiar y formarme profesionalmente.

También quiero agradecer a todas las personas que conforman la empresa Biologicsoft quienes me han motivado y apoyado durante la elaboración de este trabajo.

DEDICATORIA

Celio Caguano y María Millingalle mis padres quienes me han motivado a ser una mejor persona a luchar por mis objetivos y cumplir mis metas.

A Susana y Zulay mis hermanas, Joel y Max mis sobrinos por su apoyo incondicional para la conclusión de este ciclo.

A todos quienes han estado apoyándome a lo largo de este proceso y han sido parte primordial para la culminación de mi carrera Sebastián Aldás, Ing. Fabricio Tamayo y todas las personas de la empresa Biologicsoft

A French y Estrella quienes han estado acompañándome en mis desveladas y han sido parte importante en mi vida.

ÍNDICE GENERAL

DECLARATORIA.....	ii
LICENCIA DE USO NO COMERCIAL	iii
AGRADECIMIENTO	v
DEDICATORIA	vi
RESUMEN EJECUTIVO	xiv
ABSTRACT.....	xv
INTRODUCCIÓN	xvi
CAPÍTULO I.....	1
1. Antecedentes.....	1
1.01 Contexto	1
1.02 Justificación	2
1.03 Definición del Problema Central	2
CAPÍTULO II	6
2 Análisis de Involucrados	6
2.01 Requerimientos	6
2.01.01 Descripción del Sistema Actual	6
2.01.02 Visión y Alcance	6
2.01.03 Entrevistas	8
2.01.04 Matriz de Requerimientos	10
2.01.05 Descripción Detallada	12
2.02 Mapeo de Involucrados	22
2.03 Matriz de Involucrados	23
CAPÍTULO III.....	24
3 Problemas y Objetivos	24
3.01 Árbol de Problemas	24
3.02 Árbol de objetivos	25
3.03 Diagramas de Caso de Uso	26
3.04 Especificación de Casos de Uso	31
3.05 Casos de Uso de Realización y Especificación	35
3.06 Diagramas de Secuencia del Sistema	43
CAPÍTULO IV.....	52
4 Análisis de Alternativas	52
4.01 Matriz de Análisis de Alternativas	52
4.02 Matriz de Impactos de Objetivos	53

4.03 Estándares para el Diseño de Clases	54
4.04 Diagrama de Clases	55
4.05 Modelo Lógico	56
4.06 Modelo Físico de la Base de Datos	57
4.07 Diagrama de Componentes	58
4.08 Diagrama de Estrategias	58
4.09 Matriz de Marco Lógico	59
4.10 Vistas Arquitectónicas	60
4.01.01 Vista Lógica	60
4.10.02 Vista Física	60
4.10.03 Vista de Desarrollo	61
4.10.04 Vista de Procesos	62
CAPÍTULO V	66
5 Propuesta	66
5.01 Especificación de Estándares de Programación	66
5.04 Pruebas de Aceptación	80
5.05 Especificación de pruebas de carga	87
5.06 Configuración del Ambiente Mínima Ideal	88
CAPÍTULO VI	89
6 Aspectos Administrativos	89
6.01 Recursos	89
6.02 Presupuesto	90
6.03 Cronograma	91
CAPÍTULO VII	92
7 Conclusiones y Recomendaciones	92
7.01 Conclusiones	92
7.02 Recomendaciones	92
7.03 Referencias Bibliográficas	93
Anexos	94
Anexo 1 Manual de Usuario	
Anexo 2 Manual Técnico	
Anexo 3 Manual de Instalación	

ÍNDICE DE TABLAS

Tabla 1 Matriz de Fuerzas T	3
Tabla 2 Matriz de Preguntas	9
Tabla 3 Requerimientos Funcionales	10
Tabla 4 Requerimientos No Funcionales	11
Tabla 5 Requerimiento Funcional 001 (RF001)	12
Tabla 6 Requerimiento Funcional 002 (RF002)	13
Tabla 7 Requerimiento Funcional 003 (RF003)	14
Tabla 8 Requerimiento Funcional 004 (RF004)	15
Tabla 9 Requerimiento Funcional 005 (RF005)	16
Tabla 10 Requerimiento Funcional 006 (RF006)	17
Tabla 11 Requerimiento No Funcional 001 (RF001)	18
Tabla 12 Requerimiento No Funcional 002 (RF002)	20
Tabla 13 Requerimiento No Funcional 003 (RF003)	21
Tabla 14 Matriz de involucrados	23
Tabla 15 Caso de uso UC001	31
Tabla 16 Caso de uso UC002	32
Tabla 17 Caso de uso UC003	32
Tabla 18 Caso de uso UC004	33
Tabla 19 Caso de uso UC005	33
Tabla 20 Caso de uso UC006	34
Tabla 21 Caso de uso UC007	34
Tabla 22 Especificación de caso de realización UCR001	36
Tabla 23 Especificación de caso de realización UCR002	37
Tabla 24 Especificación de caso de realización UCR003	38
Tabla 25 Especificación de caso de realización UCR004	39
Tabla 26 Especificación de caso de realización UCR005	40
Tabla 27 Especificación de caso de realización UCR006	41
Tabla 28 Especificación de caso de realización UCR007	43
Tabla 29 Matriz de Análisis de Alternativas	52
Tabla 30 Matriz de Impactos de Objetivos	53
Tabla 31 Matriz de Marco Lógico	59
Tabla 32 Estándares de programación	66
Tabla 33 Ingreso al Sistema	76
Tabla 34 Registro de usuarios	76
Tabla 35 Validación de la cédula	77
Tabla 36 Redundancia de Datos	77
Tabla 37 Validación de campos numéricos y de letras	78
Tabla 38 Validación de filtros de búsqueda	78
Tabla 39 Validación de botones	79
Tabla 40 Prueba de Aceptación de registro de usuarios	80
Tabla 41 Prueba de Aceptación de asignación de permisos	81
Tabla 42 Prueba de Aceptación de creación de ventanas	82
Tabla 43 Prueba de Aceptación de creación de pasos a seguir	83
Tabla 44 Prueba de Aceptación de ejecución de pruebas (ejecutar ventana)	84

Tabla 45 Prueba de Aceptación de seguir pasos de ejecución	85
Tabla 46 Prueba de Aceptación de generar reportes.....	86
Tabla 47 Pruebas de Carga	87
Tabla 48 Ambiente Mínimo Ideal.....	88
Tabla 49 Recursos Utilizados	89
Tabla 50 Tabla de Presupuesto	90

ÍNDICE DE FIGURAS

Figura 1: Mapeo de Involucrados. Esta figura muestra los involucrados directa e indirectamente en el sistema	22
Figura 2: Esta figura corresponde al árbol de problemas, donde se muestra el problema central con sus causas y efectos.	24
Figura 3: Esta figura corresponde al Árbol de objetivos de la empresa con respecto al sistema en desarrollo.....	25
Figura 4: Caso de uso general.....	26
Figura 5: Caso de Uso entrega de datos y registro de usuarios	27
Figura 6: Asignación de permisos	28
Figura 7: Caso de uso creación de ventanas	28
Figura 8: Caso de uso crear pasos a seguir	29
Figura 9: Caso de uso ejecución de ventanas	29
Figura 10: Caso de uso seguir pasos de ejecución.....	30
Figura 11: Resultado de la Ejecución	30
Figura 12: Caso de uso generar reportes.....	31
Figura 13: UCR001 Caso de uso de realización de registro de usuario.....	35
Figura 14: UCR002 Caso de uso de realización asignar permisos	36
Figura 15: UCR003 Caso de uso de realización creación ventana	37
Figura 16: UCR004 Caso de uso de creación de pasos a seguir.....	38
Figura 17: UCR005 Caso de uso de realización de ejecución de ventanas	39
Figura 18: UCR006 Caso de uso de seguir pasos de ejecución.....	40
Figura 19: UCR007 Caso de Realización Resultados de Ejecución.....	41
Figura 20: UCR008 Caso de realización generar reportes	42
Figura 21: Diagrama de Secuencia Registro de Usuarios.....	43
Figura 22: Diagrama de Secuencia Asignación de Permisos	44
Figura 23: Diagrama de Secuencia Crear Ventanas	45
Figura 24: Diagrama de Secuencia Pasos a Seguir.....	46
Figura 25: Diagrama de Secuencia Ejecutar Ventanas	47
Figura 26: Diagrama de Secuencia Pasos a Seguir.....	48
Figura 27: Diagrama de Colaboración Registro de Usuarios	49
Figura 28: Diagrama de Colaboración Asignación de Permisos	49
Figura 29: Diagrama de Colaboración Creación de Ventanas.....	50
Figura 30: Diagrama de Colaboración Creación de Pasos a Seguir	50
Figura 31: Diagrama de Colaboración Ejecutar Ventanas.....	51
Figura 32: Diagrama de Colaboración Seguir pasos de ejecución	51
Figura 33: Diagrama de Clases.....	55
Figura 34: Modelo lógico	56
Figura 35: Diagrama del Modelo Físico de la Base de Datos	57
Figura 36: Diagrama de Componentes	58
Figura 37: Diagrama de Estrategias.....	58
Figura 38: Vista lógica de la solución	60
Figura 39: Vista física de la solución	60
Figura 40: En este gráfico se pueden visualizar las capas con cada uno de sus componentes que conforman el sistema.....	61
Figura 41: En este gráfico se puede visualizar el proceso que debe seguir un usuario para ingresar el sistema.....	62

Figura 42: En este gráfico se pueden visualizar el proceso que sigue el Administrador para registrar los usuarios	63
Figura 43: En este gráfico se pueden visualizar el proceso que sigue el diseñador para crear las páginas y posteriormente diseñarlas	64
Figura 44: En este gráfico se pueden visualizar el proceso que sigue el usuario Tester para la ejecución de pruebas unitarias	65
Figura 45: En este gráfico se pueden visualizar el login, en el cual se debe ingresar las credenciales de usuario para ingresar al sistema, si las credenciales están correctas permite el ingreso caso contrario le saldrá un mensaje indicando que la información no	67
Figura 46: En este gráfico se pueden visualizar la interfaz del usuario Administrador.....	68
Figura 47: En este gráfico se pueden visualizar la interfaz del Administrador, donde crea los usuarios	69
Figura 48: En este gráfico se pueden visualizar la interfaz del Diseñador	70
Figura 49: En este gráfico se pueden visualizar la interfaz del Diseñador donde agrega el lenguaje, versión, aplicación y entidades.....	71
Figura 50: En este gráfico se pueden visualizar la interfaz del Diseñador donde crea las ventanas	72
Figura 51: En este gráfico se pueden visualizar la interfaz del Tester.....	73
Figura 52: En esta figura se puede observar la ventana de envío y generación de un reporte	74
Figura 53: Esta figura muestra el reporte en formato PDF	75
Figura 54: En este gráfico se puede visualizar el cronograma de actividades.....	91
Figura 55: Login	3
Figura 56: Interfaz Administrador	4
Figura 57: Diseñador - Lenguaje	5
Figura 58: Diseñador - Versión	6
Figura 59: Diseñador - Aplicación	7
Figura 60: Diseñador - Entidades	8
Figura 61: Tester - Ejecución de Pruebas	9
Figura 62: Tester - Probar Sistema	10
Figura 63: Tester – Resultados	11
Figura 64: Tester – Reportes.....	12
Figura 65: Tester - Envío Reportes.....	13
Figura 66: Descarga de Visual Studio	3
Figura 67: Selecciona Cargas de Trabajo	3
Figura 68: Seleccionar componentes individuales.....	4
Figura 69: Seleccionar paquetes de idioma	4
Figura 70: Instalación de productos.....	5
Figura 71: Instalación correcta - Iniciar Sesión.....	5
Figura 72: Configurar visual – Colores	6
Figura 73: Visual ya casi listo	6
Figura 74: Listo para iniciar	7
Figura 75: Instalador de SQL Server	7
Figura 76: Instalando SQL Server	7
Figura 77: Selecciona idioma y paquete de descarga	8
Figura 78: Descarga de Archivos para instalación	8
Figura 79: Descarga completa de SQL Server.....	9
Figura 80: Instalador de SSMS.....	9
Figura 81: Instalar o cerrar SSMS	9

Figura 82: Instalando SSMS	10
Figura 83: Instalando paquetes - progreso general	10
Figura 84: Configuración Completa	11
Figura 85: En este gráfico se encuentra la interfaz del SQL Server 2017	11

RESUMEN EJECUTIVO

El presente proyecto tiene como objetivo agilizar el proceso de pruebas unitarias automáticas con la finalidad de dar apoyo en las valoraciones de los productos desarrollados y así cumplir con los tiempos establecidos al inicio de cada Sprint.

El sistema probará cada una de las ventanas para ello el usuario Administrador debe crear los usuario asignándoles los perfiles respectivos, el usuario diseñador será quien se encarga de la creación de las ventanas que van a ser sometidas a la evaluación, una vez que las entidades estén levantadas y parametrizadas con la información, el usuario tester será el responsable de la ejecución y la posterior emisión de los reportes en formato PDF, los mismos que indicarán los resultados de las ventanas que pasaron y las que deben ser sometidas a correcciones.

También el tester debe enviar un correo a los involucrados en el proyecto que son el Administrador, Scrum Master, Desarrolladores, para la creación de historias y fijar los objetivos del nuevo periodo, esto permitirá mejorar los estándares de calidad, rapidez, así mismo, su nivel de competitividad tecnológica.

El proyecto se pone a disposición de la empresa Biologicsoft a los departamentos de Aseguramiento y Control de Calidad.

ABSTRACT

This project aims to streamline the process of automatic unit tests in order to provide support in the valuations of the developed products and thus meet the times established at the start of each Sprint.

The system will try each one of the windows for it the user Administrator must be created the users assigning them the respective profiles, the user designer will be the one in charge of the creation of the windows that are going to be subjected to for evaluation, once the entities are raised and parameterized with the information, the user tester will be responsible for the execution and the subsequent issuance of the reports in PDF format, the same ones that will indicate the results of the windows that passed and those that should be submitted to corrections.

The tester must also send an email and involved in the project who is the Administrator, Scrum Master, Developers, for the creation of stories and set the goal of the new period, this will improve the standards of quality, speed, likewise, its level of technological competitiveness.

The job is made available to the company Biologicsoft to the departments of Assurance and Quality Control.

INTRODUCCIÓN

El presente documento tiene como finalidad dar a conocer a los usuarios la estructura y funcionamiento del sistema que se está desarrollando para la ejecución de pruebas unitarias beneficiando a la empresa Biologicsoft, con el fin de agilizar el proceso de evaluaciones de los productos, ya que actualmente se los viene realizando manualmente lo que conllevar una demora y las pruebas no son repetibles. Con el propósito de elaborar este software se ha usado la información levantada al principio del trabajo, se lo va a realizar en el lenguaje de programación C# y el motor de base de datos SQL Server Management. Este programa tendrá la funcionalidad de realizar pruebas de caja negra, validando campos numéricos, alfabéticos, documentos de identificación, filtros de búsqueda y funcionalidad de botones, una vez que finalice el proceso de valoración emitirá un reporte que revele la funcionalidad de las ventanas que se han sometido al análisis, finalmente el usuario Tester emitirá reportes.

CAPÍTULO I

1. Antecedentes

1.01 Contexto

La empresa Biologicsoft ubicada en la ciudad de Quito en la Avenida 12 de Octubre N211-55 y Ramón Roca, Edificio Mariana de Jesús misma que se dedica a la prestación de servicios de desarrollo de software.

Biologicsoft es una compañía ecuatoriana con 11 años en el mercado local desarrollando soluciones tecnológicas online en las áreas de móviles y tecnologías. Certificada por la operadora celular CLARO-CONECCEL a nivel nacional, ofrece una serie de productos y servicios puntuales, esquemas simples de gran utilidad para las organizaciones

El área de Investigación y Desarrollo ha construido múltiples soluciones que nos permiten ofertar al mercado toda una línea de aplicaciones modulares para equipos móviles.

Este formato de diseño –módulos– proporciona la capacidad de interactuar no solo con las herramientas y base de datos sobre las cuales fueron construidos. Uno de los beneficios que brindamos al mercado es la suficiencia de poder integrar los módulos de manera individual tanto a cualesquier aplicación, sin importar la herramienta en la que haya sido construida, cuanto también a cualesquier base de datos que el cliente posea -en caso de tenerla-, permitiéndole gestionar el eficiente manejo de sus recursos económicos. Biologicsoft se ve en la necesidad de obtener un sistema que realice las pruebas unitarias de los productos que se desarrollan en la empresa con el

fin de garantizar la calidad del producto y mejorar los tiempos de entrega de los mismos.

1.02 Justificación

Se requiere realizar un sistema para la ejecución de las pruebas unitarias de los productos desarrollados en la empresa Biologicsoft con el fin de agilizar los procesos de evaluación.

Los departamentos de aseguramiento y de control de calidad han venido realizando los procesos de valoración de forma manual, es decir, mediante archivos que se van creando conforme avanza el desarrollo.

Es por eso que el sistematizar el procedimiento de evaluaciones pretende disminuir los tiempos de entrega de los productos a los clientes.

El sistema tendrá un impacto tecnológico ya que se sistematizará y mejorará la productividad, mejorará el ambiente de trabajo y ayudará a la empresa a organizarse de mejor manera.

1.03 Definición del Problema Central

El proceso de pruebas unitarias que se realizan actualmente dentro de la empresa es de forma manual por lo que resultan ser demorosas, costosas y no son repetibles en el día ya que los tiempos de entrega son pequeños y se los establece al inicio de cada Sprint que dura tres semanas. Por cuanto al no cumplirse las metas señaladas la demora conlleva a un incumplimiento de objetivos dentro de la empresa y una insatisfacción por parte del cliente.

Tabla 1

*Matriz de Fuerzas T***MATRIZ DE FUERZAS T**

SITUACIÓN EMPEORADA	SITUACIÓN ACTUAL				SITUACIÓN MEJORADA
Pérdida de tiempo en la realización de los procesos de pruebas unitarias manuales	Demora en la realización de pruebas unitarias manuales				Sistema automatizado para la ejecución de pruebas unitarias
FUERZAS IMPULSADORAS	CALIFICACIÓN				FUERZAS BLOQUEADORAS
	I	PC	I	PC	
Ampliación de plazos para la entrega del trabajo	4	5	3	4	Inadecuada asignación de recursos económicos
Cumplimiento de objetivos dentro del sprint	4	5	3	4	Rechazo a aprender el manejo de un nuevo sistema
Actualización de versión para el desarrollo	3	4	3	4	Inadecuada comunicación entre los departamentos de Aseguramiento y control de la calidad
Pruebas repetibles en el día	4	4	4	4	Desconfianza de los resultados finales

NOTA: Determinación de las fuerzas bloqueadoras e impulsadoras.

PC: Potencial de Cambio

I: Intensidad

Nota: Análisis de fuerzas impulsadoras y bloqueadoras.

Análisis de Fuerzas Impulsadoras

FI 1: Ampliación de plazos para la entrega del trabajo

I=4: Tiene una escala valorativa de cinco ya que la empresa ha tenido que ampliar los plazos de entrega de los productos

C=5: Tiene una escala valorativa de cinco ya que se debe cumplir los plazos de entrega en los tiempos establecidos

FI 2: Cumplimiento de objetivos dentro del Sprint

I=4: Tiene una escala valorativa de cuatro, ya que se requiere cumplir con todos los objetivos planteados al inicio de cada Sprint.

PC=5: Tiene una escala valorativa de cinco, y que al cumplir con los objetivos los clientes estarán más satisfechos y con un producto de calidad

FI 3: Actualización de versión para el desarrollo

I=3: Tiene una escala valorativa de tres, ya que la actualización de versión no es una prioridad

PC=4: Tiene una escala valorativa de cuatro, ya que el actualizarse en las herramientas en positivo para la empresa

FI 4: Pruebas repetibles en el día

I=3: Tiene una escala valorativa de cuatro, y que es necesario repetir las pruebas en el día por cualquier cambio que se haya generado

PC=4: Tiene una escala valorativa de cuatro, y que realizar las pruebas sería la mejor manera de garantizar la calidad del sistema

Análisis de Fuerzas Bloqueadoras

FB 1: Inadecuada asignación de recursos económicos

I=3: Tiene una escala valorativa de tres ya que el presupuesto es importante y sin él no se puede recurrir a ningún proyecto

PC=4: Tiene una escala valorativa de cuatro, ya que los recursos económicos son la parte más importante para la ejecución de un proyecto

FB 2: Rechazo a aprender el manejo de un nuevo sistema

I=3: Tiene una escala valorativa de tres, ya que el personal está acostumbrado a manejar el proceso de manera manual

PC=4: Tiene una escala valorativa de cuatro, ya que se realizará una capacitación para el uso correcto del aplicativo

FB 3: Inadecuada comunicación entre los departamentos de Aseguramiento y control de la calidad

I=3: Tiene una escala valorativa de tres, ya que los departamentos no tienen una comunicación continua

PC=4: Tiene una escala valorativa de cuatro, ya que al implementar el sistema habrá un acercamiento de los departamentos para un mejor desenvolvimiento.

FB 4: Desconfianza de los resultados finales

I=4: Tiene una escala valorativa de cuatro, ya que se posee un miedo a que el sistema no cumpla lo requerido

PC=4: Tiene una escala valorativa de cuatro, ya que el usuario tendrá la posibilidad de realizar pruebas al sistema

CAPÍTULO II

2 Análisis de Involucrados

2.01 Requerimientos

2.01.01 Descripción del Sistema Actual

Actualmente la empresa realiza las pruebas unitarias de los procesos de forma manual. Inicia con la creación de carpetas con su respectiva codificación asignando las jerarquías de los módulos correspondientes, después de eso se revisa con que componentes cuenta cada ventana del sistema y se describe en tres campos, que son pasos a seguir, resultados esperados y el método de ejecución, una vez ingresada la información esta se guarda con la secuencia que se creó.

El usuario encargado de probar debe leer paso a paso el documento asignado e ir indicando si los campos del sistema realizan la ejecución ingresada y seleccionar si el campo pasó o no la prueba realizada.

Una vez finalizadas las ejecuciones se genera un informe detallado que será enviado al Scrum Master y al Administrador, para la creación de historias y corrección de incidencias.

2.01.02 Visión y Alcance

2.01.02.01 Visión

El presente proyecto será implementado en la empresa Biologicsoft como un módulo que se encargará de la realización de pruebas unitarias a los sistemas, de esta manera se agilizará el tiempo de evaluaciones a fin de entregar un producto con los estándares de calidad requeridos por los clientes, de igual

manera aportará al cumplimiento de objetivos planteados al inicio de cada Sprint.

2.01.02.02 Alcance

El sistema que se va a implementar en la empresa Biologicsoft es una herramienta tecnológica que permitirá realizar las pruebas unitarias a los sistemas desarrollados.

Se pone a consideración que el sistema no tendrá la capacidad de crear historias automáticamente. Se denomina como historia al trabajo que se le asigna a cada desarrollador con el fin de que se corrijan las incidencias que se muestre al finalizar la evaluación.

Módulo de Seguridad

Este módulo contiene el registro de los usuarios los que serán creados por el Administrador, el que les asignará un rol para cada función que va a cumplir cada uno. Posteriormente el software desplegará una ventana de ingreso o Login en donde el usuario deberá ingresar sus credenciales asignadas por Administrador (Usuario y Contraseña), los que serán autenticados, de estar correctos permitirá el ingreso a la interfaz correspondiente, caso contrario, se mostrará un mensaje indicando que ingrese las credenciales correctas para poder ingresar, y no permitirá el acceso al aplicativo.

Módulo de Mantenimiento

En este módulo se realizarán las acciones de creación, modificación, anulación, activación de un registro anulado y eliminación, se lo realizará a los Usuarios, Lenguajes, Versiones, Aplicaciones y Entidades.

Reglas del Negocio

Módulo de Administración

Este módulo será manipulado únicamente por el Usuario Administrador. En este módulo se podrá crear los usuarios así como asignar los roles y dar los permisos correspondientes de acuerdo a su perfil.

Módulo de Diseño

En este módulo será manipulado por el usuario Diseñador. En este módulo se van a crear las entidades o ventanas que se van a probar y también se va a diseñar cada una de ellas para su posterior ejecución, se las creará campo por campo indicando que si es una caja de texto, un combobox, ect.

Módulo de pruebas

Este módulo será manipulado por el usuario Tester. En este módulo se va a realizar la ejecución de las pruebas a las ventanas o entidades, para que se pueda realizar la evaluación las entidades deben estar creadas con anterioridad por el usuario.

Reportes

Los reportes se generarán una vez que se haya realizado la ejecución de las entidades mostrando los resultados de las evaluaciones que superaron las pruebas y las ventanas que tiene incidencias por corregir, de igual manera se enviará por correo a los involucrados en el desarrollo.

2.01.03 Entrevistas

En esta sección se presenta la descripción de cada una de las preguntas realizadas a las personas involucradas en la elaboración del sistema.

Tabla 2

Matriz de Preguntas

DISEÑO DE ENTREVISTAS		
Identificador: Ing. Fabricio Tamayo DIRECTOR DE TI		
Identificador: Departamentos QA & QC		
Preguntas	Objetivos	Análisis Posterior
¿Cuánto se demora en la realización de las pruebas actualmente?	Medir los tiempos y mejorar con la automatización del proceso	Lo que se busca es agilizar los tiempos para la ejecución de pruebas
¿Cuánto tiempo se demora en realizar un caso de prueba?	Fijar los procesos para la creación y el diseño de las ventanas	El caso de prueba se demora de acuerdo a la complejidad de las ventanas a probar
¿Las pruebas pueden ser repetidas en el día?	Que el sistema permita hacer pruebas repetibles en el día	Verificar cuantas pruebas se logran hacer en el día y si son repetibles
¿Cómo se realizan las pruebas actualmente?	Conocer el proceso que se realiza para la ejecución de las pruebas	Lo que se busca es conocer el proceso exacto de la ejecución para así poder realizar un sistema óptimo y de calidad
¿Qué se busca mejorar con el sistema en gestión?	Conocer cuál es el objetivo de la realización del sistema	Con este sistema se busca agilizar el proceso de pruebas automáticas
¿Se necesitará que el sistema genere reportes?	Generar reportes de las ejecuciones realizadas	El sistema deberá generar un reporte por cada ejecución que realice
¿El sistema será orientado a la WEB?	Saber para qué ambiente se desarrollará el sistema	El sistema no estará orientado a WEB será un sistema de escritorio
¿En qué lenguaje de programación se realizará el desarrollo?	Conocer en que lenguaje se va a realizar el desarrollo	Familiarizarse con el lenguaje de programación con el que se va a realizar el desarrollo
¿Qué motor de base de datos utilizará?	Tener claro a base de datos que se va a utilizar	Conocer el motor de base de datos que se va a desarrollar
NOTA: Desglosamiento de las preguntas realizadas a la persona entrevistada Ing. Fabricio Tamayo Elaborado por: Lucía Caguano		

2.01.04 Matriz de Requerimientos

Tabla 3

Requerimientos Funcionales

Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares.

MATRIZ DE REQUERIMIENTOS						
Identifi-cador	Descripción	Fuente	Priori- dad	Tipo	Esta- Do	Usuarios Involu- crados
REQUERIMIENTOS FUNCIONALES						
RF 001	El sistema deberá validar el ingreso al sistema (usuario y contraseña)	Director TI	Alta	Soft- ware	Revi- sión	Sistema
RF 002	El sistema deberá validar que no se repitan los usuarios	Director TI	Alta	Soft- ware	Revi- sión	Sistema
RF 003	El sistema deberá validar que el documento de identificación sea correcto	Director TI	Alta	Soft- ware	Revi- sión	Sistema
RF 004	El sistema deberá validar los campos que permitan el ingreso solo de letras	Software	Alta	Soft- ware	Revi- sión	Sistema
RF 005	El sistema deberá validar los campos que permitan el ingreso solo de números	Director TI	Alta	Soft- ware	Revi- sión	Sistema
RF 006	El sistema deberá validar que los filtros de búsqueda realicen la acción correspondiente	Director TI	Alta	Soft- ware	Revi- sión	Sistema
RF 007	El sistema deberá validar que los botones cumplan la funcionalidad asignada	Director TI	Alta	Soft- ware	Revi- sión	Sistema

NOTA: Esta tabla muestra la matriz de los requerimientos funcionales para el desarrollo del sistema
Elaborado por: Lucía Caguano

Tabla 4

Requerimientos No Funcionales

Los requerimientos no funcionales son características generales del sistema, restricciones o condiciones que pone el usuario para la elaboración, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema.

MATRIZ DE REQUERIMIENTOS						
Identificador	Descripción	Fuente	Prioridad	Tipo	Estado	Usuarios Involucrados
REQUERIMIENTOS NO FUNCIONALES						
RNF 001	Para el desarrollo del sistema se utilizará el lenguaje de programación C#	Usuario	Alta	Usuario	Revisión	Administrador
RNF 002	Para el desarrollo del sistema se va a utilizar el motor de base de datos SLQ Server Management	Usuario	Alta	Usuario	Revisión	Administrador
RNF 003	Los colores de la interfaz de usuario deben ser amigables para el usuario	Usuario	Media	Usuario	Revisión	Usuario
NOTA: Esta tabla muestra los requerimientos no funcionales que va a tener el sistema Elaborado por: Lucía Caguano						

2.01.05 Descripción Detallada

Tabla 5

Requerimiento Funcional 001 (RF001)

Descripción de Requerimiento			
El sistema deberá validar el ingreso al sistema (usuario y contraseña)		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RF 001		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	Funcional
Datos de Entrada	El usuario deberá ingresar las credenciales asignadas por el Administrador (Usuario y Contraseña)		
Descripción	El sistema deberá dejar ingresar si las credenciales están correctas, caso contrario se despliega un mensaje que muestre que los datos de usuario son incorrectos		
Datos de salida	Ingreso a la interfaz correspondiente		
Resultados Esperados	Si se ingresa las credenciales correctas permite el ingreso a la interfaz correspondiente, caso contrario, muestra mensaje de error, y valide que la contraseña cumpla con la política de la empresa en longitud, y consistencia de la misma.		
Origen	Usuario		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Los usuarios deberán ser únicamente empleados de la empresa Biologicsoft.		
Poscondiciones	El sistema debe desplegar la interfaz correcta del usuario que ingresó las credenciales ya sea el Administrador, Diseñador, y Tester.		
Criterios de Aceptación	El usuario no podrá ingresar al sistema mientras no tenga un usuario y una contraseña		
NOTA: Requerimientos funcionales para el desarrollo			
Elaborado por: Lucía Caguano			

Tabla 6

Requerimiento Funcional 002 (RF002)

Descripción de Requerimiento			
El sistema deberá validar que no se repitan los usuarios		Estado	Revisión
Creado por:	Lucía Caguano	Actualizado por:	Lucía Caguano
Fecha Creación:	25\11\2017	Fecha de Actualización:	25\11\2017
Identificador	RF 002		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	Funcional
Datos de Entrada	Nombre de usuario, Nombre, Apellido, Cédula, Correo, Teléfono, Clave y el Rol		
Descripción	El sistema debe validar que los nombres de usuario no se repitan en el sistema		
Datos de salida	Usuario creado y no duplicado		
Resultados Esperados	El sistema deberá permitir el registro de un usuario nuevo, caso contrario, se despliega un mensaje que muestra que ya hay un usuario existente con el mismo nombre de usuario		
Origen	Administrador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Debe tener todos los datos del usuario, un número de cédula válida, y un correo válido		
Poscondiciones	Los datos como nombre, apellido, correo, cédula de identidad y rol pueden duplicarse, la validación de duplicidad se dará únicamente si se repite el nombre de usuario.		
Criterios de Aceptación	Los usuarios deben ser empleados de la empresa.		

NOTA: *Requerimientos funcionales para el desarrollo*

Elaborado por: Lucía Caguano

Tabla 7

Requerimiento Funcional 003 (RF003)

Descripción de Requerimiento			
El sistema deberá validar que el documento de identificación sea correcto		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RF 003		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	Funcional
Datos de Entrada	El número del documento de identificación del usuario		
Descripción	El Administrador debe ingresar el documento de identificación del usuario a registrar		
Datos de salida	El sistema valida número por número que el documento sea correcto, si al final el documento no es válido la caja de texto se pinta de color naranja y al colocar el cursor sobre la caja se muestra un tooltip que indica un mensaje “Ingresa documento válido”		
Resultados Esperados	Permitir que solo se ingresen documento válidos		
Origen	Diseñador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Si no es un documento válido no permite guardar el registro		
Poscondiciones	El documento debe ser una Cédula de identidad, y la caja de texto solo permitirá el ingreso de números		
Criterios de Aceptación	El campo deberá dejar ingresar un máximo de 10 caracteres que es el número de consta en el número de cédula.		

NOTA: *Requerimientos funcionales para el desarrollo*
Elaborado por: Lucía Caguano

Tabla 8

Requerimiento Funcional 004 (RF004)

Descripción de Requerimiento			
El sistema deberá validar los campos que permitan el ingreso solo de letras		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RF 004		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	Funcional
Datos de Entrada	Ingreso de información en las cajas de texto que permiten únicamente letras		
Descripción	Los campos definidos tienen la validación de permitir el ingreso solo de letras		
Datos de salida	Si se intenta ingresar números el campo no debe permitir		
Resultados Esperados	Que el campo se encuentre validado y solo permita ingresar letras		
Origen	Administrador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	RF 002		
ESPECIFICACIÓN			
Precondiciones	El campo debe permitir el ingreso de letras mayúsculas y minúsculas		
Poscondiciones	La caja de texto solo debe permitir el ingreso de letras, no de números ni símbolos.		
Criterios de Aceptación	Los campos para el ingreso serán cajas de texto, y las podrán manipular los usuarios, Administrador, Diseñador, y Tester.		

NOTA: *Requerimientos funcionales para el desarrollo*

Elaborado por: Lucía Caguano

Tabla 9

Requerimiento Funcional 005 (RF005)

Descripción de Requerimiento			
El sistema deberá validar los campos que permitan el ingreso solo de números		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RF 005		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	Funcional
Datos de Entrada	El usuario debe intentar ingresar en los campos letras, símbolos		
Descripción	Las cajas de texto o campos deben estar validados y permitir el ingreso únicamente de números		
Datos de salida	El sistema no debe permitir el ingreso de letras ni símbolos en los campos que están definidos que solo se ingrese números.		
Resultados Esperados	Que los campos se encuentren validados de acuerdo a su función		
Origen	Tester		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Los campos deben permitir el ingreso solo de números		
Poscondiciones	Las cajas de texto deben permitir el ingreso solo de números, no se letras ni símbolos.		
Criterios de Aceptación	Los campos serán cajas de texto y los usuarios que las pueden manipular son, Administrador, Diseñador, Tester.		

NOTA: *Requerimientos funcionales para el desarrollo*
Elaborado por: Lucía Caguano

Tabla 10

Requerimiento Funcional 006 (RF006)

Descripción de Requerimiento			
El sistema deberá validar que los filtros de búsqueda realicen la acción correspondiente		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RF 006		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	Funcional
Datos de Entrada	El usuario debe ingresar los criterios de búsqueda en los filtros para comprobar su funcionalidad		
Descripción	Los filtros deben estar validados de acuerdo a la ventana en la que se encuentra el usuario		
Datos de salida	Resultados de la búsqueda de acuerdo al criterio ingresado		
Resultados Esperados	Verificar que se muestren los registros de acuerdo al criterio de búsqueda ingresado		
Origen	Administrador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	El criterio de búsqueda debe ser ingresado de acuerdo a la pantalla		
Poscondiciones	Solo podrá buscar de acuerdo a lo que la empresa solicitó por cada una de las pantallas		
Criterios de Aceptación	Los criterios de búsqueda serán establecidos en el código del sistema y los podrán realizar los usuarios, Administrador, Diseñador, Tester.		

NOTA: Requerimientos funcionales para el desarrollo

Elaborado por: Lucía Caguano

Tabla 11

Requerimiento Funcional 007 (RF007)

Descripción de Requerimiento			
El sistema deberá validar que que los botones cumplan la funcionalidad asignada		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RF 007		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	Funcional
Datos de Entrada	El usuario debe ingresar información y presionar los botones Guardar, Eliminar, Anular, Aceptar, Cancelar.		
Descripción	El sistema debe comprobar la funcionalidad de los botones		
Datos de salida	Registro guardado, modificado, eliminado o anulado.		
Resultados Esperados	Verificar que los botones cumplan la funcionalidad dispuesta.		
Origen	Administrador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Una vez que se ingrese la información se debe presionar los botones para comprobar la funcionalidad		
Poscondiciones	En el sistema se puede, guardar, eliminar, anular, aceptar y cancelar		
Criterios de Aceptación	El sistema debe permitir realizar las acciones requeridas por el usuario		

NOTA: *Requerimientos funcionales para el desarrollo*

Elaborado por: Lucía Caguano

Tabla 12

Requerimiento No Funcional 001 (RF001)

Descripción de Requerimiento			
Para el desarrollo del sistema se utilizará el lenguaje de programación C#		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RNF 001		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	No Funcional
Datos de Entrada	Lenguaje de programación		
Descripción	El sistema debe estar desarrollado en el lenguaje de programación determinado		
Datos de salida	Aplicación de usuario		
Resultados Esperados	Sistema desarrollado en este entorno de escritorio		
Origen	Administrador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Desarrollar en el lenguaje determinado		
Poscondiciones	Gestión en el lenguaje de programación		
Criterios de Aceptación	Sistema en buenas condiciones		

NOTA: *Requerimientos No funcionales para el desarrollo*
Elaborado por: Lucía Caguano

Tabla 13

Requerimiento No Funcional 002 (RF002)

Descripción de Requerimiento			
Para el desarrollo del sistema de va a utilizar el motor de base de datos SLQ Server Management		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RNF 002		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	No Funcional
Datos de Entrada	Base de datos con relación al aplicativo		
Descripción	Administración de datos seguros		
Datos de salida	El sistema se desarrollará en este motor de base de datos		
Resultados Esperados	Estructura y seguridad		
Origen	Administrador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Crear la base con las tablas necesarias		
Poscondiciones	Gestión de Base de Datos		
Criterios de Aceptación	Estructura y seguridad		
NOTA: Requerimientos No funcionales para el desarrollo			
Elaborado por: Lucía Caguano			

Tabla 14

Requerimiento No Funcional 003 (RF003)

Descripción de Requerimiento			
Los colores de la interfaz de usuario deben ser amigables para el usuario		Estado	Revisión
Creado por:		Actualizado por:	
	Lucía Caguano		Lucía Caguano
Fecha Creación:		Fecha de Actualización:	
	25\11\2017		25\11\2017
Identificador	RNF 003		
Tipo de Requerimiento	Crítico	Tipo de Requerimiento	No Funcional
Datos de Entrada	Diseño de la interfaz		
Descripción	Manejo de colores para que el sistema se muestre atractivo debe contener los colores distintivos de la empresa		
Datos de salida	Visualización del sistema		
Resultados Esperados	Interfaz atractiva e intuitiva		
Origen	Administrador		
Dirigido a	Biologicsoft		
Prioridad	Alta		
Requerimientos Asociados	Ninguno		
ESPECIFICACIÓN			
Precondiciones	Los colores serán elegidos por el usuario		
Poscondiciones	Una vez que los colores se muestren en el prototipo no se podrá cambiar		
Criterios de Aceptación	Una interfaz amigable		

NOTA: *Requerimientos No funcionales para el desarrollo*
Elaborado por: Lucía Caguano

2.02 Mapeo de Involucrados

En este punto se muestra la identificación de las personas involucradas en el proceso de ejecución de las pruebas unitarias, se indicará los involucrados directos e indirectos que van a interactuar con el sistema.

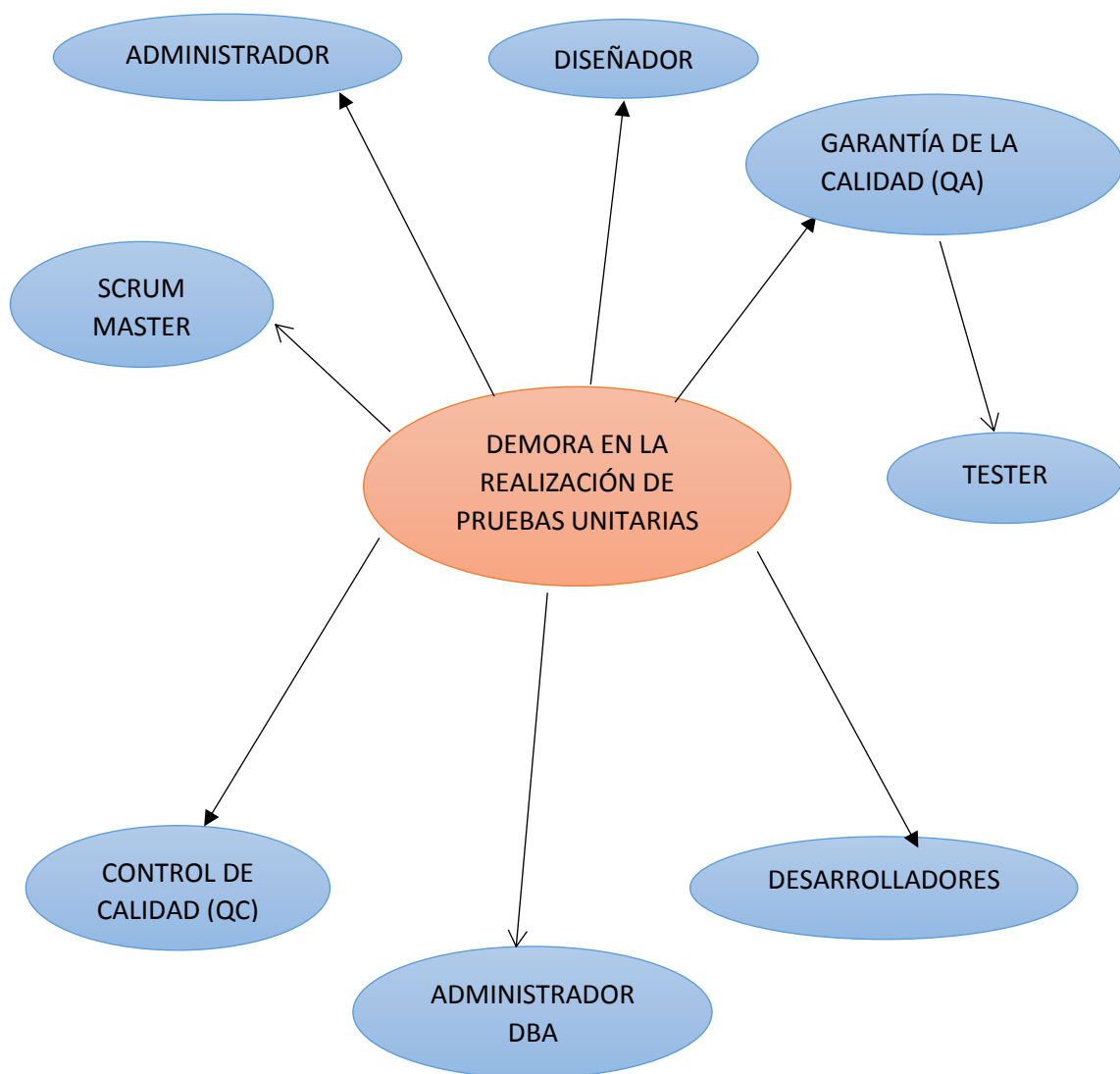


Figura 1: Mapeo de Involucrados. Esta figura muestra los involucrados directa e indirectamente en el sistema

2.03 Matriz de Involucrados

Esta matriz contiene la información de las personas que participan en la problemática del proceso de pruebas unitarias de la Empresa Biologicsoft.

Tabla 15

Matriz de involucrados

Actores Involucrados	Intereses sobre el problema central	Problemas Percibidos	Recursos Mandatos y Capacidades	Intereses sobre el proyecto	Conflictos Potenciales
Adminis-trador	Facilidad en el manejo de la información	Inexistencia de un sistema informático para ejecución de pruebas unitarias	Controlar la calidad de los productos	Mantener el control de los productos	No conocer diariamente el estado de los sistemas
Adminis-trador DBA	Base de datos correcta para mejores resultados	No hay información concreta	Controlar la calidad de los productos	Base con información correcta	Un base no diseñada correctamente
Scrum Master	Crear historias para la corrección de incidencias	Demora en conocer los estados de las ventanas	Lograr tiempos de entrega	Conocer el funcionamiento del sistema	Retraso en cumplimiento de objetivos de Sprint
Diseñador	Diseñar las ventanas de los sistemas a probar	No probar completamente las ventanas	Diseñar correctamente para facilitar las pruebas	Auditar mejor los sistemas	Demora por la complejidad de las ventanas
Garantía de la Calidad (QA)	Facilitar y agilizar el proceso de pruebas	Lentitud en la ejecución de pruebas	Alertar de errores en los productos	Controlar la calidad de los productos	Falta de tiempo para el control de todos los módulos
Control de Calidad (QC)	Mejorar la rapidez de las pruebas	El no cumplimiento de los objetivos	Controlar los tiempos de entrega	Entregar a tiempo los productos	Entrega de sistemas no verificados
Desarro-lladores	Agilizar la corrección de incidencias	Demora en detectar errores	Entregar con mejoras los productos	Agilizar las correcciones	Demora en la corrección de incidencias

NOTA: Requerimientos No funcionales para el desarrollo

Elaborado por: Lucía Caguano

CAPÍTULO III

3 Problemas y Objetivos

3.01 Árbol de Problemas

En este esquema se detalla el problema central que tiene la empresa al realizar las pruebas unitarias de forma manual, causando una demora significativa y ello conlleva a un no cumplimiento de los objetivos planteados.

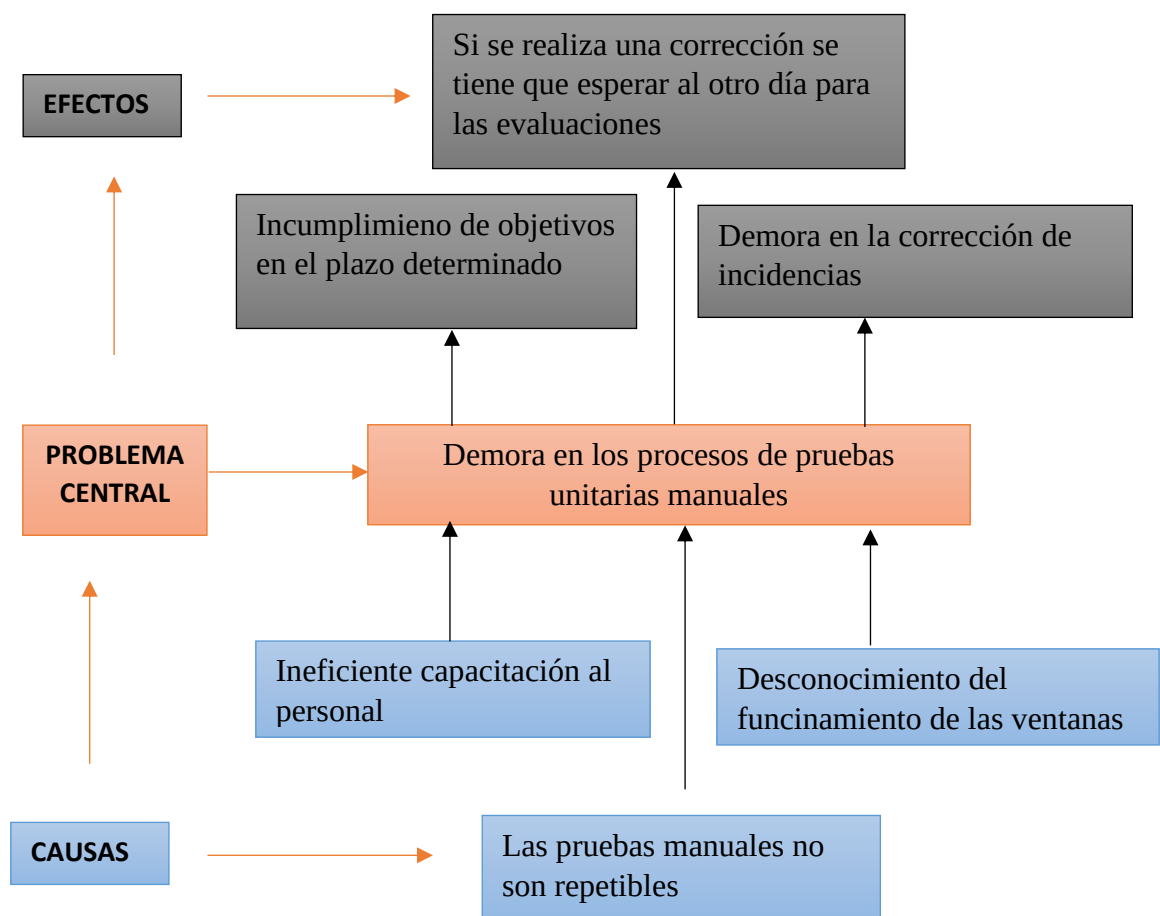


Figura 2: Esta figura corresponde al árbol de problemas, donde se muestra el problema central con sus causas y efectos.

3.02 Árbol de objetivos

En este diagrama se visualizarán los objetivos de la empresa Biologicsoft en referencia al sistema de pruebas unitarias, que consiste en lo que la entidad espera del sistema en desarrollo.

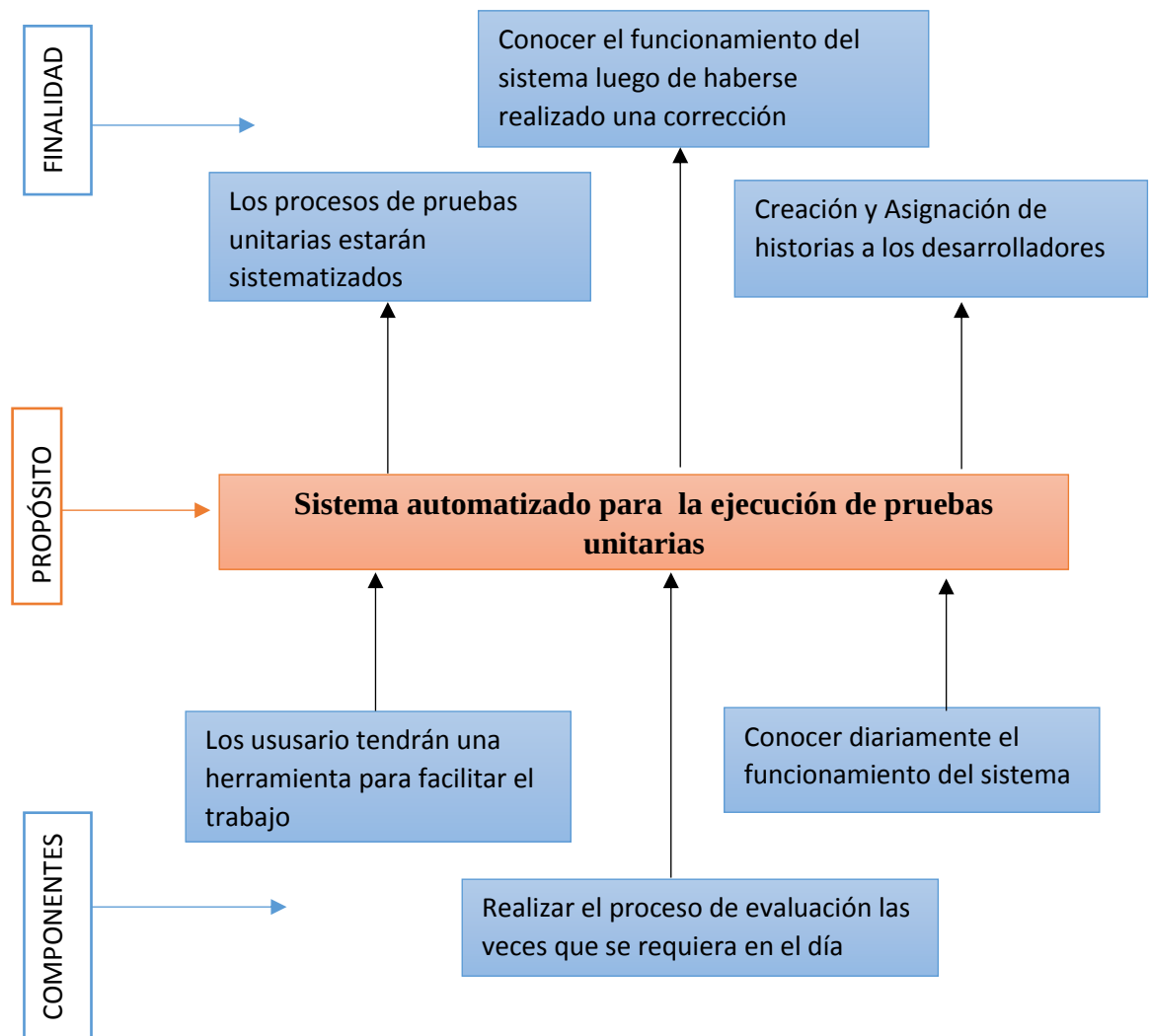


Figura 3: Esta figura corresponde al Árbol de objetivos de la empresa con respecto al sistema en desarrollo.

3.03 Diagramas de Caso de Uso

Este diagrama de casos de uso muestra los procesos que van a realizar cada uno de los usuarios (Administrador, Diseñador, Tester) del sistema. Muestra la relación que van a tener Usuario – Sistema.

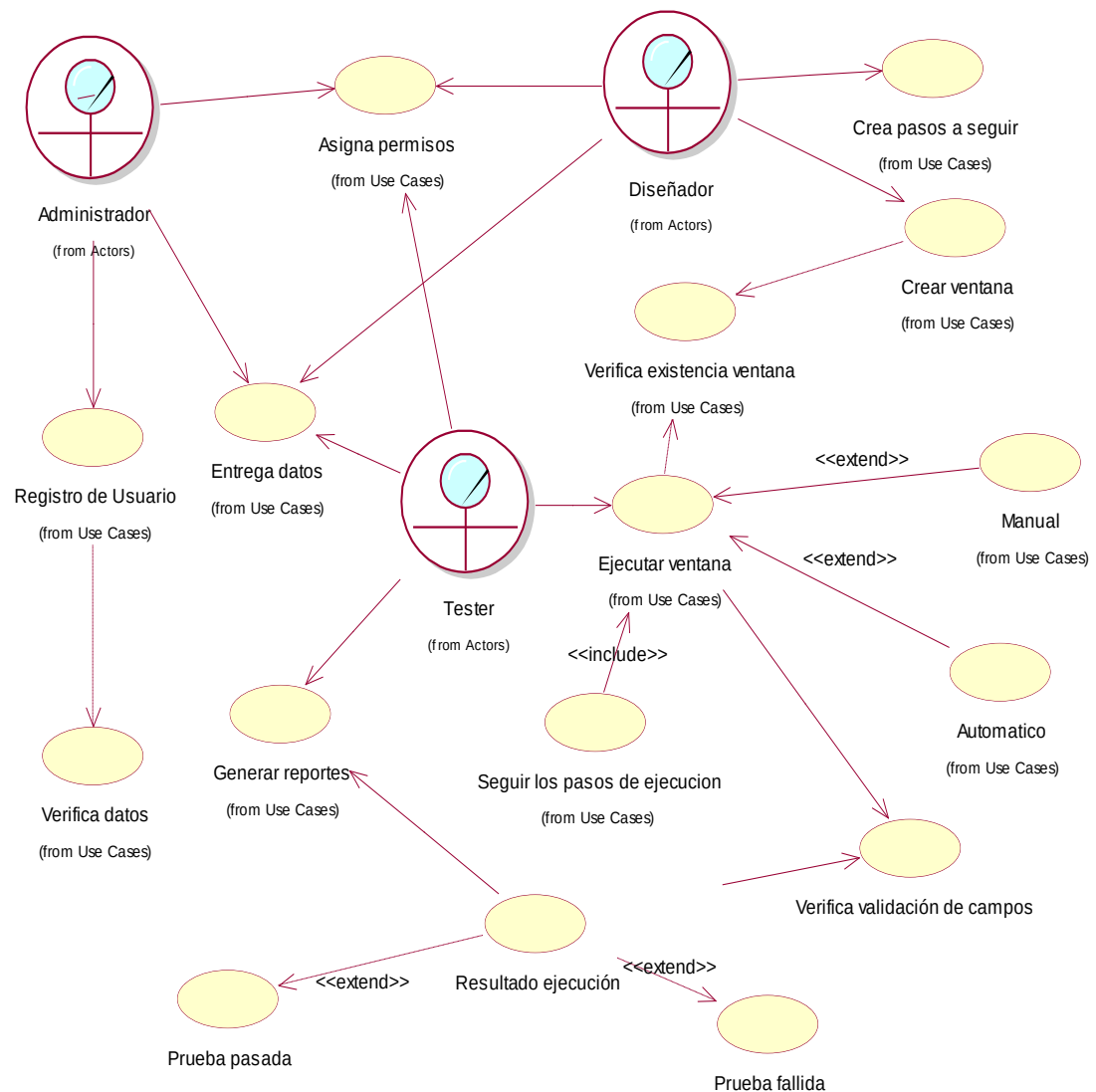


Figura 4: Caso de uso general

DIAGRAMAS DE CASOS DE USO INDIVIDUAL

CASO DE USO UC001: Entrega de datos y registro de usuarios

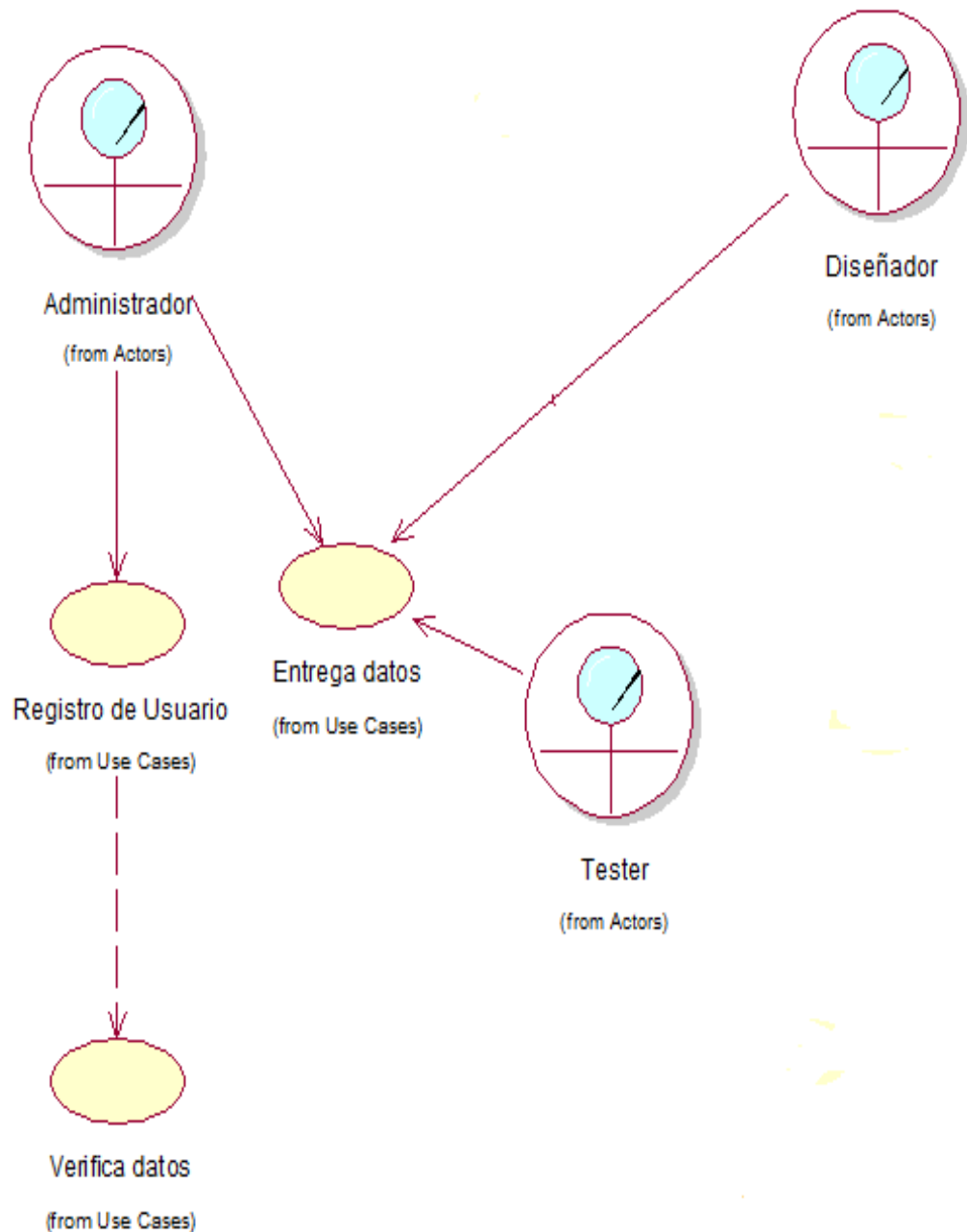


Figura 5: Caso de Uso entrega de datos y registro de usuarios

CASO DE USO UC002: Asignación de permisos

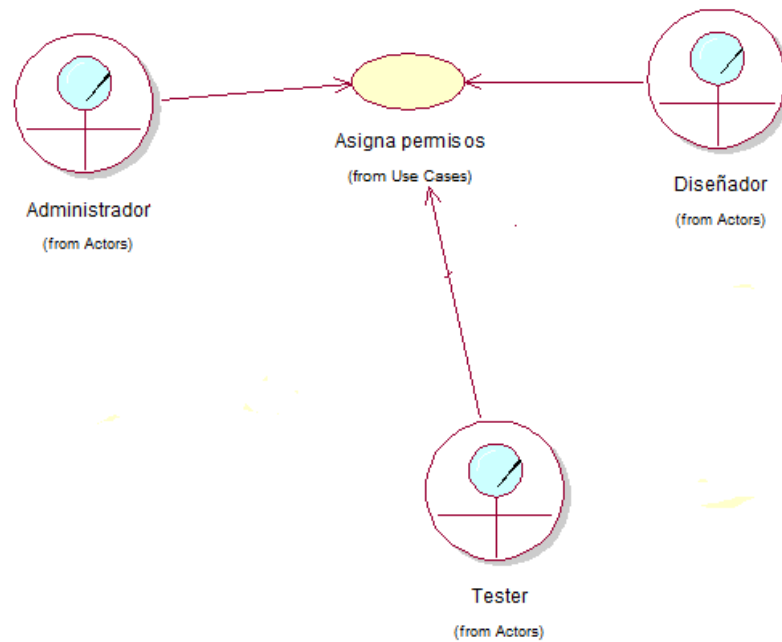


Figura 6: Asignación de permisos

CASO DE USO UC003: Creación de ventanas

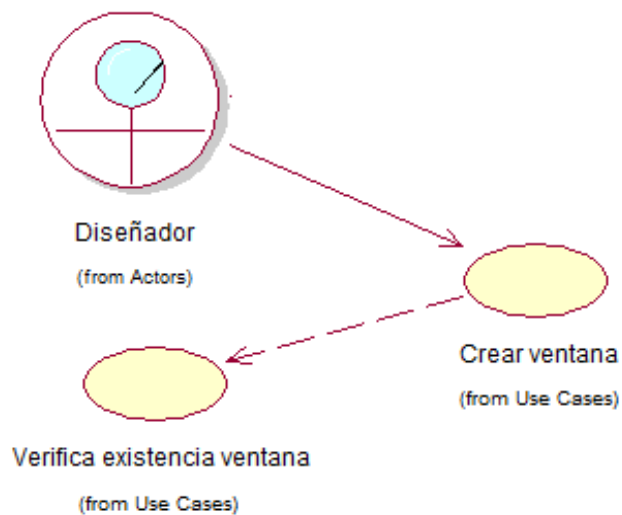


Figura 7: Caso de uso creación de ventanas

CASO DE USO UC004: Crear Pasos a seguir

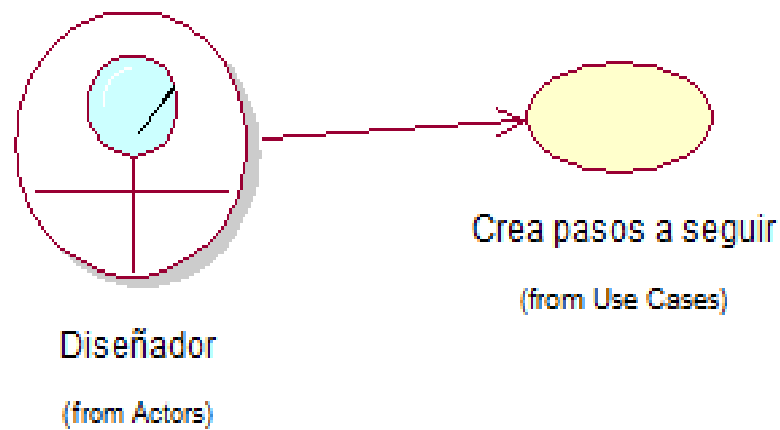


Figura 8: Caso de uso crear pasos a seguir

CASO DE USO UC005: Ejecutar ventanas

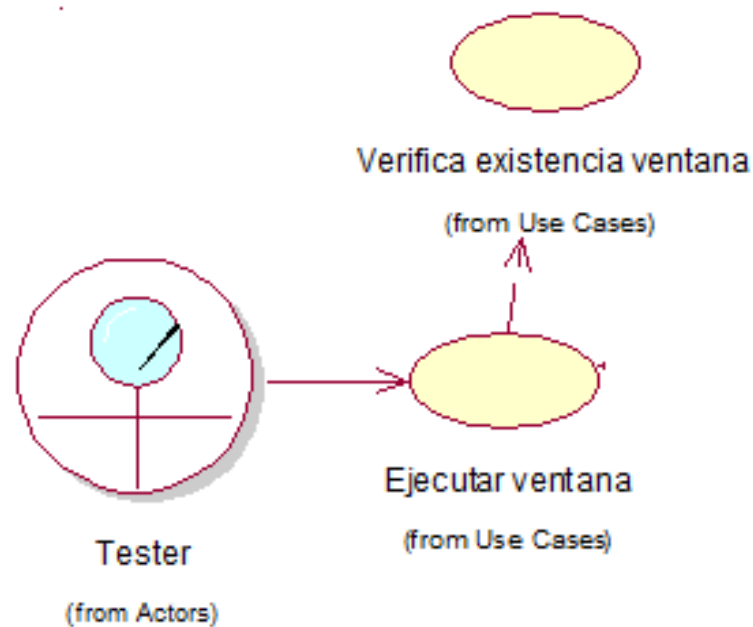


Figura 9: Caso de uso ejecución de ventanas

CASO DE USO UC006: Seguir pasos de ejecución

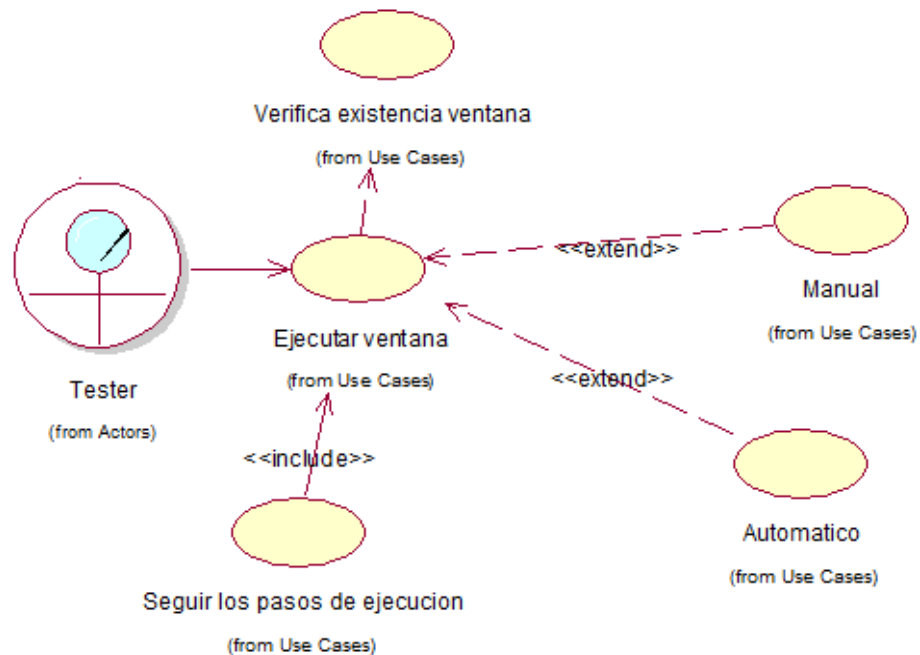


Figura 10: Caso de uso seguir pasos de ejecución

CASO DE USO UC007: Resultado de la Ejecución

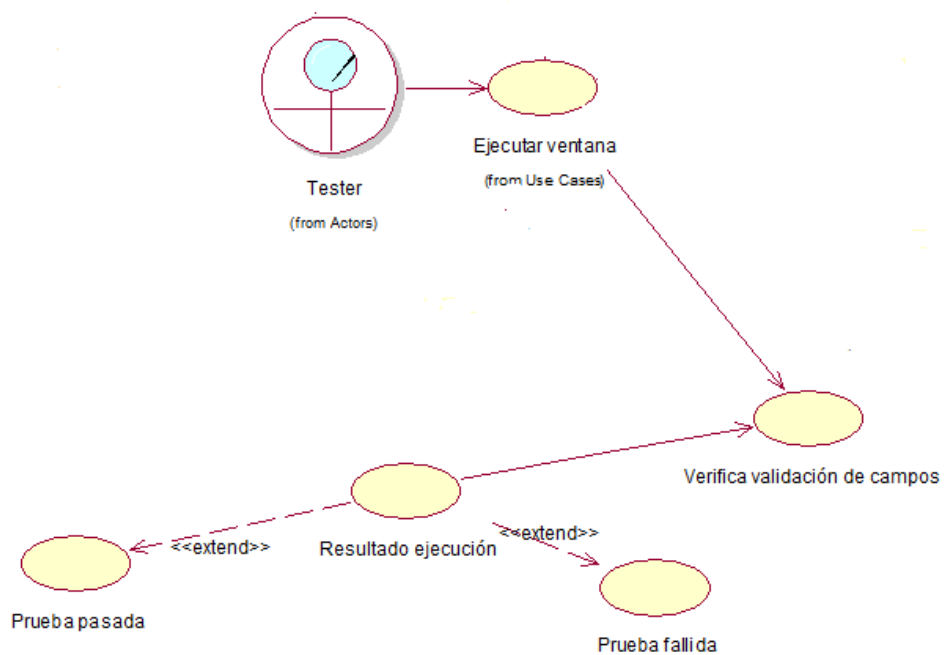


Figura 11: Resultado de la Ejecución

CASO DE USO UC008: Generar Reportes

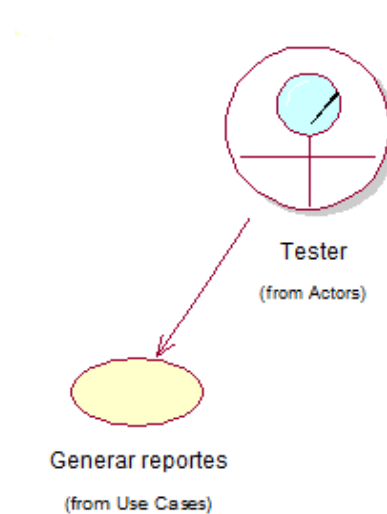


Figura 12: Caso de uso generar reportes

3.04 Especificación de Casos de Uso

Tabla 16

Caso de uso UC001

CASO DE USO	Entrega de datos y Registro de usuarios	
IDENTIFICADOR	UC001	
CURSO TÍPICO DE EVENTOS		
	USUARIOS	SISTEMA
	El administrador solicita al usuario a registrar los datos que necesita para su registro	El sistema solicita los datos que se requieren para la creación de un nuevo usuario
	El administrador comprueba que los datos entregados se puedan guardar	El sistema verifica que los datos entregados por el usuario son reales o que ya no se encuentren registrados
	El administrador guardará los datos del nuevo usuario	El sistema notificará el registro los datos ingresados para el nuevo usuario del sistema
FLUJO ALTERNATIVO		
1.- En caso de que los datos no sean reales el sistema no permitirá que se guarde ningún dato		
2.- En caso de que algún dato como el número de cédula o el código ya se encuentren registrados se mostrará un mensaje de que la existen usuarios registrados con esos datos y no permitirá la creación del usuario.		

Nota: En esta tabla se muestra el caso de uso UC001

Tabla 17

Caso de uso UC002

CASO DE USO	Asignar permisos
IDENTIFICADOR	UC002
CURSO TÍPICO DE EVENTOS	
USUARIOS	SISTEMA
El administrador al momento de registrar el nuevo usuario le asignará el perfil que le corresponde a este.	El sistema permitirá asignar el rol de administrador, diseñador o tester al usuario creado
El administrador asignará el estado de activo al usuario recién creado	El sistema asignará automáticamente el estado de activo al usuario al momento de su creación.
El administrador podrá cambiar el estado de cualquier usuario.	El sistema permitirá cambiar el estado del usuario para su ingreso al sistema
FLUJO ALTERNATIVO	
1.- El usuario solo podrá ingresar a las ventanas que se le dieron los permisos con su interfaz correspondiente	

NOTA: En esta tabla se muestra el caso de uso UC002

Tabla 18

Caso de uso UC003

CASO DE USO	Crear ventanas
IDENTIFICADOR	UC003
CURSO TÍPICO DE EVENTOS	
USUARIOS	SISTEMA
El diseñador creará una ventana en base a la ventana del proyecto a realizar la prueba	El sistema creará una ventana para la ejecución de las pruebas de cualquier sistema
El diseñador debe revisar ventana por ventana en el sistema para crear una para probar	El sistema debe crear la cantidad de ventanas que el usuario requiera
El diseñador le debe dar un nombre a la ventana	El sistema debe receptar el nombre de la ventana que ingresó
FLUJO ALTERNATIVO	
1.- La ventanas no podrán ser repetidas	

NOTA: En esta tabla se muestra el caso de uso UC003

Tabla 19

Caso de uso UC004

CASO DE USO	Crear pasos a seguir
IDENTIFICADOR	UC004
CURSO TÍPICO DE EVENTOS	
USUARIOS	SISTEMA
El diseñador creará los pasos mediante los cuales se debe realizar la ejecución de la ventana para pruebas.	El sistema permitirá el ingreso de las instrucciones mediante los cuales se podrá ejecutar la ventana de pruebas.
El diseñador definirá el orden mediante el cual se debe ejecutar la realización de la ventana de prueba.	El sistema asignará el orden en el cual debe ejecutarse la ventana de pruebas.
El diseñador puede diseñar una ventana sin un orden	El sistema debe probar las ventanas sea de forma secuencial como individual
FLUJO ALTERNATIVO	
1.- Se permitirá probar las ventanas de forma individual, se permitirá probar las ventanas en secuencia	
NOTA: En esta tabla se muestra el caso de uso UC004	

Tabla 20

Caso de uso UC005

CASO DE USO	Ejecutar ventana
IDENTIFICADOR	UC005
CURSO TÍPICO DE EVENTOS	
USUARIOS	SISTEMA
El tester ejecutará la ventana que prueba la funcionalidad del sistema	El sistema abrirá la ventana que escoja el tester para realizar la ejecución de las pruebas unitarias
El tester comprobará que la ventana de prueba exista en el sistema a probar	El sistema comprobará que la ventana creada existe dentro del sistema al cual se le realizará las pruebas unitarias.
El Tester solo ejecutará las ventanas diseñadas, no tiene la potestad de diseñar ventanas	El sistema solo permite que el Tester realice la ejecución de pruebas de las ventanas
FLUJO ALTERNATIVO	
1.- En caso de no existir la ventana dentro del proyecto el sistema de pruebas unitarias mostrará un mensaje que indique que no existe la ventana en el sistema a probar	
NOTA: En esta tabla se muestra el caso de uso UC005	

Tabla 21

Caso de uso UC006

CASO DE USO	Seguir pasos de ejecución
IDENTIFICADOR	UC006

CURSO TÍPICO DE EVENTOS

USUARIOS	SISTEMA
El tester seguirá las instrucciones establecidas para realizar la prueba de la ventana.	El sistema mostrará y ejecutará los pasos que se tienen que seguir para la ejecución de la prueba unitaria.
El tester escogerá si la prueba se realiza de manera automática o paso por paso.	El sistema dará la opción de escoger si la prueba de la ventana se realizará de manera automática o manual para comprobar su funcionamiento.

FLUJO ALTERNATIVO

1.- Se tendrá dos formas de realizar las pruebas unitarias

NOTA: En esta tabla se muestra el caso de uso UC006

Tabla 22

Caso de uso UC007

CASO DE USO	Resultados de la ejecución
IDENTIFICADOR	UC007

CURSO TÍPICO DE EVENTOS

USUARIOS	SISTEMA
El tester verifica los resultados	El sistema mostrará la información de los resultados
Verificar los resultados de la ejecución	El sistema debe mostrar todos los campos que pasaron la prueba y los campos que fallaron

FLUJO ALTERNATIVO

1.- Se tendrá un reporte de la ejecución de las pruebas

NOTA: En esta tabla se muestra el caso de uso UC006

Tabla 23

Caso de uso UC008

CASO DE USO	Generar reporte
IDENTIFICADOR	UC008
CURSO TÍPICO DE EVENTOS	
USUARIOS	SISTEMA
El tester generará un reporte con los resultados de las pruebas realizadas	El sistema genera un reporte con todos los resultados de la prueba de la ventana tales como errores, tiempo de ejecución, etc..
El tester enviará el reporte generado al encargado de la supervisión.	El sistema genera el reporte final en formato PDF con todos los detalles de la prueba y lo enviará al supervisor encargado.
FLUJO ALTERNATIVO	
1.- El reporte será generado una vez que se realicen las pruebas	
<i>NOTA: En esta tabla se muestra el caso de uso UC007</i>	

3.05 Casos de Uso de Realización y Especificación

Caso de Uso Realización UCR001 Registro de Usuario

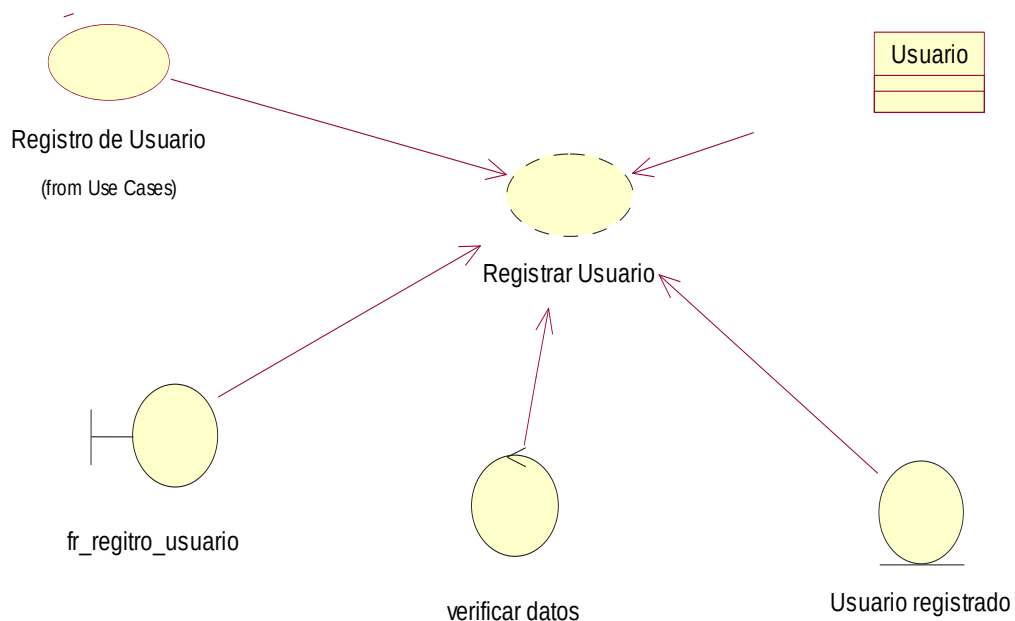


Figura 13: UCR001 Caso de uso de realización de registro de usuario

Tabla 24

Especificación de caso de realización UCR001

NOMBRE	Registro de Usuario
IDETIFICACIÓN	UC001
RESPONSABILIDADES	Registro de usuarios en el sistema
TIPO	Usuario
REFERENCIA DE CASOS DE USO	Entregada de datos Verificar datos
REFERENCIAS REQUISITOS	Solo el usuario administrador ejecutará esta ventana.
PRECONDICIONES	
El usuario administrador solicita los datos necesarios al usuario para su registro en el sistema.	
POSCONDICIONES	
El usuario será registrado en la base de datos del sistema.	
SALIDAS PANTALLA	
Se mostrará un mensaje de éxito en el registro de datos.	

NOTA: En esta tabla se muestra la especificación de casos de uso UCR001

Caso de Uso Realización UCR002 Asignar Permisos

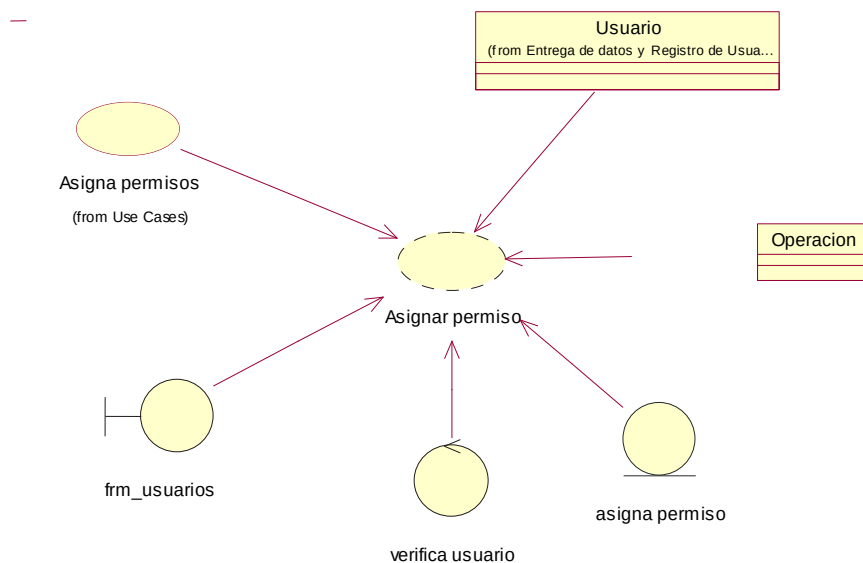


Figura 14: UCR002 Caso de uso de realización asignar permisos

Tabla 25

Especificación de caso de realización UCR002

NOMBRE	Asignar permisos
IDETIFICACIÓN	UC002
RESPONSABILIDADES	Asignar los permisos que tendrá el usuario.
TIPO	Sistema
REFERENCIA DE CASOS DE USO	
REFERENCIAS REQUISITOS	El sistema define el estado del usuario de manera automática.
PRECONDICIONES	
Al momento del registro de un nuevo usuario el administrador escogerá el tipo de operación que este realizará.	
POSCONDICIONES	
El usuario creado solo podrá realizar las acciones según el tipo de operación con el que fue registrado.	
El sistema registrará al nuevo usuario con estado de activo de manera automática al momento del registro	
SALIDAS PANTALLA	
Se mostrará un mensaje de éxito en el registro de datos y la operación que tendrá este usuario.	

NOTA: En esta tabla se muestra la especificación de casos de uso UCR002

Caso de Uso Realización UCR003 Creación Ventana

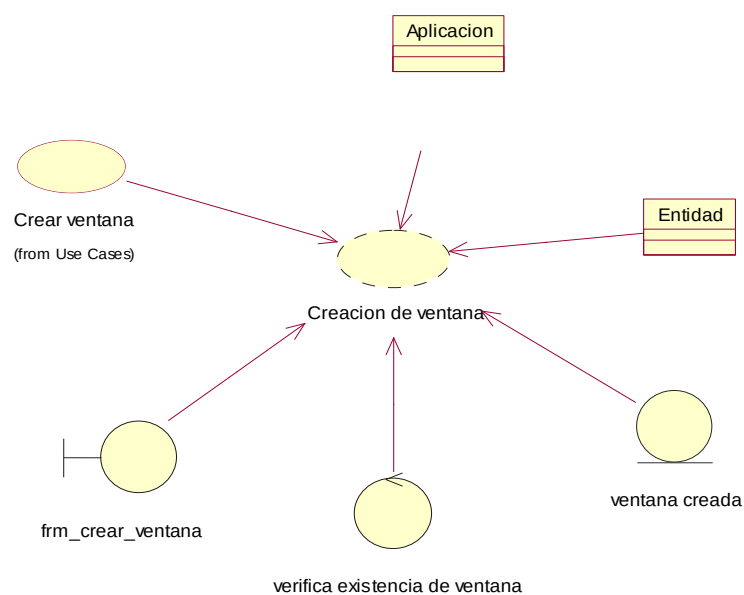


Figura 15: UCR003 Caso de uso de realización creación ventana

Tabla 26

Especificación de caso de realización UCR003

NOMBRE	Crear ventana
IDETIFICACIÓN	UC003
RESPONSABILIDADES	Crear la ventana que realizará la ejecución de la prueba unitaria.
TIPO	Usuario
REFERENCIA DE CASOS DE USO	Verifica existencia de la ventana
REFERENCIAS REQUISITOS	El sistema permitirá la creación de una ventana acorde a lo que se necesita para la realización de la prueba unitaria

PRECONDICIONES

El sistema comprobará que exista la ventana en el sistema al cual se le va a realizar la prueba

POSCONDICIONES

El usuario diseñador creará la ventana para realizar la prueba unitaria a la aplicación a probar.

SALIDAS PANTALLA

El sistema mostrará el diseño final de la ventana creada.

NOTA: En esta tabla se muestra la especificación de casos de uso UCR003

Caso de Uso Realización UCR004 Crear pasos a seguir

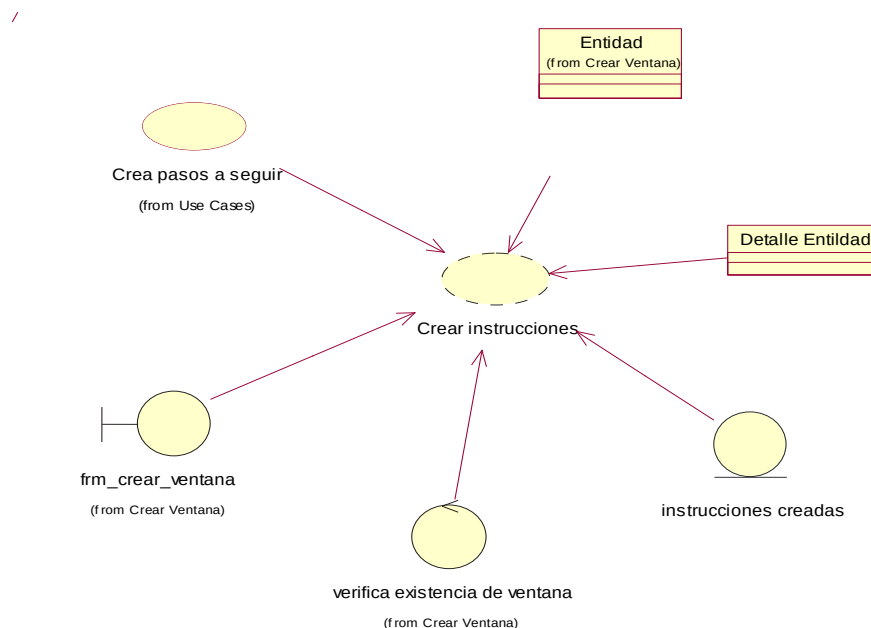


Figura 16: UCR004 Caso de uso de creación de pasos a seguir

Tabla 27

Especificación de caso de realización UCR004

NOMBRE	Crear pasos a seguir
IDETIFICACIÓN	UC004
RESPONSABILIDADES	Crear las instrucciones que se deben seguir para la ejecución de la prueba unitaria
TIPO	Usuario
REFERENCIA DE CASOS DE USO	
REFERENCIAS REQUISITOS	El usuario registrará los pasos a seguir para la realización de la prueba unitaria, así como el orden en el que se ejecuten.

PRECONDICIONES

El usuario diseñador será el único que podrá crear las instrucciones y deberá conocer el objetivo final de la ventana a probar.

POSCONDICIONES

Se registra las instrucciones para ejecutar la ventana.

Se da la opción que las instrucciones se ejecuten de manera automática o de manera manual.

SALIDAS PANTALLA

El sistema indicará que las instrucciones fueron creadas con éxito y la ventana a la cual están asociadas.

NOTA: En esta tabla se muestra la especificación de casos de uso UCR004

Caso de Uso Realización UCR005 Ejecutar Ventanas

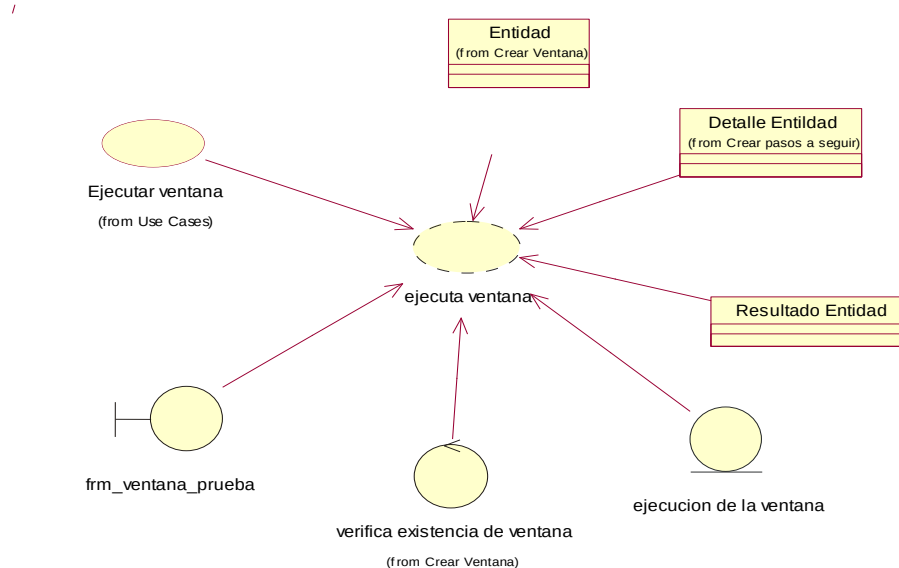


Figura 17: UCR005 Caso de uso de realización de ejecución de ventanas

Tabla 28

Especificación de caso de realización UCR005

NOMBRE	Ejecutar ventana
IDETIFICACIÓN	UC005
RESPONSABILIDADES	Ejecutar la ventana creada para la prueba unitaria
TIPO	Usuario
REFERENCIA DE CASOS DE USO	Verifica existencia ventana
REFERENCIAS REQUISITOS	La ventana de prueba y la del proyecto a probar deberán coincidir con las especificaciones.

PRECONDICIONES

El usuario tester será el único que puede ejecutar la ventana al momento de realizar la prueba unitaria.

POSCONDICIONES

El sistema mostrará los pasos a seguir para realizar la ejecución de la prueba unitaria.

SALIDAS PANTALLA

Se mostrará un cuadro con las instrucciones que se deben seguir en la ejecución de la ventana.

NOTA: En esta tabla se muestra la especificación de casos de uso UCR005

Caso de Uso Realización UCR006 Seguir instrucciones

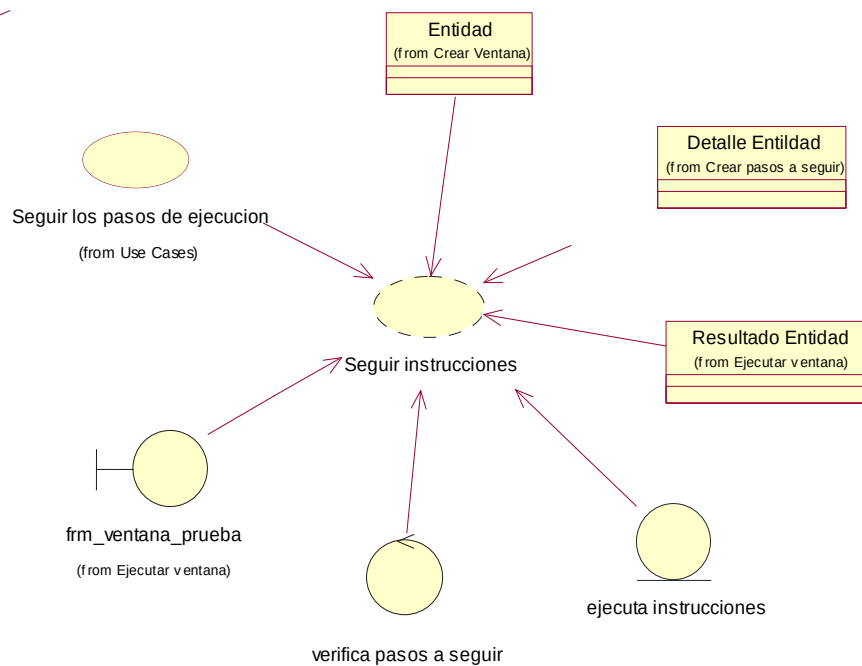


Figura 18: UCR006 Caso de uso de seguir pasos de ejecución

Tabla 29

Especificación de caso de realización UCR006

NOMBRE	Seguir pasos de ejecución
IDETIFICACIÓN	UC006
RESPONSABILIDADES	Seguir las instrucciones establecida para la realización de la ventana de prueba
TIPO	Sistema
REFERENCIA DE CASOS DE USO	Ejecutar ventana
REFERENCIAS REQUISITOS	Las instrucciones ya tienen que estar creadas antes de la ejecución de la ventana

PRECONDICIONES

Antes de la ejecución de las instrucciones se permite escoger si las pruebas se realizarán de manera manual o automática.

POSCONDICIONES

Si se escogió que las instrucciones se ejecuten de manera automática se ejecutarán todos los pasos juntos.

Si se escogió que las instrucciones se ejecuten de manera manual se mostrará los resultados obtenidos en cada paso y no se procederá al siguiente hasta que el tester lo indique

SALIDAS PANTALLA

Se mostrará un resumen de los resultados obtenidos durante la ejecución de la ventana.

NOTA: En esta tabla se muestra la especificación de caso de realización UCR006

Caso de Uso Realización UCR007 Resultados de la ejecución

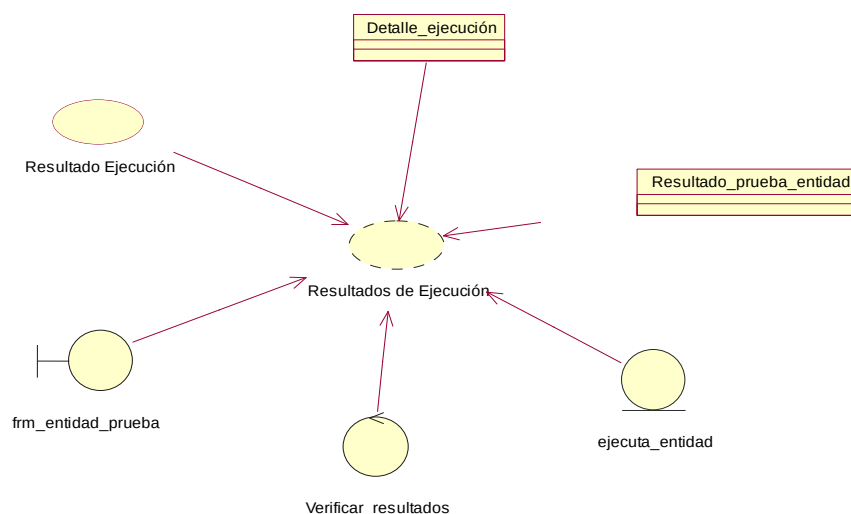


Figura 19: UCR007 Caso de Realización Resultados de Ejecución

Tabla 30

Especificación de caso de realización UCR007

NOMBRE	Resultados de Ejecución
IDENTIFICACIÓN	UC007
RESPONSABILIDADES	Se mostrará los resultados de los campos pasados y fallados
TIPO	Sistema
REFERENCIA DE CASOS DE USO	Ejecución de Ventanas
REFERENCIAS REQUISITOS	Haber ejecutado de manera correcta las instrucciones para la prueba unitaria
PRECONDICIONES	
El usuario tester ejecuta las entidades a evaluar	
POSCONDICIONES	
Una vez que se genere los resultados se genera un reporte	
SALIDAS PANTALLA	
Se muestran los resultados de la ejecución realizada mostrando los campos que pasaron y fallaron las pruebas	

NOTA: En esta tabla se muestra la especificación de caso de realización UCR007

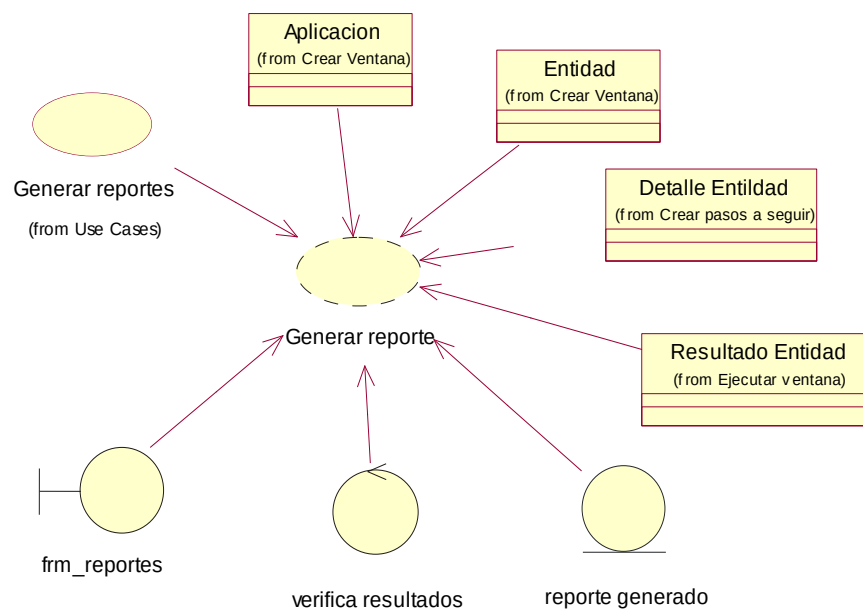
Caso de Uso Realización UCR008 Generar Reportes

Figura 20: UCR008 Caso de realización generar reportes

Tabla 31

Especificación de caso de realización UCR008

NOMBRE	Generar reportes
IDETIFICACIÓN	UC008
RESPONSABILIDADES	Se creará un reporte con todos los resultados que se obtuvo durante la prueba unitaria
TIPO	Sistema
REFERENCIA DE CASOS DE USO	Seguir pasos de ejecución
REFERENCIAS REQUISITOS	Haber ejecutado de manera correcta las instrucciones para la prueba unitaria

PRECONDICIONES

El usuario tester visualizará un resumen de los resultados obtenidos para dar paso a generar un reporte detallado de los resultados

POSCONDICIONES

El sistema generará un reporte y se le enviará al supervisor encargado de la prueba unitaria

SALIDAS PANTALLA

Se muestra un cuadro de mensaje que indica la generación del reporte con éxito.

NOTA: En esta tabla se muestra la especificación de caso de realización UCR008

3.06 Diagramas de Secuencia del Sistema

Diagramas de Secuencia 001 Registro de Usuarios

En este diagrama representa el proceso que debe realizar el usuario Administrador para registrar los usuarios

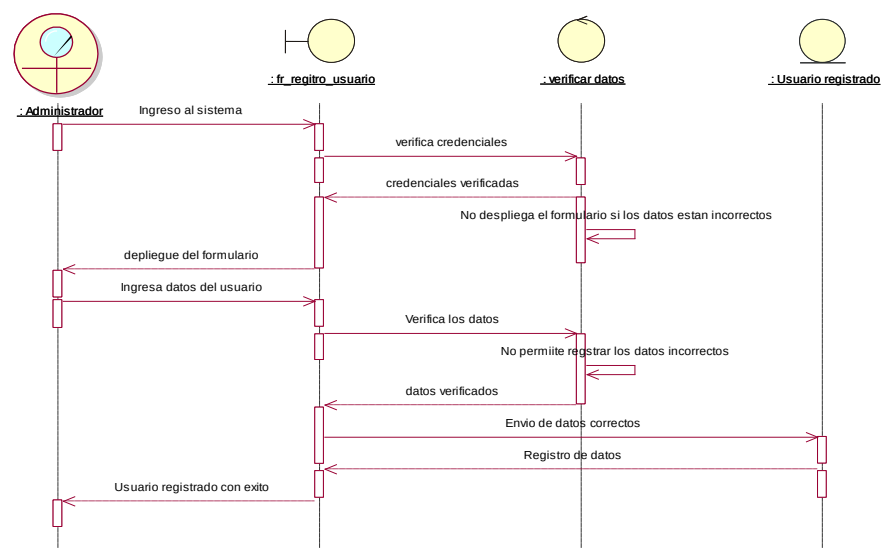


Figura 21: Diagrama de Secuencia Registro de Usuarios

Diagramas de Secuencia 002

Asignar Permisos

En este diagrama se despliega el proceso que realiza el usuario Administrador para la asignación de permisos a los usuarios creados

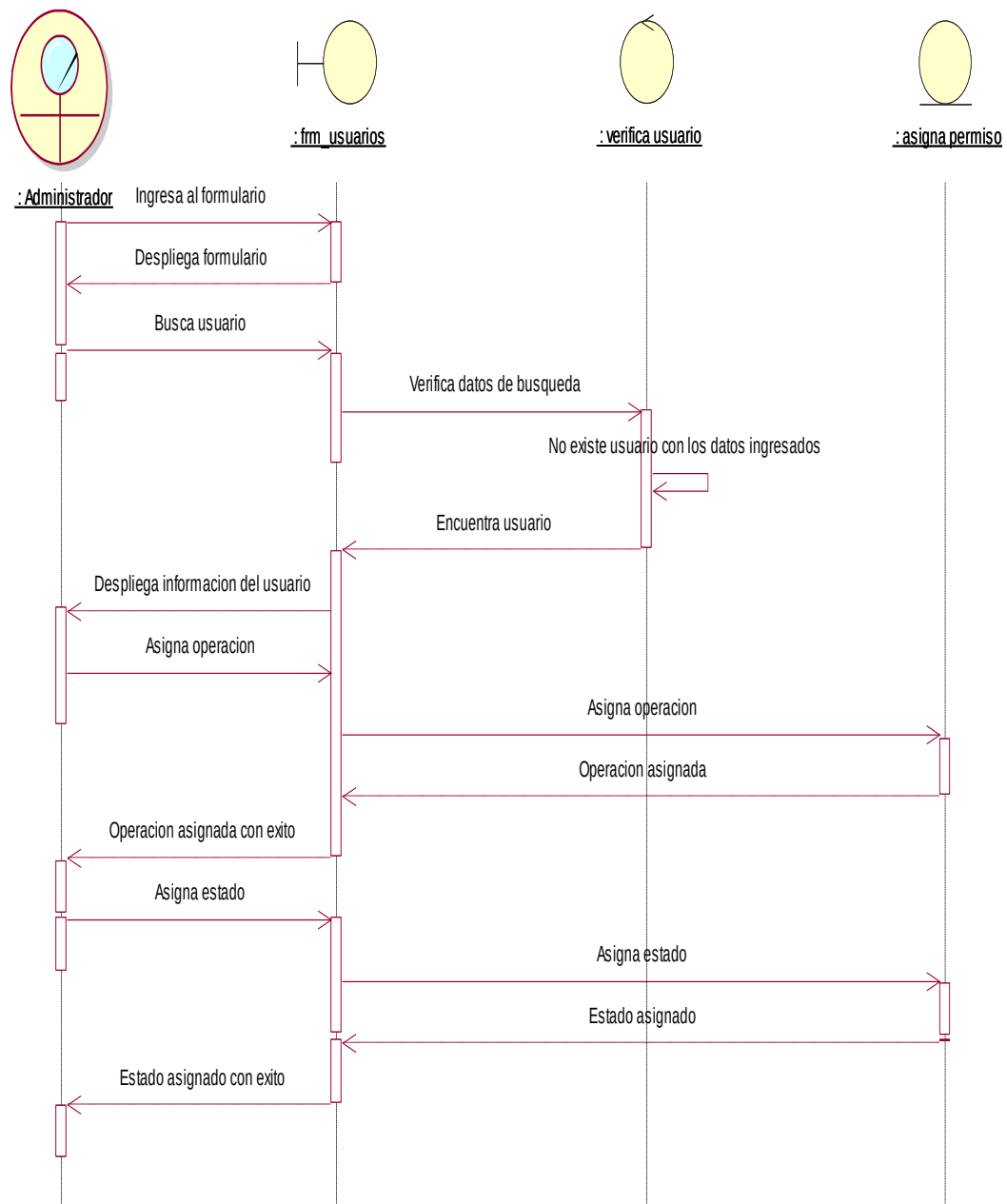


Figura 22: Diagrama de Secuencia Asignación de Permisos

Diagramas de Secuencia 003

Crear Ventanas

En este diagrama de secuencia se muestra el proceso para realizar la creación de las ventanas, este proceso lo realiza el usuario Diseñador.

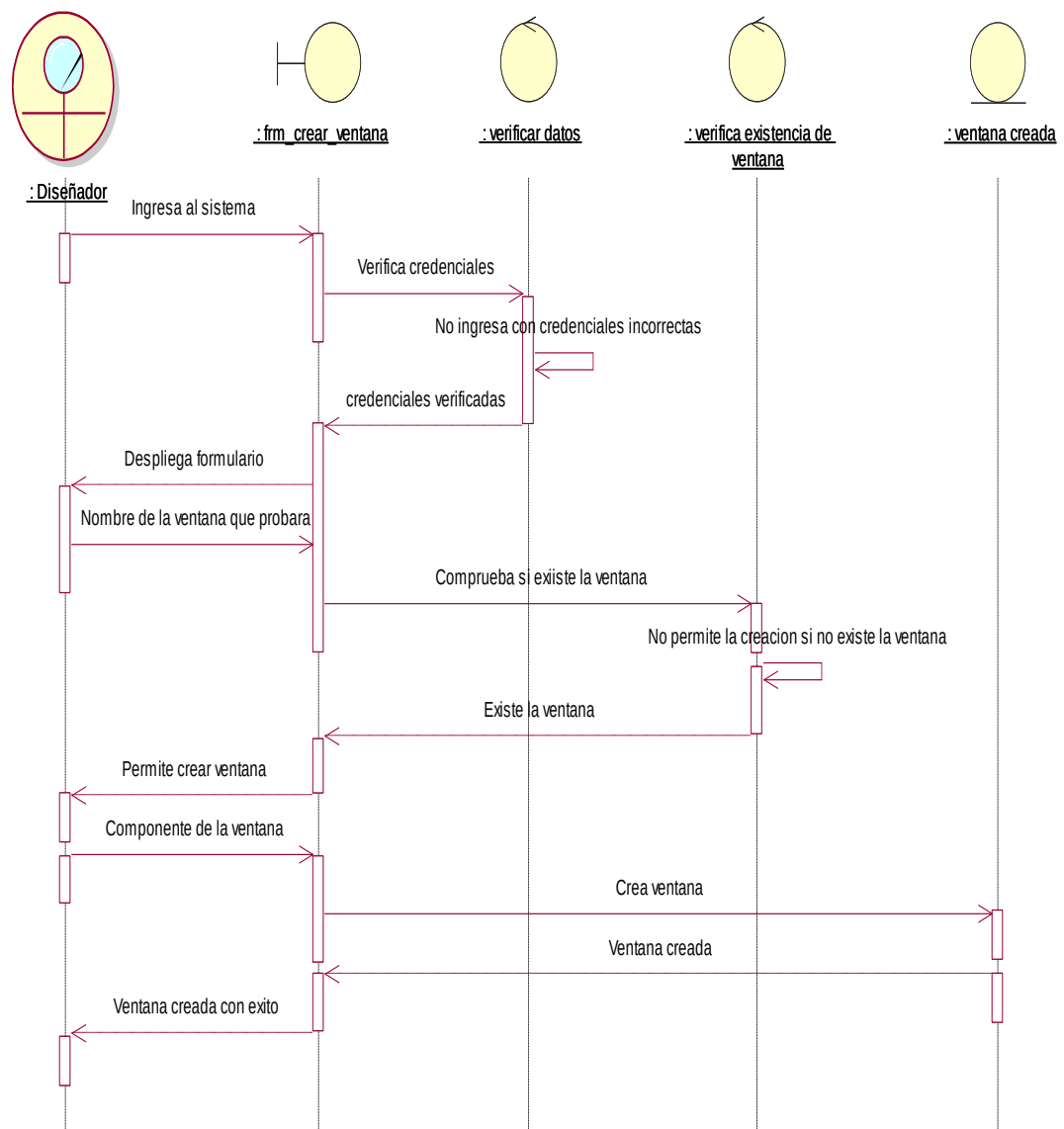


Figura 23: Diagrama de Secuencia Crear Ventanas

Diagramas de Secuencia 004

Crear Pasos a Seguir

En este diagrama de secuencia muestra el proceso que realiza el usuario diseñador con los pasos a seguir

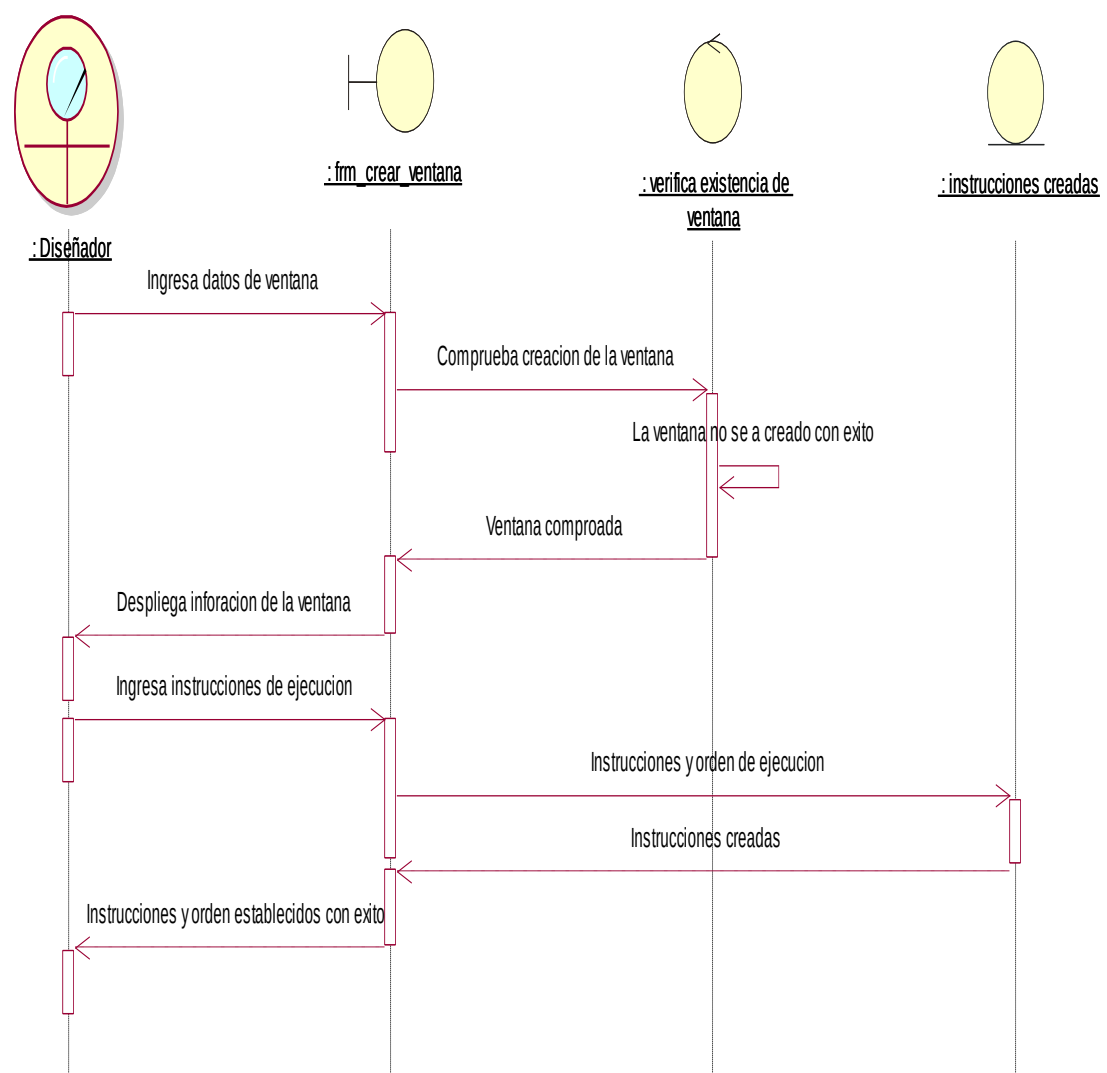


Figura 24: Diagrama de Secuencia Pasos a Seguir

Diagramas de Secuencia 005

Ejecución de Ventanas

Este diagrama de secuencia muestra el proceso que realiza el usuario Tester para ejecutar las pruebas de las ventanas

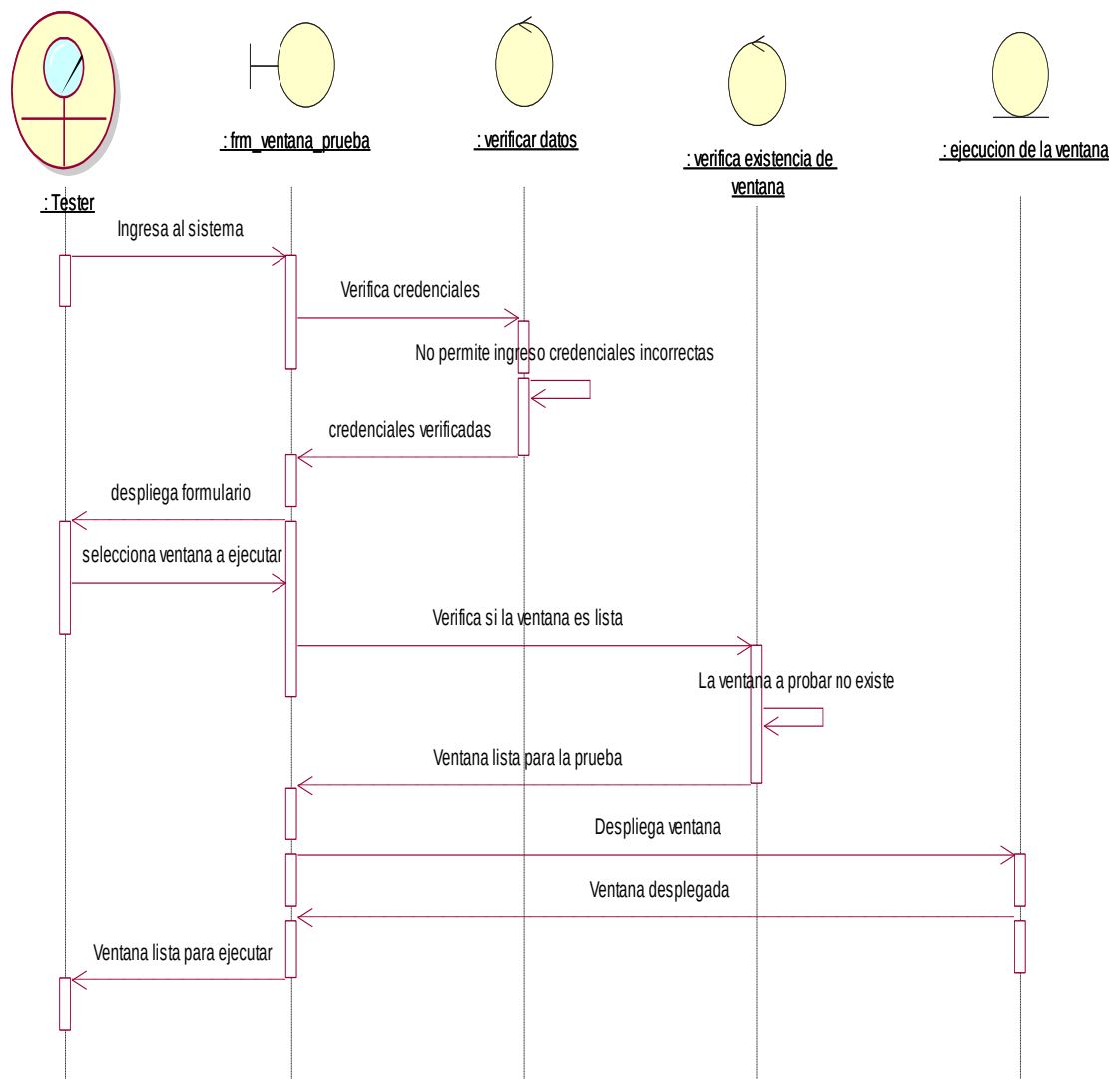


Figura 25: Diagrama de Secuencia Ejecutar Ventanas

Diagramas de Secuencia 006

Seguir Pasos de Ejecución

En este diagrama de secuencia se muestra el proceso que debe seguir el usuario Tester para la ejecución de las ventanas

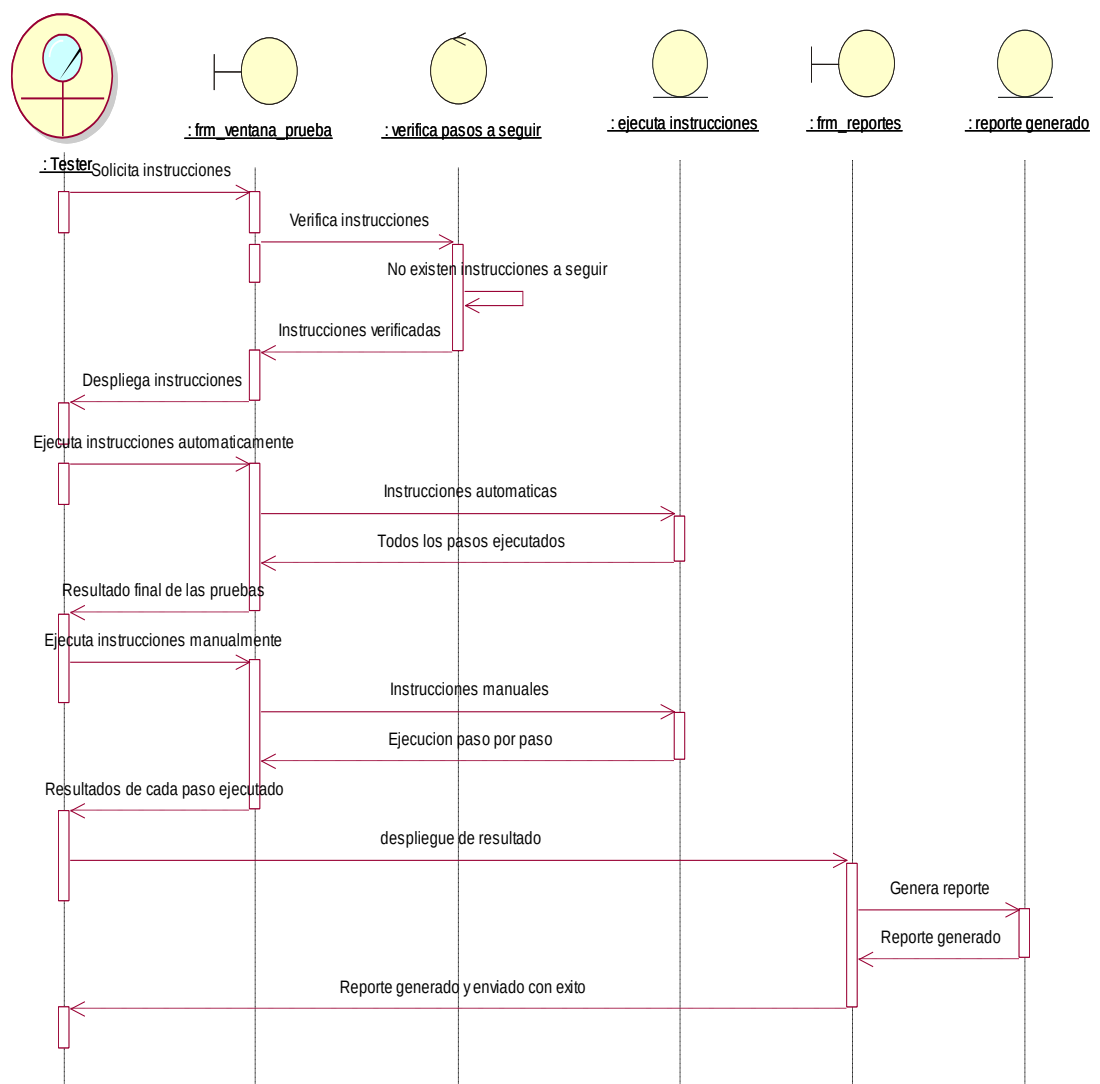


Figura 26: Diagrama de Secuencia Pasos a Seguir

Diagrama de colaboración 001

Registro de usuarios

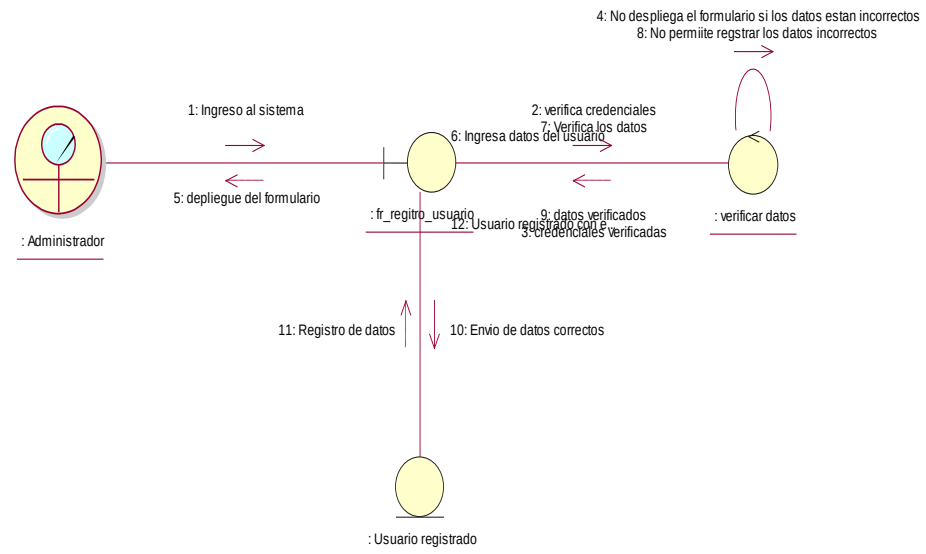


Figura 27: Diagrama de Colaboración Registro de Usuarios

Diagrama de colaboración 002

Asigna Permisos

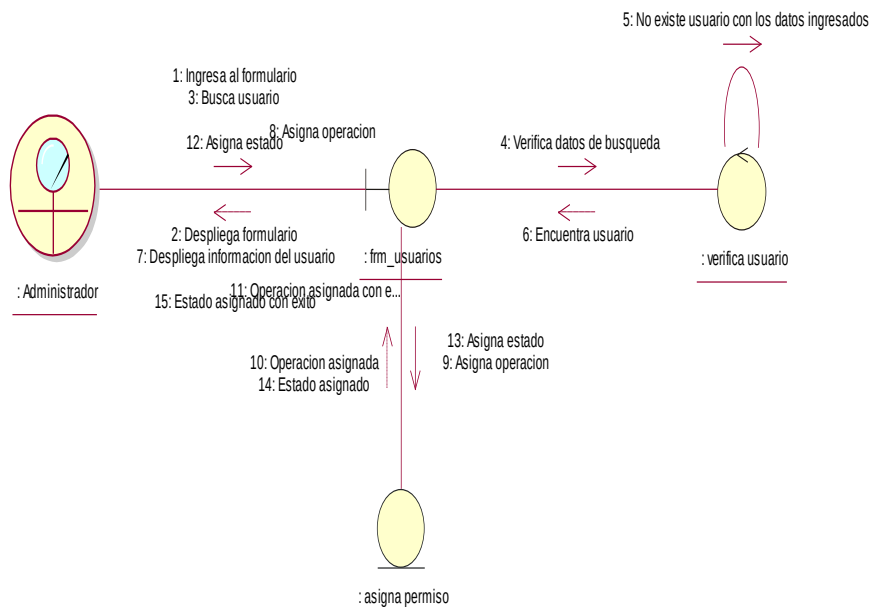


Figura 28: Diagrama de Colaboración Asignación de Permisos

Diagrama de colaboración 003

Crear ventana

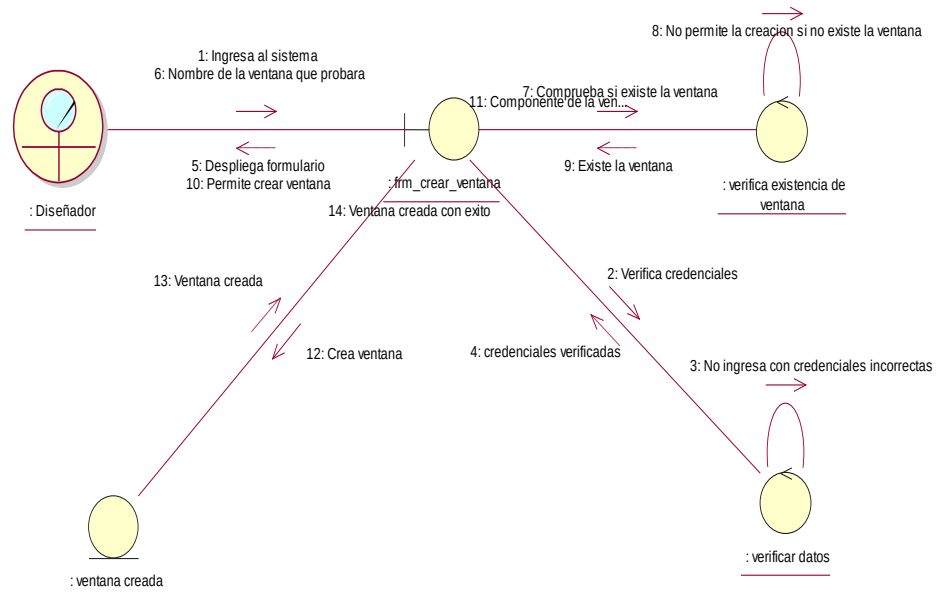


Figura 29: Diagrama de Colaboración Creación de Ventanas

Diagrama de colaboración 004

Crear pasos a seguir

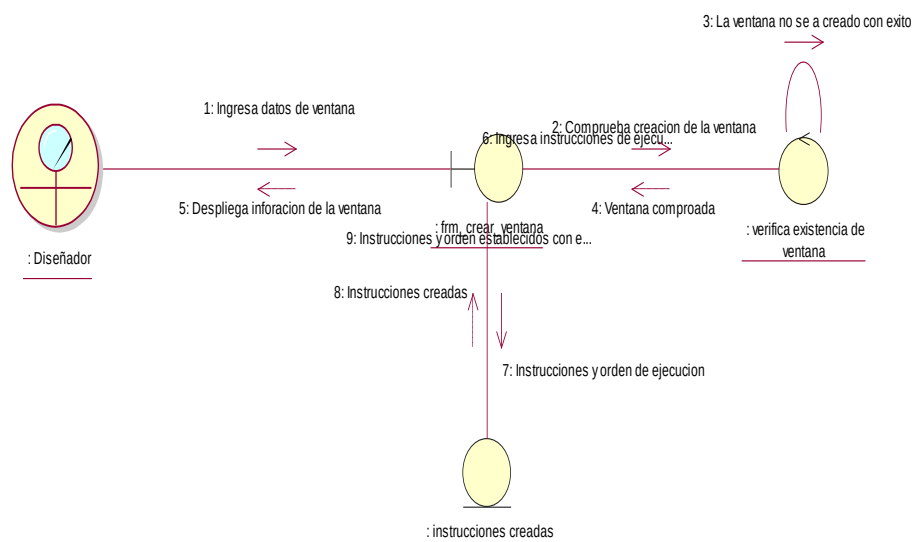


Figura 30: Diagrama de Colaboración Creación de Pasos a Seguir

Diagrama de colaboración 005

Ejecutar ventana

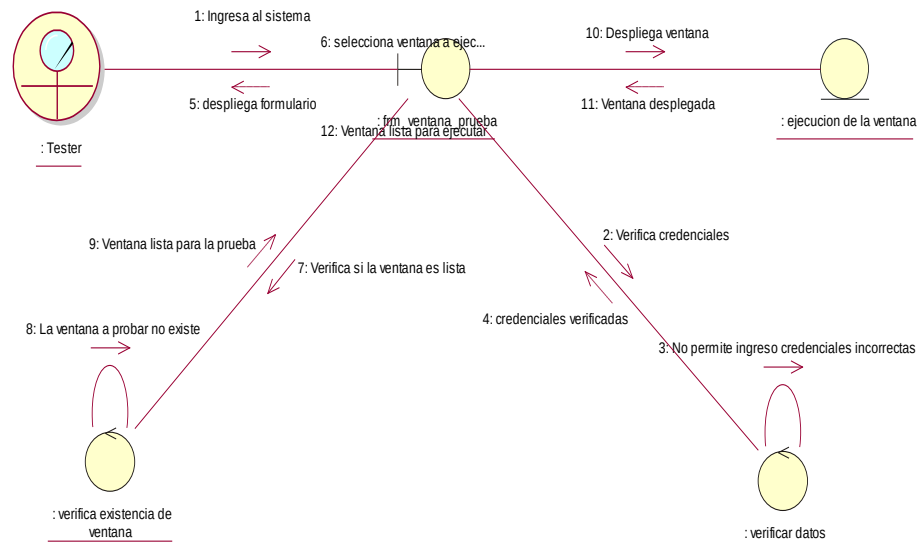


Figura 31: Diagrama de Colaboración Ejecutar Ventanas

Diagrama de colaboración 006

Seguir pasos de ejecución

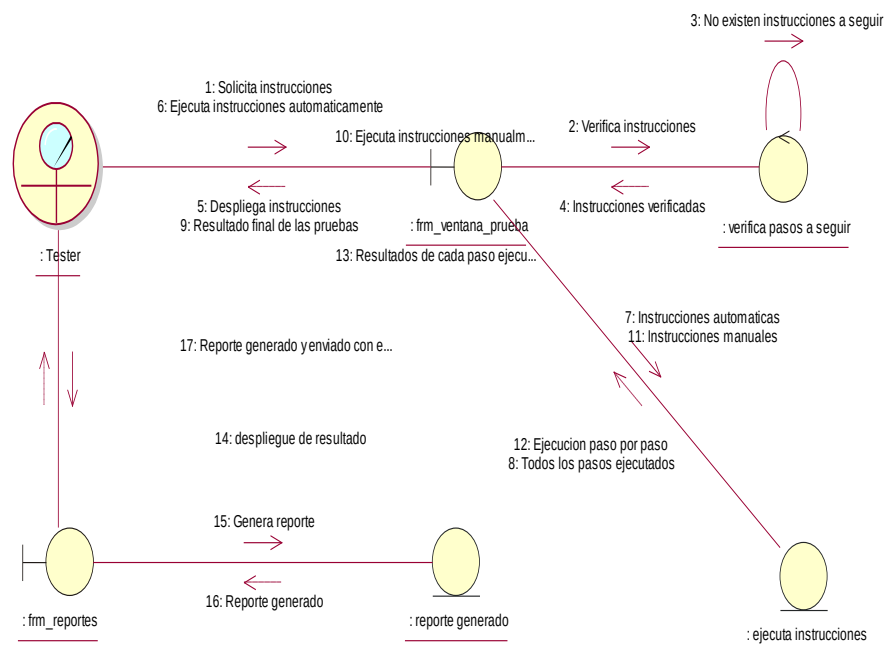


Figura 32: Diagrama de Colaboración Seguir pasos de ejecución

CAPÍTULO IV

4 Análisis de Alternativas

4.01 Matriz de Análisis de Alternativas

Tabla 32

Matriz de Análisis de Alternativas

OBJETIVOS	IMPACTO SOBRE EL PROPÓSITO	FACTIBILIDAD TÉCNICA	FACTIBILIDAD FINANCIERA	FACTIBILIDAD SOCIAL	FACTIBILIDAD POLÍTICA	TOTAL	CATEGORÍA
Agilizar el proceso de pruebas unitarias	5	4	4	4	2	19	ALTA
Aumentar el control de los productos desarrollados	5	4	3	3	2	17	ALTA
Garantizar la calidad del sistema auditado	5	4	4	2	1	16	MEDIA BAJA
Cumplir con los objetivos planteados al inicio del Sprint	4	4	4	4	3	19	ALTA
Generar reportes con información verídica	5	4	3	4	3	19	ALTA
Corregir con rapidez las incidencias que se presenten en las ventanas	4	3	3	3	2	15	ALTA
TOTAL	28	23	21	20	13	105	

NOTA: Matriz de análisis de alternativas muestra un análisis para determinar la viabilidad del proyecto.

Elaborado por: Lucía Caguano

4.02 Matriz de Impactos de Objetivos

Tabla 33

Matriz de Impactos de Objetivos

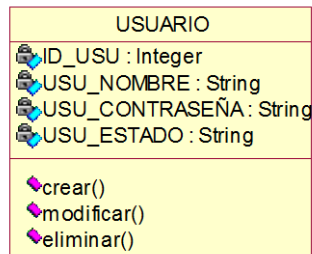
OBJETIVO	FACTIBILIDAD DE LOGRARSE	IMPACTO EN GÉNERO	IMPACTO AMBIENTAL	RELEVANCIA	SOSTENIBILIDAD
Agilizar el proceso de pruebas unitarias	Facilita la ejecución de las pruebas y también agiliza optimizando el tiempo	Será de fácil manejo por lo que cualquier usuario lo podrá manipular	El entorno social se ve mejorado	Responde a las expectativas de los usuarios involucrados	Agiliza el proceso de conocer el funcionamiento del sistema
Aumentar el control de los productos desarrollados	Conocer la funcionalidad de los productos en desarrollo y terminados	Será más fácil detectar las incidencias	Entregar un sistema de calidad	El producto o sistema saldrá a producción una vez terminadas las pruebas	Un cliente satisfecho del producto
Garantizar la calidad del sistema auditado	No se requieren conocimientos sofisticados para el uso del sistema	Es ideal para motivar al departamento de control de calidad		Protege el uso de los recursos	Fortalece la organización de la empresa
Cumplir con los objetivos planteados al inicio del Sprint	Cumplir los tiempos establecidos para la entrega de avances	Cumple con los objetivos establecidos	Tiempo mejor administrado	Cumplir con las exigencias de los clientes	El departamento de control de calidad es el beneficiario directo
Generar reportes con información verídica	Generar reportes en formato PDF	El departamento de control de calidad tendrá un informe inmediato del funcionamiento del sistema	Reducción de papel ya que no se realizarán manualmente las pruebas	El reporte se podrá abrir en cualquier dispositivo electrónico	Facilita la información a los usuarios
Corregir con rapidez las incidencias que se presenten en las ventanas	Corregir de manera rápida y adecuada las incidencias de los sistemas probados		Protege el uso de los recursos de la empresa	Corregir los procesos más importantes en el menor tiempo posible	Optimización del sistema con las correcciones

NOTA: Matriz de impactos de objetivos se describe los siguientes puntos Objetivo, Factibilidad de Lograrse, Impacto en género, Impacto ambiental, Relevancia y Sostenibilidad.

4.03 Estándares para el Diseño de Clases

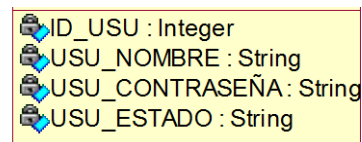
Clase

La clase es una unidad que contiene la información de un objeto.



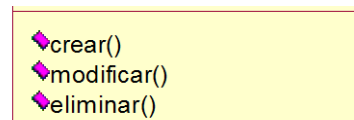
Atributos

Los atributos o características de una clase



Métodos

Especifica la acción que realiza un método



Asociaciones

- Uno a muchos 1..* (1..n)
- Cero a muchos 0..* (0..n)
- Número fijo m (m muestra el número)

Herencia (Generalización/Especialización)

Permite asociar objetos que contribuyen entre sí



4.04 Diagrama de Clases

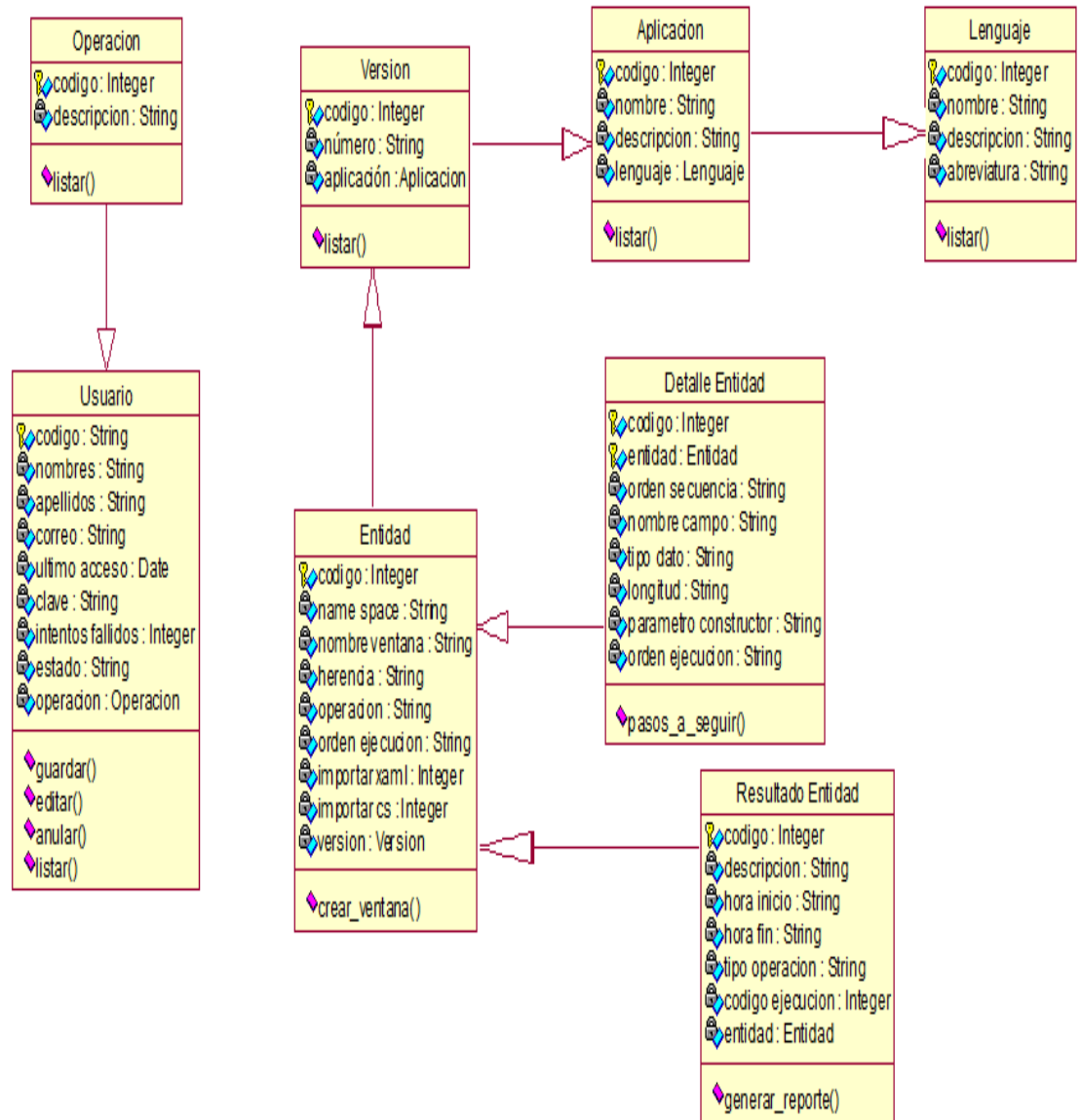


Figura 33: Diagrama de Clases

4.05 Modelo Lógico

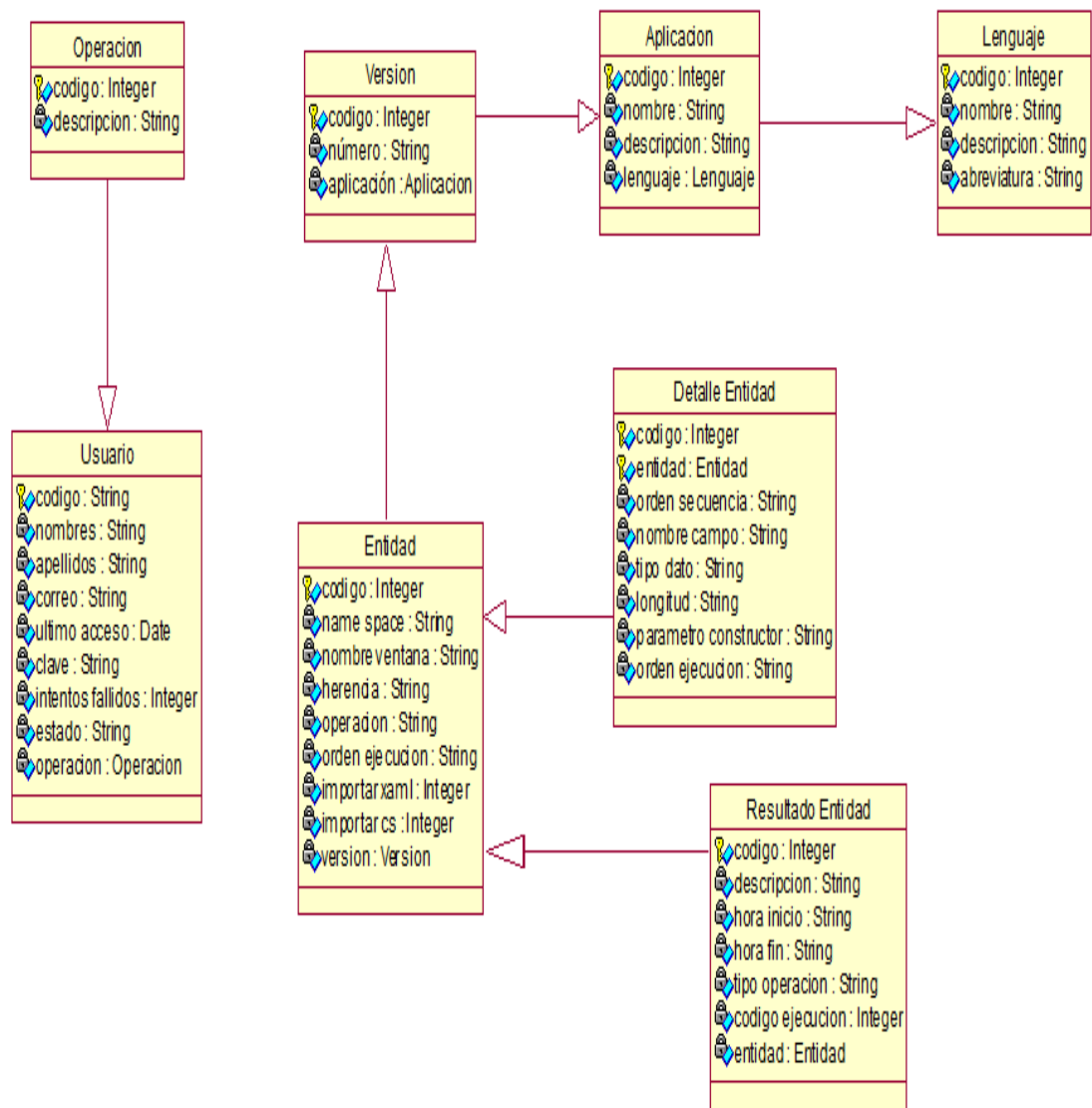


Figura 34: Modelo lógico

4.06 Modelo Físico de la Base de Datos

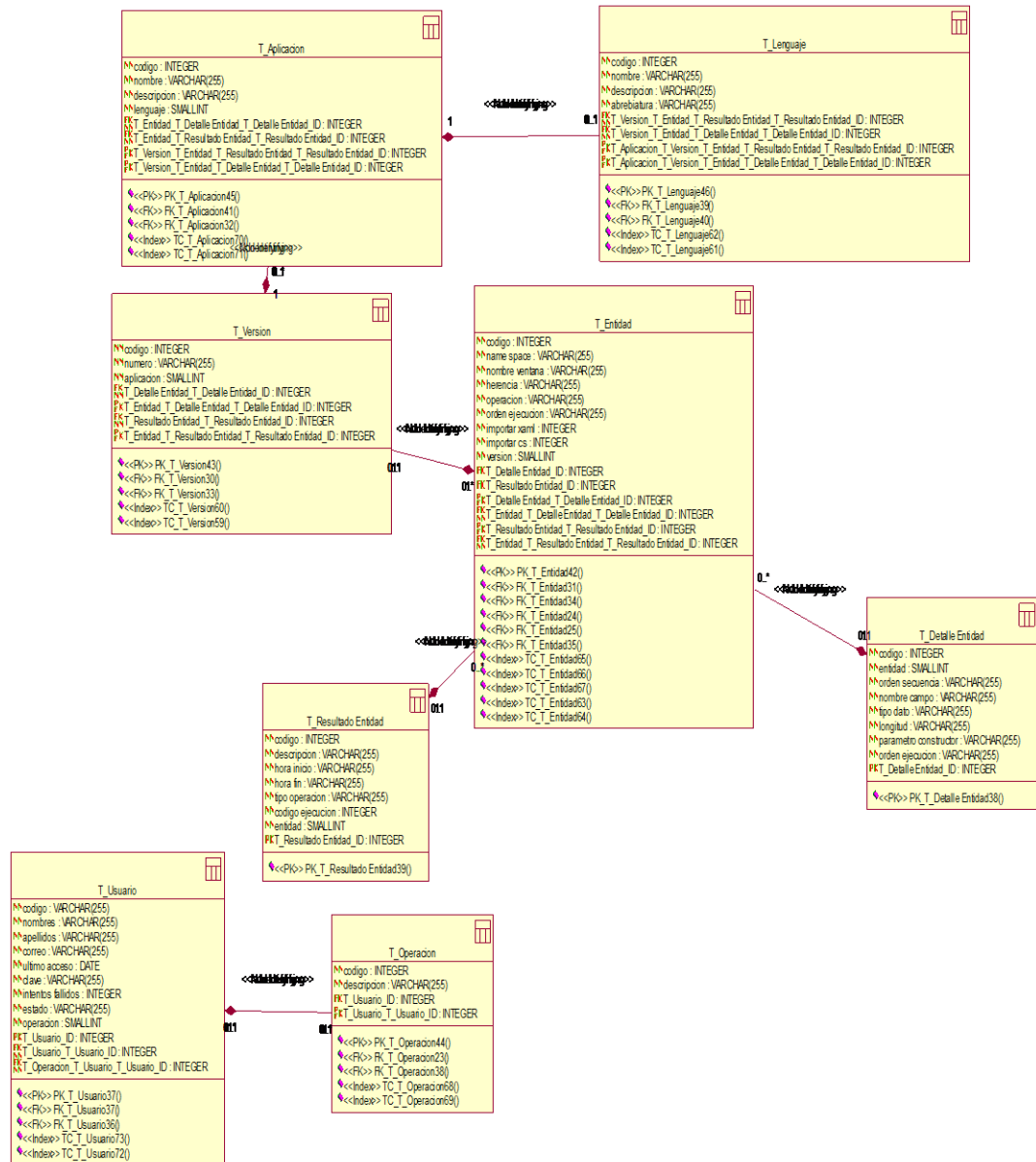


Figura 35: Diagrama del Modelo Físico de la Base de Datos

4.07 Diagrama de Componentes

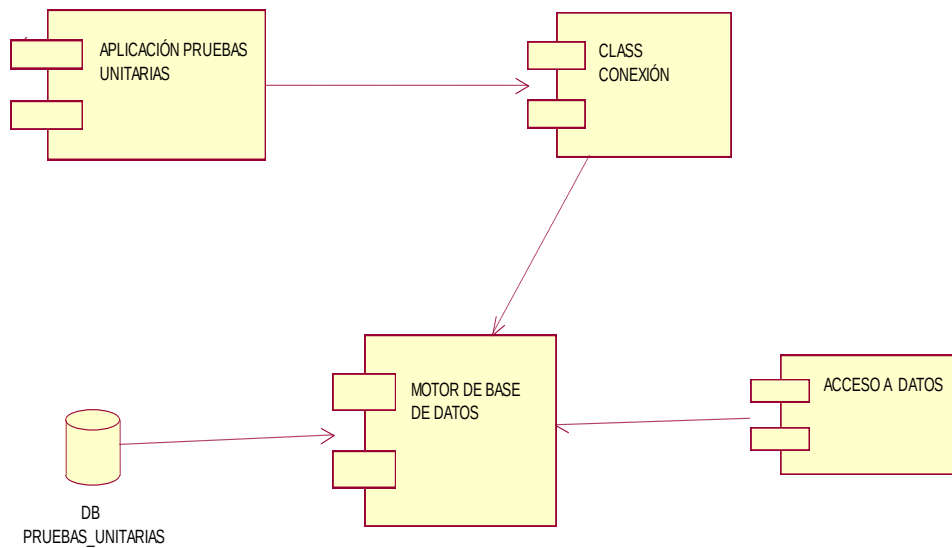


Figura 36: Diagrama de Componentes

4.08 Diagrama de Estrategias

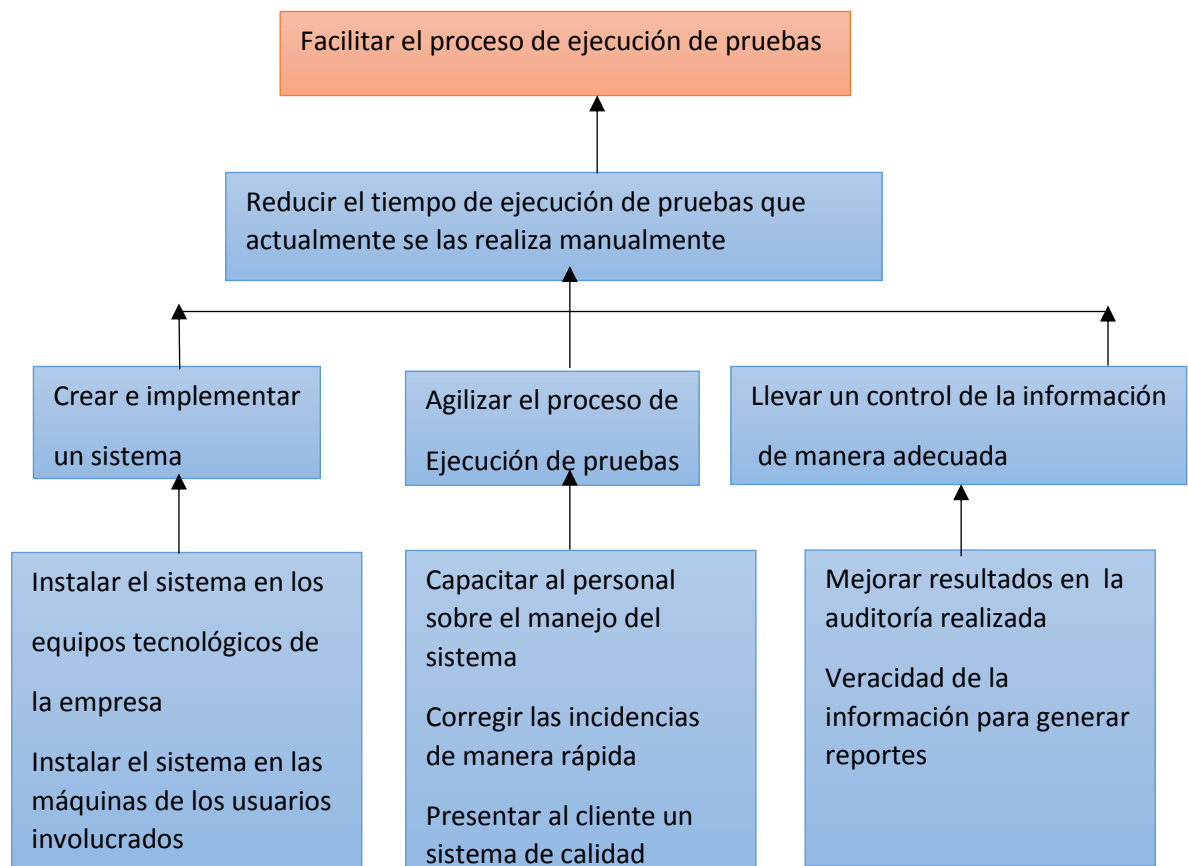


Figura 37: Diagrama de Estrategias

4.09 Matriz de Marco Lógico

Tabla 34

Matriz de Marco Lógico

FINALIDAD	INDICADORES	MEDIOS DE VERIFICACIÓN	SUPUESTO
FINALIDAD			
Sistematizar el proceso de pruebas unitarias	Confiabilidad de los resultados de las pruebas en un 90%	Evaluación al sistema como bueno, malo, regular	Mejorar el tiempo de ejecución de pruebas
PROPÓSITO			
Agilizar el proceso de pruebas unitarias a los sistemas	Reportes de las ejecuciones de las pruebas de los sistemas	Comparación de los reportes anteriores con los actuales	Visualización de mejoras al corregir incidencias
COMPONENTES			
Aplicativo con una interfaz amigable para los usuarios	Porcentaje de aceptación del sistema por parte de los usuarios	Análisis de los involucrados acerca de la efectividad del sistema	Interfaz no sea de agrado para todos los usuarios
ACTIVIDADES			
Conocer el proceso total que realiza el sistema	Obtener el resultado del funcionamiento del sistema	Analizar el cumplimiento del sistema en relación a los requerimientos y alcance del mismo	Verificar que puedan presentarse procesos que no fueron relevantes al momento de realizar el levantamiento de requisitos

NOTA: En esta tabla se muestra la matriz del marco lógico mostrando la finalidad, los indicadores, medios de verificación y supuesto

4.10 Vistas Arquitectónicas

4.01.01 Vista Lógica

• SERVIDOR

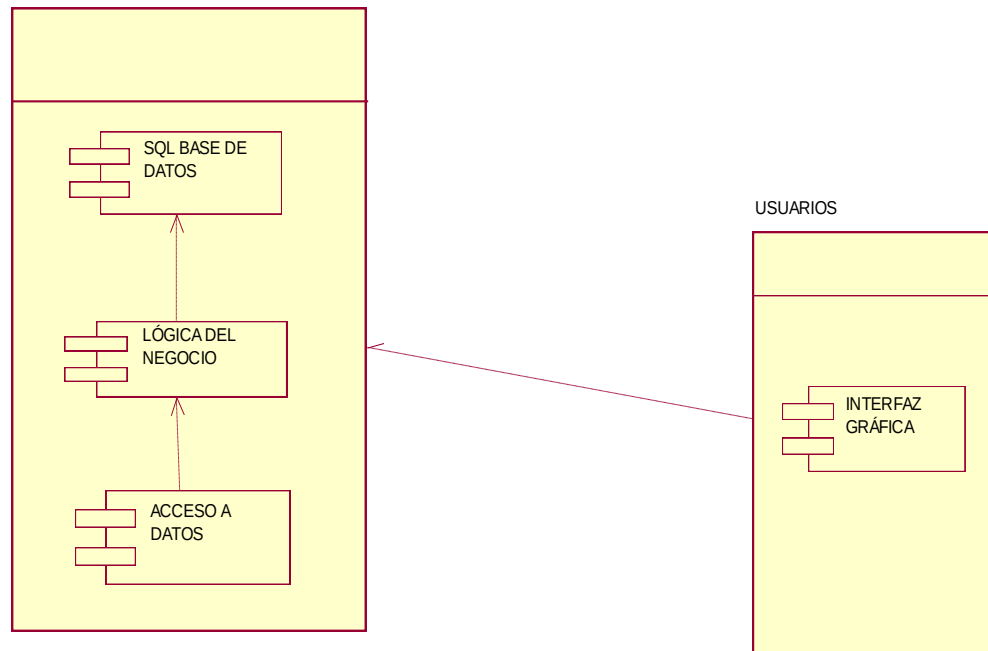


Figura 38: Vista lógica de la solución

4.10.02 Vista Física

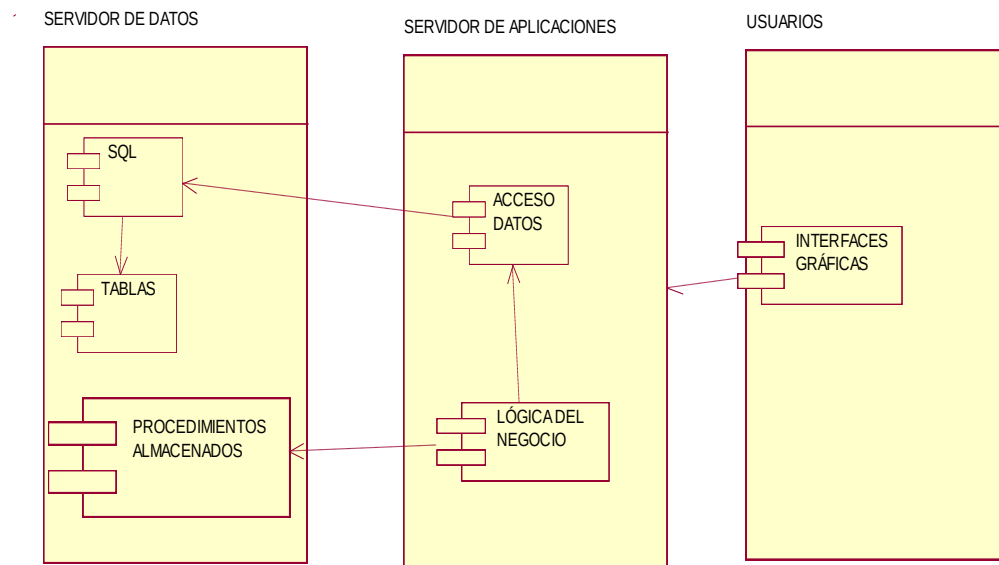


Figura 39: Vista física de la solución

4.10.03 Vista de Desarrollo

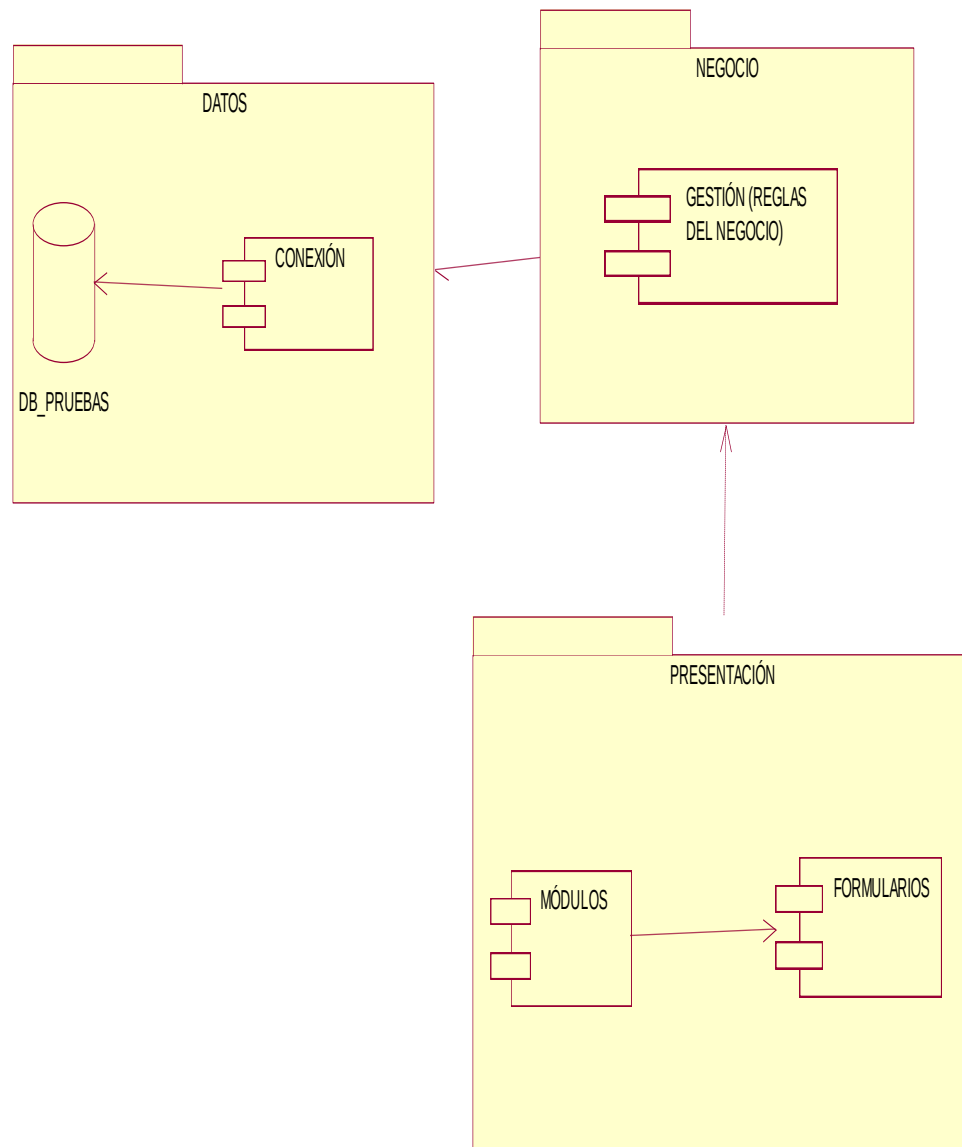


Figura 40: En este gráfico se pueden visualizar las capas con cada uno de sus componentes que conforman el sistema

4.10.04 Vista de Procesos

Inicio de Sesión (Login)

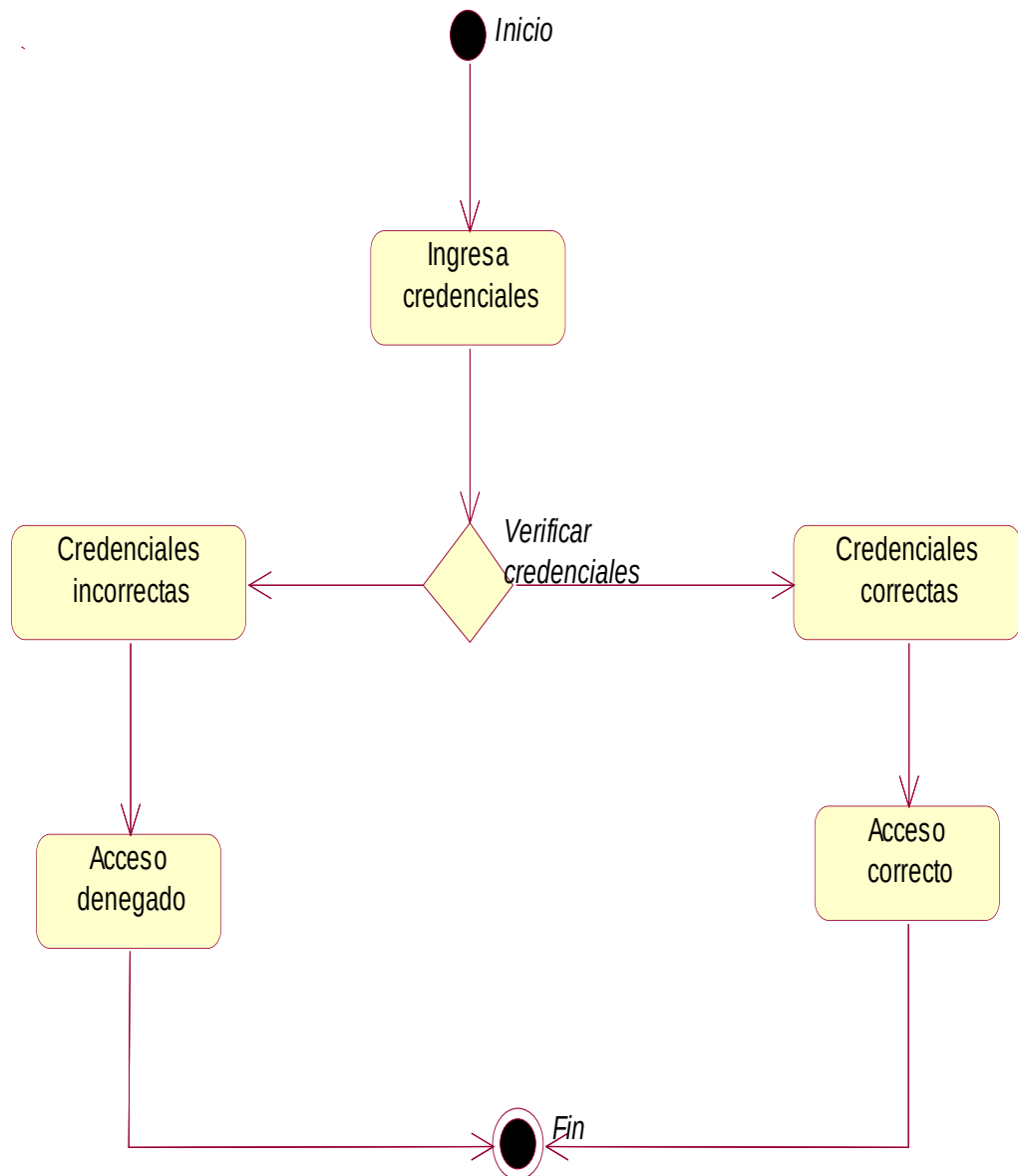


Figura 41: En este gráfico se puede visualizar el proceso que debe seguir un usuario para ingresar el sistema

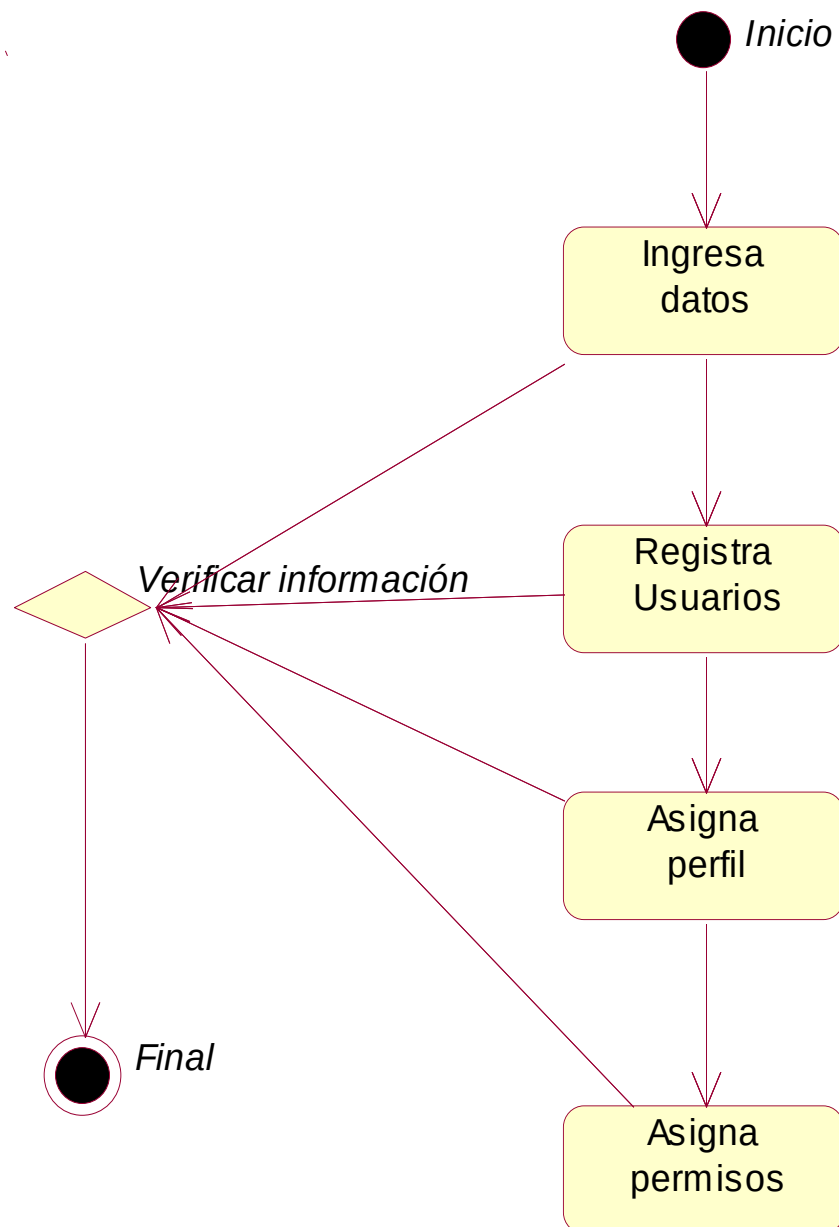
Administrador Registro de Usuarios

Figura 42: En este gráfico se pueden visualizar el proceso que sigue el Administrador para registrar los usuarios

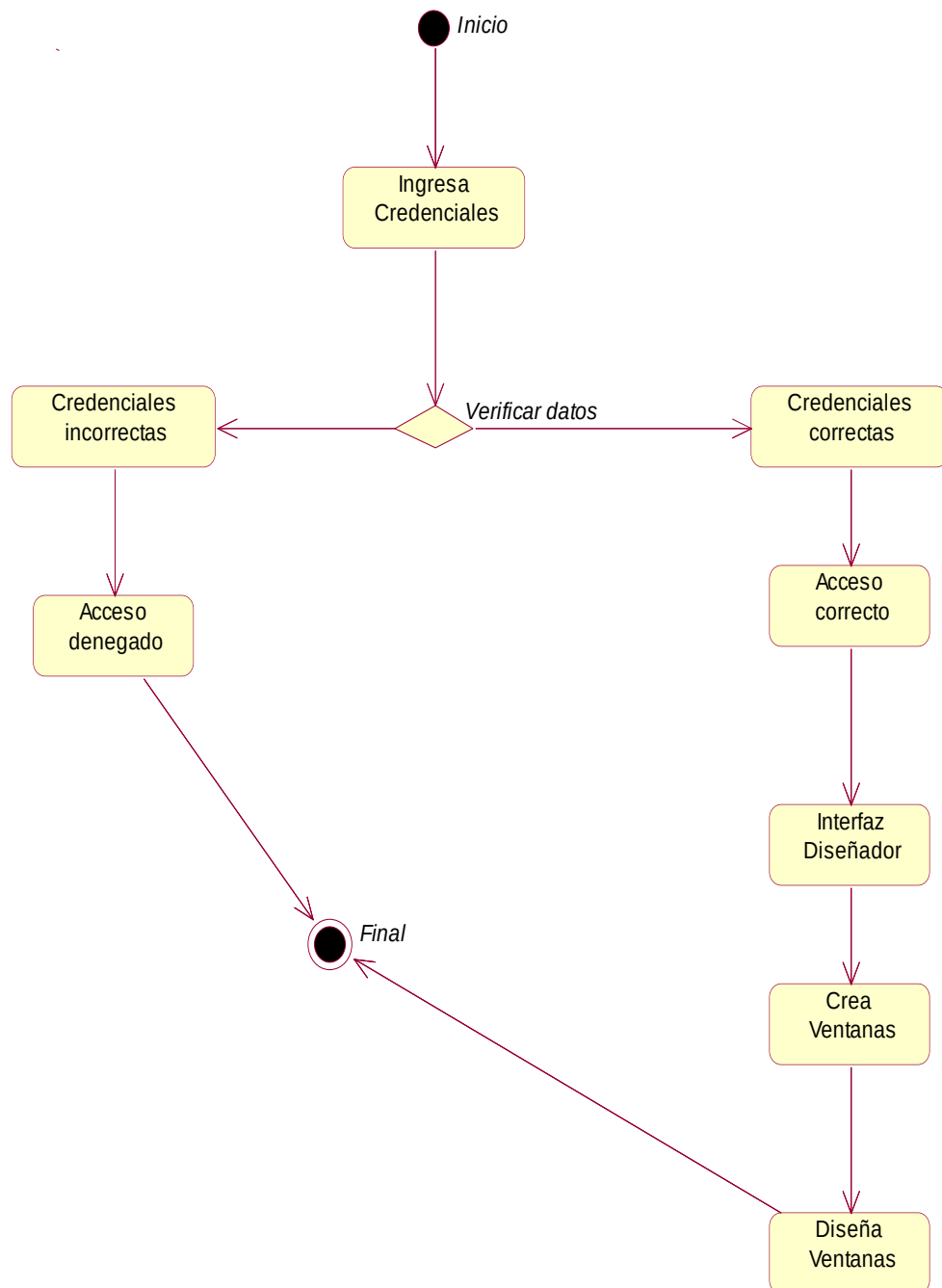
Diseñador Crear y Diseñar Ventanas

Figura 43: En este gráfico se pueden visualizar el proceso que sigue el diseñador para crear las páginas y posteriormente diseñarlas

Tester Ejecución de Pruebas

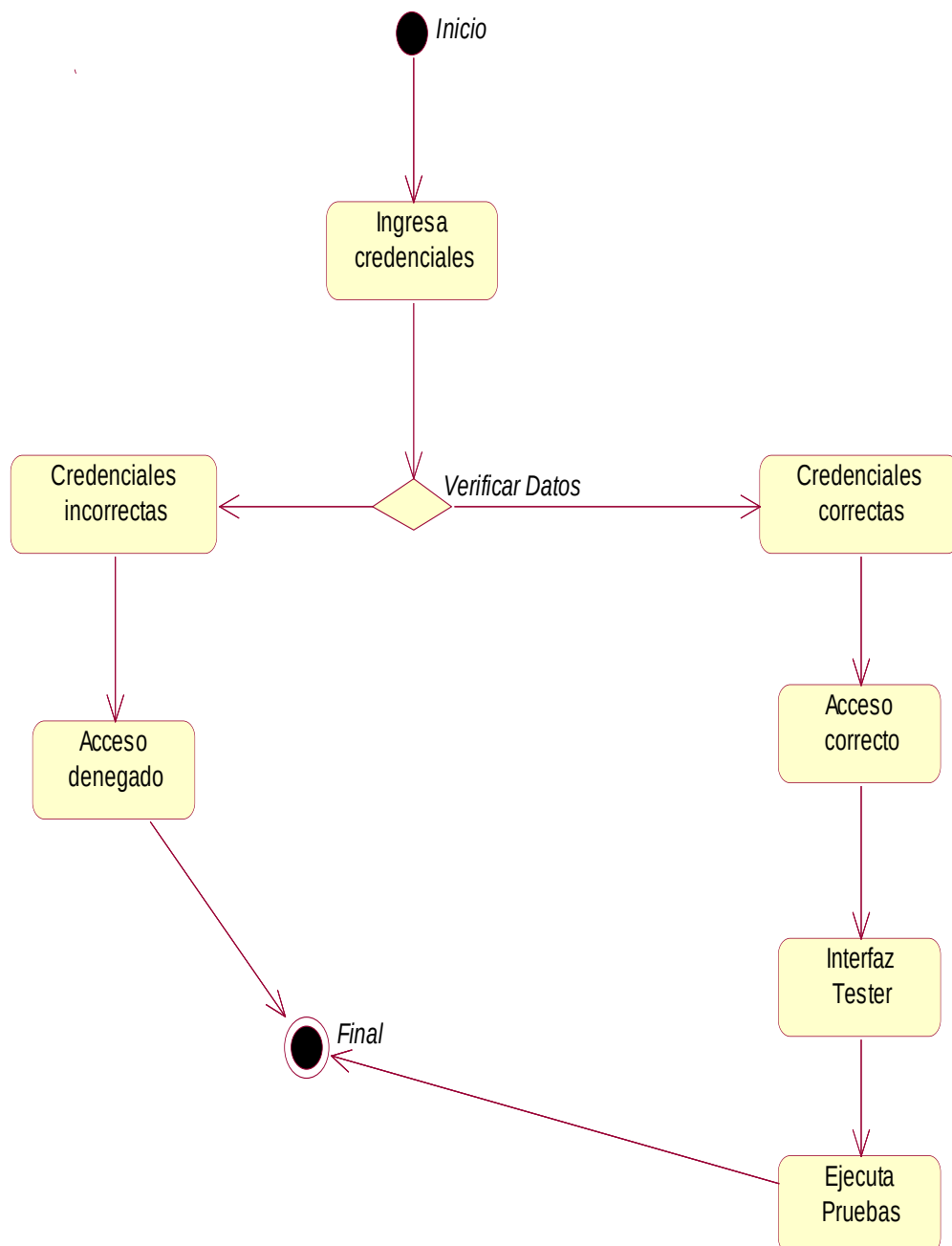


Figura 44: En este gráfico se pueden visualizar el proceso que sigue el usuario Tester para la ejecución de pruebas unitarias

CAPÍTULO V

5 Propuesta

5.01 Especificación de Estándares de Programación

Los estándares de programación son ayuda de referencia a cada una de las variables y a las herramientas que se utilizaron para el desarrollo del sistema.

Tabla 35

Estándares de programación

TIPO DE CONTROL	PREFIJO	EJEMPLO
TextBox	Txt	txtnombre
Grid	Grd	grdnombre
Label	Lbl	lblnombre
ListView	Lvw	lvwnombre
Button	btn	bttnombre
Image	img	imgnombre

TIPO DE VARIABLE	DESCRIPCIÓN
String	Cadena de caracteres
Int	Caracter entero
DateTime	Caracter de fecha y hora

NOTA: En esta tabla se puede visualizar los estándares de programación que se han utilizado en el desarrollo del sistema

5.02 Diseño de Interfaces de Usuario

La interfaz del sistema debe ser dinámica e intuitiva para el usuario.

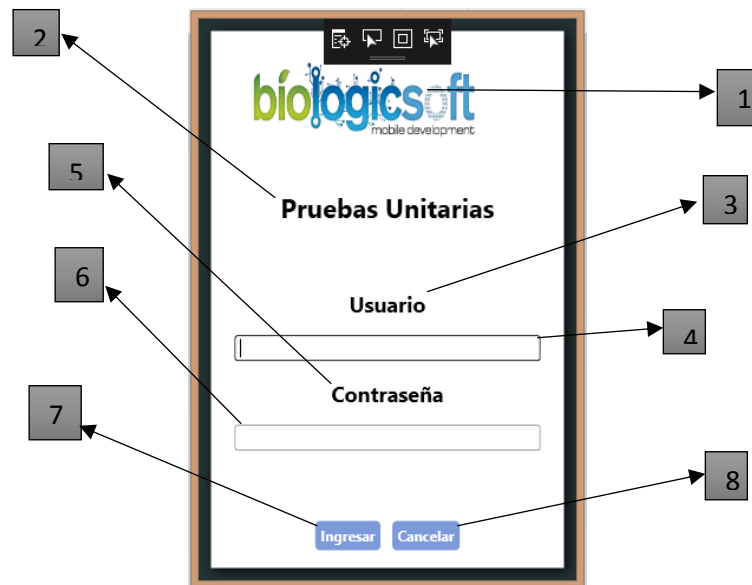


Figura 45: En este gráfico se pueden visualizar el login, en el cual se debe ingresar las credenciales de usuario para ingresar al sistema, si las credenciales están correctas permite el ingreso caso contrario le saldrá un mensaje indicando que la información no

Descripción de los componentes de la ventana Login

- 1.- Imagen.- Es el logo de la empresa Biologicsoft
- 2.- Label.- Que contiene el título de la ventana, Pruebas Unitarias
- 3.- Label.- Muestra que se debe ingresar el usuario
- 4.- TextBox.- Caja de texto, donde de ingresa el usuario, se lo denomina txtusuario
- 5.- Label.- Muestra que se debe ingresar la contraseña
- 6.- TextBox.- Caja de Texto, donde se debe ingresar la contraseña, se la denomina txtcontraseña
- 7.- Button.- Es el botón que permite ingresar al aplicativo, se lo llama btningresar
- 8.- Button.- Es el botón que cancela el ingreso al aplicativo, se lo llama btncancelar

Interfaz Usuario Administrador

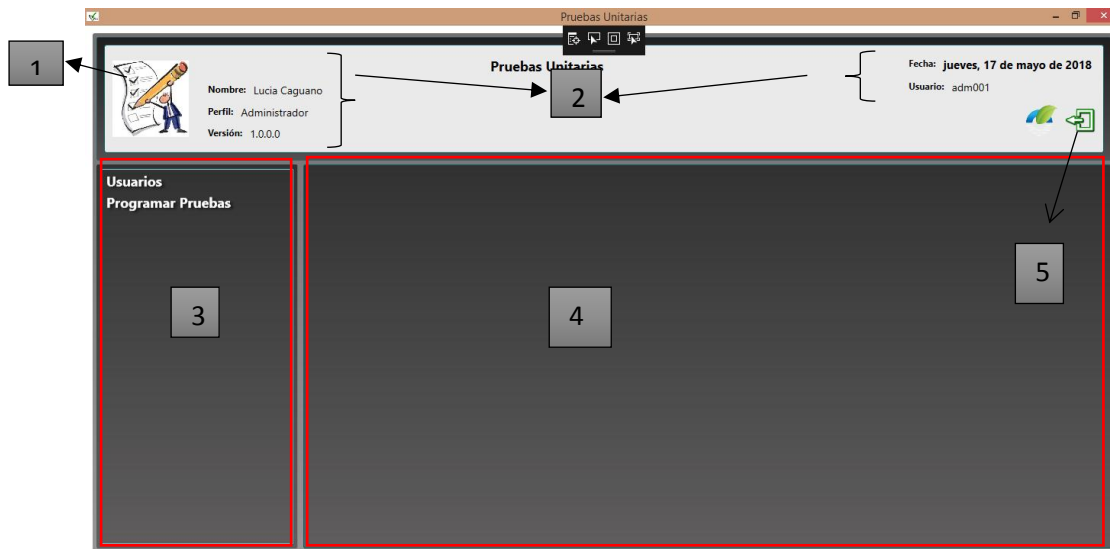


Figura 46: En este gráfico se pueden visualizar la interfaz del usuario Administrador

Descripción de los componentes de la interfaz del usuario Administrador

1.- Imagen.- Es la imagen de la presentación del sistema se encuentra denominada como `imgUsuario`

2.- Label.- Contienen los textos que se pueden visualizar en la interfaz se encuentran denominados así: `Lblinfo` – `Lblnombres` – `Lblperfil` – `Lblfecha` – `Lblusuario`

3.- ListView.- Es el campo donde se muestran las opciones que tiene el Administrador, se lo denomina `lvwListaOpciones`

4.- Content Control.- En un control de contenido donde se muestran todas las operaciones a realizar, se encuentra como `cncMain`

5.- Botón Salir.- Es el botón que se utiliza para salir de la aplicación

Creación de Usuarios

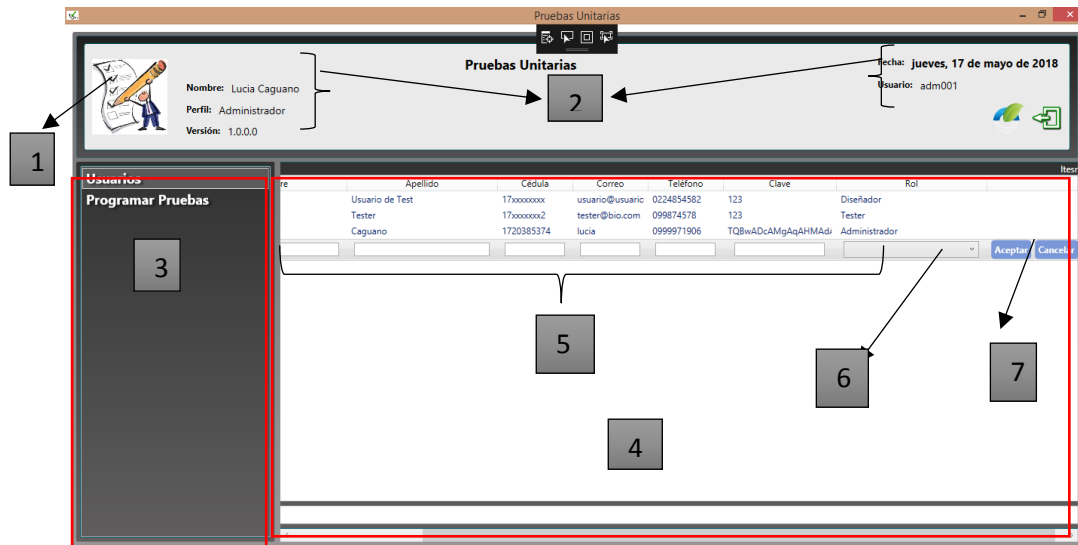


Figura 47: En este gráfico se pueden visualizar la interfaz del Administrador, donde crea los usuarios

Descripción de los componentes de la interfaz de creación de usuarios

- 1.- Imagen.- Muestra la imagen de la interfaz se denomina como img
- 2.- Label.- Son los campos de texto que se muestran en la interfaz se denomina con lbl antes de los nombres correspondientes
- 3.- ListView.- Es el campo que muestra las opciones del usuario y se muestra lvw
- 4.- Content Control.- Es el campo donde se muestran las operaciones realizadas se lo nombra como cnc
- 5.- Text Box.- Son cajas de texto que permiten ingresar información se denomina txt
- 6.- Combob Box.- Son cuadros desplegables que permiten seleccionar información se los denomina como cmb
- 7.- Button.- Son botones que cumplen con la función que se les asigna se denominan como btnnombre del campo o función

Interfaz del usuario diseñador

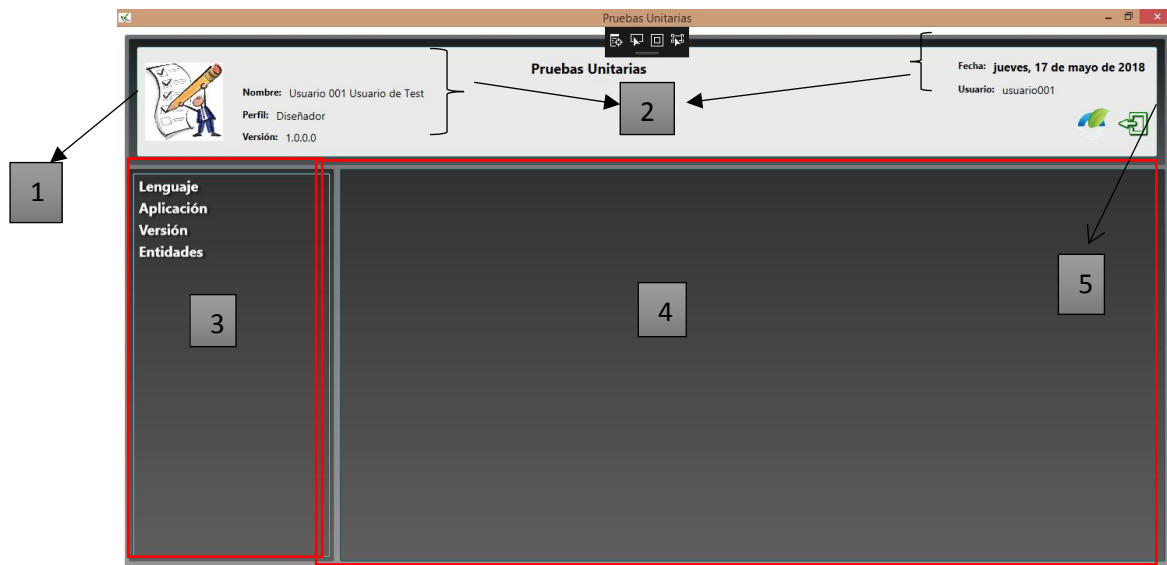


Figura 48: En este gráfico se pueden visualizar la interfaz del Diseñador

Descripción de los componentes de la interfaz del usuario Diseñador

1.- Imagen.- Es la imagen de la presentación del sistema se encuentra denominada como `imgUsuario`

2.- Label.- Contienen los textos que se pueden visualizar en la interfaz se encuentran denominados como `lbl`

3.- ListView.- Es el campo donde se muestran las opciones que tiene el Administrador, se lo denomina `lvw`

4.- Content Control.- En un control de contenido donde se muestran todas las operaciones a realizar, se encuentra como `cnc`

5.-Button.- Es el botón que se utiliza para salir del aplicativo (Cerrar Sesión), se denomina como `btnnombre del campo`

Opciones Usuario Diseñador

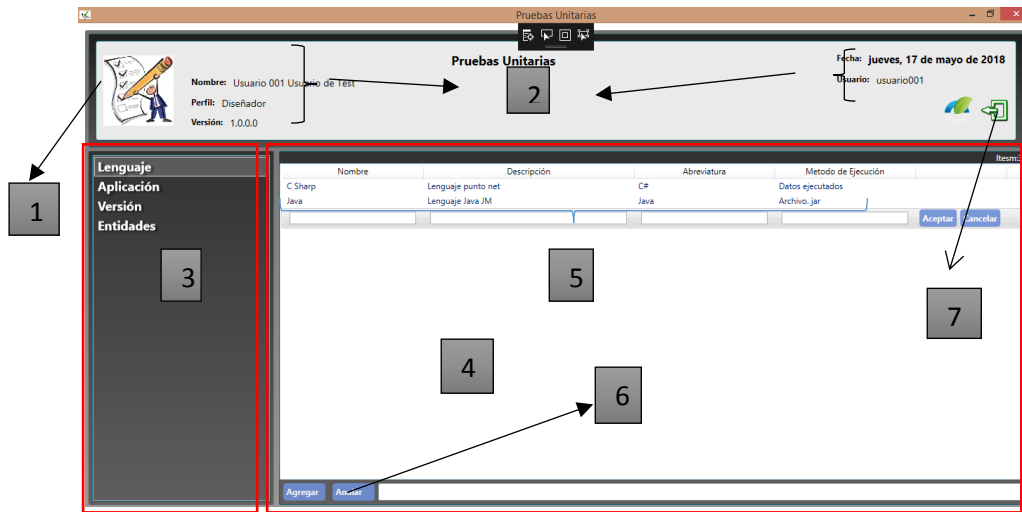


Figura 49: En este gráfico se pueden visualizar la interfaz del Diseñador donde agrega el lenguaje, versión, aplicación y entidades.

Descripción de los componentes de la interfaz del usuario Diseñador

- 1.- Imagen.- Se muestra en la interfaz y se denomina como img
- 2.- Label.- Son los textos que se muestran en la interfaz se muestran como lbl
- 3.- List View.- Es el campo donde se muestran las opciones y es lvw
- 4.- Content Control.- Es donde se muestran las operaciones realizadas y se denomina como ccn
- 5.- Text Box.- Son cajas de texto y se denominan como txt
- 6.- Button.- Son botones que cumplen una función y se muestran btn
- 7.- Button.- Es el botón que tiene la funcionalidad de cerrar la sesión y al presionar este botón se regresa a la ventana de inicio o Login

Creación de Entidades

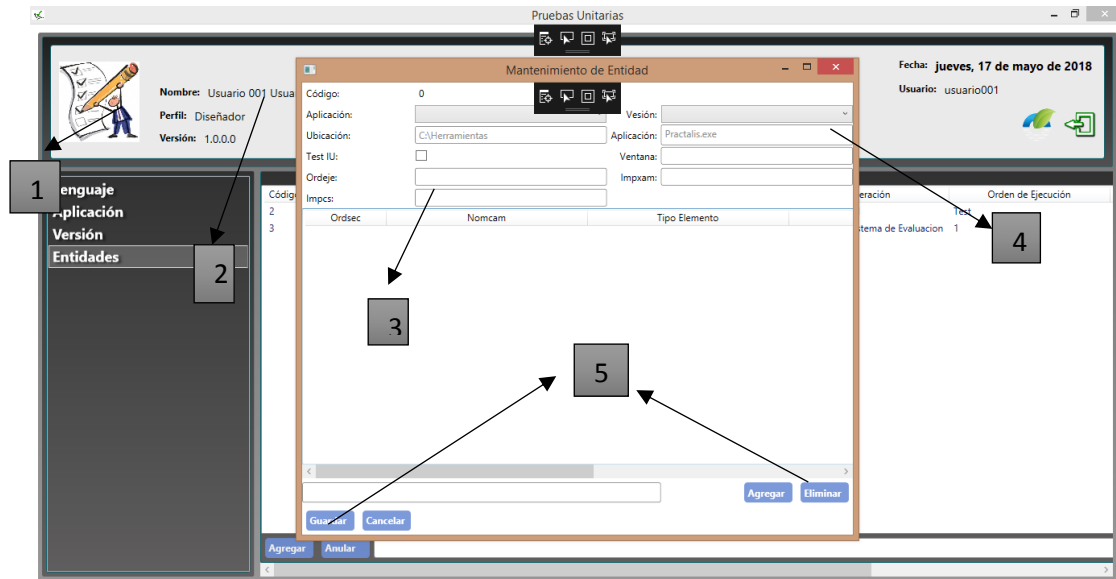


Figura 50: En este gráfico se pueden visualizar la interfaz del Diseñador donde crea las ventanas

Descripción de los componentes de la creación de entidades

- 1.- Imagen.- Se muestra en la interfaz y se denomina como img
- 2.- Label.- Son los textos que se muestran en la pantalla y se los denomina como lbl seguido un nombre
- 3.- Text Box.- Son cajas de texto para el ingreso de información y se las denomina como txt seguido del nombre correspondiente
- 4.- Combo Box.- Son cuadros desplegables que muestran información y permiten seleccionar se denomina como cmb
- 5.- Button.- Son botones que cumplen una acción requerida por el usuario y se denomina como btn seguido de un nombre

Resultados de la ejecución

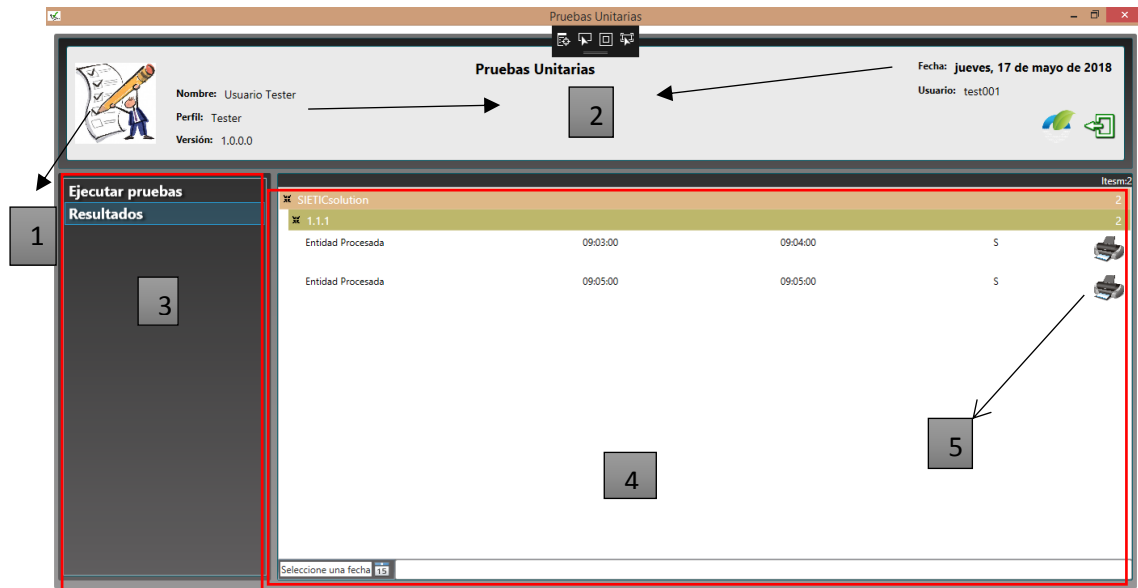
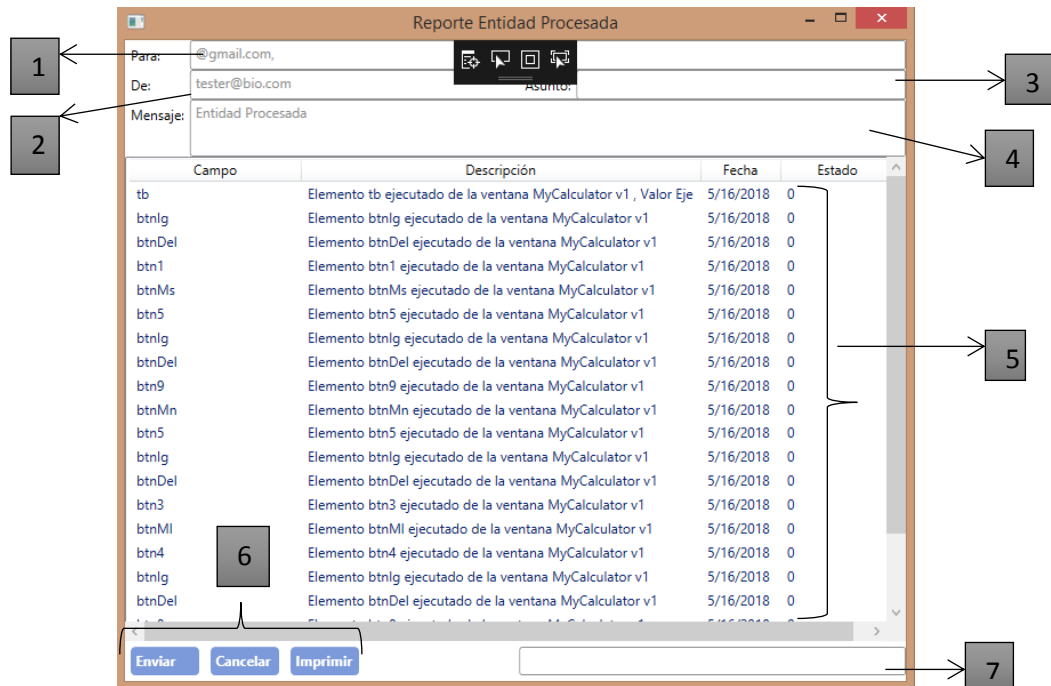


Figura 51: En este gráfico se pueden visualizar la interfaz del Tester

Descripción de los componentes de la interfaz de tester

- 1.-Imagen.- Es la imagen que se muestra en la interfaz del usuario Tester y se la denomina como img
- 2.-Label.- Son todos los textos que se muestran en la interfaz del Usuario y se denomina como lbl
- 3.-List View.- Es el campo donde se muestran todas las opciones del usuario y se denomina como lvw
- 4.-Content Control.- Es el campo donde se muestran todas las operaciones que el usuario realizar y se denomina como cnc
- 5.- Imagen.- Es la imagen de una impresora que al presionarla se muestra una ventana para el envío del correo o el reporte

Ventana de envío y generación de reportes



The screenshot shows a window titled "Reporte Entidad Procesada". It contains a form for sending an email and a table of test results. Numbered callouts point to specific components:

- 1: "Para:" text box (recipient email)
- 2: "De:" text box (sender email)
- 3: "Asunto:" text box (subject)
- 4: "Mensaje:" text box (message body)
- 5: The table of test results
- 6: "Enviar", "Cancelar", and "Imprimir" buttons
- 7: A search or filter input field at the bottom right

Campo	Descripción	Fecha	Estado
tb	Elemento tb ejecutado de la ventana MyCalculator v1 , Valor Eje	5/16/2018	0
btnlg	Elemento btnlg ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnDel	Elemento btnDel ejecutado de la ventana MyCalculator v1	5/16/2018	0
btn1	Elemento btn1 ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnMs	Elemento btnMs ejecutado de la ventana MyCalculator v1	5/16/2018	0
btn5	Elemento btn5 ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnlg	Elemento btnlg ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnDel	Elemento btnDel ejecutado de la ventana MyCalculator v1	5/16/2018	0
btn9	Elemento btn9 ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnMn	Elemento btnMn ejecutado de la ventana MyCalculator v1	5/16/2018	0
btn5	Elemento btn5 ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnlg	Elemento btnlg ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnDel	Elemento btnDel ejecutado de la ventana MyCalculator v1	5/16/2018	0
btn3	Elemento btn3 ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnMI	Elemento btnMI ejecutado de la ventana MyCalculator v1	5/16/2018	0
btn4	Elemento btn4 ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnlg	Elemento btnlg ejecutado de la ventana MyCalculator v1	5/16/2018	0
btnDel	Elemento btnDel ejecutado de la ventana MyCalculator v1	5/16/2018	0

Figura 52: En esta figura se puede observar la ventana de envío y generación de un reporte

Descripción de los componentes de la ventana de reportes y envío

- 1.- Text Box.- Es el campo donde se ingresará la dirección del destinatario, a quién va dirigido el correo
- 2.- Text Box.- Es el campo donde se ingresará la dirección de correo electrónico de la persona emisora
- 3.- Text Box.- Es el campo donde se ingresará el asunto del envío del correo
- 4.- Text Box.- Es el campo donde se ingresa un pequeño mensaje que llegará a receptor del correo
- 5.- Reporte.- Muestra cada uno de los campos probados con la fecha de la evaluación y el estado de si pasó o no
- 6.- Button.- Son botones que cumplen funciones como Enviar, Cancelar o Imprimir

7.- Filtro.- Es un filtro donde al agregar un criterio de búsqueda este trae los registros que contengan el mismo.

Reporte

Resultado			
Campo	Descripción	Fecha	Estado
tb	Elemento tb ejecutado de la ventana MyCalculator v1 , Valor Ejecutado 1+7	16/05/2018 11:51:13	0
btnlg	Elemento btnlg ejecutado de la ventana MyCalculator v1	16/05/2018 11:51:14	0
btnDel	Elemento btnDel ejecutado de la ventana MyCalculator v1	16/05/2018 11:51:14	0
btn1	Elemento btn1 ejecutado de la ventana MyCalculator v1	16/05/2018 11:51:14	0
btnMs	Elemento btnMs ejecutado de la	16/05/2018 11:51:14	0

Figura 53: Esta figura muestra el reporte en formato PDF

En esta Figura se puede visualizar el reporte que se genera al presionar el botón Imprimir, este documento se genera en formato PDF, y se despliega un mensaje indicando la ubicación del archivo

5.03 Especificación de pruebas de unidad

Tabla 36

Ingreso al Sistema

Identificador	PU001
de la prueba:	
Método a	Ingreso al sistema
Probar:	
Objetivo de la	Verificar que se encuentren validados los campos
Prueba:	
Datos de Entrada	
Datos de usuario como son Nombre de usuario y contraseña	
Resultado Esperado	
El sistema debe validar que el usuario se encuentre registrado caso contrario no debe permitir el acceso al sistema	
Comentarios	
Debe ingresar a la interfaz que le corresponde	

NOTA: En esta tabla se muestra la prueba unitaria del ingreso al sistema (Login)

Tabla 37

Registro de usuarios

Identificador	PU002
de la prueba:	
Método a	Registro de usuarios
Probar:	
Objetivo de la	Verificar que los campos se encuentren validados
Prueba:	
Datos de Entrada	
Información de los usuarios como nombre, apellido, cédula, nombre de usuario, contraseña	
Resultado Esperado	
Registro exitoso de la prueba	
Comentarios	
Los campos requeridos deben estar necesariamente con datos	

NOTA: En esta tabla se muestra la prueba unitaria del registro de usuarios

Tabla 38

Validación de la cédula

Identificador	PU003
de la prueba:	
Método a	Validación de la cédula
Probar:	
Objetivo de la	Verificar que el número de identificación se encuentre validado
Prueba:	
Datos de Entrada	
	Número de cédula
Resultado Esperado	
	Que valide que solo sean 10 números
	Que el campo permita el ingreso solo de números
	Que el número de cédula sea válido
Comentarios	
	Si el número de cédula no es un número válido no debe permitir guardar el registro

NOTA: En esta tabla se muestra la prueba unitaria de la validación del documento de identificación

Tabla 39

Redundancia de Datos

Identificador	PU004
de la prueba:	
Método a	Validación de la redundancia de datos
Probar:	
Objetivo de la	Demostrar que no permita ingresar dos veces o más un mismo nombre de usuario
Prueba:	
Datos de Entrada	
	Nombre de usuario
Resultado Esperado	
	Si el registro de usuario ya existe que muestre un mensaje indicando que el usuario ya existe caso contrario que deje crear un nuevo registro de usuario
Comentarios	
	Que no existan registros duplicados

NOTA: En esta tabla se muestra la prueba unitaria de la validación de la redundancia de datos

Tabla 40

Validación de campos numéricos y de letras

Identificador	PU005
de la prueba:	
Método a	Validación de campos numéricos y de letras
Probar:	
Objetivo de la Prueba:	Demostrar que los campos permitan el ingreso solo de números o de letras según sea el caso
Datos de Entrada	
	Ingresar números
	Ingresar Letras
Resultado Esperado	
	Que se muestre un mensaje indicando que solo se acepta números o solo se acepta letras según sea el caso
Comentarios	
	Que no permita el ingreso de letras o números según sea el caso, que no haya información errónea

NOTA: En esta tabla se muestra la prueba unitaria de la validación de campos numéricos y de letras

Tabla 41

Validación de filtros de búsqueda

Identificador	PU006
de la prueba:	
Método a	Validación de filtros de búsqueda
Probar:	
Objetivo de la Prueba:	Demostrar que el filtro de búsqueda muestre la información correcta
Datos de Entrada	
	Criterio de búsqueda de acuerdo al registro o dato que requiera encontrar
Resultado Esperado	
	Registros que contengan el criterio de búsqueda ingresado
Comentarios	
	Tenga diferentes criterios de búsqueda

NOTA: En esta tabla se muestra la prueba unitaria del filtro de búsqueda, que busque con los criterios que necesita el usuario final

Tabla 42

Validación de botones

Identificador	PU007
de la prueba:	
Método a	Validación de botones
Probar:	
Objetivo de la	Demostrar que los botones cumplen las funciones que se les asignó y que
Prueba:	muestran en su interfaz
Datos de Entrada	
	Ingresar la información y presionar los botones necesarios
	Presionar los botones sin ingresar información
	Presionar los botones con información incompleta
Resultado Esperado	
	Que se realice la acción que muestra el botón
	Que los campos se muestren validados al presionar el botón sin ingresar información, que no deje guardar la información
	Que verifique si hay información incompleta y no permita guardar el registro
Comentarios	
	Solo se realizará la acción una vez que se presione el botón
	Los botones realizarán la acción que se muestra en su interfaz
NOTA: En esta tabla se muestra la prueba unitaria de la validación de botones	

5.04 Pruebas de Aceptación

Las pruebas de aceptación se realizan con la finalidad de valorar o probar el funcionamiento del sistema, es decir, que el sistema cumpla con los requerimientos solicitados por el usuario.

Tabla 43

Prueba de Aceptación de registro de usuarios

Identificador de la prueba	PA001
Caso de uso	UC001
Tipo de Usuario	Administrador
Objetivo de la prueba	Registrar la información de los usuarios
Secuencia de eventos	

Verificar que no permita crear dos usuarios con el mismo nombre de usuario

Verificar que la contraseña contenga mínimo 8 y máximo 12 caracteres, y contenga una letra mayúscula, una minúscula, un número y un carácter especial.

Que el correo electrónico y el documento de identificación sean válidos

Resultados esperados

Registro exitoso de usuarios

Validación de nombre de usuario

El usuario interactue con el perfil que se le asignó

Comentarios

Para registrar un usuario se va a validar que el correo electrónico y el número de cédula se encuentren validados, en caso de ser un documento de identificación o un correo electrónico no válidos no permite guardar la información, de igual forma la contraseña que se le asigna debe ser mínimo de 8 y máximo de 12 caracteres.

Estado: Pasado

NOTA: En esta tabla se muestra la prueba de aceptación por parte del usuario al registrar la información de los usuarios

Tabla 44

Prueba de Aceptación de asignación de permisos

Identificador de la prueba	PA002
Caso de uso	UC002
Tipo de Usuario	Administrador
Objetivo de la prueba	Al crear los usuarios el usuario administrador, le asigna el rol y con ello los permisos
Secuencia de eventos	El Administrador crea el usuario En la creación de un usuario el Administrador le asigna el rol y con ello se le asignan los permisos a los nuevos usuarios.
Resultados esperados	Los Usuarios solo pueden ingresar a la interfaz de acuerdo al perfil que el Administrador les asignó, ya sea como Diseñador o Tester. Los usuarios diseñadores, pueden crear, modificar, eliminar y activar un registro de Lenguaje, Versión, Aplicación y Entidades. Los usuarios Tester solo tiene acceso a las ejecuciones de pruebas y generación de reportes
Comentarios	Solo pueden realizar acciones de acuerdo a sus permisos de usuario. Administrador (Crear y modifica usuarios), Diseñador (Crear y Diseña Lenguajes, Versiones, Aplicaciones y Entidades) Tester(Ejecuta pruebas a las ventanas diseñadas, Genera reportes y envía correos a los involucrados en el desarrollo del sistema en evaluación)
Estado:	Pasado

NOTA: En esta tabla se muestra la prueba de aceptación sobre los permisos de usuario

Tabla 45

Prueba de Aceptación de creación de ventanas

Identificador de la prueba	PA003
Caso de uso	UC003
Tipo de Usuario	Diseñador
Objetivo de la prueba	Crear ventanas para pruebas
Secuencia de eventos	El usuario diseñador debe crear el lenguaje El usuario diseñador debe crear una versión El usuario diseñador debe crear una aplicación Una vez que los demás componentes se encuentren creados se procede a crear la ventana para la evaluación
Resultados esperados	Que los registros de lenguaje se muestren Que los registros de versión de muestren Que los registros de aplicación se muestren Que las entidades se muestren correctamente creadas
Comentarios	El nombre de las ventanas no pueden estar repetidas, son entidades únicas mientras estén dentro de una misma aplicación
Estado:	Pasado

NOTA: En esta tabla se muestra la prueba de aceptación de la creación de las ventanas

Tabla 46

Prueba de Aceptación de creación de pasos a seguir

Identificador de la prueba	PA004
Caso de uso	UC004
Tipo de Usuario	Diseñador
Objetivo de la prueba	Crear pasos o campos de las ventanas
Secuencia de eventos	El usuario diseñador debe crear el nombre de la entidad El usuario diseñador debe ingresar la ubicación del ejecutable El usuario diseñador debe ingresar los campos que contiene la ventana que se va a evaluar
Resultados esperados	Que se muestren las ventanas que el usuario diseñador creó y diseñó Que se encuentren correctas las direcciones donde se encuentran los ejecutables Que se encuentren todos los campos de la ventana a evaluar
Comentarios	Las ventanas no pueden estar vacías Si la dirección del ejecutable no está correcto, la aplicación no se levantará Se debe mostrar todas las entidades creadas
Estado:	Pasado

NOTA: En esta tabla se muestra la prueba de aceptación de la creación de los campos de las ventanas por parte del usuario diseñador

Tabla 47

Prueba de Aceptación de ejecución de pruebas (ejecutar ventana)

Identificador de la prueba	PA005
Caso de uso	UC005
Tipo de Usuario	Tester
Objetivo de la prueba	Realizar la ejecución de las ventanas
Secuencia de eventos	El usuario Tester ingresa con sus credenciales a su respectiva interfaz Selecciona la opción de Ejecutar Pruebas Seleccionar la ventana que va a ejecutar Presionar el botón de ejecución de pruebas
Resultados esperados	Que se levante una ventana emergente que muestre si desea ejecutar la prueba seleccionada Que se ejecute la ventana con los parámetros que se ingresó en la creación Que al final se muestre el resultado que indique si la ventana pasó o no la evaluación
Comentarios	Solo permitirá ejecutar las ventanas que se encuentren diseñadas El usuario tester no puede editar las ventanas creadas El usuario tester no puede cambiar los resultados que arroja el sistema
Estado:	Pasado

NOTA: En esta tabla se muestra la prueba de aceptación de la realización de ejecución de pruebas por parte del usuario tester

Tabla 48

Prueba de Aceptación de seguir pasos de ejecución

Identificador de la prueba	PA006
Caso de uso	UC006
Tipo de Usuario	Tester
Objetivo de la prueba	Sigue los pasos par al ejecución de las ventanas para comprobar el efectivo comportamiento del sistema
Secuencia de eventos	Selecciona la opción ejecutar pruebas Seleccionar o filtra la ventana o entidad que se va a evaluar Seleccionar el botón play que realiza la acción de ejecución Acepta la ejecución la ventana seleccionada
Resultados esperados	Ejecución correctas de las ventanas Ingreso correcto de información
Comentarios	Solo se ejecutan las ventanas diseñadas
Estado:	Pasado

NOTA: En esta tabla se muestra la prueba de aceptación de la ejecución de las ventanas siguiendo los pasos

Tabla 49

Prueba de Aceptación de generar reportes

Identificador de la prueba	PA007
Caso de uso	UC007
Tipo de Usuario	Tester
Objetivo de la prueba	Genera reportes
Secuencia de eventos	Una vez que ejecuta las pruebas seleccionar el ícono de impresora para la generación de los reportes Se levanta una ventana de envío de reportes y presionar el botón imprimir Verificar que se muestre la ubicación de donde se va a descargar el documento Verificar la ubicación y verificar el documento, que se haya generado en formato PDF
Resultados esperados	Reporte en formato pdf Que muestre correctamente la ubicación del documento Que el reporte contenga todos los campos que tiene la entidad
Comentarios	Solo emitirá reportes una vez que se ejecuten las pruebas, caso contrario el reporte se mostrará vacío El reporte se despliega únicamente en formato pdf
Estado:	Pasado

NOTA: En esta tabla se muestra la prueba de aceptación de la generación de los reportes por parte del usuario Tester

5.05 Especificación de pruebas de carga

Las pruebas de carga son las encargadas de evaluar al sistema en diferentes condiciones, se refiere a ingresar datos reales y en gran cantidad para verificar que el sistema soporte de manera adecuada.

Tabla 50

Pruebas de Carga

Identificador de la prueba	PC001
Tipo de Prueba	Carga del Sistema
Objetivo de la Prueba	El objetivo de las pruebas de carga son probar el sistema con una gran cantidad de información y con datos reales para verificar el funcionamiento correcto del sistema
Descripción	Pruebas a cada una de las ventanas Ingreso de información (Gran cantidad de información) Ingreso de información errónea
Resultados Esperados	Que el sistema funcione correctamente con la información ingresada Que el sistema no colapse por la gran información ingresada Que los campos del sistema se encuentren validados Que el sistema muestre la interfaz del perfil correspondiente, al ingreso de las credenciales
Comentarios	El sistema soportó de manera positiva la carga

NOTA: En esta tabla se muestra la prueba de carga que se realizó al sistema con los resultados correspondientes

5.06 Configuración del Ambiente Mínima Ideal

Los requerimientos mínimos que necesita el sistema para operar sin inconvenientes son los siguientes:

Tabla 51

Ambiente Mínimo Ideal

AMBIENTE MÍNIMO IDEAL
Requerimientos de Hardware
Equipo de tipo servidor
8GB en RAM
Requerimientos de Software
Sistema Operativo Windows 7 en adelante
Visual Studio 2017 con sus componentes
Motor de Base de dato SQL Server 2016
Adobe Acrobat Reader 7 o superior
NOTA: En esta tabla se muestra un listado de los requisitos mínimos para el funcionamiento correcto del sistema

CAPÍTULO VI

6 Aspectos Administrativos

6.01 Recursos

Dentro de los recursos se debe considerar Recurso Humano, Recurso Lógico y Recursos Materiales. Se debe especificar todo lo que se ha utilizado para el desarrollo del sistema.

Tabla 52

Recursos Utilizados

RECURSOS UTILIZADOS		
RECURSO HUMANO		
CARGO	NOMBRE	INSTITUCIÓN
Tutor	Ing. Jaime Basantes	Instituto “Cordillera”
Lector	Lcda. Patricia Garzón	Instituto “Cordillera”
Estudiante	Lucia Caguano	Instituto “Cordillera”
Director TI	Ing. Fabricio Tamayo	Biologicsoft
Departamento QA&QC	Ing. Jorge Ramírez	Biologicsoft
RECURSO LÓGICO		
Visual Studio 2017		
SQL Server 2016		
RECURSO TECNOLÓGICO		
Computador HP Core I5		
Internet		
Impresora		
Hojas		
Esferográficos		
Carpetas		
NOTA: En esta tabla se muestra un listado de los recursos que se han utilizado en el desarrollo del sistema		

6.02 Presupuesto

Tabla 51

Tabla de Presupuesto

DESCRIPCIÓN	CANTIDAD	VALOR UNITARIO	VALOR TOTAL
Computador HP Core I5	1	1000	1000
Impresora	1	250	250
Flash Memory	1	15	15
Proyecto Titulación	1	815.54	815.54
Internet	5 Meses	40	200
Resmas de papel	2	4.50	9
Alimentación	30	2.50	75
Energía Eléctrica	5	8	40
Empastados	0	65	65
Anillados	3	8	24
CD'S	0	10	10
Gastos Varios			75
TOTAL			2578.54

NOTA: Esta tabla muestra la cantidad total de dinero que se invirtió en el desarrollo del sistema

6.03 Cronograma

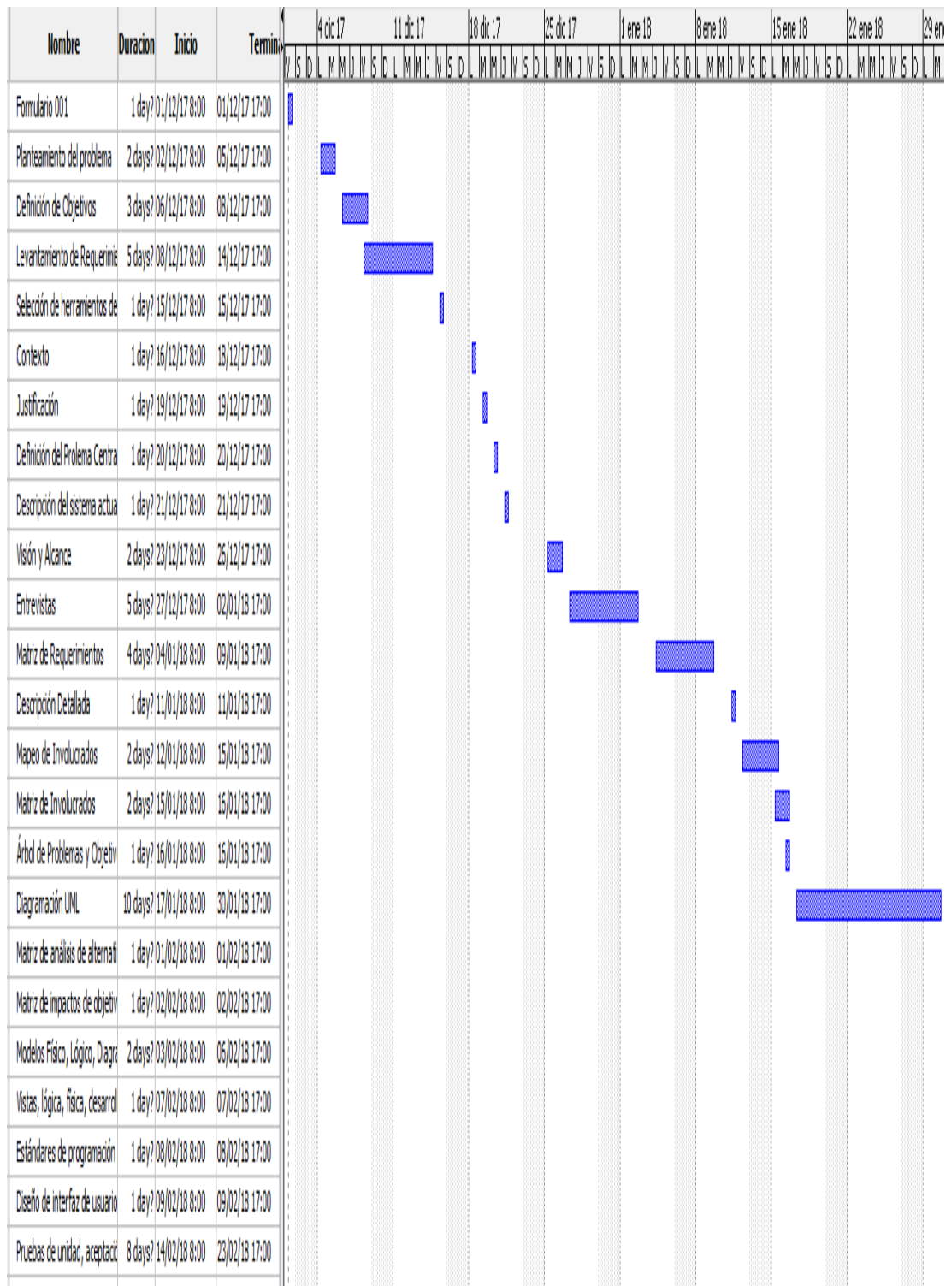


Figura 54: En este gráfico se puede visualizar el cronograma de actividades

CAPÍTULO VII

7 Conclusiones y Recomendaciones

7.01 Conclusiones

Con la implementación del sistema logramos la reducción del tiempo en la ejecución de pruebas unitarias en la empresa Biologicsoft.

La empresa queda totalmente satisfecha es así que todo se a cumplido de acuerdo a los requerimientos solicitados al inicio de la realización del sistema, no queda nada pendiente.

La empresa tendrá información clara y precisa del funcionamiento de los productos. Se contribuyó con la Empresa Biologicsoft, facilitando a los departamentos de Aseguramiento de la Calidad (QA) y Control de Calidad (QC) la carga de trabajo.

Los conocimientos adquiridos en el instituto han sido de vital ayuda, pero se ha recibido adicional una capacitación por parte de la empresa ya que en la misma se manejan distintos criterios en los desarrollos.

7.02 Recomendaciones

Los usuarios del sistema deben recibir una capacitación para manipular el sistema de manera correcta

Para el buen funcionamiento del sistema los usuarios deben revisar los manuales de usuarios adjuntos. (Manual de Usuario, Manual Técnico y Manual de Instalación)

Para que el sistema funcione correctamente los usuarios no deben borrar la información base del sistema, ya que esto podría ocasionar un resultado no favorable del mismo.

El instituto debería aplicar métodos de estudio reales.

7.03 Referencias Bibliográficas

<http://www.aragonvalley.com/es/arbol-de-objetivos-goal-tree-metodo-establecer-criterios/>

<https://users.dcc.uchile.cl/~psalinas/uml/modelo.html>

<http://www.biologicsoft.com/>

<https://www.computrabajo.com.ec/empresas/ofertas-de-trabajo-de-biologicsoft-CC222B153DCB4999>

<http://www.pmoinformatica.com/2015/04/requerimientos-no-funcionales-una.html>

https://es.wikipedia.org/wiki/Requisito_no_funcional

ANEXOS



CARRERA DE ANÁLISIS DE SISTEMAS

**AUTOMATIZACIÓN DEL PROCESO DE EJECUCIÓN DE PRUEBAS
AUTOMÁTICAS UNITARIAS DE APLICATIVOS PARA LA EMPRESA
BIOLOGICSOFT UBICADA EN LA CIUDAD DE QUITO.**

MANUAL DE USUARIO

AUTOR: Caguano Millingalle Lucía Janeth

DIRECTOR: Ing. Basantes Basantes Jaime Neptalí

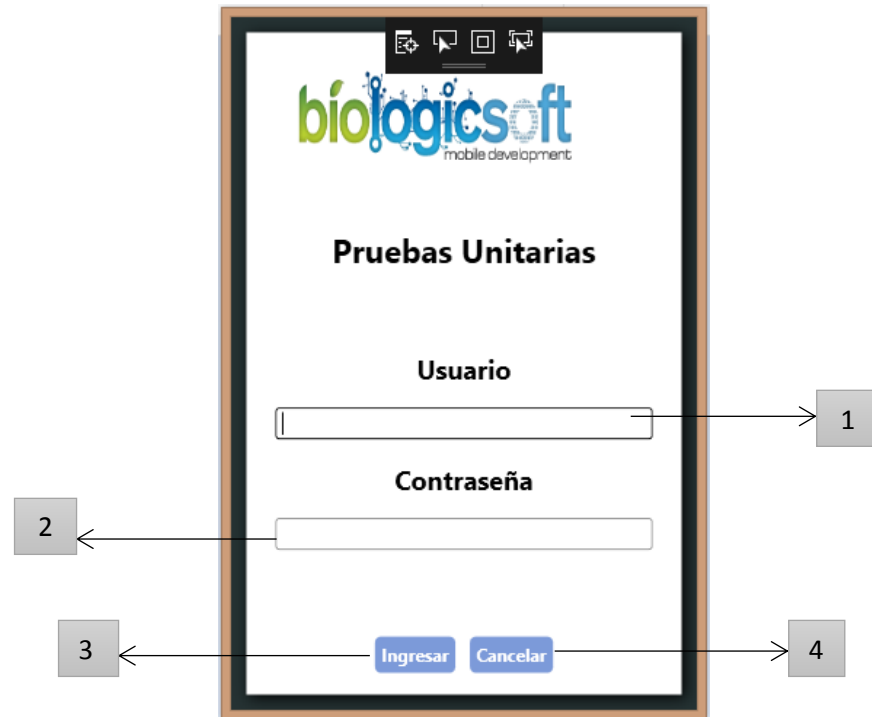
Quito, 2018

INDICE

MANUAL DE USUARIO

INGRESO AL SISTEMA	3
INTERFAZ DE ADMINISTRADOR	4
INTERFAZ DE DISEÑADOR MANTENIMIENTO LENGUAJE	5
INTERFAZ DE DISEÑADOR MANTENIMIENTO VERSIÓN.....	6
INTERFAZ DE DISEÑADOR MANTENIMIENTO APLICACIÓN.....	7
INTERFAZ DE DISEÑADOR MANTENIMIENTO ENTIDADES	8
INTERFAZ DEL TESTER EJECUCIÓN DE PRUEBAS	9
INTERFAZ DEL TESTER PROBAR SISTEMA	10
INTERFAZ TESTER RESULTADOS.....	11
INTERFAZ TESTER GENERACIÓN DE REPORTES	12
INTERFAZ TESTER ENVÍO DE REPORTES	13

INGRESO AL SISTEMA



The image shows a mobile application interface for 'Pruebas Unitarias' (Unit Tests). At the top, there is a logo for 'biologicssoft mobile development'. Below the logo, the title 'Pruebas Unitarias' is displayed. The interface contains two input fields: 'Usuario' (User) and 'Contraseña' (Password). Below these fields are two buttons: 'Ingresar' (Login) and 'Cancelar' (Cancel). Four numbered callouts are present: 1 points to the 'Usuario' input field, 2 points to the 'Contraseña' input field, 3 points to the 'Ingresar' button, and 4 points to the 'Cancelar' button.

Figura 55: Login

Ingreso al Sistema (Login)

Es la ventana de ingreso al sistema todos los usuarios deben ingresar sus credenciales (Usuario y Contraseña), para el ingreso a la interfaz correspondiente.

1.- Usuario.- Debe ingresar el nombre de usuario que le asignó el Administrador para el ingreso a la interfaz.

2.- Contraseña.- Una vez que ingresa el usuario debe ingresar la contraseña de usuario.

3.- Ingresar.- Una vez que se ingrese el usuario y la contraseña debe presionar el botón Ingresar para poder ingresar a la interfaz del Usuario correspondiente

4.- Cancelar.- El botón sirve para cancelar la operación de ingreso al sistema.

INTERFAZ DE ADMINISTRADOR

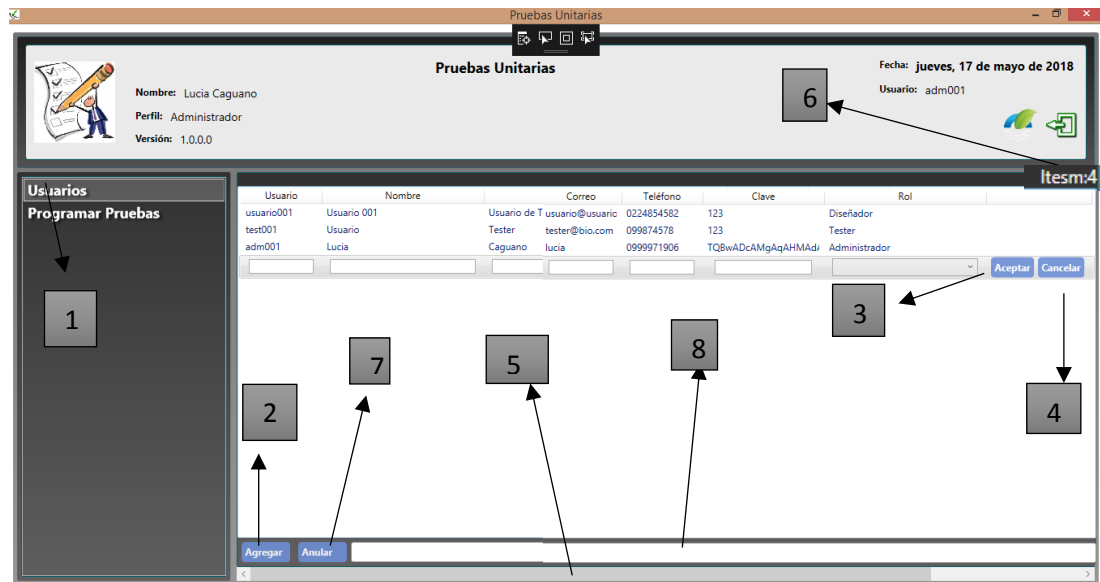


Figura 56: Interfaz Administrador

- 1.- **Usuarios.**- Menú de administrador solo puede crear, editar y anular usuarios
- 2.- **Agregar.**- El botón agregar despliega los campos para el ingreso de la información
- 3.- **Aceptar.**- Una vez ingresada la información seleccionar el botón aceptar para guardar
- 4.- **Cancelar.**- El botón cancelar sirve para cancelar la operación ya sea de creación o edición
- 5.- **Scroll.**- Es la barra para desplazarse de forma horizontal en el sistema
- 6.- **Ítems.**- Muestra la cantidad de ítems o la cantidad de usuarios registrados
- 7.- **Anular.**- Este botón sirve para anular el registro de usuario que requiera
- 8.- **Filtro de Búsqueda.**- Este filtro realiza búsquedas de los usuarios registrados en el sistema.

INTERFAZ DE DISEÑADOR MANTENIMIENTO LENGUAJE

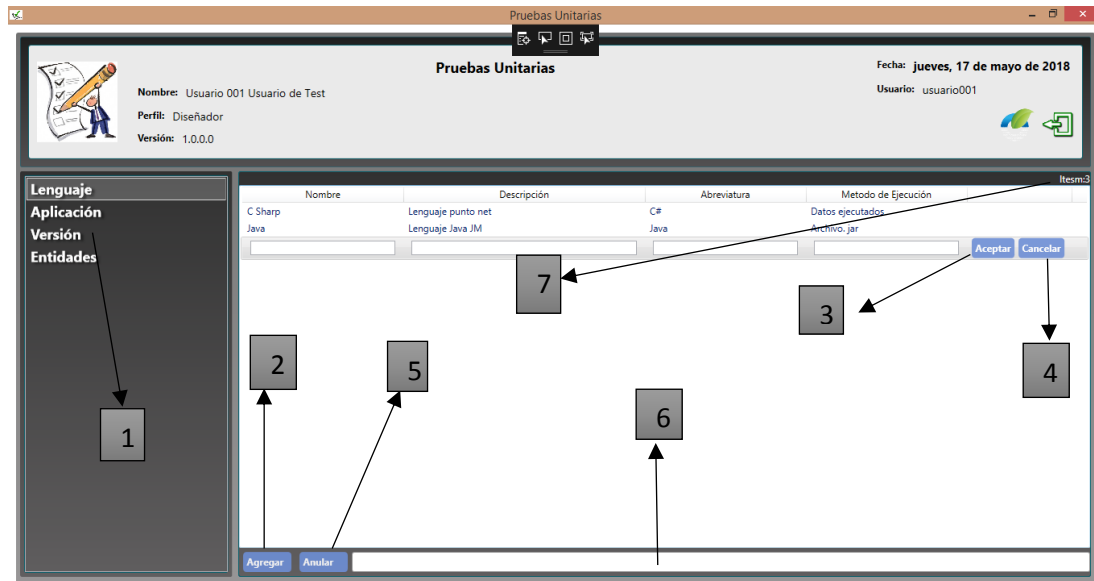


Figura 57: Diseñador - Lenguaje

- 1.- Menú.-** Muestra las opciones que tiene el diseñador para realizar el mantenimiento
- 2.- Agregar Lenguaje.-** Al seleccionar el botón agregar se desplazan los campos para el ingreso de información
- 3.- Aceptar.-** Una vez que se ingresen todos los datos seleccionar el botón Aceptar para que la información se guarde
- 4.- Cancelar.-** Si desea cancelar la operación ya sea de creación o modificación se debe presionar el botón Cancelar
- 5.- Anular.-** Este botón anula el registro de Lenguaje
- 6.- Filtro de búsqueda.-** Mediante este filtro se realizarán búsquedas de entre los registros
- 7.- Ítems.-** Muestra la cantidad de registros ingresados

INTERFAZ DE DISEÑADOR MANTENIMIENTO VERSIÓN

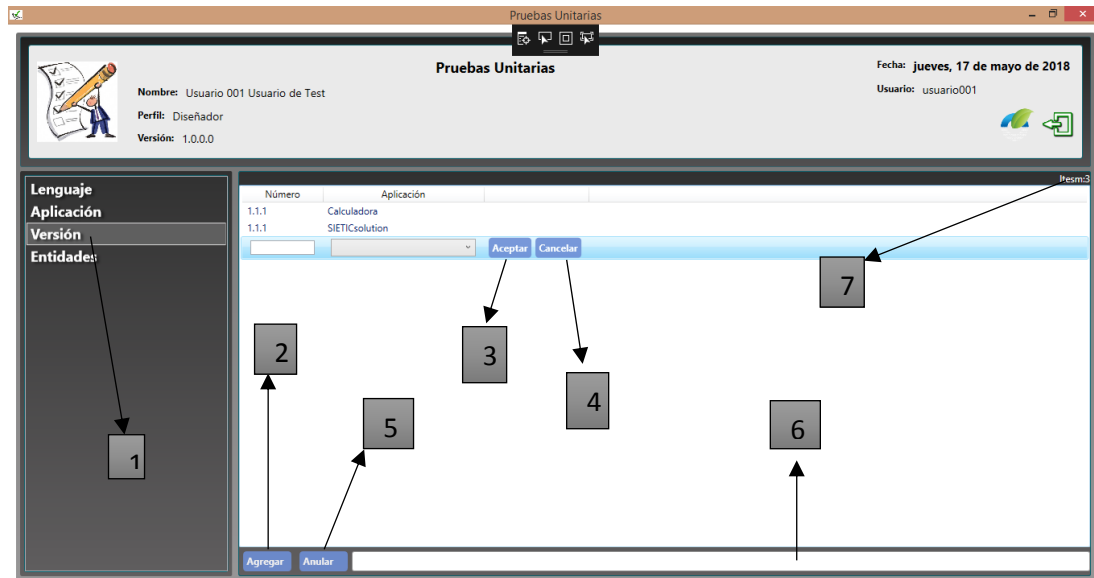


Figura 58: Diseñador - Versión

- 1.- **Menú.-** Muestra las opciones disponibles para el mantenimiento del diseñador
- 2.- **Agregar Versión.-** Al presionar el botón agregar se despliegan los campos para el ingreso de información de versión
- 3.- **Aceptar.-** Una vez ingresada la información seleccionar la opción Aceptar para que la información se guarde
- 4.- **Cancelar.-** Sirve para cancelar la información ya sea de creación o edición
- 5.- **Anular.-** Sirve para anular los registros requeridos
- 6.- **Filtro de búsqueda.-** Se debe ingresar un criterio de búsqueda en la caja de texto y trae solo los registros que contengan el criterio ingresado
- 7.- **Ítems.-** Muestra la cantidad total de registros de versión ingresados

INTERFAZ DE DISEÑADOR MANTENIMIENTO APLICACIÓN

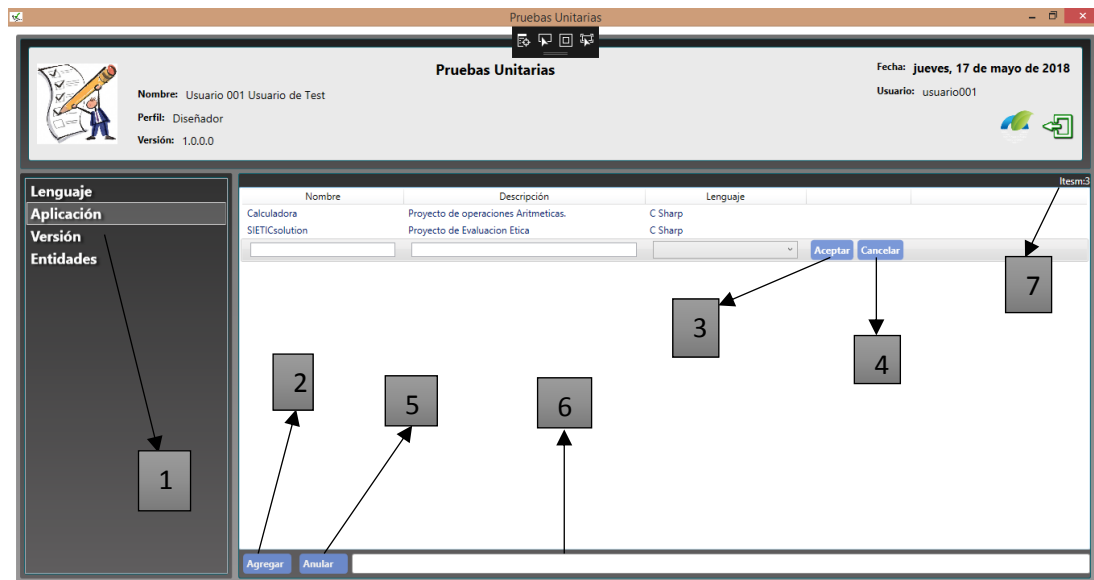


Figura 59: Diseñador - Aplicación

- 1.- Menú.-** Muestra el menú que tiene el usuario diseñador para realizar el mantenimiento
- 2.- Agregar.-** Al presionar este botón se despliegan los campos en modo de creación para el ingreso de información
- 3.- Aceptar.-** Para almacenar la información ingresada
- 4.- Cancelar.-** Presionar este botón si desea cancelar cualquiera de las operaciones ya sea de creación o modificación
- 5.- Anular.-** Esta opción sirve para anular uno de los registros ingresados
- 6.- Filtro de búsqueda.-** Ingresar un criterio de búsqueda, mediante este filtro solo mostrará los registros que contengan el criterio de búsqueda registrado
- 7.- Ítems.-** Muestra la cantidad total de registros de aplicación registrados

INTERFAZ DE DISEÑADOR MANTENIMIENTO ENTIDADES

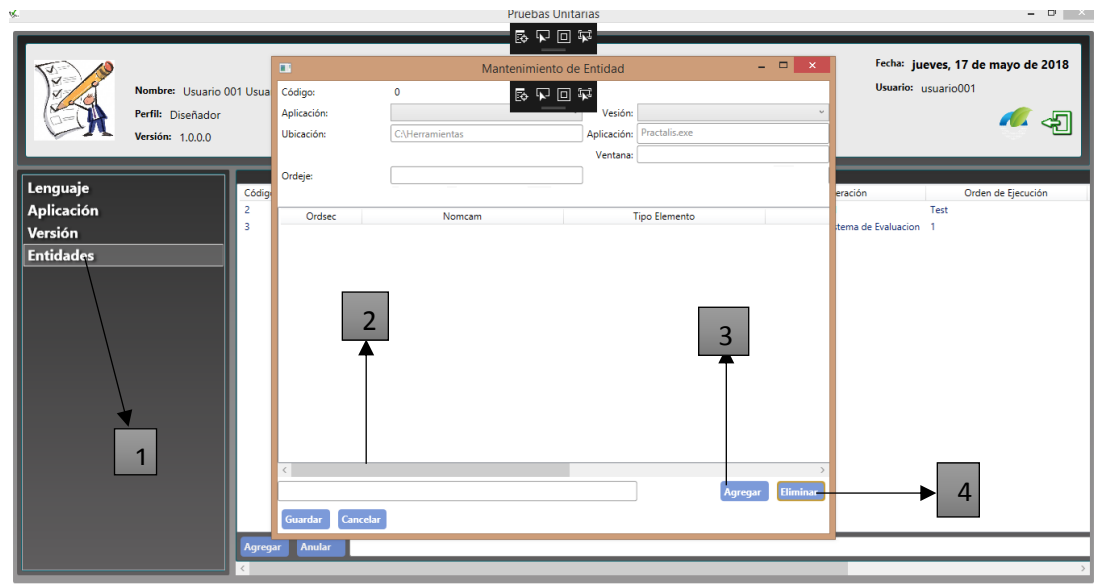


Figura 60: Diseñador - Entidades

Entidades

En esta página se muestran todos los registros las entidades (ventanas) que han sido creadas por el usuario Diseñador.

1.- Menú.- Muestra las ventanas de mantenimiento del usuario Diseñador, en la cual debe seleccionar la opción Entidades, y se despliega la ventana que corresponde a la información ingresada en modo de edición o creación según sea el caso.

2.- Scroll.- Es la barra que permite desplazarse de forma horizontal dentro del aplicativo para al ingreso o consulta de los registros

3.- Agregar.- Al presionar la opción Agregar se despliega la pantalla para la creación de ventanas

4.-Eliminar.- Este botón se presiona para la realizar el proceso de borrado de un campo de la ventana

INTERFAZ DEL TESTER EJECUCIÓN DE PRUEBAS

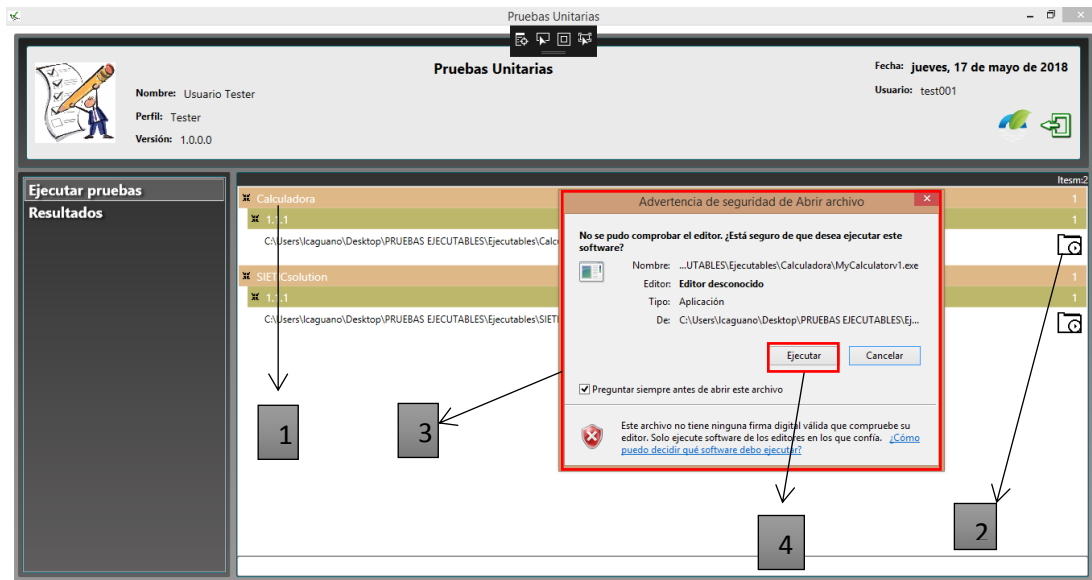


Figura 61: Tester - Ejecución de Pruebas

Ejecución de Pruebas

En esta ventana se muestran todas las entidades creadas por el usuario Diseñador.

1.- Nombre de la Aplicación.- El número 1 señala el nombre de las aplicaciones que se muestran creadas para la evaluación

2.- Botón Ejecutar.- Es el botón que se encuentra al lado derecho de cada una de las entidades creadas, y es el que al presionar se levantará el ejecutable del sistema

3.- Ventana de ejecución.- El número 3 apunta a la ventana emergente que se despliega una vez que se presiona el botón de ejecutar, la cual indica si se va a ejecutar o no el sistema a probar

4.- Botón Ejecutar.- El número 4 apunta al botón ejecutar dentro de la ventana emergente, el mismo al ser presionado se despliega el sistema a probar

INTERFAZ DEL TESTER PROBAR SISTEMA

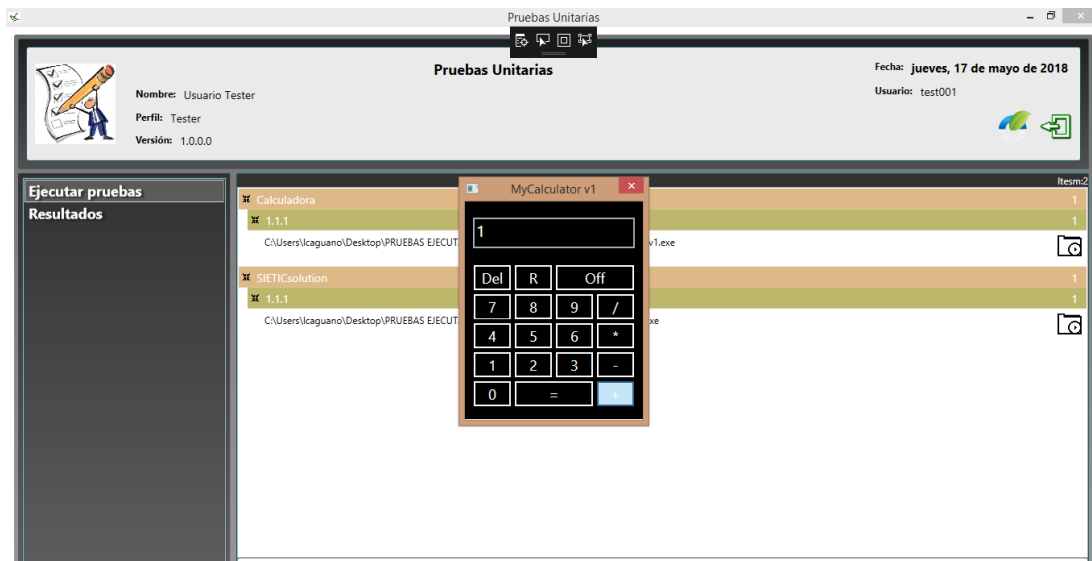


Figura 62: Tester - Probar Sistema

Probar el sistema

Esta interfaz es donde se muestra el aplicativo a evaluar, en funcionamiento, una vez que se presiona el botón ejecutar el sistema se despliega y comienza a ingresar o realizar las acciones que el usuario diseñador parametrizó.

Las aplicaciones que son probadas se levantan automáticamente y al finalizar el proceso de evaluación se cierra y se despliega el resultado.

INTERFAZ TESTER RESULTADOS

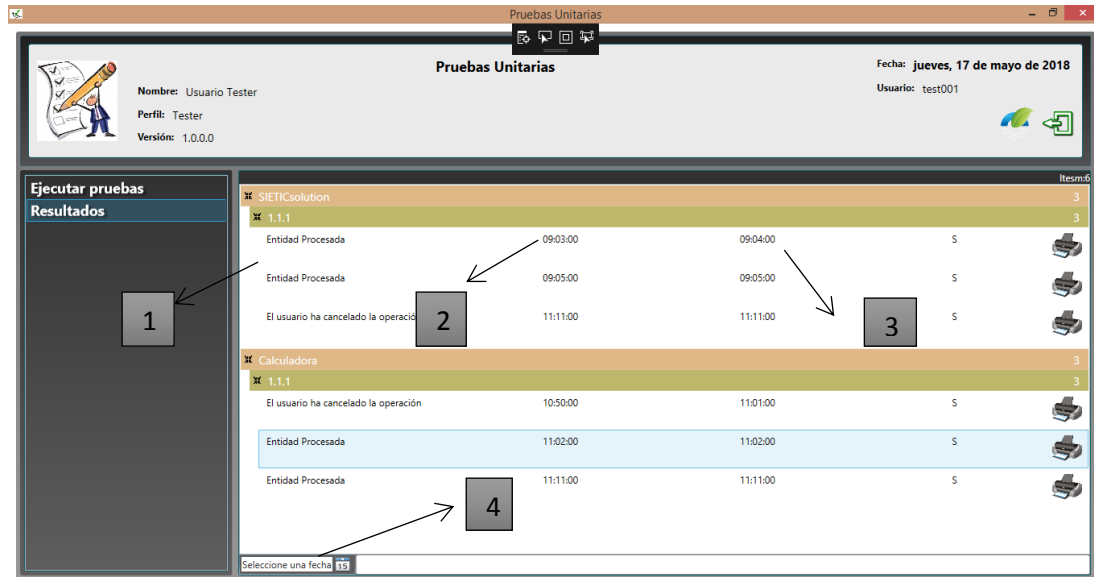


Figura 63: Tester – Resultados

Resultados

En la segunda opción de la interfaz del usuario tester se encuentran los Resultados de las ejecuciones de las ventanas

- 1.- Resultado.-** El número 1 muestra el resultado de la operación ya se que la entidad de procesó correctamente o si fue cancelada por el usuario
- 2.- Hora de inicio.-** El número 2 apunta a la hora en que se inició la evaluación
- 3.- Hora de Fin.-** El número 3 apunta a la hora de finalización de la prueba
- 4.- Calendario.-** El número 4 apunta al calendario, que sirve para realizar las búsquedas de los resultados de la fecha que se seleccione

INTERFAZ TESTER GENERACIÓN DE REPORTES

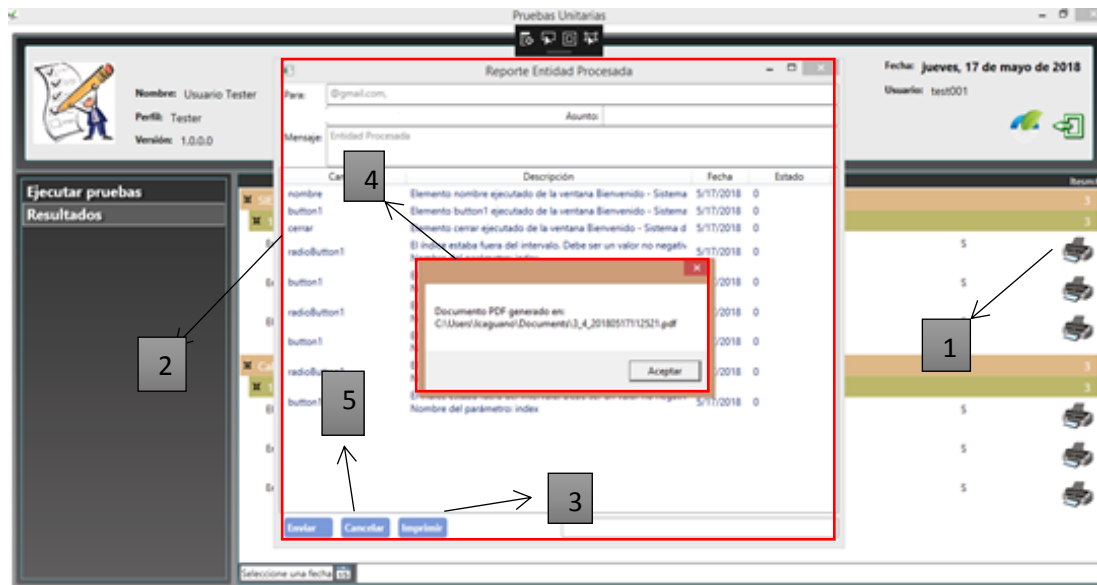


Figura 64: Tester – Reportes

Reportes

En esta interfaz se muestra el reporte de la evaluación

- 1.- Botón Impresora.-** Al seleccionar el botón de impresora se muestra la ventana de envío y generación de reporte
- 2.- Ventana.-** El número 2 muestra la ventana de Reporte Entidad Procesada que se despliega al presionar el botón de impresora
- 3.- Botón Imprimir.-** Para generar el reporte se debe seleccionar el botón imprimir
- 4.- Ventana información.-** Al presionar el botón imprimir se genera el reporte y se muestra una ventana emergente de donde indica la dirección donde se guarda el documento PDF
- 5.- Botón Cancelar.-** Al presionar el botón cancelar la ventana se cierra

INTERFAZ TESTER ENVÍO DE REPORTE

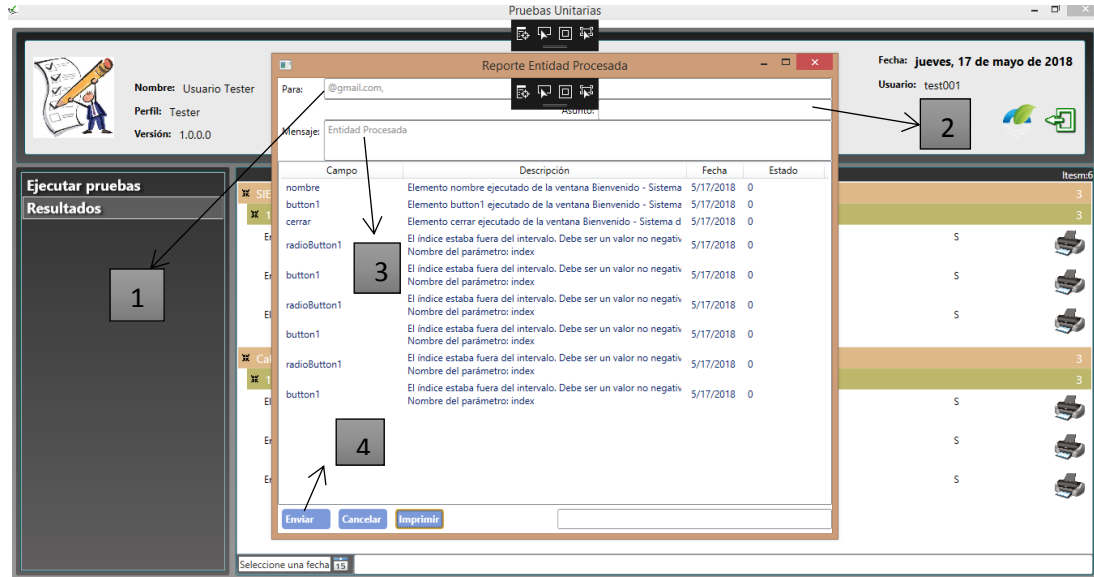


Figura 65: Tester - Envío Reportes

Envío de Reportes

En esta interfaz se visualiza el proceso para el envío de reportes a los involucrados en el proceso del desarrollo del sistema

- 1.- Para:** Es el campo donde ingresar la información del receptor del correo, puede ser una o más direcciones de correo
- 2.- Asunto:** En este campo de ingresar un asunto del envío del correo
- 3.- Mensaje:** En este campo se debe ingresar un mensaje corto
- 4.- Enviar:** Este botón se presiona una vez que hayan llenado los campos par que el mensaje pueda ser enviado



CARRERA DE ANÁLISIS DE SISTEMAS

**AUTOMATIZACIÓN DEL PROCESO DE EJECUCIÓN DE PRUEBAS
AUTOMÁTICAS UNITARIAS DE APLICATIVOS PARA LA EMPRESA
BIOLOGICSOFT UBICADA EN LA CIUDAD DE QUITO.**

MANUAL TÉCNICO

AUTOR: Caguano Millingalle Lucía Janeth

DIRECTOR: Ing. Basantes Basantes Jaime Neptalí

Quito, 2018

INDICE

MANUAL TÉCNICO

SCRIPT DE LA BASE DE DATOS	3
CAPA ACCESO.....	15
CAPA ACCESO A DATOS BASE	15
MÉTODOS DATOS	19
SQL CAPA ACCESO DATOS.....	20
CAPA LÓGICA	49
APLICACIÓN.....	49
BASE.....	55
ENTIDAD	57
USUARIOS	63
EJECUTAR ENTIDAD	69
UNIT TEST.....	69

SCRIPT DE LA BASE DE DATOS

```
USE [master]
GO
/***** Object: Database [dbo_pruebas_automatizadas]  Script Date: 18/3/2018
21:34:13 *****/
CREATE DATABASE [dbo_pruebas_automatizadas]
    CONTAINMENT = NONE
    ON PRIMARY
    ( NAME = N'dbo_pruebas_automatizadas', FILENAME = N'C:\Program
Files\Microsoft SQL
Server\MSSQL14.SQLEXPRESS\MSSQL\DATA\dbo_pruebas_automatizadas.mdf'
, SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )
    LOG ON
    ( NAME = N'dbo_pruebas_automatizadas_log', FILENAME = N'C:\Program
Files\Microsoft SQL
Server\MSSQL14.SQLEXPRESS\MSSQL\DATA\dbo_pruebas_automatizadas_log.
ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
COMPATIBILITY_LEVEL = 140
GO
IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC [dbo_pruebas_automatizadas].[dbo].[sp_fulltext_database] @action = 'enable'
end
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET ANSI_NULL_DEFAULT
OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET ANSI_NULLS OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET ANSI_PADDING OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET ANSI_WARNINGS OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET ARITHABORT OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET AUTO_CLOSE ON
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET AUTO_SHRINK OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
AUTO_UPDATE_STATISTICS ON
GO
```

```
ALTER DATABASE [dbo_pruebas_automatizadas] SET
CURSOR_CLOSE_ON_COMMIT OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET CURSOR_DEFAULT
GLOBAL
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
CONCAT_NULL_YIELDS_NULL OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
NUMERIC_ROUNDABORT OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET QUOTED_IDENTIFIER
OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
RECURSIVE_TRIGGERS OFF
```

```
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET ENABLE_BROKER
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
AUTO_UPDATE_STATISTICS_ASYNC OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
DATE_CORRELATION_OPTIMIZATION OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET TRUSTWORTHY OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
ALLOW_SNAPSHOT_ISOLATION OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET PARAMETERIZATION
SIMPLE
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
READ_COMMITTED_SNAPSHOT OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
HONOR_BROKER_PRIORITY OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET RECOVERY SIMPLE
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET MULTI_USER
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET PAGE_VERIFY
CHECKSUM
GO
```

```
ALTER DATABASE [dbo_pruebas_automatizadas] SET DB_CHAINING OFF
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET FILESTREAM(
NON_TRANSACTED_ACCESS = OFF )
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
TARGET_RECOVERY_TIME = 60 SECONDS
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET
DELAYED_DURABILITY = DISABLED
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET QUERY_STORE = OFF
GO
USE [dbo_pruebas_automatizadas]
GO
ALTER DATABASE SCOPED CONFIGURATION SET IDENTITY_CACHE =
ON;
GO
ALTER DATABASE SCOPED CONFIGURATION SET
LEGACY_CARDINALITY_ESTIMATION = OFF;
GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET
LEGACY_CARDINALITY_ESTIMATION = PRIMARY;
GO
ALTER DATABASE SCOPED CONFIGURATION SET MAXDOP = 0;
GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET
MAXDOP = PRIMARY;
GO
ALTER DATABASE SCOPED CONFIGURATION SET
PARAMETER_SNIFFING = ON;
GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET
PARAMETER_SNIFFING = PRIMARY;
GO
ALTER DATABASE SCOPED CONFIGURATION SET
QUERY_OPTIMIZER_HOTFIXES = OFF;

GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET
QUERY_OPTIMIZER_HOTFIXES =

PRIMARY;
GO
USE [dbo_pruebas_automatizadas]
GO
/***** Object: Table [dbo].[tbl_roles]  Script Date: 18/3/2018 21:34:13 *****/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_roles](
    [rol_codigo] [int] IDENTITY(1,1) NOT NULL,
    [rol_descri] [varchar](100) NULL,
PRIMARY KEY CLUSTERED
(
    [rol_codigo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[tbl_usuarios]  Script Date: 18/3/2018 21:34:14
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_usuarios](
    [usu_codigo] [int] IDENTITY(1,1) NOT NULL,
    [usu_nombre] [varchar](100) NULL,
    [usu_apelli] [varchar](100) NULL,
    [usu_cedula] [varchar](10) NULL,
    [usu_correo] [varchar](250) NULL,
    [usu_celula] [varchar](15) NULL,
    [usu_usuario] [varchar](100) NULL,
    [usu_passwd] [varchar](100) NULL,
    [rol_codigo] [int] NULL,
    [usu_estado] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [usu_codigo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY],
CONSTRAINT [uq_usuario] UNIQUE NONCLUSTERED
(
    [usu_usuario] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY],
CONSTRAINT [uq_usuario_cedula] UNIQUE NONCLUSTERED
(
    [usu_cedula] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,  
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,  
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY],
```

```
CONSTRAINT [uq_usuario_correo] UNIQUE NONCLUSTERED  
(  
    [usu_correo] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,  
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,  
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]  
) ON [PRIMARY]
```

```
GO
```

```
/****** Object: View [dbo].[vi_usuarios]   Script Date: 18/3/2018 21:34:14  
*****/
```

```
SET ANSI_NULLS ON
```

```
GO
```

```
SET QUOTED_IDENTIFIER ON
```

```
GO
```

```
create view [dbo].[vi_usuarios]
```

```
as
```

```
select u.usu_codigo, u.usu_nombre, u.usu_apelli, u.usu_cedula, u.usu_correo,  
u.usu_celula, u.usu_usuari, u.usu_passwd, u.rol_codigo,  
(select r.rol_descri from tbl_rols r where u.rol_codigo = r.rol_codigo) as rol,  
u.usu_estado
```

```
from tbl_usuarios u;
```

```
GO
```

```
/****** Object: Table [dbo].[tbl_aplicacion]   Script Date: 18/3/2018 21:34:14  
*****/
```

```
SET ANSI_NULLS ON
```

```
GO
```

```
SET QUOTED_IDENTIFIER ON
```

```
GO
```

```
CREATE TABLE [dbo].[tbl_aplicacion](
```

```
    [apl_codigo] [int] IDENTITY(1,1) NOT NULL,
```

```
    [apl_nombre] [varchar](50) NULL,
```

```
    [apl_descri] [varchar](250) NULL,
```

```
    [len_codigo] [int] NULL,
```

```
PRIMARY KEY CLUSTERED
```

```
(
```

```
    [apl_codigo] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,  
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,  
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
```

```
) ON [PRIMARY]
```

```
GO
```

/****** Object: Table [dbo].[tbl_entidad] Script Date: 18/3/2018 21:34:14
*****/

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_entidad](
[ent_codigo] [int] IDENTITY(1,1) NOT NULL,
[apl_codigo] [int] NULL,
```

```
[ver_codigo] [int] NULL,
[ent_namesp] [varchar](250) NULL,
[ent_nomven] [varchar](250) NULL,
[ent_afirma] [varchar](1) NULL,
[ent_operac] [varchar](100) NULL,
[ent_ordeje] [varchar](250) NULL,
[ent_impxam] [int] NULL,
[ent_impcs] [int] NULL,
```

```
PRIMARY KEY CLUSTERED
```

```
(
```

```
[ent_codigo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

/****** Object: Table [dbo].[tbl_entidad_detalle] Script Date: 18/3/2018
21:34:14 *****/

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_entidad_detalle](
[ede_codigo] [int] IDENTITY(1,1) NOT NULL,
[ent_codigo] [int] NOT NULL,
[ede_ordsec] [varchar](50) NULL,
[ede_nomcam] [varchar](100) NULL,
[ede_tipdat] [varchar](100) NULL,
[ede_longit] [varchar](10) NULL,
[ede_parcon] [varchar](150) NULL,
[ede_ordeje] [varchar](100) NULL,
```

```
PRIMARY KEY CLUSTERED
```

```
(
```

```
[ede_codigo] ASC,
```

```
[ent_codigo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[tbl_entidad_resultado]  Script Date: 18/3/2018
21:34:14 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_entidad_resultado](

    [ere_codigo] [int] IDENTITY(1,1) NOT NULL,
    ent_codigo [int] NULL,
        [ere_descri] [varchar](250) NULL,
        [ere_horini] [time](7) NULL,

        [ere_horfin] [time](7) NULL,
        [ere_tipope] [varchar](100) NULL,
        [ere_codeje] [int] NULL,
    PRIMARY KEY CLUSTERED
(

    [ere_codigo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[tbl_lenguaje]  Script Date: 18/3/2018 21:34:14
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_lenguaje](
    [len_codigo] [int] IDENTITY(1,1) NOT NULL,
    [len_nombre] [varchar](100) NULL,
    [len_descri] [varchar](250) NULL,
    [len_abrevi] [varchar](100) NULL,
    [len_meteje] [varchar](100) NULL,
    PRIMARY KEY CLUSTERED
(
```

```
[len_codigo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[tbl_usuario]  Script Date: 18/3/2018 21:34:14
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_usuario](
    [usu_codigo] [varchar](20) NOT NULL,
    [usu_coreo] [varchar](200) NOT NULL,
    [usu_clave] [varchar](30) NOT NULL,
    [usu_fecha_acceso] [datetime] NOT NULL,
    [usu_permitir_operacion] [int] NOT NULL,
    [usu_intentos_fallidos] [int] NOT NULL,
    [usu_imagen] [image] NULL,
    [usu_estado] [char](1) NOT NULL,
    CONSTRAINT [PK_tbl_usuario] PRIMARY KEY CLUSTERED

    [usu_codigo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/***** Object: Table [dbo].[tbl_version]  Script Date: 18/3/2018 21:34:14
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[tbl_version](

    [apl_codigo] [int] NOT NULL,
    [ver_codigo] [int] NOT NULL,
    [ver_numero] [varchar](10) NULL,
    PRIMARY KEY CLUSTERED
(
    [apl_codigo] ASC,
    [ver_codigo] ASC
```

```
) WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,  
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,  
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]  
) ON [PRIMARY]  
GO
```

```
ALTER TABLE [dbo].[tbl_usuario] ADD CONSTRAINT  
[DF_tbl_usuario_usu_fecha_acceso]
```

```
DEFAULT (getdate()) FOR [usu_fecha_acceso]
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_usuario] ADD CONSTRAINT  
[DF_tbl_usuario_usu_permitir_operacion] DEFAULT ((0)) FOR  
[usu_permitir_operacion]
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_usuario] ADD CONSTRAINT  
[DF_tbl_usuario_usu_intentos_fallidos] DEFAULT ((0)) FOR  
[usu_intentos_fallidos]
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_usuario] ADD CONSTRAINT  
[DF_tbl_usuario_usu_imagen] DEFAULT (NULL) FOR [usu_imagen]
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_usuario] ADD CONSTRAINT  
[DF_tbl_usuario_usu_estado] DEFAULT ('A') FOR [usu_estado]
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_aplicacion] WITH CHECK ADD CONSTRAINT  
[fk_lenguaje_aplicacion] FOREIGN KEY([len_codigo])  
REFERENCES [dbo].[tbl_lenguaje] ([len_codigo])
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_aplicacion] CHECK CONSTRAINT  
[fk_lenguaje_aplicacion]
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_entidad] WITH CHECK ADD CONSTRAINT  
[fk_version_entidad] FOREIGN KEY([apl_codigo], [ver_codigo])  
REFERENCES [dbo].[tbl_version] ([apl_codigo], [ver_codigo])
```

```
GO
```

```
ALTER TABLE [dbo].[tbl_entidad] CHECK CONSTRAINT [fk_version_entidad]  
GO
```

```
ALTER TABLE [dbo].[tbl_entidad_detalle] WITH CHECK ADD CONSTRAINT  
[fk_entidad_ent_detalle] FOREIGN KEY([ent_codigo])  
REFERENCES [dbo].[tbl_entidad] ([ent_codigo])  
GO
```

```
ALTER TABLE [dbo].[tbl_entidad_detalle] CHECK CONSTRAINT
[fk_entidad_ent_detalle]
GO
ALTER TABLE [dbo].[tbl_entidad_resultado] WITH CHECK ADD
CONSTRAINT [fk_entidad_ent_resultado] FOREIGN KEY([ent_codigo])
REFERENCES [dbo].[tbl_entidad] ([ent_codigo])
GO
ALTER TABLE [dbo].[tbl_entidad_resultado] CHECK CONSTRAINT
[fk_entidad_ent_resultado]

GO
ALTER TABLE [dbo].[tbl_usuarios] WITH CHECK ADD CONSTRAINT
[fk_rol_usuarios] FOREIGN KEY([rol_codigo])
REFERENCES [dbo].[tbl_rol] ([rol_codigo])
GO
ALTER TABLE [dbo].[tbl_usuarios] CHECK CONSTRAINT [fk_rol_usuarios]
GO
ALTER TABLE [dbo].[tbl_version] WITH CHECK ADD CONSTRAINT
[fk_aplicacion_version] FOREIGN KEY([apl_codigo])
REFERENCES [dbo].[tbl_aplicacion] ([apl_codigo])
GO
ALTER TABLE [dbo].[tbl_version] CHECK CONSTRAINT
[fk_aplicacion_version]
GO
/***** Object: StoredProcedure [dbo].[sp_eliminar_usuarios]  Script Date:
18/3/2018 21:34:14 *****/
SET ANSI_NULLS ON

GO
SET QUOTED_IDENTIFIER ON
GO
create procedure [dbo].[sp_eliminar_usuarios]
@codigo int
as
begin
    delete tbl_usuarios where usu_codigo = @codigo;
end
GO
/***** Object: StoredProcedure [dbo].[sp_guardar_usuarios]  Script Date:
18/3/2018 21:34:14 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
create procedure [dbo].[sp_guardar_usuarios]
@codigo int,

@nombres varchar(100),
```

@apellidos **varchar**(100),

@cedula **varchar**(10),

@correo **varchar**(250),

@celular **varchar**(15),

@usuario **varchar**(100),

@passwd **varchar**(100),

@rol **int**,

@estado **int**

as

begin

if exists(**select** * **from** tbl_usuarios **where** usu_codigo = @codigo)

begin

update tbl_usuarios **set**

 usu_nombre = @nombres,

 usu_apelli = @apellidos,

 usu_cedula = @cedula,

 usu_correo = @correo,

 usu_celula = @celular,

 usu_usuari = @usuario,

 usu_passwd = @passwd,

 rol_codigo = @rol

where usu_codigo = @codigo;

end

else

begin

insert into tbl_usuarios(usu_nombre, usu_apelli, usu_cedula,
usu_correo, usu_celula, usu_usuari, usu_passwd, rol_codigo, usu_estado)

values(@nombres, @apellidos, @cedula, @correo, @celular,
@usuario, @passwd, @rol, @estado);

end

end

GO

/****** Object: StoredProcedure [dbo].[sp_login] Script Date: 18/3/2018

21:34:14 *****/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp_login]

@usuario **varchar**(100),

@password **varchar**(100)

AS

BEGIN

```
IF exists(select * from tbl_usuarios where usu_usuari = @usuario and
usu_passwd = @password and usu_estado = 1)
    BEGIN
        select 1 as CODIGO, rol_codigo from tbl_usuarios where usu_usuari
=
@usuario and usu_passwd = @password and usu_estado = 1;

    END

ELSE IF exists(select * from tbl_usuarios where usu_usuari = @usuario and
usu_passwd <> @password)
    BEGIN
        select 2 as CODIGO, 0 as rol_codigo;
    END
ELSE IF exists(select * from tbl_usuarios where usu_usuari = @usuario and
usu_passwd = @password and usu_estado = 0)
    BEGIN
        select 3 as CODIGO, 0 as rol_codigo;
    END
ELSE
    BEGIN
        select 0 as CODIGO, 0 as rol_codigo;
    END
END

GO
/***** Object: StoredProcedure [dbo].[sp_obtener_usuarios]  Script Date:
18/3/2018 21:34:14 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
create procedure [dbo].[sp_obtener_usuarios]
@opcion int,
@parametro varchar(100)
as
begin
    if(@opcion = 1)
    begin
        select * from vi_usuarios;
    end
    else if(@opcion = 2)
```

```
begin
    select * from vi_usuarios where usu_apelli like '%' + @parametro +
    '%' or usu_usuari like '%' + @parametro + '%' or usu_correo like '%' + @parametro
    + '%';
end
else if(@opcion = 3)

begin
    select * from vi_usuarios where usu_cedula like '%' + @parametro +
    '%';
end
end;

execute sp_obtener_usuarios 1, "
```

```
GO
USE [master]
GO
ALTER DATABASE [dbo_pruebas_automatizadas] SET READ_WRITE
GO
```

CAPA ACCESO

CAPA ACCESO A DATOS BASE

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;

namespace AccesoDatos
{
    /// <summary>
    /// Descripción breve de CapaAccesoDatos.
    /// </summary>
    public abstract class CapaAccesoDatosBase
    {
        /// <summary>
        /// Cadena de Conexión.
        /// </summary>
        protected string CadenaConeccion
        {
            get
            {
                string strTemp = ConfigurationSettings.AppSettings.Get("CadenaConeccion");
                if (strTemp == null || strTemp.Length <= 0)
```

```
        throw (new ApplicationException("La configuración de la cadena de
conexión (CadenaConeccion) no se encuentra en el archivo web.config.));
    else
        return strTemp;
    }
}

// Funciones:Aplicacion
public abstract bool CrearAplicacion(out int p_codigoAplicacion, int
p_codigoLenguaje, string p_nombre,
    string p_descri, int p_estado);
public abstract bool ActualizarAplicacion(int p_codigoAplicacion, int
p_codigoLenguaje, string p_nombre,
    string p_descri, int p_estado);
public abstract bool EliminarAplicacion(int p_codigoAplicacion, int
p_codigoLenguaje);
public abstract DataTable ObtenerAplicacion();
public abstract DataTable ObtenerAplicacion(int p_codigoAplicacion);
public abstract DataTable ObtenerAplicacionPorLenguaje(int p_codigoLenguaje);
public abstract DataRow ObtenerAplicacion(int p_codigoAplicacion, int
p_codigoLenguaje);

// Funciones:Entidad
public abstract bool CrearEntidad(out int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion, string p_namesp,
    string p_nomven, string p_afirma, string p_operac, string p_ordeje, int
p_impjam, int p_impjcs, int p_estado);
public abstract bool ActualizarEntidad(int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion,
    string p_namesp, string p_nomven, string p_afirma, string p_operac, string
p_ordeje, int p_impjam, int p_impjcs, int p_estado);
public abstract bool EliminarEntidad(int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion);
public abstract DataTable ObtenerEntidad();
public abstract DataTable ObtenerEntidad(int p_codigoEntidad);
public abstract DataTable ObtenerEntidad(int p_codigoEntidad, int
p_codigoAplicacion);
public abstract DataRow ObtenerEntidad(int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion);
public abstract List<Dictionary<string, object>> ObtenerEntidadCompleto();

// Funciones:Entidad detalle
public abstract bool CrearEntidadDetalle(int p_codigoEntidad, out int
p_codigoEntidadDetalle, string p_ordsec,
    string p_nomcam, string p_tipdat, string p_longit, string p_parcon, string
p_ordeje, int p_tipelm);
public abstract bool ActualizarEntidadDetalle(int p_codigoEntidad, int
p_codigoEntidadDetalle, string p_ordsec,
```

```
    string p_nomcam, string p_tipdat, string p_longit, string p_parcon, string
p_ordeje, int p_tipelm);
    public abstract bool EliminarEntidadDetalle(int p_codigoEntidad, int
p_codigoEntidadDetalle);
    public abstract DataTable ObtenerEntidadDetalle();
    public abstract DataTable ObtenerEntidadDetallePorEntidad(int
p_codigoEntidad);
    public abstract DataRow ObtenerEntidadDetalle(int p_codigoEntidad, int
p_codigoEntidadDetalle);

    // Funciones:Entidad Resultado
    public abstract bool CrearEntidadResultado(int p_codigoEntidad, out int
p_codigoEntidadResultado, string p_descri, DateTime p_fecha,
    DateTime p_horini, DateTime p_horfin, string p_tipope, int p_codeje, int
p_estado);
    public abstract bool ActualizarEntidadResultado(int p_codigoEntidad, int
p_codigoEntidadResultado, string p_descri, DateTime p_fecha,
    DateTime p_horini, DateTime p_horfin, string p_tipope, int p_codeje, int
p_estado);
    public abstract bool EliminarEntidadResultado(int p_codigoEntidad, int
p_codigoEntidadResultado);
    public abstract DataTable ObtenerEntidadResultado();
    public abstract DataTable ObtenerEntidadResultadoPorFecha(DateTime p_fecha);
    public abstract DataTable ObtenerEntidadResultado(int p_codigoEntidad);
    public abstract DataRow ObtenerEntidadResultado(int p_codigoEntidad, int
p_codigoEntidadResultado);
    public abstract List<Dictionary<string, object>>
ObtenerEntidadResultadoCompleto(DateTime p_fecha);

    // Funciones:Entidad Resultado Detalle
    public abstract bool CrearEntidadResultadoDetalle(int p_codigoEntidad, int
p_codigoEntidadResultado, out int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle, string p_descri,
    DateTime p_fecha, int p_estado);
    public abstract bool ActualizarEntidadResultadoDetalle(int p_codigoEntidad, int
p_codigoEntidadResultado, int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle, string p_descri,
    DateTime p_fecha, int p_estado);
    public abstract bool EliminarEntidadResultadoDetalle(int p_codigoEntidad, int
p_codigoEntidadResultado, int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle);
    public abstract DataTable ObtenerEntidadResultadoDetalle();
    public abstract DataTable ObtenerEntidadResultadoDetalle(int p_codigoEntidad,
int p_codigoEntidadResultado);
    public abstract DataTable ObtenerEntidadResultadoDetalle(int p_codigoEntidad,
int p_codigoEntidadResultado, int p_codigoEntidadDetalle);
    public abstract DataRow ObtenerEntidadResultadoDetalle(int p_codigoEntidad,
int p_codigoEntidadResultado, int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle);
```

```
public abstract List<Dictionary<string, object>>
ObtenerEntidadResultadoDetalleCompleto(int p_codigoEntidad, int
p_codigoEntidadResultado);

// Funciones:Lenguaje
public abstract bool CrearLenguaje(out int p_codigoLenguaje, string p_nombre,
string p_descr, string p_abrevi,
string p_meteje, int p_estado);
public abstract bool ActualizarLenguaje(int p_codigoLenguaje, string p_nombre,
string p_descr, string p_abrevi,
string p_meteje, int p_estado);
public abstract bool EliminarLenguaje(int p_codigoLenguaje);
public abstract DataTable ObtenerLenguaje();
public abstract DataRow ObtenerLenguaje(int p_codigoLenguaje);

// Funciones:Usuario
public abstract bool CrearUsuario(out int p_codigo, string p_nombre, string
p_apelli, string p_cedula, string p_correo,
string p_celula, string p_usuari, string p_passwd, int p_rolCodigo, int p_estado);
public abstract bool ActualizarUsuario(int p_codigo, string p_nombre, string
p_apelli, string p_cedula,
string p_correo, string p_celula, string p_usuari, string p_passwd, int
p_rolCodigo, int p_estado);
public abstract bool AnularUsuario(string p_codigoUsuario);
public abstract DataTable ObtenerUsuarios();
public abstract DataRow ObtenerUsuario(string p_codigoUsuario);

// Funciones:Roles
public abstract bool CrearRoles(out int p_codigoRoles, string p_descri);
public abstract bool ActualizarRoles(int p_codigo, string p_descri);
public abstract bool EliminarRoles(int p_codigo);
public abstract DataTable ObtenerRoles();
public abstract DataRow ObtenerRoles(int p_codigo);

// Funciones:Versión
public abstract bool CrearVersion(out int p_codigoVersion, int
p_codigoAplicacion, string p_numero, int p_estado);
public abstract bool ActualizarVersion(int p_codigoVersion, int
p_codigoAplicacion, string p_numero, int p_estado);
public abstract bool EliminarVersion(int p_codigoVersion, int
p_codigoAplicacion);
public abstract DataTable ObtenerVersion();
public abstract DataTable ObtenerVersion(int p_codigoAplicacion);
public abstract DataRow ObtenerVersion(int p_codigoVersion, int
p_codigoAplicacion);

}
public class UsarCapaAccesoDatos
{
```

```
public static CapaAccesoDatosBase ObtenerCapaAccesoDatos()
{
    Type trp = Type.GetType("AccesoDatos.SQLCapaAccesoDatos");
    CapaAccesoDatosBase dc =
(CapaAccesoDatosBase)Activator.CreateInstance(trp);
    return dc;
}
}
```

MÉTODOS DATOS

```
using System;
using System.Data;
using System.Data.SqlClient;

namespace AccesoDatos
{
    public class MetodosDatos
    {
        public static SqlCommand CrearComando(string p_cadenaConexion)
        {
            SqlConnection conexion = new SqlConnection {
ConnectionString = p_cadenaConexion };
            SqlCommand comando =conexion.CreateCommand();
            comando.CommandType = CommandType.Text;
            return comando;
        }

        public static DataTable EjecutarComandoSelect(SqlCommand
comando)
        {
            DataTable tabla = new DataTable();
            try
            {
                comando.Connection.Open();
                SqlDataAdapter adaptador = new SqlDataAdapter();
                adaptador.SelectCommand = comando;
                adaptador.Fill(tabla);
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }
            finally
            {
                comando.Connection.Close();
            }
            return tabla;
        }
    }
}
```

```
    }

    public static bool EjecutarNomQuery(SqlCommand comando)
    {
        try
        {
            comando.Connection.Open();
            int valor = comando.ExecuteNonQuery();
            if (valor > 0) return true;
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
        finally
        {
            comando.Connection.Close();
        }
        return false;
    }

    public static int EjecutarNomQueryRetornaId(SqlCommand comando)
    {
        try
        {
            comando.Connection.Open();
            return (int)comando.ExecuteScalar();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
        finally
        {
            comando.Connection.Close();
        }
        return -1;
    }
}
```

SQL CAPA ACCESO DATOS

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Data;

using System.Data.SqlClient;

using AccesoDatos.Utilitarios;

namespace AccesoDatos
{
    class SQLCapaAccesoDatos : CapaAccesoDatosBase
    {

        #region Aplicación

        public override bool CrearAplicacion(out int p_codigoAplicacion, int
p_codigoLenguaje, string p_nombre,
        string p_descri, int p_estado)
        {
            SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
            comando.CommandText =

                "INSERT INTO tbl_aplicacion (apl_nombre,apl_descri,len_codigo,apl_estado)
OUTPUT inserted.apl_codigo
VALUES(@param2,@param3,@param4,@param5)";

            comando.Parameters.Add("@param2", SqlDbType.VarChar).Value = p_nombre;
            comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_descri;
            comando.Parameters.Add("@param4", SqlDbType.Int).Value =
p_codigoLenguaje;
            comando.Parameters.Add("@param5", SqlDbType.Int).Value = p_estado;

            // id generado
            p_codigoAplicacion = MetodosDatos.EjecutarNomQueryRetornaId(comando);
            return p_codigoAplicacion > 0;
        }
    }
}
```

```
public override bool ActualizarAplicacion(int p_codigoAplicacion, int
p_codigoLenguaje, string p_nombre,
    string p_descri, int p_estado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "UPDATE tbl_aplicacion SET
apl_nombre=@param1,apl_descri=@param2,len_codigo=@param3,apl_estado=@p
aram5 WHERE apl_codigo=@param4";
    comando.Parameters.Add("@param1", SqlDbType.VarChar).Value = p_nombre;
    comando.Parameters.Add("@param2", SqlDbType.VarChar).Value = p_descri;
    comando.Parameters.Add("@param3", SqlDbType.VarChar).Value =
p_codigoLenguaje;
    comando.Parameters.Add("@param4", SqlDbType.VarChar).Value =
p_codigoAplicacion;
    comando.Parameters.Add("@param5", SqlDbType.Int).Value = p_estado;

    return MetodosDatos.EjecutarNomQuery(comando);
}

public override bool EliminarAplicacion(int p_codigoAplicacion, int
p_codigoLenguaje)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "DELETE FROM tbl_aplicacion WHERE apl_codigo=@param1";
    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoAplicacion;

    return MetodosDatos.EjecutarNomQuery(comando);
}
```

```
public override DataTable ObtenerAplicacion()
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_aplicacion";
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataTable ObtenerAplicacion(int p_codigoAplicacion)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_aplicacion
WHERE apl_codigo='{0}'", p_codigoAplicacion);
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataTable ObtenerAplicacionPorLenguaje(int p_codigoLenguaje)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_aplicacion
WHERE len_codigo='{0}'", p_codigoLenguaje);
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataRow ObtenerAplicacion(int p_codigoAplicacion, int
p_codigoLenguaje)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_aplicacion
WHERE apl_codigo='{0}' AND len_codigo={1}",
    p_codigoAplicacion, p_codigoLenguaje);
}
```

```
        return
        Utiles.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));
    }

#endregion

#region Entidad

    public override bool CrearEntidad(out int p_codigoEntidad, int
    p_codigoAplicacion, int p_codigoVersion,

        string p_namesp, string p_nomven, string p_afirma, string p_operac, string
    p_ordeje, int p_impjam, int p_impjcs, int p_estado)
    {
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
        comando.CommandText =

            "INSERT INTO tbl_entidad
(apl_codigo,ver_codigo,ent_namesp,ent_nomven,ent_afirma,ent_operac,ent_ordeje,e
nt_impjam,ent_impjcs,ent_estado) OUTPUT inserted.ent_codigo
VALUES(@param2,@param3,@param4,@param5,@param6,@param7,@param8,
@param9,@param10,@param11)";

        comando.Parameters.Add("@param2", SqlDbType.Int).Value =
    p_codigoAplicacion;

        comando.Parameters.Add("@param3", SqlDbType.Int).Value =
    p_codigoVersion;

        comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_namesp;
        comando.Parameters.Add("@param5", SqlDbType.VarChar).Value =
    p_nomven;

        comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_afirma;
        comando.Parameters.Add("@param7", SqlDbType.VarChar).Value = p_operac;
        comando.Parameters.Add("@param8", SqlDbType.VarChar).Value = p_ordeje;
        comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_impjam;
        comando.Parameters.Add("@param10", SqlDbType.Int).Value = p_impjcs;
        comando.Parameters.Add("@param11", SqlDbType.Int).Value = p_estado;
```

```
// id generado

p_codigoEntidad = MetodosDatos.EjecutarNomQueryRetornaId(comando);

return p_codigoEntidad > 0;

}

public override bool ActualizarEntidad(int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion,

    string p_namesp, string p_nomven, string p_afirma, string p_operac, string
p_ordeje, int p_impjam, int p_impjcs, int p_estado)

{

    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);

    comando.CommandText =

        "UPDATE tbl_entidad SET
ent_namesp=@param4,ent_nomven=@param5,ent_afirma=@param6,ent_operac=@
param7,ent_ordeje=@param8,ent_impjam=@param9,ent_impjcs=@param10,ent_est
ado=@param11 WHERE ent_codigo=@param1 AND apl_codigo=@param2 AND
ver_codigo=@param3";

    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;

    comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoAplicacion;

    comando.Parameters.Add("@param3", SqlDbType.Int).Value =
p_codigoVersion;

    comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_namesp;

    comando.Parameters.Add("@param5", SqlDbType.VarChar).Value =
p_nomven;

    comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_afirma;

    comando.Parameters.Add("@param7", SqlDbType.VarChar).Value = p_operac;

    comando.Parameters.Add("@param8", SqlDbType.VarChar).Value = p_ordeje;

    comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_impjam;

    comando.Parameters.Add("@param10", SqlDbType.Int).Value = p_impjcs;

    comando.Parameters.Add("@param11", SqlDbType.Int).Value = p_estado;
```

```
        return MetodosDatos.EjecutarNomQuery(comando);
    }

    public override bool EliminarEntidad(int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion)
    {
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
        comando.CommandText =
            "DELETE FROM tbl_entidad WHERE ent_codigo=@param1 AND
apl_codigo=@param2 AND ver_codigo=@param3";
        comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;
        comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoAplicacion;
        comando.Parameters.Add("@param3", SqlDbType.Int).Value =
p_codigoVersion;

        return MetodosDatos.EjecutarNomQuery(comando);
    }

    public override DataTable ObtenerEntidad()
    {
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
        comando.CommandText = "SELECT * FROM tbl_entidad";
        return MetodosDatos.EjecutarComandoSelect(comando);
    }

    public override DataTable ObtenerEntidad(int p_codigoEntidad)
    {
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
```

```
comando.CommandText = string.Format("SELECT * FROM tbl_entidad
WHERE ent_codigo='{0}'", p_codigoEntidad);

return MetodosDatos.EjecutarComandoSelect(comando);

}

public override DataTable ObtenerEntidad(int p_codigoEntidad, int
p_codigoAplicacion)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_entidad
WHERE ent_codigo='{0}' AND apl_codigo={1}",
    p_codigoEntidad, p_codigoAplicacion);
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataRow ObtenerEntidad(int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        string.Format("SELECT * FROM tbl_entidad WHERE ent_codigo='{0}' AND
apl_codigo={1} AND ver_codigo={2}",
        p_codigoEntidad, p_codigoAplicacion, p_codigoVersion);
    return
        Utils.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));
}

public override List<Dictionary<string, object>> ObtenerEntidadCompleto()
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT B.apl_nombre,C.ver_numero,A.* FROM
tbl_entidad A INNER JOIN tbl_aplicacion B ON B.apl_codigo=A.apl_codigo
```

```
INNER JOIN tbl_version C ON C.apl_codigo=A.apl_codigo AND  
C.ver_codigo=A.ver_codigo WHERE A.ent_estado=0";  
  
    return  
    Utils.ConvertColeccionGenerica(MetodosDatos.EjecutarComandoSelect(comando)  
    );  
}
```

```
#endregion
```

```
#region Entidad Detalle
```

```
    public override bool CrearEntidadDetalle(int p_codigoEntidad, out int  
p_codigoEntidadDetalle, string p_ordsec,  
  
    string p_nomcam, string p_tipdat, string p_longit, string p_parcon, string  
p_ordeje, int p_tipelm)  
  
    {  
  
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);  
        comando.CommandText =  
  
            "INSERT INTO  tbl_entidad_detalle  
(ent_codigo,ede_ordsec,ede_nomcam,ede_tipdat,ede_longit,ede_parcon,ede_ordeje,  
ede_tipelm) OUTPUT inserted.ede_codigo  
VALUES(@param2,@param3,@param4,@param5,@param6,@param7,@param8,  
@param9)";  
  
        comando.Parameters.Add("@param2", SqlDbType.Int).Value =  
p_codigoEntidad;  
  
        comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_ordsec;  
        comando.Parameters.Add("@param4", SqlDbType.VarChar).Value =  
p_nomcam;  
  
        comando.Parameters.Add("@param5", SqlDbType.VarChar).Value = p_tipdat;  
        comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_longit;  
        comando.Parameters.Add("@param7", SqlDbType.VarChar).Value = p_parcon;  
        comando.Parameters.Add("@param8", SqlDbType.VarChar).Value = p_ordeje;  
        comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_tipelm;
```

```
// id generado

p_codigoEntidadDetalle =
MetodosDatos.EjecutarNomQueryRetornaId(comando);

return p_codigoEntidad > 0;
}

public override bool ActualizarEntidadDetalle(int p_codigoEntidad, int
p_codigoEntidadDetalle, string p_ordsec,

string p_nomcam, string p_tipdat, string p_longit, string p_parcon, string
p_ordeje, int p_tipelm)
{
SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
comando.CommandText =

"UPDATE tbl_entidad_detalle SET
ede_ordsec=@param3,ede_nomcam=@param4,ede_tipdat=@param5,ede_longit=@
param6,ede_parcon=@param7,ede_ordeje=@param8,ede_tipelm=@param9
WHERE ede_codigo=@param1 AND ent_codigo=@param2";

comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidadDetalle;

comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidad;

comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_ordsec;
comando.Parameters.Add("@param4", SqlDbType.VarChar).Value =
p_nomcam;

comando.Parameters.Add("@param5", SqlDbType.VarChar).Value = p_tipdat;
comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_longit;
comando.Parameters.Add("@param7", SqlDbType.VarChar).Value = p_parcon;
comando.Parameters.Add("@param8", SqlDbType.VarChar).Value = p_ordeje;
comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_tipelm;

return MetodosDatos.EjecutarNomQuery(comando);
}
```

```
public override bool EliminarEntidadDetalle(int p_codigoEntidad, int
p_codigoEntidadDetalle)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "DELETE FROM tbl_entidad_detalle WHERE ede_codigo=@param1 AND
ent_codigo=@param2";

    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidadDetalle;

    comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidad;

    return MetodosDatos.EjecutarNomQuery(comando);
}

public override DataTable ObtenerEntidadDetalle()
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_entidad_detalle";
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataTable ObtenerEntidadDetallePorEntidad(int
p_codigoEntidad)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);

    comando.CommandText = string.Format("SELECT * FROM tbl_entidad_detalle
WHERE ent_codigo='{0}' ORDER BY CAST(ede_ordsec AS INT)",
p_codigoEntidad);

    return MetodosDatos.EjecutarComandoSelect(comando);
}
```

```
public override DataRow ObtenerEntidadDetalle(int p_codigoEntidad, int
p_codigoEntidadDetalle)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_entidad_detalle
WHERE ede_codigo={0} AND ent_codigo={1}",
    p_codigoEntidadDetalle, p_codigoEntidad);
    return
    Utiles.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));
}

#endregion

#region Entidad Resultado

public override bool CrearEntidadResultado(int p_codigoEntidad, out int
p_codigoEntidadResultado, string p_descri, DateTime p_fecha,
    DateTime p_horini, DateTime p_horfin, string p_tipope, int p_codeje, int
p_estado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "INSERT INTO tbl_entidad_resultado
(ent_codigo,ere_descri,ere_fecha,ere_horini,ere_horfin,ere_tipope,ere_codeje,ere_es
tado) OUTPUT inserted.ere_codigo
VALUES(@param2,@param3,@param4,@param5,@param6,@param7,@param8,
@param9)";
    comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidad;
    comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_descri;
    comando.Parameters.Add("@param4", SqlDbType.DateTime).Value = p_fecha;
```

```
comando.Parameters.Add("@param5", SqlDbType.Time).Value =
p_horini.ToString("HH:mm");

comando.Parameters.Add("@param6", SqlDbType.Time).Value =
p_horfin.ToString("HH:mm"); ;

comando.Parameters.Add("@param7", SqlDbType.VarChar).Value = p_tipope;
comando.Parameters.Add("@param8", SqlDbType.Int).Value = p_codeje;
comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_estado;

// id generado

p_codigoEntidadResultado =
MetodosDatos.EjecutarNomQueryRetornaId(comando);

return p_codigoEntidadResultado > 0;
}

public override bool ActualizarEntidadResultado(int p_codigoEntidad, int
p_codigoEntidadResultado, string p_descri, DateTime p_fecha,
DateTime p_horini, DateTime p_horfin, string p_tipope, int p_codeje, int
p_estado)
{
SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
comando.CommandText =

"UPDATE tbl_entidad_resultado SET
ere_descri=@param3,ere_fecha=@param4,ere_horini=@param5,ere_horfin=@para
m6,ere_tipope=@param7,ere_codeje=@param8,ere_estado=@param9 WHERE
ere_codigo=@param1 AND ent_codigo=@param2";

comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidadResultado;

comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidad;

comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_descri;
comando.Parameters.Add("@param4", SqlDbType.DateTime).Value = p_fecha;
comando.Parameters.Add("@param5", SqlDbType.Time).Value =
p_horini.ToString("HH:mm"); ;
```

```
comando.Parameters.Add("@param6", SqlDbType.Time).Value =
p_horfin.ToString("HH:mm"); ;

comando.Parameters.Add("@param7", SqlDbType.VarChar).Value = p_tipope;
comando.Parameters.Add("@param8", SqlDbType.Int).Value = p_codeje;
comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_estado;

return MetodosDatos.EjecutarNomQuery(comando);
}

public override bool EliminarEntidadResultado(int p_codigoEntidad, int
p_codigoEntidadResultado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =

        "DELETE FROM tbl_entidad_resultado WHERE ere_codigo=@param1 AND
ent_codigo=@param2";

    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidadResultado;

    comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidad;

    return MetodosDatos.EjecutarNomQuery(comando);
}

public override DataTable ObtenerEntidadResultado()
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_entidad_resultado";
    return MetodosDatos.EjecutarComandoSelect(comando);
}
```

```
public override DataTable ObtenerEntidadResultadoPorFecha(DateTime p_fecha)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_entidad_resultado WHERE
ere_fecha=@param1";
    comando.Parameters.Add("@param1", SqlDbType.DateTime).Value = p_fecha;
    return MetodosDatos.EjecutarComandoSelect(comando);
}
```

```
public override DataTable ObtenerEntidadResultado(int p_codigoEntidad)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM
tbl_entidad_resultado WHERE ent_codigo='{0}'", p_codigoEntidad);
    return MetodosDatos.EjecutarComandoSelect(comando);
}
```

```
public override DataRow ObtenerEntidadResultado(int p_codigoEntidad, int
p_codigoEntidadResultado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM
tbl_entidad_resultado WHERE ere_codigo={0} AND ent_codigo={1}",
    p_codigoEntidadResultado, p_codigoEntidad);
    return
    Utiles.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));
}
```

```
public override List<Dictionary<string, object>>
ObtenerEntidadResultadoCompleto(DateTime p_fecha)
{
```

```
SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);

comando.CommandText = string.Format("SELECT
C.apl_nombre,D.ver_numero,A.* FROM tbl_entidad_resultado A INNER JOIN
tbl_entidad B ON B.ent_codigo=A.ent_codigo INNER JOIN tbl_aplicacion C ON
C.apl_codigo=B.apl_codigo INNER JOIN tbl_version D ON
D.apl_codigo=B.apl_codigo AND D.ver_codigo=B.ver_codigo WHERE
A.ere_fecha BETWEEN CONVERT(datetime,'{0} 00:00:00', 121) AND
CONVERT(datetime,'{0} 23:59:59', 121)", p_fecha.ToString("MM/dd/yyyy"));

return
Utiles.ConvertColeccionGenerica(MetodosDatos.EjecutarComandoSelect(comando)
);
}
```

#endregion

#region Entidad Resultado detalle

```
public override bool CrearEntidadResultadoDetalle(int p_codigoEntidad, int
p_codigoEntidadResultado, out int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle, string p_descri,
DateTime p_fecha, int p_estado)
{
SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
comando.CommandText =
"INSERT INTO tbl_entidad_resultado_detalle
(ent_codigo,ere_codigo,ede_codigo,erd_descri,erd_fecha,erd_estado) OUTPUT
inserted.ere_codigo
VALUES(@param1,@param2,@param3,@param4,@param5,@param6)";
comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;
comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidadResultado;
comando.Parameters.Add("@param3", SqlDbType.Int).Value =
p_codigoEntidadDetalle;
```

```
comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_descri;
comando.Parameters.Add("@param5", SqlDbType.DateTime).Value = p_fecha;
comando.Parameters.Add("@param6", SqlDbType.Int).Value = p_estado;

// id generado

p_codigoEntidadResultadoDetalle =
MetodosDatos.EjecutarNomQueryRetornaId(comando);

return p_codigoEntidadResultadoDetalle > 0;
}

public override bool ActualizarEntidadResultadoDetalle(int p_codigoEntidad, int
p_codigoEntidadResultado, int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle, string p_descri,
    DateTime p_fecha, int p_estado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =

        "UPDATE tbl_entidad_resultado_detalle SET
        erd_descri=@param5,erd_fecha=@param6,erd_estado=@param7 WHERE
        ent_codigo=@param1 AND ere_codigo=@param2 AND erd_codigo=@param3
        AND ede_codigo=@param4";

    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;

    comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidadResultado;

    comando.Parameters.Add("@param3", SqlDbType.Int).Value =
p_codigoEntidadResultadoDetalle;

    comando.Parameters.Add("@param4", SqlDbType.Int).Value =
p_codigoEntidadDetalle;

    comando.Parameters.Add("@param5", SqlDbType.VarChar).Value = p_descri;
    comando.Parameters.Add("@param6", SqlDbType.DateTime).Value = p_fecha;
    comando.Parameters.Add("@param7", SqlDbType.Int).Value = p_estado;

    return MetodosDatos.EjecutarNomQuery(comando);
```

```
}

public override bool EliminarEntidadResultadoDetalle(int p_codigoEntidad, int
p_codigoEntidadResultado, int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle)

{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =

        "DELETE FROM tbl_entidad_resultado_detalle WHERE
ent_codigo=@param1 AND ere_codigo=@param2 AND erd_codigo=@param3
AND ede_codigo=@param4";

    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;

    comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidadResultado;

    comando.Parameters.Add("@param3", SqlDbType.Int).Value =
p_codigoEntidadResultadoDetalle;

    comando.Parameters.Add("@param4", SqlDbType.Int).Value =
p_codigoEntidadDetalle;

    return MetodosDatos.EjecutarNomQuery(comando);
}

public override DataTable ObtenerEntidadResultadoDetalle()

{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_entidad_resultado_detalle";
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataTable ObtenerEntidadResultadoDetalle(int p_codigoEntidad,
int p_codigoEntidadResultado)

{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_entidad_resultado_detalle
WHERE ent_codigo=@param1 AND ere_codigo=@param2";
```

```
        comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;

        comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidadResultado;

        return MetodosDatos.EjecutarComandoSelect(comando);
    }

    public override DataTable ObtenerEntidadResultadoDetalle(int p_codigoEntidad,
int p_codigoEntidadResultado, int p_codigoEntidadDetalle)
    {
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);

        comando.CommandText = "SELECT * FROM tbl_entidad_resultado_detalle
WHERE ent_codigo=@param1 AND ere_codigo=@param2 AND
ede_codigo=@param4";

        comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;

        comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidadResultado;

        comando.Parameters.Add("@param3", SqlDbType.Int).Value =
p_codigoEntidadDetalle;

        return MetodosDatos.EjecutarComandoSelect(comando);
    }

    public override DataRow ObtenerEntidadResultadoDetalle(int p_codigoEntidad,
int p_codigoEntidadResultado, int p_codigoEntidadResultadoDetalle, int
p_codigoEntidadDetalle)
    {
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);

        comando.CommandText = "SELECT * FROM tbl_entidad_resultado_detalle
WHERE ent_codigo=@param1 AND ere_codigo=@param2 AND
erd_codigo=@param3 AND ede_codigo=@param4";

        comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoEntidad;

        comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoEntidadResultado;

        comando.Parameters.Add("@param3", SqlDbType.Int).Value =
p_codigoEntidadResultadoDetalle;
```

```
        comando.Parameters.Add("@param4", SqlDbType.Int).Value =  
p_codigoEntidadDetalle;  
  
        return  
        Utils.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));  
    }  
  
    public override List<Dictionary<string, object>>  
    ObtenerEntidadResultadoDetalleCompleto(int p_codigoEntidad, int  
p_codigoEntidadResultado)  
    {  
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);  
  
        comando.CommandText = string.Format("SELECT B.ede_nomcam,* FROM  
tbl_entidad_resultado_detalle A INNER JOIN tbl_entidad_detalle B ON  
A.ent_codigo = B.ent_codigo AND A.ede_codigo = B.ede_codigo WHERE  
A.ent_codigo = {0} AND A.ere_codigo = {1}",  
p_codigoEntidad,p_codigoEntidadResultado);  
  
        return  
        Utils.ConvertColeccionGenerica(MetodosDatos.EjecutarComandoSelect(comando)  
);  
    }  
  
#endregion  
  
#region Lenguaje  
  
    public override bool CrearLenguaje(out int p_codigoLenguaje, string p_nombre,  
string p_descr, string p_abrevi,  
string p_meteje, int p_estado)  
    {  
        SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);  
  
        comando.CommandText =  
  
        "INSERT INTO tbl_lenguaje  
(len_nombre,len_descr,len_abrevi,len_meteje,len_estado) OUTPUT  
inserted.len_codigo VALUES(@param2,@param3,@param4,@param5,@param6)";  
  
        comando.Parameters.Add("@param2", SqlDbType.VarChar).Value = p_nombre;
```

```
comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_descr;  
comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_abrevi;  
comando.Parameters.Add("@param5", SqlDbType.VarChar).Value = p_meteje;  
comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_estado;
```

```
// id generado
```

```
p_codigoLenguaje = MetodosDatos.EjecutarNomQueryRetornaId(comando);  
return p_codigoLenguaje > 0;  
}
```

```
public override bool ActualizarLenguaje(int p_codigoLenguaje, string p_nombre,  
string p_descr, string p_abrevi,
```

```
string p_meteje, int p_estado)
```

```
{
```

```
SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
```

```
comando.CommandText =
```

```
"UPDATE tbl_lenguaje SET
```

```
len_nombre=@param2,len_descri=@param3,len_abrevi=@param4,len_meteje=@pa  
ram5,len_estado=@param6 WHERE len_codigo=@param1";
```

```
comando.Parameters.Add("@param1", SqlDbType.Int).Value =  
p_codigoLenguaje;
```

```
comando.Parameters.Add("@param2", SqlDbType.VarChar).Value = p_nombre;
```

```
comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_descr;
```

```
comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_abrevi;
```

```
comando.Parameters.Add("@param5", SqlDbType.VarChar).Value = p_meteje;
```

```
comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_estado;
```

```
return MetodosDatos.EjecutarNomQuery(comando);
```

```
}
```

```
public override bool EliminarLenguaje(int p_codigoLenguaje)
```

```
{  
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);  
    comando.CommandText =  
        "DELETE FROM tbl_lenguaje WHERE len_codigo=@param1";  
    comando.Parameters.Add("@param1", SqlDbType.Int).Value =  
        p_codigoLenguaje;  
  
    return MetodosDatos.EjecutarNomQuery(comando);  
}  
  
public override DataTable ObtenerLenguaje()  
{  
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);  
    comando.CommandText = "SELECT * FROM tbl_lenguaje";  
    return MetodosDatos.EjecutarComandoSelect(comando);  
}  
  
public override DataRow ObtenerLenguaje(int p_codigoLenguaje)  
{  
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);  
    comando.CommandText = string.Format("SELECT * FROM tbl_lenguaje  
WHERE len_codigo='{0}'", p_codigoLenguaje);  
    return  
    Utils.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));  
}  
  
#endregion  
  
#region Usuario
```

```
public override bool CrearUsuario(out int p_codigo, string p_nombre, string
p_apelli, string p_cedula, string p_correo,

    string p_celula, string p_usuari, string p_passwd, int p_rolCodigo, int p_estado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =

        "INSERT INTO tbl_usuarios
        (usu_nombre,usu_apelli,usu_cedula,usu_correo,usu_celula,usu_usuari,usu_passwd,r
        ol_codigo,usu_estado) OUTPUT inserted.usu_codigo
        VALUES(@param2,@param3,@param4,@param5,@param6,@param7,@param8,
        @param9,@param10)";

    comando.Parameters.Add("@param2", SqlDbType.VarChar).Value = p_nombre;
    comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_apelli;
    comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_cedula;
    comando.Parameters.Add("@param5", SqlDbType.VarChar).Value = p_correo;
    comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_celula;
    comando.Parameters.Add("@param7", SqlDbType.VarChar).Value = p_usuari;
    comando.Parameters.Add("@param8", SqlDbType.VarChar).Value = p_passwd;
    comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_rolCodigo;
    comando.Parameters.Add("@param10", SqlDbType.Int).Value = p_estado;

    // id generado
    p_codigo = MetodosDatos.EjecutarNomQueryRetornaId(comando);
    return p_codigo > 0;
}

public override bool ActualizarUsuario(int p_codigo, string p_nombre, string
p_apelli, string p_cedula,

    string p_correo, string p_celula, string p_usuari, string p_passwd, int
p_rolCodigo, int p_estado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
```

```
comando.CommandText =

    "UPDATE tbl_usuarios SET
usu_nombre=@param1,usu_apelli=@param2,usu_cedula=@param3,usu_correo=@
param4,usu_celula=@param5,usu_passwd=@param6,rol_codigo=@param7,usu_est
ado=@param8 WHERE usu_codigo=@param9";

comando.Parameters.Add("@param1", SqlDbType.VarChar).Value = p_nombre;
comando.Parameters.Add("@param2", SqlDbType.VarChar).Value = p_apelli;
comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_cedula;
comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_correo;
comando.Parameters.Add("@param5", SqlDbType.VarChar).Value = p_celula;
comando.Parameters.Add("@param6", SqlDbType.VarChar).Value = p_passwd;
comando.Parameters.Add("@param7", SqlDbType.Int).Value = p_rolCodigo;
comando.Parameters.Add("@param8", SqlDbType.Int).Value = p_estado;
comando.Parameters.Add("@param9", SqlDbType.Int).Value = p_codigo;

return MetodosDatos.EjecutarNomQuery(comando);
}

public override bool AnularUsuario(string p_codigoUsuario)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =

        "UPDATE tbl_usuarios SET usu_estado=@param2 WHERE
usu_usuari=@param1";

    comando.Parameters.Add("@param1", SqlDbType.VarChar).Value =
p_codigoUsuario;

    comando.Parameters.Add("@param2", SqlDbType.Int).Value = 1;

    return MetodosDatos.EjecutarNomQuery(comando);
}
```

```
public override DataTable ObtenerUsuarios()
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_usuarios";
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataRow ObtenerUsuario(string p_codigoUsuario)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_usuarios
    WHERE usu_usuari='{0}' AND usu_estado={1}",
    p_codigoUsuario, 0);
    return
    Utiles.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));
}

#endregion

#region Roles

public override bool CrearRoles(out int p_codigo, string p_descri)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "INSERT INTO tbl_roles (rol_descri) OUTPUT
    inserted.rol_codigo VALUES(@param2)";
    comando.Parameters.Add("@param2", SqlDbType.VarChar).Value = p_descri;

    // id generado
    p_codigo = MetodosDatos.EjecutarNomQueryRetornaId(comando);
    return p_codigo > 0;
}
```

```
}

public override bool ActualizarRoles(int p_codigo, string p_descri)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "UPDATE tbl_roles SET rol_descri=@param1 WHERE
rol_codigo=@param2";
    comando.Parameters.Add("@param1", SqlDbType.VarChar).Value = p_descri;
    comando.Parameters.Add("@param2", SqlDbType.Int).Value = p_codigo;

    return MetodosDatos.EjecutarNomQuery(comando);
}

public override bool EliminarRoles(int p_codigoRoles)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "DELETE FROM tbl_roles WHERE rol_codigo=@param1";
    comando.Parameters.Add("@param1", SqlDbType.Int).Value = p_codigoRoles;

    return MetodosDatos.EjecutarNomQuery(comando);
}

public override DataTable ObtenerRoles()
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_roles";
    return MetodosDatos.EjecutarComandoSelect(comando);
}
```

```
public override DataRow ObtenerRoles(int p_codigo)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_roles WHERE
rol_codigo='{0}'", p_codigo);
    return
    Utiles.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));
}
```

#endregion

#region Versión

```
public override bool CrearVersion(out int p_codigoVersion, int
p_codigoAplicacion, string p_numero, int p_estado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "INSERT INTO tbl_version (apl_codigo,ver_numero,ver_estado) OUTPUT
inserted.ver_codigo VALUES(@param1,@param3,@param4)";
    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoAplicacion;
    comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_numero;
    comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_estado;

    // id generado
    p_codigoVersion = MetodosDatos.EjecutarNomQueryRetornaId(comando);
    return p_codigoVersion > 0;
}
```

```
public override bool ActualizarVersion(int p_codigoVersion, int
p_codigoAplicacion, string p_numero, int p_estado)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "UPDATE tbl_version SET ver_numero=@param3,ver_estado=@param4
WHERE apl_codigo=@param1 AND ver_codigo=@param2";
    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoAplicacion;
    comando.Parameters.Add("@param2", SqlDbType.VarChar).Value =
p_codigoVersion;
    comando.Parameters.Add("@param3", SqlDbType.VarChar).Value = p_numero;
    comando.Parameters.Add("@param4", SqlDbType.VarChar).Value = p_estado;

    return MetodosDatos.EjecutarNomQuery(comando);
}

public override bool EliminarVersion(int p_codigoVersion, int
p_codigoAplicacion)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText =
        "DELETE FROM tbl_version WHERE ver_codigo=@param1 AND
apl_codigo=@param2";
    comando.Parameters.Add("@param1", SqlDbType.Int).Value =
p_codigoAplicacion;
    comando.Parameters.Add("@param2", SqlDbType.Int).Value =
p_codigoVersion;

    return MetodosDatos.EjecutarNomQuery(comando);
}
```

```
public override DataTable ObtenerVersion()
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = "SELECT * FROM tbl_version";
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataTable ObtenerVersion(int p_codigoAplicacion)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_version
WHERE apl_codigo='{0}'",
    p_codigoAplicacion);
    return MetodosDatos.EjecutarComandoSelect(comando);
}

public override DataRow ObtenerVersion(int p_codigoVersion, int
p_codigoAplicacion)
{
    SqlCommand comando = MetodosDatos.CrearComando(CadenaConeccion);
    comando.CommandText = string.Format("SELECT * FROM tbl_version
WHERE apl_codigo='{0}' AND ver_codigo={1}",
    p_codigoAplicacion, p_codigoVersion);
    return
    Utiles.ConvertDataTable(MetodosDatos.EjecutarComandoSelect(comando));
}

#endregion

}
```

}

CAPA LÓGICA

APLICACIÓN

```
using System.Linq;
using System.Collections.Generic;
using System.Data;
using AccesoDatos;

namespace CapaLogica
{
    public class Aplicacion : Base
    {
        // Atributos
        private int m_codigoAplicacion;
        private int m_codigoLenguaje;
        private string m_nombre;
        private string m_descri;
        private int m_estado;
        private Aplicacion m_copia;

        // constructor
        public Aplicacion()
        {
        }

        public Aplicacion(int p_codigoAplicacion, int p_codigoLenguaje,
string p_nombre, string p_descri, int p_estado)
        {
            m_codigoAplicacion = p_codigoAplicacion;
            m_codigoLenguaje = p_codigoLenguaje;
            m_nombre = p_nombre;
```

```
m_descri = p_descri;
m_estado = p_estado;
}

// Propiedades

public int CodigoAplicacion
{
    get { return m_codigoAplicacion; }
    set
    {
        OnPropertyChanged("CodigoAplicacion");
        m_codigoAplicacion = value;
        OnPropertyChanged("CodigoAplicacion");
    }
}

public int CodigoLenguaje
{
    get { return m_codigoLenguaje; }
    set
    {
        OnPropertyChanged("CodigoLenguaje");
        m_codigoLenguaje = value;
        OnPropertyChanged("CodigoLenguaje");
    }
}

public string Nombre
{
    get { return m_nombre; }
    set
    {
```

```
OnPropertyChanging("Nombre");
m_nombre = value;

        OnPropertyChanged("Nombre");
    }
}

public string Descri
{
    get { return m_descri; }
    set
    {
OnPropertyChanging("Descri");
m_descri = value;

        OnPropertyChanged("Descri");
    }
}

public int Estado
{
    get { return m_estado; }
    set
    {
OnPropertyChanging("Estado");
m_estado = value;

        OnPropertyChanged("Estado");
    }
}

// Métodos
private static Aplicacion TransFormarObjeto(DataRow p_row)
{
    if (p_row == null) return null;
    object[] objects = p_row.ItemArray;
```

```
        if (objects.Length < 4) return null;

        Aplicacion obj = new Aplicacion((int)objects[0],
        (int)objects[3], (string)objects[1], (string)objects[2], (int)objects[4]);
        return obj;
    }

    // Funciones

    public static bool Insertar(Aplicacion p_aplicacion)
    {
        CapaAccesoDatosBase capaAcceso =
        UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        int value;
        bool estado = capaAcceso.CrearAplicacion(out value,
        p_aplicacion.CodigoLenguaje, p_aplicacion.Nombre, p_aplicacion.Descri,
        p_aplicacion.Estado);
        p_aplicacion.CodigoAplicacion = value;

        return estado;
    }

    public static bool Actualizar(Aplicacion p_aplicacion)
    {
        CapaAccesoDatosBase capaAcceso =
        UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return
        capaAcceso.ActualizarAplicacion(p_aplicacion.CodigoAplicacion,
        p_aplicacion.CodigoLenguaje, p_aplicacion.Nombre, p_aplicacion.Descri,
        p_aplicacion.Estado);
    }
}
```

```
public static bool Anular(int p_codigoAplicacion, int
p_codigoLenguaje)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    return capaAcceso.EliminarAplicacion(p_codigoAplicacion,
p_codigoLenguaje);
}

public static List<Aplicacion> ObtenerAplicacion()
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    return (from DataRow row in
capaAcceso.ObtenerAplicacion().Rows select TransFormarObjeto(row)).ToList();
}

public static List<Aplicacion> ObtenerAplicacion(int
p_codigoAplicacion)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    return
        (from DataRow row in
capaAcceso.ObtenerAplicacion(p_codigoAplicacion).Rows select
TransFormarObjeto(row)).ToList();
}

public static List<Aplicacion> ObtenerAplicacionPorLenguaje(int
p_codigoLenguaje)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
```

```
        return
        (from DataRow row in
capaAcceso.ObtenerAplicacionPorLenguaje(p_codigoLenguaje).Rows select
TransformarObjeto(row))
        .ToList();
    }

    public static Aplicacion ObtenerAplicacion(int p_codigoAplicacion,
int p_codigoLenguaje)
    {
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return
TransformarObjeto(capaAcceso.ObtenerAplicacion(p_codigoAplicacion,
p_codigoLenguaje));
    }

    #region PropertyChanged

    public override void BeginEdit()
    {
        m_copia = new Aplicacion();
        m_copia.m_codigoAplicacion = m_codigoAplicacion;
        m_copia.m_nombre = m_nombre;
        m_copia.m_descri = m_descri;
        m_copia.m_codigoLenguaje = m_codigoLenguaje;
        m_copia.m_estado = m_estado;
    }

    public override void EndEdit()
    {
        m_copia=null;
    }
}
```

```
        public override void CancelEdit()
        {
            if (m_copia != null)
            {
                m_codigoAplicacion = m_copia.m_codigoAplicacion;
                m_nombre = m_copia.m_nombre;
                m_descri = m_copia.m_descri;
                m_codigoLenguaje = m_copia.m_codigoLenguaje;
                m_estado = m_copia.m_estado;
            }

            // eliminar datos guardados
            m_copia = null;

            // notificar para cancelar los modificados
            OnPropertyChanged("");
        }

        #endregion
    }
}

BASE

using System.ComponentModel;

namespace CapaLogica
{
    public abstract class Base : INotifyPropertyChanging, INotifyPropertyChanged,
    IEditableObject
    {
        // Atributos
        private bool m_estadoProceso;
        private bool m_nuevo;

        // Propiedades
```

```
public bool EstadoProceso
{
    get { return m_estadoProceso; }
    set
    {
        OnPropertyChanging("EstadoProceso");
        m_estadoProceso = value;
        OnPropertyChanged("EstadoProceso");
    }
}

public bool Nuevo
{
    get { return m_nuevo; }
    set
    {
        OnPropertyChanging("Nuevo");
        m_nuevo = value;
        OnPropertyChanged("Nuevo");
    }
}

#region INotifyPropertyChanging Members

public event PropertyChangingEventHandler PropertyChanging;

protected void OnPropertyChanging(string propertyName)
{
    if (this.PropertyChanging != null)
    {
        PropertyChanging(this, new PropertyChangingEventArgs(propertyName));
    }
}

#endregion

#region INotifyPropertyChanged Members

public event PropertyChangedEventHandler PropertyChanged;

protected void OnPropertyChanged(string propertyName)
{
    if (this.PropertyChanged != null)
    {
        PropertyChanged(this, new PropertyChangedEventArgs(propertyName));
    }
}

#endregion
```

```
#region IEditableObject Members

public virtual void BeginEdit()
{
}

public virtual void EndEdit()
{
}

public virtual void CancelEdit()
{
}

#endregion
}
```

ENTIDAD

```
using System.Collections.Generic;
using System.Data;
using System.Linq;
using AccesoDatos;

namespace CapaLogica
{
    public class Entidad : Base
    {
        // Atributos
        private int m_codigoEntidad;
        private int m_codigoAplicacion;
        private int m_codigoVersion;
        private string m_namesp;
        private string m_nomven;
        private string m_afirma;
        private string m_operac;
        private string m_ordeje;
        private int m_impjam;
        private int m_impcs;
        private int m_estado;
        private Entidad m_copia;

        // constructor
        public Entidad()
```

```
{  
}
```

```
public Entidad(int p_codigoEntidad, int p_codigoAplicacion, int  
p_codigoVersion, string p_namesp, string p_nomven, string p_afirma, string  
p_operac, string p_ordeje, int p_impjam, int p_impjcs, int p_estado)  
{  
    m_codigoEntidad = p_codigoEntidad;  
    m_codigoAplicacion = p_codigoAplicacion;  
    m_codigoVersion = p_codigoVersion;  
    m_namesp = p_namesp;  
    m_nomven = p_nomven;  
    m_afirma = p_afirma;  
    m_operac = p_operac;  
    m_ordeje = p_ordeje;  
    m_impjam = p_impjam;  
    m_impjcs = p_impjcs;  
    m_estado = p_estado;  
}
```

// Propiedades

```
public int CodigoEntidad  
{  
    get { return m_codigoEntidad; }  
    set  
    {  
        OnPropertyChanging("CodigoEntidad");  
        m_codigoEntidad = value;  
        OnPropertyChanged("CodigoEntidad");  
    }  
}  
  
public int CodigoAplicacion  
{  
    get { return m_codigoAplicacion; }  
    set  
    {  
        OnPropertyChanging("CodigoAplicacion");  
        m_codigoAplicacion = value;  
        OnPropertyChanged("CodigoAplicacion");  
    }  
}  
  
public int CodigoVersion  
{  
    get { return m_codigoVersion; }  
    set  
    {
```

```
OnPropertyChanging("VersionCodigo");  
m_codigoVersion = value;  
OnPropertyChanged("VersionCodigo");  
}  
}
```

```
public string Namesp  
{  
    get { return m_namesp; }  
    set  
    {  
        OnPropertyChanging("Namesp");  
        m_namesp = value;  
        OnPropertyChanged("Namesp");  
    }  
}
```

```
public string Nomven  
{  
    get { return m_nomven; }  
    set  
    {  
        OnPropertyChanging("Nomven");  
        m_nomven = value;  
        OnPropertyChanged("Nomven");  
    }  
}
```

```
public string Afirma  
{  
    get { return m_afirma; }  
    set  
    {  
        OnPropertyChanging("Afirma");  
        m_afirma = value;  
        OnPropertyChanged("Afirma");  
    }  
}
```

```
public string Operac  
{  
    get { return m_operac; }  
    set  
    {  
        OnPropertyChanging("Operac");  
        m_operac = value;  
        OnPropertyChanged("Operac");  
    }  
}
```

```
public string Ordeje
{
    get { return m_ordeje; }
    set
    {
        OnPropertyChanging("Ordeje");
        m_ordeje = value;
        OnPropertyChanged("Ordeje");
    }
}

public int Impxam
{
    get { return m_impjam; }
    set
    {
        OnPropertyChanging("Impjam");
        m_impjam = value;
        OnPropertyChanged("Impjam");
    }
}

public int Impcs
{
    get { return m_impcs; }
    set
    {
        OnPropertyChanging("Impcs");
        m_impcs = value;
        OnPropertyChanged("Impcs");
    }
}

public int Estado
{
    get { return m_estado; }
    set
    {
        OnPropertyChanging("Estado");
        m_estado = value;
        OnPropertyChanged("Estado");
    }
}

// Métodos
private static Entidad TransFormarObjeto(DataRow p_row)
{
    if (p_row == null) return null;
}
```

```
object[] objects = p_row.ItemArray;

if (objects.Length < 10) return null;

Entidad obj = new Entidad((int)objects[0], (int)objects[1], (int)objects[2],
(string)objects[3],
(string)objects[4], (string)objects[5], (string)objects[6], (string)objects[7],
(int)objects[8], (int)objects[9], (int)objects[10]);
return obj;
}

// Funciones
public static bool Insertar(Entidad p_entidad)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    int value;
    bool estado = capaAcceso.CrearEntidad(out value, p_entidad.CodigoAplicacion,
p_entidad.CodigoVersion,
    p_entidad.Namesp, p_entidad.Nomven, p_entidad.Afirma, p_entidad.Operac,
p_entidad.Ordeje, p_entidad.Impxam,
    p_entidad.Impcs, p_entidad.Estado);
    p_entidad.CodigoEntidad = value;
    return estado;
}

public static bool Actualizar(Entidad p_entidad)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    return capaAcceso.ActualizarEntidad(p_entidad.CodigoEntidad,
p_entidad.CodigoAplicacion, p_entidad.CodigoVersion,
    p_entidad.Namesp, p_entidad.Nomven, p_entidad.Afirma, p_entidad.Operac,
p_entidad.Ordeje, p_entidad.Impxam,
    p_entidad.Impcs, p_entidad.Estado);
}

public static bool Eliminar(int p_codigoEntidad, int p_codigoAplicacion, int
p_codigoVersion)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    return capaAcceso.EliminarEntidad(p_codigoEntidad, p_codigoAplicacion,
p_codigoVersion);
}

public static List<Entidad> ObtenerEntidad()
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
```

```
        return (from DataRow row in capaAcceso.ObtenerEntidad().Rows select
TransformarObjeto(row)).ToList();
    }
    public static List<Entidad> ObtenerEntidad(int p_codigoEntidad)
    {
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return (from DataRow row in
capaAcceso.ObtenerEntidad(p_codigoEntidad).Rows select
TransformarObjeto(row)).ToList();
    }
    public static List<Entidad> ObtenerEntidad(int p_codigoEntidad, int
p_codigoAplicacion)
    {
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return (from DataRow row in capaAcceso.ObtenerEntidad(p_codigoEntidad,
p_codigoAplicacion).Rows select TransformarObjeto(row)).ToList();
    }
    public static Entidad ObtenerEntidad(int p_codigoEntidad, int
p_codigoAplicacion, int p_codigoVersion)
    {
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return TransformarObjeto(capaAcceso.ObtenerEntidad(p_codigoEntidad,
p_codigoAplicacion, p_codigoVersion));
    }
    public static IEnumerable<Dictionary<string, object>> ObtenerEntidadGenerico()
    {
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return capaAcceso.ObtenerEntidadCompleto();
    }

    #region PropertyChanged

    public override void BeginEdit()
    {
        m_copia = new Entidad();
        m_copia.m_codigoEntidad = m_codigoEntidad;
        m_copia.m_codigoAplicacion = m_codigoAplicacion;
        m_copia.m_codigoVersion = m_codigoVersion;
        m_copia.m_namesp = m_namesp;
        m_copia.m_nomven = m_nomven;
        m_copia.m_afirma = m_afirma;
        m_copia.m_operac = m_operac;
        m_copia.m_ordeje = m_ordeje;
        m_copia.m_impjam = m_impjam;
        m_copia.m_impes = m_impes;
    }
}
```

```
m_copia.m_estado = m_estado;
}

public override void EndEdit()
{
    m_copia = null;
}

public override void CancelEdit()
{
    if (m_copia != null)
    {
        m_codigoEntidad = m_copia.m_codigoEntidad;
        m_codigoAplicacion = m_copia.m_codigoAplicacion;
        m_codigoVersion = m_copia.m_codigoVersion;
        m_namesp = m_copia.m_namesp;
        m_nomven = m_copia.m_nomven;
        m_afirma = m_copia.m_afirma;
        m_operac = m_copia.m_operac;
        m_ordeje = m_copia.m_ordeje;
        m_impjam = m_copia.m_impjam;
        m_impjcs = m_copia.m_impjcs;
        m_estado = m_copia.m_estado;
    }

    m_copia = null;
    OnPropertyChanged("");
}

#endregion

}
}
```

USUARIOS

```
using System.Collections.Generic;
using System.Data;
using System.Linq;
using AccesoDatos;

namespace CapaLogica
{
    public class Usuario : Base
    {
        // Atributos
    }
}
```

```
private int m_codigoUsuario;
private string m_nombre;
private string m_apellido;
private string m_cedula;
private string m_correo;
private string m_celula;
private string m_acceso;
private string m_password;
private int m_codigoRol;
private int m_estado;
private Usuario m_copia;

// constructor

public Usuario()
{

}

public Usuario(int p_codigoUsuario, string p_nombre, string
p_apellido, string p_cedula, string p_correo,
string p_celula, string p_acceso, string p_password, int
p_codigoRol, int p_estado)
{
    m_codigoUsuario = p_codigoUsuario;
    m_nombre = p_nombre;
    m_apellido = p_apellido;
    m_cedula = p_cedula;
    m_correo = p_correo;
    m_celula = p_celula;
    m_acceso = p_acceso;
    m_password = p_password;
    m_codigoRol = p_codigoRol;
    m_estado = p_estado;
}

// Propiedades

public int CódigoUsuario
{
    get { return m_codigoUsuario; }
    set
    {
        OnPropertyChanging("CódigoUsuario");
        m_codigoUsuario = value;
        OnPropertyChanged("CódigoUsuario");
    }
}
```

```
public string Nombre
{
    get { return m_nombre; }
    set
    {
        OnPropertyChanging("Nombre");
        m_nombre = value;
        OnPropertyChanged("Nombre");
    }
}

public string Apellido
{
    get { return m_apellido; }
    set
    {
        OnPropertyChanging("Apellido");
        m_apellido = value;
        OnPropertyChanged("Apellido");
    }
}

public string Cedula
{
    get { return m_cedula; }
    set
    {
        OnPropertyChanging("Cedula");
        m_cedula = value;
        OnPropertyChanged("Cedula");
    }
}

public string Correo
{
    get { return m_correo; }
    set
    {
        OnPropertyChanging("Correo");
        m_correo = value;
        OnPropertyChanged("Correo");
    }
}

public string Celula
{
    get { return m_celula; }
    set
```

```
        {
OnPropertyChanging("Celula");
m_celula = value;
                OnPropertyChanged("Celula");
        }
    }

    public string Acceso
    {
        get { return m_acceso; }
        set
        {
OnPropertyChanging("Acceso");
m_acceso = value;
                OnPropertyChanged("Acceso");
        }
    }

    public string Password
    {
        get { return m_password; }
        set
        {
OnPropertyChanging("Password");
m_password = value;
                OnPropertyChanged("Password");
        }
    }

    public int CódigoRol
    {
        get { return m_codigoRol; }
        set
        {
OnPropertyChanging("CódigoRol");
m_codigoRol = value;
                OnPropertyChanged("CódigoRol");
        }
    }

    public int Estado
    {
        get { return m_estado; }
        set
        {
OnPropertyChanging("Estado");
m_estado = value;
                OnPropertyChanged("Estado");
        }
    }
}
```

```
}

// Métodos
private static Usuario TransFormarObjeto(DataRow p_row)
{
    if (p_row == null) return null;
    object[] objects = p_row.ItemArray;

    if (objects.Length < 8) return null;

    var usuario = new Usuario((int) objects[0], (string) objects[1],
(string) objects[2],
                        (string) objects[3], (string) objects[4], (string)
objects[5], (string) objects[6], (string) objects[7],
                        (int) objects[8], (int) objects[9]);

    return usuario;
}

// Funciones
public static bool Insertar(Usuario p_usuario)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    int value;
    bool estado = capaAcceso.CrearUsuario(out value,
p_usuario.Nombre, p_usuario.Apellido, p_usuario.Cedula,
                        p_usuario.Correo,
                        p_usuario.Celula, p_usuario.Acceso,
p_usuario.Password, p_usuario.CodigoRol, p_usuario.Estado);
    p_usuario.CodigoUsuario = value;
    return estado;
}

public static bool Actualizar(Usuario p_usuario)
{
    CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
    return
capaAcceso.ActualizarUsuario(p_usuario.CodigoUsuario, p_usuario.Nombre,
p_usuario.Apellido, p_usuario.Cedula,
                        p_usuario.Correo,
                        p_usuario.Celula, p_usuario.Acceso,
p_usuario.Password, p_usuario.CodigoRol, p_usuario.Estado);
}

public static bool Anular(string p_usuario)
{

```

```
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return capaAcceso.AnularUsuario(p_usuario);
    }

    public static Usuario ObtenerUsuario(string p_usuario)
    {
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return
TransformarObjeto(capaAcceso.ObtenerUsuario(p_usuario));
    }

    public static List<Usuario> ObtenerTodosUsuario()
    {
        CapaAccesoDatosBase capaAcceso =
UsarCapaAccesoDatos.ObtenerCapaAccesoDatos();
        return (from DataRow row in
capaAcceso.ObtenerUsuarios().Rows select TransformarObjeto(row)).ToList();
    }

    #region PropertyChanged

    public override void BeginEdit()
    {
        m_copia = new Usuario();
        m_copia.m_codigoUsuario = m_codigoUsuario;
        m_copia.m_nombre = m_nombre;
        m_copia.m_apellido = m_apellido;
        m_copia.m_cedula = m_cedula;
        m_copia.m_correo = m_correo;
        m_copia.m_celula = m_celula;
        m_copia.m_acceso = m_acceso;
        m_copia.m_password = m_password;
        m_copia.m_codigoRol = m_codigoRol;
        m_copia.m_estado = m_estado;
    }

    public override void EndEdit()
    {
        m_copia= null;
    }

    public override void CancelEdit()
    {
        if (m_copia != null)
        {
            m_codigoUsuario = m_copia.m_codigoUsuario;
            m_nombre = m_copia.m_nombre;
        }
    }
}
```

```
m_apellido = m_copia.m_apellido;
m_cedula = m_copia.m_cedula;
m_correo = m_copia.m_correo;
m_celula = m_copia.m_celula;
m_acceso = m_copia.m_acceso;
m_password = m_copia.m_password;
m_codigoRol = m_copia.m_codigoRol;
m_estado = m_copia.m_estado;
}

m_copia = null;
OnPropertyChanged("");
}

#endregion

}
}
```

EJECUTAR ENTIDAD

UNIT TEST

```
using System;
using System.Collections.Generic;
using CapaLogica;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using TestStack.White;
using TestStack.White.UIItems;
using TestStack.White.UIItems.Finders;
using TestStack.White.UIItems.ListBoxItems;
using TestStack.White.UIItems.WindowItems;

namespace EjecutarEntidad
{
    [TestClass]
    public class UnitTestIU
    {
        [TestMethod]
        public bool Main(Entidad entidad)
        {
            EntidadResultado ent = null;
            Application application = null;
            try
```

```
{  
    // verificar  
    if (entidad == null) return false;  
  
    // inicializar  
    ent = new EntidadResultado();  
    ent.CodigoEntidad = entidad.CodigoEntidad;  
    ent.Horini = DateTime.Now;  
    ent.Tipope = entidad.Afirma;  
    ent.Fecha = DateTime.Now;  
    ent.Descri = "Entidad Procesada";  
  
    // crear entidad  
    if (!EntidadResultado.Insertar(ent)) return false;  
  
    // mostrar aplicación  
    application = Application.Launch(string.Format(@"{0}\{1}", entidad.Namesp,  
entidad.Nomven));  
  
    // listado de imagenes  
    List<Window> lista = application.GetWindows();  
    Window window =  
application.GetWindow(entidad.Operac);  
    window.WaitWhileBusy();  
  
    // obtener entidades  
    EjecutarEntidadesDetalle(ent, window, application);  
  
    // cerrar ventana  
    lista = application.GetWindows();  
    if (lista != null && lista.Count > 0)  
    {  
        for (int i = 0; i < lista.Count; i++)  
        {  
            Window fmMensaje = lista[i];  
            fmMensaje.WaitWhileBusy();  
            fmMensaje.Close();  
            fmMensaje.WaitWhileBusy();  
        }  
    }  
  
    // actualizar  
    ent.Horfin = DateTime.Now;  
}  
catch (Exception e)  
{  
    ent.Descri = e.Message;  
    ent.Horfin = DateTime.Now;  
}
```

```
        finally
        {
            if (ent != null) EntidadResultado.Actualizar(ent);
            if (application != null) application.Close();
        }

        // estado
        return true;
    }

    /// <summary>
    /// Ejecutar campos de ventana
    /// </summary>
    /// <param name="entidadResultado">Pantalla Inicial</param>
    private bool EjecutarEntidadesDetalle(EntidadResultado
entidadResultado, Window window, Application application)
    {
        // lista de items
        List<EntidadDetalle> detalles =
EntidadDetalle.ObtenerEntidadDetallePorEntidad(entidadResultado.CodigoEntidad);
        if (detalles == null || detalles.Count == 0) return false;

        // fecha actual.
        string fechaActual =
DateTime.Now.ToString("yyyyMMdd_hhmmss");

        // recorrer items
        foreach (var item in detalles)
        {
            // inicializar
            EntidadResultadoDetalle detalle = new EntidadResultadoDetalle();
            detalle.CodigoEntidad = item.CodigoEntidad;
            detalle.CodigoEntidadResultado = entidadResultado.CodigoEntidadResultado;
            detalle.CodigoEntidadDetalle = item.CodigoEntidadDetalle;
            detalle.Fecha = DateTime.Now;
            detalle.Descri = string.Format("Elemento {0} ejecutado de la ventana {1}",
item.Nomcam, window.Name);

            // procesar
            try
            {
                // control de tipo MessageBox cerrar y continuar
                if (item.Tipelm == 5)
                {
                    List<Window> lista = application.GetWindows();
                    if (lista != null && lista.Count > 1)
                    {
                        Window m = lista[lista.Count - 1];
                        if (m != null)
```

```
{
    m.Close();
}
}
// notificar a usuario
window.WaitWhileBusy();
continue;
}
else
{
    // cantidad de ventanas
    List<Window> lista = application.GetWindows();
    window = lista[lista.Count - 1];
}
// tipo de campo
var campo = window.Get(SearchCriteria.ByAutomationId(item.Nomcam));
if (campo == null)
{
    detalle.Descri = string.Format("Elemento {0} no encontrado en la ventana
Actual {1}", item.Nomcam, window.Name);
    continue;
}

// validar por tipo de campo
Random alatorio = new Random();
switch (item.Tipelm)
{
    case 0:
        ((Button)campo).Click();
        break;
    case 1:
        bool value = alatorio.Next(0, 1) == 1;
        ((CheckBox)campo).Checked = value;
        detalle.Descri += " , Valor Ejecutado " + value.ToString();
        break;
    case 2:
        ComboBox control = ((ComboBox)campo);
        control.Select(alatorio.Next(0, control.Items.Count));
        detalle.Descri += " , Valor Ejecutado " + control.SelectedItemText;
        break;
    case 3:
        string valor = "";
        if (item.Parcon.Equals("1"))// no ejecuta valor por defecto
        {
            for (int i = 0; i < Convert.ToInt32(item.Longit); i++)
            {
                if (item.Tipdat.Equals("0"))// alfa numerico
                {
                    valor += (char)alatorio.Next(48, 57);
                }
            }
        }
    }
```

```
    }  
    else  
        valor += (char)alatorio.Next(97, 122);  
    }  
}  
else valor = item.Ordeje;  
campo.Enter(valor);  
detalle.Descri += " , Valor Ejecutado " + valor;  
break;  
case 4:  
    value= alatorio.Next(0, 1) == 1;  
    ((RadioButton)campo).IsSelected = value;  
    detalle.Descri += " , Valor Ejecutado " + value.ToString();  
    break;  
default:  
    continue;  
}  
  
// notificar a usuario  
window.WaitWhileBusy();  
}  
catch (Exception e)  
{  
    detalle.Descri = e.Message;  
}  
finally  
{  
    if (detalle != null) EntidadResultadoDetalle.Insertar(detalle);  
}  
}  
  
// estado  
return true;  
}
```



CARRERA DE ANÁLISIS DE SISTEMAS

**AUTOMATIZACIÓN DEL PROCESO DE EJECUCIÓN DE PRUEBAS
AUTOMÁTICAS UNITARIAS DE APLICATIVOS PARA LA EMPRESA
BIOLOGICSOFT UBICADA EN LA CIUDAD DE QUITO.**

MANUAL DE INSTALACIÓN

AUTOR: Caguano Millingalle Lucía Janeth

DIRECTOR: Ing. Basantes Basantes Jaime Neptalí

Quito, 2018

INDICE

MANUAL DE INSTALACIÓN

INSTALACIÓN DE VISUAL STUDIO	3
SELECCIONAR COMPONENTES.....	3
SELECCIONAR COMPONENTES INDIVIDUALES	4
SELECCIONAR PAQUETES DE IDIOMA.....	4
VENTANA INSTALANDO.....	5
VENTANA BIENVENIDA.....	5
SELECCIONAR TEMA.....	6
CONFIGURANDO TEMA	6
VENTANA DE INICIO.....	7
INSTALACIÓN DE SQL SERVER.....	7
SELECCIONAR IDIOMA DEL PROGRAMA.....	8
DESCARGANDO MEDIOS	8
DESCARGA DE MEDIOS COMPLETA	9
INSTALADOR SSMS	9
INSTALAR SSMS.....	9
INSTALANDO SSMS 17.4.....	10
INSTALANDO SSMS PAQUETE, GENERAL.....	10
INSTALACIÓN COMPLETA	11
SQL SERVER INSTALADO	11

INSTALACIÓN DE VISUAL STUDIO

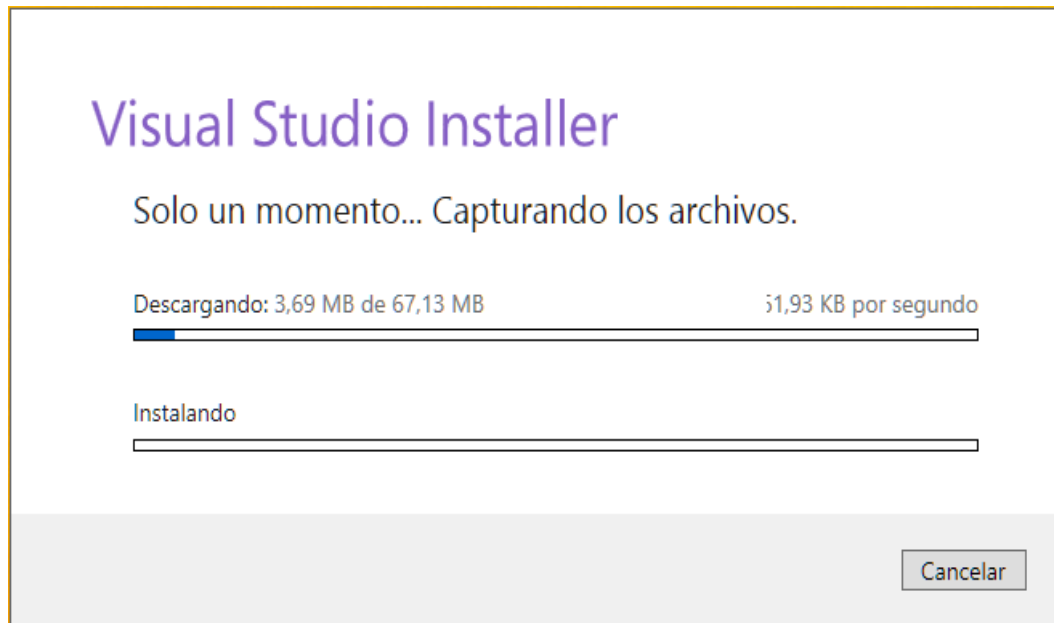


Figura 66: Descarga de Visual Studio

SELECCIONAR COMPONENTES

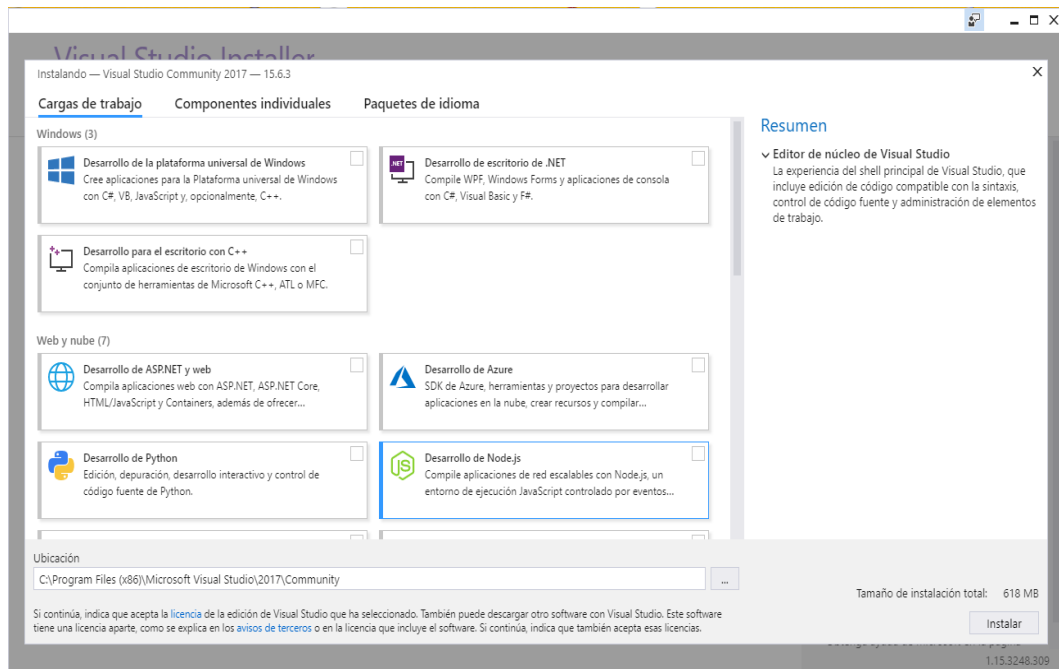


Figura 67: Selecciona Cargas de Trabajo

SELECCIONAR COMPONENTES INDIVIDUALES

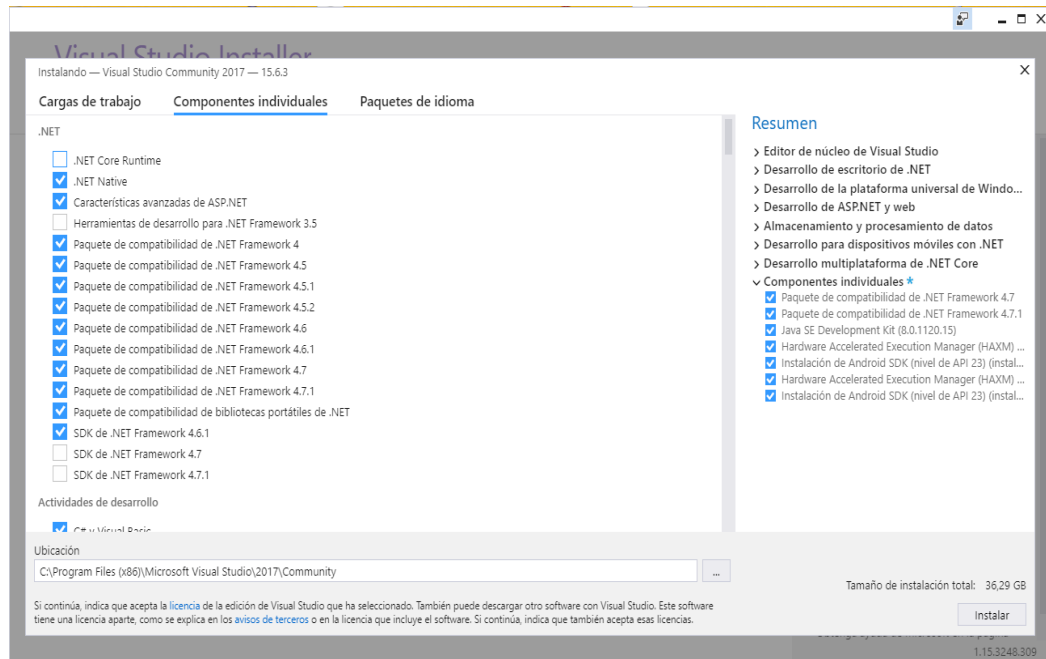


Figura 68: Seleccionar componentes individuales

SELECCIONAR PAQUETES DE IDIOMA

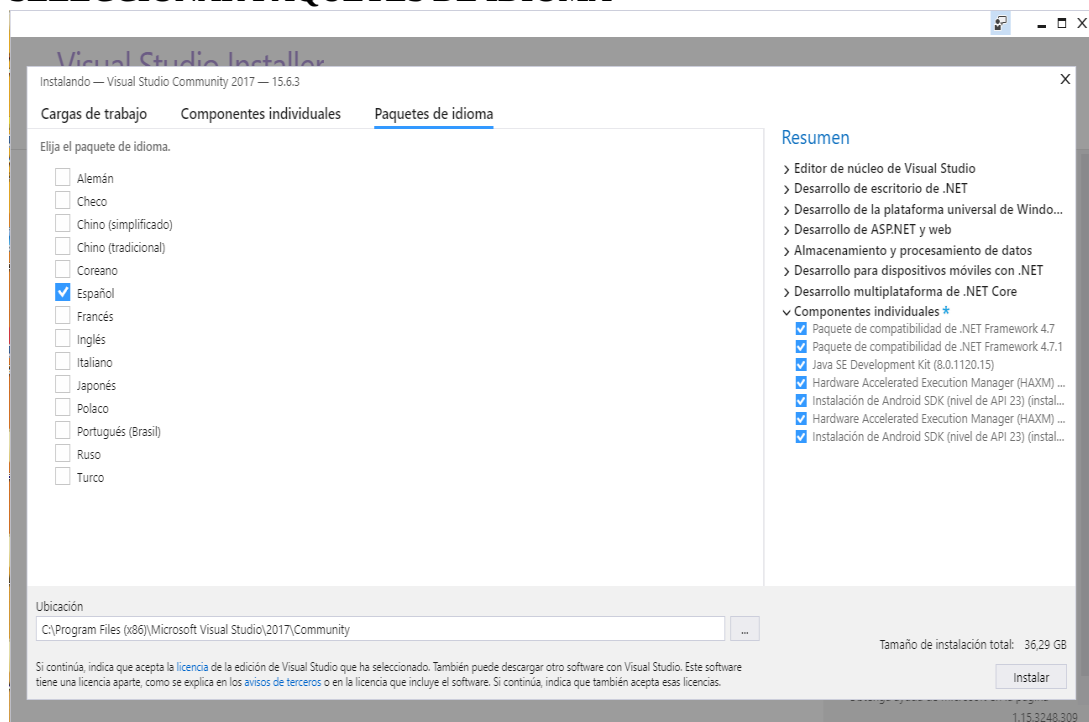


Figura 69: Seleccionar paquetes de idioma

VENTANA INSTALANDO

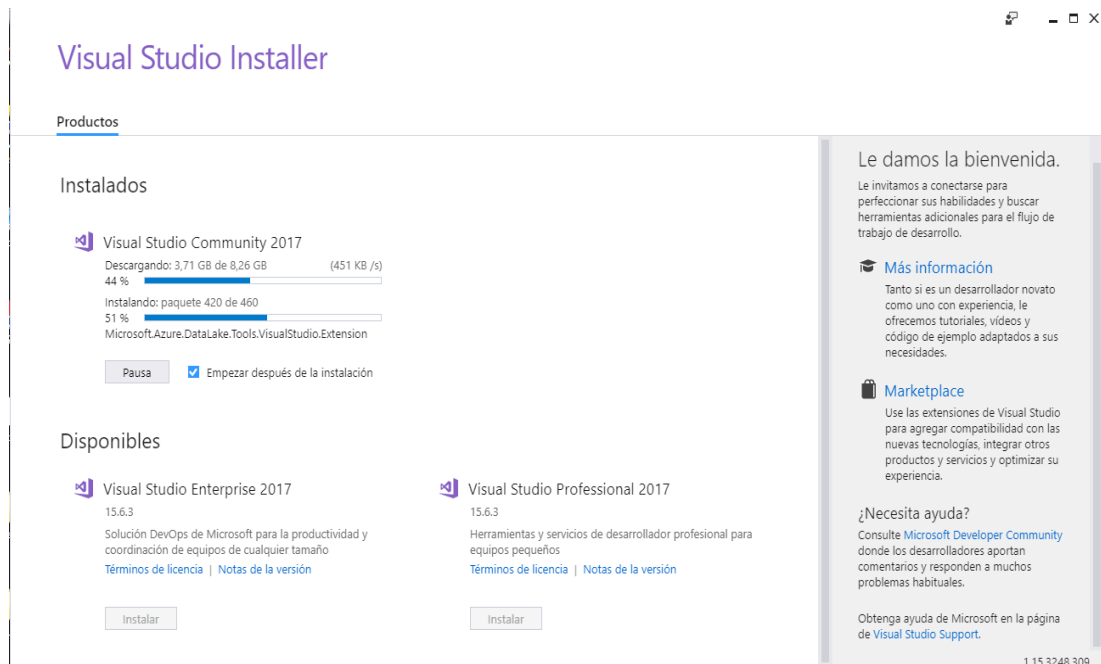


Figura 70: Instalación de productos

VENTANA BIENVENIDA

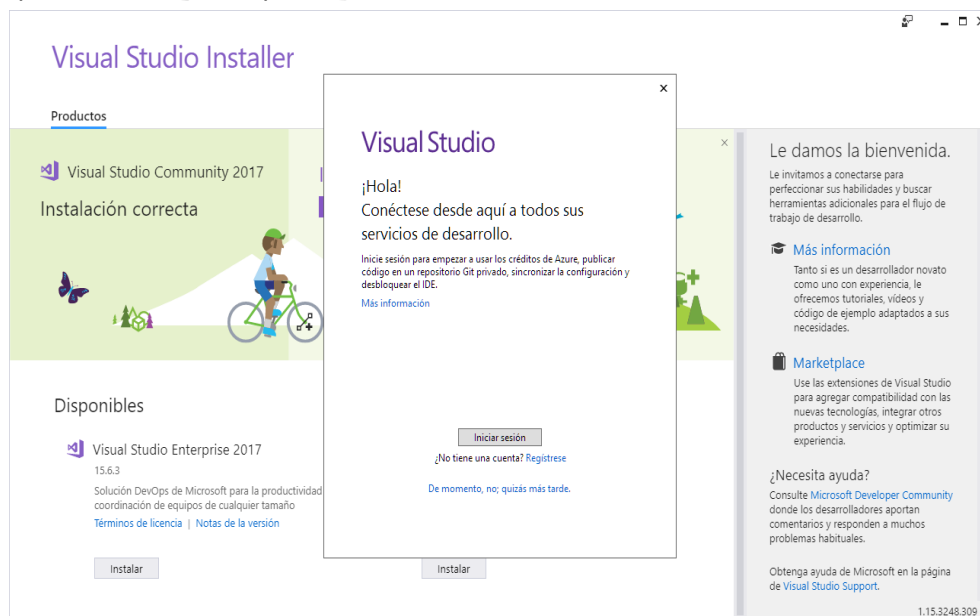


Figura 71: Instalación correcta - Iniciar Sesión

SELECCIONAR TEMA

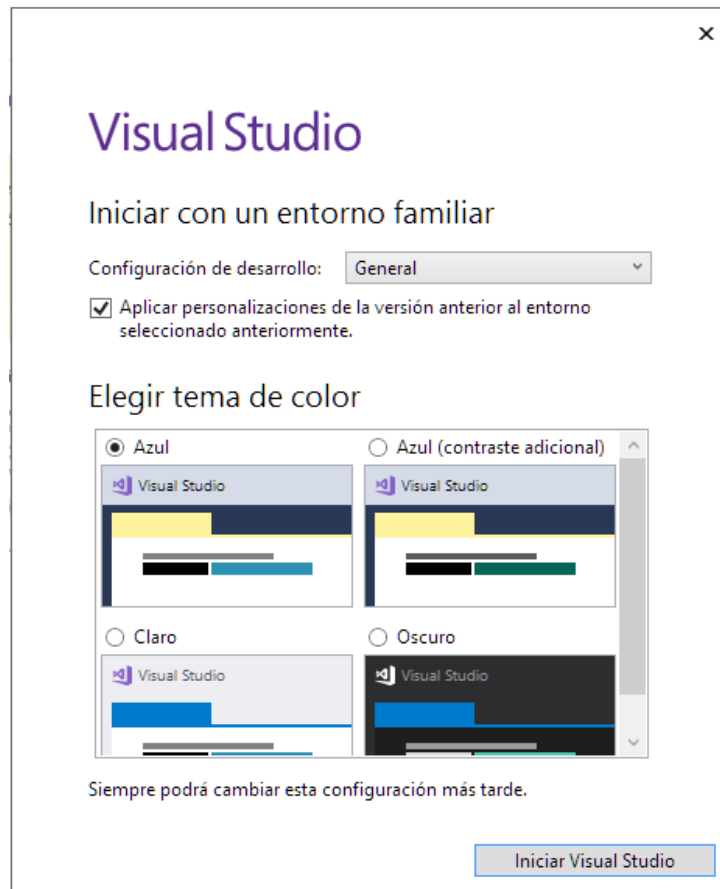


Figura 72: Configurar visual – Colores

CONFIGURANDO TEMA

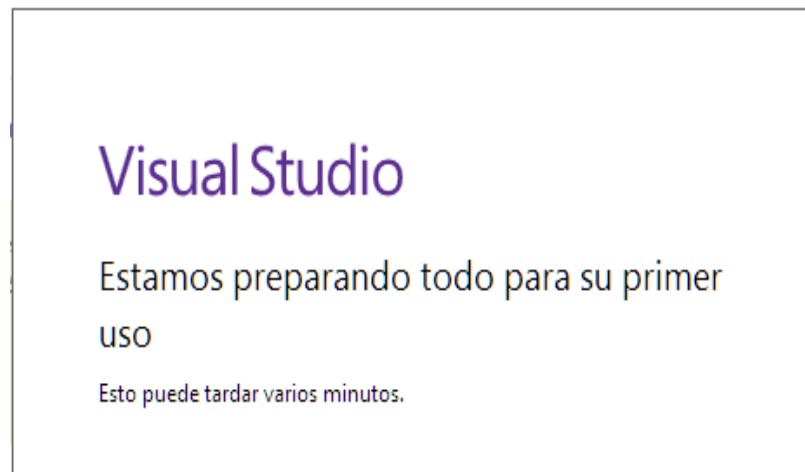


Figura 73: Visual ya casi listo

VENTANA DE INICIO

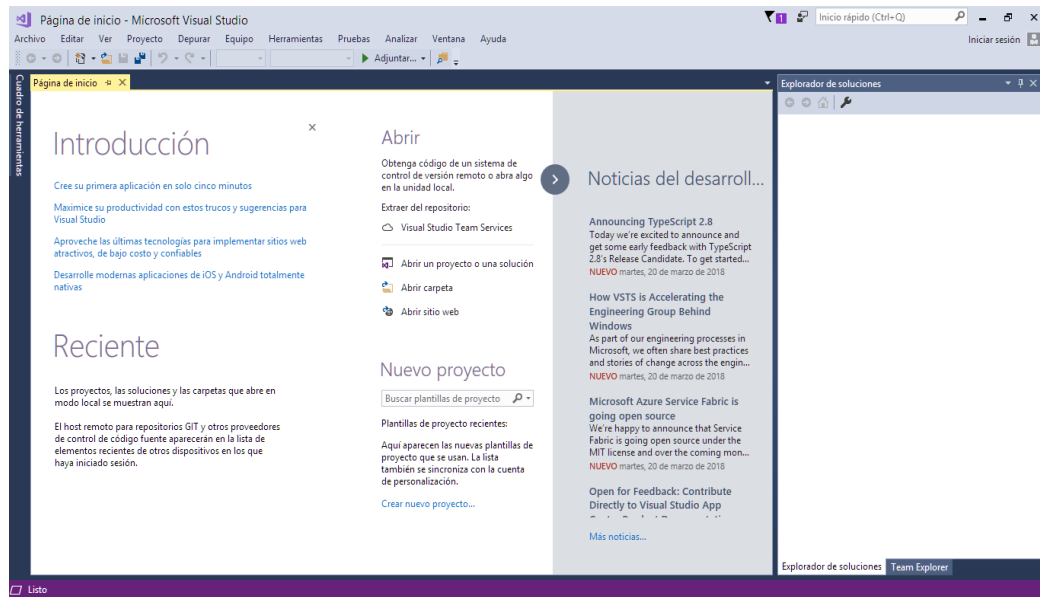


Figura 74: Listo para iniciar

INSTALACIÓN DE SQL SERVER

INSTALADOR SQL

 SQLServer2017-SSEI-Expr	08/01/2018 21:55	Aplicación	4.885 KB
---	------------------	------------	----------

Figura 75: Instalador de SQL Server

INSTALANDO SQL SERVER

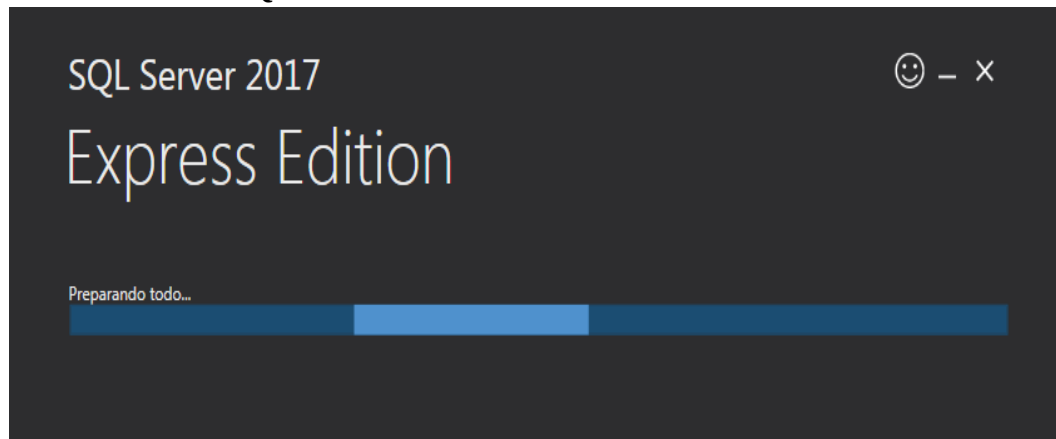


Figura 76: Instalando SQL Server

SELECCIONAR IDIOMA DEL PROGRAMA

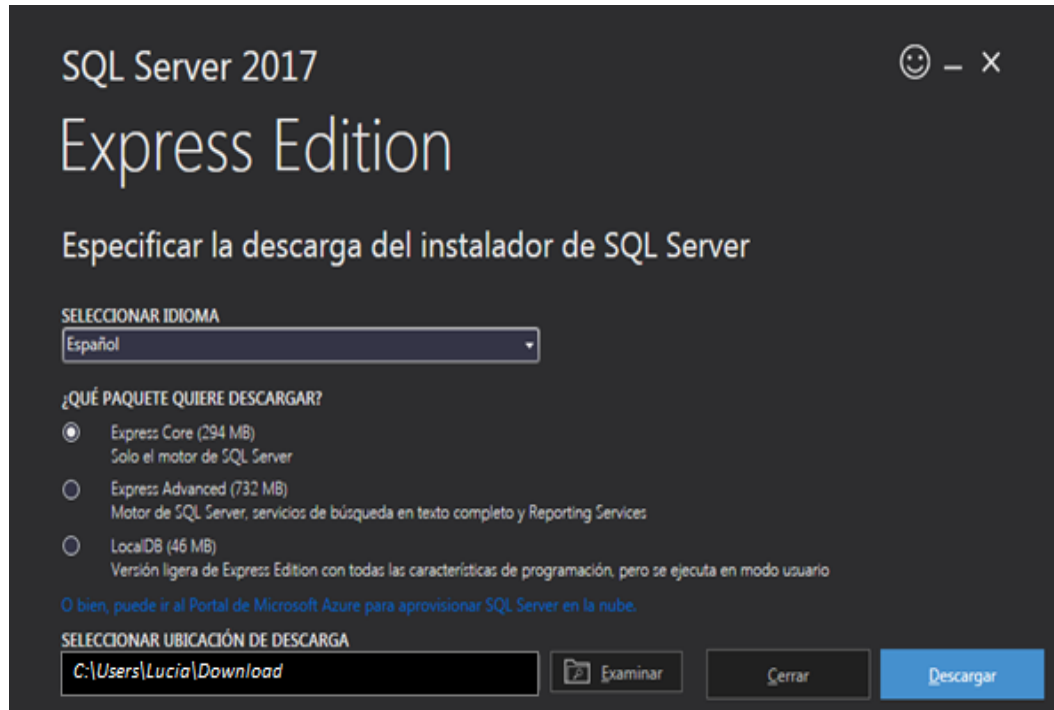


Figura 77: Selecciona idioma y paquete de descarga

DESCARGANDO MEDIOS

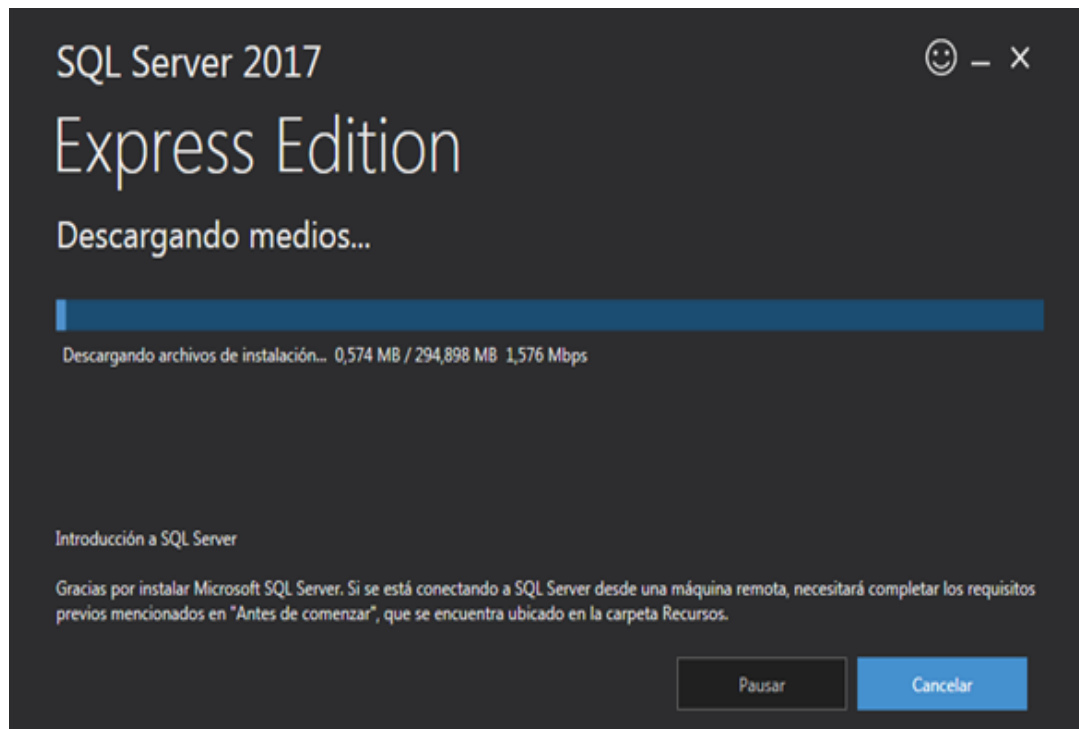


Figura 78: Descarga de Archivos para instalación

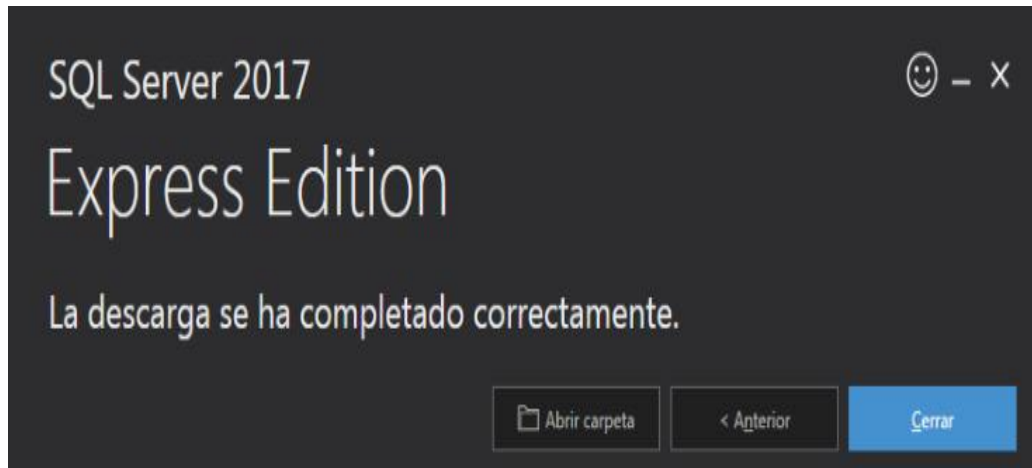
DESCARGA DE MEDIOS COMPLETA

Figura 79: Descarga completa de SQL Server

INSTALADOR SSMS

 SSMS-Setup-ENU	08/01/2018 22:34	Aplicación	821.371 KB
--	------------------	------------	------------

Figura 80: Instalador de SSMS

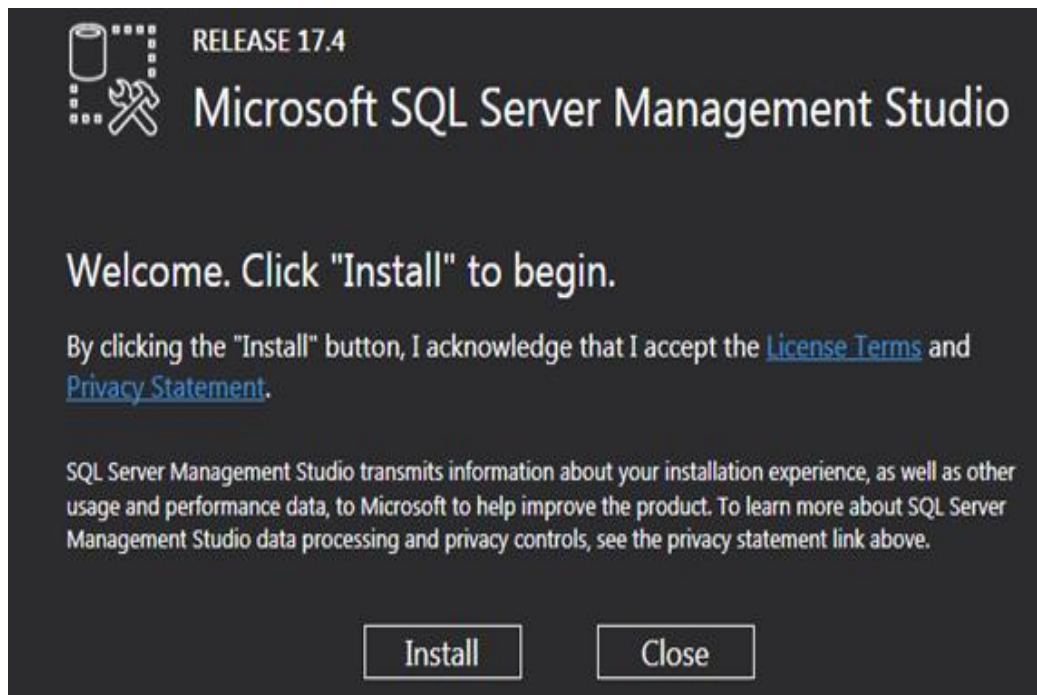
INSTALAR SSMS

Figura 81: Instalar o cerrar SSMS

INSTALANDO SSMS 17.4

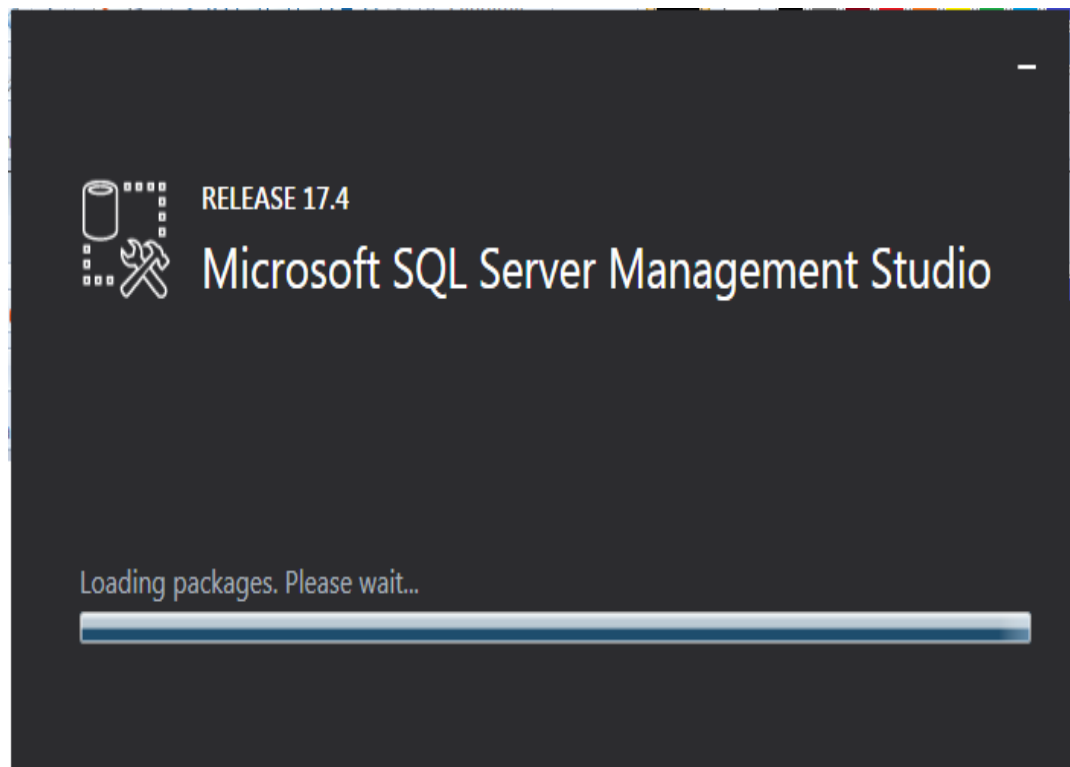


Figura 82: Instalando SSMS

INSTALANDO SSMS PAQUETE, GENERAL

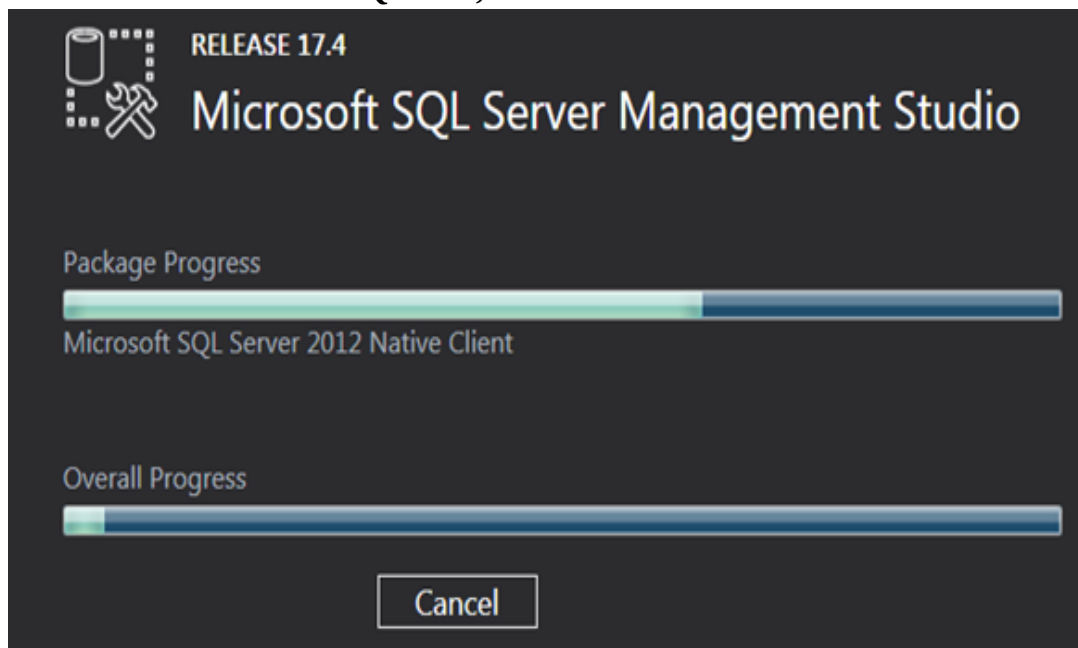


Figura 83: Instalando paquetes - progreso general

INSTALACIÓN COMPLETA

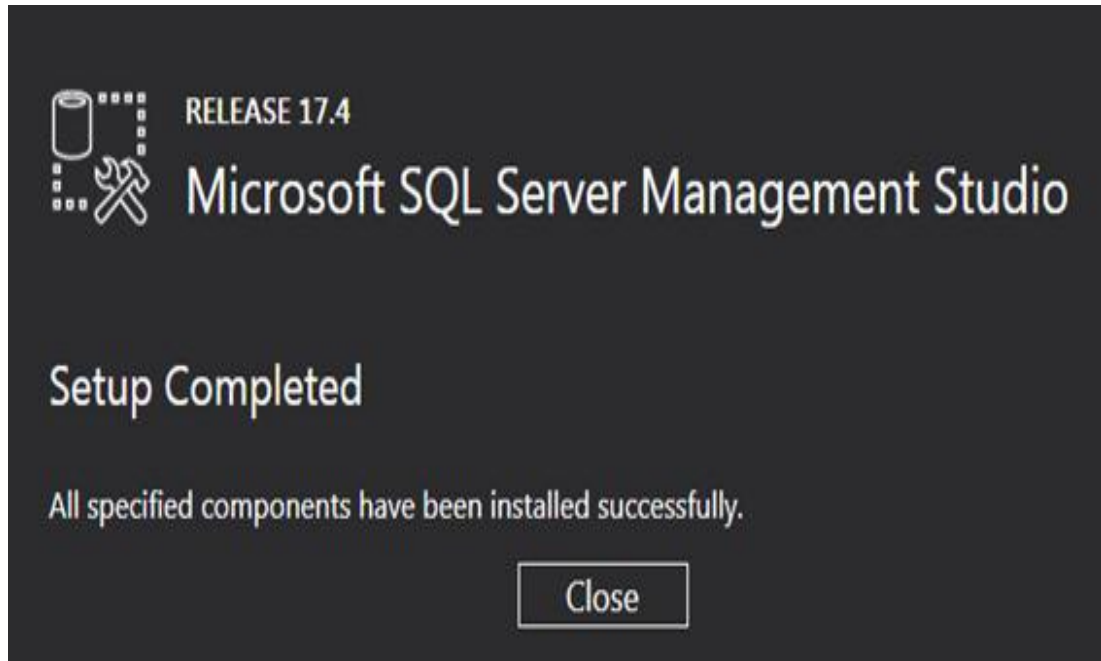


Figura 84: Configuración Completa

SQL SERVER INSTALADO

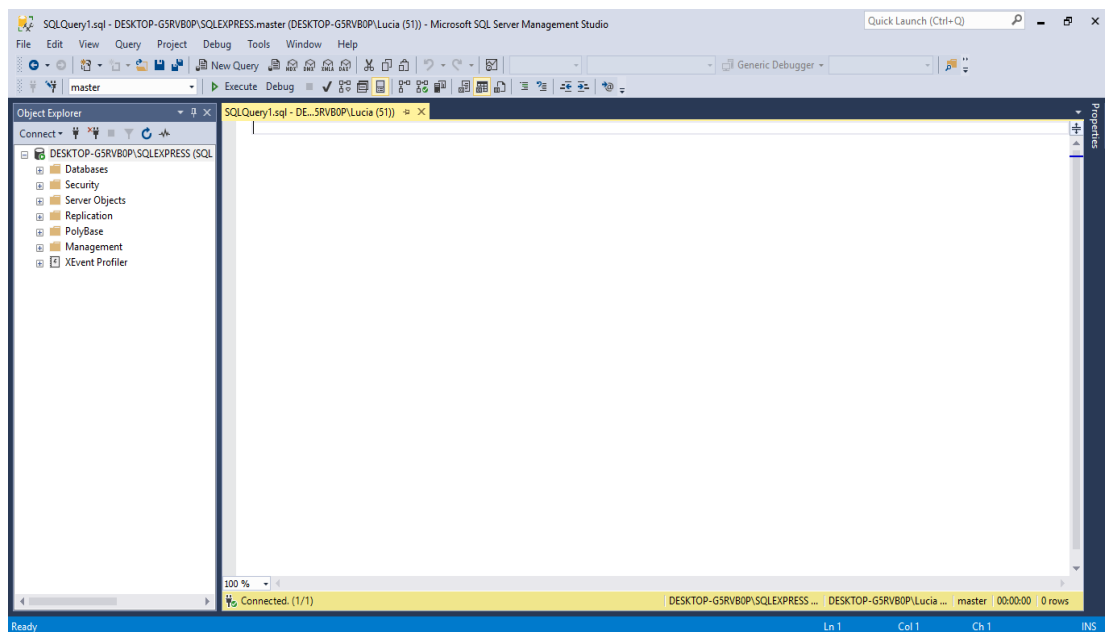


Figura 85: En este gráfico se encuentra la interfaz del SQL Server 2017

Urkund Analysis Result

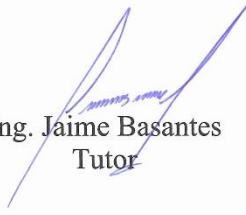
Analysed Document: LUCIA CAGUANO.pdf (D37010993)
Submitted: 3/27/2018 4:23:00 PM
Submitted By: lucyjanth_1993@hotmail.com
Significance: 9 %

Sources included in the report:

Valle_Toromoreno_Edgar_Sistemas.pdf (D15719220)
Kleber_Gonzalo_Vargas_Padilla_Sistemas.pdf (D19695718)
Rodríguez_Campaña_Tania_Sistemas.pdf (D15719890)
PROYECTO TERMINADO PORRAS ESCOBAR.pdf (D19727140)
TESIS_RICARDO_SIGCHA_2017.pdf (D26603505)
Tesis J.A.pdf (D19622293)
Sistematización del Proceso de titulación Freddy Narvaez.pdf (D26664465)
TESIS.pdf (D19612280)
Sistematización del Proceso de titulación Freddy Narvaez.pdf (D26664041)
Rodríguez_Campaña_Tania_Sistemas.pdf (D15723301)
<https://lovelessisma.wordpress.com/2010/07/27/webservice-aplicacion-sql-parte2/>
<https://www.safaribooksonline.com/library/view/adonet-in-a/0596003617/ch04s04.html>

Instances where selected sources appear:

60



Ing. Jaime Basantes
Tutor

www.biologicssoft.com
info@biologicssoft.com

biologicssoft
mobile development

CERTIFICADO DE FUNCIONAMIENTO E IMPLEMENTACIÓN

Quito, 17 de Mayo del 2018.

Señores

INSTITUTO TECNOLÓGICO SUPERIOR CORDILLERA

Presente.-

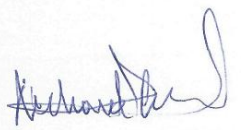
De mis consideraciones,

Me permito emitir el presente certificado correspondiente a la entrega e implementación del Software Desarrollado en el Instituto Tecnológico Superior Cordillera, ya que ha cumplido con los requerimientos solicitados por parte de nuestra institución (Biologicssoft) y ha permitido implantar el sistema de pruebas unitarias en los departamentos de Investigación & Desarrollo y Aseguramiento y Control de Calidad, de manera profesional para la Srta. **Lucía Janeth Caguano Millingalle** con cédula de identidad **1720385374**.

El trabajo sobre **AUTOMATIZACIÓN DEL PROCESO DE EJECUCIÓN DE PRUEBAS AUTOMÁTICAS UNITARIAS DE APLICATIVOS PARA LA EMPRESA BIOLOGICSOFT UBICADA EN LA CIUDAD DE QUITO**. Se encuentra terminado e implantado satisfactoriamente en la institución desde el 01 de Mayo del 2018.

Es todo cuanto puedo decir en honor a la verdad.

Atentamente,



Richard Cueva Alvarado
GERENTE GENERAL

Biologic-Com
RUC. 1792625416001



Fabricio Tamayo Proaño
DIRECTOR TI

Tel: (593 - 2) 290 - 6287
Telefax: (593 - 2) 255 - 4117

Quito - Ecuador - Sudamérica

INSTITUTO TECNOLÓGICO SUPERIOR CORDILLERA

ANÁLISIS DE SISTEMAS

ORDEN DE EMPASTADO

Una vez verificado el cumplimiento de los requisitos establecidos para el proceso de Titulación, se **AUTORIZA** realizar el empastado del trabajo de titulación, del alumno(a) **CAGUANO MILLINGALLE LUCÍA JANETH**, portador de la cédula de identidad N° 172038537-4, previa validación por parte de los departamentos facultados.

Quito, 25 de abril del 2018



Sra. Mariela Balseca

CAJA VISTO FINANCIERO



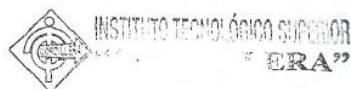
Ing. Johnny Coronel

DELEGADO DE LA UNIDAD DE TITULACIÓN

Análisis de Sistemas



Ing. William Parra
BIBLIOTECA



25 ABR 2018

9,51 BS
COORDINACIÓN PRÁCTICAS

Ing. Samira Villalba

PRÁCTICAS PREPROFESIONALES



Ing. Hugo Heredia

DIRECTOR DE CARRERA

DIRECCIÓN DE CARRERA



Tgla. Carolina Guerra
SECRETARIA ACADÉMICA