ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMATICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN SISTEMAS

"ANALISIS COMPARATIVO ENTRE LOS FRAMEWORKS DE INYECCIÓN DE DEPENDENCIAS UNITY 2.0 Y SPRING.NET. CASO PRÁCTICO: ERPE"

TESIS DE GRADO

PREVIA A LA OBTENCION DEL TÍTULO DE:
INGENIERO EN SISTEMA INFORMÁTICOS

PRESENTADO POR:
RAMIRO JAVIER LUGMANIA VACA
HENRY MAURICIO VILLA YÁNEZ

RIOBAMBA – ECUADOR
2012

A G R A D E C I M I E N T O

Expresamos nuestros mas sinceros agradecimientos, primero a Dios por darnos la oportunidad de concluir nuestros estudios, a todos y cada uno de nuestros familiares que con su apoyo diario nos ha permitido el cumplimiento de uno de nuestros objetivos en la vida, a todos nuestros maestros que cada día con sus enseñanzas y experiencias nos han encaminado hacia un futuro lleno de oportunidades; y finalmente, de manera especial al Dr. Julio Santillán quien nos ofrecieron todas las facilidades para iniciar y finalizar con éxito el presente trabajo.

D E D I C A T O R I A

La fe, el esfuerzo y optimismo dedicado a lo largo de los años de estudio, son el fruto de quienes creyeron en nosotros, apoyándonos en todo sentido extendiendo la mano a través de la educación.

El presente trabajo de investigación está dedicado a nuestros queridos padres, quienes siempre han estado en los momentos más difíciles apoyándonos y dándonos ánimos cuando mas lo necesitamos.

HENRY VILLA Y.

RAMIRO LUGMANIA V.

FIRMAS DE RESPONSABILIDAD

NOMBRE	FIRMAS	FECHA
Ing. Iván Menes **DECANO DE LA FACULTAD DE INFORMATICA Y ELECTRONICA**	_____	_____
Ing. Raúl Rosero **DIRECTOR DE ESCUELA DE INGENIERIA EN SISTEMAS**	_____	_____
Dr. Julio Santillán **DIRECTOR DE TESIS**	_____	_____
Ing. Jorge Menendez **MIEMBRO**	_____	_____
Lic. Carlos Rodríguez **DIRECTOR DEL CENTRO DE DOCUMENTACION**	_____	_____

NOTA DE LA TESIS: _____

RESPONSABILIDAD DE LOS AUTORES

"*Nosotros, HENRY MAURICIO VILLA YÁNEZ Y RAMIRO JAVIER LUGMANIA VACA, somos responsables de las ideas, doctrinas y resultados expuestos en este Trabajo de Investigación; y el patrimonio intelectual de la Tesis de Grado pertenece a la ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO"*

……. …….

Henry Mauricio Villa Yánez Ramiro Javier Lugmania Vaca

ÍNDICE GENERAL

CAPÍTULO II

ÍNDICE DE ABREVIATURAS Y ACRÓNIMOS

A

ASP Active Server Pages (Páginas de Servidor Activas)

B

BD Base de Datos

C

CLR Common Languaje Runtime (Lenguaje Común de Tiempos de Ejecución)

CSHTML C Sharp HTML

D

DAO Data Access Object

DI Dependency Inyection (Inyección de Dependecia)

DOM Document Object Model (Modelo de Objetos del Documento)

E

EIS Escuela de Ingeniería en Sistemas

ERPE Escuelas Radiofónicas Populares del Ecuador

ESPOCH Escuela Superior Politécnica de Chimborazo

F

FIE Facultad de Informática y Electrónica

G

GoF Gang of Four

GUI Graphic User Interface (Interfaz Gráfica de Usuario)

H

HTML HiperText Markup Languaje (Lenguaje de Marcado de Hipertexto)

HTTP HiperText Transfer Protocol (Protocolo de Transferencia de Hipertexto)

I

ID Inyección de Dependencia

IDE Integrated Development Enviroment (Entorno de Desarrollo Integrado)

IIS Internet Information Services (Servicios de Información de Internet)

IoC Inversion of Cpntrol (Inversión de Control)

IP Internet Protocol (Protocolo de Internet)

ISO International Estándar Organization (Organización Internacional de Normalización)

L

LINQ Languaje Integrated Query (Lenguaje Integrado de Consulta)

M

MSF Microsoft Solution Framework

MVC Model View Controller (Modelo Vista Controlador)

P

POCO Plain Old Java Object

POJO Plain Old C# Object

R

RIA Rich Internet Aplications (Aplicaciones Ricas de Internet)

S

SQL Structure Query Languaje (Lenguaje de Consultas Estructurado)

U

URL Uniform Resource Locutor (Localizador Uniforme de Recurso)

UML Unified Modeling Languaje (Lenguaje Unificado de Modelado)

W

WWW World Wide Web

X

XML Extended Markup Language (Lenguaje de Marcas Extensible)

ÍNDICE TABLAS

ÍNDICE FIGURAS

INTRODUCCIÓN

Hoy en día la implementación de patrones de diseño en el desarrollo de aplicaciones empresariales, es baja o nula, desaprovechando de esta forma las potencialidades que brindan los mismos, Dichos patrones ayudan a solucionar problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

El uso de la técnica de inversión de control y el patrón de inyección de dependencias permiten acoplar características fundamentales en las aplicaciones tales como: modularidad, reutilización de componentes, además de un "Contenedor" que se encarga de gestionar las instancias de los objetos del usuario.

La presente investigación muestra el estudio comparativo entre los Frameworks de Inyección de Dependencias Microsoft Unity 2.0 Y Spring.Net, para el desarrollo de aplicaciones desacopladas; contemplando el posterior desarrollo de una aplicación para las Escuelas Radiofónicas Populares del Ecuador (ERPE) con el Framework que brinde las mayores y mejores prestaciones.

Este proyecto de investigación esta estructurado de en cuatro capítulos.

En el **Capítulo I,** se tratará sobre el marco referencial, en el cual se encuentra descrita de manera general la justificación del proyecto de tesis y los objetivos a alcanzar con el desarrollo de la misma.

En el **Capítulo II**, se detallarán las definiciones conceptuales de la técnica de Inversión de Control, del patrón de Inyección de Dependencias, las herramientas de desarrollo, el motor de la base de datos, los medidores de rendimiento (Perfomance Counters),

Arquitectura DDD y además de una breve reseña de las Escuelas Radiofónicas del Ecuador (ERPE)

En el **Capítulo III**, se enfoca en la realización del análisis comparativo entre los Frameworks de Inyección de Dependencias, teniendo como objetivo demostrar de una forma práctica (a través de prototipos de prueba) y de forma teórica (a través de investigaciones y puntajes) los beneficios, debilidades, fortalezas, etc. De los frameworks de inyección de dependencias Unity y Spring.Net en el desarrollo y desenvolvimiento de cada parámetro de comparación y de cada una de sus variables expuestas en este capítulo.

En el **Capítulo IV**, con el framework que seleccionamos en el Capitulo III, es decir, el que brinde mayores beneficios y fortalezas, procedemos al desarrollo del sistema para las Escuelas Radiofónicas Populares del Ecuador (ERPE), en este capítulo consta la documentación de este sistema aplicando la metodología Microsoft Solution Framework (MSF), logrando de esta manera vincular la teoría con la práctica.

Como parte final del trabajo investigativo se expone las conclusiones y recomendaciones que se tienen después de la realización de este trabajo.

Además de los capítulos detallados anteriormente, se tiene en la parte de anexos la información adicional utilizada para la realización del estudio comparativo, los manuales de usuario y de administración del sistema.

El presente trabajo, servirá de soporte para la toma de decisiones, al elegir ente los dos frameworks de Inyección de Dependencias mencionados con anterioridad, y decidir cual es el más adecuado para el desarrollo de aplicaciones desacopladas.

CAPÍTULO I:

En este capítulo se encuentra de manera general los antecedentes, la justificación del proyecto de tesis, los objetivos a alcanzar con el desarrollo de la misma y el planteamiento de la hipótesis que se desea demostrar.

1. MARCO REFERENCIAL

En el marco referencial se pretende determinar un enfoque u orientación general de la investigación, además precisar la delimitación del problema, circunscribiéndolo a aspectos definidos y dejando claramente establecidos aquellos que escapan a su alcance, para la elaboración del sistema teórico.

1.1. Antecedentes

1.1.1. Planteamiento del Problema

Las Escuelas Radiofónicas Populares del Ecuador ERPE en el año 2009 han visto la necesidad de incorporar un software que permita administrar el control productivo de los socios asociados, ante tal necesidad se ha desarrollado un software de Gestión creado bajo tecnología Microsoft.

En el año 2010 ERPE incrementa sus oportunidades de negocio ingresando a los mercados de Japón y China con los productos orgánicos cosechados por sus socios. Ante dicha oportunidad surgió la necesidad de introducir un sistema de control de calidad del producto que certifique el proceso productivo por tal motivo la necesidad de contar con un sistema informático que facilite el control de producción orgánica se estableció como prioridad.

Debido a los cambios operativos y tecnológicos observados la aplicación demandó de un mecanismo que permita actualizar sin tener que forzar una reconstrucción, ante lo cual apareció la necesidad de implantar un patrón que solucione este problema.

El patrón escogido fue el de Inversión del Control teniendo dos alternativas: Microsoft Unity 2.0 y Spring.NET 1.3.1 basados en decisión de los técnicos de ERPE para el manejo de IoC, por lo cual es necesario escoger el framework que tenga el mejor rendimiento para el sistema, para su implementación.

La gran cantidad de software desarrollado en el pasado, carecía de modelos ricos de arquitectura, por tal motivo las aplicaciones desarrolladas requerían de grandes esfuerzos de mantenimiento, lo cual en algunos casos conducía a la reingeniería completa de la aplicación.

En la actualidad un considerable número de sistemas informáticos siguen siendo desarrollados sin tomar en cuenta factores importantes de negocio como escalabilidad, estabilidad, compatibilidad y seguridad, por tal motivo las tareas de actualización, migración, y reconstrucción de aplicaciones, son una necesidad fundamental para la empresa.

La dependencia entre componentes en aplicaciones de gran envergadura se ha convertido en otro problema en sistemas que dependen de APIs del sistema operativo,

por tal motivo un modelo de arquitectura consistente que soporte un conjunto de operaciones desacopladas se presenta como necesario en sistemas informáticos productivos que se acoplen a las necesidades temporales de la empresa.

Con la aparición del modelo de inversión de control se plasmó una idea conceptual para la invocación indirecta del código de usuario, y más tarde con el patrón de Inyección de dependencia se puedo separar cada invocación del código ejecutado en la llamada, tras lo cual se desarrollaron numerosos frameworks open source y Propietarios que enfrentaban el problema.

Microsoft Unity 2.0 y Spring.Net son los frameworks open source más populares que enfrentan el problema de la inyección de dependencias en lo concerniente a tecnología .NET, por tal motivo es necesario un estudio comparativo que enfrente sus principales características referentes al patrón de Inyección de dependencia para la mantención del desacople entre ensamblados.

Un análisis comparativo entre los frameworks más populares de ID (Inyección de dependencia), proveerá un conjunto de alternativas para la elección de patrones para la creación de sistemas informáticos escalables y seguros.

En la Escuela de Ingeniería en Sistemas se dicta la materia de Arquitectura de Software, uno de los temas tratados es estilos y patrones, para lo cual se requiere de un análisis y un conjunto de elementos de estudio para la aplicación de patrones en arquitecturas empresariales. Como resultado de la tesis “ANÁLISIS COMPARATIVO ENTRE LOS FRAMEWORKS DE INYECCIÓN DE DEPENDENCIAS UNITY 2.0 Y SPRING.NET” se desarrollará una aplicación que demuestre los escenarios de uso evaluativos para cada framework.

El estudio contemplará todo lo concerniente a la aplicación y comparación del patrón de ID a través del enfoque de los frameworks de IoC (Inversión de Control) y DI (Inyección de dependencia) Microsoft Unity 2.0 y Spring.NET.

1.1.1.1.LUGAR DE APLICACIÓN

Provincia: Chimborazo

Ciudad: Riobamba

Lugar: Escuela Radiofónicas del Ecuador (ERPE)

1.2. Problematización

1.2.1. Formulación del Problema

¿Cuál es el framework de inyección de dependencias (Unity 2.0 o Spring.Net) que nos brinda mayores prestaciones y beneficios al momento de desarrollar aplicaciones desacopladas?

1.2.2. Sistematización

- ¿Cuáles son los beneficios de utilizar un framework de inyección de dependencias?
- ¿Cuáles son los resultados visibles que presenta una aplicación desarrollada con el framework de inyección de dependencias?
- ¿Cuáles son los factores que influyen directamente en el desarrollo de aplicaciones utilizando framework de inyección de dependencias?

1.3. Justificación

1.3.1. Justificación Teórica

El desarrollo de aplicaciones empresariales mediante el uso de la técnica de inversión de control y el patrón de inyección de dependencias permiten acoplar características fundamentales en las aplicaciones tales como: modularidad, reutilización de componentes, además de un "Contenedor" que se encarga de gestionar las instancias (así como sus creaciones y destrucciones) de los objetos del usuario.

El framework de Unity es un contenedor de inyección de dependencias ligero y extensible soporta inyección en el constructor, inyección en propiedades, inyección en llamadas a métodos y contenedores anidados. Unity permite mapear Interfaces hacia clases las que podemos instanciar bajo demanda.

El framework de Spring.Net ofrece un amplio soporte de infraestructura para el desarrollo de la empresa. NET. Le permite eliminar la complejidad accidental cuando se utilizan las bibliotecas de clases base hace que las mejores prácticas, tales como el desarrollo basado en pruebas, prácticas fáciles. Spring.NET se crea, apoyada y sostenida por SpringSource.

1.3.2. Justificación Aplicativa

Actualmente uno de los temas dictados en la materia de Arquitectura de Software de la Escuela de Ingeniería en Sistemas es “Estilos y Patrones”, mediante la investigación propuesta se desarrollará una aplicación que permita evaluar los 2 frameworks en escenarios específicos de prueba, para posteriormente ser implantado en el software de control de producción orgánica de ERPE.

1.4. Objetivos

1.4.1. Objetivo General

Realizar un estudio comparativo entre los frameworks de inyección de dependencias (Unity 2.0 y Spring.Net) para determinar el de mejor rendimiento para el sistema de producción ERPE.

1.4.2. Objetivos Específicos

- Estudiar la estructura de la arquitectura de Inversión de control orientado al desarrollo de aplicaciones desacopladas.
- Determinar los escenarios en los cuales se realizarán las pruebas, que ayudarán al estudio comparativo.
- Realizar un análisis comparativo de las principales características del patrón DI enfocado a cada framework.
- Desarrollar una aplicación que será evaluada en los escenarios que se definan.
- Crear una guía de selección de plataformas para aplicaciones.
- Determinar los parámetros para poder evaluar el objeto de estudio.
- Desarrollar las capas necesarias para la incorporación del patrón de IoC, en el sistema de control de producción de ERPE.

1.5. Hipótesis

La comparación estadística entre el rendimiento del framework de inversión de control originario de tecnología .net Microsoft Unity 2.0 y el framework adaptado de Java para

.NET Spring.net determinará la plataforma de mejor rendimiento para el desarrollo de aplicaciones desacopladas y escalables.

1.5.1. Determinación de Variables

Los indicadores para el presente trabajo de investigación son los frameworks de inyección de dependencia Unity y Spring.Net, en el desarrollo de aplicaciones desacopladas.

1.5.2. Operacionalización Conceptual

En la Tabla I. I se detalla el tipo y descripción de la variable utilizada en la hipótesis.

Tabla I. I Operacionalización de variables

VARIABLE	TIPO	DESCRIPCIÓN
Frameworks de inyección de dependencias Unity y Spring.Net	Independiente	Alternativas de desarrollo para aplicaciones desacopladas

Fuente: Elaborada por los autores

1.5.3. Operacionalización Metodológica

En la Tabla I. II se detalla los indicadores, técnicas e instrumentos que se utilizarán en el estudio de las variables.

Tabla I. II Operacionalización metodológica

VARIABLE	INDICADOR	TÉCNICA	INSTRUMENTOS
Frameworks de inyección de dependencias Unity y Spring.Net para el desarrollo de aplicaciones desacopladas	Rendimiento Complejidad Requerimientos de Hardware y Software Seguridad	Búsqueda información Pruebas	Visual Studio en las versiones 2010 y 2012 SQL Server 2008 R2

Fuente: Elaborada por los autores

1.6.Métodos y Técnicas

1.6.1. Métodos

El método utilizado como guía para la presente investigación es el método científico, el cual contempla los siguientes puntos:

- El planteamiento del problema que en este caso es el análisis comparativo entre los frameworks de inyección de dependencias Unity 2.0 y Spring.Net.
- El apoyo del proceso previo a la formulación de la hipótesis.
- Definición de las variables que permitirán la comparación entre los frameworks.
- Análisis e interpretación de resultados.
- Proceso de comprobación de la hipótesis.

  Para el desarrollo de la aplicación se ocupará la metodología MSF (Microsoft Solutions Framework)

1.6.2. Técnicas

Para la recopilación de la información necesaria que sustente el presente trabajo de investigación, se ha establecido como técnicas las siguientes:

- Revisión de libros Unity 2.0 y Spring.Net
- Revisión de blogs y de sitios oficiales sobre los frameworks de inyección de dependencias.
- Revisión de artículos científicos acerca de Unity 2.0 y Spring.Net.
- Observación

- Técnicas de comprobación de hipótesis.

CAPÍTULO II

En el presente capítulo se detallarán las definiciones conceptuales de la técnica de Inversión de Control, del patrón de Inyección de Dependencias, las herramientas de desarrollo, el motor de la base de datos, los medidores de rendimiento (Perfomance Counters), Arquitectura DDD (Domain Driven Design), además de una breve reseña de las Escuelas Radiofónicas del Ecuador (ERPE).

2. MARCO TEÓRICO

A continuación se detalla la información documental para confeccionar el diseño metodológico de la investigación

2.1.Introducción a la técnica de inversión de control (IoC)

2.1.1. Conceptualización

En el desarrollo de software moderno se ha comenzado a generalizar el uso de patrones de diseño para implementar partes de nuestros desarrollos. Los patrones proponen soluciones estándares a los problemas comunes de diseño de nuestras aplicaciones. Hay muchos patrones diferentes, aunque sin duda los más famosos son los conocidos como patrones “GoF”.

Realmente podemos definir nosotros mismos otros patrones, siempre que encontremos una solución estándar a un problema común.

La Inversión de Control es un principio de desarrollo pensado para permitir un menor acoplamiento entre componentes de una aplicación y fomentar así el rehúso de los mismos.

La Inversión de Control es una aplicación práctica del principio de inversión de dependencia que se usa en aquellas situaciones donde se desea que el código genérico controle la ejecución de componentes específicos.

A día de hoy la inversión de control a menudo se asocia a un conjunto de frameworks que ofrecen características avanzadas para inyectar código en tiempo de ejecución.

La mayoría de los frameworks de inversión de control están implementados sobre un objeto contenedor que resuelve las dependencias a través de un fichero de configuración, el procedimiento suele ser en la mayoría de los casos el mismo:

- Se instancia el contenedor
- Se pasa la interfaz como argumento
- El framework IoC devuelve un objeto que implementa la interfaz

2.1.2. Historia y Problematización

En los comienzos de la programación, los programas eran lineales y monolíticos. El flujo de ejecución era simple y predecible, ejecutándose línea tras línea.

Aparecieron dos conceptos que revolucionaron la programación:

- La modularidad

- La reutilización de los componentes: se crean bibliotecas de componentes reutilizables.

El flujo se complica, debido a que un componente tiene dependencias de servicios o componentes cuyos tipos concretos son especificados en tiempo de diseño.

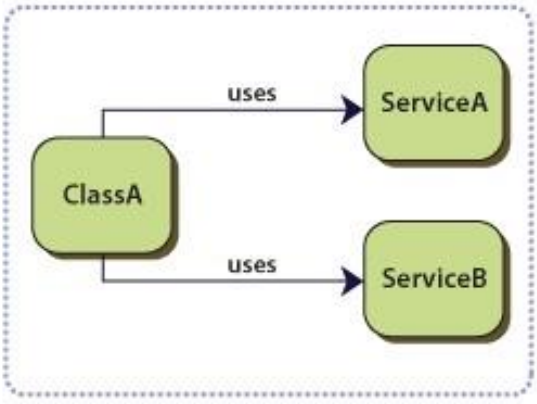


Figura II. 1 Dependencias de servicios

En el ejemplo de la imagen, la Clase A depende del Servicio A y del Servicio B.

Los problemas que esto plantea son:

- Al remplazar o actualizar las dependencias, se necesita cambiar el código fuente de la clase A.
- Las implementaciones concretas de las dependencias tienen que estar disponibles en tiempo de compilación.
- Las clases son difíciles de testear aisladamente porque tienen directas definiciones a sus dependencias. Lo mencionado anteriormente significa que las dependencias no pueden ser reemplazadas por componentes stubs o mocks.
- Las clases tienen código repetido para crear, localizar y gestionar sus dependencias.

Aquí la solución pasa por delegar la función de seleccionar una implementación concreta de las dependencias a un componente externo.

El control de cómo un objeto A obtiene la referencia de un objeto B es invertido. El objeto A no es responsable de obtener una referencia al objeto B sino que es el Componente Externo el responsable de esto. Esta es la base del patrón de Inversión de Control (IoC).

La inversión de control IOC aplica un principio de diseño denominado principio de Hollywood (*No nos llames, nosotros te llamaremos*).

2.1.3. Usos

El principio de Inversión de Control IoC se puede utilizar cuando:

- Se desee desacoplar las clases de sus dependencias de manera de que las mismas puedan ser remplazadas o actualizadas con muy pocos o casi ningún cambio en el código fuete de sus clases.
- Desea escribir clases que dependan de clases cuyas implementaciones no son conocidas en tiempo de compilación.
- Desea testar las clases aisladamente sin sus dependencias.
- Desea desacoplar sus clases de ser responsables de localizar y gestionar el tiempo de vida de sus dependencias.

2.1.4. Técnicas de Implementación

Según diversos enfoques, la inversión de control puede implementarse de diversas maneras. Entre las más conocidas tenemos:

- Inyección de dependencias
- Service Locator

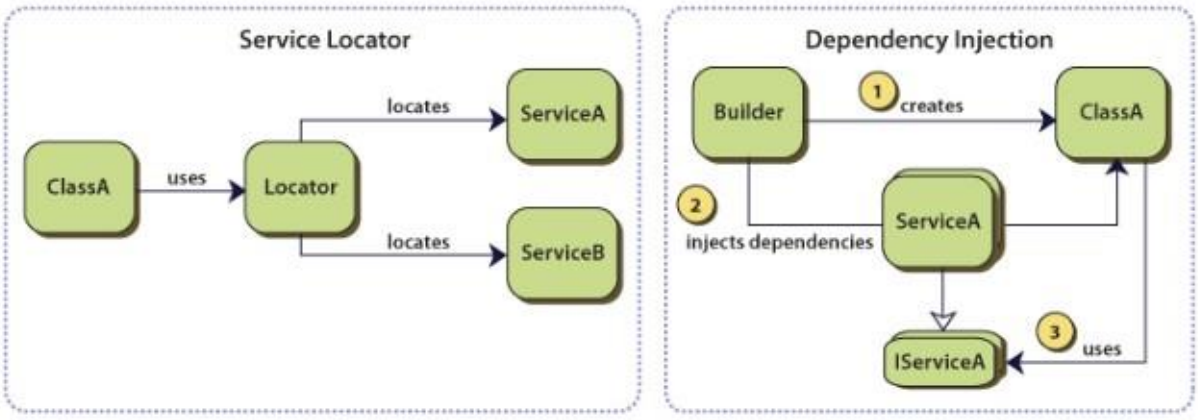


Figura II. 2 Técnicas de Implementación de IoC

2.1.4.1. Service Locator

Un Service Locator es un componente que contiene referencias a los servicios y encapsula la lógica que los localiza dichos servicios. Así, en nuestras clases, utilizamos el Service Locator para obtener instancias de los servicios que realmente necesitamos.

El Service Locator no instancia los servicios. Provee una manera de registrar servicios y mantener una referencia a dichos servicios. Luego de que el servicio es registrado, el Service Locator puede localizarlo.

El Service Locator debe proveer una forma de localizar un servicio sin especificar el tipo. Por ejemplo, puede usar una clave string o el tipo de interface. Esto permite un fácil reemplazo de la dependencia sin modificar el código fuente de la clase.

Es asunto de la implementación de esta forma de IoC, el determinar la forma en que se localizará el servicio requerido.

2.1.4.2. Inyección de Dependencias (DI)

La inyección de dependencia es un conjunto de principios de diseño de software y patrones que nos permitan desarrollar un código de acoplamiento flexible.

La Inyección de Dependencia (en inglés Dependency Injection, DI) es un patrón de diseño orientado a objetos, en el que se inyectan objetos a una clase en lugar de ser la propia clase quien cree el objeto.

La forma habitual de implementar este patrón es mediante un "Contenedor DI" y objetos POJO o POCO. El contenedor inyecta a cada objeto los objetos necesarios según las relaciones plasmadas en un fichero de configuración. Típicamente este contenedor es implementado por un framework externo, como puede ser por ejemplo Spring Framework.

2.2.El patrón de Inyección de Dependencias (DI)

2.2.1. Conceptualización

Una dependencia entre un componente y otro, puede establecerse estáticamente o en tiempo de compilación, o bien, dinámicamente o en tiempo de ejecución. Es en éste último escenario es donde cabe el concepto de inyección, y para que esto fuera posible, debemos referenciar interfaces y no implementaciones directas.

En general, las dependencias son expresadas en términos de interfaces en lugar de clases concretas. Esto permite un rápido remplazo de las implementaciones dependientes sin modificar el código fuente de la clase.

Lo que propone entonces la Inyección de dependencias, es no instanciar las dependencias explícitamente en su clase, sino que declarativamente expresarlas en la definición de la clase. La esencia de la inyección de las dependencias es contar con un

componente capaz de obtener instancias validas de las dependencias del objeto y pasárselas durante la creación o inicialización del objeto.

Ejemplo:

```
public class A{
  private B b;

  public A(){
  }

  public setB(B b){
   this.b=b;
 }

}
```

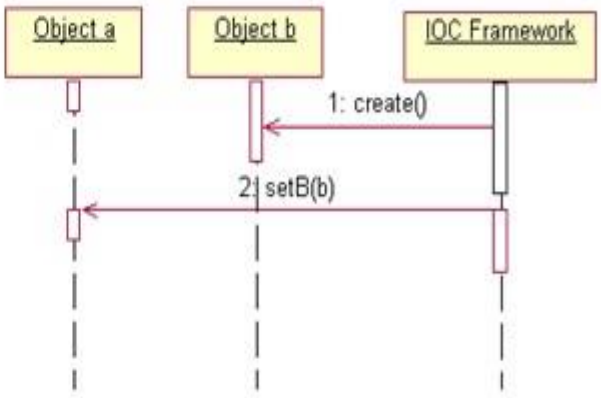


Figura II. 3 Inyección de Dependencias

- A necesita una referencia a B, pero no necesita saber cómo debe crear la instancia de B, solo quiere una referencia a ella.
- B puede ser una interface, clase abstracta o clase concreta.
- Antes de que una instancia de A sea usada, necesitara una referencia a una instancia de B.

Aquí no hay un alto acoplamiento entre A y B, ambas pueden cambiar internamente sin afectar a la otra. Los detalles de cómo se crea y gestiona un objeto B, no es decidido en la implementación de A. Es un framework IoC quien usa un método como setB() de A para inyectar luego a un objeto B.

2.2.2. Maneras de Implementación de la Inyección de Dependencias (DI)

Existen tres principales maneras de implementar la inyección de dependencias:

2.2.2.1. Inyección basada en el Constructor

Las dependencias se inyectan utilizando un constructor con parámetros del objeto dependiente. Éste constructor recibe las dependencias como parámetros y las establece en los atributos del objeto.

Considerando un diseño de dos capas donde tenemos una capa de BusinessFacade y otra de BusinessLogic, la capa BusinessFacade depende de la BusinessLogic para operar correctamente. Todas las clases de lógica de negocio implementan la interface IBusinessLogic.

En la inyección basada en un constructor, se creará una instancia de BusinessFacade usando un constructor parametrizado al cual se le pasará una referencia de un IBusinessLogic para poder inyectar la dependencia.

```
interface IBusinessLogic
{
  //Some code
}
class ProductBL implements IBusinessLogic
{
  //Some code
}
class CustomerBL implements IBusinessLogic
{
  //Some code
}
public class BusinessFacade
{
  private IBusinessLogic businessLogic;
  public BusinessFacade(IBusinessLogic businessLogic)
  {
  this.businessLogic = businessLogic;
  }
}
```

El objeto responsable de las dependencias realizará la inyección de la siguiente forma:

```
IBusinessLogic productBL = new ProductBL();
BusinessFacade businessFacade = new BusinessFacade(productBL);
```

La principal desventaja de la IoC basada en constructor es que una vez que la clase BusinessFacade es instanciada no podemos cambiar las dependencias inyectadas.

2.2.2.2. Inyección basada en Métodos Setters

En este tipo de IoC, se utiliza un método setters para inyectar una dependencia en el objeto que la requiere. Se invoca así al setter de cada dependencia y se le pasa como parámetro una referencia a la misma.

```
public class BusinessFacade
{
  private IBusinessLogic businessLogic;
  public setBusinessLogic(IBusinessLogic
businessLogic)
  {
          this.businessLogic = businessLogic;
  }
}
```

El objeto responsable de las dependencias realizará la inyección de la siguiente forma:

```
IBusinessLogic productBL = new ProductBL();
BusinessFacade businessFacade = new BusinessFacade();
businessFacade.setBusinessLogic(productBL);
```

La ventaja aquí es que uno puede cambiar la dependencia entre BusinessFacade y la implementación de IBusinessLogic luego de haber instanciado la clase BusinessFacade.

La desventaja es que un objeto con setters no puede ser inmutable y suele ser complicado determinar cuáles son las dependencias que se necesitan y en que momento se las necesita. Se recomienda así utilizar IoC basada en constructor a menos que

necesite cambiar la dependencia y sepa claramente los momentos en los cuales realizar estos cambios.

Otra desventaja es que al utilizar setters (necesarios para la inyección), estamos exponiendo las propiedades de un objeto al mundo exterior cuando en realidad no era un requerimiento de negocio hacerlo.

2.2.2.3. Inyección basada en Interfaces

Aquí se utiliza una interfaz común que otras clases implementan para poderles luego inyectar dependencias. En el siguiente ejemplo, a toda clase que implemente IBusinessFacade se le podrá inyectar cualquier objeto que implemente IBusinessLogic mediante el método injectBLObject

```
interface IBusinessLogic
{
  //Some code
}
interface IBusinessFacade
{
  public void injectBLObject (IBusinessLogic businessLogic);
}
class ProductBL implements IBusinessLogic
{
  //Some code
}
class CustomerBL implements IBusinessLogic
{
  //Some code
}
class BusinessFacade implements IBusinessFacade
{
  private IBusinessLogic businessLogic;
  public void injectBLObject (IBusinessLogic businessLogic)
  {
          this.businessLogic = businessLogic;
  }
}
```

El objeto responsable de las dependencias realizará la inyección de la siguiente forma:

```
IBusinessLogic businessLogic = new ProductBL();
BusinessFacade businessFacade = new
BusinessFacade();
businessFacade.injectBLObject(businessLogic);
```

Las tres formas de inyección presentadas, pasan una referencia a una implementación de IBusinessLogic más que un tipo particular de BusinessLogic. Esto presenta muchas ventajas, como lo declara Jeremy Weiskotten, un reconocido ingeniero de software de la empresa Kronos.

"Codificar contra interfaces bien definidas es la clave para alcanzar un menor acoplamiento. Acoplando un objeto a una interface en lugar de a una implementación específica, permite utilizar cualquier implementación con el mínimo cambio y riesgo"

2.2.3. Aplicabilidad

La inyección de dependencias no debería usarse siempre que una clase dependa de otra, sino más bien es efectiva en situaciones específicas como las siguientes:

- Inyectar datos de configuración en un componente.
- Inyectar la misma dependencia en varios componentes.
- Inyectar diferentes implementaciones de la misma dependencia.
- Inyectar la misma implementación en varias configuraciones
- Se necesitan alguno de los servicios provistos por un contenedor.

La IoC no es necesaria si uno va a utilizar siempre la misma implementación de una dependencia o la misma configuración, o al menos, no reportará grandes beneficios en estos casos.

2.2.4. Contenedores

En pos de implementar el patrón IoC en alguna de sus variantes, existen los denominados contenedores de inyección de dependencias (CID), que son los componentes encargados de instanciar las dependencias y realizar las inyecciones necesarias entre todos los objetos y además gestionar sus respectivos ciclos de vida.

En cuanto a la instanciación e inyección, un CID se configura para especificarle que dependencias debe instanciar y donde deberá luego inyectarlas. Para la instanciación además, se debe definir el modo de instanciación, es decir, si se creará una nueva siempre que se lo requiera, o se reusará la instancia creada (singleton).

En cuanto a la gestión de los ciclos de vida de los objetos creados, implica que son capaces de administrar las diferentes etapas de la vida del componente (instanciación, configuración, eliminación).

El hecho de que el contenedor a veces mantenga una referencia a los componentes creados luego de la instanciación es lo que lo hace un contenedor.

No todos los objetos deben ser gestionados. El contenedor mantiene una referencia a aquellos objetos que son reusados para futuras inyecciones, como singletones. Cuando configuramos el contenedor para que siempre cree nuevas instancias, entonces éste se suele olvidar de dichos objetos creados, dejando la tarea al GC para recolectarlos luego de que no sean más usados.

Existen varios CID, según el lenguaje de programación que soportan, que modos de IoC soportan, etc. Para Java tenemos el Butterfly Container, Spring, Pico Container, Guice, y otros.

2.3.Microsoft .Net Visual Studio

Es un entorno de desarrollo integrado (IDE), que soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET.

Se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles. Visual Studio 2010 es la versión más reciente de esta herramienta, acompañada por .NET Framework 4.0.

Entre sus más destacables características, se encuentran la capacidad para utilizar múltiples monitores, así como la posibilidad de desacoplar las ventanas de su sitio original y acoplarlas en otros sitios de la interfaz de trabajo. Además de esto, aparece una edición que compila las características de todas las ediciones comunes de Visual Studio: Professional, Team Studio, Test, conocida como Visual Studio Ultimate.

La versión de este IDE que ocuparemos en el desarrollo de este trabajo de investigación es Visual Studio 2010 Ultimate Edition.

2.4.SQL Server

Es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL. Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos.

Algunas características que podemos destacar de este SGBD

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Este sistema incluye una versión reducida, llamada MSDE con el mismo motor de base de datos pero orientado a proyectos más pequeños, que en sus versiones 2005 y 2008 pasa a ser el SQL Express Edition, que se distribuye en forma gratuita.

Es común desarrollar completos proyectos complementando Microsoft SQL Server y Microsoft Access a través de los llamados ADP (Access Data Project). De esta forma se completa la base de datos (Microsoft SQL Server), con el entorno de desarrollo (VBA Access), a través de la implementación de aplicaciones de dos capas mediante el uso de formularios Windows.

En el manejo de SQL mediante líneas de comando se utiliza el SQLCMD

Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para varias plataformas de desarrollo, entre ellas .NET, pero el servidor sólo está disponible para Sistemas Operativos.

Para este proyecto de investigación utilizaremos SQL Server en su versión 2008 R2.

2.5.Medidores de Rendimiento (Perfomance Counters)

Los medidores de rendimiento se utilizan para proporcionar información referente a qué tan bien el sistema operativo, una aplicación, servicio o un controlador se está ejecutando. Los datos emitidos pueden ayudar a determinar los cuellos de botella del sistema y con el objetivo de ajustarlo y mejorar el rendimiento de las aplicaciones en el computador.

Los medidores de rendimiento proporcionan datos sobre el sistema operativo, red y diferentes dispositivos, estos datos se los puede reflejan en gráficas que permiten al usuario saber que tan bien está funcionando el sistema.

2.6.Arquitectura DDD (Domain Driven Design)

El objetivo de esta arquitectura es proporcionar una base consolidada y guías de arquitectura para un tipo concreto de aplicaciones: “Aplicaciones empresariales complejas‟. Este tipo de aplicaciones se caracterizan por tener una vida relativamente larga y un volumen de cambios evolutivos considerable. Por lo tanto, en estas aplicaciones es muy importante todo lo relativo al mantenimiento de la aplicación, la facilidad de actualización, o la sustitución de tecnologías y frameworks/ORMs (Objectrelational mapping) por otras versiones más modernas o incluso por otros diferentes, etc.

El objetivo es que todo esto se pueda realizar con el menor impacto posible sobre el resto de la aplicación. En definitiva, que los cambios de tecnologías de infraestructura de una aplicación no afecten a capas de alto nivel de la aplicación, especialmente, que afecten lo mínimo posible a la capa del “Dominio de la aplicación‟.

2.6.1. Marco N-Capas con Orientación al Dominio

El objetivo de esta arquitectura marco es estructurar de una forma limpia y clara la complejidad de una aplicación empresarial basada en las diferentes capas de la arquitectura, siguiendo el patrón N- Layered y las tendencias de arquitecturas en DDD.

El patrón N-Layered distingue diferentes capas y sub-capas internas en una aplicación, delimitando la situación de los diferentes componentes por su tipología.

Por supuesto, esta arquitectura concreta N-Layer es personalizable según las necesidades de cada proyecto y/o preferencias de Arquitectura. Simplemente proponemos una Arquitectura marco a seguir que sirva como punto base a ser modificada o adaptada por arquitectos según sus necesidades y requisitos.

En concreto, las capas y sub-capas propuestas para aplicaciones "N-Layered con Orientación al Dominio" son:

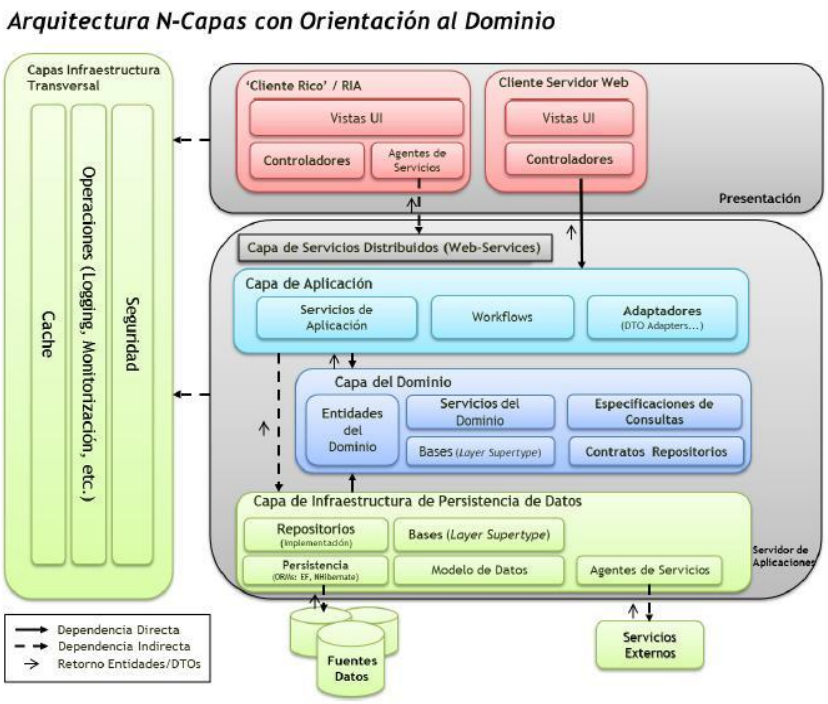


Figura II. 4 Arquitectura N-Capas con Orientación al Dominio

Capa de Presentación

- Subcapas de Componentes Visuales (Vistas)
- Subcapas de Proceso de Interfaz de Usuario (Controladores y similares)

Capa de Servicios Distribuidos (Servicios-Web)

- Servicios-Web publicando las Capas de Aplicación y Dominio

Capa de Aplicación

- Servicios de Aplicación (Tareas y coordinadores de casos de uso)
- Adaptadores (Conversores de formatos, etc.)
- Subcapa de *Workflows* (Opcional)
- Clases base de Capa Aplicación (Patrón Layer-Supertype)

Capa del Modelo de Dominio

- Entidades del Dominio
- Servicios del Dominio
- Especificaciones de Consultas (Opcional)
- Contratos/Interfaces de *Repositorios*
- Clases base del Dominio (Patrón *Layer-Supertype*)

Capa de Infraestructura de Acceso a Datos

- Implementación de Repositorios‟

- Modelo lógico de Datos
- Clases Base (Patrón *Layer-Supertype*)
- Infraestructura tecnología ORM
- Agentes de Servicios externos

Componentes/Aspectos Horizontales de la Arquitectura

- Aspectos horizontales de Seguridad, Gestión de operaciones,
- Monitorización, Correo Electrónico automatizado, etc

2.6.2. Interacción en la Arquitectura DDD

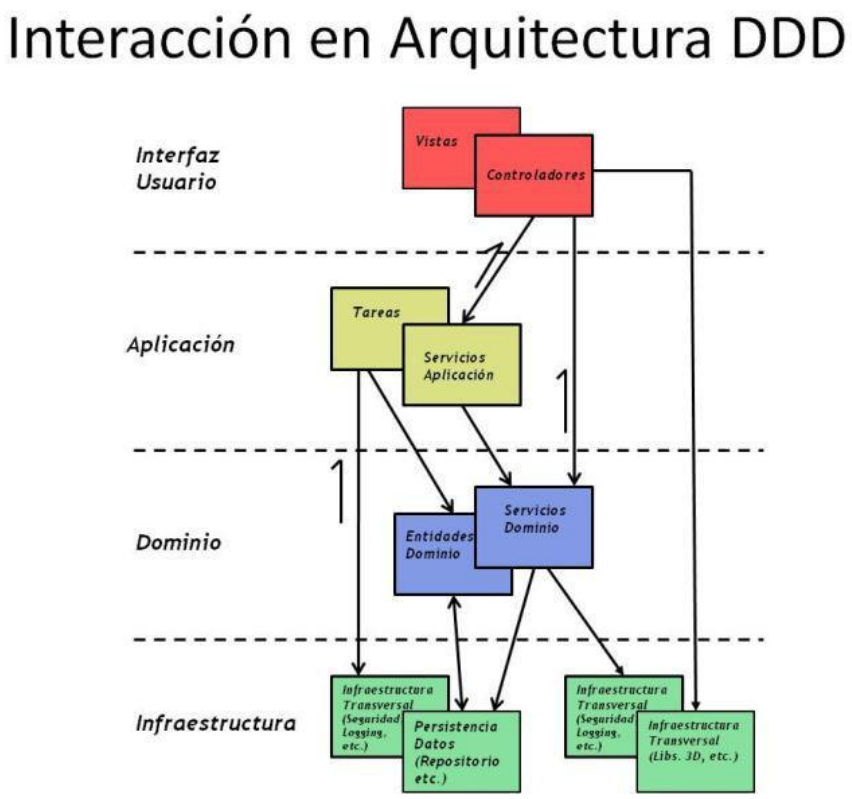


Figura II. 5 Interacción Arquitectura DDD

Primeramente, podemos observar que la Capa de Infraestructura que presenta una

arquitectura con tendencia DDD, es algo muy amplio y para muchos contextos muy diferentes (Contextos de Servidor y de Cliente). La Capa de infraestructura contendrá todo lo ligado a tecnología/infraestructura. Ahí se incluyen conceptos fundamentales como Persistencia de Datos (Repositorios, etc.), pasando por aspectos transversales como Seguridad, *Logging*, Operaciones, etc. e incluso podría llegar a incluirse librerías específicas de capacidades gráficas para UX (librerías 3D, librerías de controles específicos para una tecnología concreta de presentación, etc.). Debido a estas grandes diferencias de contexto y a la importancia del acceso a datos, en nuestra arquitectura propuesta hemos separado explícitamente la Capa de Infraestructura de "Persistencia de Datos" del resto de capas de "Infraestructura Transversal", que pueden ser utilizadas de forma horizontal/transversal por cualquier capa.

El otro aspecto interesante que adelantábamos anteriormente, es el hecho de que el acceso a algunas capas no es con un único camino ordenado por diferentes capas.

Concretamente podremos acceder directamente a las capas de Aplicación, de Dominio y de Infraestructura Transversal siempre que lo necesitemos. Por ejemplo, podríamos acceder directamente desde una Capa de Presentación Web (no necesita interfaces remotos de tipo Servicio-Web) a las capas inferiores que necesitemos (Aplicación, Dominio, y algunos aspectos de Infraestructura Transversal). Sin embargo, para llegar a la "*Capa de Persistencia de Datos*" y sus objetos *Repositorios* (puede recordar en algunos aspectos a la *Capa de Acceso a Datos (DAL)* tradicional, pero no es lo mismo), es recomendable que siempre se acceda a través de los objetos de coordinación (Servicios) de la Capa de Aplicación, puesto que es la parte que los orquesta.

Queremos resaltar que la implementación y uso de todas estas capas debe ser algo flexible. Relativo al diagrama, probablemente deberían existir más combinaciones de

flechas (accesos). Y sobre todo, no tiene por qué ser utilizado de forma exactamente igual en todas las aplicaciones.

2.7.Escuela Radiofónicas del Ecuador (ERPE)

2.7.1. Introducción

Escuelas Populares Radiofónicas del Ecuador nació el 19 de marzo de 1962. Fue fundada por Monseñor Leonidas Proaño V., en ese entonces Obispo de la Diócesis de Riobamba. Fue la primera radio popular educativa del país. El objetivo principal es que la población indígena y mestiza de la zona rural y urbana sean los actores de su propio desarrollo.

A lo largo de su historia ha sido reconocida por la labor en alfabetización, tele-educación, comunicación popular, agricultura orgánica y nuevas tecnologías de la información y comunicación en las zonas rurales. También es reconocida por su seriedad y veracidad en el manejo de la información noticiosa en kichwa y castellano.

Actualmente mantiene dos emisoras populares participativas, una en Amplitud Modulada y otra en Frecuencia Modulada; departamento de Agricultura Orgánica y de Telecentros Comunitarios.

Pertenecemos a la Coordinadora de Coordinadora de Radio Popular Educativa del Ecuador (CORAPE) y la Asociación Latinoamericana de Educación Radiofónica (ALER). Pertenecemos a la Asociación Mundial de Radios Comunitarias (AMARC).

Por 50 años hemos trabajado apasionadamente en la misión y principios originales. A partir de este cincuentenario el compromiso es doble para que los pueblos particularmente indígenas tengan vida digna haciéndose escuchar su voz y pensamiento, participado de las cadenas productivas y generando ingresos económicos y junto con la naturaleza y producción sana.

Las Escuelas Populares Radiofónicas del Ecuador es una fundación no gubernamental, cuyo objetivo es que la población indígena y mestiza de la zona rural y urbana sean los actores de su propio desarrollo.

ERPE genera facilita y acompaña procesos sustentables con estos grupos sociales y fortalece las capacidades locales. Trabajamos conjuntamente con la COPROBICH. la población indígena y mestiza interactúa en la radio participación de ERPE. Otros se informan en los centros de información llamados Telecentros. Pero la comunicación y educación no llega al estómago, ERPE fortalece desde los años 80 las y los campesinos en producción orgánica y economía propia.

Para la comercialización y explotación se responsabiliza la empresa de ERPE y COPROBICH: SUMAKLIFE, fundada en el 2006. En el área de salud ERPE lleva a cabo actividades preventivas.

ERPE creó y es parte de una red de muchos actores que luchan juntos contra la pobreza y para un desarrollo social propio de la región. Está asociada a redes radiofónicas nacionales, latinoamericanas y mundiales.

2.7.2. Misión

Desarrollar e implementar programas y procesos participativos e innovadores de formación, información, mejoramiento de la alimentación y la salud, economía propia, fortalecimiento organizativo, protección del ambiente y comunicación radiofónica que faciliten a los grupos sociales mejorar sus capacidades y competencias para que sean actores de su propio desarrollo.

2.7.3. Visión

Fundación sostenible, propositiva y dinámica que apoya y desarrolla acciones, competencias y capacidades para el mejoramiento de la calidad de vida de los sectores sociales excluidos.

2.7.4. Valores

- Interculturalidad
- Solidaridad
- Equidad
- Eficiencia
- Pro-actividad
- Democracia

CAPÍTULO III

Este capítulo se enfoca en el análisis comparativo entre los Frameworks de Inyección de Dependencia, y tiene como objetivo demostrar en forma práctica (a través de prototipos de prueba) y de forma teórica (a través de investigaciones y puntajes) los beneficios, fortalezas, debilidades, facilidades, etc. de cada uno de los frameworks, tomando como base los parámetros planteados en los capítulos anteriores. Al finalizar la comparación y evaluación de los resultados, se selecciona el framework mas adecuado para el desarrollo del sistema.

3. ANALISIS COMPARATIVO DE LOS FRAMEWORKS DE INYECCIÓN DE DEPENDENCIAS UNITY 2.0 Y SPRING .NET

Para poder determinar que aspectos se necesitan desarrollar en una comparación de frameworks de inyección de dependencias se debe describir los ítems que son importantes evaluar para poder realizar pruebas que determinan fortalezas y debilidades de cada tecnología.

Para realizar esto se definen ciertos parámetros de comparación, los cuales contienen variables que representan los aspectos comparativos que servirán de base para pruebas

posteriores en la que serán evaluados los Frameworks de Inyección de Dependencia: Microsoft Unity 2.0 y Spring.Net 1.3.1.

La importancia de este estudio radica en determinar conclusiones sobre los frameworks de inyección de dependencias, basado en el análisis de los grados de efectividad, debilidades y aciertos para cada parámetro.

3.1.Determinación de los parámetros de comparación

Los parámetros y variables que a continuación se definen para la realización del presente estudio comparativo entre los frameworks de inyección de dependencias Microsoft Unity y Spring.Net, fueron seleccionados por los autores de esta tesis, en base a la información obtenida.

Los parámetros considerados para este estudio son:

- **Rendimiento:** El objetivo de este parámetro es determinar con qué efectividad y eficiencia los frameworks de inyección de dependencias administran los recursos del computador.
- **Complejidad:** El objetivo principal del presente parámetro es determinar el nivel de dificultad, en el aprendizaje, manejo e implementación del framework de inyección de dependencias.
- **Requerimientos de hardware y software:** El objetivo del parámetro es determinar los requerimientos necesarios para la ejecución de los frameworks de Inyección de dependencias.
- **Seguridad:** El objetivo principal de este parámetro es determinar los diferentes niveles de seguridad que proporciona el framework de inyección de dependencias.

3.2.Método para la evaluación de resultados

Se definen una serie de pasos, los cuales servirán para evaluar los frameworks de forma comparativa, lo cual permite reflejar en datos estadísticos y de forma clara los resultados obtenidos por cada framework en las respectivas pruebas de cada parámetro.

Mediante la creación de tablas se desea ilustrar y aclara la comparación entre los dos frameworks de inyección de dependencias, con esto se podrá comparar el nivel de cumplimiento, las cuales serán utilizadas en las conclusiones de las pruebas.

La forma de evaluar los dos frameworks se las realizará en base de cuatro parámetros descritos con anterioridad, lo que nos permitirá obtener resultados cuantitativos y cualitativos que permitirán una selección sustentada de uno de los frameworks analizados.

En la siguiente tabla se muestran los niveles de cumplimiento que serán manejados en el desarrollo de este trabajo de investigación, de esta forma se refleja de mejor manera el grado de desempeño del framework, y la supremacía de la una sobre la otra.

Tabla III. I Niveles de cumplimiento

VALORACIÓN	CALIFICACIÓN	FORMA GRÁFICA	VALOR	VALOR PARAMETRO NEGATIVO	DESCRIPCIÓN
Excelente (VA)	>80%y<=100%	★★★★★	5	-5	**Cumple con todas las expectativas.**
Muy bueno (VB)	>60% y <=80%	★★★★☆	4	-4	**Cumple con la mayoría de las expectativas.**
Bueno (VC)	> 40% y <=60%	★★★☆☆	3	-3	**Cumple varias expectativas.**
Regular (VD)	> 20% y <=40%	★★☆☆☆	2	-2	**Cumple pocas expectativas.**
Malo (VE)	>=0% y <=20%	★☆☆☆☆	1	-1	**No cumple las expectativas.**

Fuente: Elaborada por los autores

La calificación definitiva en base a cada parámetro de comparación se obtiene sumando los puntajes obtenidos del análisis, utilizando las siguientes fórmulas:

$$C_u = \sum_{i=1}^{n} V_i$$

$$C_s = \sum_{i=1}^{n} V_i$$

$$C_{max} = \sum_{i=1}^{n} VM_i$$

Donde:

n: Número de variables del parámetro

V_i: Valor de calificación de cada variable

VM_i: Valor máximo de calificación de cada variable en el parámetro

C_u: Calificación obtenida por Unity en el parámetro

C_s: Calificación obtenida por Spring.Net en el parámetro

C_{max}: Calificación máxima sobre el que se califica el parámetro

En la siguiente tabla se muestran las fórmulas para calcular los porcentajes de cada uno de los parámetros de comparación por cada tecnología.

Tabla III. II Fórmulas para el cálculo de porcentaje de cada framework

DESCRIPCIÓN	RESULTADO
$\boldsymbol{P_u}$	$(C_u / C_{max}) * 100\%$
$\boldsymbol{P_s}$	$(C_s / C_{max}) * 100\%$

Fuente: Elaborada por los autores

Donde:

P_u: Porcentaje que obtuvo el framework de inyección de dependencias Unity

P_s: Porcentaje que obtuvo el framework de inyección de dependencias Spring.NET

3.2.1. Descripción del Escenario de Pruebas

Equipo Utilizado

El escenario de pruebas está constituido por una única máquina donde se ha ejecutado todas las pruebas. La máquina es un computador portátil que tiene las siguientes especificaciones hardware detalladas en la Tabla III.III

Tabla III. III Descripción del hardware utilizado

CARACTERÍSTICA	DESCRIPCIÓN
Procesador	Procesador Intel Core i7 2630-QM 2.0GHz
Memoria RAM	8 GB DDR3
Disco Duro	500 GB

Fuente: Elaborada por los autores

Software Utilizado

El software que hemos utilizado para la realización de las pruebas son los que a continuación detallamos en la Tabla III.IV

Tabla III. IV Descripción del software utilizado

CARACTERÍSTICA	DESCRIPCIÓN
Sistema Operativo	Windows 7 Ultimate x64
IDE	Microsoft Visual Studio 2010 Ultimate
.Net Framework	4.0
Hosting Web	Internet Information Services 7.0
Monitor de Recursos	Perfmon – Monitor del sistema

Fuente: Elaborada por los autores

3.3. Desarrollo de las pruebas en los parámetros de comparación

Se procede realizar las pruebas en cada uno de los cuatro parámetros establecidos para cada una de los frameworks de inyección de dependencias, en los que se detalla el

proceso para determinar cada variable del parámetro, se los califica, se evalúan los resultados obtenidos, se grafican y se realiza su interpretación.

3.3.1. Rendimiento

El objetivo de este parámetro es determinar la eficiencia, efectividad con la que el framework maneja los diferentes procesos de rendimientos tales como: gestión de memoria, velocidad en la creación de objetos, velocidad en la obtención de objetos

Tabla III. V Variables del parámetro de comparación rendimiento

RENDIMIENTO	
VARIABLE	**DESCRIPCIÓN**
Tiempo de creación de objetos	Tiempo que toma el contenedor de objetos en iniciar los objetos en base a la configuración.
Tiempo de recuperación de objetos	Tiempo que toma al contenedor de objeto retornar un objeto creado o duplicar uno ya existente para la inyección sobre la aplicación.
Memoria usada por el contenedor	Cantidad de memoria reservada para la aplicación con el contenedor de IoC

Fuente: Elaborada por los autores

A partir de cuatro aplicaciones prototipos, una aplicación con el framework de inyección de dependencias Spring.Net y Microsoft Unity se ejecutaron un conjunto de pruebas las pruebas de rendimiento de manera automatizada y darle una mayor certificación a los tiempos obtenidos.

Además, se utiliza la variable “frecuencia” para determinar el número de veces que se ejecutarían dichas pruebas para medir la capacidad de respuesta mostrada por cada framework de inyección de dependencias.

3.3.1.1. Variable 1: Tiempo Creación Objetos

Esta variable se obtiene cuando el framework de inyección de dependencias inicializa los objetos en base a la configuración del contendor.

Para la ejecución de las pruebas se consideró una frecuencia de 385 a partir de lo cual se calculará el promedio del tiempo de respuesta obtenido por cada proceso, medido en milisegundos. En el **Anexo 1** se encuentra el cálculo para determinar el tamaño de la muestra para esta prueba. En el **Anexo 2** se encuentra la prueba de tiempos de respuesta de cada operación en las interfaces de los prototipos.

Tabla III. VI Resultado de la medición del tiempo para la creación de objetos

	Frecuencia	Sumatoria de valores	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC
Microsoft Unity	385	22096	57.3922	±1.0299	1.7945	0.1175	$57.1619 - 57.6225$
Spring.net	385	176719	459.0104	±1.3765	0.2244	0.2864	$458.4490 - 459.5718$

Fuente: Elaborada por los autores

Interpretación del resultado

Microsoft Unity presenta mejor tiempo de respuesta para la creación de objetos al emplear un tiempo de 57.3922 milisegundos en comparación con Spring.Net que tarda 459.0104 milisegundos, presentando una relación de 1:7.99, sin embargo la creación de objetos es un proceso que aparece únicamente cuando la aplicación web se inicializa, por lo cual se determina que los tiempos de respuesta son aceptables al no superar el 1 segundo.

Calificación

En base a los criterios de calificación:

Tabla III. VII Calificación para la variable tiempo para la creación de objetos

Rango de Valores	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
1 a 100	5	Excelente	
101 a 200	4	Muy Buena	
201 a 300	3	Buena	
301 a 400	2	Regular	
401 a 500	1	Malo	

Fuente: Elaborada por los autores

El framework Microsoft Unity emplea un tiempo de 57.3922 milisegundos, de acuerdo a la tabla de calificación para esta variable se le asigna una calificación de Excelente, su forma gráfica será de cinco estrellas con un valor de 5.

Spring.Net al tomar un tiempo de 459.0104 milisegundos se le asigna para la variable tiempo de creación un valor de 1 equivalente al valor Malo.

3.3.1.2. Variable 2: Tiempo de recuperación de objetos

Esta variable se obtiene cuando la aplicación demanda de un objeto que use cierto tipo de interfaz, en ese momento el contendor extrae y devuelve las referencias sobre dichos objetos hacia la aplicación.

Considerando la frecuencia de 385 luego de realizar la prueba (En el Anexo 3 se detalla el cálculo de la prueba) se presentan los resultados:

Tabla III. VIII Resultado de la medición del tiempo de obtención de objetos

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC
Microsoft Unity	385	0.1100	2.2132×10^{-3}	2.0128	2.9934×10^{-3}	0.1041-0.1159
Spring.net	385	0.3140	5.3612×10^{-3}	1.7072	2.9456×10^{-3}	0.3082-0.3198

Fuente: Elaborada por los autores

Interpretación del resultado

Microsoft Unity presenta mejor tiempo de respuesta para la obtención de objetos al emplear un tiempo de 0.1100 milisegundos en comparación con Spring.Net que tarda 0.3140 milisegundos, presentando una relación de 1:2.85.

Calificación

En base a los criterios de calificación:

Tabla III. IX Calificación para la variable tiempo para la obtención de objetos

Rango de Valores	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
Menor a 0.1	5	Excelente	★★★★★
0.11 a 0.2	4	Muy Buena	★★★★☆
0.21 a 0.3	3	Buena	★★★☆☆
0.31 a 0.4	2	Regular	★★☆☆☆
0.41 a 0.5	1	Malo	★☆☆☆☆

Fuente: Elaborada por los autores

El framework Microsoft Unity cumple varias expectativas de la variable y que su tiempo de respuesta corresponde a 0.1100 milisegundos Utilizando la tabla anterior como referencia, se le asigna una calificación de **Muy Buena**, su forma gráfica será de cuatro estrellas y tiene un valor de 4.

Spring.Net por el contrario presenta tiempos de 0.3140 milisegundos, por tal motivo su calificación será 2 y en forma gráfica será 2 estrellas.

3.3.1.3. Variable 3: Memoria Usada por el contenedor

La variable representa la Cantidad de memoria reservada por el contenedor de IoC para la carga de objetos.

Esta variable es directamente proporcional al tiempo de obtención de los objetos del contenedor.

Para la ejecución de la prueba se considero la muestra de 385 elementos, a partir de lo cual se obtuvo el resultado (En el Anexo 4 se detalla el cálculo de la prueba):

Tabla III. X Resultado de la medición del uso de memoria de los frameworks

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC	Relación
Microsoft Unity	385	12.7355	0.4909	3.8548	2.6936×10^{-3}	12.7302-12.7408	1.0090:1
Spring.net	385	12.3507	0.2540	2.0562	2.8134×10^{-3}	12.3452-12.3562	1.0086:1
Aplicación sin IoC	385	12.2450	0.1240	1.0124	2.0543×10^{-3}	12.2410-12.2490	1:1

Fuente: Elaborada por los autores

Interpretación del resultado

Spring.Net es el contenedor de IoC que ocupa menos memoria para el almacenamiento de objetos presentando una relación de 1.0086:1 en relación a la aplicación sin ningún framework de IoC.

Calificación

En base a los criterios de calificación para el índice de relación:

Tabla III. XI Calificación para la variable memoria

Rango de Valores	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
1.0010: a 1:1.0089	5	Excelente	★★★★★
1.0090: a 1:1.0179	4	Muy Buena	★★★★☆
1.0180: a 1:1.0259	3	Buena	★★★☆☆
1.0260: a 1:1.0339	2	Regular	★★☆☆☆
1.0340: a 1:n	1	Malo	★☆☆☆☆

Fuente: Elaborada por los autores

El framework Spring.Net cumple varias expectativas de la variable y su uso de memoria tiene una relación de 1.0086:1 utilizando la tabla anterior como referencia, se le asigna una calificación de Excelente, su forma gráfica será de cinco estrellas y tiene un valor de 5.

Microsoft Unity por le contrario muestra una relación de 1.0090:1, por tal motivo su calificación será 4 y en forma gráfica será 4 estrellas.

3.3.1.4. Evaluación de resultados

Para la evaluación de las variables de este parámetro de comparación se utiliza las calificaciones establecidas en cada prueba.

Los resultados obtenidos se representan en forma numérica, porcentual, gráfica, y luego se realiza la respectiva interpretación de resultados

El valor máximo del parámetro de comparación se obtiene sumando el valor máximo de calificación de cada variable:

$$C_{max} = \sum_{i=1}^{n} (VM_i) = 5 + 5 + 5 = 15$$

El puntaje total para el framework Microsoft Unity en el parámetro de comparación se calcula en base a la siguiente fórmula:

$$C_u = \sum_{i=1}^{n} V_i = 5 + 4 + 4 = 13$$

El porcentaje de cumplimiento para Unity es:

$$P_u = \frac{C_u}{C_{max}} \times 100\% = \frac{13}{15} \times 100\% = 87.67\%$$

Para el framework Spring.Net los resultados son:

$$C_s = \sum_{i=1}^{n} V_i = 1 + 2 + 5 = 8$$

El porcentaje de cumplimiento para Spring.Net es:

$$P_s = \frac{C_s}{C_{max}} \times 100\% = \frac{8}{15} \times 100\% = 53.33\%$$

En la siguiente tabla se resume el puntaje de las variables en forma gráfica conforme a las pruebas realizadas:

Tabla III. XII Evaluación de resultados del parámetro rendimiento

FRAMEWORK / VARIABLES	MICROSOFT UNITY	SPRING.NET
Tiempo Creación Objetos	★★★★★	★☆☆☆☆
Tiempo Recuperación Objetos	★★★★☆	★★☆☆☆
Memoria usada	★★★★☆	★★★★★
Total	**13**	**8**

Fuente: Elaborada por los autores

Los resultados del parámetro de forma gráfica son:

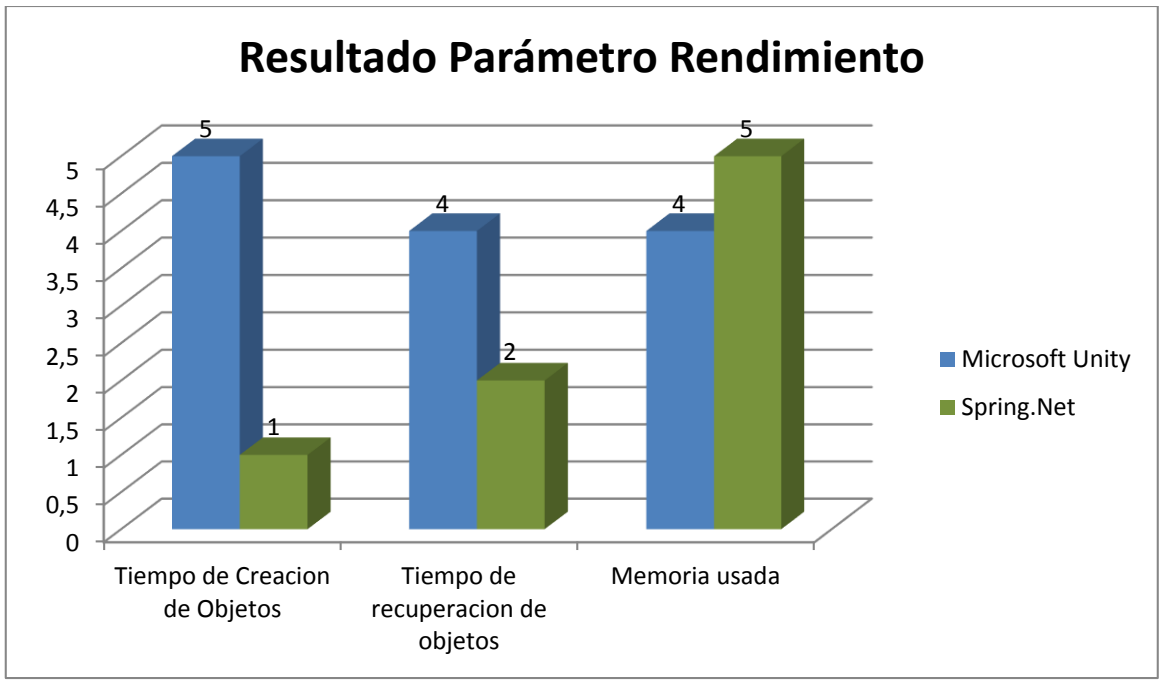


Figura III. 1 Resultados del parámetro Rendimiento

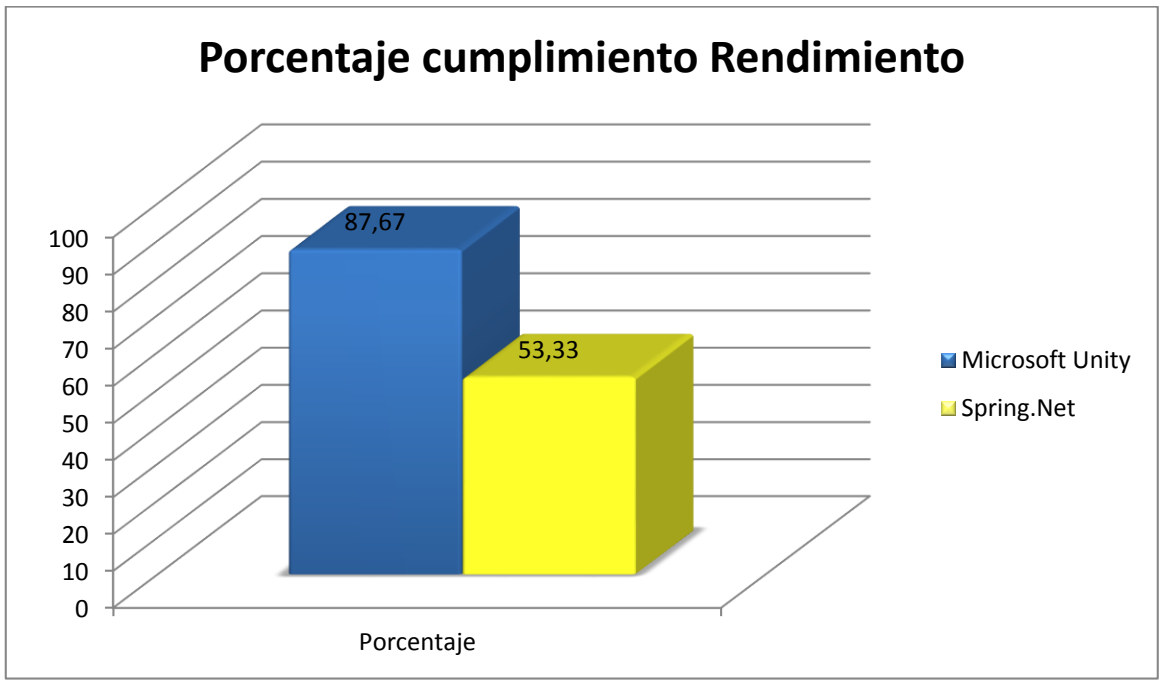


Figura III. 2 Porcentajes totales del parámetro Rendimiento

3.3.1.5. Interpretación de Resultados

El resultado equivalente cualitativo se lo realiza en base a la Tabla III. I. de acuerdo al porcentaje obtenido en cada una de las tecnologías.

Los resultados obtenidos en el Parámetro de Rendimiento indican que el framework **Microsoft Unity** cumple con el 87.67% de las variables establecidas, equivalente a **Excelente**. A diferencia del framework **Spring.Net** el cual cumple con el 53.33% de las variables establecidas, equivalente a **Bueno**.

3.3.2.Complejidad

El objetivo del parámetro de complejidad es determinar la dificultad de aprender, manejar y dar soporte a las aplicaciones que implementan el patrón de inyección de dependencias.

Tabla III. XIII Variables del parámetro de complejidad

COMPLEJIDAD	
VARIABLE	**DESCRIPCIÓN**
Disponibilidad de Información	Define la disponibilidad de información sobre el framework.
Esfuerzo de desarrollo	Especifica la dificultad de desarrollo para la integración de la aplicación con el framework de DI.
Esfuerzo de Implementación	Define el esfuerzo necesario para la implementación de la solución con el framework de DI.

Fuente: Elaborada por los autores

3.3.2.1. Variable 1: Disponibilidad de Información

Esta variable determina la disponibilidad de información para el framework de inyección de dependencias, considerando la información publicada en el sitio oficial de cada uno de los frameworks de DI, libros oficiales, ejemplos, documentación y referencias de las versiones, tutoriales y código fuente.

Los indicadores a evaluar son los siguientes:

- Manuales
- Libros
- Tutoriales/Ejemplos
- Artículos
- Podcast/Webcasts
- Foros

El cumplimiento de todos estos valores tendrá una valoración de Excelente,

a. Framework de Inyección de Dependencias Spring.NET

- *Manuales de Referencia* Spring provee manuales de referencia para cada una de sus 8 versiones (1.0.2 - 1.1 - 1.1.1 - 1.1.2 - 1.2.0 - 1.3.0 - 1.3.1 - 1.3.2). En total existen 15 manuales de referencia sobre este framework.
- *Libros* según la documentación oficial existe un libro oficialmente editado,
  i. "*Applying Domain-Driven Design and Patterns: With Examples in C# and .NET"* por Jimmy Nilsson el mismo que contiene secciones sobre IoC, DI y programación orientada a aspectos y describe el uso de Spring.NET,

  Según Amazon se encontraron los libros:

  ii. Spring Recipes: A Problem-Solution Approach por Gary Mak
  iii. The Definitive Guide to Spring for .NET Russ Miles

iv. Dependency Injection in .NET Mark Seeman

v. Pro Spring 3.0 –Clarence Ho

vi. Spring In Action Craig Walls

vii. Just Spring - Madhusudhan Konda

- *Presentaciones, Videos, Entrevistas, Podcast* Spring .Net cuenta con diferentes presentaciones del uso de este framework que se han realizado en diferentes conferencias. Además cuenta con diferentes podcasts describiendo introducciones, uso del framework, etc.
- *Artículos* A mas de la documentación oficial sobre las versiones Spring.Net cuenta con diferentes artículos aislados sobre este framework,
- *Foros* Srping.Net posee foros donde los diferentes usuarios de este framework pueden dar a conocer sus dudas que se presentan al momento de desarrollar aplicaciones.
- *Ejemplos/Tutoriales* el sitio oficial de provee de 14 tutoriales con sus respectivos códigos fuentes.

b. Framework de Inyección de Dependencias Unity

- *Manuales de Referencia* Unity provee manuales de referencia para cada una de sus 6 versiones (2.1 – 2.1 (Silverlight) - 2.0 – 2.0(Silverlight) - 1.2 – 1.2 (Silverlight)). En total existen 6 manuales de referencia sobre este framework.
- *Libros* Los libros encontrados en Amazon para el aprendizaje de Unity:

i. Dependency Injection in .NET Mark Seeman

ii. Developer's Guide to Microsoft Enterprise Library, C# Edition

iii. Microsoft Enterprise Library 5.0 - Sachin Joshi

iv. Microsoft® .NET: Architecting Applications for the Enterprise (Pro-Developer)

- *Presentaciones, Videos, Entrevistas, Podcast* La ayuda de Unity presenta diferentes videos de webcasts, podcasts, y entrevistas sobre este framework.
- *Artículos* Complementando la documentación oficial Unity, este framework posee diferentes artículos sobre actualizaciones, novedades, implementaciones, etc. De este framework.
- *Foros* El Sitio Oficial de Unity implementa un apartado foros donde diferentes personas exponen sus dudas al momento de desarrollar aplicaciones.
- *Ejemplos/Tutoriales* el sitio oficial de un tutorial con sus respectivos ejemplos por cada una de sus versiones.

Tabla III. XIV Parámetros de valoración para los indicadores de la variable disponibilidad de información

PARÁMETROS DE VALORACIÓN PARA LOS INDICADORES	
CRITERIO	**VALOR**
Mayor a 10	5
Entre 5 y 10	3
De 1 a 4	2
Ninguno	0

Fuente: Elaborada por los autores

Resultados Totales

Tabla III. XV Resultados de la variable disponibilidad de información

Indicadores / Framework	Manuales	Libros	Tutoriales /Ejemplos	Artículos	Podcasts /Webcasts	Foros	Total
Microsoft Unity	**5**	**2**	**5**	**5**	**5**	**5**	**27**
Spring.net	**5**	**3**	**5**	**5**	**5**	**5**	**28**

Fuente: Elaborada por los autores

Calificación

Tabla III. XVI Calificación para la variable disponibilidad de información

Rango de Valores	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
30 – 27	1	Excelente	
26 – 20	2	Muy Buena	
19 – 15	3	Buena	
14 – 10	4	Regular	
9 – 0	5	Malo	

Fuente: Elaborada por los autores

Los dos frameworks cumplen con todas las expectativas de la variable ya que poseen en sus sitios oficiales información detallada para el aprendizaje del framework de inyección de dependencias.

Utilizando la Tabla III.I como referencia, se le asigna a los dos frameworks la calificación de **Excelente,** su forma gráfica será de **cinco estrellas,** teniendo un valor de **1**.

3.3.2.2. Variable 2: Esfuerzo de desarrollo

Esta variable mide el esfuerzo de desarrollo para la integración de la aplicación con el framework de inyección de dependencias. El valor de esta variable esta directamente relacionada con las métricas de código.

Los indicadores a considerar para la evaluación de la variable son los siguientes:

- Índice de mantenimiento: es un valor de índice entre 0 y 100 que representa la facilidad relativa de mantenimiento del código
- Complejidad Ciclomática: mide la complejidad estructural del código de acuerdo al nivel de anidamiento.

- Profundidad de herencia: indica el número de definiciones de clase que se extienden a la raíz de la jerarquía de clases.
- Acoplamiento de clases: mide las relaciones de llamada entre las clases del sistema.
- Líneas de código: indica el número aproximado de líneas del código basado en el código IL

La prueba fue ejecutada con las herramientas de Visual Studio para el cálculo de métricas, el resultado se presenta en la tabla:

Tabla III. XVII Resultado de la medición de la variable esfuerzo de desarrollo

Indicador Framework	Índice de mantenimiento	Complejidad Ciclomática	Profundidad de herencia	Acoplamiento de clases	Líneas de código
Microsoft Unity	65	3	1	10	26
Spring.Net	62	4	1	14	29

Fuente: Elaborada por los autores

Calificación

Las siguientes tablas presentan el sistema de valoración para cada uno de los indicadores, los valores de mayor puntaje indican mayor complejidad:

Tabla III. XVIII Parámetros de valoración para el indicador índice de mantenimiento

PARÁMETROS DE VALORACIÓN PARA EL INDICADOR ÍNDICE DE MANTENIMIENTO	
CRITERIO	**VALOR**
Entre 40 y 100	1
Entre 30 y 39	2
Entre 20 y 29	3
Entre 10 y 19	4
Entre 0 y 9	5

Fuente: Elaborada por los autores

Tabla III. XIX Parámetros de valoración para el indicador complejidad ciclomática

PARÁMETROS DE VALORACIÓN PARA EL INDICADOR COMPLEJIDAD CICLOMÁTICA	
CRITERIO	**VALOR**
Entre 1 y 3	1
Entre 4 y 6	2
Entre 6 y 9	3
Entre 10 y 13	4
Mayor que 13	5

Fuente: Elaborada por los autores

Tabla III. XX Parámetros de valoración para el indicador profundidad de herencia

PARÁMETROS DE VALORACIÓN PARA EL INDICADOR PROFUNDIDAD DE HERENCIA	
CRITERIO	**VALOR**
Entre 1 y 2	1
Entre 3 y 4	2
Entre 5 y 6	3
Entre 7 y 8	4
Mayor que 8	5

Fuente: Elaborada por los autores

Tabla III. XXI Parámetros de valoración para el indicador acoplamiento de clases

PARÁMETROS DE VALORACIÓN PARA EL INDICADOR ACOPLAMIENTO DE CLASES	
CRITERIO	**VALOR**
Entre 1 y 10	1
Entre 11 y 20	2
Entre 21 y 30	3
Entre 31 y 40	4
Mayor que 40	5

Fuente: Elaborada por los autores

Tabla III. XXII Parámetros de valoración para el indicador líneas de código

PARÁMETROS DE VALORACIÓN PARA EL INDICADOR LÍNEAS DE CÓDIGO	
CRITERIO	**VALOR**
Entre 1 y 15	1
Entre 16 y 30	2
Entre 31 y 45	3
Entre 46 y 60	4
Mayor que 60	5

Fuente: Elaborada por los autores

Resultados Totales:

Tabla III. XXIII Resultados de la variable esfuerzo de desarrollo

Indicador / Framework	Índice de mantenimiento	Complejidad Ciclomática	Profundidad de herencia	Acoplamiento de clases	Líneas de código	Total
Microsoft Unity	1	1	1	1	2	6
Spring.Net	1	2	1	2	2	8

Fuente: Elaborada por los autores

Para la calificación a la variable se considera un sistema de valoración invertido ya que la variable complejidad es negativa para la evaluación del parámetro:

Tabla III. XXIV Calificación para la variable esfuerzo de desarrollo

Rango de Valores	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
1 – 6	1	Fácil	
7 – 12	2	Poco Fácil	
13 – 18	3	Normal	
19 – 24	4	Poco Difícil	
>25	5	Difícil	

Fuente: Elaborada por los autores

Utilizando la Tabla anterior como referencia, se asigna a Microsoft Unity la calificación de **Fácil,** su forma gráfica será de **una estrella,** teniendo un valor de **1**, de igual forma la calificación para Spring.Net será **Poco Fácil** con un valor de **dos estrellas** y un valor de **2**.

El resultado indica que requiere de más esfuerzo una implementación con el framework spring.net.

3.3.2.3. Variable 3: Esfuerzo de Implementación

Esta variable define el esfuerzo necesario para la implementación de la solución con el framework de DI.

La medición de esta variable se lo hace a través del soporte existente para el despliegue de la aplicación web.

a. Generación de archivos habilitantes

Los frameworks Microsoft Unity y Spring.Net son integrados como librerías dentro de la generación de archivos habilitantes para efectuar el despliegue, las librerías inyectadas no son incluidas en la generación de habilitantes debido a lo cual es necesario en ambos casos incluir a las librerías que se inyectan a través del framework. Por lo cual los frameworks Spring.Net y Microsoft Unity cumplen al 100% con el indicador.

b. Dificultad en la instalación

Para ambas tecnologías el proceso de despliegue es sencillo y consiste en copiar los habilitantes al directorio de instalación y configurar el sitio desde a consola del Proveedor de Servicios Web. Por lo cual los frameworks Spring.Net y Microsoft Unity cumplen al 100% con el indicador.

Calificación

De acuerdo a la tabla:

Tabla III. XXV Calificación para la variable esfuerzo de implementación

Rango de Valores	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
>80%y<=100%	1	Fácil	
>60% y <=80%	2	Poco Fácil	
> 40% y <=60%	3	Normal	
> 20% y <=40%	4	Poco Difícil	
>=0% y <=20%	5	Difícil	

Fuente: Elaborada por los autores

Los framewoks Microsoft Unity y Spring.Net tiene una calificación de Fácil con un equivalente a 5 estrellas y una valoración de 1.

3.3.2.4. Evaluación de resultados

Para la evaluación de las variables de este parámetro de comparación se utiliza las calificaciones establecidas en cada prueba. El resultado se tomará con signo negativo ya que la complejidad es una variable negativa.

Los resultados obtenidos se representan en forma numérica, porcentual, gráfica, y luego se realiza la respectiva interpretación de resultados

El valor máximo del parámetro de comparación se obtiene sumando el valor máximo de calificación de cada variable:

$$C_{max} = \sum_{i=1}^{n} (VM_i) = 5 + 5 + 5 = 15$$

El puntaje total para el framework Microsoft Unity en el parámetro de comparación se calcula en base a la siguiente fórmula:

$$C_u = \sum_{i=1}^{n} V_i = 1 + 1 + 1 = 3$$

El porcentaje de cumplimiento para Unity es:

$$P_u = \frac{C_u}{C_{max}} \times 100\% = \frac{3}{15} \times 100\% = 20\%$$

Para el framework Spring.Net los resultados son:

$$C_s = \sum_{i=1}^{n} V_i = 1 + 2 + 1 = 4$$

El porcentaje de cumplimiento para Spring.Net es:

$$P_s = \frac{C_s}{C_{max}} \times 100\% = \frac{4}{15} \times 100\% = 26.67\%$$

En la siguiente tabla se resume el puntaje de las variables en forma gráfica conforme a las pruebas realizadas:

Tabla III. XXVI Evaluación de resultados del parámetro complejidad

FRAMEWORK / VARIABLES	MICROSOFT UNITY	SPRING.NET
Disponibilidad de Información	★★★★★	★★★★★
Esfuerzo de Desarrollo	★★★★★	★★★★☆
Esfuerzo de Implementación	★★★★★	★★★★★
Total	**3**	**4**

Fuente: Elaborada por los autores

Los resultados del parámetro de forma gráfica son:

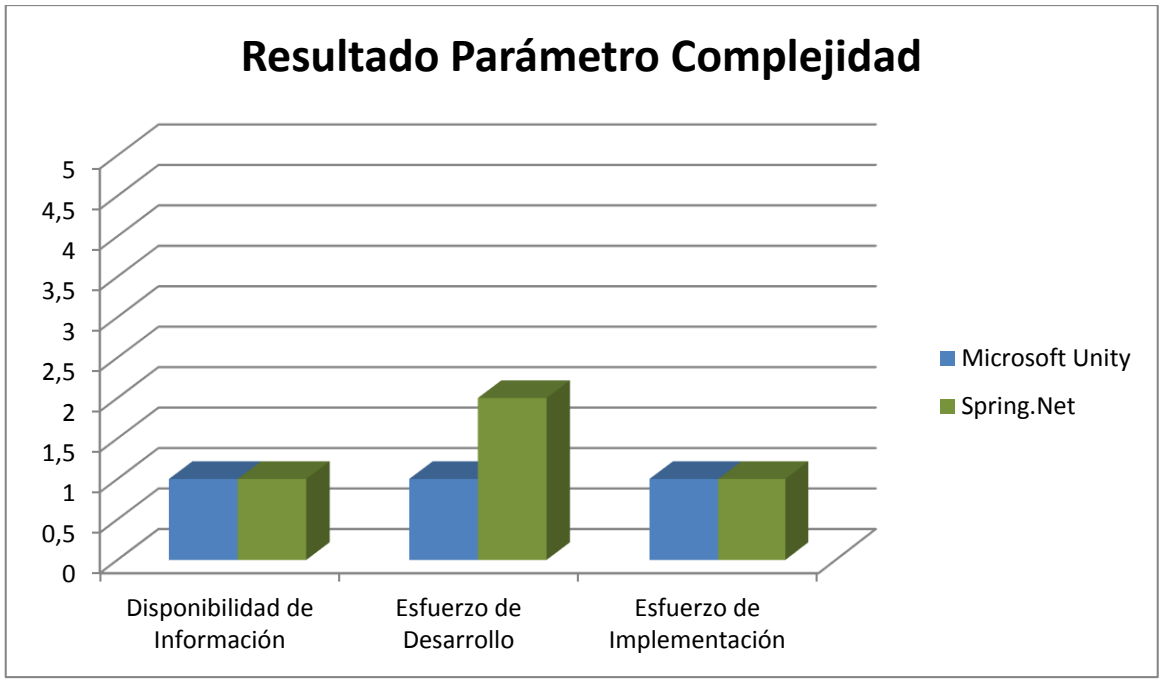


Figura III. 3 Resultados totales del parámetro Complejidad

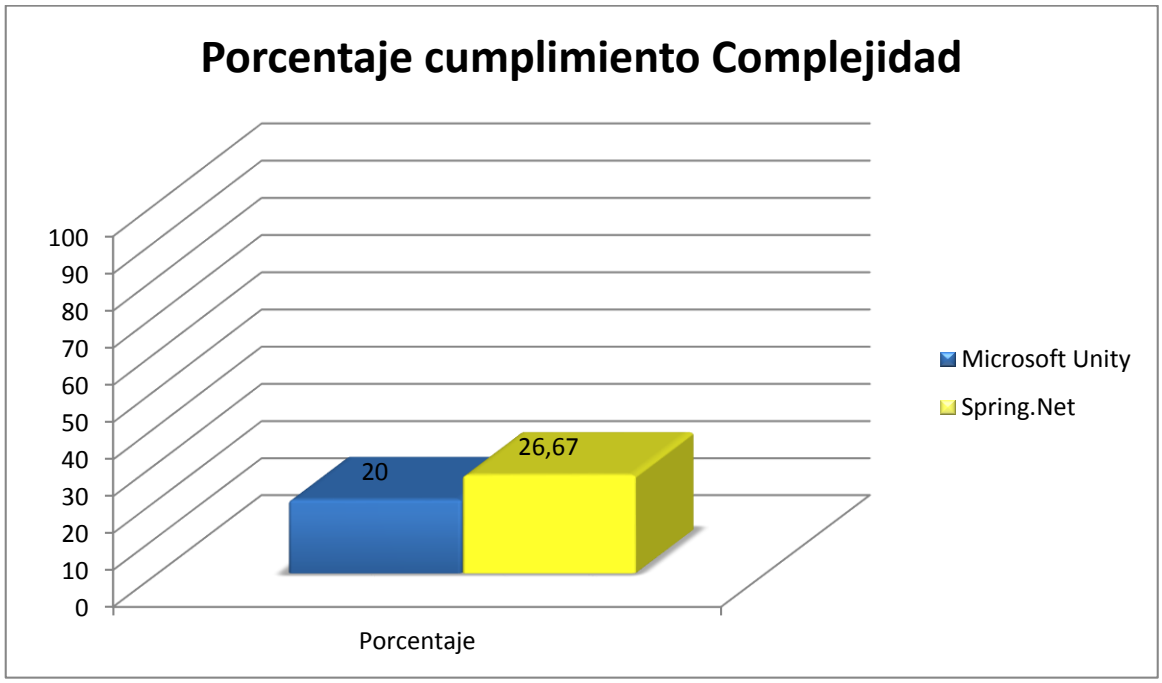


Figura III. 4 Porcentajes totales del parámetro Complejidad

3.3.2.5. Interpretación de Resultados

El resultado equivalente cualitativo se lo realiza en base a la Tabla III. I. de acuerdo al porcentaje obtenido en cada una de las tecnologías.

Los resultados obtenidos en el Parámetro de Complejidad indican que el framework **Microsoft Unity** cumple con el 20% de las variables establecidas, indicando una baja complejidad con una calificación de -2. A diferencia del framework **Spring.Net** el cual cumple con el 26.67% de las variables establecidas, demostrando así un mayor nivel de complejidad con una calificación de -3.

3.3.3. Requerimientos de hardware y software

El objetivo del parámetro es definir los requisitos necesarios de hardware y software para usar el contenedor de objetos.

Tabla III. XXVII Variables del parámetro de requerimientos de hardware y software

REQUERIMIENTOS DE HARDWARE Y SOFTWARE	
VARIABLE	**DESCRIPCIÓN**
Uso de Procesador	Define le porcentaje de procesador usado por el framework de DI.
Disco Duro	Define la cantidad de espacio en disco necesaria para almacenar un volcado de memoria.

Fuente: Elaborada por los autores

3.3.3.1. Variable 1: Uso de Procesador

Define le porcentaje de procesador usado por el framework de DI. Para la obtención de datos se uso la herramienta Perfmon integrada con Microsoft Windows. El cálculo del uso del procesador datos se explica en el anexo 5.

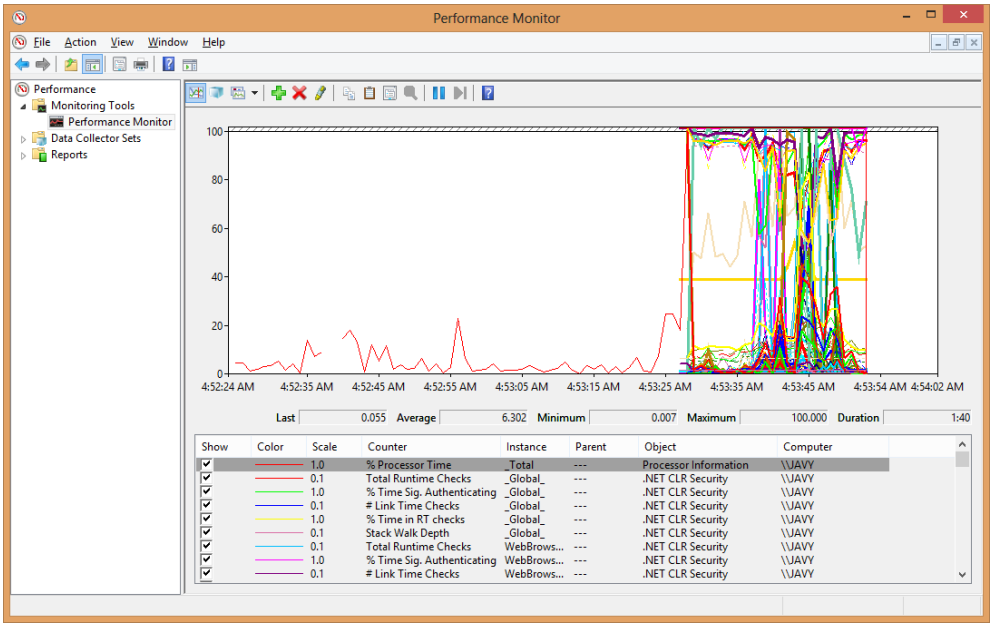


Figura III. 5 Uso de procesador con la herramienta Perfom

Luego de efectuar la medición se tiene los resultados:

Tabla III. XXVIII Resultado de la medición del uso del procesador

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\overline{x}}$	IC	Relación
Microsoft Unity	385	24.2540	0.5300	2.1854	2.3302	24.4871-24.4871	1.2034:1
Spring.net	385	26.9893	0.5227	1.9369	1.5412	26.8351-27.1435	1.3391:1
Aplicación sin IoC	385	20.1551	0.4815	2.4388	1.7020	19.9849-20.3253	1:1

Fuente: Elaborada por los autores

Calificación

En base a los criterios de calificación para el índice de relación:

Tabla III. XXIX Calificación para la variable uso del procesador

Rango de Valores Relación	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
1 a 1.5	5	Excelente	
1.51 a 2	4	Muy Buena	
2.1: a 2.5	3	Buena	
2.51 a 3	2	Regular	
3 a n	1	Malo	

Fuente: Elaborada por los autores

Ambos frameworks tienen un buen desempeño en relación al uso de CPU, por tal motivo son evaluados con la calificación de Excelente, su forma gráfica será de cinco estrellas y tienen un valor de 5.

3.3.3.2. Variable 2: Uso de Disco Duro

La variable representa la cantidad de almacenamiento necesaria por el contenedor para efectuar un volcado de la estructura de almacenamiento de instancias y sus objetos contenidos.

Para la ejecución de la prueba se considero la muestra de 385 elementos, a partir de lo cual se obtuvo el resultado (En el Anexo 6 se detalla el cálculo de la prueba):

Tabla III. XXX Resultado de la medición del uso de disco duro

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC
Microsoft Unity	385	124.9433	2.8162	2.2547	0.0103	124.9230-124.9636
Spring.net	385	124.5585	2.5055	2.0115	0.0058	124.5471-124.5699

Fuente: Elaborada por los autores

Calificación

En base a los criterios de calificación para el almacenamiento del volcado sobre la prueba de 5 objetos:

Tabla III. XXXI Calificación para la variable uso de disco duro

Rango de Valores	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
100 a 125 KB	5	Excelente	★★★★★
126 a 150 KB	4	Muy Buena	★★★★☆
151 a 200KB	3	Buena	★★★☆☆
201 a 250 KB	2	Regular	★★☆☆☆
>251KB	1	Malo	★☆☆☆☆

Fuente: Elaborada por los autores

Ambos frameworks tienen un buen desempeño en al uso de almacenamiento para el volcado, por tal motivo son evaluados con la calificación de Excelente, su forma gráfica será de cinco estrellas y tienen un valor de 5.

3.3.3.3. Evaluación de resultados

Para la evaluación de las variables de este parámetro de comparación se utiliza las calificaciones establecidas en cada prueba.

Los resultados obtenidos se representan en forma numérica, porcentual, gráfica, y luego se realiza la respectiva interpretación de resultados

El valor máximo del parámetro es:

$$C_{max} = \sum_{i=1}^{n} (VM_i) = 5 + 5 = 10$$

El puntaje total para el framework Microsoft Unity en el parámetro de comparación se calcula en base a la siguiente fórmula:

$$C_u = \sum_{i=1}^{n} V_i = 5 + 5 = 10$$

El porcentaje de cumplimiento para Unity es:

$$P_u = \frac{C_u}{C_{max}} \times 100\% = \frac{10}{10} \times 100\% = 100\%$$

Para el framework Spring.Net los resultados son:

$$C_s = \sum_{i=1}^{n} V_i = 5 + 5 = 10$$

El porcentaje de cumplimiento para Spring.Net es:

$$P_s = \frac{C_s}{C_{max}} \times 100\% = \frac{10}{10} \times 100\% = 100\%$$

En la siguiente tabla se resume el puntaje de las variables en forma gráfica conforme a las pruebas realizadas:

Tabla III. XXXII Evaluación de resultados del parámetro hardware y software

FRAMEWORK / VARIABLES	MICROSOFT UNITY	SPRING.NET
Uso de Procesador	★★★★★	★★★★★
Uso de disco duro	★★★★★	★★★★★
Total	**5**	**5**

Fuente: Elaborada por los autores

Los resultados del parámetro de forma gráfica son:

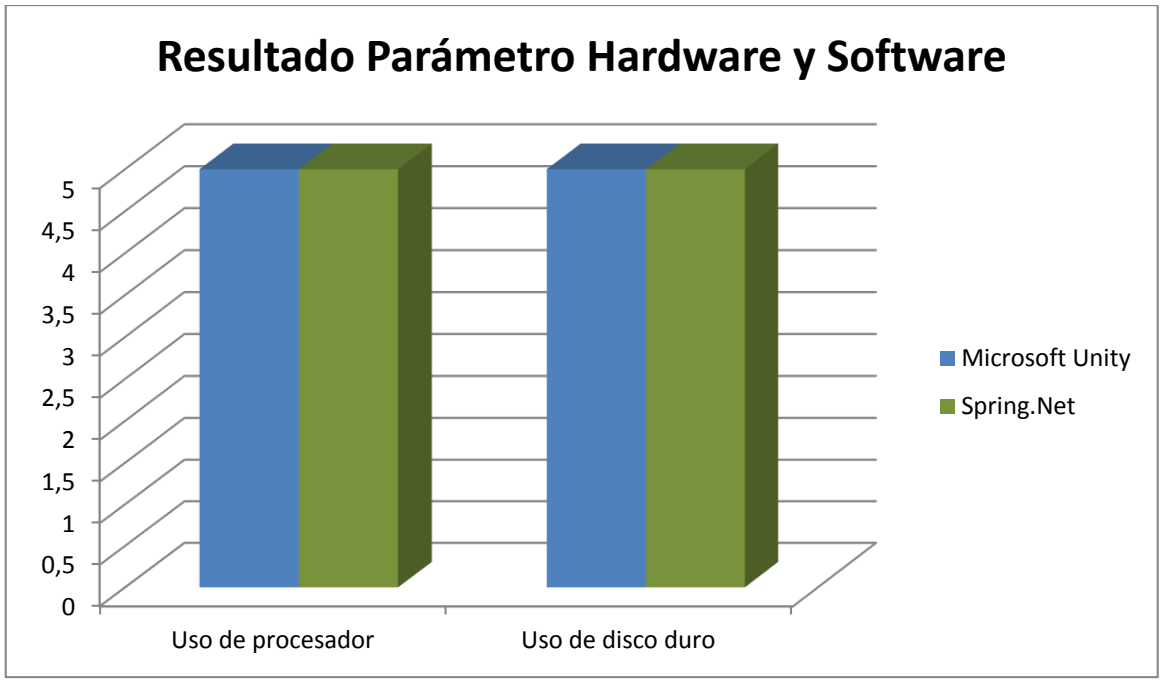


Figura III. 6 Resultados totales del parámetro Hardware y Software

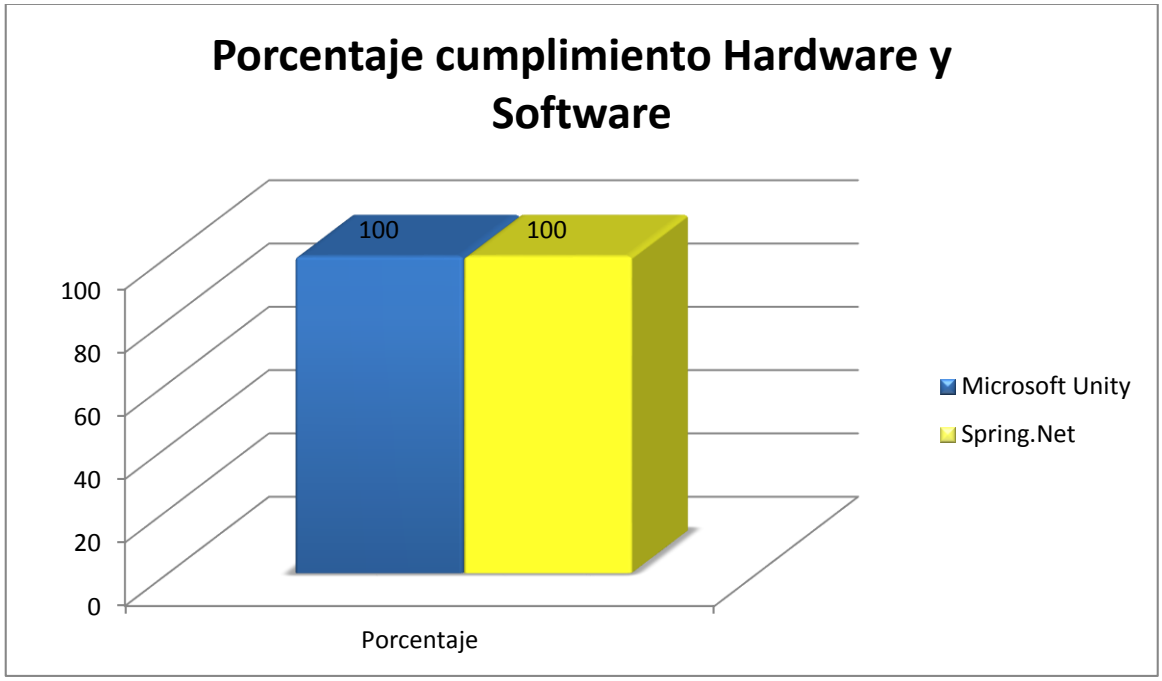


Figura III. 7 Porcentajes totales del parámetro hardware y software

3.3.3.4. Interpretación de Resultados

El resultado equivalente cualitativo se lo realiza en base a la Tabla III. I. de acuerdo al porcentaje obtenido en cada una de las tecnologías.

Los resultados obtenidos en el Parámetro de Hardware y Software indican que el framework **Microsoft Unity** y **Spring.Net** cumplen con el 100% de las variables establecidas, equivalente a una calificación de **Excelente**.

3.3.4. Seguridad

Este parámetro define las características de seguridad que poseen los objetos y los contenedores de inversión de control.

Para la medición de este parámetro se considera un análisis de seguridad sobre las fuentes de los frameworks considerando los parámetros usados para la resolución de tipos, el control de seguridad, y la suplantación de identidad.

Tabla III. XXXIII Variables del parámetro seguridad

SEGURIDAD	
VARIABLE	**DESCRIPCIÓN**
Mecanismos de seguridad sobre la inyección y resolución de objetos.	Define las falencias de seguridad que presenta el código de cada framework cuando realiza la carga de ensamblados.

Fuente: Elaborada por los autores

3.3.4.1. Variable 1: Mecanismos de seguridad sobre la inyección y resolución de objetos

Esta variable define las falencias de seguridad que presenta el código de cada framework cuando realiza la carga de ensamblados. Para la medición de la variable se aplican los siguientes indicadores:

- Evitar el manejo de seguridad imperativa: Indica si existen atributos de seguridad establecidos en forma imperativa (escritos directamente en el código con la inicialización de nuevos objetos de seguridad) los cuales pueden ser usados para inyectar código con omisión de reglas y niveles de seguridad.
- Aseguramiento de Aserciones: Indica si existen métodos que validen permisos sin efectuar la comprobación de seguridad en el invocador dejando una debilidad de seguridad en el código.
- Verificación de identidad de Métodos en ensamblados seguros: indica si permite la llamada a métodos desde un ensamblado firmado a otro con el atributo AllowPartiallyTrustedCallersAttribute el cual suprime la verificación de identidad de la invocación introduciendo un problema de suplantación.

La prueba fue ejecutada con las herramientas de Visual Studio para el análisis del código estático con la aplicación de reglas de seguridad recomendadas por Microsoft, el resultado se presenta en la tabla:

Tabla III. XXXIV Resultado de medir mecanismos de seguridad sobre la inyección y resolución de objetos

Indicador / Framework	Evitar el manejo de seguridad imperativa	Aseguramiento de Aserciones	Verificación de identidad de Métodos en ensamblados seguros	Totales
Microsoft Unity	1	5	5	11

Spring.Net	1	1	1	3

Fuente: Elaborada por los autores

Calificación

De acuerdo a la tabla:

Tabla III. XXXV Calificación para la variable mecanismos de seguridad sobre la inyección y resolución de objetos

Rango de totales	Valoración Cuantitativa	Valoración Cualitativa	Forma Gráfica
13-15	5	Muy Seguro	
10-12	4	Seguro	
7-9	3	Poco Seguro	
3-6	1	Inseguro	

Fuente: Elaborada por los autores

El framework Microsoft Unity presenta una buena prestación de seguridad en relación a la carga de ensamblados y resolución de dependencias, por tal motivo se le asigna una calificación de Seguro, su forma gráfica será de cuatro estrellas y tienen un valor de 4, para Spring.Net la calificación asignada es Inseguro, su forma grafica es de una estrella con un valor de 1.

3.3.4.2. Evaluación de resultados

Los resultados obtenidos se representan en forma numérica, porcentual, gráfica, y luego se realiza la respectiva interpretación de resultados

El valor máximo del parámetro de comparación se obtiene sumando el valor máximo de calificación de cada variable:

$$C_{max} = \sum_{i=1}^{n} (VM_i) = 5$$

El puntaje total para el framework Microsoft Unity en el parámetro de comparación se calcula en base a la siguiente fórmula:

$$C_u = \sum_{i=1}^{n} V_i = 4$$

El porcentaje de cumplimiento para Unity es:

$$P_u = \frac{C_u}{C_{max}} \times 100\% = \frac{4}{5} \times 100\% = 80\%$$

Para el framework Spring.Net los resultados son:

$$C_s = \sum_{i=1}^{n} V_i = 1$$

El porcentaje de cumplimiento para Spring.Net es:

$$P_s = \frac{C_s}{C_{max}} \times 100\% = \frac{1}{5} \times 100\% = 20\%$$

En la siguiente tabla se resume el puntaje de las variables en forma gráfica conforme a las pruebas realizadas:

Tabla III. XXXVI Evaluación de resultados del parámetro seguridad

FRAMEWORK VARIABLES	MICROSOFT UNITY	SPRING.NET
Mecanismos de seguridad sobre la inyección y resolución de objetos	★★★★☆	★☆☆☆☆
Total	**4**	**1**

Fuente: Elaborada por los autores

Los resultados del parámetro de forma gráfica son:

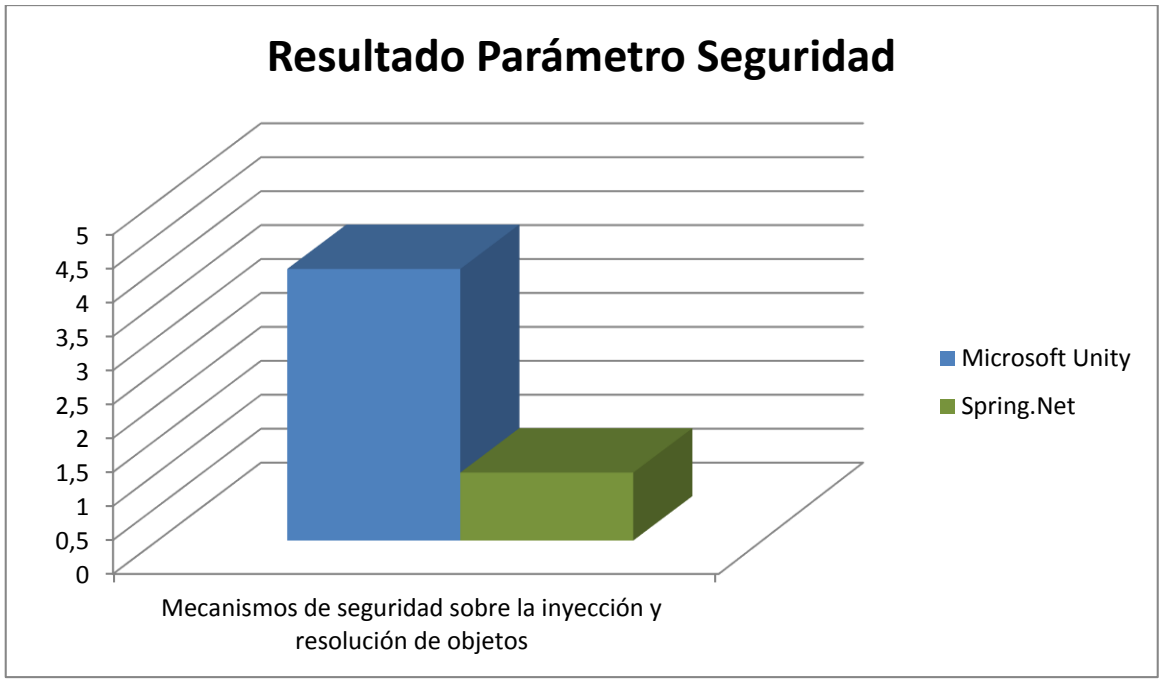


Figura III. 8 Resultados totales del parámetro seguridad

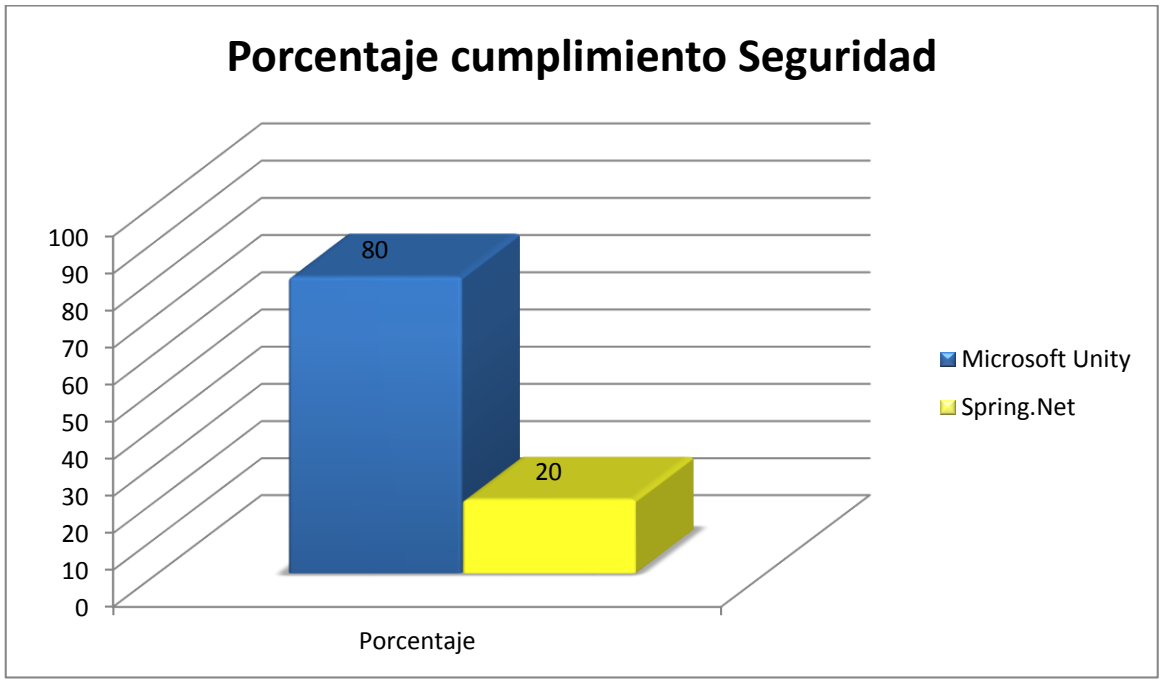


Figura III. 9 Resultados totales del parámetro seguridad

3.3.4.3. Interpretación de Resultados

El resultado equivalente cualitativo se lo realiza en base a la Tabla III. I. de acuerdo al porcentaje obtenido en cada una de las tecnologías.

Los resultados obtenidos en el Parámetro de Seguridad indican que el framework **Microsoft Unity** cumple con el 80% de las variables establecidas, equivalente a **Muy Bueno**. A diferencia del framework **Spring.Net** el cual cumple con el 20% de las variables establecidas, equivalente a **Malo**.

3.4. Demostración de la Hipótesis

Considerando los parámetros establecidos en el estudio, se presenta un cuadro de resumen de cada parámetro, en donde se indica la calificación de cada uno

Tabla III. XXXVII Resumen de cada parámetro

	Rendimiento		Complejidad		Requerimientos de hardware y Software		Seguridad	
	Valor	**Calificación**	**Valor**	**Calificación**	**Valor**	**Calificación**	**Valor**	**Calificación**
Microsoft Unity	5	Excelente	-2	Poco Complejo	5	Excelente	4	Muy Bueno
Spring.Net	3	Bueno	-3	Parcialmente Complejo	5	Excelente	1	Malo

Fuente: Elaborada por los autores

El puntaje resumido es:

Tabla III. XXXVIII Puntaje resumido de cada parámetro

	Rendimiento	Complejidad	Requerimientos de Hardware y Software	Seguridad	Total	%
Microsoft	5	-2	5	4	12	85.71

Unity						
Spring.net	3	-3	5	1	6	42.86
Calificación Máxima	5	-1	5	5	14	100

Fuente: Elaborada por los autores

El puntaje grafico es:

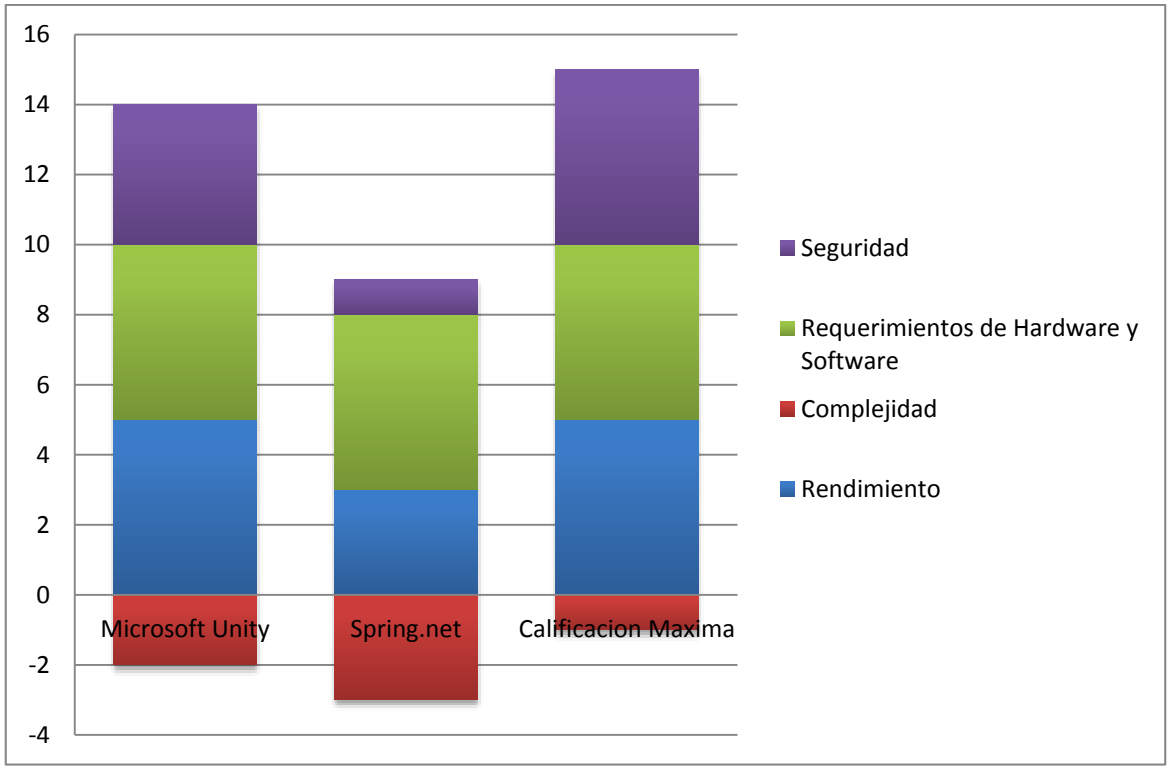


Figura III. 10 Puntaje gráfico resumido de los parámetros

En base a los resultados se interpreta:

- El framework Microsoft Unity 2.0 presenta una gran ventaja en rendimiento respecto a Spring.Net cumpliendo el 87.67% de las variables asignadas para el estudio contra el 53.33%.

- El framework mas complejo es Spring.Net demandando de mayor esfuerzo en desarrollo mostrando un 26.67% de complejidad frente al 20% del framework Unity.
- En relación al uso de hardware y software ambos frameworks utilizan una cantidad de recursos similares.
- En relación al parámetro seguridad Microsoft Unity provee de mejores características de seguridad a momento de verificar la identidad de los ensamblados y asegurar la introducción de código malicioso, por tal motivo el cumplimento del parámetro es de 80% a diferencia del Spring.Net el cual cumple con el 20%.

3.4.1. Comprobación de Hipótesis

En base a las mediciones realizadas y a los datos obtenidos para su posterior análisis haciendo uso de la estadística descriptiva se puede llegar a comprobar la hipótesis planteada en la presente tesis, mediante la cual se puede afirmar que el framework Microsoft Unity 2.0 es la plataforma que provee las mejores características de rendimiento, seguridad, complejidad y uso de recursos cumpliendo con el 85.71% de los parámetros elegidos para el estudio frete al 42.86% de cumplimiento del framework Spring.Net.

CAPÍTULO IV

4. DESARROLLO DEL SISTEMA PARA EL CONTROL DE PRODUCCIÓN EN LAS ESCUELAS RADIOFÓNICAS POPULARES DEL ECUADOR (ERPE)

En este capítulo, con el framework de inyección de dependencias seleccionado en el Capítulo III, se procede al desarrollo de la aplicación “TerraTech” para las Escuelas Radiofónicas Populares del Ecuador (ERPE), utilizando la metodología Microsoft Solutions Framework (MSF), de esta manera se logra vincular la parte teórica con la parte práctica.

4.1. Escuelas Radiofónicas Populares del Ecuador

ERPE, nació para educar a la gente a través de la radio, se inició con alfabetización, luego con el Sistema de Teleducación y finalmente con la capacitación en agricultura como una alternativa para mejorar el nivel de vida. Consciente de la problemática del campesino en la agricultura, inició hace 18 años a producir orgánicamente hortalizas, al tiempo que daba talleres a los campesinos en el Centro de Capacitación de San Antonio, lugar en donde se encuentra la Granja.

Pero los campesinos necesitaban una alternativa a la producción orgánica sustentable y nace el proyecto de Producción, Acopio, procesamiento y Comercialización de Quinua Orgánica el año de 1998, proceso que cada día va tomando más fuerza. Pero la gente necesita de más oportunidades, y ERPE los apoya capacitándoles, es así que con el apoyo del FOES, se inició La Formación Profesional en Agropecuaria Orgánica hace dos años, los mismos que ya vienen dando buenos resultados.

Contexto del Negocio

La organización ERPE desde hace algunos años brinda servicios de apoyo comunitario con un alto enfoque productivo orgánicos, tanto para consumo local como para la exportación.

Con el apoyo de la Unión Europea, ERPE ha desarrollado una serie de proyectos cuyo objetivo es el crecimiento económico de la población ecuatoriana, fomentando los principios de agricultura orgánica y responsable.

Los principales productos de comercialización son: quinua orgánica, amaranto y hierbas aromáticas, cultivados por una serie de comunidades registradas en el programa.

Visión

Actualmente como una visión tecnológica empresarial E.R.P.E ha visto conveniente la concepción de una estrategia informática que le permita automatizar todos los procesos que controlan la producción y la distribución de los productos de la organización al mercado.

Tal implementación tecnológica permitirá a la organización E.R.P.E proyectarse internacionalmente como una empresa exportadora de renombre mundial, facilitando el control de calidad del producto de exportación por parte de los clientes mediante

tecnologías Web.

Objetivos

- El principal objetivo de estas capacitaciones es llegar a abrir un mercado orgánico, en donde los agricultores serán los proveedores directos de los productos sin la participación de intermediarios,
- Con la apertura del mercado orgánico, los productores se pueden organizar y asociarse para emprender con microempresas para producir elaborados, de esta manera mejorar sus economías.
- Tener un grupo de agricultores capacitados en producción agropecuaria orgánica.
- Promocionar internacionalmente los diferentes insumos agrícolas que produce la organización E.R.P.E
- Gestionar el control de producción de los diferentes productos en tiempo real.

4.2. Microsoft Solutions Framework (MSF)

Microsoft Solutions Framework (MSF) es la nueva propuesta de Microsoft en el mundo de procesos y prácticas ágiles de desarrollo de software.

Microsoft Solutions Framework es una metodología ágil para el desarrollo de software que brinda un conjunto de modelos y principios para encontrar soluciones a los problemas de las empresas inmersas en el diseño y desarrollo, asegurando que los elementos del proyecto, es decir, procesos, recursos humanos y herramientas sean manejados con éxito.

Lo interesante de la metodología Microsoft Solutions Framework es que combina las mejores características y principios de los modelos de cascada y espiral. Del primero extrae la claridad que caracteriza a este modelo; mientras que del segundo aprovecha las ventajas de los puntos de transición.

Características

Microsoft Solutions Framework tiene como principales características:

- Metodología ágil para el desarrollo de software.
- Puede ser utilizado a pequeña o gran escala.
- Incentiva al trabajo en equipo y la comunicación con el usuario

Ventajas

Las principales ventajas de esta metodología son:

- Tiene mayor soporte y mantenimiento.
- Flexibilidad pues se lo puede aplicar a proyectos grandes y pequeños.
- Incentiva al trabajo en equipo y a la colaboración.
- Crea una disciplina de análisis de riesgos que ayuda y evoluciona con el proyecto.

Principios Básicos

Los principios básicos de Microsoft Solutions Framework son:

- Potenciar a todos los miembros de un equipo.
- Potenciar las comunicaciones entre el equipo y el cliente.

- Establecer una visión compartida de los valores de negocio del proyecto.
- Asegurar una contabilización clara de las responsabilidades compartidas.
- Mantener ágiles siempre las expectativas de cambios
- Aprender de las experiencias

El modelo de equipo de Microsoft Solutions Framework describe el enfoque en lo que respecta a la estructuración de personas y sus actividades con el fin de asegurarse el éxito del proyecto.

Estructura

La metodología Microsoft Solutions Framework 5.0 se basa en Scrum para su ejecución el cual define la entrega de elementos de software en periodos de tiempo cortos para la evaluación con el cliente.

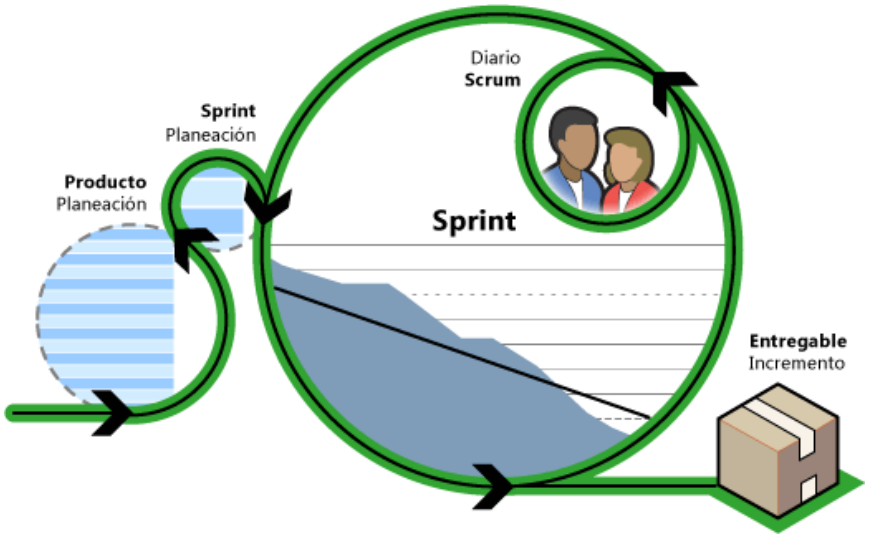


Figura IV. 1 Fases de la Metodología MSF

MSF 5.0 se divide en los siguientes componentes:

- **Ciclo de Planeamiento o Planificación:** En esta fase se obtiene la visión y ámbito del proyecto compartido, comunicado, extendido y alineado con los objetivos del proyecto para solucionar el problema existente. Además se recaba toda la información posible del proceso que se necesita automatizar; gestionar, mejorar los procesos de recepción y manejar la información de las actividades que realiza la Cooperativa, con la implementación del sistema a desarrollarse. Para esto se realiza el diseño conceptual, lógico y físico. Se planifican los tipos de pruebas que se realizarán.

- **Ciclo de Desarrollo:** En este ciclo, junto con el ciclo de Planeación y de Estabilización se obtienen subversiones versiones del producto para proporcionar al cliente una visión e idea clara de la aplicación desarrollada. Cada subciclo es un proceso completo de desarrollo, incluyendo el control de calidad respectivo y la estabilización del release.

- **Ciclo de Implementación:** En esta fase, una vez finalizado el desarrollo y pruebas del sistema, se procede a la implementación y entrega del sistema para lo cual se instala y configura el software requerido.

4.3. Metodología MSF aplicada al desarrollo del sistema "TerraTech"

Las fases utilizadas en el modelo Microsoft Solutions Framework 5.0 serán aplicadas al desarrollo del sistema "TerraTech"

4.3.1. Ciclo de Planeación

La planeación de un sistema de información es la parte fundamental para que un proyecto resulte exitoso. Es necesario recabar toda la información posible del proceso que se

necesita automatizar, gestionar, mejorar los procesos de recepción y manejar la información de las actividades que realiza ERPE con la implementación del sistema a desarrollarse.

Para el desarrollo eficiente del proyecto es importante obtener una visión y ámbito del proyecto compartido, comunicado, extendido y alineado con los objetivos del proyecto para solucionar el problema existente. Además identificar los perfiles de usuario, beneficios, requerimientos funcionales y no funcionales, los riesgos inherentes al proceso y el análisis de factibilidad.

En el diseño conceptual describe los escenarios, las definiciones del negocio, del sistema y de la tecnología aplicada a la solución. El diseño lógico define las herramientas a utilizar, los diagramas de secuencia, diagramas de clases y el diseño de las interfaces. El diseño físico presenta el modelo de los objetos de la base de datos, su respectivo diccionario de datos y la planificación de los tipos de pruebas que se realizarán.

4.3.1.1 Definición del problema

El apoyo de instituciones como APROBICH, ERPE y SUMAKLIFE contribuye al desarrollo de comunidades indígenas promoviendo capacitación para sus habitantes, para así obtener mayor y mejor organización y control sobre sus cultivos; al igual que instituciones privadas como Microsoft que también contribuyen con el adelanto de estos pueblos dotándolas de tecnología.

Los habitantes de las comunidades o pueblos rurales actualmente cuentan con recursos tecnológicos, pero estos no son aprovechados adecuadamente para el adelanto económico y la mejora en sus condiciones de vida, ya que como su principal actividad es la agricultura y sus productos cosechados máximo llegan a venderse en los mercados locales por el hecho de no contar con una herramienta que les ayude a

promover sus productos.

La propuesta va orientada a la implementación de un sistema informático que promueva estos productos a nivel nacional e internacional, además del control local sobre la producción de los mismos.

Es indispensable la existencia de un organismo que controle la producción, venta y comercialización de los productos, la propuesta contribuye gran parte de la solución y está encaminada a los productores, a la forma de ofrecer sus productos y a que de cierta forma los compradores puedan tener conocimiento del origen de los productos.

4.3.1.2 Visión del proyecto

El sistema informático TerraTech ofrecerá la posibilidad de que las personas del campo puedan producir y vender sus productos de una manera controlada y justa, permitiendo que lleguen a mercados locales y extranjeros.

El análisis, diseño y automatización del sistema TerraTech permitirá automatizar el proceso de control, comercialización y venta de los productos, mediante el uso de un sistema informático; permitiendo que compradores puedan conocer aspectos fundamentales sobre dichos productos respondiendo a las siguientes preguntas, ¿Cómo están producidos?, ¿Dónde están siendo producidos?, ¿Por quienes están siendo producidos?, ¿Qué nivel de calidad tienen?

El sistema TerraTech producirá un gran beneficio al permitir que las personas del campo, especialmente jóvenes y mujeres hagan uso de los recursos tecnológicos tanto en hardware como software disponibles para ofrecer sus productos, se crearan oportunidades de trabajo permitiendo el desarrollo personal, familiar y comunitario.

4.3.1.3 Perfiles de Usuario

Administrador ERPE

Este usuario está encargado del registro, modificación y eliminación de las cuentas de inspector, y del sondeo de los informes de inspección emitidos.

Sus funciones son:

- Gestión de Inspectores
- Gestión de informes de inspección
- Administración de la información del usuario

  El tipo de acceso que va a tener es: Escritorio/Web

Inspector

Cada inspector controla y emite informes de la producción de cada una de las parcelas de cultivo.

Sus funciones son:

- Gestión de informes de inspección
- Administración de la información del usuario

El tipo de acceso que tiene es: Web

Productor

Este usuario podrá ingresar la información correspondiente al Diario de Campo, como: datos sobre el cultivo de sus parcelas, rotación de cultivos, tenencia de recursos, etc., así como poder visualizar el monto de sus ingresos y egresos.

Sus funciones son:

- Gestión de producción
- Administración de la información del usuario

El tipo de acceso que va a tener este perfil es: Web

Certificador

Este usuario podrá visualizar los informes de inspección y de acuerdo al análisis de el mismo poder determinar la calidad de dicho producto inspeccionado.

Las funciones de este perfil de usuario son:

- Visualización de la información correspondiente a los informes de inspección
- Determinación de la calidad de un producto cosechado e inspeccionado

El tipo de acceso que va a tener este perfil es: Escritorio/Web

Administrador del Sistema

Este usuario tiene control total sobre el sistema en general.

Entre las funciones de este perfil están:

- Tener el control total de la aplicación
- Información y contactos

El tipo de acceso que tiene este perfil es: Escritorio/Web

Invitado

Este usuario solo tendrá acceso a ciertas funciones no muy relevantes desde la web.

Las funciones de este perfil son:

- Visualizar la información de ERPE y ver los contactos que maneja esta institución

El tipo de acceso que manejará este perfil es: Web

4.3.1.4 Ámbito del Proyecto

Controlar de mejor forma los procesos de: Registro de Producción, Inspección, Comercialización y Venta de productos orgánicos.

Este sistema informático permitirá facilitar la comercialización de los productos orgánicos en los mercados locales e internacionales.

4.3.1.5 Objetivos del proyecto

Los objetivos principales del sistema “TerraTech” son los siguientes:

Objetivos del negocio

- Limitar el acceso sólo al personal autorizado de acuerdo a su perfil.
- Automatizar las actividades que se realizan en las Escuelas Radiofónicas del Ecuador (ERPE)
- Generar diferentes reportes que ayude a la toma de decisiones.

Objetivos del diseño

- Proteger contra el acceso de intrusos o personal no autorizado mediante el uso de métodos de autenticación y a través de la arquitectura.
- Proporcionar una interfaz de usuario amigable fácil de manejar.

4.3.1.6 Requerimientos funcionales del sistema

Los requerimientos tienen como propósito definir las especificaciones funcionales del sistema para la implementación de una aplicación informática que permita gestionar los diferentes procesos de las Escuelas Radiofónicas del Ecuador (ERPE)

Alcance

Análisis, diseño, desarrollo e implementación del sistema para el control de producción en las Escuelas Radiofónicas del Ecuador (ERPE)

El Sistema TerraTech se centrara en los principales procesos, como:

- Gestión de datos de los diferentes Usuarios del Sistema (permitiendo habilitar solamente determinadas funciones para cada tipo de usuario)
- Gestión de la información relacionada al Cultivo, Inspección, Certificación, Comercialización
- Venta de Productos Orgánicos (permitiéndole al productor gestionar la información del Diario de Campo, al inspector gestionar el Informe de Inspección, al Certificador verificar y certificar la calidad de los productos, entre otros);
- Gestión de la información de los Clientes nacionales y extranjeros (permitiéndoles a los mismos poder gestionar sus ventas y visualizar información muy detallada sobre el origen y la calidad de los productos).

El desarrollo del sistema TerraTech esta dividido en 4 sprints

Requerimientos funcionales

Para identificar los requerimientos funcionales se utilizará el formato que se muestra en la Tabla IV.I.

Tabla IV. I Formato para identificar los requerimientos funcionales

REQ	-	Dos letras en mayúscula que identifique sprint al que pertenece	-	Número incremental

Fuente: Elaborada por los autores

El desarrollo del presente proyecto informático se manejara mediante el uso de sprints

Los requerimientos para el desarrollo del sistema “TerraTech” son los siguientes:

REQ-SP0-01: Reunión con el coordinador del proyecto

REQ-SP0-02: Pruebas de la base de datos

REQ-SP0-03: Diseño de la base de datos

REQ-SP0-04: Rediseño del módulo de geografía

REQ-SP0-05: Rediseño del módulo de producción

REQ-SP0-06: Rediseño del módulo de marketing

REQ-SP0-07: Adaptación de los módulos de geografía y marketing

REQ-SP0-08: Adaptación del módulo de producción

REQ-SP0-09: Creación del plan de generación de datos

REQ-SP0-10: Pre diseño de la arquitectura

REQ-SP0-11: Creación del proyecto de infraestructura y acceso a datos módulo producción

REQ-SP0-12: Creación del proyecto para el manejo de inversión de control

REQ-SP0-13: Creación del proyecto de infraestructura y acceso a datos módulo geografía

REQ-SP0-14: Creación del proyecto de infraestructura y acceso a datos módulo marketing

REQ-SP0-15: Creación del proyecto de inversión de control para la web asp .net mvc 3

REQ-SP0-16: Creación de los proyectos core de dominio y entities- core

REQ-SP0-17: Creación del proyecto de entidades para el módulo de producción

REQ-SP0-18: Creación del proyecto de entidades para el módulo de geografía

REQ-SP0-19: Creación del proyecto de entidades para el módulo de marketing

REQ-SP0-20: Creación del proyecto de dominio para producción

REQ-SP0-21: Creación del proyecto de aplicación para el módulo de producción

REQ-SP0-22: Creación del proyecto de aplicación para el módulo de marketing

REQ-SP0-23: Creación del proyecto de aplicación para el módulo de geografía

REQ-SP0-24: Creación del proyecto de aplicación para el módulo de inspección

REQ-SP0-25: Creación del proyecto web para el subsistema ERPE – Creación del informe de inspección

REQ-SP1-01: Como inspector, deseo ingresar el informe de inspección

REQ-SP1-02: Como inspector, deseo modificar el informe de inspección

REQ-SP1-03: Como inspector, deseo buscar informes de inspección

REQ-SP1-04: Como inspector, deseo eliminar informes de inspección

REQ-SP1-05: Como inspector, deseo generar los reportes sobre informes de inspección.

REQ-SP2-01: Como administrador general, deseo ingresar los datos de los inspectores.

REQ-SP2-02: Como administrador general, deseo modificar los datos de los inspectores.

REQ-SP2-03: Como administrador general, deseo buscar los datos de los inspectores.

REQ-SP2-04: Como administrador general, deseo eliminar los datos de los inspectores.

REQ-SP2-05: Como administrador general, deseo generar los reportes de los inspectores.

REQ-SP2-06: Como certificador, deseo visualizar los informes de inspección de cada cultivo.

REQ-SP2-07: Como administrador de ERPE, deseo visualizar los reportes estadísticos de producción

REQ-SP2-08: Como administrador de ERPE, deseo visualizar el ingreso económico de cada productor

REQ-SP2-09: Como administrador, inspector, deseo visualizar los informes de producción por parcela.

REQ-SP2-10: Como productor asociado, deseo visualizar mi ingreso económico.

REQ-SP3-01: Como Inspector, administrador de ERPE, deseo visualizar el estado de agroforestería de cada parcela.

REQ-SP3-02: Como administrador, certificador, deseo visualizar los reportes de producción orgánica.

4.3.1.7 Requerimientos no funcionales del sistema

A continuación se muestran los requerimientos no funcionales y su solución en el sistema:

- **Rendimiento**

"La infraestructura de red, así como sus terminales deben cumplir con normas de rendimiento especificadas, para garantizar tiempos de respuesta máximos."

- **Fácil de manejar**

"La interfaz de usuario debe ser intuitiva para que de esta manera los usuarios nuevos del sistema se adapten rápidamente"

Para cumplir este requerimiento se diseñan pantallas amigables para el usuario del sistema.

- **Integridad**

"La información ingresada a la base de datos debe ser la adecuada"

Para cumplir con este requerimiento se realizan procesos de validación en todos los campos que pueden ocasionar problemas de integridad de los datos.

- **Seguridad**

"Garantizar el acceso a la información pertinente de cada perfil de usuario y se debe manejar sistemas de autenticación segura"

Para el cumplimiento de este requerimiento se determinará que perfiles tiene acceso a que tipo de información, es decir que tipo de privilegios serán asignados a cada perfil.

El acceso para los usuarios del sistema es controlado por medio de Nombres de usuario y sus respectivas contraseñas formuladas por el administrador del sistema. Para realizar algún tipo de transacción o utilizar los servicios del sistema se deberá comprobar que se trata de un usuario autorizado.

Si la clave de acceso introducida no corresponde a un usuario autorizado o la clave

no coincide con la almacenada, se dará un mensaje de error.

4.3.1.8 Riesgos

Un proyecto de ingeniería en Software implica un proceso de análisis de todos los factores que intervienen en el desarrollo y mantenimiento; es así, que para la elaboración de nuestro proyecto informático orientado a la administración de los procesos de informes de inspección, inspectores, ingresos económicos, hoja de problemas, informes de abonos, generar reportes etc.; con el objetivo de automatizar el servicio y optimizar recursos para brindar un mejor servicio a la comunidad.

El análisis y gestión de riesgo, es el área dedicada al estudio de todos los posibles escenarios que se pueden presentar debido a vulnerabilidades, o amenazas originadas por diferentes circunstancias durante el desarrollo del proyecto.

Para analizar y gestionar los posibles riesgos que se presenten el transcurso del funcionamiento del software es necesario identificar cada uno de los riesgos, su categoría, consecuencias, probabilidad, impacto y exposición anexando una hoja de información de cada riesgo que contiene un plan de reducción, supervisión y gestión para así evitar que se conviertan en un problema.

Identificación del riesgo

La identificación de riesgos consiste en la determinación de elementos de amenazas potenciales mediante la utilización de algún método consistente y estructurado.

Este es, probablemente, el paso más importante entre todos aquellos que componen las actividades de administración de riesgos, ya que sin la correcta determinación de los mismos, no es posible desarrollar e implementar anticipadamente respuestas apropiadas a los problemas que puedan surgir en el proyecto.

La categoría de los riesgos se clasificará en:

- **Riesgo técnico:** amenazan la calidad y la planificación temporal del software que se va a producir. Si un riesgo técnico se convierte en realidad, la implementación puede llegar a ser difícil o imposible. Los riesgos técnicos identifican problemas potenciales de diseño, implementación, de interfaz. verificación y de mantenimiento.

- **Riesgo del Proyecto:** amenazan al plan del proyecto. Es decir, si los riesgos del proyecto se hacen realidad, es probable que la planificación temporal del proyecto se retrase y que los costos aumenten. Los riesgos del proyecto identifican los problemas potenciales de presupuesto, planificación temporal, personal (asignación y organización), recursos, cliente, requisitos y su impacto en un proyecto de software.

- **Riesgos del Negocio:** amenazan la viabilidad del software a construir, los riesgos del negocio a menudo ponen en peligro el proyecto.

Para el identificador de los riesgos se utilizarán la letra R, seguida de un número incremental.

En la Tabla IV. II se muestra la identificación de los riesgos potenciales que han sido determinados para el proyecto, cada uno posee un identificador que lo representa, su descripción, se lo clasifica en la categoría correspondiente y se determina la consecuencia que podría causar si se convierte en realidad.

Por ejemplo para el primer riesgo, el identificador sería R01, su descripción es “Modificación de los requerimientos constantemente por parte de los usuarios”, se encuentra en la categoría de Riesgo del Proyecto ya que amenaza al plan del proyecto y es probable que la planificación temporal

Tabla IV. II Identificación del Riesgo

ID	DESCRPCIÓN DEL RIESGO	CATEGORÍA	CONSECUENCIA
R01	Modificación de los requerimientos constantemente por parte de los usuarios.	R. Proyecto	Retraso en las etapas de desarrollo, incremento en el costo
R02	Mal diseño de la base de datos	R. Técnico	Pérdida de recursos
R03	Mala administración	R. de Negocio R. Proyecto	Retraso de proyecto
R04	Que los técnicos no cumplan con los tiempos establecidos para el desarrollo del proyecto	R. Proyecto	Retraso en las etapas de desarrollo
R05	Que los equipos adquiridos no cumplan con las características planeadas	R. Técnico	Provoquen el bajo desempeño del software
R06	No poder hacer validos los datos	R. Técnico	Mala calidad de proyecto
R07	Se cambie de DBMS (Administrador de base de datos)	R. Técnico	Retraso del proyecto
R08	Demora en la entrega de recursos necesarios para la implementación del software	R. Proyecto	Retraso del Proyecto
R09	Interfaces complejas o fuera de contexto con el sistema	R. Técnico	Dificultad en el uso del sistema por parte del usuario
R10	Cantidades excesivas de datos	R. Técnico	Retraso en el proyecto. Pérdida de recursos
R11	Los desarrolladores no elijan adecuadamente las herramientas de desarrollo	R. Técnico	Software de baja calidad
R12	Malos cálculos.	R. de Negocio R. Proyecto	Retraso de proyecto
R13	Redundancia de datos	R. Técnico	Pérdida de recursos
R14	Diferentes datos al momento de visualizar	R. Proyecto	Mala calidad del software
R15	No se pueda ver reportes por deficiencia de las herramientas de reportes.	R. Técnico	Mala calidad de proyecto
R16	Demoras en la programación	R. Técnico	Retraso del proyecto
R17	El producto no satisface las expectativas del cliente	R. de Negocio	Pérdida de recursos
R18	Datos incoherentes	R. Negocio	Pérdida de Recursos

Fuente: Elaborada por los autores

Tabla de Referencias

Se utilizarán tablas como referencia para poder determinar la exposición al riesgo, las probabilidades y el impacto de ocurrencia, las cuales serán utilizadas para determinar la prioridad de los riesgos descritos en la Tabla IV. II.

Para determinar la probabilidad de ocurrencia del riesgo se utiliza la Tabla IV. III como referencia, para esto se ha dividido en 3 rangos iguales de probabilidad, y se ha asignado su equivalencia cualitativa y un valor relacionado a cada uno.

Por ejemplo si luego del análisis de un riesgo se determina que el la probabilidad de ocurrencia es del 20%, utilizando la tabla de referencia, significa que la probabilidad de ocurrencia de ese riesgo es Baja y se le asignaría el valor de 1.

Tabla IV. III Tabla de referencias para determinar la probabilidad de ocurrencia

RANGO DE PROBABILIDADES	DESCRIPCIÓN	VALOR
1-33%	BAJA	1
34-67%	MEDIA	2
68-99%	ALTA	3

Fuente: Elaborada por los autores

Para determinar el impacto de ocurrencia del riesgo se utiliza la Tabla IV. IV como referencia, en la cual se muestra el nivel de impacto del riesgo, el tiempo de retraso que ocasiona, el impacto técnico que provoca, el porcentaje que se incrementa al costo del proyecto, y un valor relacionado a cada uno.

Por ejemplo si el impacto tiene un valor de 1 equivale a que el impacto de ocurrencia es Bajo, el proyecto se podría retrasar aproximadamente 1 semana, el impacto técnico tendría un Ligero efecto en el desarrollo del proyecto, el costo se incrementaría un porcentaje menor al 1%, lo cual no implicaría un impacto económico importante en el presupuesto del proyecto.

Tabla IV. IV Tabla de referencia para determinar el impacto de ocurrencia

IMPACTO	RETRASO	IMPACTO TÉCNICO	COSTO	VALOR
BAJO	Hasta 1 semana	Ligero efecto en el desarrollo del proyecto	<1%	1
MODERADO	2 semanas	Moderado efecto en el desarrollo del proyecto	<5%	2
ALTO	1 mes	Severo efecto en el desarrollo del proyecto	<10%	3
CRÍTICO	Más de 1 mes	Proyecto no puede ser terminado	>100%	4

Fuente: Elaborada por los autores

Para determinar la exposición al riesgo se utiliza la Tabla IV. V como referencia, en la cual se muestra el valor de exposición al riesgo, su equivalencia cualitativa y el color que lo representa.

Por ejemplo si el valor de exposición al riesgo es de 1 o 2 equivale a que la exposición al riesgo es Baja y se lo identifica con el color Verde.

Tabla IV. V Tabla de referencia para determinar la exposición al riesgo

EXPOSICIÓN AL RIESGO	VALOR	COLOR
BAJA	1 o 2	Verde
MEDIA	3 o 4	Amarillo
ALTA	> 6	Rojo

Fuente: Elaborada por los autores

Para crear la Tabla IV. VI de referencia, la cual representa la exposición al riesgo, se multiplican los valores del impacto y de probabilidad, estos valores determinan la exposición al riesgo y el color que lo representa, en caso de que el color sea rojo equivale a Alta, en caso de que sea amarillo equivale a Media y en caso de que sea verde equivale a Baja.

Por ejemplo si el valor de impacto es de 1 y el valor de probabilidad es de 3, el resultado de multiplicar ambos valores es de 3, por lo que la exposición al riesgo tiene el valor de 3, equivalente a Media y el color que lo identifica es el amarillo.

Tabla IV. VI Tabla de referencia para determinar la exposición al riesgo

	IMPACTO			
PROBABILIDAD	**BAJO = 1**	**MODERADO = 2**	**ALTO = 3**	**CRITICO = 4**
ALTA = 3	3	6	9	12
MEDIA = 2	2	4	6	8
BAJA = 1	1	2	3	4

Fuente: Elaborada por los autores

Determinación de la prioridad al riesgo

En la Tabla IV. VII se muestra la prioridad de los riesgo determinados, utilizando las tablas de referencia mencionadas anteriormente, para lo cual se determinan los valores de probabilidad, el impacto y la exposición del riesgo, con lo cual se priorizan los riesgos. El proceso para obtener estos valores es el siguiente:

- Para determinar la probabilidad se analiza cada uno de los riesgos determinando la probabilidad de ocurrencia, con estas probabilidades se determina el valor equivalente y su equivalencia de probabilidad.
- Para determinar el impacto se analiza el valor equivalente de impacto que puede ocasionar cada uno de los riesgos en caso de ocurrencia, con esto se determina la equivalencia correspondiente.
- Para determinar la exposición al riesgo se utilizan los valores de probabilidad, de impacto, con los cuales se determina el valor correspondiente y su equivalencia de exposición.

- Una vez determinados los valores de exposición al riesgo, a estos se les asigna valores para identificar su prioridad, cuando la exposición es alta se asigna el valor de 1, si es media el valor de 2 y si es baja el valor de 3.

Por ejemplo, para el riesgo R01 “Modificación de los requerimientos constantemente por parte de los usuarios”, luego del análisis se determinó que la probabilidad de ocurrencia es del 80% ya que si el usuario modifica constantemente los requerimientos con cambios importantes, las consecuencias sería el retraso en las etapas de desarrollo del proyecto e incrementos en los costos, utilizando la Tabla IV. III. se obtuvo que el valor es 3 y equivale a Alta.

Al determinar el impacto que ocasionaría al proyecto este riesgo se determina que será de 2, ya que tiene un moderado efecto en el desarrollo del proyecto, utilizando la Tabla IV. IV. se obtuvo que equivale a Moderado y que el costo del proyecto podría incrementarse hasta un 5%.

Utilizando los valores de probabilidad y de impacto se obtiene el valor de la exposición al riesgo, utilizando la Tabla IV. V. se obtuvo que el valor de exposición al riesgo es 6, lo que equivale a Alta, y se lo identificará con el color rojo.

Tabla IV. VII Determinación de la prioridad al riesgo

ID	PROBABILIDAD			IMPACTO		EXPOSICIÓN		PRIORIDAD
	%	VALOR	PROBABILIDAD	VALOR	IMPACTO	VALOR	EXPOSICIÓN	
R01	80	3	Alta	2	Moderado	6	Alta	1
R02	30	1	Baja	2	Moderada	2	Baja	4
R03	50	2	Media	3	Alta	6	Alta	1
R04	45	2	Media	3	Alto	6	Alta	1
R05	20	1	Baja	2	Moderado	2	Baja	4
R06	30	1	Alta	2	Media	2	Baja	4
R07	40	2	Media	3	Alto	6	Alta	1

R08	40	2	Media	2	Moderada	4	Alta	2
R09	25	1	Baja	2	Moderado	2	Baja	4
R10	40	2	Media	3	Alta	6	Alta	1
R11	30	1	Baja	3	Alta	3	Alta	3
R12	50	1	Baja	3	Alta	3	Media	3
R13	20	3	Alta	2	Moderada	6	Alta	1
R14	75	1	Alta	2	Media	2	Baja	4
R15	30	3	Alta	1	Baja	3	Media	3
R16	75	1	Baja	2	Moderada	2	Baja	4
R17	25	1	Baja	3	Alto	3	Alta	3
R18	80	1	Baja	2	Moderada	2	Baja	4

Fuente: Elaborada por los autores

Prioridad

Los riesgos de la Tabla IV.VII se organizan de manera ascendente en base al valor de la prioridad, formando la Tabla IV. VIII.

Los riesgos con valor de exposición 6 tendrán la prioridad de 1 (Alta), los que tengan valor de exposición 3 tendrán prioridad 2 (Media) y los que tengan valor de exposición 2 tendrán prioridad de 3 (Baja).

Se crea además la línea de corte para los riesgos principales de prioridad 1 ya que estos riesgos pueden convertirse en problemas con mayor rapidez, para realizar la planeación y programación de estos riesgos.

Tabla IV. VIII Riegos priorizados

RIESGO	EXPOSICIÓN	PRIORIDAD
R01	6	1
R03	6	1
R04	6	1
R07	6	1
R10	6	1
R13	6	1
R08	4	2
R11	3	3
R12	3	3
R15	3	3
R17	3	4
R02	2	4
R05	2	4
R09	2	4
R14	2	4
R16	2	4
R18	2	4

Línea de corte

Fuente: Elaborada por los autores

Según lo que podemos observar en la Tabla IV. VIII que el desarrollo de la aplicación tiene 7 riesgos de prioridad Alta, 2 riesgos de prioridad Media y 7 riesgos de prioridad Baja.

El análisis de los riesgos antes mencionados, nos permiten determinar que el desarrollo del sistema TerraTech presenta un riego Alto (50%), y que debemos realizar una buena gestión de los riesgos de prioridad Alta, a continuación veremos como se mitiga el impacto de estos riesgos.

Planeación y programación del riesgo

Para mitigar los 7 riesgos de prioridad 1 se utiliza la Hoja de Gestión de Riesgo, en la cual se presenta la información del riesgo, las causas que pueden ocasionar el riesgo y las consecuencias si el riesgo se convierte en problema, las acciones que se realizarán para evitar el riesgo, los responsables de supervisarlos y gestionarlos. La planeación y programación de los riesgos se encuentra en el Anexo 7.

4.3.1.9 Análisis de Factibilidad

La factibilidad es la disponibilidad de los recursos necesarios para llevar a cabo los objetivos señalados. Para esto se detallará la factibilidad técnica, operativa y económica.

Factibilidad Técnica

La Factibilidad técnica consiste en la evaluación de la tecnología que existente, lo que permitirá recolectar información sobre los componentes técnicos con los que cuenta y la posibilidad de hacer uso de los mismos en el desarrollo e implementación del sistema informático propuesto.

Además permite definir, de ser necesario, los requerimientos tecnológicos que deben ser adquiridos para el desarrollo y puesta en marcha del sistema.

Tomando en cuenta la tecnología necesaria para el análisis, diseño e implementación del sistema para las Escuelas Radiofónicas Populares del Ecuador, se evalúa dos puntos de vista: hardware y software, para esto se realiza el diagnóstico para determinar los recursos que posee ERPE

Hardware existente

En las Escuelas Radiofónicas Populares del Ecuador (ERPE), al momento posee 1 estación de trabajo (CPU, Monitor, Teclado, Mouse, Regulador) que se encuentran en Buen estado, además de una impresora láser. Este hardware no es suficiente para la

implementación del sistema TerraTech, en especial referente sobre los requerimientos del CPU.

La Tabla IV.IX muestra el hardware existente en ERPE.

Tabla IV. IX Hardware disponible

Cantidad	Descripción	Estado
1	Estación de trabajo	Bueno
1	Impresora laser	Bueno

Fuente: Elaborada por los autores

Hardware requerido

La Tabla IV.X se muestra el hardware requerido para la implementación del sistema informático “TerraTech”, ya que el existente no permite la implementación del sistema. Lo que se necesita es una computadora que sirva de servidor, otra que sirva para el acceso a los clientes, una impresora, un switch, y cable directo para la implementación de la red física y lógica en ERPE.

Tabla IV. X Hardware Requerido

Cantidad	Descripción	Estado
1	Computadora: DUAL CORE 2Gb de memoria RAM disco duro con 320Gb libres Tarjeta de red 10/100/1000	Bueno
1	Impresoras láser	Bueno
1	Switch	Bueno

4	Cable directo	Bueno
2	Servidor: QUAD CORE 4Gb de memoria RAM disco duro con 500Gb libres Tarjeta de red 10/100/1000	Bueno

Fuente: Elaborada por los autores

Software existente

La Tabla IV.XI muestra el software existente en ERPE, por el momento existe las licencias de Microsoft Windows 7, con el Microsoft Office 2007.

Tabla IV. XI Software existente

Nombre	Descripción	Estado	No. de licencias existentes
Windows Seven	Sistema Operativo	legal	1
Microsoft Office 2007	Herramienta de trabajo	legal	1

Fuente: Elaborada por los autores

Software requerido

La Tabla IV.XII describe el software requerido para la implementación del sistema informático “TerraTech”, dentro de los cuales tenemos SQL Server 2008 R2, Visual Studio Ultimate 2010, Windows 7, Windows Server 2008.

Tabla IV. XII Software requerido

Nombre	Descripción	No. de licencias
SQL Server 2008 R2	Servidor de base de datos	1
Visual Studio Ultimate	Lenguaje de programación orientado a objetos	2
Windows 7	Sistema Operativo	2
Windows Server 2008	Sistema Operativo	1

Fuente: Elaborada por los autores

El recurso humano para el desarrollo del sistema se encuentra integrado por las siguientes personas con sus roles que desempeñarán en el proyecto:

Tabla IV. XIII Roles y funciones

Nombre	Función dentro del proyecto
Ramiro Lugmania	Programador/Analista/Diseñador
Henry Villa	Programador/Analista/Diseñador

Fuente: Elaborada por los autores

La metodología Microsoft Solutions Framework (MSF) en su versión 5 utiliza Scrum, dentro de los cuales tenemos los roles antes mencionado, esta metodología recomienda que el equipo completo de trabajo como mínimo debiera ser de cinco personas, con el objetivo de que cada uno de los miembros desempeñe un único papel en el desarrollo del sistema informático.

A pesar del conocimiento de esta ***best practice*** recomendada por Microsoft, hemos optado por esta metodología ya que los beneficios que nos presentan al momento de desarrollar el sistema informático son muchos. Entre los más importantes podemos destacar los siguientes:

- Gestión regular de las expectativas del cliente
- Resultados anticipados
- Flexibilidad y adaptación
- Mitigación sistemática de los riegos del proyecto
- Productividad y calidad
- Alineamiento continuo entre el cliente y el equipo de desarrollo
- Equipo motivado

Factibilidad Operativa

La Factibilidad operativa nos permite predecir si existe la posibilidad de poner en marcha el sistema para ERPE, de tal manera que se pueda aprovechar los beneficios que el sistema ofrecerá a los usuarios involucrados.

La necesidad de cambiar la manera actual en la que se llevan los procesos en ERPE y de ofrecer resultados con calidad, reduciendo la posibilidad de errores humanos, fue el motivo que inspiró la concepción del sistema TerraTech.

La aceptación del sistema dependerá de cuan amigable y sencillo sea, de cubrir todos los requerimientos exigidos por los usuarios, la satisfacción de sus expectativas y de brindar la información de forma oportuna y confiable.

El recurso humano necesario para el manejo del sistema informático para las Escuelas Radiofónicas del Ecuador, deberá contar con conocimientos básicos de computación.

El sistema informático “TerraTech” para ERPE debe ser intuitivo y presentar una interfaz amigable para que los usuarios puedan, de inmediato, identificarse con el mismo y navegar con facilidad al realizar sus tareas, sin necesidad de hacer uso de conocimientos avanzados o de cualquier requisito técnico para manejar el sistema

Factibilidad Económica

La factibilidad económica permite evaluar los costos para el desarrollo del Sistema TerraTech. Exige el análisis de la disponibilidad de los recursos necesarios para llevar a cabo los objetivos señalados, para esto se realizará un análisis costo-beneficio.

Costos Asociados Al Desarrollo Del Sistema

Dícese de los costos que se va a tener durante el desarrollo del Sistema Informático, estos costos son referentes a las personas que trabajará en el proyecto, y los costos de hardware y software

Costos del Personal

La Tabla muestra los perfiles de las personas que van a desarrollar el sistema informático, mostrándonos el sueldo mensual, los meses trabajados y el total del costo.

Tabla IV. XIV Costo del Personal

Técnico	Costo/mes	Meses	Total
Programador	150	4	600
Analista	100	4	400
Diseñador	100	4	400
Jefe de Proyecto	200	4	800
TOTAL			**2200**

Fuente: Elaborada por los autores

Costos de Hardware

A continuación detallamos el análisis de costo referente a hardware que se utilizará en el desarrollo del sistema ERPE

Tabla IV. XV Costo hardware

Equipo	Costo	Numero	Total
Switch Tricom	60	1	60
Servidor	1000	1	1000
Computador	800	1	800

TOTAL	**1860**

Fuente: Elaborada por los autores

Costos de capacitación

Se refiere al monto de dinero destinado para la capacitación a las personas a usar el sistema TerraTech.

Tabla IV. XVI Costo de capacitación

Técnico	Capacitación	Costo/mes	Meses	Total
Programadores	.Net, SQL Server 2008 R2	400	1	400
TOTAL				**400**

Fuente: Elaborada por los autores

Costos de Suministro:

La tabla siguiente detalla los diferentes suministros que se utilizará durante el desarrollo del sistema TerraTech.

Tabla IV. XVII Costos de Suministros

Suministro	Cantidad	Precio	Total
DVD's	20	0.30	6
Papel/resma	10	6	60
Toner	3	40	120
Cinta	10	15	150
TOTAL			**336**

Fuente: Elaborada por los autores

Costos de Instalación del Sistema:

Describe los costos de instalación del sistema TerraTech en ERPE, detallamos los costos de capacitación a los usuarios del sistema, el costo del personal que vayan a hacer la

instalación, etc.

Costos de capacitación a los Usuarios

En la tabla IV.XVIII se muestra al personal que va a ser capacitado, además del tiempo que tomará esta capacitación así como también del costo.

Tabla IV. XVIII Costos de capacitación

TÉCNICO	COSTO/MES	MESES	TOTAL
Secretaria	80	1	80
TOTAL			**80**

Fuente: Elaborada por los autores

Costos de Personal de Instalación

En la tabla IV.XIX se muestra que detalladamente el personal que colaborará en la instalación y configuración de equipos para el correcto funcionamiento del sistema TerraTech.

Tabla IV. XIX Costo de personal de instalación

TÉCNICO	COSTO/MES	MESES	TOTAL
Técnico	150	1	150
Asistente	50	1	50
TOTAL			**200**

Fuente: Elaborada por los autores

Costos Hardware:

Se utilizarán los mismos equipos (Servidor, equipos de cómputo, infraestructura económica), los costos están detallados en Factibilidad Económica, Costos de Desarrollo.

Costos Software:

Se utilizarán los mismos equipos (Servidor, equipos de cómputo), los costos están detallados en Factibilidad Económica, Costos de Desarrollo.

Costos de Operación:

Estos costos indican el monto de dinero que costará el sistema en ejecución. Se describen los costos del personal de operación y el costo de mantenimiento.

Costos del Personal de Operación:

La tabla IV.XX muestra el personal que será encargado del manejo del sistema TerraTech.

Tabla IV. XX Costo del personal de operación

PERSONAL	COSTO/MES
Secretaria	400
TOTAL	**400**

Fuente: Elaborada por los autores

Costos de Mantenimiento:

En la tabla IV.XXI se muestra los equipos que recibirán mantenimiento y cada que período de tiempo, lo que garantiza la correcta ejecución del sistema TerraTech. El tiempo en el que se debe realzar mantenimiento a los equipos es sólo una recomendación.

Tabla IV. XXI Costos Mantenimiento

EQUIPO	NO.	COSTO/TRIMESTRAL
Computadores	2	120
Servidor	1	90
Impresoras	2	60
TOTAL		**270**

Fuente: Elaborada por los autores

NOTA: A menos que el problema de mantenimiento sea de consideración, los costos aumentan.

Costos de suministros y Materiales

La tabla IV.XXII muestra los diferentes costos referentes a suministros y materiales que va a generar el sistema TerraTech, para el correcto funcionamiento para con ERPE.

Tabla IV. XXII Costo de suministros y materiales en operación

SUMINISTRO	CANTIDAD AL MES	PRECIO	TOTAL
Papel/kardex/resma	6	6	36
Toner	1	20	20
TOTAL			**56**

Fuente: Elaborada por los autores

Como conclusión al presente análisis de factibilidad se determina que el proyecto para el sistema informático TerraTech es factible económicamente, operacionalmente, y técnicamente, ya que ERPE presenta todas las facilidades para la ejecución del mismo.

4.3.1.10 Análisis de Costo Beneficio

Este análisis nos permite determinar que beneficios podemos obtener del desarrollo del sistema, tantos beneficios del sistema, tangibles e intangibles.

Análisis de flujo de efectivo

Es un resumen del flujo de efectivo que se manejará en el desarrollo del sistema TerraTech.

Tabla IV. XXIII Análisis de flujo de efectivo

	MESES			
	1	**2**	**3**	**4**
INGRESOS	8200	800	800	1000
COSTOS				
Compra de Equipos	2600			
Salarios personal operación	400	400	400	400
Capacitación	400			
Suministros	336	336	336	336
Mantenimiento de HW	270			270
Costo del Proyecto				
Personal Técnico	2200			
Equipo de desarrollo	1860			
Costo suministro	56			
Costo Total	8122	736	736	1006
Flujo de Activo	78	64	64	-6

Fuente: Elaborada por los autores

Beneficios del Sistema

- Permitir el fácil manejo de información de cada uno de los informes, inspectores y reportes de producción.
- Obtener los reportes en menor tiempo con datos confiables y actualizados.

- Determinar el personal actualizado que trabaja en la organización
- Determinar la información de los productores asociados de la organización

Beneficios Tangibles

- Reducir errores en el almacenamiento y procesamiento de datos en un 60%
- Menos errores en el ingreso de datos en un 40%
- Mayor rapidez en brindar la información a los usuarios en un 30%
- Mayor Seguridad en un 50%

Beneficios Intangibles:

- Mejorar el ambiente de trabajo porque se disminuye la redundancia de tareas.
- Mejor atención a los usuarios
- Mejor organización
- Mejor presentación de los reportes
- Mejora en la calidad del servicio

4.3.1.11 Diseño Conceptual

El diseño conceptual se considera un análisis de actividades y consisten en la solución de negocio para el usuario.

Se detalla las definiciones del negocio, del sistema y de la tecnología utilizadas para el desarrollo del proyecto.

4.3.1.12 Planificación de los sprints del sistema TerraTech

La planificación del sistema TerraTech ha sido dividida en sprints ya que facilita el desarrollo y cumplimiento de tiempos por parte del equipo de desarrollo.

El detalle de la planificación de cada uno de los sprints del proyecto informático se encuentra en el Anexo 8.

4.3.1.13. Definiciones

Para una mejor compresión de los principales términos utilizados en la realización de este proyecto de investigación se ha determinado un glosario.

Tabla IV. XXIV Definiciones

Término	Categoría	Descripción
Cliente	Concepto	Persona o empresa que esta dado de alta dentro del sistema
Administrador	Concepto	Persona con acceso total al sistema y a la base de datos para realizar actualizaciones.
Sistema	Concepto	Conjunto de elementos interrelacionados e interactuantes en uno o más de los procesos que proporcionan la capacidad de satisfacer una necesidad u objetivo definido
Producto	Concepto	Resultado de un proceso, está relacionado con cuatro categorías: hardware, software, comunicaciones y servicios.
Especificación	Concepto	Documento que establece de una manera completa, precisa, verificable, los requisitos, comportamiento u otras características de un sistema o componente.
Requisito	Concepto	Necesidad o expectativa que se establece de forma explícita o implícita.

Sitio Web	Concepto	Se canaliza a través del URL o identificador único de cada página de contenidos. Este sistema permite a los usuarios el acceso a una gran cantidad de información: leer publicaciones periódicas, buscar referencias en bibliotecas, realizar paseos virtuales por pinacotecas, compras electrónicas o audiciones de conciertos, buscar trabajo y otras muchas funciones
Servidor	Concepto	Computadora conectada a una red que pone sus recursos a disposición del resto de los integrantes de la red. Suele utilizarse para mantener datos centralizados o para gestionar recursos compartidos.
Backup	Concepto	Copia de seguridad, copia de un programa informático, de un disco o de archivo de datos, realizada para archivar su contenido o para proteger archivos valiosos contra su pérdida en caso de que la copia activa se dañe o quede destruida.
Periférico	Concepto	En informática, término utilizado para dispositivos, como unidades de disco, impresoras, módem o joysticks, que están conectados a un ordenador o computadora y son controlados por su microprocesador.
SRS	Acrónimo	Especificación de Requerimientos de Software.
ESPOCH	Acrónimo	Escuela Superior Politécnica de Chimborazo.
DBMS	Acrónimo	Sistema manipulador de base de datos, esta diseñado para manejar solo cierto tipo predeterminado de estructura lógica.
GUI	Acrónimo	En informática, tipo de entorno que permite al usuario elegir comandos, iniciar programas, ver listas de archivos y otras opciones utilizando las representaciones visuales (iconos) y las listas de elementos.

TCP / IP	Acrónimo	Acrónimo de Transmisión Control Protocol/Internet Protocol (protocolo de control de transmisiones/protocolo de Internet), protocolos usados para el control de la transmisión en Internet. Permite que diferentes tipos de ordenadores o computadoras se comuniquen a través de redes heterogéneas.
CI	Acrónimo	Cédula de Identidad
Validar Datos	Caso de Uso	Proceso por el cual el sistema se encarga de verificar que los datos ingresados cumplan con las reglas establecidas
Administrar Sistema	Caso de Uso	Proceso por el cual el sistema actualiza la información de dependencias, usuarios, productos
Emitir reportes	Caso de Uso	Proceso por el cual el sistema muestra los distintos listados (departamentos, usuarios, productos, lista

Fuente: Elaborada por los autores

4.3.1.14. Diseño Lógico

El diseño lógico traduce los escenarios creados en el diseño conceptual en un conjunto de objetos de negocio y sus servicios. El diseño lógico se convierte en parte en la especificación funcional que se usa en el diseño físico. El diseño lógico es independiente de la tecnología.

El diseño lógico refina, organiza y detalla la solución de negocios y define formalmente las reglas y políticas específicas de negocios.

Para esto se detallarán las herramientas que se utilizarán en el proyecto para realizar el sistema “TerraTech” y se utilizarán los diagramas de secuencia, diagramas de clases y diseño de las interfaces de usuario.

4.3.1.15. Selección de herramientas

Para el desarrollo del sistema orientado hacia las Escuelas Radiofónicas Populares del Ecuador (ERPE), se han seleccionado las siguientes:

- **Visual Studio 2010**

C#. Visual Studio 2010, es un IDE (Entorno de Desarrollo Integrado) lanzado por Microsoft con soporte y mejoramiento a múltiples tecnologías, especialmente las basadas en Windows 7 y IIS 7, los proyectos realizados en esta versión podrán incorporar el nuevo marco de trabajo denominado Framework 4.

- **.Net Framework 4.0**

Es un componente de Windows que puede compilar y ejecutar aplicaciones en los entornos web y Windows.

- **SQL Server 2008 R2**

SQL Server es un sistema de gestión de bases de datos (SGDB) multiusuario, seguro, es un producto de Microsoft.

4.3.1.16. Diagrama de Clases de Diseño

El diagrama de clases de diseño describe la estructura del sistema, muestra sus clases, atributos y las relaciones entre ellos, se describe en el Anexo 9 las cuales se encuentran divididas de acuerdo a esquemas.

4.3.1.17. Diseño de Interfaces de Usuario

En el diseño de la interfaz de usuario para el sistema informático “TerraTech” busca que la interacción entre el usuario y el sistema sea amigable, intuitiva además que el sistema le ayude a realizar las tareas y que no requiera mayor conocimiento.

Además se mantiene un estándar de diseño sobre todas las pantallas para que el usuario pueda apreciar la información de manera visualmente ordenada y se logra una consistencia en cuanto al estilo y al vocabulario que se maneja en este negocio.

En el Anexo 10 se muestran los diferentes diseños de interfaces ordenados por cada uno de los requerimientos.

4.3.1.18. Definición de Roles

En la siguiente tabla se muestran los roles que desempeñan los desarrolladores para la creación del sistema.

Tabla IV. XXV Roles de los desarrolladores

Nombre	Rol1	Rol2	Rol3	Rol4	Rol5
Ramiro Lugmania	Administrador Base de Datos	Arquitectura	Diseño de Interfaces de Usuario	Desarrollo	Administrador del producto
Henry Villa	Administrador Base de Datos	Administrador de versionamiento	Diseño de Interfaces de Usuario	Desarrollo	Pruebas

Fuente: Elaborada por los autores

4.3.1.19. Nomenclaturas y estándares

Para la codificación del proyecto informático se va a ocupara la notación CamelCasing, con las siguientes variaciones.

Tabla IV. XXVI Nomenclaturas y estándares

Tipo de elemento	Notación Empleada	Ejemplo
Definiciones de clases, espacios de nombres, Interfaces, tipos enumerados, nombres de ensamblados	Upper Camel Casing	AdministradorUsuarios
Objetos, punteros, parámetros, variables	Lower Camel Casing	administradorUsuarios

Fuente: Elaborada por los autores

4.3.2. Ciclo de Desarrollo

Esta fase, junto con la fase de planeación y de estabilización se debe obtener versiones del productoluego de ejecutar un Sprint para proporcionar al cliente una visión e idea clara de la aplicación desarrollada.

Para esto se detalla los roles de los desarrolladores, la nomenclatura y estándares que se utilizan, el desarrollo de la base de datos y el diseño de la solución en Visual Studio 2010.

4.3.2.1 Desarrollo de la Base de Datos

Los objetivos principales del desarrollo de la base de datos del sistema TerraTech son los siguientes:

- Desarrollar funciones que necesite el sistema TerraTech para el correcto funcionamiento, la programación debe cumplir el estándar Transact SQL, para que la migración a otros gestores de base de datos no sea muy complicada.
- Revisar y mantener la data, en caso de ser necesario, manejar los respaldos de la base de datos.

Para el cumplimiento de estos objetivos, se ha realizado primero el modelo lógico y posteriormente el modelo físico.

Modelo lógico de la base de datos

Para establecer el modelo lógico de la base de datos se han tenido que realizar muchas reuniones con el coordinador del proyecto en ERPE, ya que el significativo número de entidades clasificadas por esquemas han sido cuidadosamente relacionadas con el objetivo de abstraer la solución a implementar en el sistema TerraTech.

Luego del proceso de normalización hasta la cuarta forma normal de Boyce-Codd se ha diseñado el diagrama lógico de la base de datos como se muestra en el Anexo 11, las mismas que se encuentran dividas de acuerdo a esquemas.

Modelo físico de la base de datos

En base al diagrama lógico de la base de datos se genera el diagrama físico con sus respectivos campos, como se muestra en el Anexo 12, en este modelo se forman nuevas tablas de las relaciones existentes de muchos a muchos, de esta forma aumentando el número de tablas.

4.3.2.2 Diseño de la solución en Visual Studio 2010

Para el proceso de diseño de la aplicación se utiliza el IDE de Visual Studio 2010, de acuerdo a la arquitectura adoptada tenemos la estructura del proyecto como lo muestra la figura IV.32

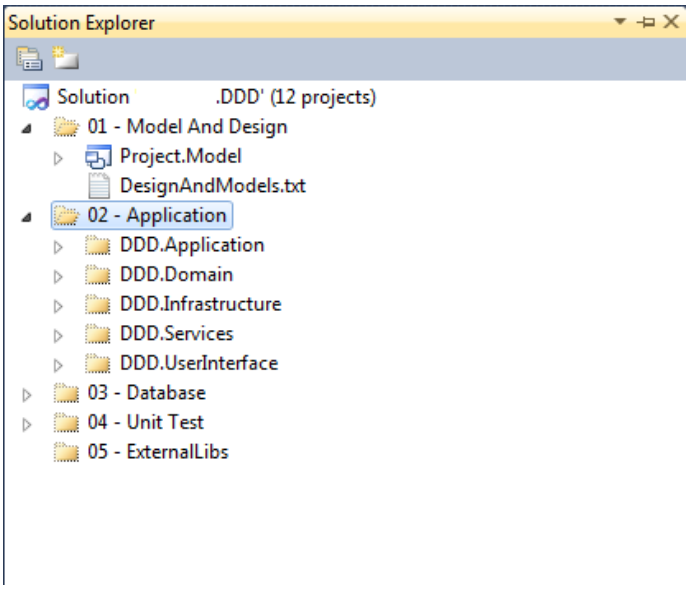


Figura IV. 2 Estructura del proyecto TerraTech

4.3.2.3 Ejecución de la planificación de desarrollo

A continuación se presenta el desarrollo de un caso de uso importante del Sprint 1, en donde se describe todos los diagramas necesarios para su implementación.

SPRINT 1

Caso De Uso: Como inspector, deseo ingresar el informe de inspección

El caso de uso como inspector, deseo ingresar el informe de inspección, se describe en la tabla IV.XXIV

La tabla IV.XXIV detalla los siguientes campos:

- **Identificador de caso de uso:** muestra o denota el nombre del caso de uso que estaremos analizando en este caso es Ingreso Informe Inspección
- **Caso de Uso:** indica el identificador del caso de uso, este es único en el desarrollo del sistema TerraTech. En este caso será CU_Ingreso_Informe_Inspección
- **Actores:** muestra los actores primarios y secundarios que interactúan en el caso de uso. Para este caso de uso sólo tenemos un actor primario Inspector
- **Propósito**: denota la razón por la cuál se hace el caso de uso. En este caso razón por la cual se tiene el caso de uso *Ingreso Informe Inspección* es para detallar la realización de la operación de ingreso de los informa de inspección.
- **Visión General**: muestra cual es la visón que se tiene al desarrollar el caso de uso. En este ejemplo es que el inspector al ingresar los datos del caso de uso pueda almacenarlos y de esa forma llevar un control detallado delos mismos.
- **Tipo**: muestra el tipo o clase de la cual es caso de uso. El presente caso de uso es

del tipo Primario y Esencial, lo que significa que este proceso es muy vital en el desarrollo del sistema TerraTech

- **Curso Típico de Eventos:** indica de que forma se ejecutará los eventos tradicionalmente, mostrando una interacción entre la Acción del Actor y la Respuesta del sistema a dicha acción.
- **Cursos Alternativos:** muestra los eventos que se ejecutarán en el caso de que ocurran diferentes situaciones que no estén dentro del curso típico de eventos.

Tabla IV. XXVII Como inspector, deseo ingresar el informa de inspección

Identificador de Caso de Uso	Ingreso Informe Inspección
Caso de Uso:	CU_Ingreso_Informe_Inspección.
Actores:	Inspector
Propósito:	Realizar la operación del ingreso de los informes de Inspección
Visión General:	El inspector es la persona encargada en ingresar los datos del informe de inspección para almacenarlos y de esta manera llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los informes de inspección
3. Escoge la opción Informe de Inspección.	4. Muestra las opciones de : Nuevo Informe, Modificar Informe, Buscar informe, Eliminar Informe y Generar Reporte
5. Escoge la opción de Nuevo Informe	6. Presentar la interfaz para buscar Productores
7. Ingresar el código o nombre o apellidos del productor	8. Buscar en la base de datos y mostrar
9. Seleccionar el productor	10. Obtener los datos del productor

11. Ingresar (Código, autoconsumo, fecha_de_inspección, año_primera_inspección)	12.IngresarInforme(Código,Inspector,Comunidad,autoconsumo,productor,fecha_de_inspección,año_primera_inspección)
13. Presionar el botón de guardar	14. Los datos se verifican y se mostrará un mensaje de éxito
	13. GuardarInforme()
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el productor se muestra un mensaje de que el productor no existe Línea 7: Si existe algún error en los campos se muestra un aviso al usuario	

Fuente: Elaborada por los autores

En la figura IV.2 se presenta el diagrama de casos de uso que describe le proceso que un inspector debe realizar para el ingreso al sistema de un nuevo informe de inspección.

A continuación se detalla las interacciones que debe realizar el inspector para poder guardar el informe de inspección.

- El inspector ingresa al sistema y se autentica con sus credenciales.
- Dentro del sistema el inspector busca al productor de acuerdo a diferentes criterios de búsqueda. La búsqueda se realiza en base a la cédula o nombre o apellido del productor.
- Procede a la selección del productor al que se le atribuirá el nuevo informe de inspección.
- Se llenan los datos del informe de inspección (código, autoconsumo, fecha_de_inspeccion, año_primera_inspeccion), dichos datos serán validados, para su ingreso.
- Al validar los datos permitirá el ingreso o rechazo del informe de inspección, notificándonos con un mensaje de éxito o fracaso.
- Si todos los datos son correctos se procederá a guardar los datos en la base de datos del sistema TerraTech.

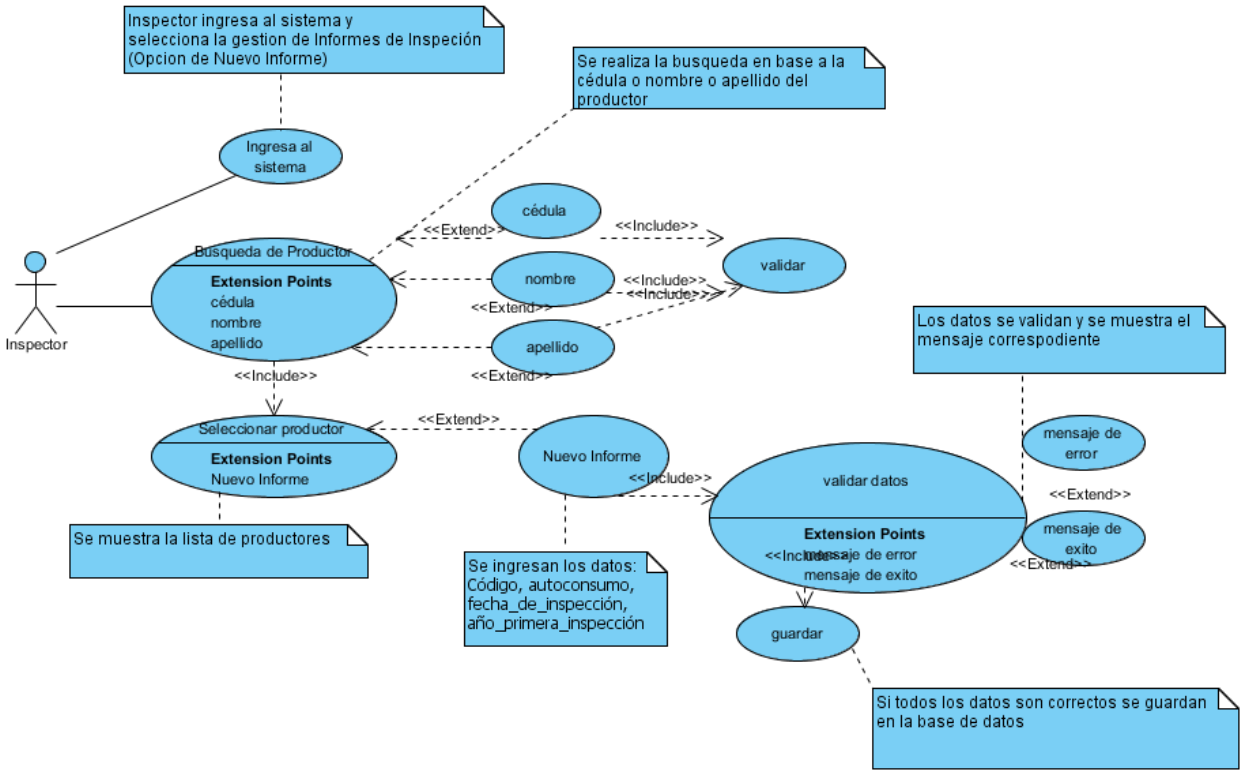


Figura IV. 3 Caso de uso ingreso de muevo informe de inspección

Los demás Casos de Usos del Sprint 1, 2 y 3 se encuentran en el Anexo 13

Diagrama de secuencia

La figura IV.17 muestra el diagrama de secuencia del caso de uso ingreso de un informe de inspección.

A continuación se detalla las diferentes interacciones de este caso de uso:

a. El inspector ingresa al sistema con sus respectivas credenciales.
b. Dentro de la GUI Principal selecciona correspondiente opción de administrar informes de inspección
c. Dentro de la GUI Administración de Informes de Inspección, se selecciona la opción de elegir un nuevo informe.
d. Dentro de la GUI Nuevo Informe se busca al productor al cual será atribuido el

nuevo informe de inspección.

e. En la GUI Búsqueda productor nosotros podemos realizar la búsqueda del productor por diferentes criterios tales como cédula, nombre, apellido.

f. Si en el caso de que existiera los datos de ese productor se seleccionaría a dicho inspector y se ingresaría los datos en la GUI Nuevo Informe de Inspección y se procedería a guarda la información el la base de datos.

g. Caso contrario no se hace nada y sale de la opción.

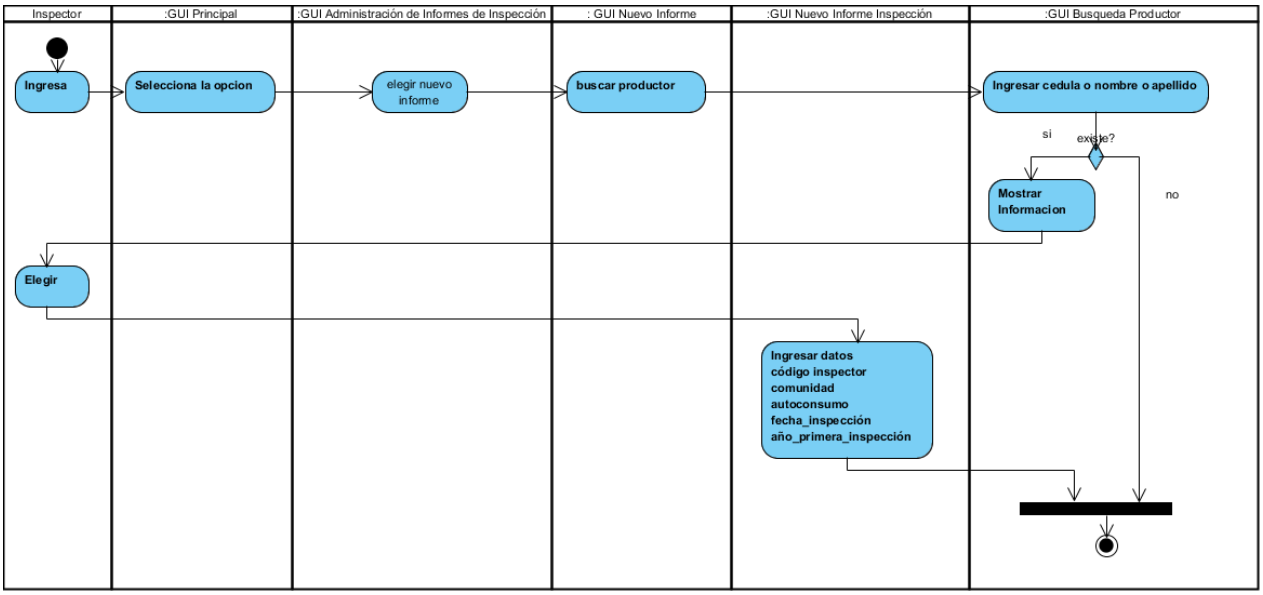


Figura IV. 4 Diagrama de secuencia del caso de uso Ingreso de un informe de inspección

Los demás diagramas de secuencia se encuentran en el Anexo 14

4.3.3 Ciclo de Implementación

Una vez finalizado el desarrollo y pruebas del sistema se procede a la implementación y entrega del sistema “TerraTech” en los equipos que dispone ERPE, para lo cual se realiza la instalación y configuración del software requerido en el equipo servidor y en los equipos clientes.

Los equipos se encuentran dentro de la red Fast-Ethernet utilizando la topología estrella.

Para facilitar la utilización e implementación del sistema “TerraTech” se encontrará disponible:

- Manual de Técnico: en cual se encuentra los procesos de instalación y configuración de los requisitos previos del sistema TerraTech. Este manual se encuentra en el Anexo 15
- Manual de Usuario en el cual se encuentran las instrucciones del funcionamiento del Sistema “TerraTech”. Este manual se encuentra en el Anexo 16.

CONCLUSIONES

1. Con los parámetros de comparación determinados en este estudio se logró seleccionar el mejor framework de inyección de dependencias entre Unity y Spring.Net que proporcionan una amplia gama de beneficios y prestaciones al desarrollar aplicaciones desacopladas, ya que incorporan las mejores características de entre todos los frameworks de inyección de dependencias.
2. Este análisis comparativo permitió determinar que el framework de inyección de dependencias Microsoft Unity 2.0 es el más adecuado para el desarrollo del sistemas con el porcentaje final de 85.71% equivalente a Excelente y 42.86% para Spring.Net equivalente a Muy Bueno, en la cual el framework de inyección de dependencias Unity se muestra superior a Spring.NET.
3. Para el desarrollo de proyectos con grupos pequeños la metodología Microsoft Solutions Framework 5.0 puede ser acoplada para los procesos de análisis, diseño y construcción de sistemas involucrando el uso de múltiples roles para los miembros del equipo.
4. La adaptación de un framework de inyección de dependencias a un proyecto demanda de un cambio en el flujo relacionado a la resolución de dependencias, por lo que es necesario conocer la arquitectura de la aplicación y el modelo de Inversión del control para su correcta implementación.
5. El modelo de programación orientado al dominio distribuido facilita el acoplamiento de la inyección de dependencias sobre el núcleo del sistema permitiendo la fácil integración de las tecnologías específicas.
6. Los ensamblados inyectados por los frameworks de DI deben contener firmas con el objetivo de evitar suplantación de identidad al instante de la instanciación de sus tipos.

RECOMENDACIONES

1. Se recomienda que el personal de las Escuelas Populares Radiofónicas del Ecuador (ERPE) utilice la documentación del sistema como referencia para el uso y mantenimiento del sistema.
2. Se recomienda investigar sobre los mecanismos de extensibilidad de cada uno de los frameworks.
3. Analizar el comportamiento de los frameworks de inyección de dependencias en sistemas altamente transacciones.
4. Se recomienda analizar la interoperabilidad en la inyección de dependencias y el soporte de cada uno de los frameworks.
5. Que se integre mecanismos de autenticación federados para el manejo de seguridad en las múltiples aplicaciones de ERPE.
6. Se recomienda la elaboración de una metodología formal basada en procesos ágiles para el manejo de procesos de desarrollo de software en equipos pequeños.
7. Que el resultado de la tesis “ANALISIS COMPARATIVO ENTRE LOS FRAMEWORKS DE INYECCIÓN DE DEPENDENCIAS UNITY 2.0 Y SPRING.NET. CASO PRÁCTICO: ERPE” sirve como base de estudios posteriores para la extensibilidad y adaptación de soluciones sobre el núcleo del sistema.

RESUMEN

Estudio comparativo entre los Frameworks de Inyección de Dependencias Unity 2.0 y Spring.Net Caso Práctico: Escuelas Radiofónicas Populares del Ecuador ERPE.

Para el estudio comparativo se utilizó el método inductivo para obtener los resultados de los parámetros de acuerdo a las pruebas realizadas en cada una de sus variables además se aplicó estadística descriptiva para la demostración de la hipótesis y determinar el mejor framework de inyección de dependencia.

Las herramientas utilizadas para el estudio y desarrollo fueron los Framework de Inyección de Dependencias Unity 2.0 y Spring.Net, Visual Studio 2010, para la generación de reportes Reporting Service y como motor base de datos SQL Server 2008 R2 y para el despliegue de la aplicación se utilizó el servidor de aplicaciones Internet Information Services.

De acuerdo a los resultados obtenidos en el estudio comparativo en base a los parámetros de comparación: Rendimiento, Complejidad, Requerimientos de Hardware y Software y Seguridad, se determinó un porcentaje final de 85.71% para Unity equivalente a Excelente y 42.86% para Spring.Net equivalente a Muy Bueno.

Se concluye, que el Framework de Inyección de Dependencias Unity es el más adecuado para la realización de sistemas desacoplados, ya que es un contenedor ligero, rápido, fácil de configurar, extensible y existe mayor cantidad de información en la web. Para el desarrollo del Sistema “TerraTech” se aplicó la metodología Microsoft Solution Framework (MSF).

Se recomienda a ERPE el uso del Sistema “TerraTech” ya que ayuda a automatizar los procesos de una manera fiable y eficiente.

GLOSARIO DE TÉRMINOS

Patrones de Diseño: son el esqueleto de las soluciones a problemas comunes en el desarrollo de software

GoF: Gang of Four compuesto por Erich Gamma, Richard Helm, Ralph Johnson y John Vlisides, en el que se recogían 23 patrones de diseño comunes.

Framework: es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.

Programación Monolítica: es un estilo de programación que se ocupaba antes de la aparición de los microprocesadores

Componentes Stubs y mocks: son objetos que simulan el comportamiento de un componente real pero pueden ser configurados para simular respuestas en escenarios de testing.

Principio de Hollywood: Es un paradigma útil que ayuda en el desarrollo de código con alta cohesión y bajo acoplamiento que es más fácil de depurar, mantener y probar.

POJO: Plain Old Java Object

POCO: Plain Old C# Object

BIBLIOGRAFÍA

1. **URQUIZO, A.,** Como Realizar la Tesis o una Investigación., Riobamba – Ecuador., Editorial “Gráficas Riobamba”., 2005., Pp. 1-148

2. **DE LA TORRE, C.**, Guía de Arquitectura N-Capas orientada al Dominio con .NET 4.0, Barcelona-España., Editorial “Krasis Press”., 2010., Pp. 1-534

3. **GOLDSHTEIN, S.,** PRO .NET Performance., New York -Estados Unidos., Editorial “Apress”., 2012., Pp. 1-333

BIBLIOGRAFÍA DE INTERNET

4. **ACOPLAMIENTO INFORMÁTICO**

   http://es.wikipedia.org/wiki/Acoplamiento_inform%C3%A1tico

   2011-03-15

5. **ARQUITECTURA DDD**

   http://microsoftnlayerapp.codeplex.com/

   2012-03-28

6. ARQUITECTURA MONOLÍTICA

http://es.sandramarramirez.wikia.com/

2011-08-15

7. INYECCIÓN DE DEPENDENCIAS

http://www.devx.com/dotnet/Article/34066/0/page/2

2011-03-22

http://www.devx.com/dotnet/Article/34066/0/page/3

2011-03-22

http://msdn.microsoft.com/en-us/library/ff921152.aspx

2012-10-04

8. INVERSIÓN DE CONTROL

http://glnconsultora.com/blog/?tag=inversion-de-control

2012-07-12

http://msdn.microsoft.com/en-us/library/ff921087.aspx

2011-01-24

9. FAKE OBJECTS, MOCKS, STUBS

http://www.amiralles.com.ar/Fake-mocks-stubs.htm

2012-06-06

10. MICROSOFT SQL SERVER

http://es.wikipedia.org/wiki/Microsoft_SQL_Server

2012-10-15

http://www.microsoft.com/sqlserver/en/us/default.aspx

2012-09-10

11. MICROSOFT VISUAL STUDIO

http://es.wikipedia.org/wiki/Microsoft_Visual_Studio

2011-11-07

12. MICROSOFT VISUAL STUDIO 2010

http://msdn.microsoft.com/es-es/library/52f3sw5c

2011-08-26

13. MODELO VISTA CONTROLADOR

http://es.wikipedia.org/wiki/Modelo_Vista_Controlador

2012-10-26

14. MOTOR DE VISTAS RAZOR

http://www.asp.net/mvc/mvc3

2011-05-03

15. PATRÓN DE DISEÑO

http://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o

2012-10-25

16. PRINCIPIO DE HOLLYWOOD

http://en.wikipedia.org/wiki/Hollywood_principle

2012-02-27

17. SERVICE LOCATOR

http://msdn.microsoft.com/en-us/library/ff921142.aspx

2011-10-04

18. SPRING .NET

http://www.springframework.net/

2011-01-25

19. UNITY

http://unity.codeplex.com/

2011-08-13

ANEXOS

Anexo 1. Cálculos para determinar el tamaño de la muestra para las pruebas

Para determinar el tamaño de la muestra para las pruebas de tiempos de respuesta y uso de recursos se utilizó la siguiente fórmula estadística cuando no se conoce la población universal

$$\boldsymbol{n = \frac{Z_a^2 \times p \times q}{d^2}}$$

Donde:

$\boldsymbol{n}$: Tamaño de la muestra.

$\boldsymbol{Z_a^2}$: Es el nivel de confianza elegido, determinado por el valor de α. Para una confianza del 95% (α= 0,05), este valor es de 1,96 según la tabla:

$\propto$	0.10	**0.05**	0.01	0.001
$\boldsymbol{Z_\propto}$	1.645	**1.960**	2.576	3.291

Tabla 1Valores de Z mas usados según el valor de $\propto$

$\boldsymbol{p}$: Proporción esperada, para el caso del estudio se considera una proporción del 50% para reducir el factor de error (0.5).

$\boldsymbol{q}$:Proporcion 1-p

$\boldsymbol{d}$: Margen de error considerado para la prueba, considerado el 5%.

$$\boldsymbol{n} = \frac{1.96^2 \times 0.5 \times 0.5}{0.05^2} = 384.16$$

Aproximando al inmediato superior la muestra es de 385.

Anexo 2: Medición del tiempo de creación de objetos

Para la medición del parámetro se empleo el procedimiento:

1. Ejecución del comando **iisreset** con el objetivo de detener los procesos levantados por el servicio de Internet Information Services
2. Petición de la aplicación prototipo a través de un explorador web.
3. De regreso al proceso 1.

La aplicación prototipo levanta un conjunto de 5 interfaces correspondientes a 5 librerías de implementación las cuales son inyectadas a través del contenedor de objetos, para la medición se aplica un medidor de ejecución que escribe el resultado del tiempo de creación de los objetos sobre un archivo plano.

El resultado de la medición se presenta en milisegundos y se resume en la siguiente tabla:

	Frecuencia	Sumatoria de valores	Promedio
Microsoft Unity	385	22096	57.3922
Spring.net	385	176719	459.0104

Calculo de errores para la prueba

Para determinar el error relativo de las pruebas de medición se empleará la formula:

$$\boldsymbol{\varepsilon_r} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{n} \; y \; \boldsymbol{\varepsilon_p} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{\bar{x} \times n}$$

Donde:

$\boldsymbol{\varepsilon_r}$: Es el error relativo de la prueba

$\boldsymbol{\varepsilon_p}$: Es el porcentaje de error relativo de la prueba

$\sum_{i=1}^{n}(x_i - \bar{x})$: Es la sumatoria de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{x}$: Es el valor promedio de la prueba.

$\boldsymbol{n}$: Es el tamaño de la muestra.

Para la prueba con Microsoft Unity el error relativo y su porcentaje es:

$$\varepsilon_{pu} = \frac{396.5142}{57.3922 \times 385} \times 100\% = 1.7945\%$$

$$\varepsilon_{ru} = \frac{396.5142}{385} = 1.0299\ ms$$

Para la prueba con Spring.Net el error relativo y su porcentaje es:

$$\varepsilon_{ps} = \frac{396.5142}{459.0104 \times 385} \times 100\% = 0.2244\%$$

$$\varepsilon_{rs} = \frac{396.5142}{385} = 1.3765\ ms$$

Así el resultado de la medición es:

	Frecuencia	Sumatoria de valores	Promedio	Error Relativo	Porcentaje de error
Microsoft Unity	385	22096	57.3922	±1.0299	1.7945
Spring.net	385	176719	459.0104	±1.3765	0.2244

Calculo de la desviación

Con el uso de la formula:

$$\boldsymbol{S} = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}{n-1}}$$

En donde:

$\boldsymbol{S}$: Desviación estándar

$\sum_{i=1}^{n} x_i^2$: Es la sumatoria cuadrada de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{\boldsymbol{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{n}$: Es el tamaño de la muestra.

El valor de la desviación estándar para Spring.net

$$\boldsymbol{S_s} = 5.6124\ ms$$

El valor de la desviación estándar para Unity:

$$\boldsymbol{S_u} = 2.3025\ \ ms$$

Cálculo del Intervalo de confianza

Con el uso de la fórmula:

$$\boldsymbol{IC} = \ \bar{x} \pm Z(\frac{S}{\sqrt{n-1}})$$

Donde:

$\boldsymbol{IC}$: Intervalo de Confianza

$\bar{\boldsymbol{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{Z}$: Es el valor de confianza de la prueba.

$\boldsymbol{S}$: Es la desviación estándar.

$\boldsymbol{n}$: Es el tamaño de la muestra.

$\frac{S}{\sqrt{n-1}}$: Equivalente al valor de $S_{\bar{x}}$ (desviación de la media)

Los intervalos de confianza para spring son:

$$\boldsymbol{IC_s} = \bar{x}_s \pm Z\left(\frac{S_s}{\sqrt{n-1}}\right) = 459.0104 \pm 1.96\left(\frac{5.6124}{\sqrt{385-1}}\right)$$

$$\boldsymbol{IC_s} = 458.4490 - 459.5718$$

$$\boldsymbol{S_{\bar{x}_s}} = \frac{5.6124}{\sqrt{385-1}} = 0.2864$$

Los intervalos de confianza para unity son:

$$\boldsymbol{IC_u} = \bar{x}_u \pm Z\left(\frac{S_u}{\sqrt{n-1}}\right) = 57.3922 \pm 1.96\left(\frac{2.3025}{\sqrt{385-1}}\right)$$

$$\boldsymbol{IC_u} = 57.1619 - 57.6225$$

$$\boldsymbol{S_{\bar{x}_u}} = \frac{2.3025}{\sqrt{385-1}} = 0.1175$$

Así el resultado de la prueba en milisegundos es:

	Frecuencia	Sumatoria de valores	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC
Microsoft Unity	385	22096	57.3922	±1.0299	1.7945	0.1175	$57.1619 - 57.6225$
Spring.net	385	176719	459.0104	±1.3765	0.2244	0.2864	$458.4490 - 459.5718$

Anexo 3: Medición del tiempo de obtención de objetos

Para la medición del parámetro se empleo el procedimiento:

1. Invocación a través de un cliente web a la aplicación prototipo
2. Análisis del resultado de la medición ubicado en logs creados para el proceso.

La medición se realiza en Ticks de la clase Stopwatch, equivalentes a una unidad de ejecución basado en el mecanismo de temporización del sistema, con el objetivo de

reducir el porcentaje de error. Para la transformación de Ticks a milisegundos se divide el número de ticks para la frecuencia del procesador, obteniendo el resultado en segundos.

El resultado de la medición se presenta en milisegundos y se resume en la siguiente tabla:

	Frecuencia	Sumatoria de valores	Promedio(Ticks)	Frecuencia Procesador(Ticks/s)	Tiempo(ms)
Microsoft Unity	385	82496	214.2753	1948698	0.1100
Spring.net	385	235603	611.9558	1948698	0.3140

Cálculo de errores para la prueba

Para determinar el error relativo de las pruebas de medición se empleará la formula:

$$\boldsymbol{\varepsilon_r} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{n} \; y \; \boldsymbol{\varepsilon_p} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{\bar{x} \times n}$$

Donde:

$\boldsymbol{\varepsilon_r}$: Es el error relativo de la prueba

$\boldsymbol{\varepsilon_p}$: Es el porcentaje de error relativo de la prueba

$\sum_{i=1}^{n}(x_i - \bar{x})$: Es la sumatoria de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{x}$: Es el valor promedio de la prueba.

$\boldsymbol{n}$: Es el tamaño de la muestra.

Para la prueba con Microsoft Unity el error relativo y su porcentaje es:

$$\boldsymbol{\varepsilon_{pu}} = \frac{1660.4792}{214.2753 \times 385} \times 100\% = 2.0128\%$$

$$\boldsymbol{\varepsilon_{ru}} = \frac{1660.4792}{385} = 4.3129 Ticks \times \frac{1s}{1948698 \; Ticks} \times \frac{1000ms}{1s} = 2.2132 \times 10^{-03} \; ms$$

Para la prueba con Spring.Net el error relativo y su porcentaje es:

$$\boldsymbol{\varepsilon_{ps}} = \frac{4022.2141}{611.9558 \times 385} \times 100\% = 1.7072\%$$

$$\boldsymbol{\varepsilon_{rs}} = \frac{4022.2141}{385} = 10.4473\ Ticks \times \frac{1s}{1948698\ Ticks} \times \frac{1000ms}{1s} = 5.3612 \times 10^{-03}\ ms$$

Calculo de la desviación

Con el uso de la formula:

$$\boldsymbol{S} = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}{n-1}}$$

En donde:

$\boldsymbol{S}$: Desviación estándar

$\sum_{i=1}^{n} x_i^2$: Es la sumatoria cuadrada de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{\boldsymbol{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{n}$: Es el tamaño de la muestra.

El valor de la desviación estándar para Spring.net

$$\boldsymbol{S_s} = 112.4827\ Ticks \times \frac{1s}{1948698\ Ticks} \times \frac{1000ms}{1s} = 5.7722 \times 10^{-2}\ ms$$

El valor de la desviación estándar para Unity:

$$\boldsymbol{S_u} = 114.3091\ Ticks \times \frac{1s}{1948698\ Ticks} \times \frac{1000ms}{1s} = 5.8659 \times 10^{-2}\ ms$$

Cálculo del Intervalo de confianza

Con el uso de la fórmula:

$$\boldsymbol{IC} = \bar{x} \pm Z\left(\frac{S}{\sqrt{n-1}}\right)$$

Donde:

$\boldsymbol{IC}$: Intervalo de Confianza

$\boldsymbol{\bar{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{Z}$: Es el valor de confianza de la prueba.

$\boldsymbol{S}$: Es la desviación estándar.

$\boldsymbol{n}$: Es el tamaño de la muestra.

$\frac{S}{\sqrt{n-1}}$: Equivalente al valor de $S_{\bar{x}}$ (desviación de la media)

Los intervalos de confianza para spring son:

$$\boldsymbol{IC_s} = \bar{x}_s \pm Z\left(\frac{S_s}{\sqrt{n-1}}\right) = 0.3140 \pm 1.96\left(\frac{5.7722 \times 10^{-2}}{\sqrt{385-1}}\right) = 0.3140 \pm 5.7734 \times 10^{-3}$$

$$\boldsymbol{IC_s} = 0.3082 - 0.3198$$

$$\boldsymbol{S_{\bar{x}_s}} = \frac{5.7722 \times 10^{-2}}{\sqrt{385-1}} = 2.9456 \times 10^{-3}$$

Los intervalos de confianza para unity son:

$$\boldsymbol{IC_u} = \bar{x}_u \pm Z\left(\frac{S_u}{\sqrt{n-1}}\right) = 0.1100 \pm 1.96\left(\frac{5.8659 \times 10^{-2}}{\sqrt{385-1}}\right) = 0.1100 \pm 5.8671 \times 10^{-3}$$

$$\boldsymbol{IC_u} = 0.1041 - 0.1159$$

$$\boldsymbol{S_{\bar{x}_u}} = \frac{5.8659 \times 10^{-2}}{\sqrt{385-1}} = 2.9934 \times 10^{-3}$$

Así el resultado de la prueba en milisegundos es:

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC

Microsoft Unity	385	0.1100	2.2132×10^{-3}	2.0128	2.9934×10^{-3}	0.1041-0.1159
Spring.net	385	0.3140	5.3612×10^{-3}	1.7072	2.9456×10^{-3}	0.3082-0.3198

Anexo 4: Medición de memoria usada por el contenedor de objetos

Para la medición del parámetro se aplicó el procedimiento sobre un prototipo de inyección de 5 objetos sobre una aplicación web MVC 3:

1. Invocación a través de un cliente web a la aplicación prototipo
2. Análisis del resultado de la medición ubicado en logs creados para el proceso.

El resultado de la medición se presenta en megabytes y se resume en la siguiente tabla:

	Frecuencia	**Sumatoria de valores**	**Promedio**
Microsoft Unity	385	4903.1856	12.7355
Spring.net	385	4755.0078	12.3507
Aplicación sin IoC	385	4714.3124	12.2450

Cálculo de errores para la prueba

Para determinar el error relativo de las pruebas de medición se empleará la formula:

$$\boldsymbol{\varepsilon_r} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{n} \; y \; \boldsymbol{\varepsilon_p} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{\bar{x} \times n}$$

Donde:

$\boldsymbol{\varepsilon_r}$: Es el error relativo de la prueba

$\boldsymbol{\varepsilon_p}$: Es el porcentaje de error relativo de la prueba

$\sum_{i=1}^{n}(x_i - \bar{x})$: Es la sumatoria de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{x}$: Es el valor promedio de la prueba.

$\boldsymbol{n}$: Es el tamaño de la muestra.

Para la prueba con Microsoft Unity el error relativo y su porcentaje es:

$$\boldsymbol{\varepsilon_{pu}} = \frac{189.0073}{12.7355 \times 385} \times 100\% = 3.8548\%$$

$$\boldsymbol{\varepsilon_{ru}} = \frac{189.0073}{385} = 0.4909 \ \ MB$$

Para la prueba con Spring.Net el error relativo y su porcentaje es:

$$\boldsymbol{\varepsilon_{ps}} = \frac{97.7727}{12.3507 \times 385} \times 100\% = 2.0562\%$$

$$\boldsymbol{\varepsilon_{rs}} = \frac{97.7727}{385} = \ \ 0.2540 \ MB$$

Calculo de errores para la prueba de memoria de la aplicación:

$$\boldsymbol{\varepsilon_{p}} = \frac{47.7278}{12.2450 \times 385} \times 100\% = 1.0124\%$$

$$\boldsymbol{\varepsilon_{r}} = \frac{47.7278}{385} = \ \ 0.1240 \ MB$$

Calculo de la desviación

Con el uso de la formula:

$$\boldsymbol{S} = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - n\overline{x}^2}{n-1}}$$

En donde:

$\boldsymbol{S}$: Desviación estándar

$\sum_{i=1}^{n} x_i^2$: Es la sumatoria cuadrada de la diferencia entre la medida tomada y el valor promedio calculado.

$\overline{\boldsymbol{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{n}$: Es el tamaño de la muestra.

El valor de la desviación estándar para Spring.net

$$\boldsymbol{S_s} = 5.3156 \times 10^{-2}\ \ MB$$

El valor de la desviación estándar para Unity:

$$\boldsymbol{S_u} = 5.2784 \times 10^{-2}\ \ MB$$

El valor de la desviación estándar para la aplicación sin IoC:

$$\boldsymbol{S} = 4.0256 \times 10^{-2}\ \ MB$$

Cálculo del Intervalo de confianza

Con el uso de la fórmula:

$$\boldsymbol{IC} = \ \bar{x} \pm Z(\frac{S}{\sqrt{n-1}})$$

Donde:

$\boldsymbol{IC}$: Intervalo de Confianza

$\bar{\boldsymbol{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{Z}$: Es el valor de confianza de la prueba.

$\boldsymbol{S}$: Es la desviación estándar.

$\boldsymbol{n}$: Es el tamaño de la muestra.

$\frac{S}{\sqrt{n-1}}$: Equivalente al valor de $S_{\bar{x}}$ (desviación de la media)

Los intervalos de confianza para spring son:

$$\boldsymbol{IC_s} = \ \bar{x}_s \pm Z\left(\frac{S_s}{\sqrt{n-1}}\right) = 12.3507 \pm 1.96\left(\frac{5.5136 \times 10^{-2}}{\sqrt{385-1}}\right)$$
$$= 12.3507 \pm 5.5148 \times 10^{-3}$$

$$\boldsymbol{IC_s} = 12.3452 - 12.3562$$

$$\boldsymbol{S_{\bar{x}_S}} = \frac{5.5136 \times 10^{-2}}{\sqrt{385-1}} = 2.8134 \times 10^{-3}$$

Los intervalos de confianza para unity son:

$$\boldsymbol{IC_u} = \bar{x}_u \pm Z\left(\frac{S_u}{\sqrt{n-1}}\right) = 12.7355 \pm 1.96\left(\frac{5.2784 \times 10^{-2}}{\sqrt{385-1}}\right)$$
$$= 12.7355 \pm 5.2795 \times 10^{-3}$$

$$\boldsymbol{IC_u} = 12.7302 - 12.7408$$

$$\boldsymbol{S_{\bar{x}_u}} = \frac{5.2784 \times 10^{-2}}{\sqrt{385-1}} = 2.6936 \times 10^{-3}$$

Los intervalos de confianza para la aplicación sin IoC son:

$$\boldsymbol{IC} = \bar{x}_A \pm Z\left(\frac{S_A}{\sqrt{n-1}}\right) = 12.2450 \pm 1.96\left(\frac{4.0256 \times 10^{-2}}{\sqrt{385-1}}\right) = 12.2450 \pm 4.0264 \times 10^{-3}$$

$$\boldsymbol{IC_A} = 12.2410 - 12.2490$$

$$\boldsymbol{S_{\bar{x}_A}} = \frac{4.0256 \times 10^{-2}}{\sqrt{385-1}} = 2.0543 \times 10^{-3}$$

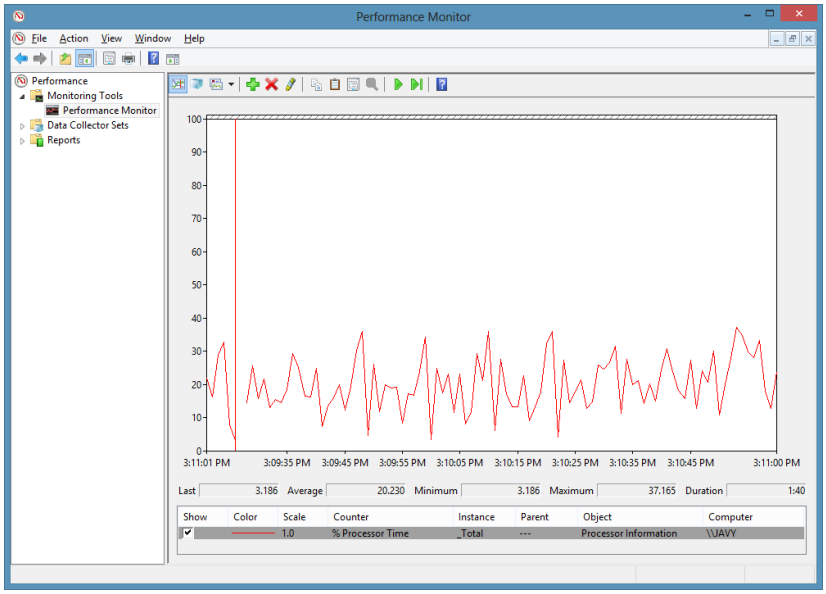
Tomando como base los resultados de medición de la aplicación sin IoC, se tiene los siguientes resultados en Megabytes:

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC	Relación
Microsoft Unity	385	12.7355	0.4909	3.8548	2.6936×10^{-3}	12.7302-12.7408	1.0090:1
Spring.net	385	12.3507	0.2540	2.0562	2.8134×10^{-3}	12.3452-12.3562	1.0086:1
Aplicación sin IoC	385	12.2450	0.1240	1.0124	2.0543×10^{-3}	12.2410-12.2490	1:1

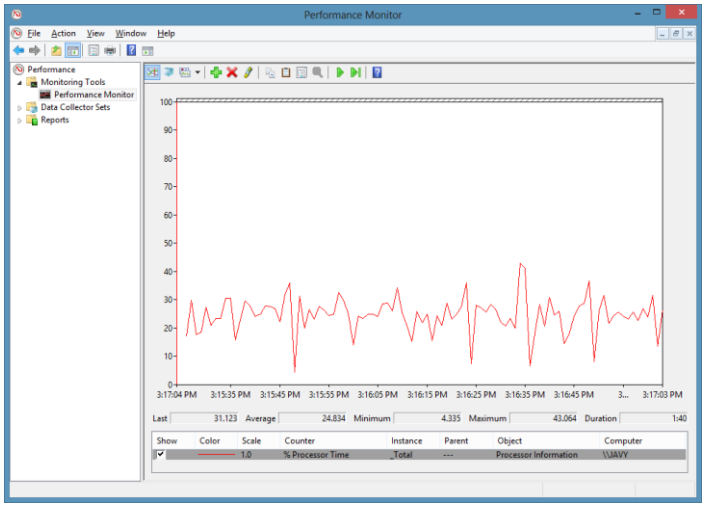
Anexo 5: Medición del uso del procesador

Para la medición de la variable se aplicó el procedimiento:

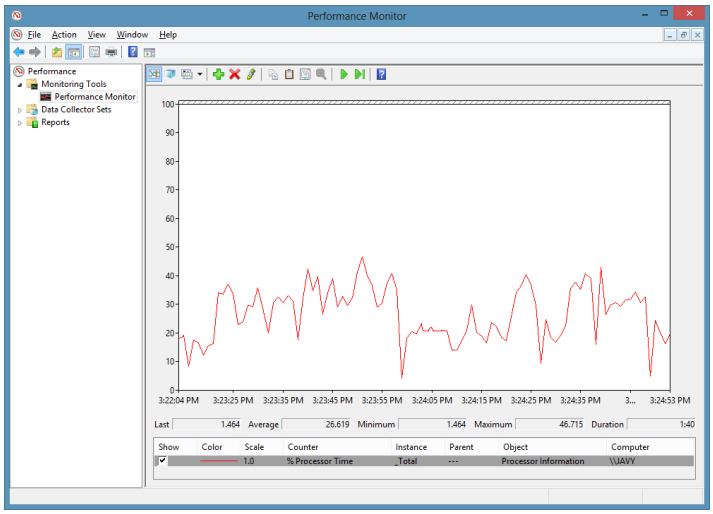
1. Invocación a través de un cliente web a la aplicación prototipo
2. Medición con la herramienta perfmon
3. Análisis del contador de rendimiento %Tiempo de Procesador



Estado del procesador en la medición de carga en una iteración de la aplicación sin IoC



Medición de uso de procesador por Microsoft Unity en una iteración



Medición de uso de procesador por Spring.Net en una iteración

El resultado de la medición se presenta luego de invocar con una frecuencia de 385 es:

	Frecuencia	Sumatoria de valores	Promedio(%)
Microsoft Unity	385	9337.7980	24.2540
Spring.net	385	10390.8810	26.9893
Aplicación sin IoC	385	7759.7140	20.1551

Cálculo de errores para la prueba

Para determinar el error relativo de las pruebas de medición se empleará la formula:

$$\boldsymbol{\varepsilon_r} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{n} \; y \; \boldsymbol{\varepsilon_p} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{\bar{x} \times n}$$

Donde:

$\boldsymbol{\varepsilon_r}$: Es el error relativo de la prueba

$\boldsymbol{\varepsilon_p}$: Es el porcentaje de error relativo de la prueba

$\sum_{i=1}^{n}(x_i - \bar{x})$: Es la sumatoria de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{x}$: Es el valor promedio de la prueba.

n: Es el tamaño de la muestra.

Para la prueba con Microsoft Unity el error relativo y su porcentaje es:

$$\varepsilon_{pu} = \frac{204.0681}{24.2540 \times 385} \times 100\% = 2.1854\%$$

$$\varepsilon_{ru} = \frac{204.0681}{385} = 0.5300$$

Para la prueba con Spring.Net el error relativo y su porcentaje es:

$$\varepsilon_{ps} = \frac{201.2564}{26.9893 \times 385} \times 100\% = 1.9369\%$$

$$\varepsilon_{rs} = \frac{201.2564}{385} = \ 0.5227$$

Calculo de errores para la prueba de memoria de la aplicación:

$$\varepsilon_{p} = \frac{189.2458}{20.1551 \times 385} \times 100\% = 2.4388\%$$

$$\varepsilon_{r} = \frac{189.2458}{385} = \ 0.4815$$

Calculo de la desviación

Con el uso de la formula:

$$S = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}{n-1}}$$

En donde:

S: Desviación estándar

$\sum_{i=1}^{n} x_i^2$: Es la sumatoria cuadrada de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{x}$: Es el valor promedio de la prueba.

n: Es el tamaño de la muestra.

El valor de la desviación estándar para Spring.net

$$S_s = 1.5412$$

El valor de la desviación estándar para Unity:

$$S_u = 2.3302$$

El valor de la desviación estándar para la aplicación sin IoC:

$$S = 1.7020$$

Cálculo del Intervalo de confianza

Con el uso de la fórmula:

$$IC = \bar{x} \pm Z\left(\frac{S}{\sqrt{n-1}}\right)$$

Donde:

IC: Intervalo de Confianza

$\bar{x}$: Es el valor promedio de la prueba.

Z: Es el valor de confianza de la prueba.

S: Es la desviación estándar.

n: Es el tamaño de la muestra.

$\frac{S}{\sqrt{n-1}}$: Equivalente al valor de $S_{\bar{x}}$ (desviación de la media)

Los intervalos de confianza para spring son:

$$\boldsymbol{IC_s} = \bar{x}_s \pm Z\left(\frac{S_s}{\sqrt{n-1}}\right) = 26.9893 \pm 1.96\left(\frac{1.5412}{\sqrt{385-1}}\right)$$

$$\boldsymbol{IC_s} = 26.8351 - 27.1435$$

$$\boldsymbol{S_{\bar{x}_s}} = \frac{1.5412}{\sqrt{385-1}} = 0.0786$$

Los intervalos de confianza para unity son:

$$\boldsymbol{IC_u} = \bar{x}_u \pm Z\left(\frac{S_u}{\sqrt{n-1}}\right) = 24.2540 \pm 1.96\left(\frac{2.3302}{\sqrt{385-1}}\right)$$

$$\boldsymbol{IC_u} = 24.4871 - 24.4871$$

$$\boldsymbol{S_{\bar{x}_u}} = \frac{2.3302}{\sqrt{385-1}} = 0.1189$$

Los intervalos de confianza para la aplicación sin IoC son:

$$\boldsymbol{IC} = \bar{x}_A \pm Z\left(\frac{S_A}{\sqrt{n-1}}\right) = 20.1551 \pm 1.96\left(\frac{1.7020}{\sqrt{385-1}}\right)$$

$$\boldsymbol{IC_A} = 19.9849 - 20.3253$$

$$\boldsymbol{S_{\bar{x}_A}} = \frac{1.7020}{\sqrt{385-1}} = 0.0869$$

Tomando como base los resultados de medición de la aplicación sin IoC, se tiene los siguientes resultados en Megabytes:

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC	Relación
Microsoft Unity	385	24.2540	0.5300	2.1854	2.3302	24.4871-24.4871	1.2034:1
Spring.net	385	26.9893	0.5227	1.9369	1.5412	26.8351-27.1435	1.3391:1

Aplicación sin IoC	385	20.1551	0.4815	2.4388	1.7020	19.9849-20.3253	1:1

Anexo 6: Medición del espacio de almacenamiento usado por el contenedor de objetos para volcar la estructura del contenedor

Para la medición del parámetro se aplicó el procedimiento sobre un prototipo de inyección de 5 objetos sobre una aplicación web MVC 3:

1. Serializacion de objetos
2. Análisis del resultado de la medición ubicado en logs creados para el proceso.

El resultado de la medición se presenta en Kilobytes y se resume en la siguiente tabla:

	Frecuencia	**Sumatoria de valores**	**Promedio**
Microsoft Unity	385	48103.1856	124.9433
Spring.net	385	47955.0258	124.5585

Cálculo de errores para la prueba

Para determinar el error relativo de las pruebas de medición se empleará la formula:

$$\boldsymbol{\varepsilon_r} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{n} \; y \; \boldsymbol{\varepsilon_p} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{\bar{x} \times n}$$

Donde:

$\boldsymbol{\varepsilon_r}$: Es el error relativo de la prueba

$\boldsymbol{\varepsilon_p}$: Es el porcentaje de error relativo de la prueba

$\sum_{i=1}^{n}(x_i - \bar{x})$: Es la sumatoria de la diferencia entre la medida tomada y el valor promedio calculado.

$\bar{x}$: Es el valor promedio de la prueba.

n: Es el tamaño de la muestra.

Para la prueba con Microsoft Unity el error relativo y su porcentaje es:

$$\boldsymbol{\varepsilon_{pu}} = \frac{1084.5822}{124.9433 \times 385} \times 100\% = 2.2547\%$$

$$\boldsymbol{\varepsilon_{ru}} = \frac{1084.5822}{385} = 2.8162 \ \ KB$$

Para la prueba con Spring.Net el error relativo y su porcentaje es:

$$\boldsymbol{\varepsilon_{ps}} = \frac{964.6153}{124.5585 \times 385} \times 100\% = 2.0115\%$$

$$\boldsymbol{\varepsilon_{rs}} = \frac{964.6153}{385} = \ \ 2.5055 \ KB$$

Calculo de la desviación

Con el uso de la formula:

$$\boldsymbol{S} = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - n\overline{x}^2}{n-1}}$$

En donde:

$\boldsymbol{S}$: Desviación estándar

$\sum_{i=1}^{n} x_i^2$: Es la sumatoria cuadrada de la diferencia entre la medida tomada y el valor promedio calculado.

$\overline{\boldsymbol{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{n}$: Es el tamaño de la muestra.

El valor de la desviación estándar para Spring.net

$$\boldsymbol{S_s} = 0.1143 \ \ KB$$

El valor de la desviación estándar para Unity:

$$\boldsymbol{S_u} = 0.2025 \ \ KB$$

Cálculo del Intervalo de confianza

Con el uso de la fórmula:

$$\boldsymbol{IC} = \bar{x} \pm Z(\frac{S}{\sqrt{n-1}})$$

Donde:

$\boldsymbol{IC}$: Intervalo de Confianza

$\boldsymbol{\bar{x}}$: Es el valor promedio de la prueba.

$\boldsymbol{Z}$: Es el valor de confianza de la prueba.

$\boldsymbol{S}$: Es la desviación estándar.

$\boldsymbol{n}$: Es el tamaño de la muestra.

$\frac{S}{\sqrt{n-1}}$: Equivalente al valor de $S_{\bar{x}}$ (desviación de la media)

Los intervalos de confianza para spring son:

$$\boldsymbol{IC_s} = \bar{x}_s \pm Z\left(\frac{S_s}{\sqrt{n-1}}\right) = 124.5585 \pm 1.96\left(\frac{0.1143}{\sqrt{385-1}}\right) = 124.5585 \pm 0.0114$$

$$\boldsymbol{IC_s} = 124.5471 - 124.5699$$

$$\boldsymbol{S_{\bar{x}_s}} = \frac{0.1143}{\sqrt{385-1}} = 0.0058$$

Los intervalos de confianza para unity son:

$$\boldsymbol{IC_u} = \bar{x}_u \pm Z\left(\frac{S_u}{\sqrt{n-1}}\right) = 124.9433 \pm 1.96\left(\frac{0.2025}{\sqrt{385-1}}\right) = 124.9433 \pm 0.0203$$

$$\boldsymbol{IC_u} = 124.9230 - 124.9636$$

$$S_{\bar{x}_u} = \frac{0.2025}{\sqrt{385 - 1}} = 0.0103$$

Los resultados de la medición son los siguientes:

	Frecuencia	Promedio	Error Relativo	Porcentaje de error	$S_{\bar{x}}$	IC
Microsoft Unity	385	124.9433	2.8162	2.2547	0.0103	124.9230-124.9636
Spring.net	385	124.5585	2.5055	2.0115	0.0058	124.5471-124.5699

Anexo 7: Planeación y programación del riesgo

Para mitigar los 3 riesgos de prioridad 1 y el un riesgo de prioridad 2 identificados anteriormente, se utiliza la Hoja de Gestión de Riesgo.

HOJA DE GESTIÓN DE RIESGOS			
ID DEL RIESGO: R01		**FECHA:** 20 de Julio del 2011	
Probabilidad: Alta	**Impacto:** Moderada **Valor:** 2	**Exposición:** Alta **Valor:** 6	**Prioridad:** 1
DESCRIPCIÓN: Modificación de los requerimientos constantemente por parte de los usuarios			
REFINAMIENTO: **Causas:** La entrega del sistema no se cumple en la fecha indicada Falla en la implementación del sistema debido al constante cambio de requerimientos Molestias por parte de los desarrolladores, debido a que no existen ideas claras y precisas sobre el sistema La inconformidad del usuario cuando ya no es posible modificar el requerimiento **Consecuencias:** El costo del sistema se incrementa El sistema no cumpla con todos los requerimientos de los usuarios Demora en el desarrollo del sistema.			

REDUCCIÓN:

Llegar a un acuerdo por escrito entre los usuarios y los desarrolladores para que los requerimientos queden definidos para evitar cambios en lo posterior
Los usuarios y los desarrolladores deben tener una idea clara de lo que se va realizar en el sistema para que cumpla con el objetivo planteado

- Investigar las necesidades que existen antes de implementar el sistema

SUPERVISIÓN:

Comprobar que los programadores y usuarios tengan claro todo el desarrollo del sistema.

- Informar al usuario los avances del sistema cada vez que sea necesario.

GESTIÓN:

Utilizaremos el documento presentado al inicio en el que se especificaba todos los requerimientos
Dar a conocer al usuario las limitaciones del sistema en ese punto del desarrollo y las consecuencias futuras.

ESTADO ACTUAL:

Fase de reducción iniciada □

Fase de Supervisión iniciada □

Gestionado el riesgo: □

RESPONSABLE:

Ramiro Lugmania

Henry Villa

HOJA DE GESTIÓN DE RIESGOS			
ID DEL RIESGO: R03		**FECHA:** 20 de Julio del 2011	
Probabilidad: Media **Valor:** 2	**Impacto:** Alta **Valor:** 3	**Exposición:** Alta **Valor:** 6	**Prioridad:** 1
DESCRIPCIÓN: Mala administración			

REFINAMIENTO:

Causas:

- Falta de seriedad por parte del administrador
- Mala relación entre los desarrolladores y el administrador

Consecuencias:

- El costo del sistema se incrementa
- El sistema no cumpla con todos los requerimientos de los usuarios
- Demora en el desarrollo del sistema.

REDUCCIÓN:

- Pérdida de tiempo y recursos por parte del equipo de trabajo.
- Paralización total del proyecto.

SUPERVISIÓN:

Notarizar el contrato a afecto de que las dos partes se vean obligadas a cumplir sus responsabilidades.

GESTIÓN:

Tener una persona especializada en el ámbito legal que verifique el cumplimiento del contrato.

ESTADO ACTUAL:

Fase de reducción iniciada □

Fase de Supervisión iniciada □

Gestionado el riesgo: □

RESPONSABLE:

Ramiro Lugmania

Henry Villa

HOJA DE GESTIÓN DE RIESGOS			
ID DEL RIESGO: R04		**FECHA:** 20 de Julio del 2011	
Probabilidad: Media **Valor:**2	**Impacto:** Alto **Valor:** 3	**Exposición:** Alta **Valor:** 6	**Prioridad:**1
DESCRIPCIÓN: Que los técnicos no cumplan con los tiempos establecidos para el desarrollo del proyecto			
REFINAMIENTO: **Causas:** □ Falta de organización y asignación de tareas entre los técnicos □ Utilización inadecuada y falta de experiencia en el manejo de las herramientas. **Consecuencias:** □ Desorden en la elaboración de cada fase del proyecto. □ Inconformidad el usuario. □ Incumplimiento del contrato.			
REDUCCIÓN: Utilización de herramientas adecuadas para el desarrollo del sistema. Comunicación y participación entre todos los desarrolladores para que la distribución de las tareas sea la mas adecuada.			
GESTIÓN: □ Incrementar la asignación de tareas para cumplir con la entrega del sistema a tiempo. □ Contratar más personal para que apoye el desarrollo del sistema.			
ESTADO ACTUAL: Fase de reducción iniciada □ Fase de Supervisión iniciada □ Gestionado el riesgo: □			

RESPONSABLE:

Ramiro Lugmania

Henry Villa

HOJA DE GESTIÓN DE RIESGOS			
ID DEL RIESGO: R07		**FECHA:** 20 de Julio del 2011	
Probabilidad: Media **Valor:**2	**Impacto:** Alto **Valor:** 3	**Exposición:** Alta **Valor:** 6	**Prioridad:**1
DESCRIPCIÓN: Se cambie de DBMS (Administrador de base de datos)			

REFINAMIENTO:

Causas:

- Falta de experiencia en los desarrolladores.
- El DBMS elegido no brinde las características requeridas en el sistema.
- Costo de Licencias.
- Desuso del DBMS y sus herramientas.
- Falta de seguridad.

Consecuencias:

- Retrasos en el desarrollo del proyecto.
- Incremento en la seguridad de los datos.
- Mejores prestaciones en el nuevo DBMS.

REDUCCIÓN:

- Seleccionar el DBMS más acorde a las necesidades y recursos de la empresa.

SUPERVISIÓN:

Analizar las características y funciones que brinda el DBMS para verificar si va a cumplir con las necesidades del sistema.

GESTIÓN:

- Contratar personal que posea la suficiente experiencia con el nuevo DBMS

ESTADO ACTUAL:

Fase de reducción iniciada □
Fase de Supervisión iniciada □
Gestionado el riesgo: □

RESPONSABLE:

Ramiro Lugmania

Henry Villa

HOJA DE GESTIÓN DE RIESGOS

ID DEL RIESGO: R10		**FECHA:** 20 de Julio del 2011	
Probabilidad: Media **Valor:**2	**Impacto:** Alto **Valor:** 3	**Exposición:** Alta **Valor:** 6	**Prioridad:**1

DESCRIPCIÓN: Cantidades excesivas de datos

REFINAMIENTO:

Causas:

- Una mala organización del grupo de trabajo.

- Contratar personal que se piensa están preparados para desarrollar Software.

Consecuencias:

- La planificación se la llega a realizar de una forma no propuesta.

- Se llegaría a obtener un producto que no de los resultados deseados.

SUPERVISIÓN: Establecer un cronograma y que la aceptación sea por parte de todos las personas interesadas.
GESTIÓN: Antes de iniciar el desarrollo del proyecto se establecerá un cronograma de actividades, al cual se deben regir los desarrolladores del proyecto.
ESTADO ACTUAL: Fase de reducción iniciada □ Fase
RESPONSABLE: Ramiro Lugmania Henry Villa

HOJA DE GESTIÓN DE RIESGOS			
ID DEL RIESGO: R13		**FECHA:** 20 de Julio del 2011	
Probabilidad: Media **Valor:**2	**Impacto:** Alto **Valor:** 3	**Exposición:** Alta **Valor:** 6	**Prioridad:**1
DESCRIPCIÓN: Redundancia de datos			
REFINAMIENTO: **Causas:** - Mal diseño de la base de datos - Falta de control al ingresar datos **Consecuencias:** - Demora de respuesta en las búsquedas - Almacenamiento de información innecesaria - Información inconsistente			
REDUCCIÓN: - Diseñar adecuadamente la base de datos Diseñar los controles necesarios en las diferentes capas para mantener consistencia en los datos			
SUPERVISIÓN: Realizar pruebas durante las fases de desarrollo para minimizar problemas en la fase de ejecución			
GESTIÓN: Antes de diseñar la base de datos tener en consideración los posibles problemas que se puedan generar al momento de ingresar datos			
ESTADO ACTUAL: Fase de reducción iniciada □ Fase de Supervisión iniciada □ Gestionado el riesgo: □			

RESPONSABLE:
Ramiro Lugmania

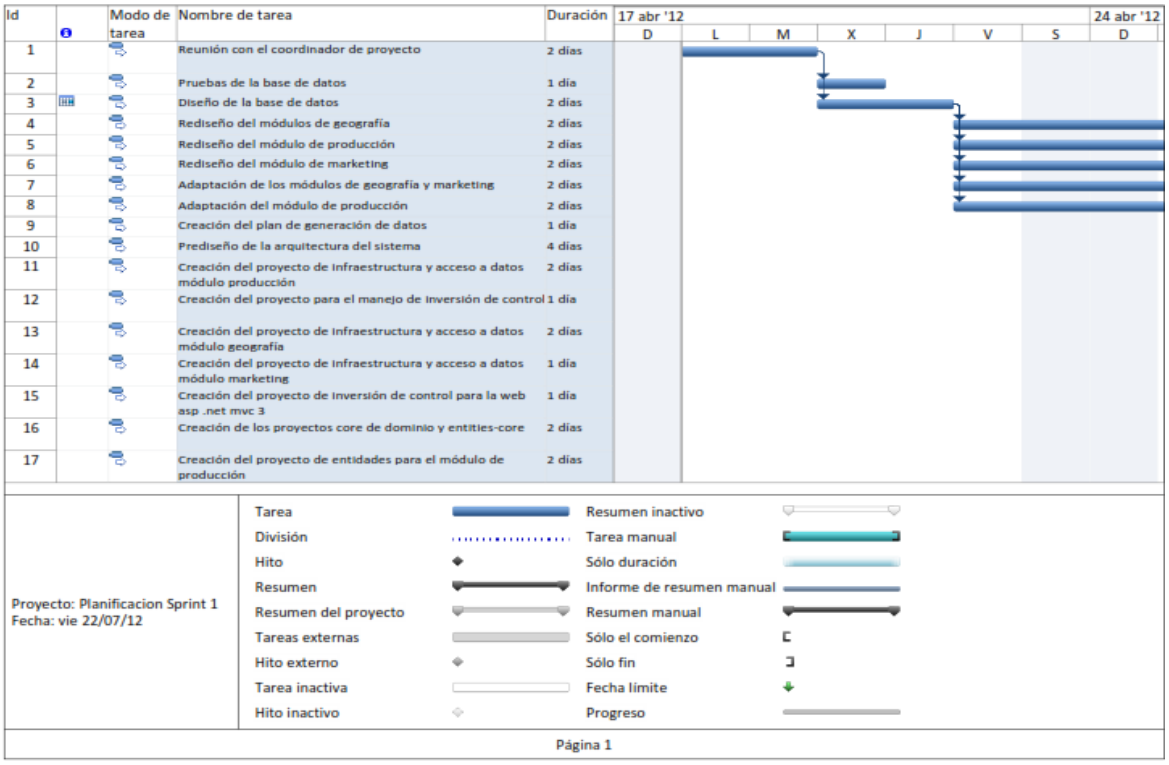
Henry Villa

HOJA DE GESTIÓN DE RIESGOS			
ID DEL RIESGO: R08		**FECHA:** 13 de Enero del 2011	
Probabilidad: MEDIA **Valor:** 2	**Impacto:** 2 **Valor:** MODERADA	**Exposición:** 4 **Valor:** MODERADA	**Prioridad:** 2
DESCRIPCIÓN: Demora en la entrega de recursos necesarios para la implementación del software			
REFINAMIENTO: **Causas:** - Falta de financiamiento por parte de las autoridades - Recursos no disponibles al momento de la implementación **Consecuencias:** - Demora en la realización de las tareas planificadas - Retraso en la entrega del proyecto			
REDUCCIÓN: - Asegurarse de que la entrega de recursos se entreguen en una fecha establecida			
SUPERVISIÓN: El jefe del proyecto debe responsabilizarse de la asignación de recursos para el desarrollo del sistema			

GESTIÓN: □ Buscar alternativas a los recursos que minimicen el retraso del proyecto Adelantar otras tareas que no necesiten esos recursos para que el proyecto siga avanzando mientras tanto
ESTADO ACTUAL: Fase de reducción iniciada □ Fase de Supervisión iniciada □
RESPONSABLE: Ramiro Lugmania Henry Villa

Anexo 8: Planificación de los sprints

SPRINT 0

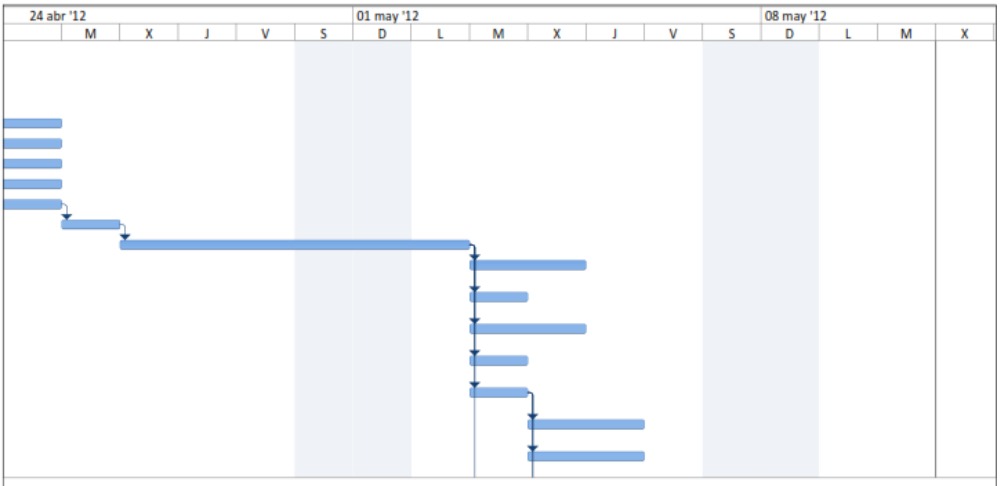


Id	Modo de tarea	Nombre de tarea	Duración
1		Reunión con el coordinador de proyecto	2 días
2		Pruebas de la base de datos	1 día
3		Diseño de la base de datos	2 días
4		Rediseño del módulos de geografía	2 días
5		Rediseño del módulo de producción	2 días
6		Rediseño del módulo de marketing	2 días
7		Adaptación de los módulos de geografía y marketing	2 días
8		Adaptación del módulo de producción	2 días
9		Creación del plan de generación de datos	1 día
10		Prediseño de la arquitectura del sistema	4 días
11		Creación del proyecto de infraestructura y acceso a datos módulo producción	2 días
12		Creación del proyecto para el manejo de inversión de control	1 día
13		Creación del proyecto de infraestructura y acceso a datos módulo geografía	2 días
14		Creación del proyecto de infraestructura y acceso a datos módulo marketing	1 día
15		Creación del proyecto de inversión de control para la web asp .net mvc 3	1 día
16		Creación de los proyectos core de dominio y entities-core	2 días
17		Creación del proyecto de entidades para el módulo de producción	2 días

Id		Modo de tarea	Nombre de tarea	Duración	17 abr '12							24 abr '12
					D	L	M	X	J	V	S	D
18			Creación del proyecto de entidades para el módulo de geografía	2 días								
19			Creación del proyecto de entidades para el módulo de marketing	1 día								
20			Creación del proyecto de dominio para producción	1 día								
21			Creación del proyecto de aplicación para el módulo de producción	1 día								
22			Creación del proyecto de aplicación para el módulo de producción	1 día								
23			Creación del proyecto de aplicación para el módulo de geografía	1 día								
24			Creación del proyecto de aplicación para el módulo de inspección de inspección	1 día								
25			Creación del proyecto web para el subsistema de ERPE-Creación del informe de inspección	2 días								

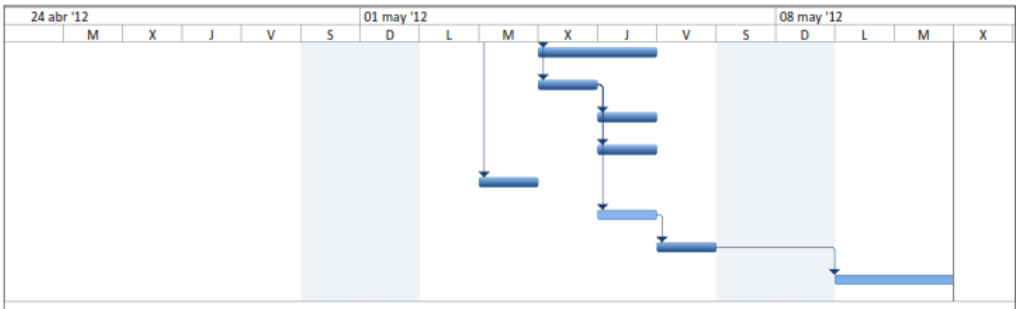
Proyecto: Planificacion Sprint 1
Fecha: vie 22/07/12

Tarea		Resumen inactivo	
División		Tarea manual	
Hito		Sólo duración	
Resumen		Informe de resumen manual	
Resumen del proyecto		Resumen manual	
Tareas externas		Sólo el comienzo	
Hito externo		Sólo fin	
Tarea inactiva		Fecha límite	
Hito inactivo		Progreso	



Proyecto: Planificacion Sprint 1
Fecha: vie 22/07/12

Tarea		Resumen inactivo	
División		Tarea manual	
Hito		Sólo duración	
Resumen		Informe de resumen manual	
Resumen del proyecto		Resumen manual	
Tareas externas		Sólo el comienzo	
Hito externo		Sólo fin	
Tarea inactiva		Fecha límite	
Hito inactivo		Progreso	

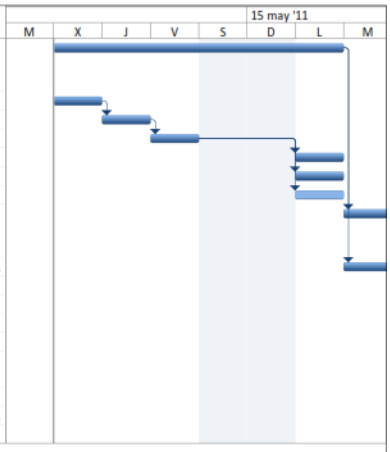


Proyecto: Planificacion Sprint 1
Fecha: vie 22/07/12

Tarea		Resumen inactivo	
División		Tarea manual	
Hito		Sólo duración	
Resumen		Informe de resumen manual	
Resumen del proyecto		Resumen manual	
Tareas externas		Sólo el comienzo	
Hito externo		Sólo fin	
Tarea inactiva		Fecha límite	
Hito inactivo		Progreso	

SPRINT 1

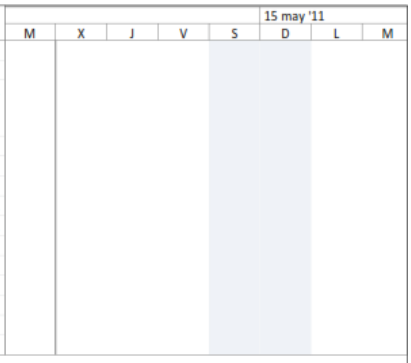
Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		**Como inspector, deseo ingresar el informe de inspección**	4 días	mié 11/05/11	lun 16/05/11
2		Generar las vistas	1 día?	mié 11/05/11	mié 11/05/11
3		Generar los controladores	1 día?	jue 12/05/11	jue 12/05/11
4		Generar los modelos	1 día?	vie 13/05/11	vie 13/05/11
5		Crear servicios	1 día?	lun 16/05/11	lun 16/05/11
6		Pruebas unitarias	1 día?	lun 16/05/11	lun 16/05/11
7		Generar la documentación	1 día?	lun 16/05/11	lun 16/05/11
8		**Como inspector, deseo modificar el informe de inspección**	4 días	mar 17/05/11	vie 20/05/11
9		Generar las vistas	1 día?	mar 17/05/11	mar 17/05/11
10		Generar los controladores	1 día?	mié 18/05/11	mié 18/05/11
11		Generar los modelos	1 día?	jue 19/05/11	jue 19/05/11
12		Pruebas unitarias	1 día?	vie 20/05/11	vie 20/05/11
13		Generar la documentación	1 día?	vie 20/05/11	vie 20/05/11
14		**Como inspector, deseo buscar informes de inspección**	3 días	lun 23/05/11	mié 25/05/11
15		Generar las vistas	1 día?	lun 23/05/11	lun 23/05/11
16		Generar los controladores	1 día?	mar 24/05/11	mar 24/05/11
17		Generar los modelos	1 día?	mié 25/05/11	mié 25/05/11



Proyecto: Planificacion Sprint 2
Fecha: vie 22/07/11

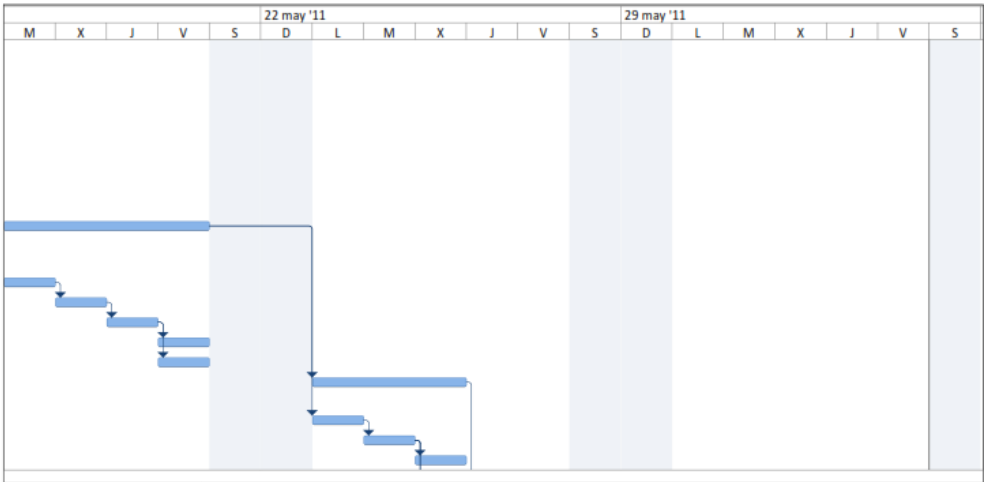
Tarea		Resumen inactivo	
División		Tarea manual	
Hito		Sólo duración	
Resumen		Informe de resumen manual	
Resumen del proyecto		Resumen manual	
Tareas externas		Sólo el comienzo	
Hito externo		Sólo fin	
Tarea inactiva		Fecha límite	
Hito inactivo		Progreso	

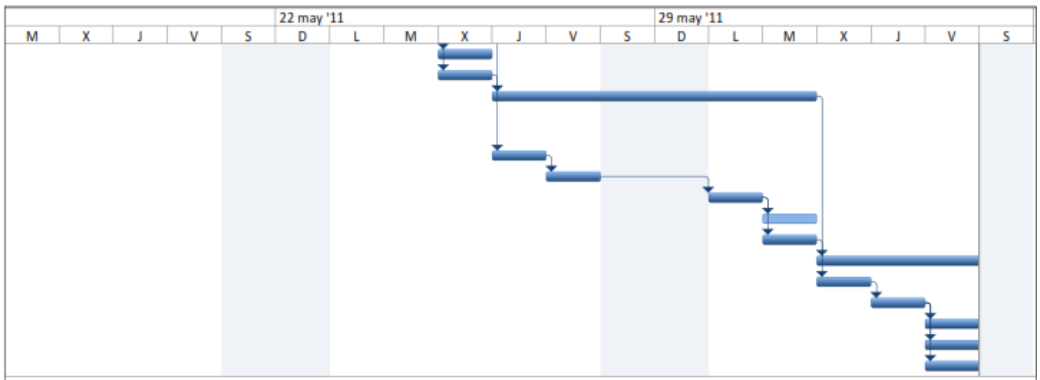
Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
18		Pruebas unitarias	1 día?	mié 25/05/11	mié 25/05/11
19		Generar la documentación	1 día?	mié 25/05/11	mié 25/05/11
20		**Como inspector, deseo eliminar informes de inspección**	4 días	jue 26/05/11	mar 31/05/11
21		Generar las vistas	1 día?	jue 26/05/11	jue 26/05/11
22		Generar los controladores	1 día?	vie 27/05/11	vie 27/05/11
23		Generar los modelos	1 día?	lun 30/05/11	lun 30/05/11
24		Pruebas unitarias	1 día?	mar 31/05/11	mar 31/05/11
25		Generar la documentación	1 día?	mar 31/05/11	mar 31/05/11
26		**Como inspector, deseo generar l**	3 días	mié 01/06/11	vie 03/06/11
27		Generar las vistas	1 día?	mié 01/06/11	mié 01/06/11
28		Generar los controladores	1 día?	jue 02/06/11	jue 02/06/11
29		Generar los modelos	1 día?	vie 03/06/11	vie 03/06/11
30		Pruebas unitarias	1 día?	vie 03/06/11	vie 03/06/11
31		Generar la documentación	1 día?	vie 03/06/11	vie 03/06/11



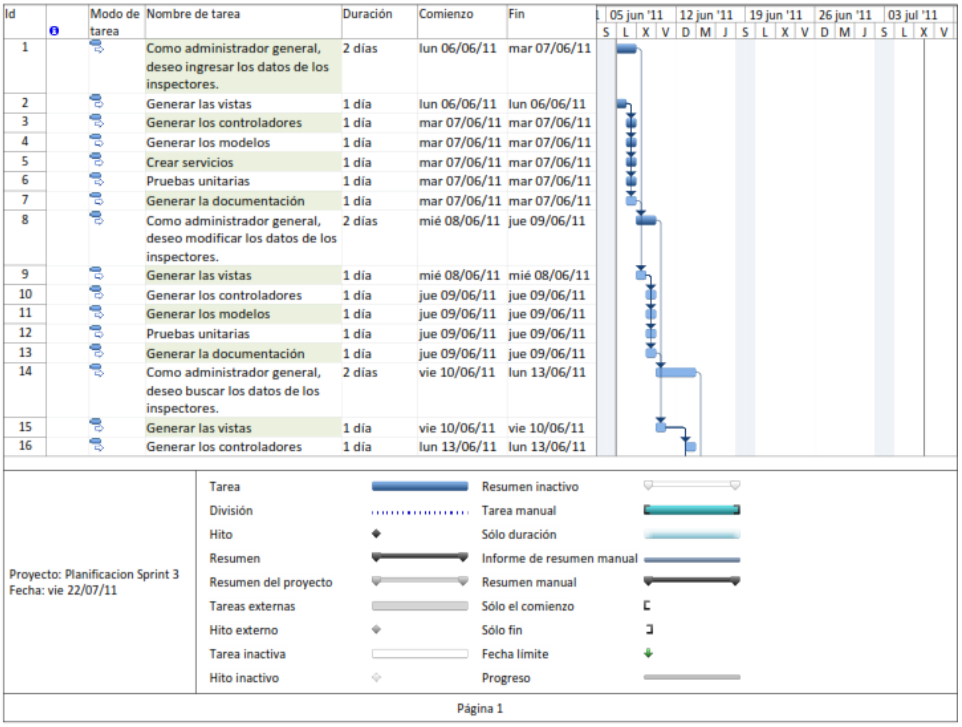
Proyecto: Planificacion Sprint 2
Fecha: vie 22/07/11

Tarea		Resumen inactivo	
División		Tarea manual	
Hito		Sólo duración	
Resumen		Informe de resumen manual	
Resumen del proyecto		Resumen manual	
Tareas externas		Sólo el comienzo	
Hito externo		Sólo fin	
Tarea inactiva		Fecha límite	
Hito inactivo		Progreso	


22 may '11
29 may '11
M X J V S D L M X J V S D L M X J V S
Proyecto: Planificacion Sprint 2
Fecha: vie 22/07/11
Tarea
División
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Tarea inactiva
Hito inactivo
Resumen inactivo
Tarea manual
Sólo duración
Informe de resumen manual
Resumen manual
Sólo el comienzo
Sólo fin
Fecha límite
Progreso


22 may '11
29 may '11
M X J V S D L M X J V S D L M X J V S
Proyecto: Planificacion Sprint 2
Fecha: vie 22/07/11
Tarea
División
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Tarea inactiva
Hito inactivo
Resumen inactivo
Tarea manual
Sólo duración
Informe de resumen manual
Resumen manual
Sólo el comienzo
Sólo fin
Fecha límite
Progreso

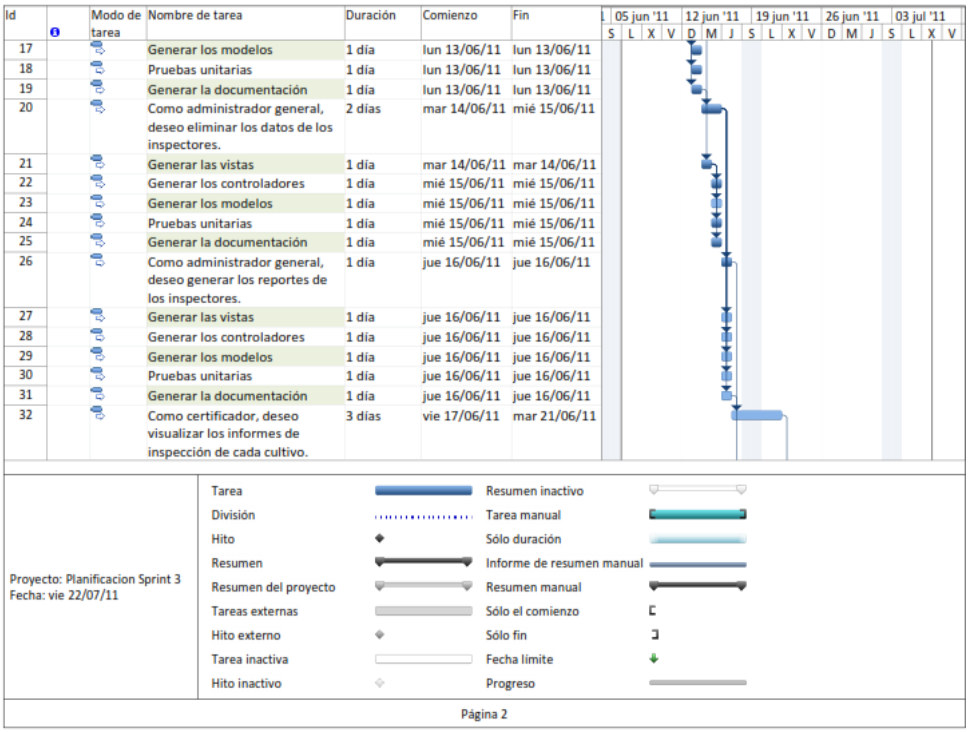
SPRINT 2



Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		Como administrador general, deseo ingresar los datos de los inspectores.	2 días	lun 06/06/11	mar 07/06/11
2		Generar las vistas	1 día	lun 06/06/11	lun 06/06/11
3		Generar los controladores	1 día	mar 07/06/11	mar 07/06/11
4		Generar los modelos	1 día	mar 07/06/11	mar 07/06/11
5		Crear servicios	1 día	mar 07/06/11	mar 07/06/11
6		Pruebas unitarias	1 día	mar 07/06/11	mar 07/06/11
7		Generar la documentación	1 día	mar 07/06/11	mar 07/06/11
8		Como administrador general, deseo modificar los datos de los inspectores.	2 días	mié 08/06/11	jue 09/06/11
9		Generar las vistas	1 día	mié 08/06/11	mié 08/06/11
10		Generar los controladores	1 día	jue 09/06/11	jue 09/06/11
11		Generar los modelos	1 día	jue 09/06/11	jue 09/06/11
12		Pruebas unitarias	1 día	jue 09/06/11	jue 09/06/11
13		Generar la documentación	1 día	jue 09/06/11	jue 09/06/11
14		Como administrador general, deseo buscar los datos de los inspectores.	2 días	vie 10/06/11	lun 13/06/11
15		Generar las vistas	1 día	vie 10/06/11	vie 10/06/11
16		Generar los controladores	1 día	lun 13/06/11	lun 13/06/11

Proyecto: Planificacion Sprint 3
Fecha: vie 22/07/11

Tarea
División
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Tarea inactiva
Hito inactivo
Resumen inactivo
Tarea manual
Sólo duración
Informe de resumen manual
Resumen manual
Sólo el comienzo
Sólo fin
Fecha límite
Progreso

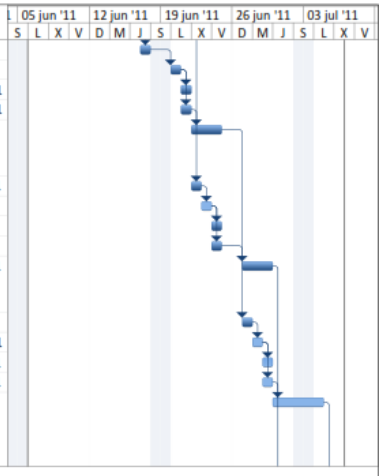


Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
17		Generar los modelos	1 día	lun 13/06/11	lun 13/06/11
18		Pruebas unitarias	1 día	lun 13/06/11	lun 13/06/11
19		Generar la documentación	1 día	lun 13/06/11	lun 13/06/11
20		Como administrador general, deseo eliminar los datos de los inspectores.	2 días	mar 14/06/11	mié 15/06/11
21		Generar las vistas	1 día	mar 14/06/11	mar 14/06/11
22		Generar los controladores	1 día	mié 15/06/11	mié 15/06/11
23		Generar los modelos	1 día	mié 15/06/11	mié 15/06/11
24		Pruebas unitarias	1 día	mié 15/06/11	mié 15/06/11
25		Generar la documentación	1 día	mié 15/06/11	mié 15/06/11
26		Como administrador general, deseo generar los reportes de los inspectores.	1 día	jue 16/06/11	jue 16/06/11
27		Generar las vistas	1 día	jue 16/06/11	jue 16/06/11
28		Generar los controladores	1 día	jue 16/06/11	jue 16/06/11
29		Generar los modelos	1 día	jue 16/06/11	jue 16/06/11
30		Pruebas unitarias	1 día	jue 16/06/11	jue 16/06/11
31		Generar la documentación	1 día	jue 16/06/11	jue 16/06/11
32		Como certificador, deseo visualizar los informes de inspección de cada cultivo.	3 días	vie 17/06/11	mar 21/06/11

Proyecto: Planificacion Sprint 3
Fecha: vie 22/07/11

Tarea
División
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Tarea inactiva
Hito inactivo
Resumen inactivo
Tarea manual
Sólo duración
Informe de resumen manual
Resumen manual
Sólo el comienzo
Sólo fin
Fecha límite
Progreso

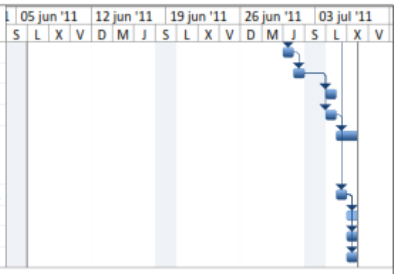
Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
33		Gererar el reporte	1 día	vie 17/06/11	vie 17/06/11
34		Integrar el reporte	1 día	lun 20/06/11	lun 20/06/11
35		Publicar el reporte	1 día	mar 21/06/11	mar 21/06/11
36		Generar la documentacion	1 día	mar 21/06/11	mar 21/06/11
37		Como administrador de ERPE, deseo visualizar los reportes estadísticos de producción	3 días	mié 22/06/11	vie 24/06/11
38		Gererar el reporte	1 día	mié 22/06/11	mié 22/06/11
39		Integrar el reporte	1 día	jue 23/06/11	jue 23/06/11
40		Publicar el reporte	1 día	vie 24/06/11	vie 24/06/11
41		Generar la documentacion	1 día	vie 24/06/11	vie 24/06/11
42		Como administrador de ERPE, deseo visualizar el ingreso económico de cada productor	3 días	lun 27/06/11	mié 29/06/11
43		Gererar el reporte	1 día	lun 27/06/11	lun 27/06/11
44		Integrar el reporte	1 día	mar 28/06/11	mar 28/06/11
45		Publicar el reporte	1 día	mié 29/06/11	mié 29/06/11
46		Generar la documentacion	1 día	mié 29/06/11	mié 29/06/11
47		Como administrador, inspector, deseo visualizar los informes de producción por parcela.	3 días	jue 30/06/11	lun 04/07/11



Proyecto: Planificacion Sprint 3
Fecha: vie 22/07/11

Tarea		Resumen inactivo	
División		Tarea manual	
Hito		Sólo duración	
Resumen		Informe de resumen manual	
Resumen del proyecto		Resumen manual	
Tareas externas		Sólo el comienzo	
Hito externo		Sólo fin	
Tarea inactiva		Fecha límite	
Hito inactivo		Progreso	

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
48		Gererar el reporte	1 día	jue 30/06/11	jue 30/06/11
49		Integrar el reporte	1 día	vie 01/07/11	vie 01/07/11
50		Publicar el reporte	1 día	lun 04/07/11	lun 04/07/11
51		Generar la documentacion	1 día	lun 04/07/11	lun 04/07/11
52		Como productor asociado, deseo visualizar mi ingreso económico.	2 días	mar 05/07/11	mié 06/07/11
53		Gererar el reporte	1 día	mar 05/07/11	mar 05/07/11
54		Integrar el reporte	1 día	mié 06/07/11	mié 06/07/11
55		Publicar el reporte	1 día	mié 06/07/11	mié 06/07/11
56		Generar la documentacion	1 día	mié 06/07/11	mié 06/07/11

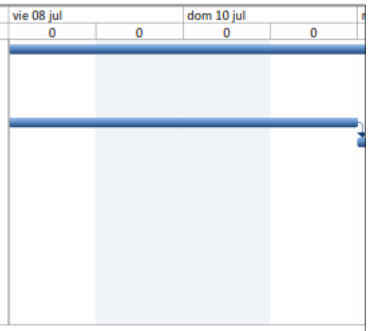


Proyecto: Planificacion Sprint 3
Fecha: vie 22/07/11

Tarea		Resumen inactivo	
División		Tarea manual	
Hito		Sólo duración	
Resumen		Informe de resumen manual	
Resumen del proyecto		Resumen manual	
Tareas externas		Sólo el comienzo	
Hito externo		Sólo fin	
Tarea inactiva		Fecha límite	
Hito inactivo		Progreso	

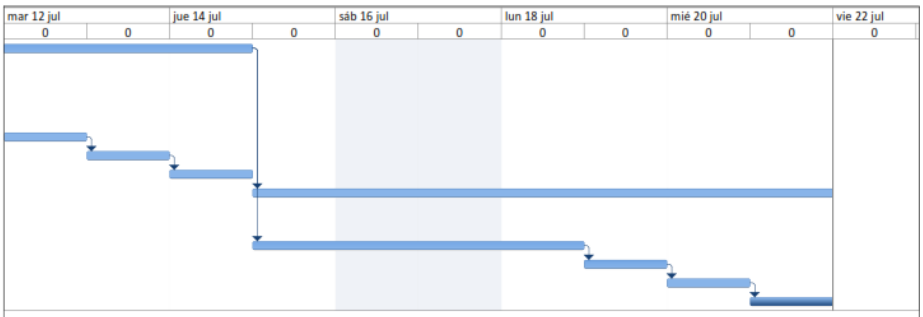
SPRINT 3

Id	Nombre de tarea	Duración	Comienzo	Fin
1	Como Inspector, administrador de ERPE, deseo visualizar el estado de agroforesteria de cada parcela.	5 días	vie 08/07/11	jue 14/07/11
2	Generar el reporte	2 días	vie 08/07/11	lun 11/07/11
3	Integrar el reporte	1 día	mar 12/07/11	mar 12/07/11
4	Publicar el reporte	1 día	mié 13/07/11	mié 13/07/11
5	Generar la documentacion	1 día	jue 14/07/11	jue 14/07/11
6	Como administrador, certificador, deseo visualizar los reportes de producción orgánica.	5 días	vie 15/07/11	jue 21/07/11
7	Gererar el reporte	2 días	vie 15/07/11	lun 18/07/11
8	Integrar el reporte	1 día	mar 19/07/11	mar 19/07/11
9	Publicar el reporte	1 día	mié 20/07/11	mié 20/07/11
10	Generar la documentacion	1 día	jue 21/07/11	jue 21/07/11



Proyecto: Planificacion Sprint 4 p
Fecha: lun 25/07/11

Tarea
División
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Tarea inactiva
Hito inactivo
Resumen inactivo
Tarea manual
Sólo duración
Informe de resumen manual
Resumen manual
Sólo el comienzo
Sólo fin
Fecha límite
Progreso

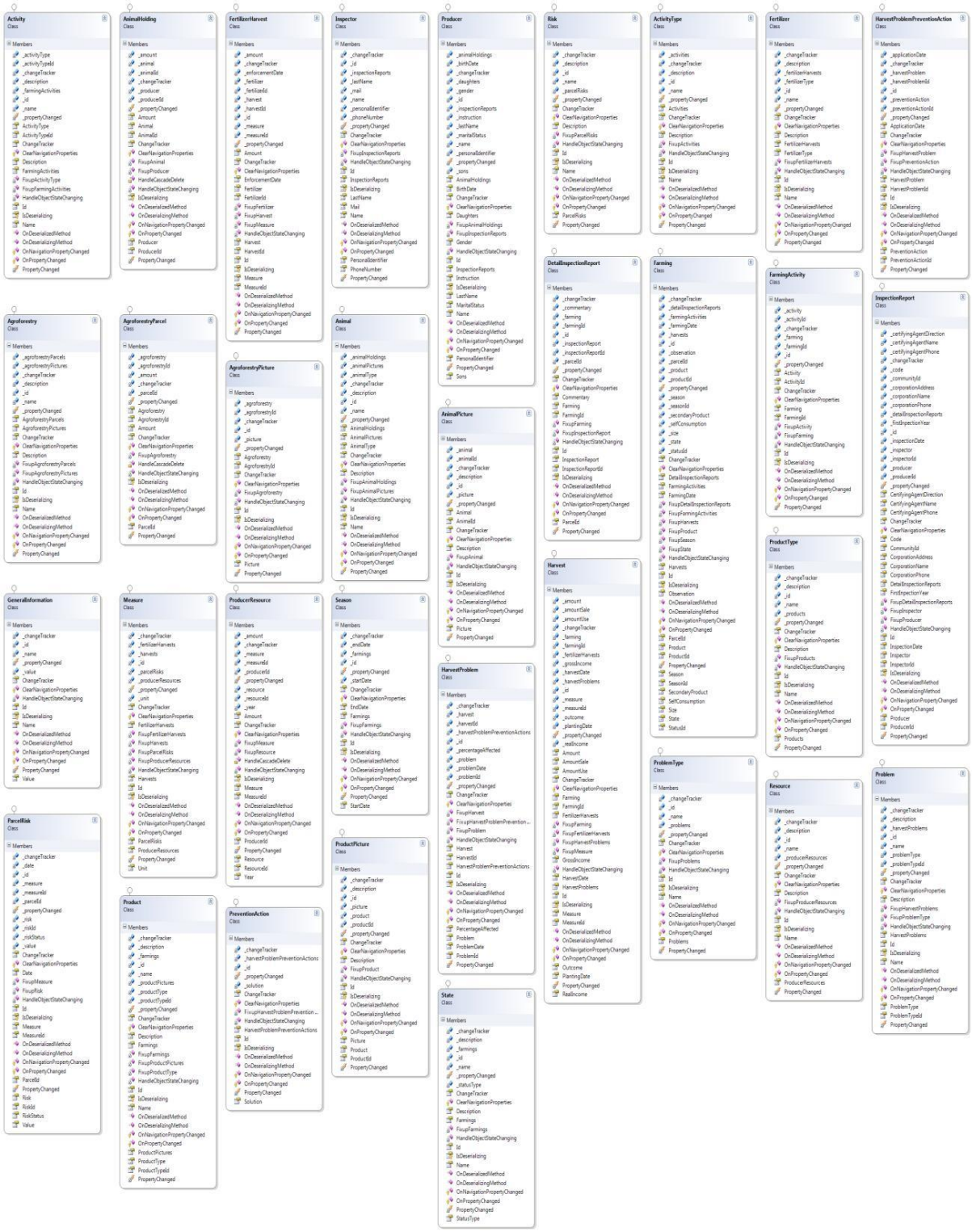


Proyecto: Planificacion Sprint 4 p
Fecha: lun 25/07/11

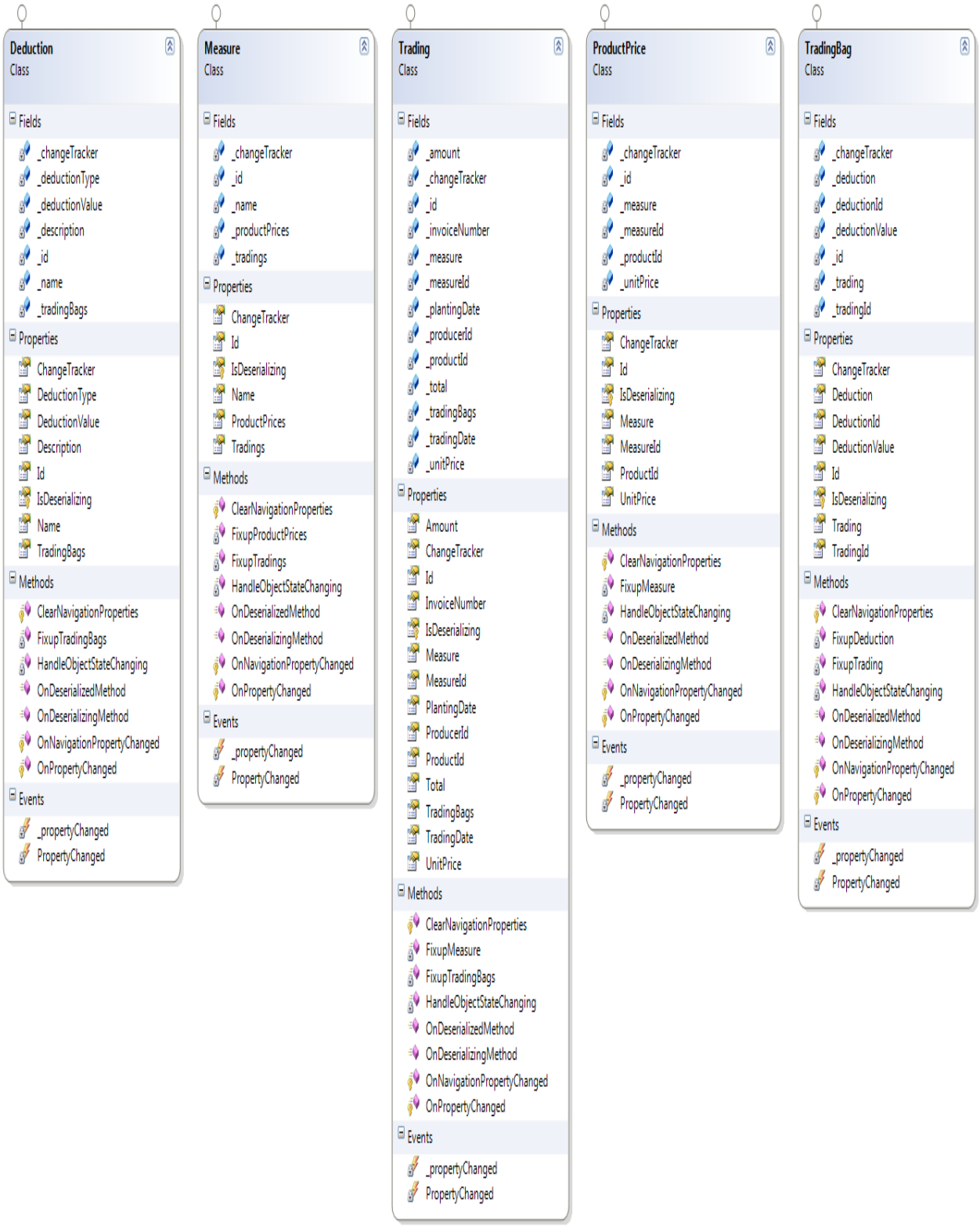
Tarea
División
Hito
Resumen
Resumen del proyecto
Tareas externas
Hito externo
Tarea inactiva
Hito inactivo
Resumen inactivo
Tarea manual
Sólo duración
Informe de resumen manual
Resumen manual
Sólo el comienzo
Sólo fin
Fecha límite
Progreso

Anexo 9: Diagrama de Clases de Diseño

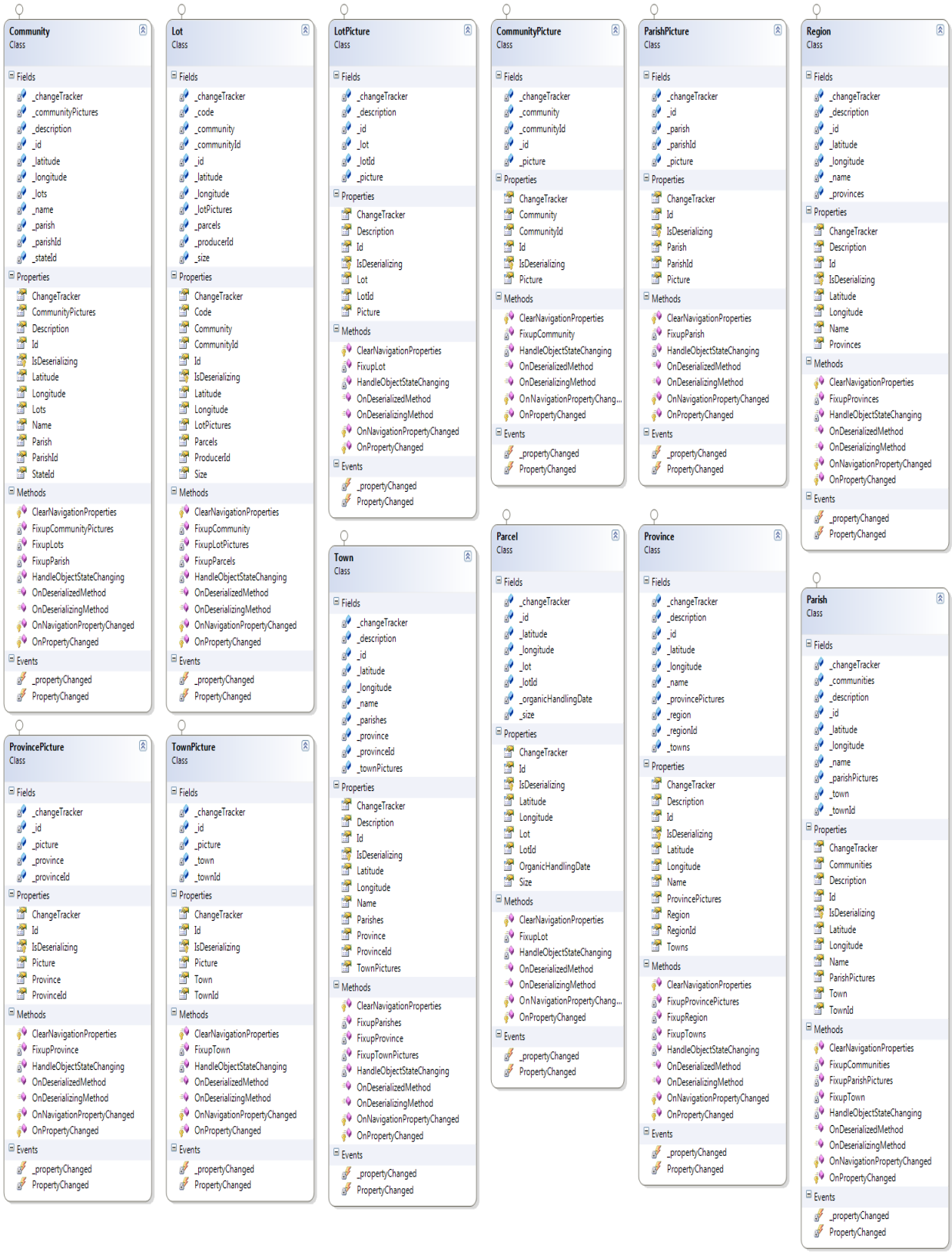
Esquema de Producción



Esquema de Marketing



Esquema de Geografía



Anexo 10: Diseño de interfaces de Usuario

Pantalla de Principal

Sprint 1

Pantalla Gestión Informe de Inspección

Pantalla Buscar Inspector

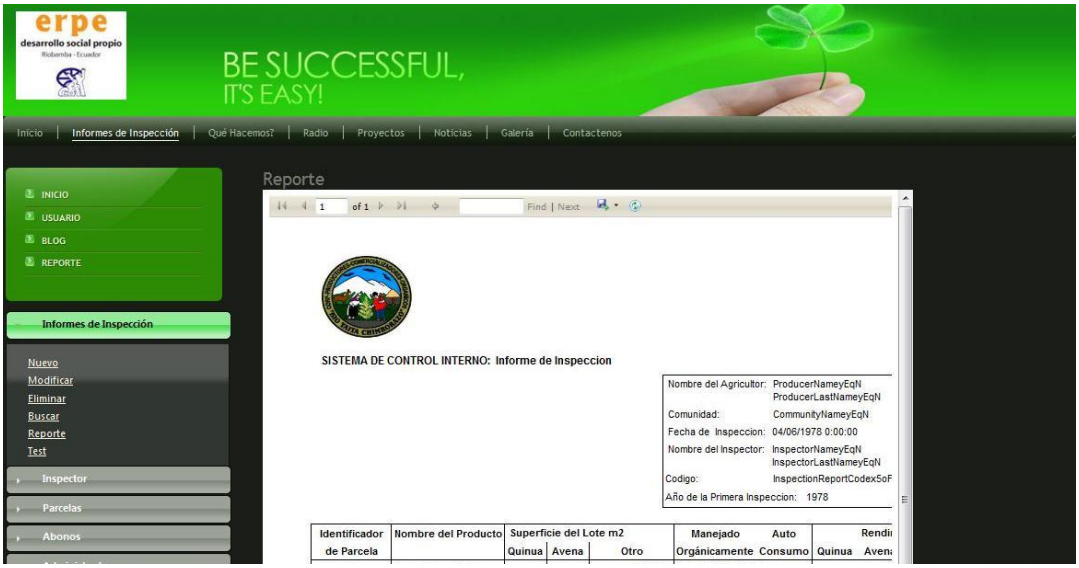
Pantalla Nuevo Informe de Inspección

Pantalla Buscar Informe de Inspección

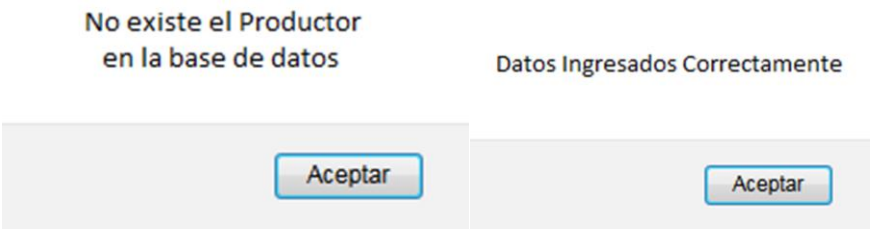
Pantalla Modificar Informe de Inspección

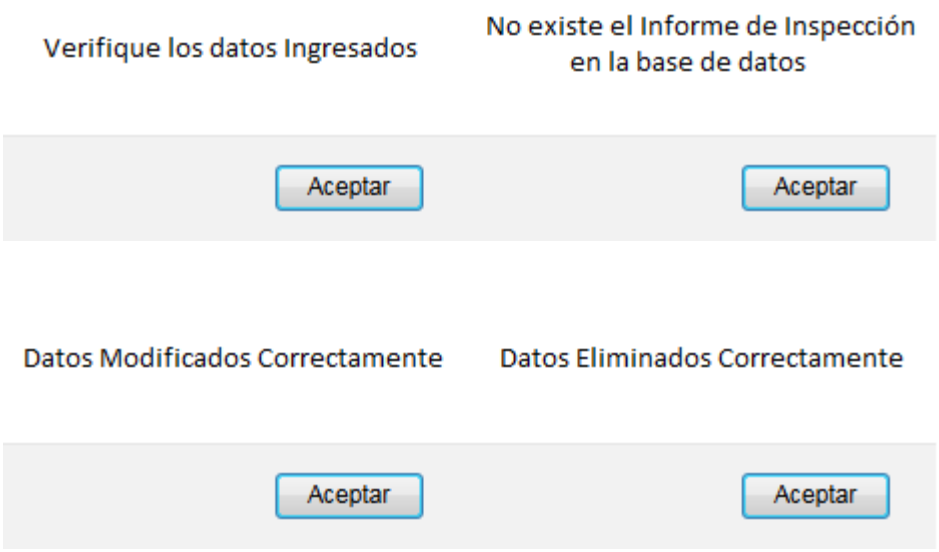
Pantalla Eliminar Informe de Inspección

Pantalla Reporte Informe de Inspección



Mensajes





Sprint 2

Pantalla Gestión Inspectores

Pantalla Ingresar Inspector

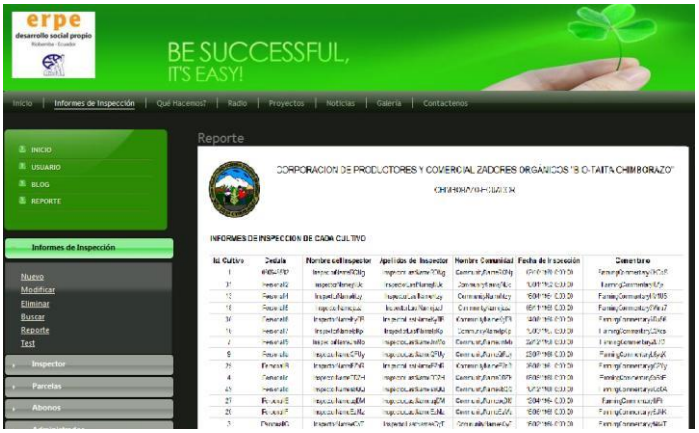
Pantalla Buscar Inspector

Pantalla Modificar Inspector

Pantalla Eliminar Inspector

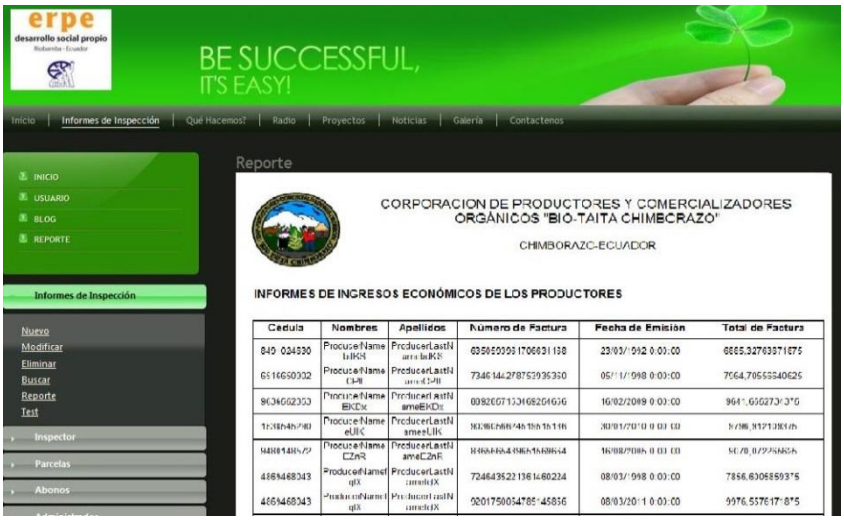
Pantalla Reporte Inspector

Pantalla Reporte Informe de Inspección



Pantalla Reporte Estadística de Producción

Pantalla Reporte Ingreso Económico



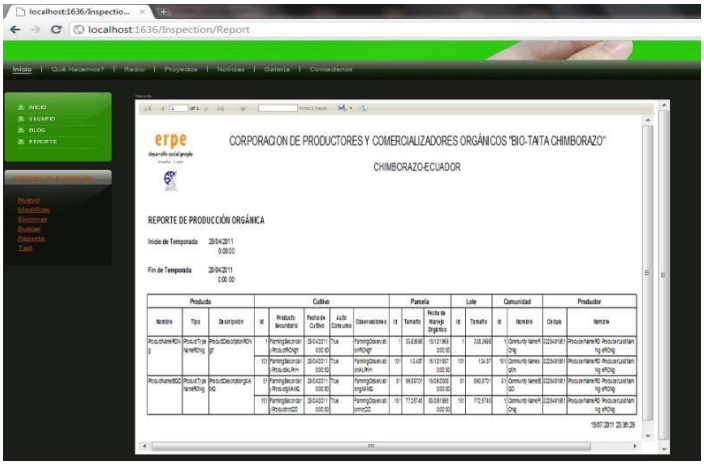
Pantalla Reporte Producción

Pantalla Reporte Ingreso Productores

Sprint 3

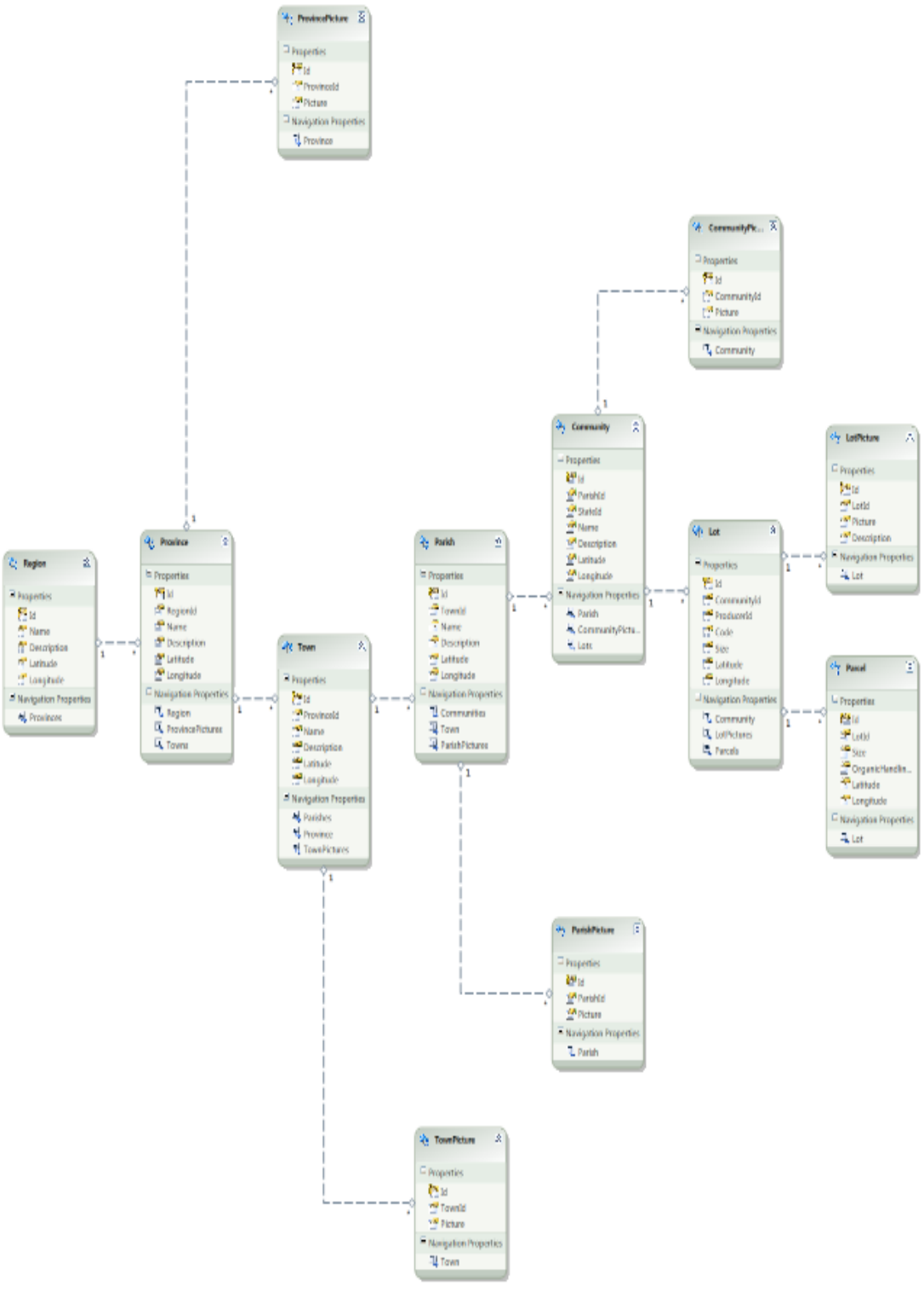
Pantalla Reporte Estado de Agroforestería

Pantalla Reporte Producción Orgánica

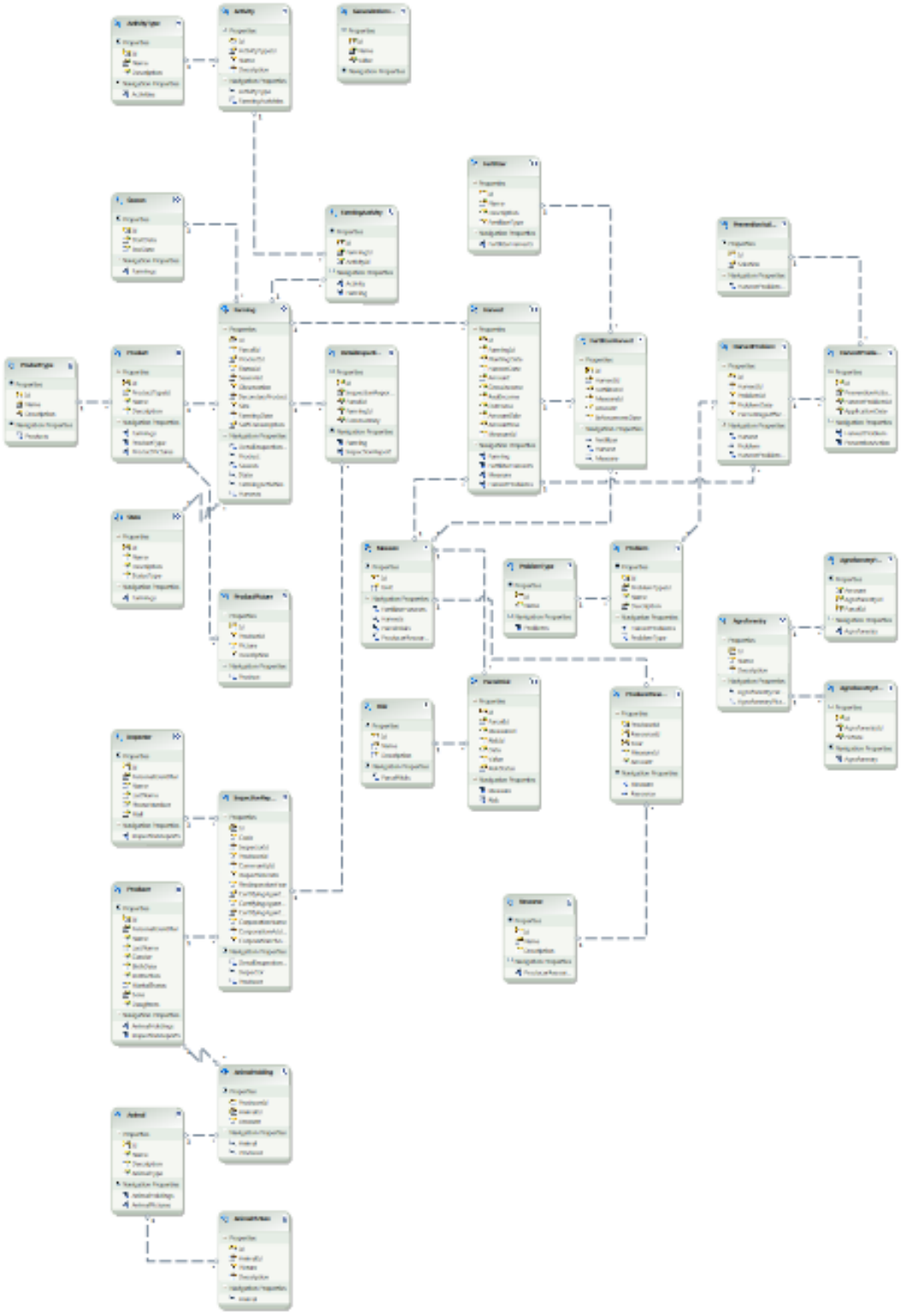


Anexo 11: Modelo Lógico de la Base de Datos

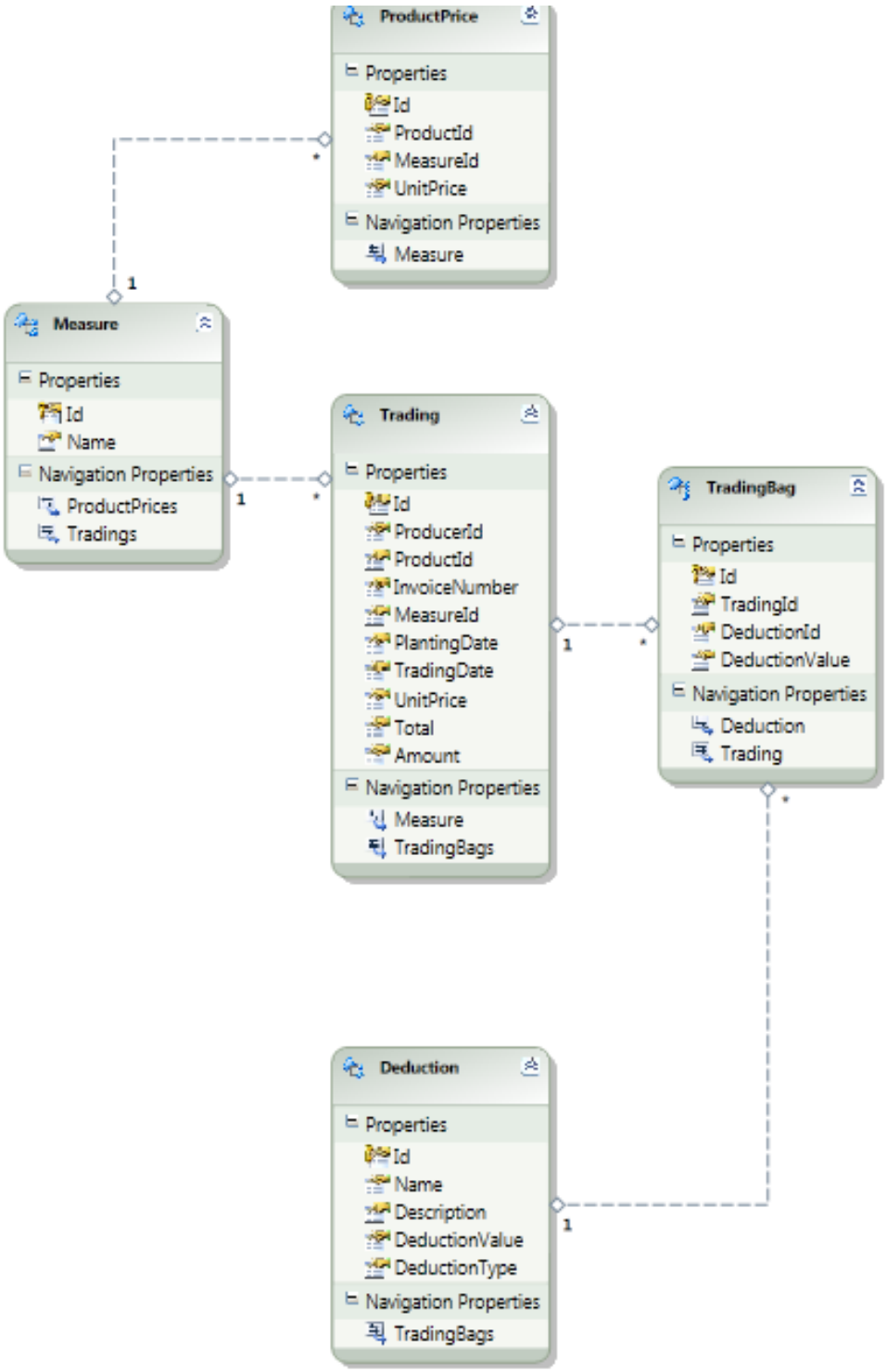
Esquema Geografía


ProvincePicture
Properties
Id
ProvinceId
Picture
Navigation Properties
Province
CommunityPic...
Properties
Id
CommunityId
Picture
Navigation Properties
Community
Community
Properties
Id
ParishId
StaffId
Name
Description
Latitude
Longitude
Navigation Properties
Parish
CommunityPicture
Lots
LotPicture
Properties
Id
LotId
Picture
Description
Navigation Properties
Lot
Region
Properties
Id
Name
Description
Latitude
Longitude
Navigation Properties
Provinces
Province
Properties
Id
RegionId
Name
Description
Latitude
Longitude
Navigation Properties
Region
ProvincePictures
Towns
Town
Properties
Id
ProvinceId
Name
Description
Latitude
Longitude
Navigation Properties
Parishes
Province
TownPictures
Parish
Properties
Id
TownId
Name
Description
Latitude
Longitude
Navigation Properties
Communities
Town
ParishPictures
Lot
Properties
Id
CommunityId
ProducerId
Code
Size
Latitude
Longitude
Navigation Properties
Community
LotPictures
Parcels
Parcel
Properties
Id
LotId
Size
OrganicHandlin...
Latitude
Longitude
Navigation Properties
Lot
ParishPicture
Properties
Id
ParishId
Picture
Navigation Properties
Parish
TownPicture
Properties
Id
TownId
Picture
Navigation Properties
Town

Esquema Producción

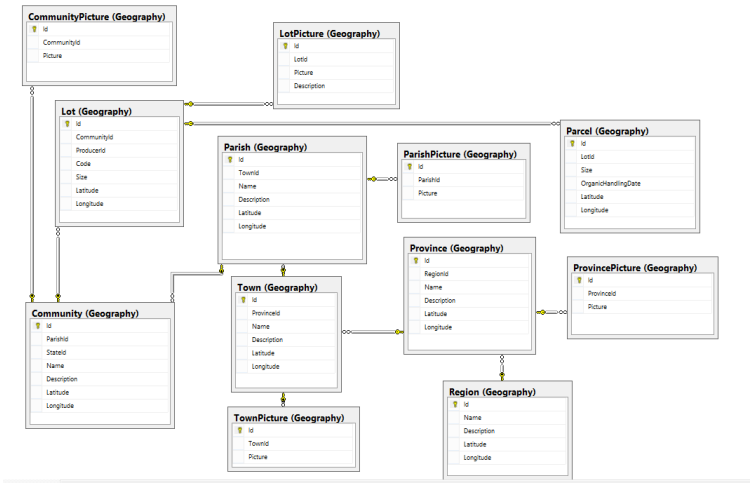


Esquema Marketing

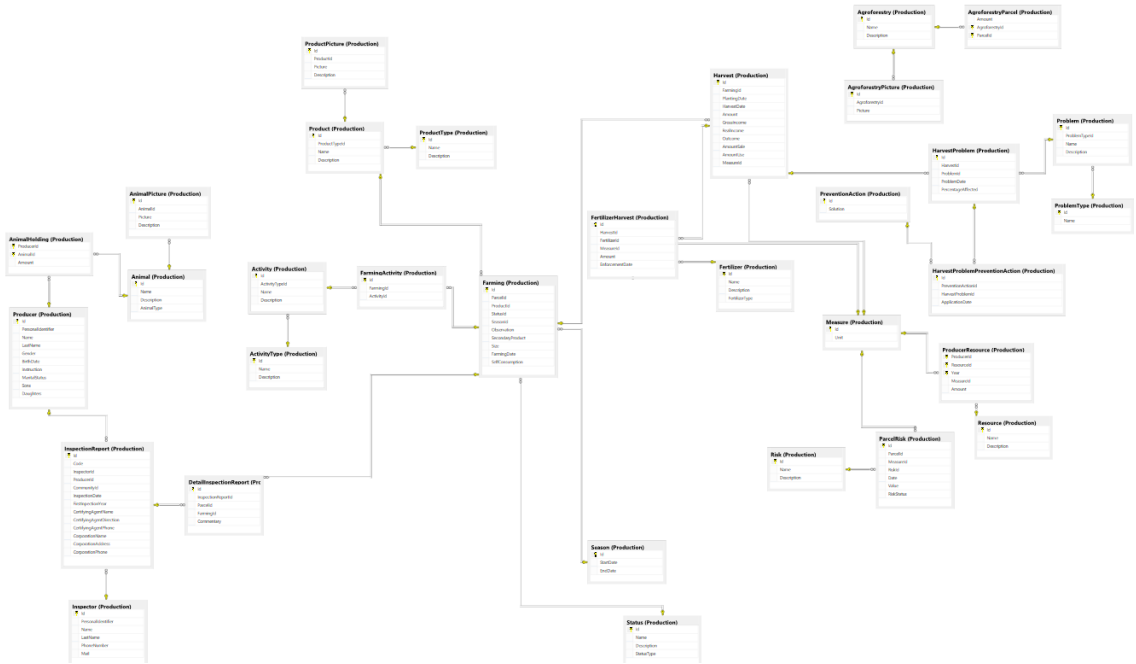


Anexo 12: Modelo Físico de la Base de Datos

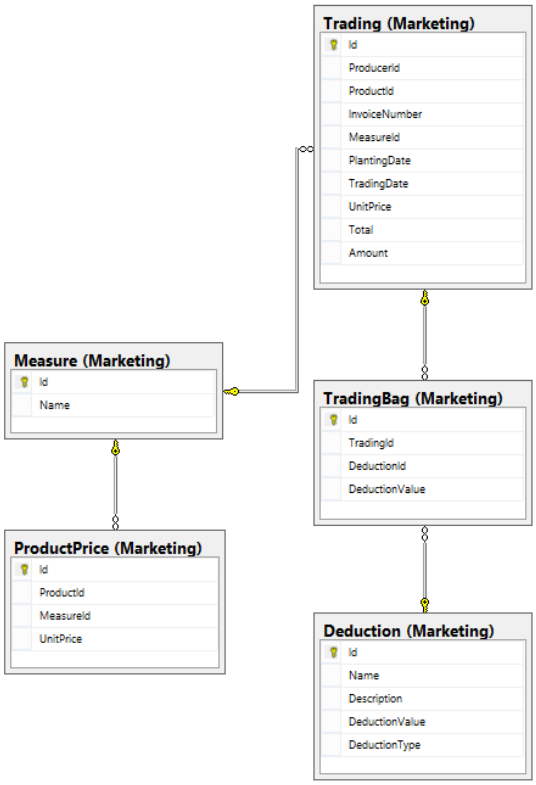
Esquema de Geografía



Esquema de Producción



Esquema de Marketing



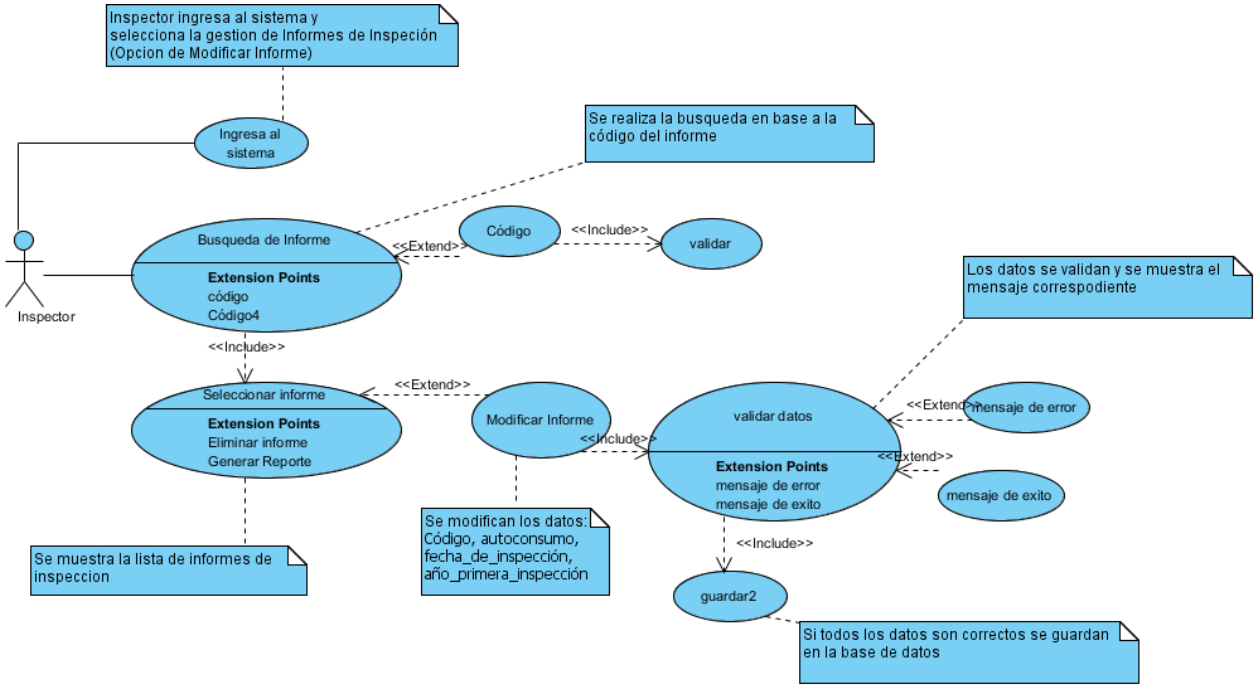
Anexo 13: Casos de Usos

SPRINT 1

Como inspector, deseo modificar el informe de inspección

Identificador de Caso de Uso	Modificación Informe Inspección
Caso de Uso:	CU_Modificación_Informe_Inspección.
Actores:	Inspector
Propósito:	Realizar la operación de modificación de los informes de
Visión General:	El inspector es la persona encargada en modificar los datos del informe de inspección en caso de existir errores mínimos en los datos, para almacenarlos y de esta manera llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	

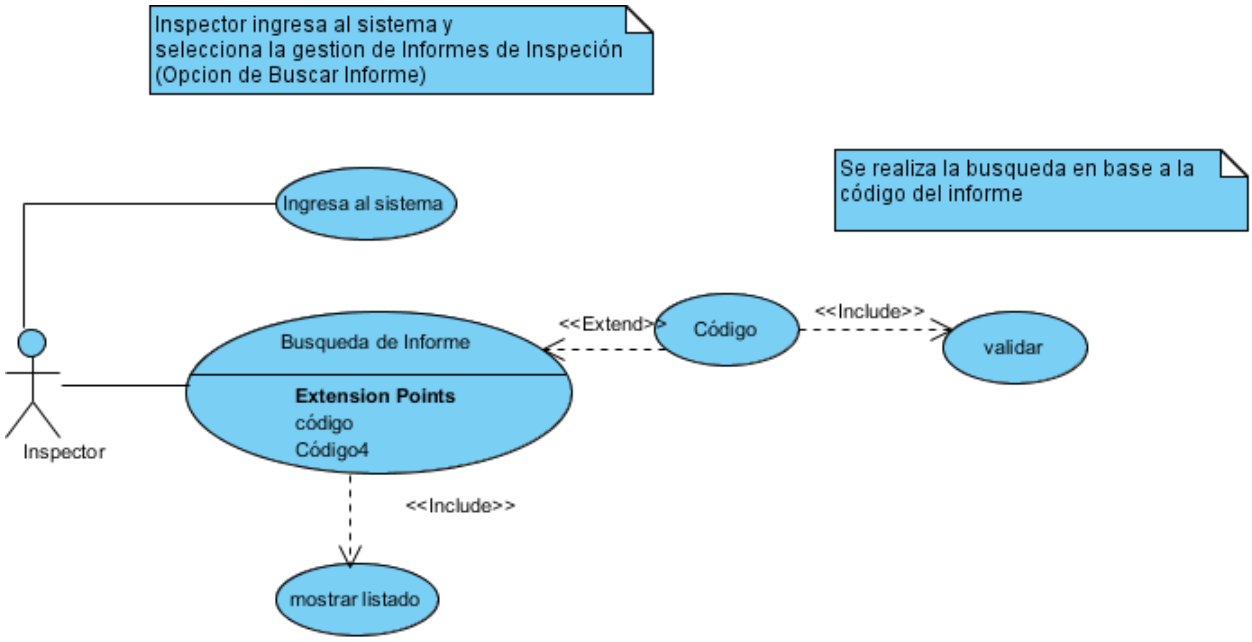
ACCION DEL ACTOR	RESPUESTA DEL SISTEMA
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los informes de inspección
3. Escoge la opción Informe de Inspección.	4. Muestra las opciones de : Nuevo Informe, Modificar Informe, Buscar informe, Eliminar Informe y Generar Reporte
5. Escoge la opción de Modificar Informe	6. Presentar la interfaz para buscar Informe de Inspección
7. Ingresar el código del informe deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el informe	10. Obtener los datos del informe de producción
11. Modificar los campos necesarios (autoconsumo, fecha_de_inspección, año_primera_inspección)	12.IngresarInforme(Código,Inspector,Comunidad,autoconsumo,productor,fecha_de_inspección,año_primera_inspe cción)
13. Presionar el botón de guardar	14. Los datos se verifican y se mostrará un mensaje de Éxito
	15. GuardarInforme()
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el informe se muestra un mensaje de que el informe no existe Línea 7: Si existe algún error en los campos se muestra un aviso al usuario	



Como inspector, deseo buscar informes de inspección

Identificador de Caso de Uso	Búsqueda Informe Inspección
Caso de Uso:	CU_Búsqueda_Informe_Inspección.
Actores:	Inspector
Propósito:	Realizar la operación de búsqueda de los informes de Inspección
Visión General:	El inspector general es la persona encargada de realizar las búsquedas de los informes para conocer la información de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los informes de inspección

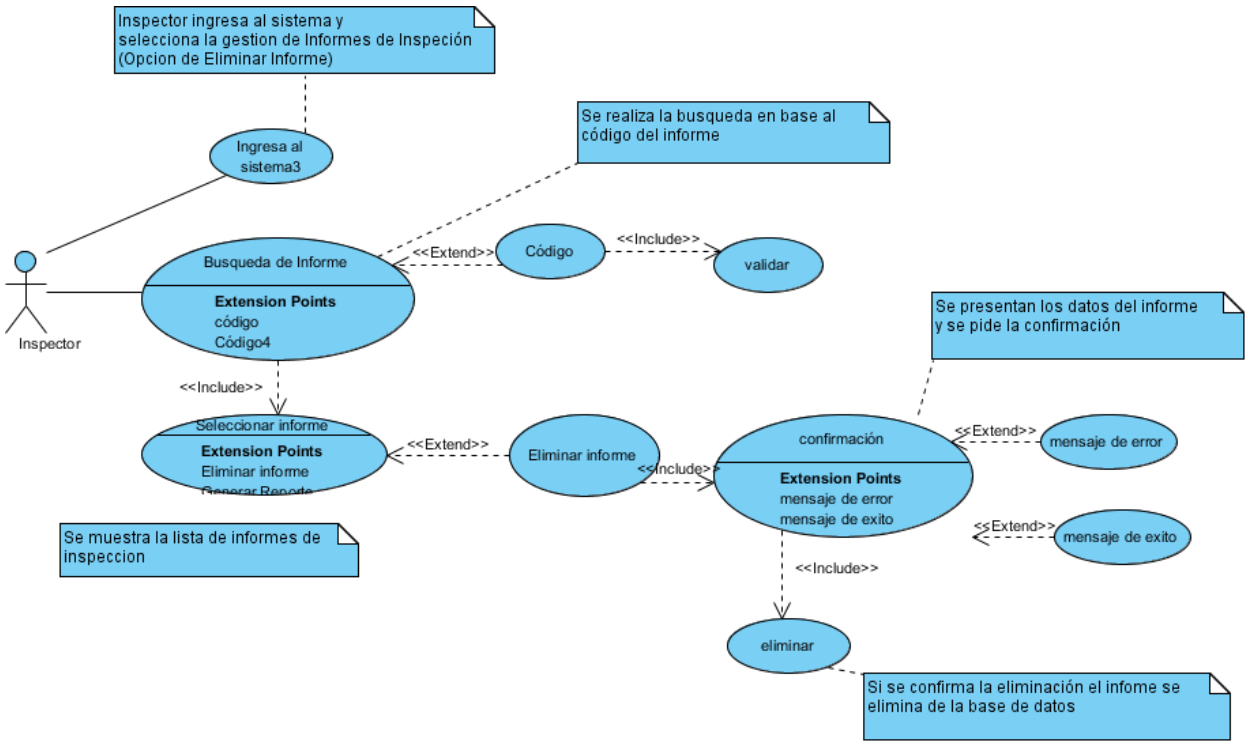
3. Escoge la opción Informe de Inspección.	4. Muestra las opciones de : Nuevo Informe, Modificar Informe, Buscar informe, Eliminar Informe y Generar Reporte
5. Escoge la opción de Buscar Informe	6. Presentar la interfaz para buscar Informe de Inspección
7. Ingresar el código del informe	8. Buscar en la base de datos y mostrar
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el informe se muestra un mensaje de que el informe no existe	



Como inspector, deseo eliminar informes de inspección

Identificador de Caso de Uso	Eliminación Informe Inspección
Caso de Uso:	CU_Eliminación_Informe_Inspección.
Actores:	Inspector

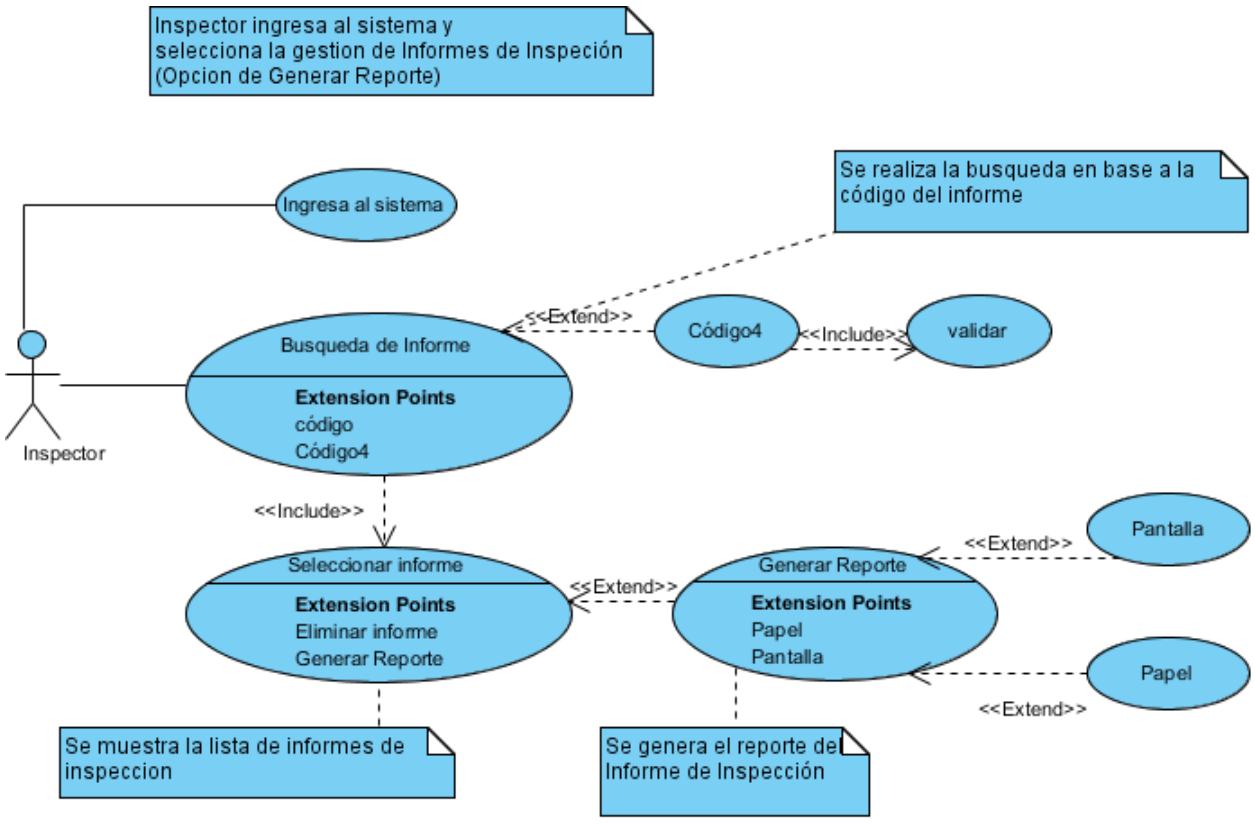
Propósito:	Realizar la operación de eliminación de los informes de Inspección
Visión General:	El inspector es la persona encargada de eliminar los informes de inspección en caso de existir errores importantes en los datos de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los informes de inspección
3. Escoge la opción Informe de Inspección.	4. Muestra las opciones de : Nuevo Informe, Modificar Informe, Buscar informe, Eliminar Informe y Generar Reporte
5. Escoge la opción de Eliminar Informe	6. Presentar la interfaz para buscar Informe de Inspección
7. Ingresar el código del informe deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el informe	10. Obtener los datos del informe de producción y mostrar para confirmar la eliminación
11. Presionar el botón de borrar	12.EliminarInforme(Código)
CURSOS ALTERNATIVOS	
Línea 6: Si no se desea borrar el informe se presiona regresar y se muestra la pantalla anterior cancelando la eliminación	



Como inspector, deseo generar los reportes sobre informes de inspección.

Identificador de Caso de Uso	Reporte Informe Inspección
Caso de Uso:	CU_Reporte_Informe_Inspección.
Actores:	Inspector
Propósito:	Realizar la operación de generar reportes de los informes de Inspección
Visión General:	El inspector es la persona encargada de generar los reportes de los informes de inspección para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**

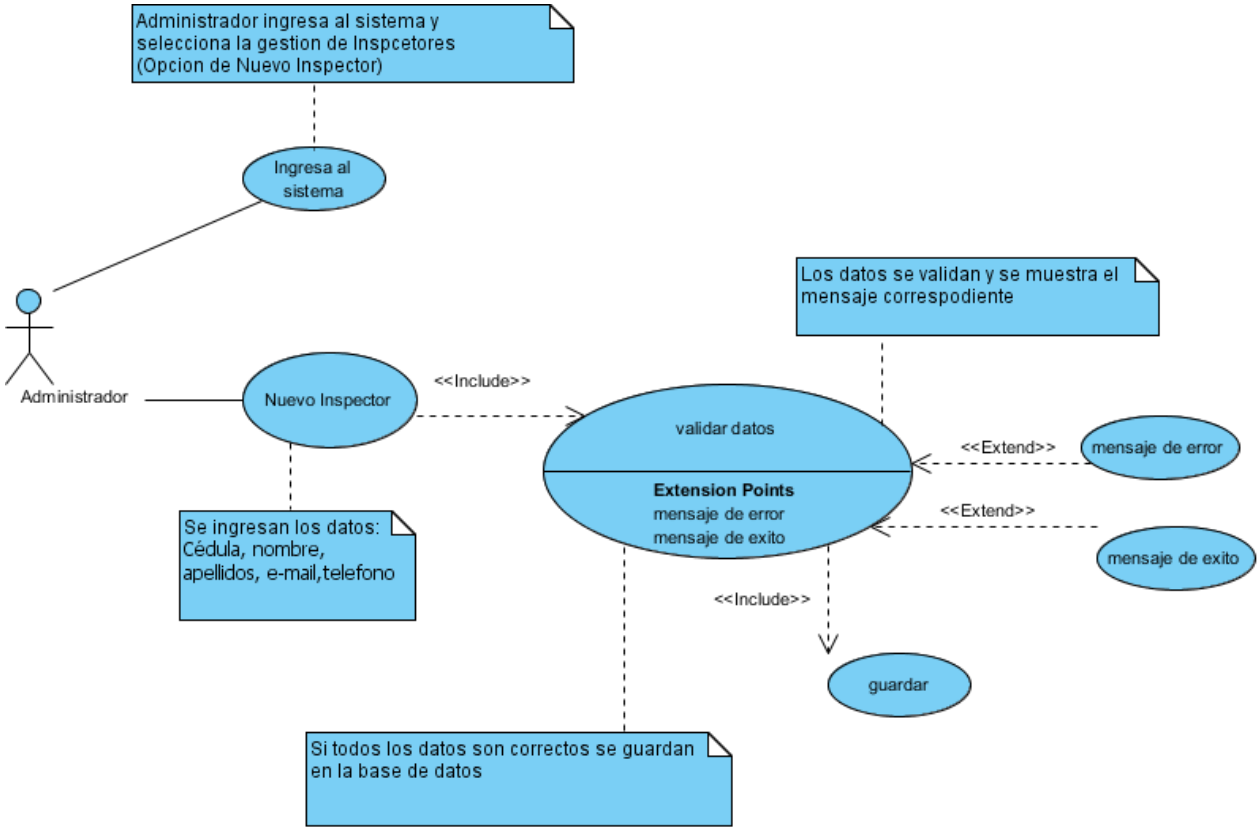
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los informes de inspección
3. Escoge la opción Informe de Inspección.	4. Muestra las opciones de : Nuevo Informe, Modificar Informe, Buscar informe, Eliminar Informe y Generar Reporte
5. Escoge la opción de Generar Reporte	6. Presentar la interfaz para buscar Informe de Inspección
7. Ingresar el código del informe deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el informe	10. Generar y mostrar reporte de Informes de Inspección
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el informe se muestra un mensaje de que el informe no existe	



SPRINT 2

Como administrador general, deseo ingresar los datos de los inspectores.

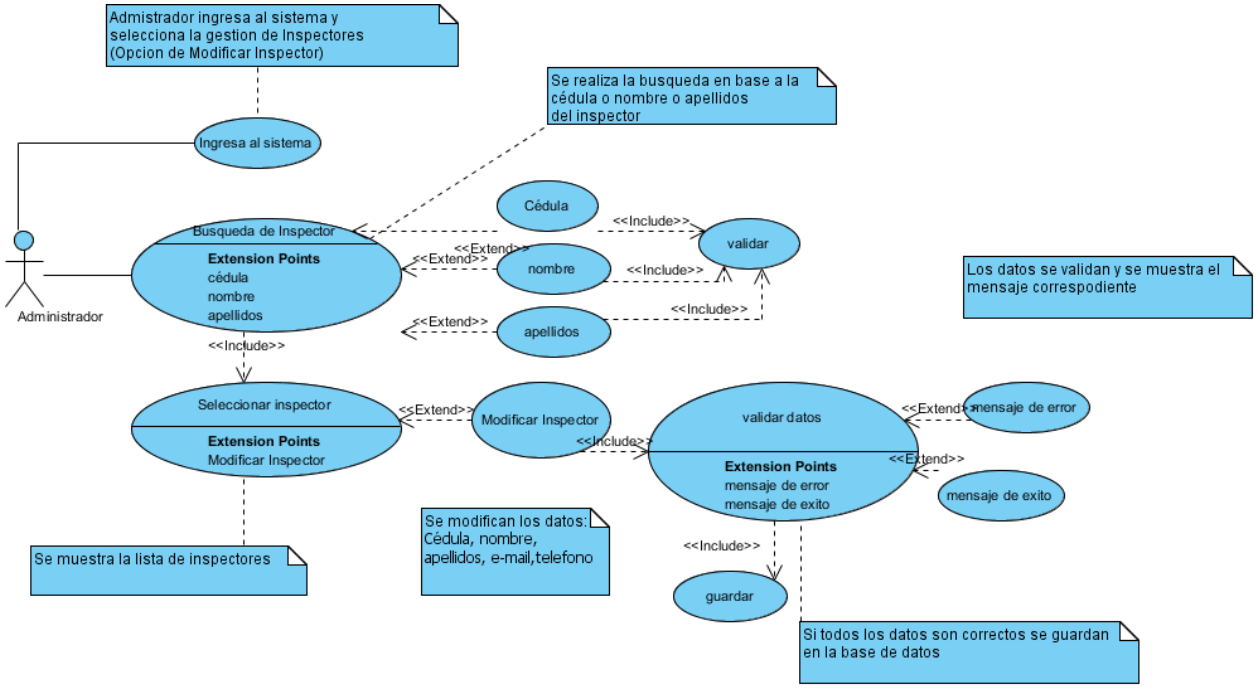
Identificador de Caso de Uso	Ingreso Inspector
Caso de Uso:	CU_Ingreso_Inspector
Actores:	Administrador
Propósito:	Realizar la operación del ingreso de los inspctores
Visión General:	El administrador general es la persona encargada en ingresar los datos de los inspectores para almacenarlos y de esta manera llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los inspectores
3. Escoge la opción Inspectores	4. Muestra las opciones de : Nuevo Inspector, Modificar Inspector , Buscar Inspector , Eliminar Inspector y Generar Reporte
5. Escoge la opción de Nuevo Inspector	6. Presentar la interfaz para ingresar el Inspector
7. Ingresar (Cédula, nombre, apellidos, e-mail,telefono)	8.IngresarInspector(Cédula, nombre, apellidos, e-mail,telefono)
9. Presionar el botón de guardar	14. Los datos se verifican y se mostrará un mensaje de éxito
	13. GuardarInspector()
CURSOS ALTERNATIVOS	
Línea 5: Si existe algún error en los campos se muestra un aviso al usuario	



Como administrador general, deseo modificar los datos de los inspectores.

Identificador de Caso de Uso	Modificación Inspector
Caso de Uso:	CU_Modificación_Inspector.
Actores:	Administrador
Propósito:	Realizar la operación de modificación de los inspectores
Visión General:	El administrador general es la persona encargada en modificar los datos de los inspectores en caso de existir errores mínimos en los datos, para almacenarlos y de esta manera llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**

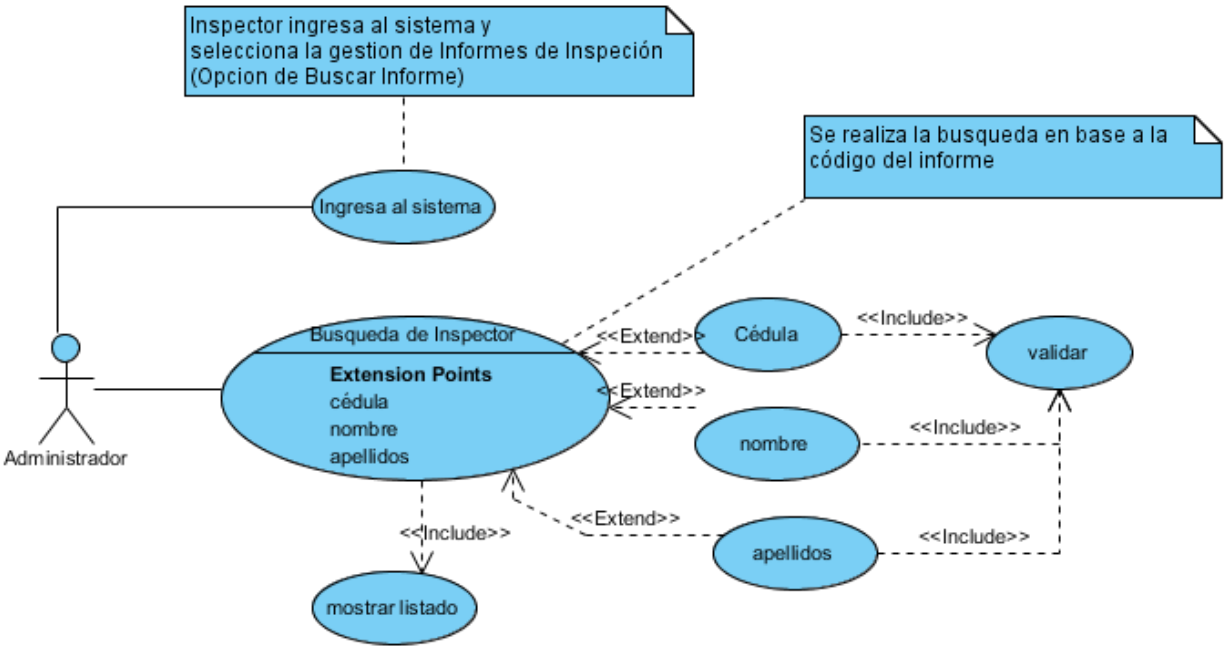
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los inspectores
3. Escoge la opción Inspectores	4. Muestra las opciones de : Nuevo Inspector, Modificar Inspector , Buscar Inspector , Eliminar Inspector y Generar Reporte
5. Escoge la opción de Modificar Inspector	6. Presentar la interfaz para buscar Inspector
7. Ingresar la cédula o nombre o apellido del inspector deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el inspector	10. Obtener los datos del inspector
11. Modificar los campos necesarios (Cédula, nombre, apellidos, e- mail,telefono)	12.IngresarInforme(Cédula, nombre, apellidos, e-mail,telefono)
13. Presionar el botón de guardar	14. Los datos se verifican y se mostrará un mensaje de éxito
	15. GuardarInspector()
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el informe se muestra un mensaje de que el inspector no existe Línea 7: Si existe algún error en los campos se muestra un aviso al usuario	



Como administrador general, deseo buscar los datos de los inspectores.

Identificador de Caso de Uso	Búsqueda Inspector
Caso de Uso:	CU_Búsqueda_Inspector
Actores:	Admininstrador
Propósito:	Realizar la operación de búsqueda de los inspector
Visión General:	El administrador general es la persona encargada de realizar las búsquedas de los inspectores para conocer la información de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los inspectores

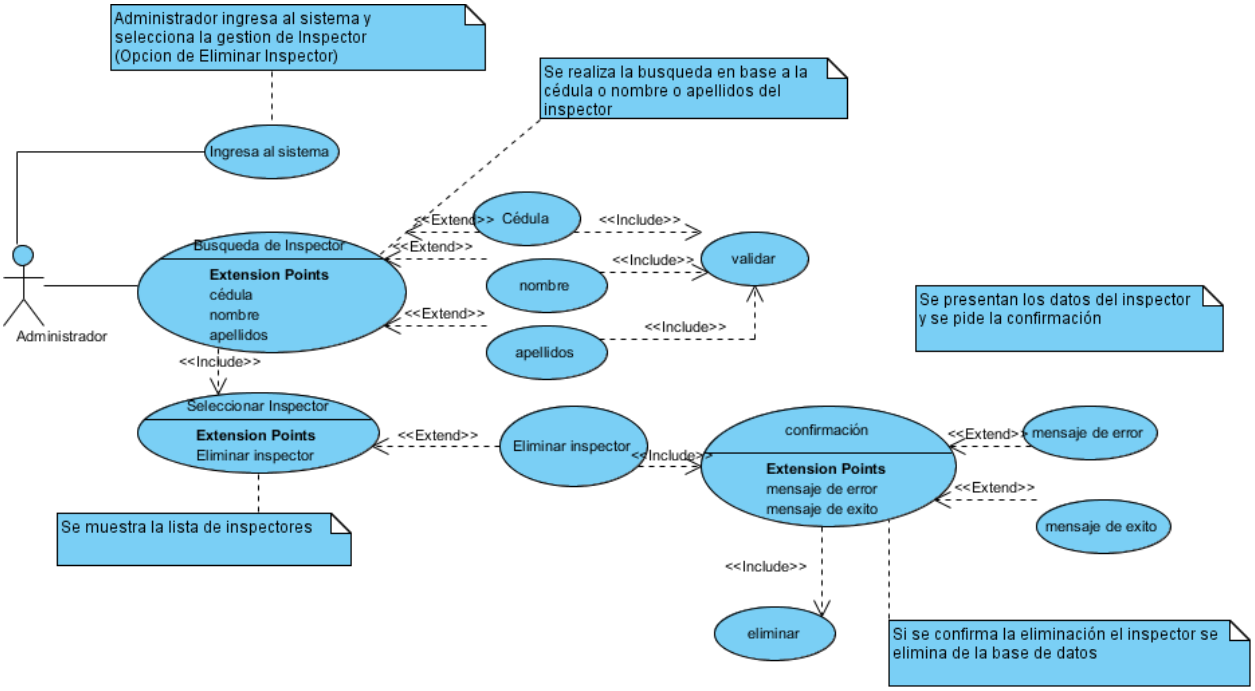
3. Escoge la opción Inspectores	4. Muestra las opciones de : Nuevo Inspector, Modificar Inspector , Buscar Inspector , Eliminar Inspector y Generar Reporte
5. Escoge la opción de Buscar inspector	6. Presentar la interfaz para buscar Inspector
7. Ingresar el código o nombre o apellidos del inspector	8. Buscar en la base de datos y mostrar
Cursos Alternativos	
Línea 4: Si no se encuentra el informe se muestra un mensaje de que el inspector no existe	



Como administrador general, deseo eliminar los datos de los inspectores.

Identificador de Caso de Uso	Eliminación Inspector
Caso de Uso:	CU_Eliminación_Inspector
Actores:	Administrador
Propósito:	Realizar la operación de eliminación de los inspectores

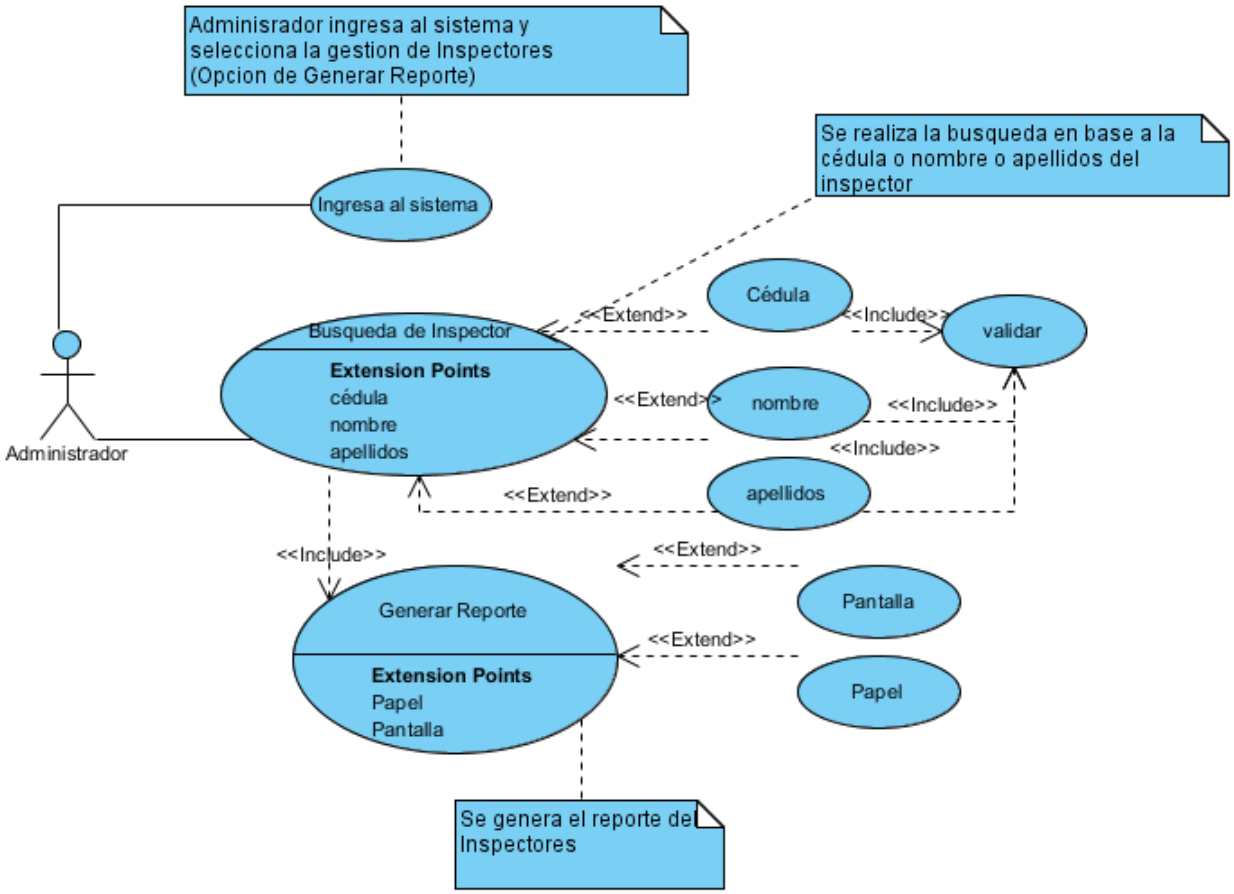
Visión General:	El administrador es la persona encargada de eliminar los datos de los inspectores en caso de existir errores importantes en los datos de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los inspectores
3. Escoge la opción Inspectores	4. Muestra las opciones de : Nuevo Inspector, Modificar Inspector , Buscar Inspector , Eliminar Inspector y Generar Reporte
5. Escoge la opción de Eliminar Inspector	6. Presentar la interfaz para buscar Inspector
7. Ingresar la cédula o nombre o apellido del inspector deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el inspector	10. Obtener los datos del informe de inspector y mostrar para confirmar la eliminación
11. Presionar el botón de borrar	12.EliminarInspector(Cédula)
CURSOS ALTERNATIVOS	
Línea 6: Si no se desea borrar el inspector se presiona regresar y se muestra la pantalla anterior cancelando la eliminación	



Como administrador general, deseo generar los reportes de los inspectores.

Identificador de Caso de Uso	Reporte Inspectores
Caso de Uso:	CU_Reporte_Inspectores
Actores:	Administrador
Propósito:	Realizar la operación de generar reportes de los inspectores
Visión General:	El administrador es la persona encargada de generar los reportes de los inspectores para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para la gestión de los inspectores

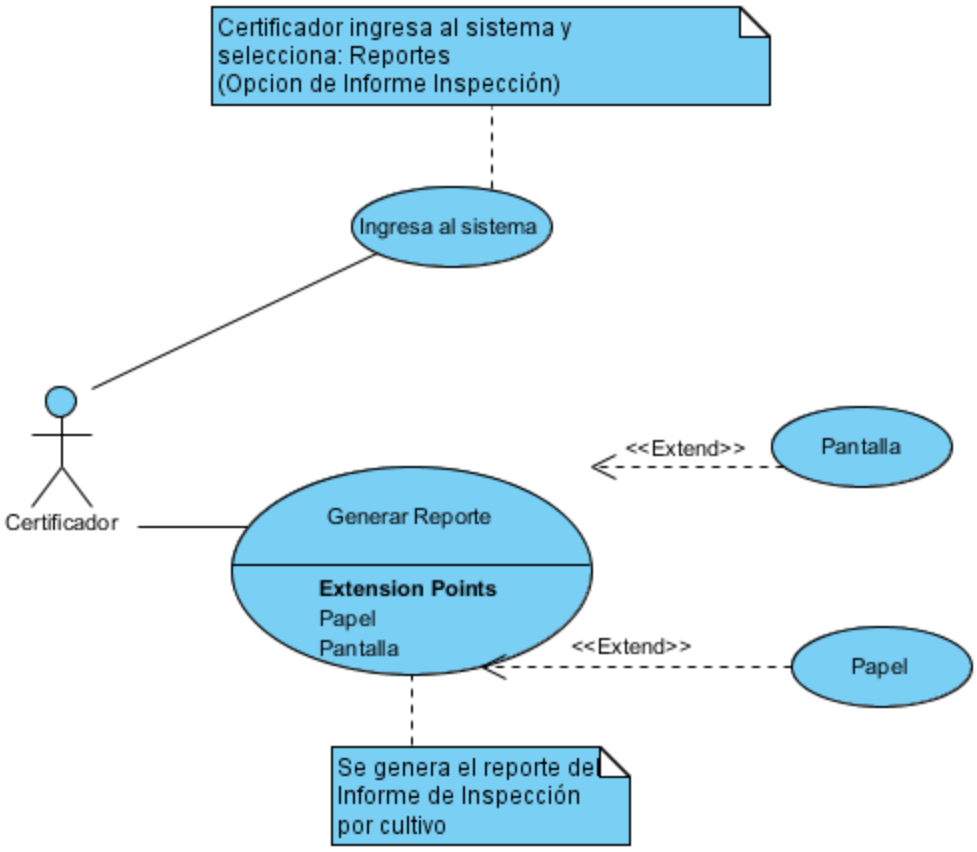
3. Escoge la opción Inspectores	4. Muestra las opciones de : Nuevo Inspector, Modificar Inspector , Buscar Inspector , Eliminar Inspector y Generar Reporte
5. Escoge la opción de Generar Reporte	6. Generar y mostrar reporte de Informes de Inspección
CURSOS ALTERNATIVOS	



Como certificador, deseo visualizar los informes de inspección de cada cultivo.

Identificador de Caso de Uso	Reporte Informe Inspección Cultivo
Caso de Uso:	CU_Reporte_Informe_Inspeccion_Cultivo

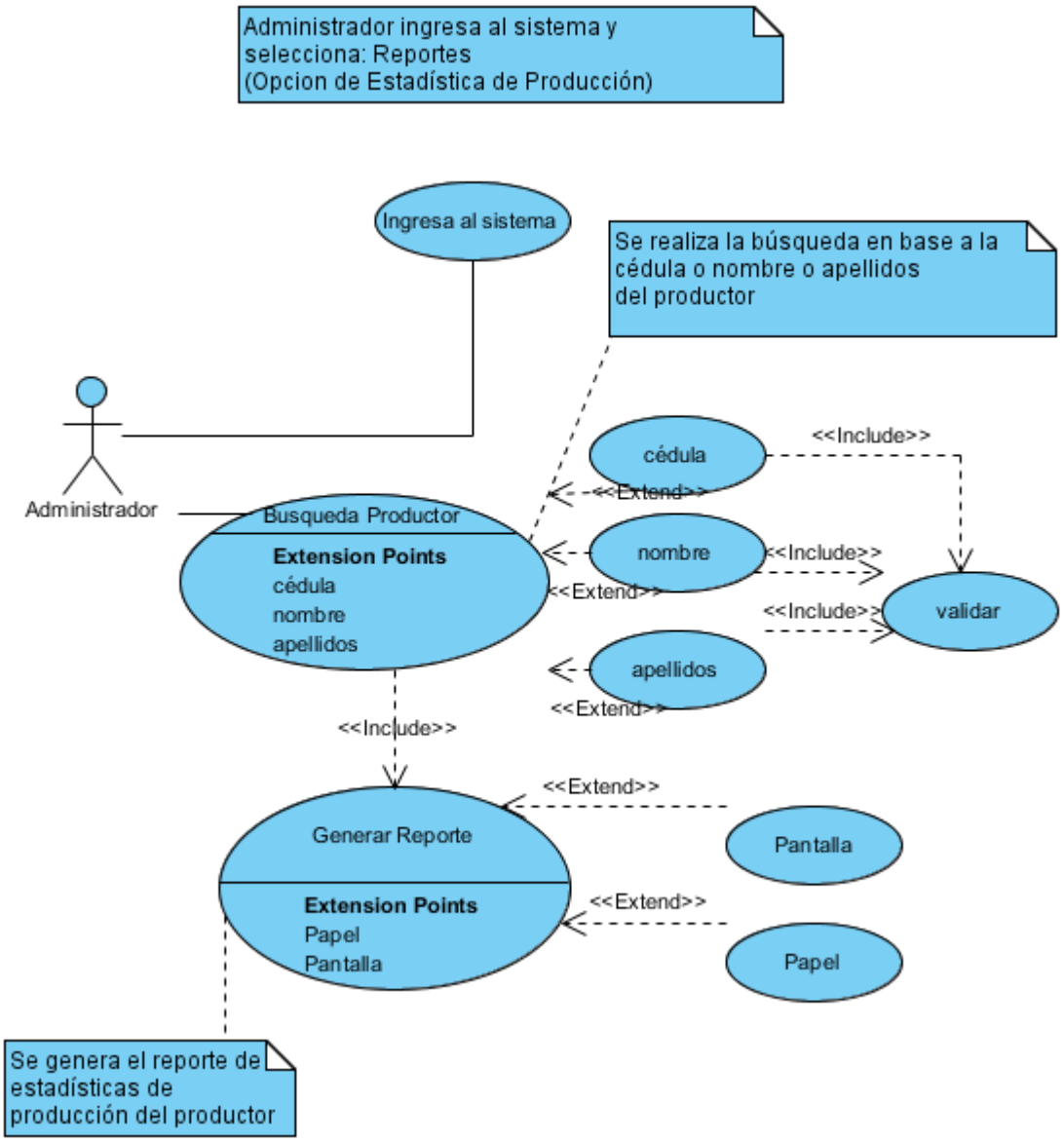
Actores:	Certificador
Propósito:	Realizar la operación de generar reportes de los informes de inspección de cada cultivo
Visión General:	El certificador es la persona encargada de generar los reportes de inspección de cada cultivo para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para generar
	reportes
3. Escoge la opción Reportes	4. Muestra las opciones de : Informe Inspección, estadística de producción, ingreso económico, producción, ingreso productores
5. Escoge la opción de Informe Inspección	6. Generar y mostrar reporte estadístico de producción
CURSOS ALTERNATIVOS	



Como administrador de ERPE, deseo visualizar los reportes estadísticos de producción

Identificador de Caso de Uso	Reporte Estadístico Producción
Caso de Uso:	CU_Reporte_Estadistico_Producción
Actores:	Administrador
Propósito:	Realizar la operación de generar reportes estadísticos de producción de un productor
Visión General:	El administrador es la persona encargada de generar los reportes estadísticos de producción de un productor para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**

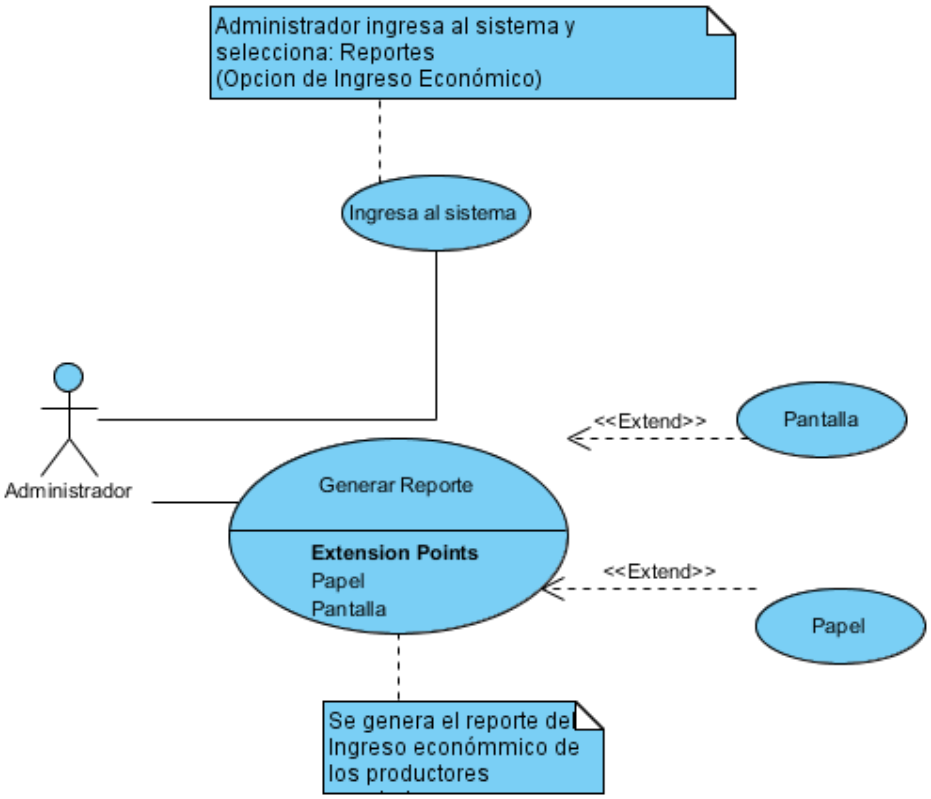
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para generar reportes
3. Escoge la opción Reportes	4. Muestra las opciones de : Informe Inspección, estadística de producción, ingreso económico, producción, ingreso productores, Agroforestería, Producción Orgánica
5. Escoge la opción de Estadística de Producción	6. Presentar la interfaz para buscar Productor
7. Ingresar la cédula o nombre o apellido del productor deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el productor	10. Generar y mostrar reporte estadístico de producción
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el productor se muestra un mensaje de que el productor no existe	



Como administrador de ERPE, deseo visualizar el ingreso económico de cada productor

Identificador de Caso de Uso	Reporte Ingreso Económico Productor
Caso de Uso:	CU_Reporte_Ingreso_Económico_Productor
Actores:	Administrador

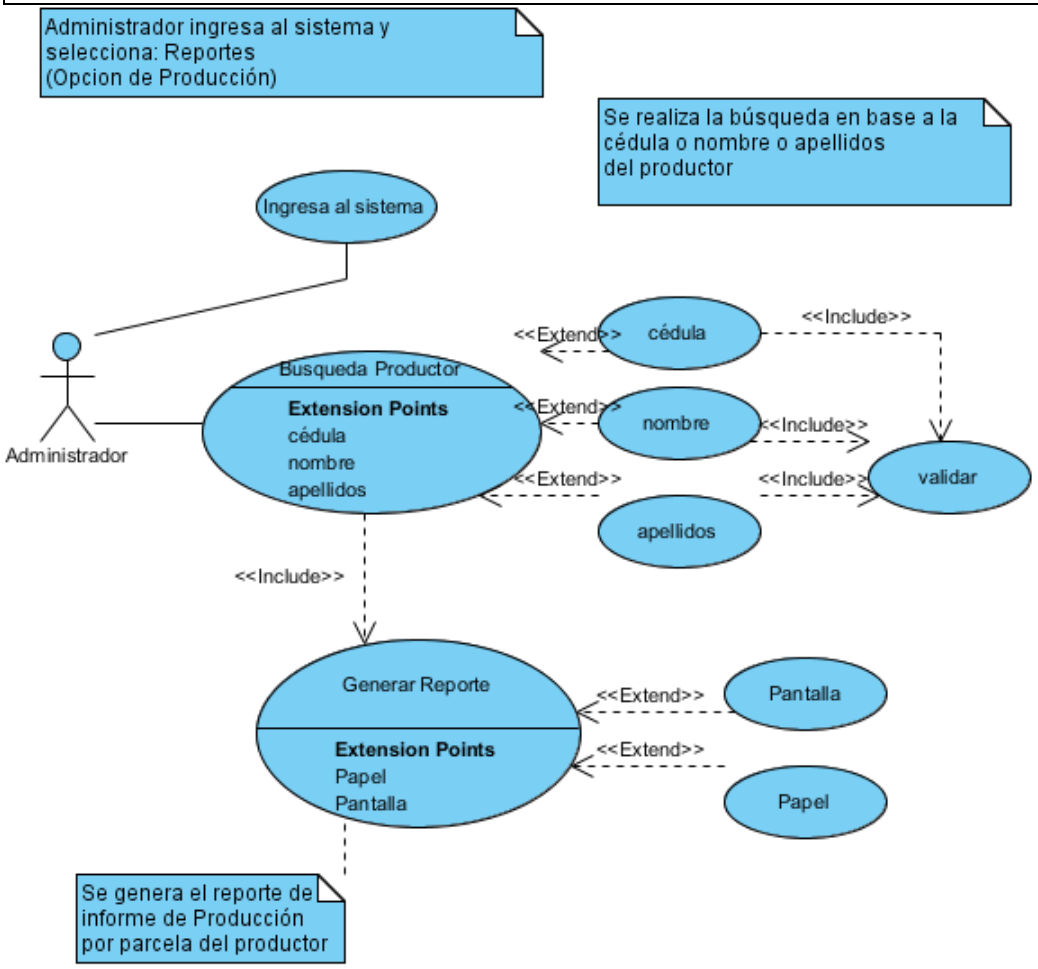
Propósito:	Realizar la operación de generar reportes de los ingresos económicos de los productores asociados a la organización
Visión General:	El administrador es la persona encargada de generar los reportes de ingresos económicos de cada productor para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para generar reportes
3. Escoge la opción Reportes	4. Muestra las opciones de : Informe Inspección, estadística de producción, ingreso económico, producción, ingreso productores, Agroforestería, Producción Orgánica
5. Escoge la opción de Ingreso económico	6. Generar y mostrar reporte de los ingresos económicos de los productores asociados
CURSOS ALTERNATIVOS	



Como administrador, inspector, deseo visualizar los informes de producción por parcela.

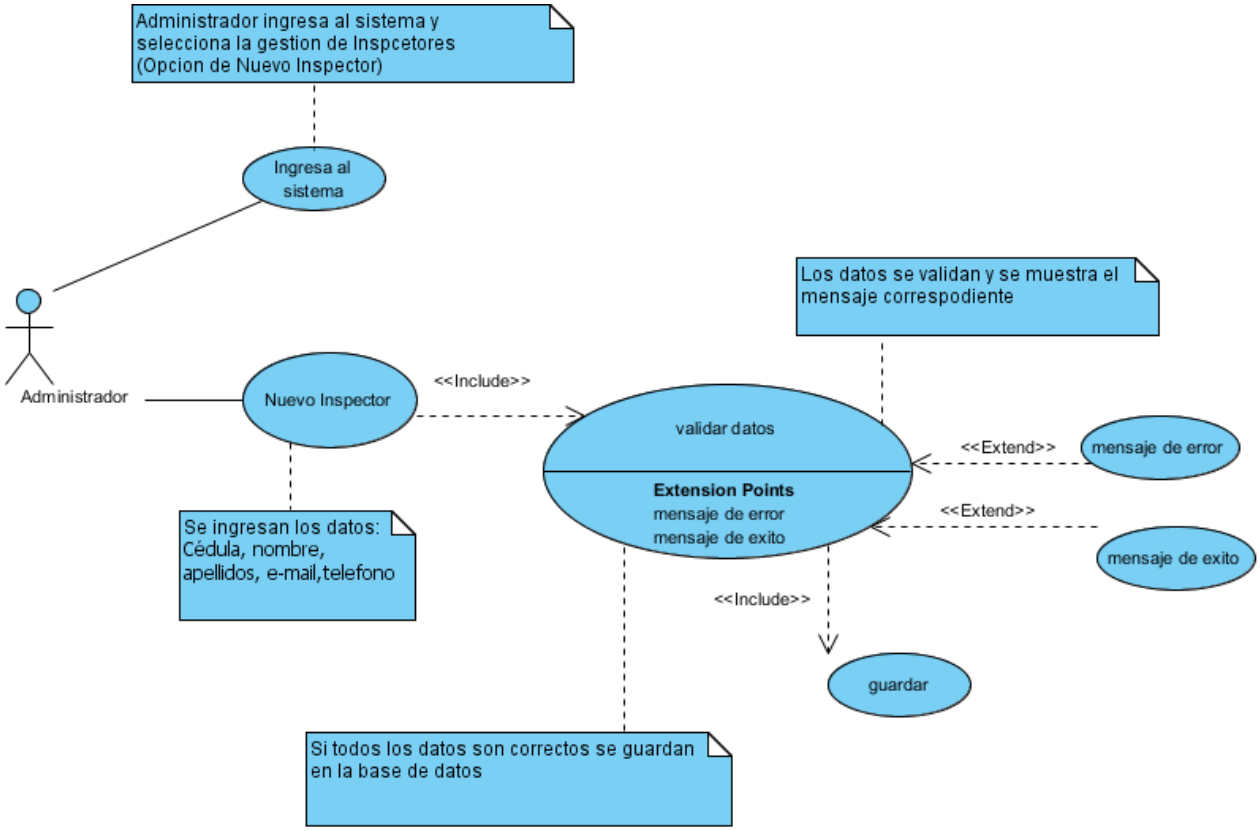
Identificador de Caso de Uso	Reporte Informe Producción Parcela
Caso de Uso:	CU_Reporte_Informe_Producción_Parcela
Actores:	Administrador
Propósito:	Realizar la operación de generar reportes estadísticos de producción de un productor
Visión General:	El administrador es la persona encargada de generar los reportes de producción por parcela de un productor para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**

1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para generar reportes
3. Escoge la opción Reportes	4. Muestra las opciones de : Informe Inspección, estadística de producción, ingreso económico, producción, ingreso productores, Agroforestería, Producción Orgánica
5. Escoge la opción de Producción	6. Presentar la interfaz para buscar Productor
7. Ingresar la cédula o nombre o apellido del productor deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el productor	10. Generar y mostrar reporte del informe de producción pro parcela
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el productor se muestra un mensaje de que el productor no existe	



Como productor asociado, deseo visualizar mi ingreso económico.

Identificador de Caso de Uso	Reporte Ingreso Económico Productor
Caso de Uso:	CU_Reporte_Informe_Producción_Parcela
Actores:	Administrador
Propósito:	Realizar la operación de generar el reporte económico de un productor
Visión General:	El productor asociado es la persona encargada de generar su informe de ingresos económicos para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para generar reportes
3. Escoge la opción Reportes	4. Muestra las opciones de : Informe Inspección, estadística de producción, ingreso económico, producción, ingreso productores, Agroforestería, Producción Orgánica
5. Escoge la opción de Ingreso Productores	6. Presentar la interfaz para buscar Productor
7. Ingresar la cédula o nombre o apellido del productor deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el productor	10. Generar y mostrar reporte del ingreso económico del productor
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el productor se muestra un mensaje de que el productor no existe	

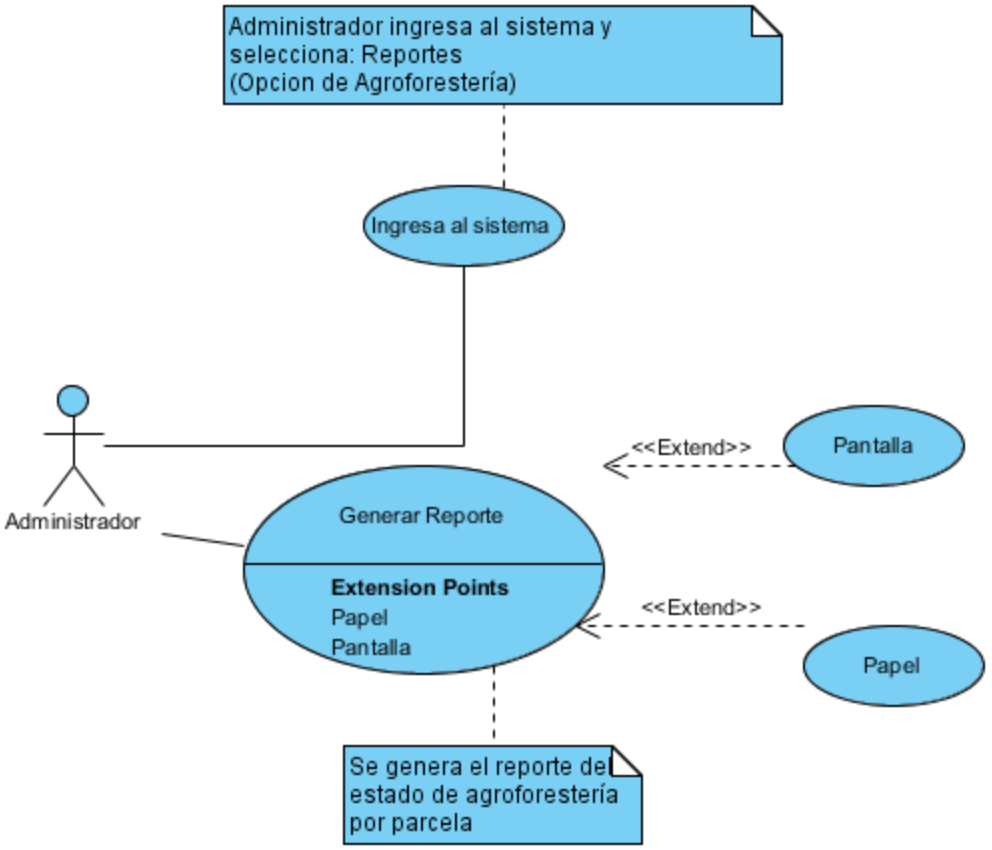


SPRINT 3

Como Inspector, administrador de ERPE, deseo visualizar el estado de agroforestería de cada parcela.

Identificador de Caso de Uso	Reporte Estado Agroforestería
Caso de Uso:	CU_Reporte_Estado_Agroforestería
Actores:	Inspector, administrador
Propósito:	Realizar la operación de generar reportes del estado de agroforestería de cada parcela
Visión General:	El inspector es la persona encargada de generar los reportes del estado de agroforestería de cada parcela para llevar un control de los mismos.
Tipo:	Primario y Esencial.

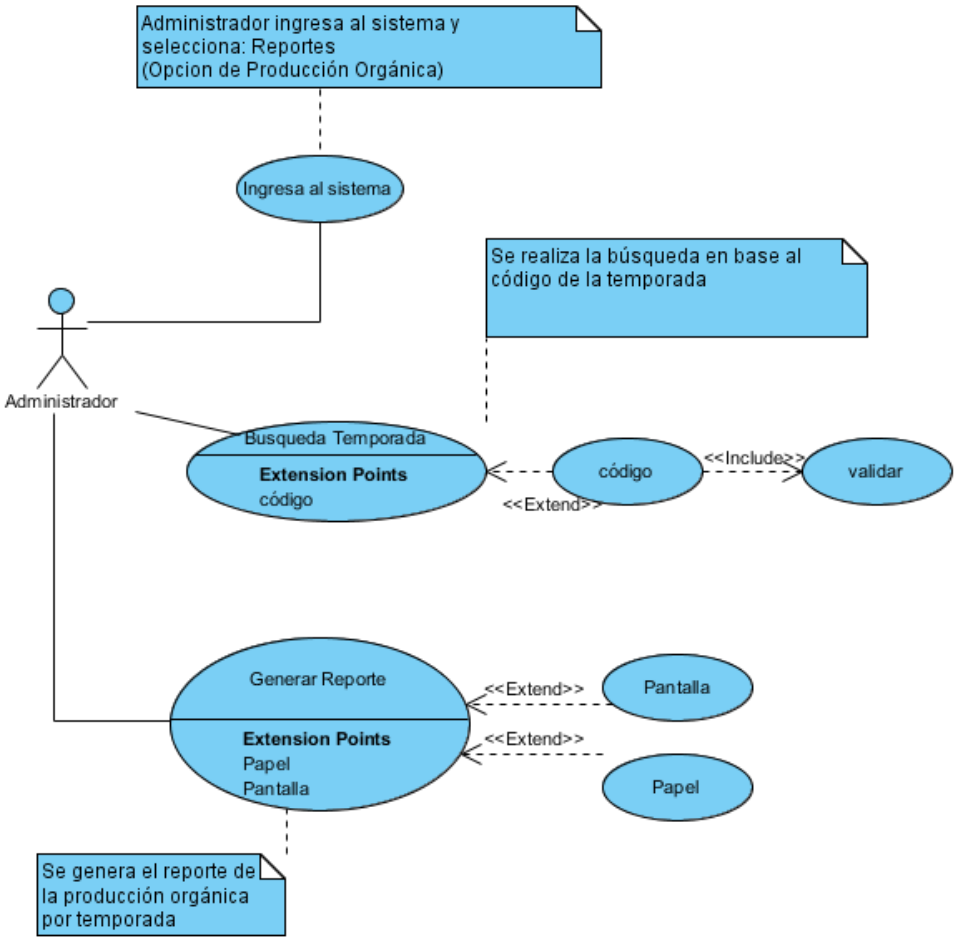
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para generar reportes
3. Escoge la opción Reportes	4. Muestra las opciones de : Informe Inspección, estadística de producción, ingreso económico, producción, ingreso productores, Agroforestería, Producción Orgánica
5. Escoge la opción de Agroforestería	6. Generar y mostrar reporte del estado de agroforestería de cada parcela
CURSOS ALTERNATIVOS	



Como administrador, certificador, deseo visualizar los reportes de producción orgánica.

Identificador de Caso de Uso	Reporte Producción Orgánica
Caso de Uso:	CU_Reporte_Producción_Orgpanica
Actores:	Administrador, certificador
Propósito:	Realizar la operación de generar el reporte de producción orgánica de un productor en una temporada
Visión General:	El administrador es la persona encargada de generar los reportes de producción orgánica en una temporada para llevar un control de los mismos.
Tipo:	Primario y Esencial.
CURSOS TIPICOS DE EVENTOS	
ACCION DEL ACTOR	**RESPUESTA DEL SISTEMA**

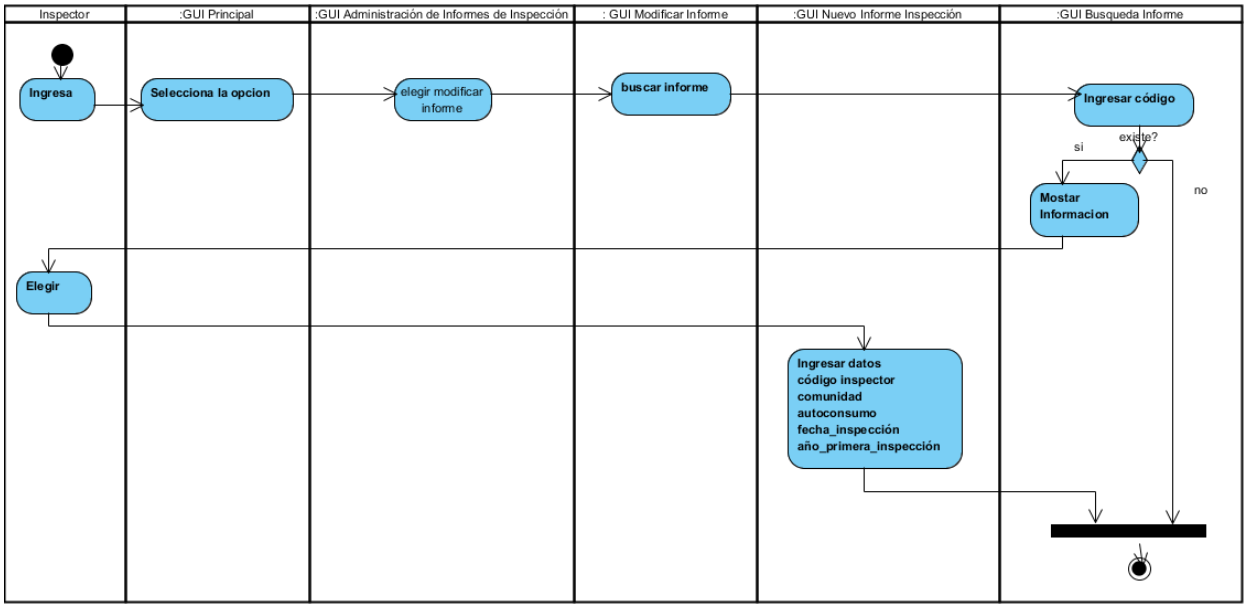
1. Accede a la Aplicación.	2. El sistema Presenta un menú de opciones para generar reportes
3. Escoge la opción Reportes	4. Muestra las opciones de : Informe Inspección, estadística de producción, ingreso económico, producción, ingreso productores, Agroforestería, Producción Orgánica
5. Escoge la opción de Producción Orgánica	6. Presentar la interfaz para buscar Productor
7. Ingresar la cédula o nombre o apellido del productor deseado	8. Buscar en la base de datos y mostrar
9. Seleccionar el productor	10. Generar y mostrar reporte del producción orgánica del productor
CURSOS ALTERNATIVOS	
Línea 4: Si no se encuentra el productor se muestra un mensaje de que el productor no existe	


Administrador ingresa al sistema y selecciona: Reportes (Opcion de Producción Orgánica)
Ingresa al sistema
Se realiza la búsqueda en base al código de la temporada
Administrador
Busqueda Temporada
Extension Points
código
código
<<Include>>
validar
<<Extend>>
Generar Reporte
<<Extend>>
Pantalla
Extension Points
Papel
Pantalla
<<Extend>>
Papel
Se genera el reporte de la producción orgánica por temporada

Anexo 14: Diagrama de Secuencia

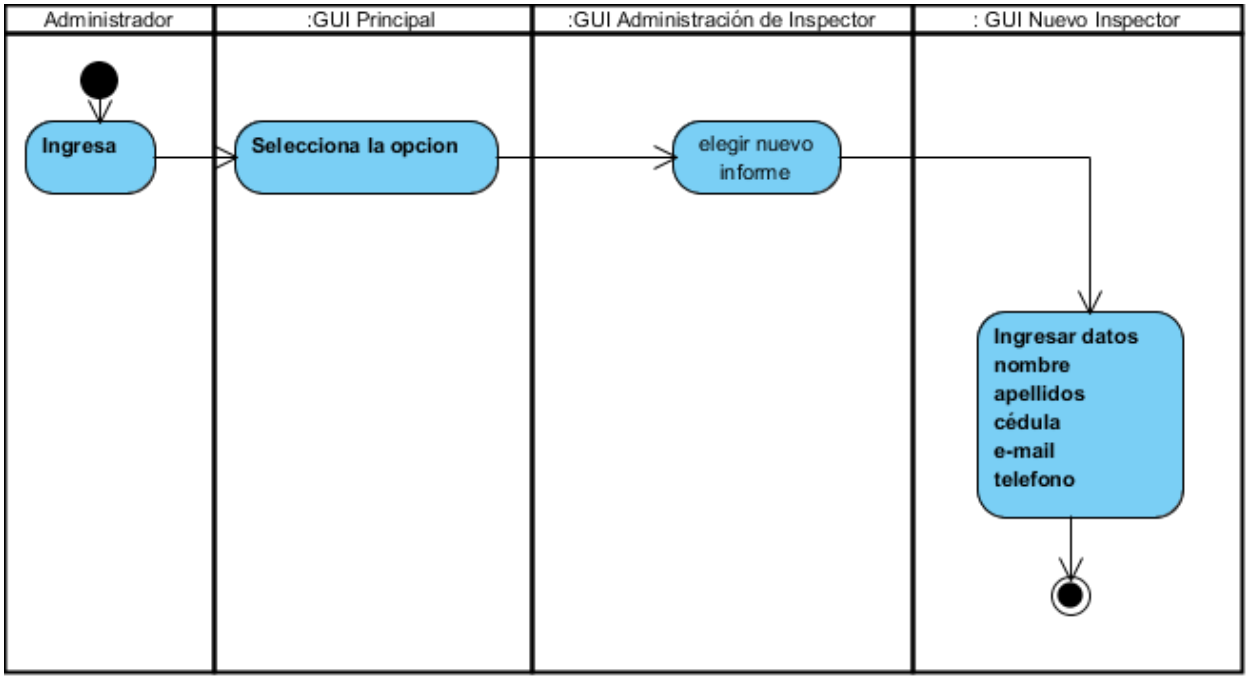
SPRINT 1

Modificación de Informe de inspección

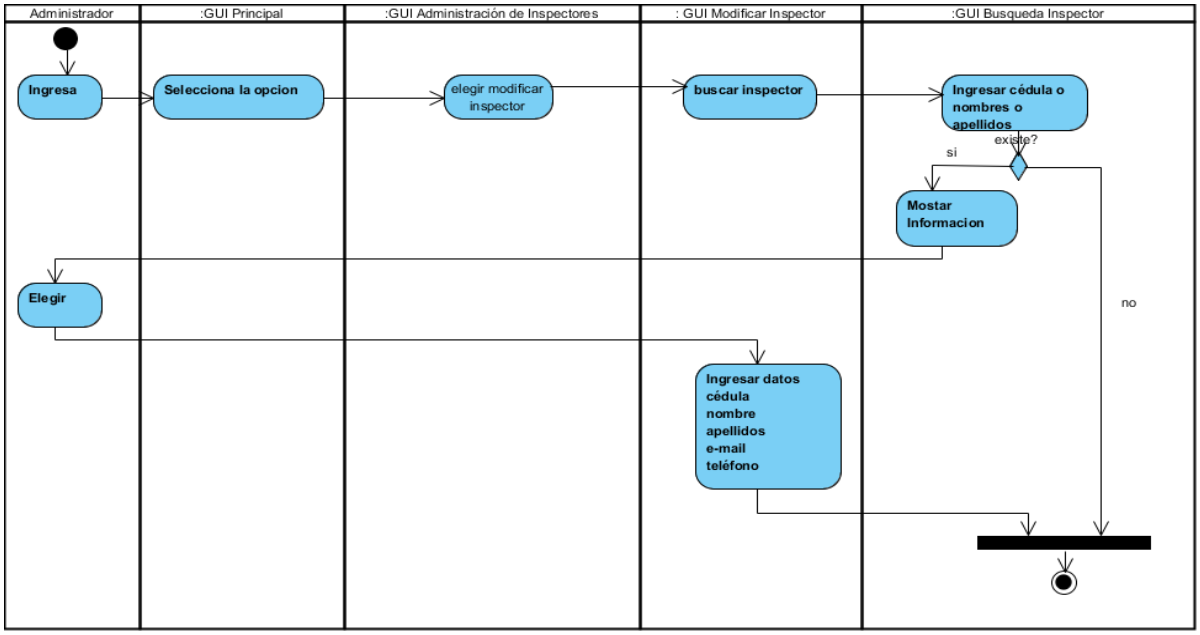


SPRINT 2

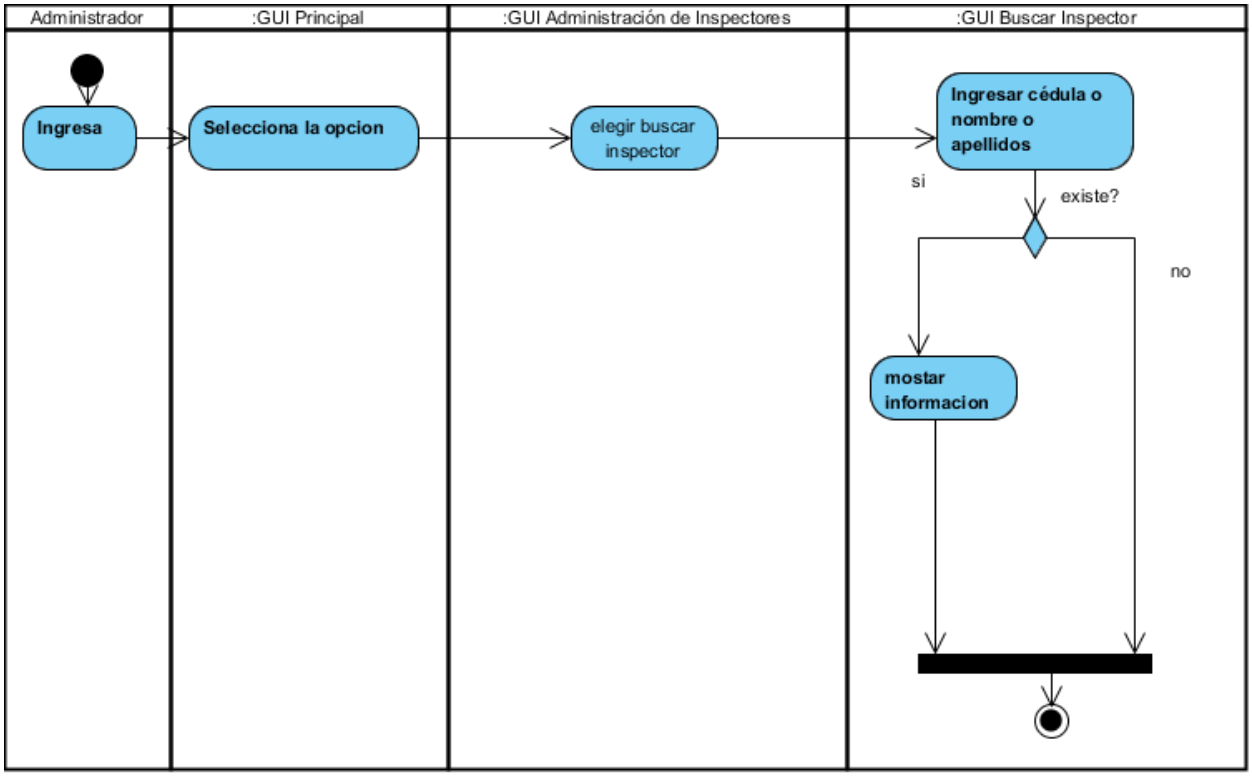
Ingreso de inspector



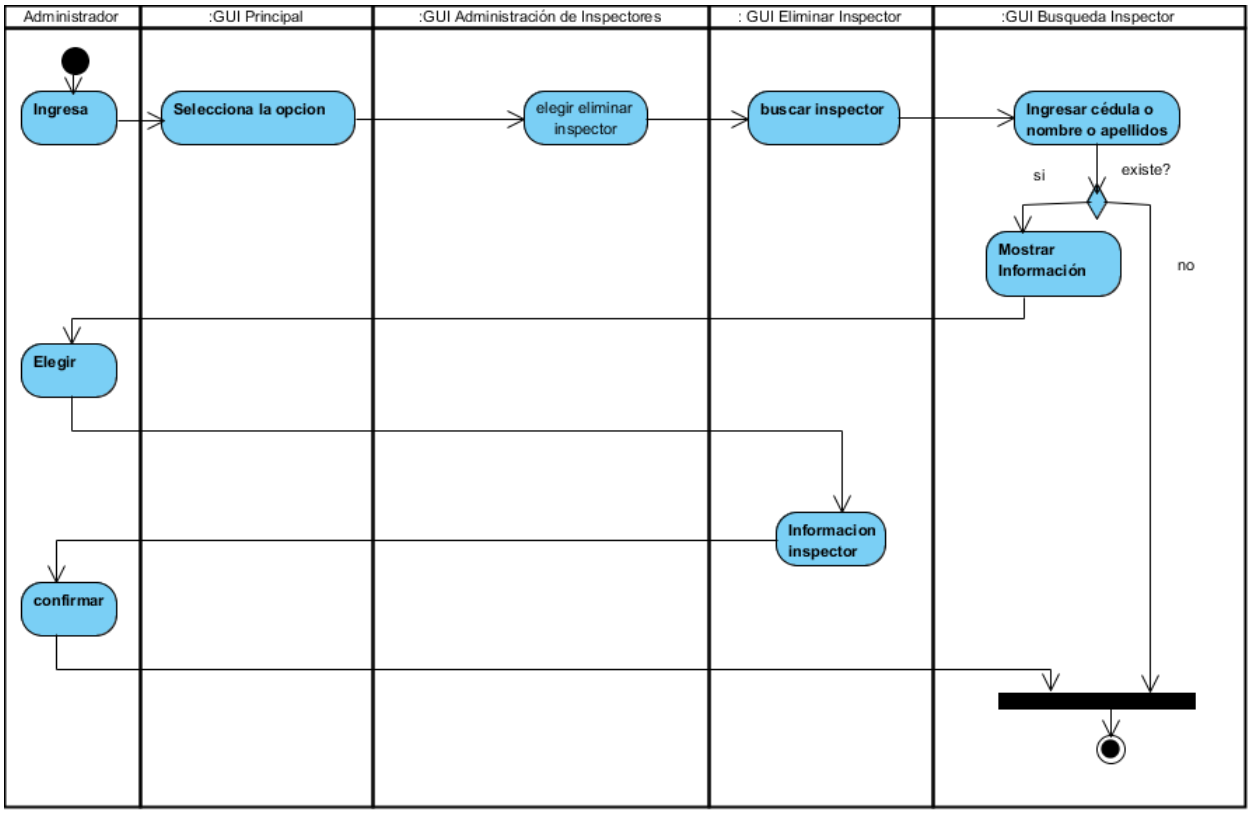
Modificación de un inspector



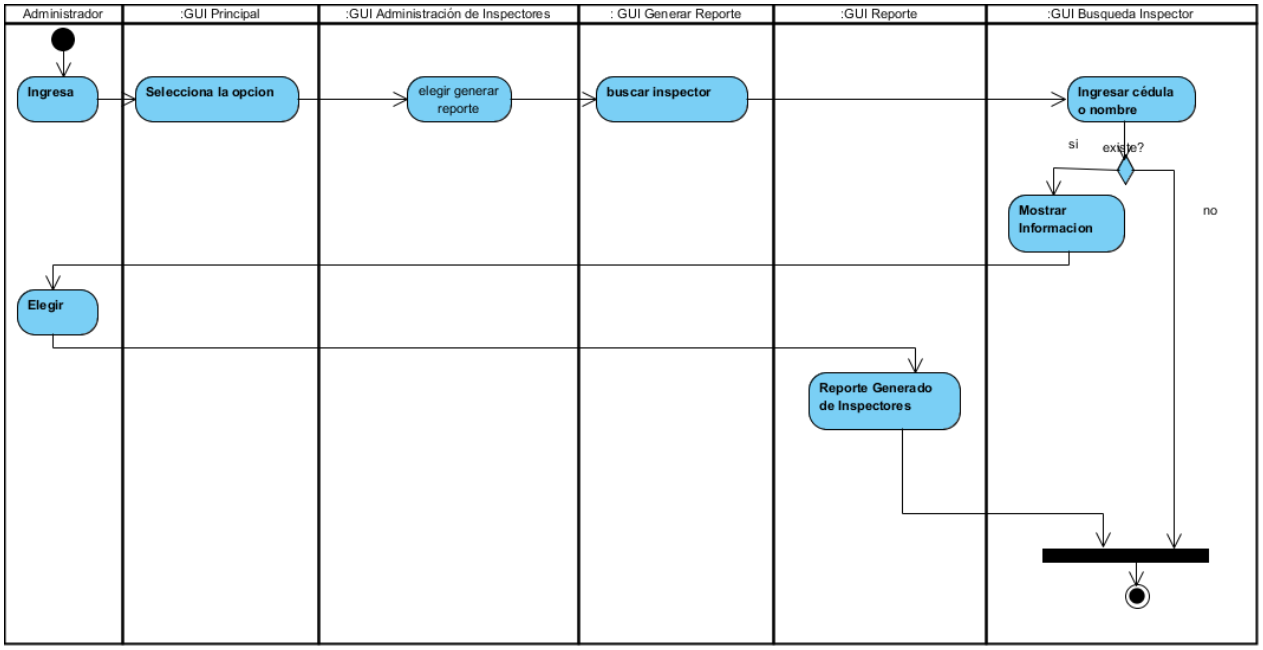
Búsqueda de inspector



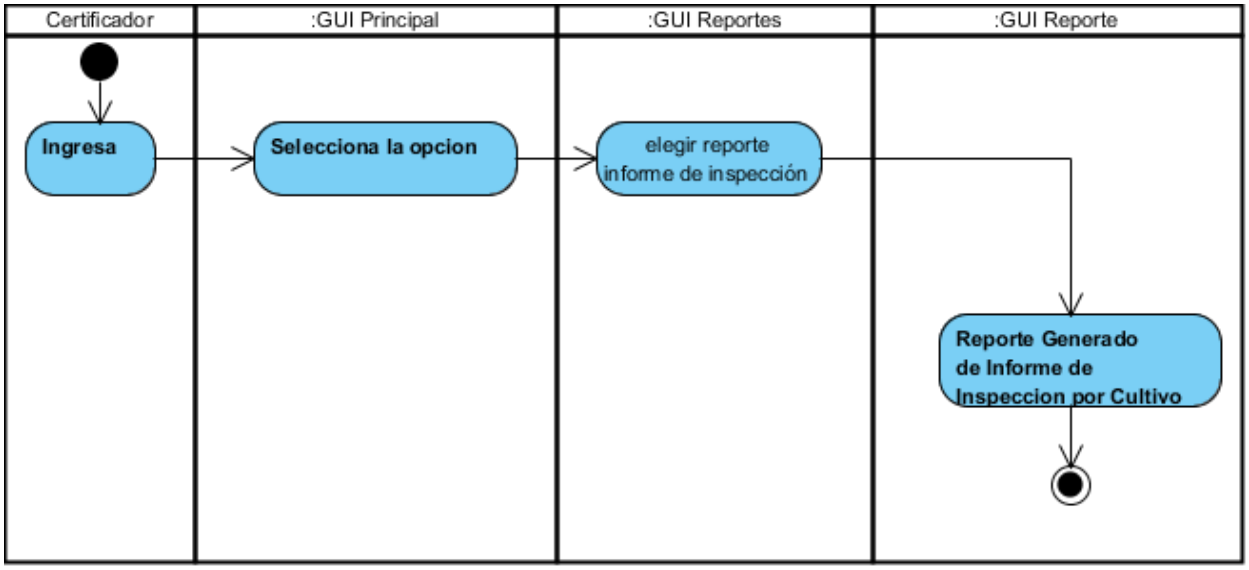
Eliminación de inspector



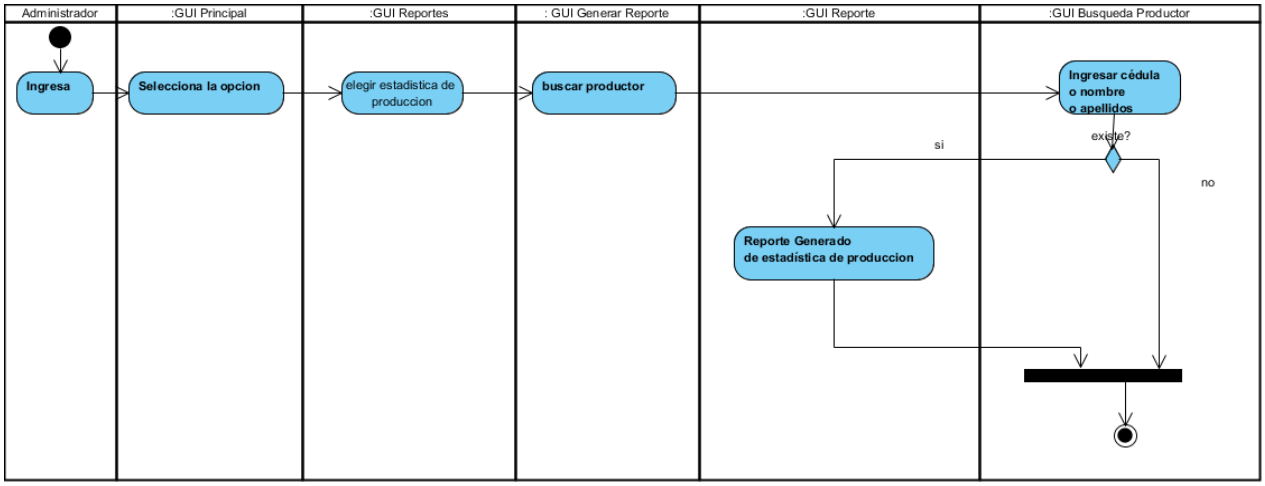
Reporte de inspectores



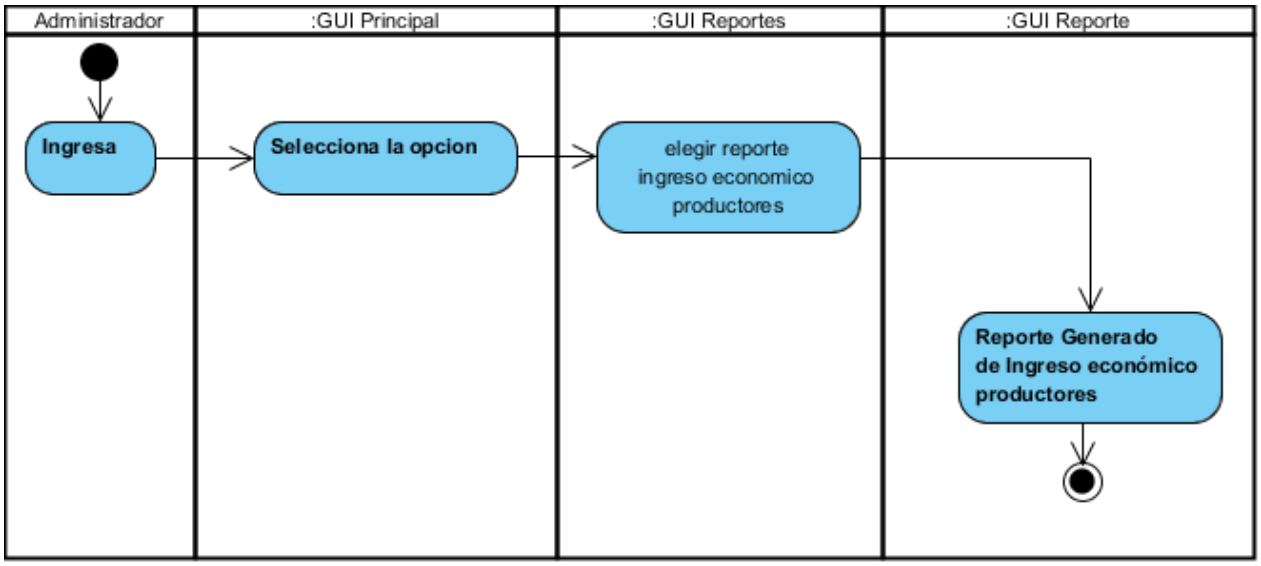
Reporte de Informe de Inspección por cultivo



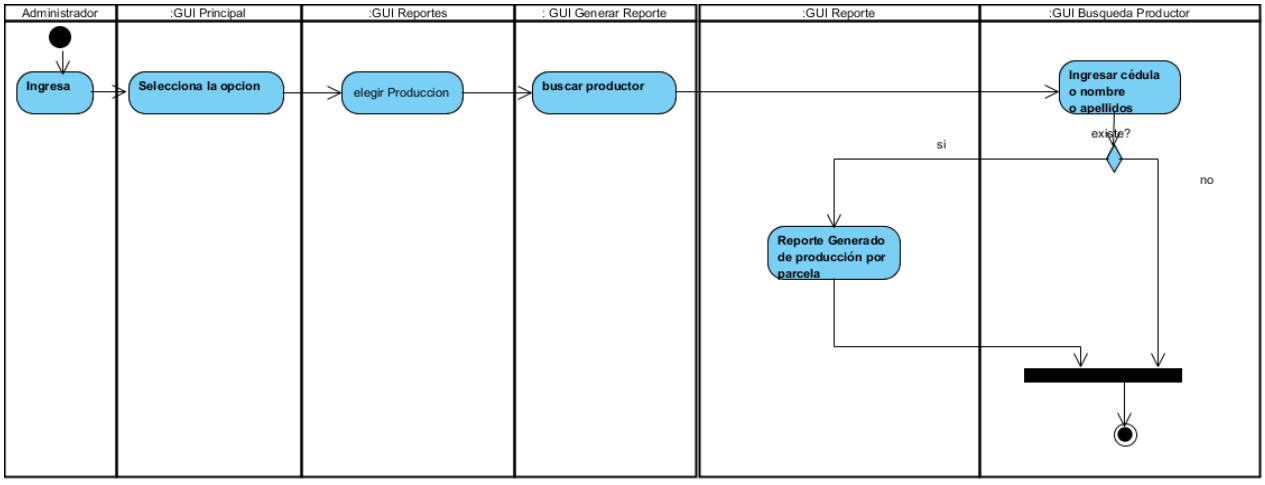
Reporte Estadístico de Producción



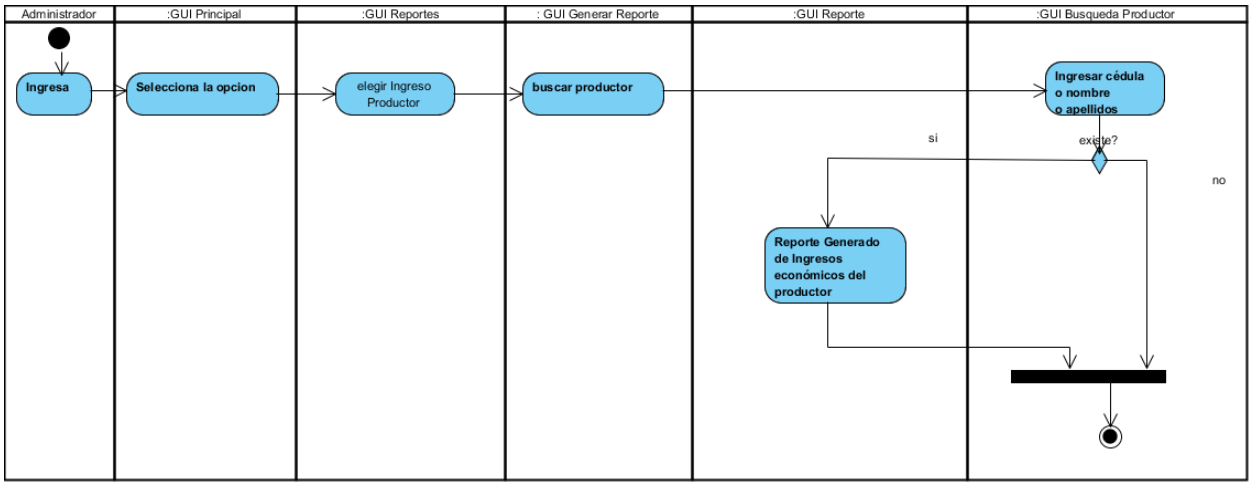
Reporte de Ingreso Económico de cada productor



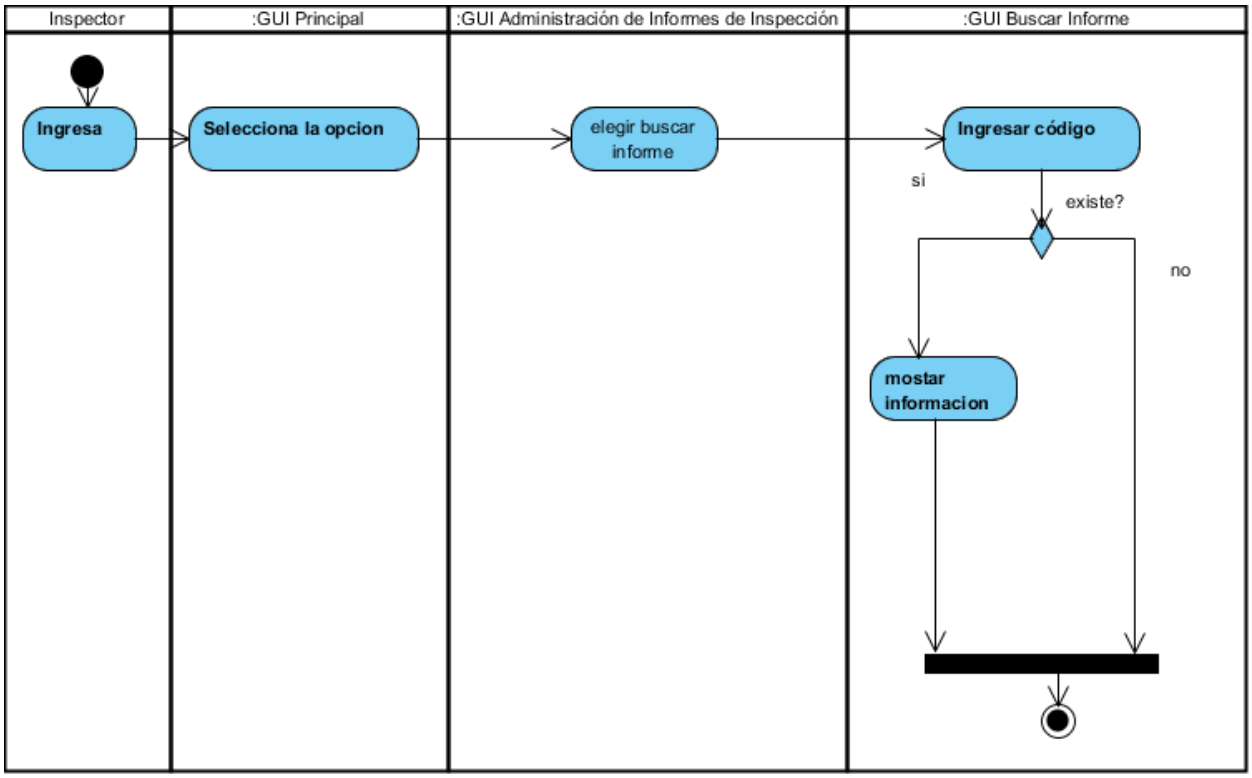
Reporte Informe de Producción por parcela



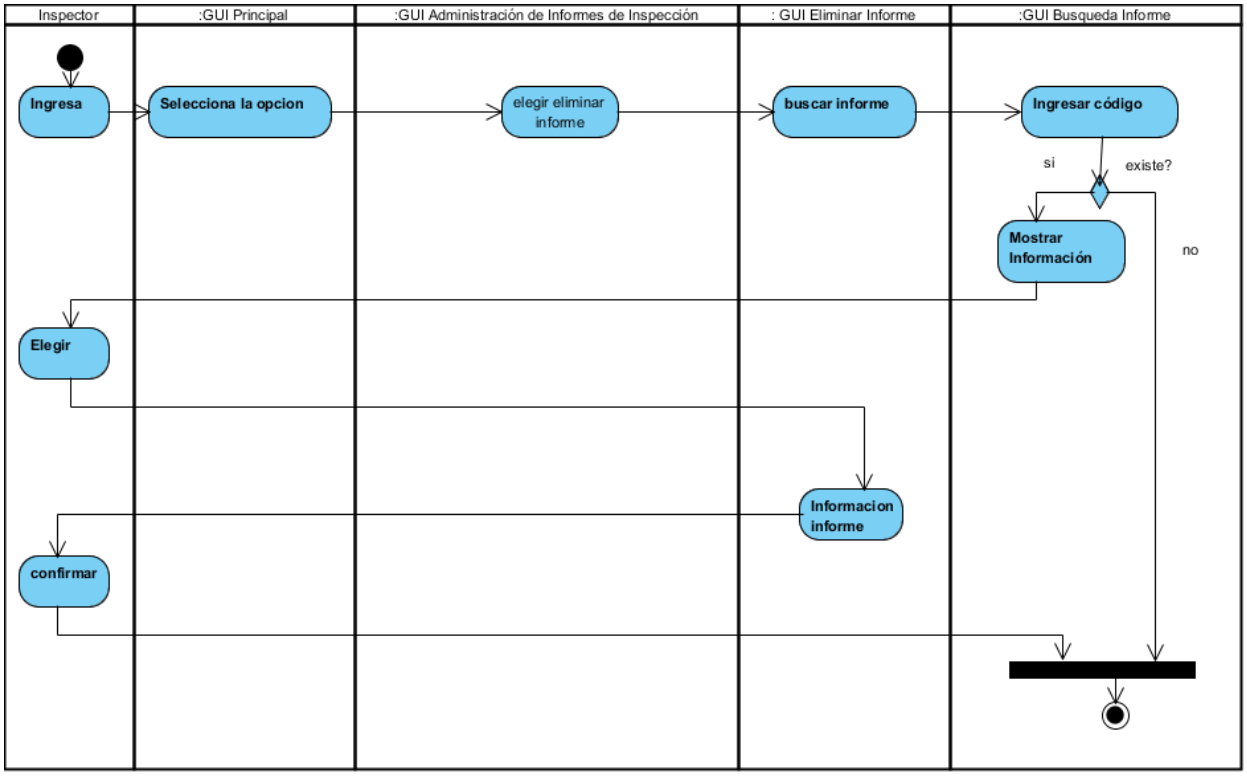
Reporte de Ingreso Económico Productor



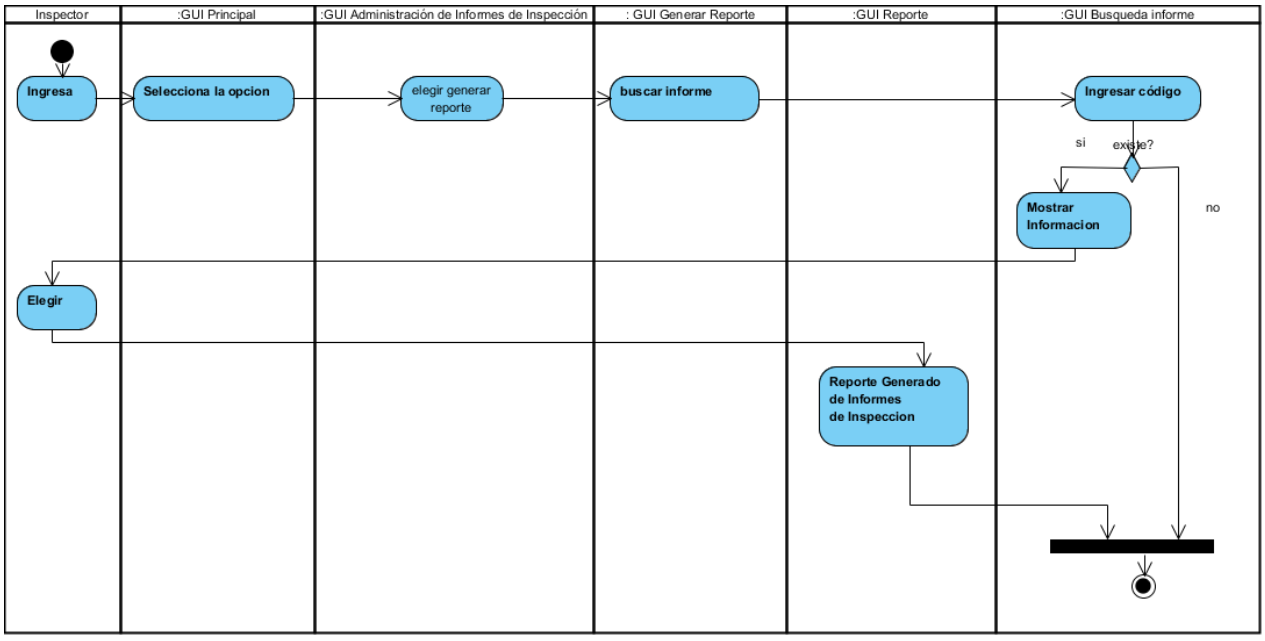
Búsqueda de Informe de inspección



Eliminación de Informe de inspección

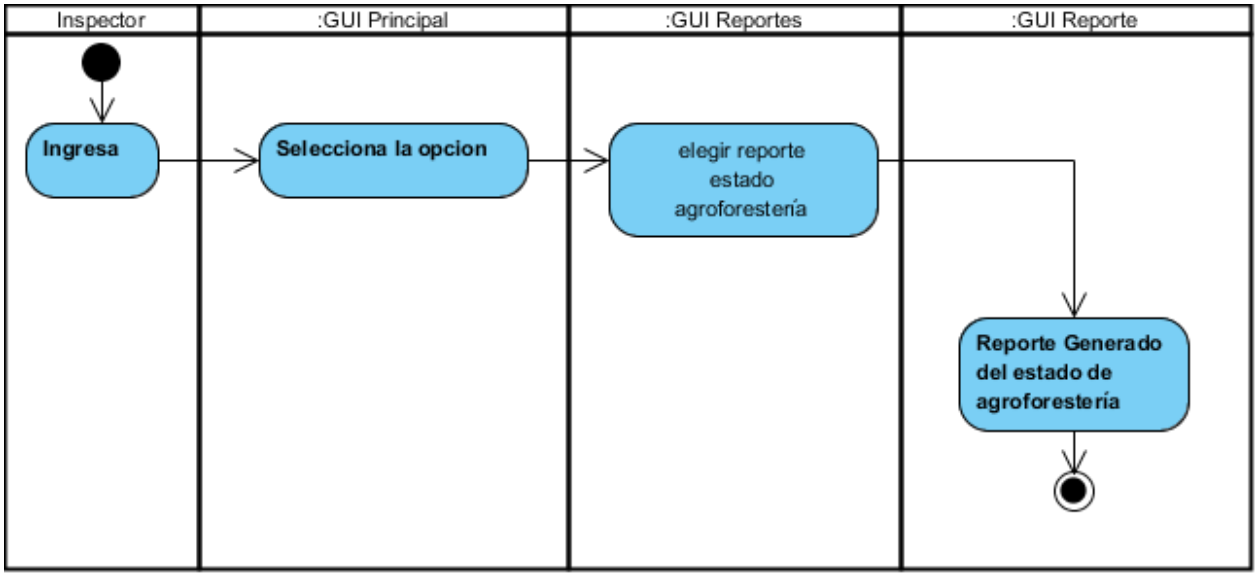


Reporte de Informe de inspección

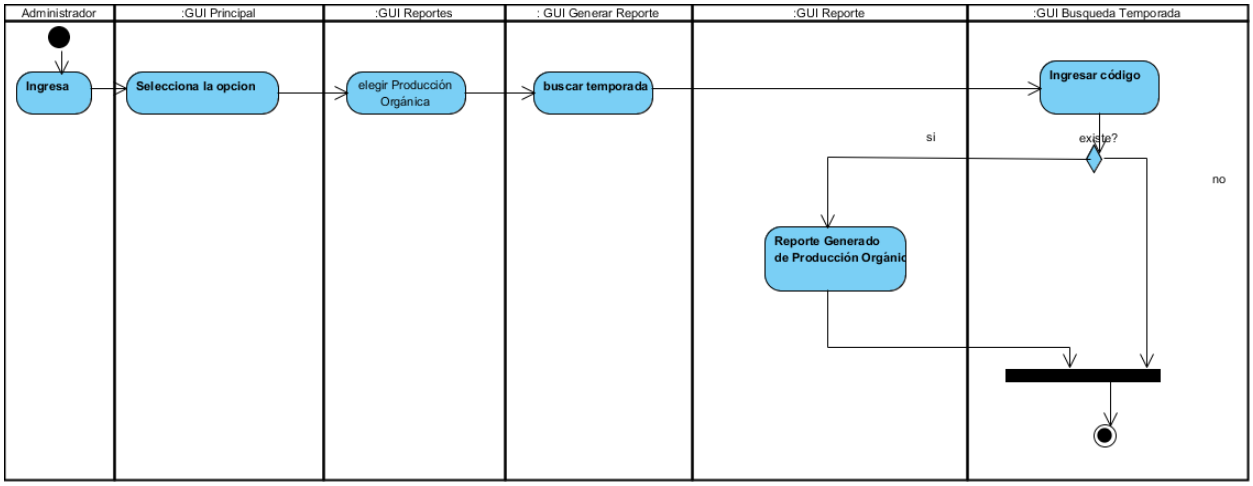


SPRINT 3

Reporte de Estado de Agroforestería



Reporte de Producción Orgánica



Anexo 15: Manual Técnico

Este manual viene adjunto en el CD

Anexo 16: Manual de Usuario

Este manual viene adjunto en el CD, organizados por Sprints.