Sistema de Gestión de indicadores de responsabilidad ambiental Restaurante Rancho Eden

Software Architecture Document

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 10/10/16 | 1.0 | Inicio del documento. | Andrés Felipe Buitrago  Erick Vargas Hernandez |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. Introduction 5](#_Toc430732337)

[1.1 Purpose 5](#_Toc430732338)

[1.2 Scope 5](#_Toc430732339)

[1.3 Definitions, Acronyms, and Abbreviations 5](#_Toc430732340)

[1.4 References 5](#_Toc430732341)

[1.5 Overview 5](#_Toc430732342)

[2. Architectural Representation 5](#_Toc430732343)

[3. Architectural Goals and Constraint 6](#_Toc430732344)

[3.1 Reservar vehículos 6](#_Toc430732345)

[3.2 Gestión de horarios 6](#_Toc430732346)

[3.3 Generar reportes 6](#_Toc430732347)

[4. Use-Case View 6](#_Toc430732348)

[5. Logical View 7](#_Toc430732349)

[5.1 Overview 7](#_Toc430732350)

[5.2 Architecturally Significant Design Packages 9](#_Toc430732351)

[5.3 Use-Case Realizations 19](#_Toc430732352)

[6. Deployment View 27](#_Toc430732353)

[7. Implementation View 27](#_Toc430732354)

[7.1 Overview 27](#_Toc430732355)

[7.2 Layers 28](#_Toc430732356)

[8. Size and Performance 30](#_Toc430732357)

[9. Quality 30](#_Toc430732358)

Software Architecture Document

# Introduction

## Purpose

Este documento provee un panorama de la arquitectura del sistema, usando un número de diferentes vistas de arquitectura para presentar diferentes aspectos del sistema. Intenta capturar y transmitir las decisiones arquitectónicas que se han hecho en el sistema.

## Scope

El alcance de este documento está relacionado con el proyecto de gestión de indicadores ambientales Restaurante Rancho Edén, nos brinda la especificación de la estructura general del producto a desarrollar, los elementos que lo conforman, las relaciones entre elementos y sus propiedades.

## Definitions, Acronyms, and Abbreviations

CRUD: Crear, leer, actualizar y eliminar (Create, Read, Update, Delete).

MVC: Modelo Vista-Controlador.  
DataForm: Metodología para modelar la interfaz de usuario.

## References

* Software Architecture in Practice de Rick Kazman, Len Bass, Paul Clements
* Patron de diseño Command

## Overview

En este documento se podrá encontrar la representación arquitectónica del proyecto, sus metas y restricciones, la descripción de cada componente que hace parte de la arquitectura y los atributos de calidad que se cumplen con esta.

# Architectural Representation

La representación de la arquitectura está basada en el modelo 4+1 vistas, la cual nos permite estructurar el proyecto en diferentes vistas: vista lógica, vista de implementación, vista de procesos, vista de despliegue y por último la vista de casos de uso.

* 1. **Vista lógica**

Esta vista tiene como elemento de modelado el diagrama de clases, el cual nos permite representar el mundo mediante entidades y relaciones.

* 1. **Vista de implementación**

Esta vista tiene como elemento de modelado el diagrama de componentes, diagrama de paquetes y nos da la visualización como puede ser el ensamblado del sistema y como este puede ser implementado.

* 1. **Vistas de despliegue**

La vista de despliegue tiene como elemento de modelado el diagrama de despliegue y el diagrama de paquetes, para poder comunicar cual será la topología del sistema, su distribución y además como se realizara la entrega y su instalación.

* 1. **Vista de caso de uso**

Esta vista tiene como elemento de modelados el diagrama de casos de uso, diagrama de actividades y nos permite identificar la labor y comportamiento de la organización.

# Architectural Goals and Constraint

## Establecer indicadores a ejecutar en el restaurante

Se tendrá la posibilidad de establecer indicadores de responsabilidad ambiental, agregando su estado y descripción.

## Gestión de indicadores

La gestión de indicadores se realiza, para facilitar la implementación de dichos indicadores y así facilitar el proceso de llevarlos a cabo.

## Generar reportes

El sistema debe soportar cualquier tipo de reporte, donde se muestren estadísticas reales, y fáciles de entender.

* 1. El sistema debe ser permitir crecer el proyecto a futuro, debe ser capaz de soportar de otros procesos del restaurante rancho eden.
  2. **Restricciones:** Vamos a utilizar versiones free o con licencia limitada, esto nos limita a la hora de trabajar en un ambiente productivo continuo, además de limitar los servicios funcionales que tenga la aplicación.

# Use-Case View

A continuación se puede ver el diagrama de los casos de uso en el cual se ven las actividades plasmadas que representan la funcionalidad central del sistema final. En este diagrama además se puede identificar que el usuario es fundamental para realizar cada uno de los casos de uso de sistema.

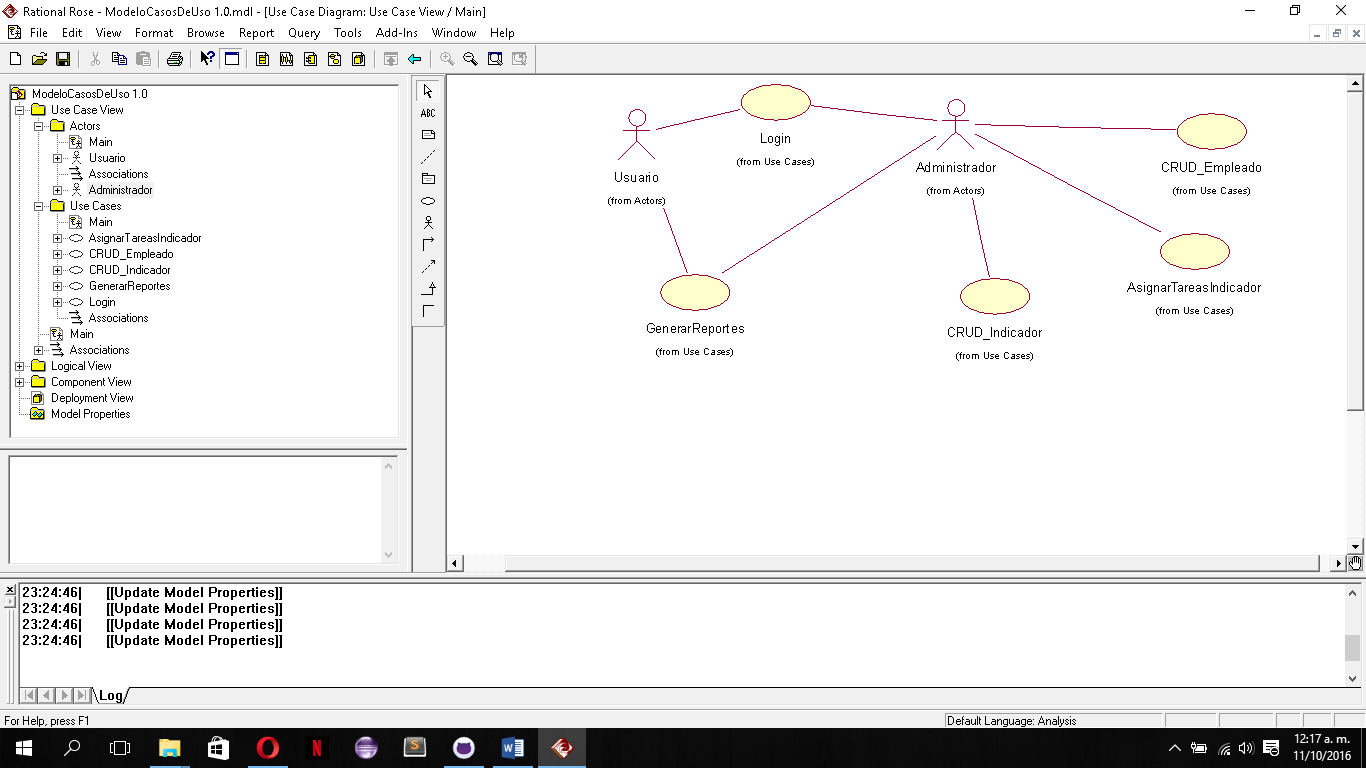


Figura 1 Diagrama de casos de uso del sistema

# Logical View

## Overview

A continuación se presentará la descomposición general de paquetes del proyecto

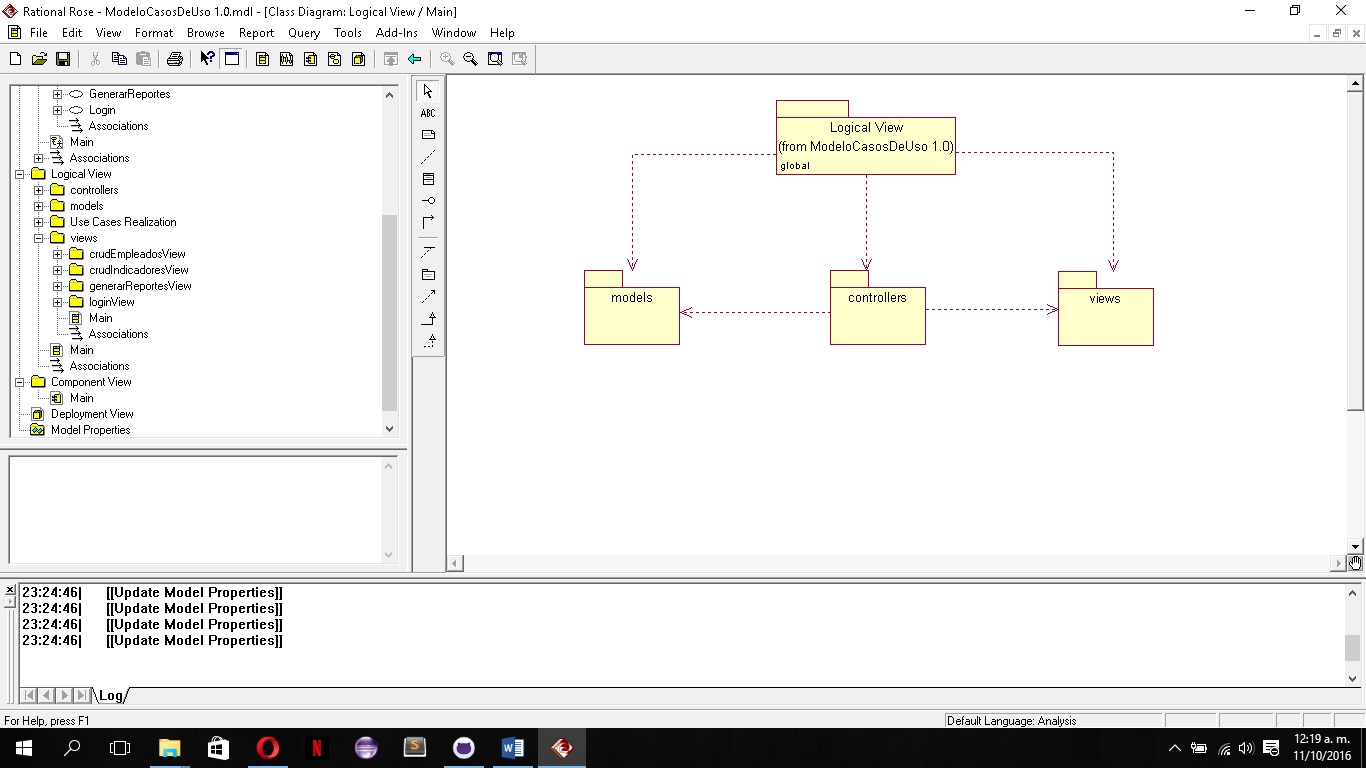


Figura Diagrama de descomposición general de paquetes

Este modelo corresponde a la vista más general del proyecto, donde se puede apreciar la estructura basada en MVC, donde en models podemos encontrar el modelo conceptual del mundo abstraído, en controllers se encuentra los controladores que ayudaran a conectar la vista con el modelo, y en views se puede encontrar los modelos de las interfaces gráficas del proyecto.

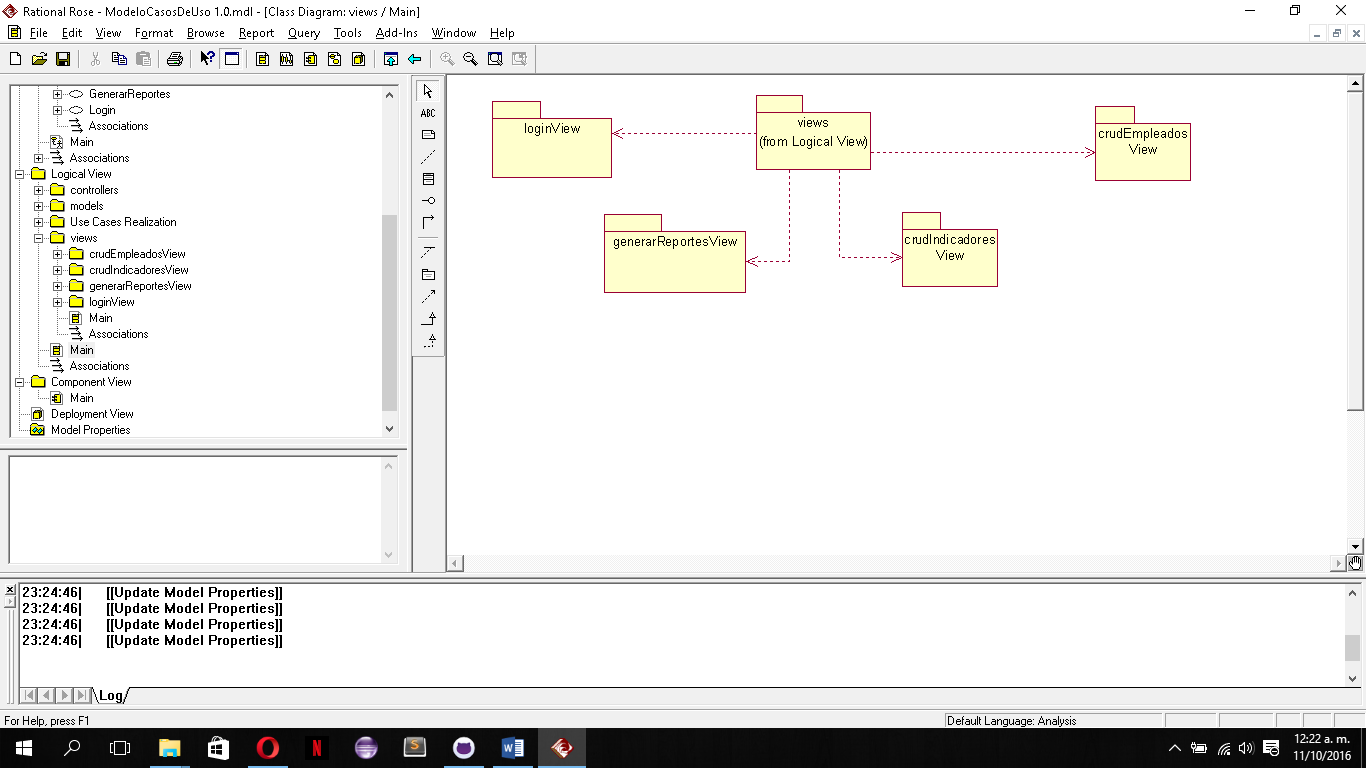


Figura Diagrama de descomposición de paquetes del componente view

Este modelo corresponde a un mirada interna del paquete views y muestra todas las vistas que hacen parte del proyecto.

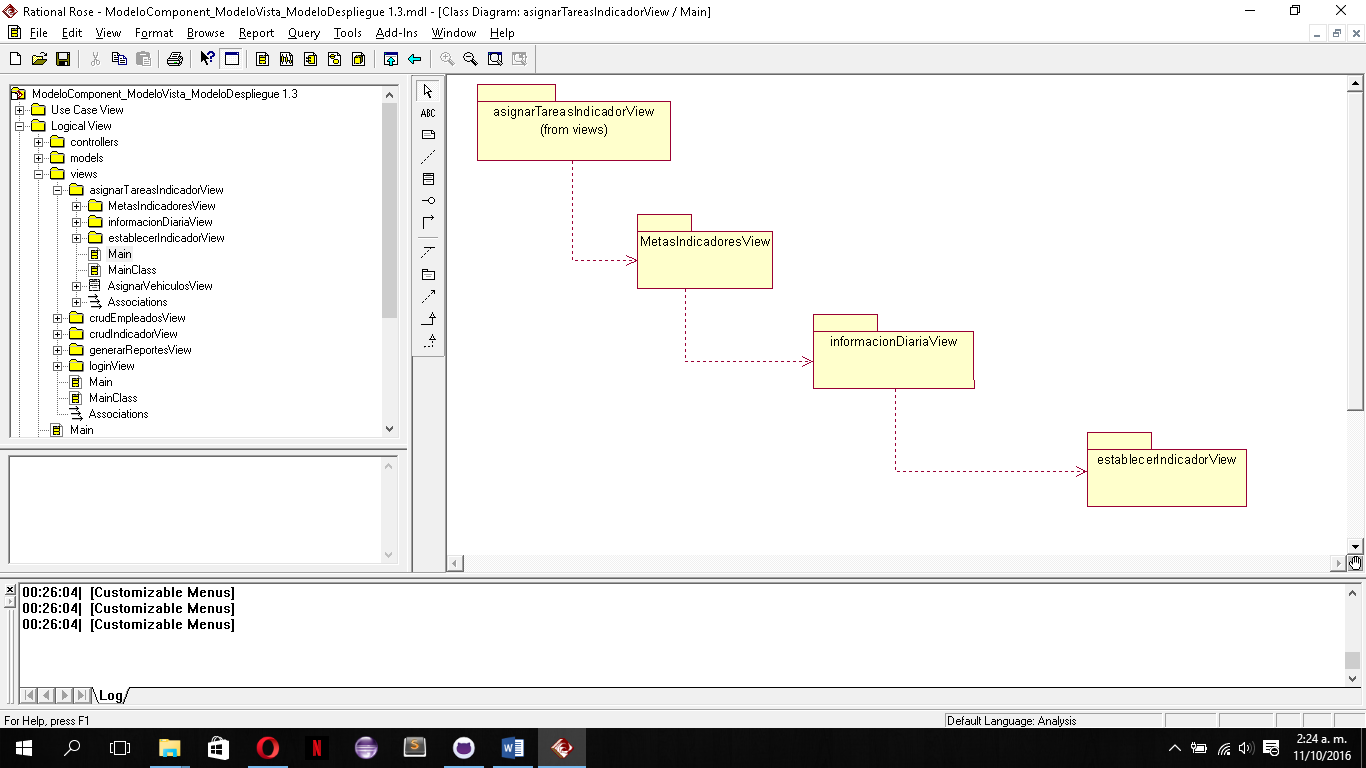


Figura Descomposición de paquetes del paquete asignarTareasIndicador view

Este modelo representa como está conformada la vista agregar vehículo internamente y que vistas hacen parte de ella.

## Architecturally Significant Design Packages

Como se había mencionado anteriormente, el sistema está basado en MVC ya que nos permite separar la capa de presentación de la lógica de negocio y los datos, el diagrama que se muestra a continuación ilustran las relaciones que hay entre el modelo, la vista y el controlador.

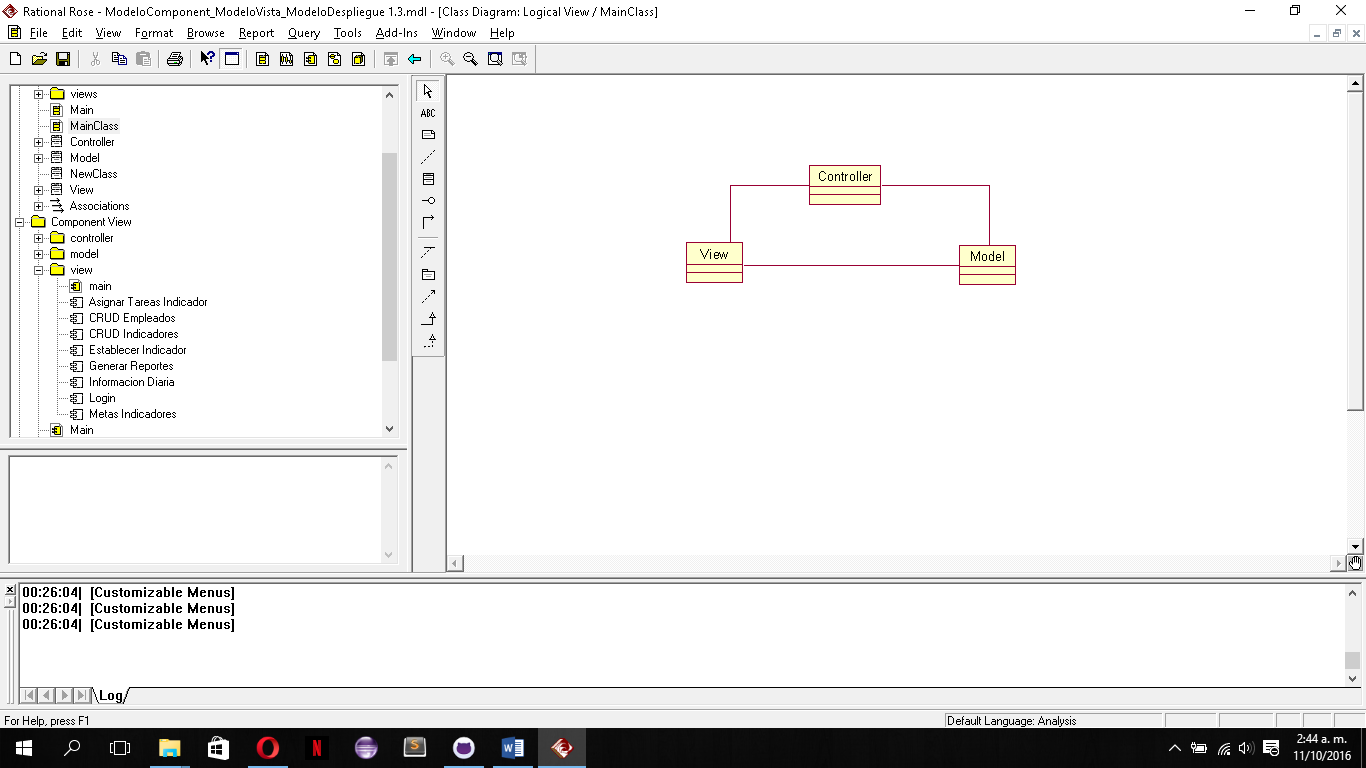


Figura Modelo de clases del patrón MVC

* + 1. **Models.**

En el siguiente diagrama se pueden observar todos los modelos que se utilizaran en el proyecto, los cuales representan las tablas en la base de datos, además estos modelos deben heredar del componente model, para poder cumplir con el patrón MVC.

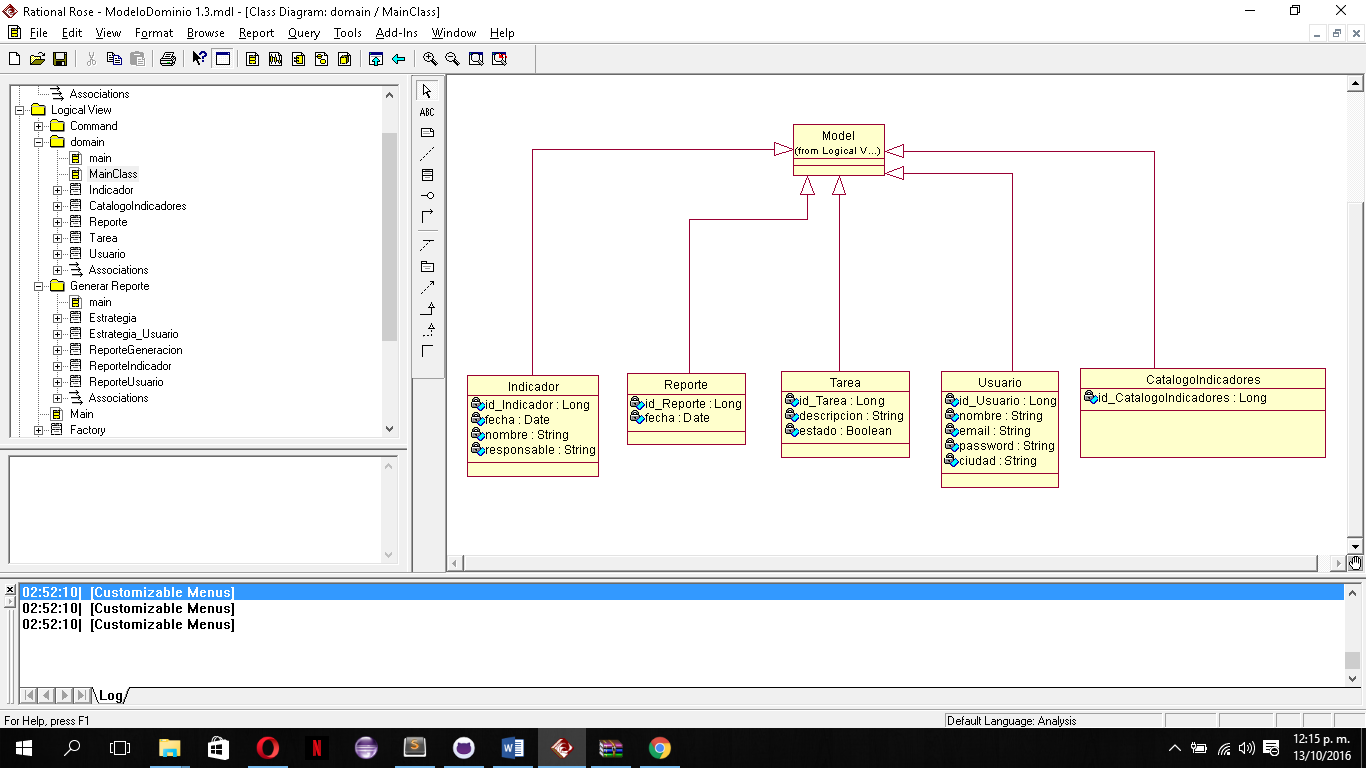


Figura Diagrama del modelo y las clases que heredan

En el diagrama de datos se puede ver que hay una entidad que recopila todos los datos, para poder tener un registro de todo lo que se ha realizado, gracias a esta clase se puede generar cualquier reporte que se encuentre en la base de datos.

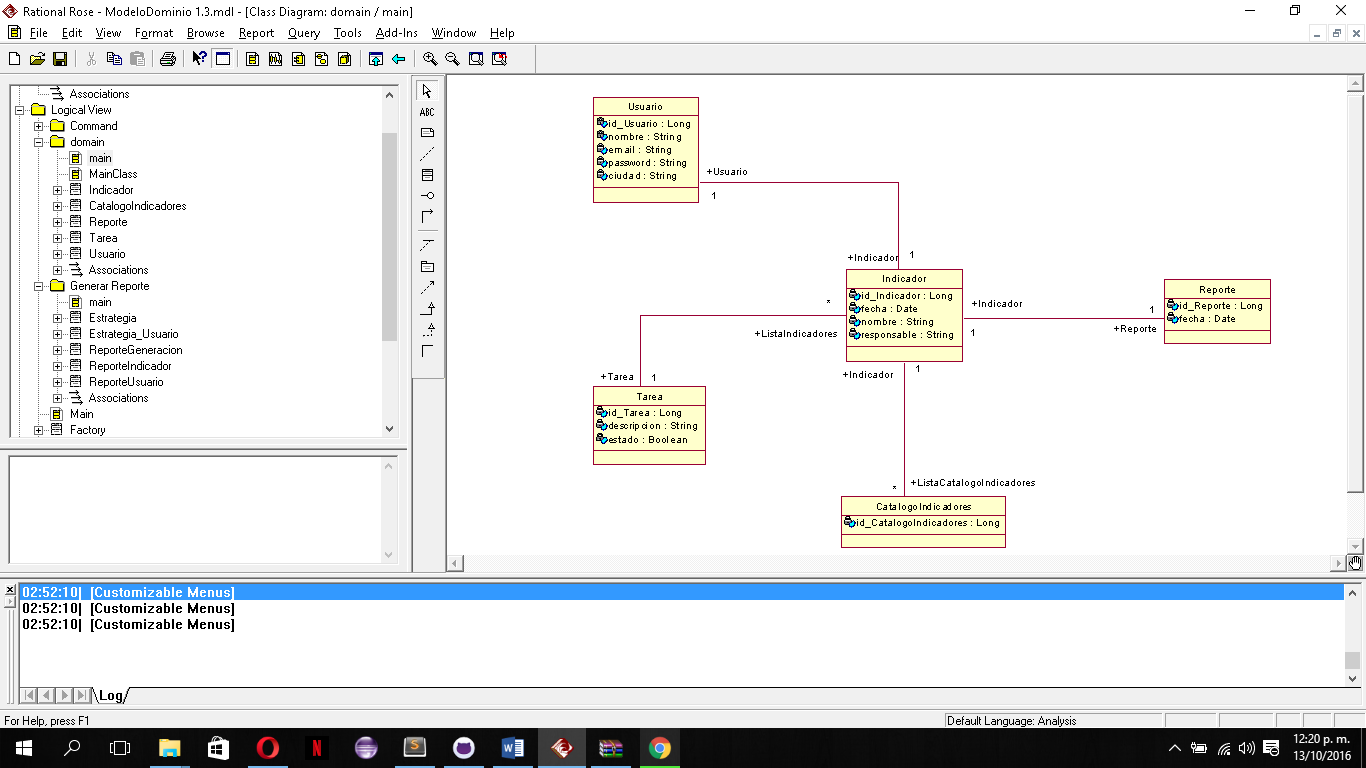


Figura Diagrama del modelo de datos

Se implementó el patrón de diseño Command, esto para poder tener un control sobre las acciones principales del negocio y poder tener un mejor interactuar mejor con ellas.

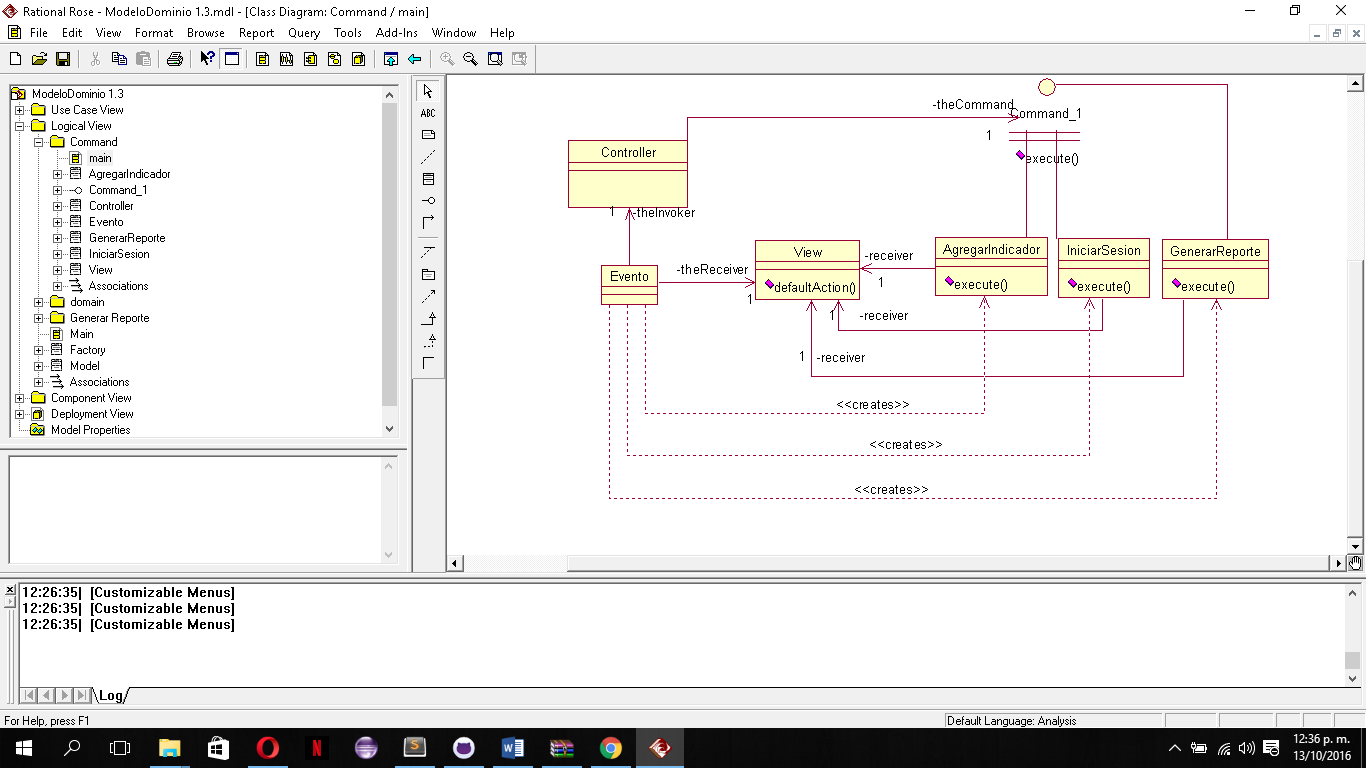


Figura Diagrama del modelo del patrón Command.

Se implementó el patrón de Strategy, esto con el fin de mejorar el diseño y tener un mayor rendimiento en las consultas para realizar las consultas en la base de datos.

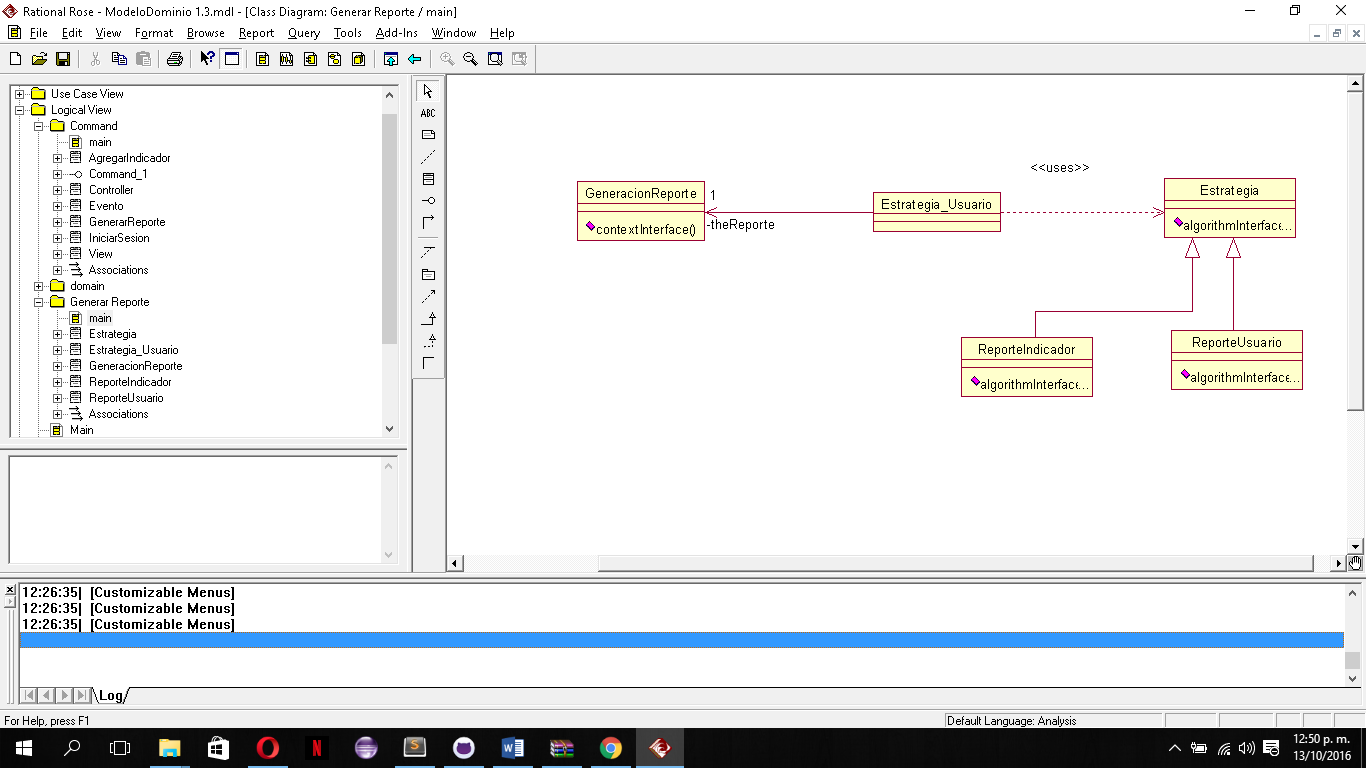


Figura Diagrama del modelo del patrón Strategy para los reportes.

* + 1. **Controllers.**

El controlador es el intermediario entre las vistas y los datos, el controlador Home será el encargado de ejecutar todas las peticiones que se hagan de un evento determinado.

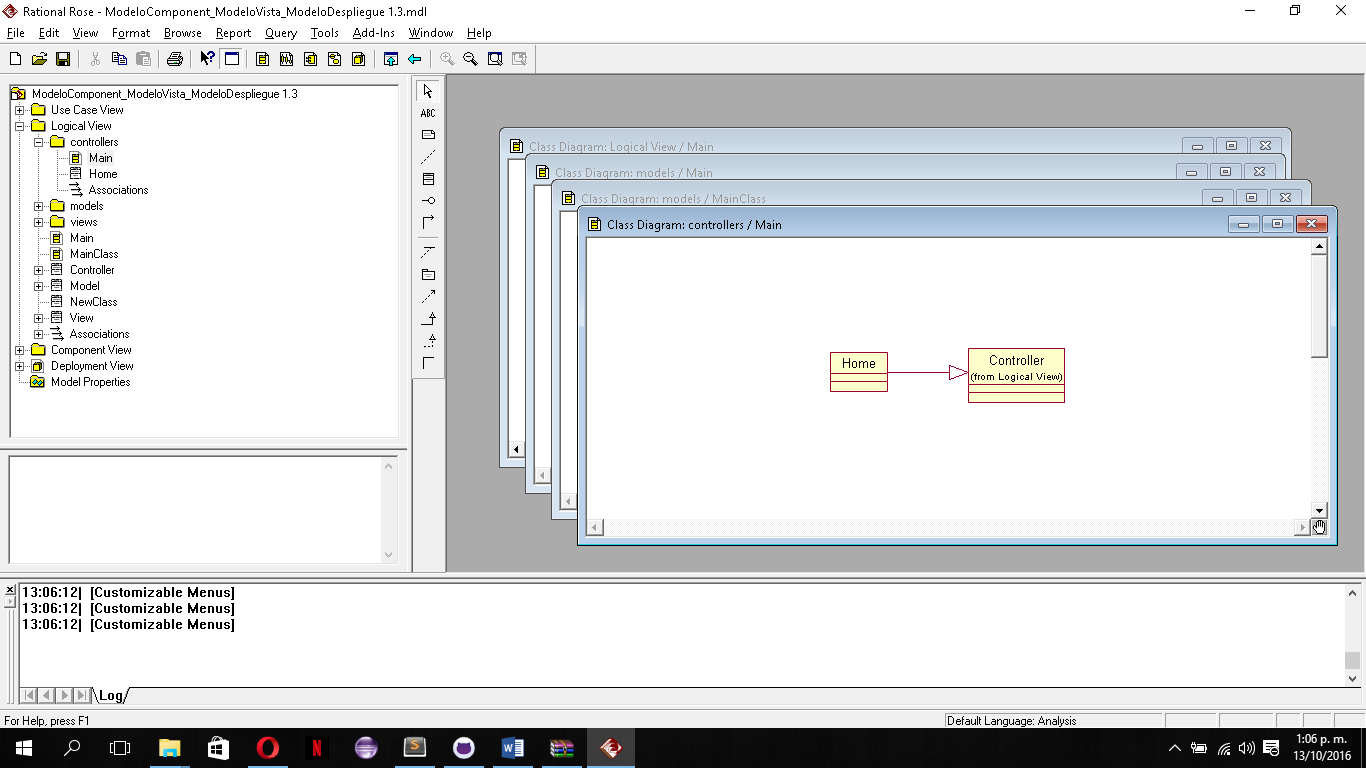


Figura Diagrama del controlador y clases que heredan

* + 1. **Views.**

Las views o vistas son las encargadas de presentar la información a los usuarios, por este motivo se necesitan diferentes views para presentar diferentes tipos de información al usuario, en el diagrama que se presenta a continuación se mostrará la representación en clases de las vistas son necesarias para la comunicación del usuario con la información.

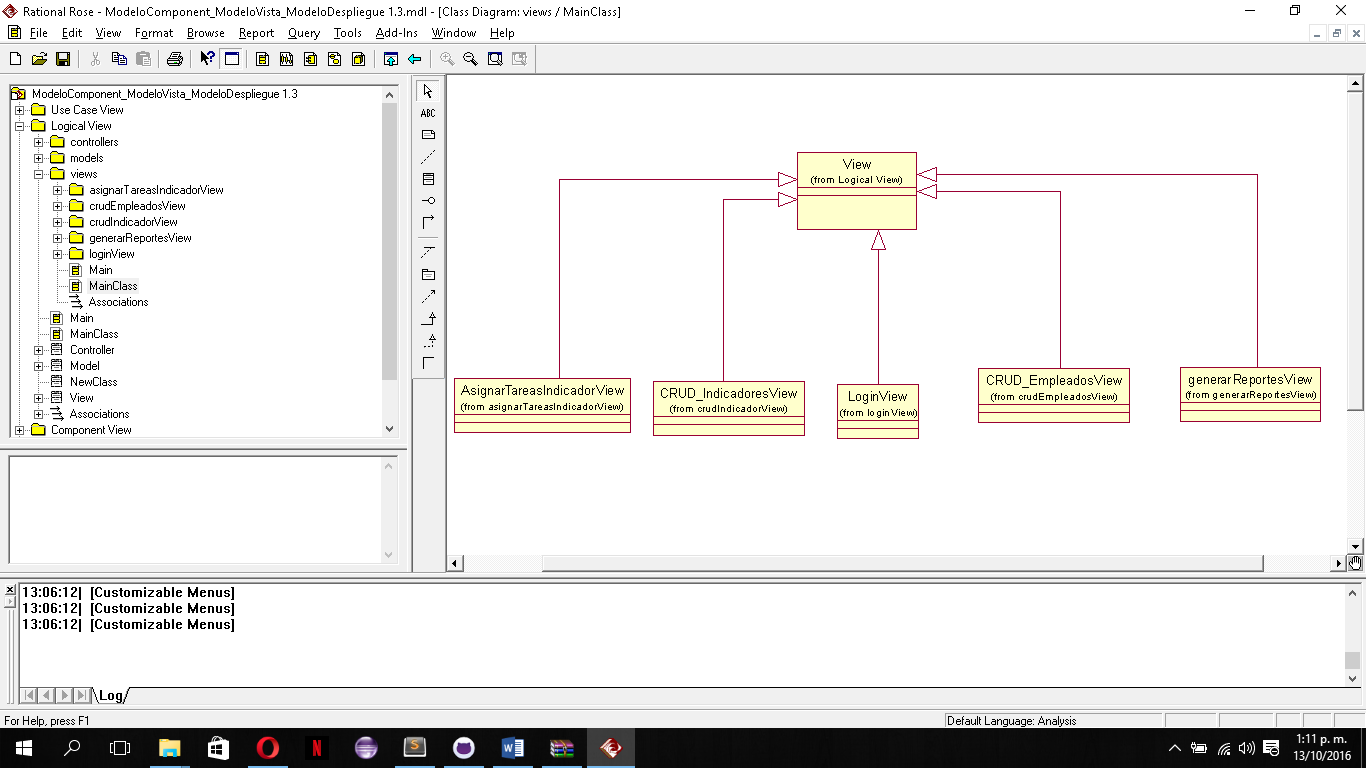


Figura Diagrama del view y las clases que heredan

* + - 1. **Login view.**

Esta vista esta basada en el estilo de diseño Dataform. Esta vista será la encargada de recibir los datos de inicio de sesión de un usuario, comunicará errores si hay datos inválidos y servirá de puerta de enlace entre el usuario y la aplicación.

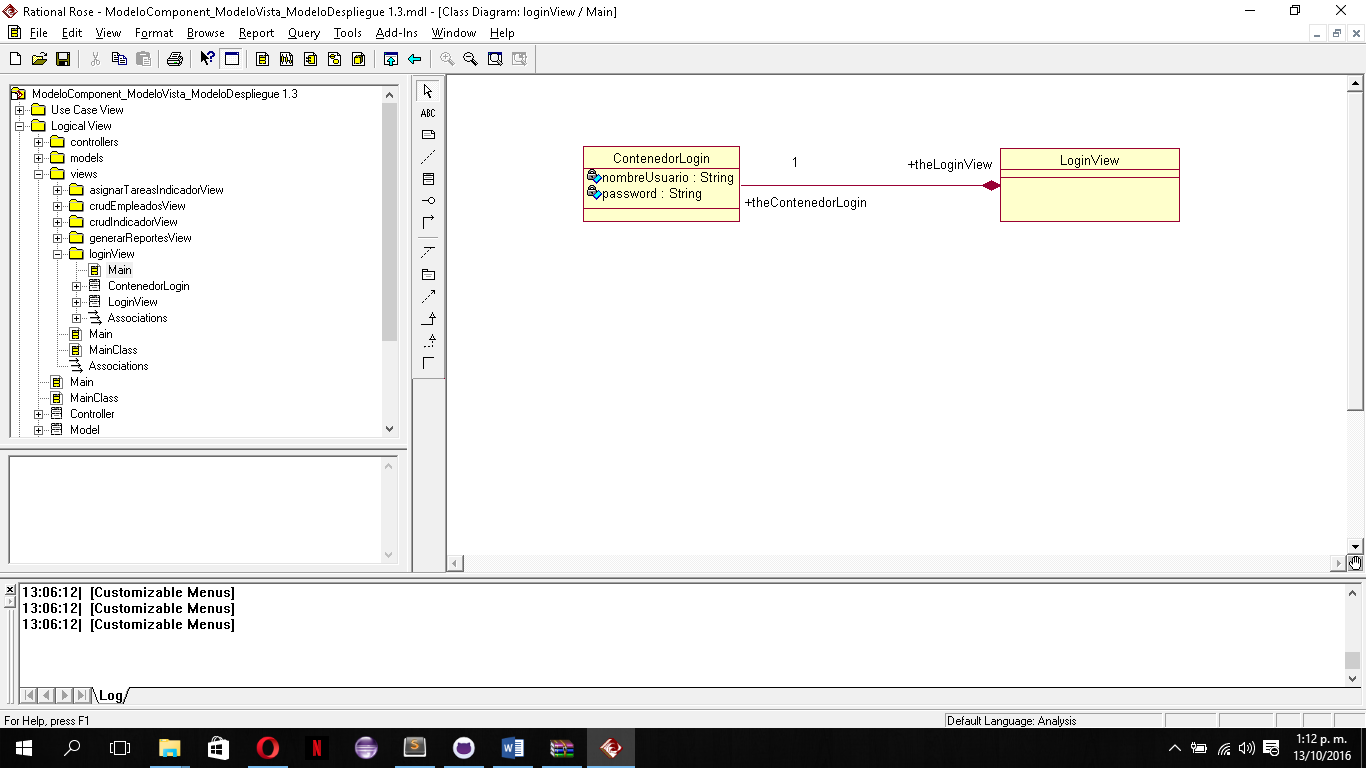


Figura Modelo DataForm del login

* + - 1. **Generar Reportes View.**

A continuación se presentan el modelado de la vista de reportes, esta vista sirve para presentar reportes de viajes que se han realizado.

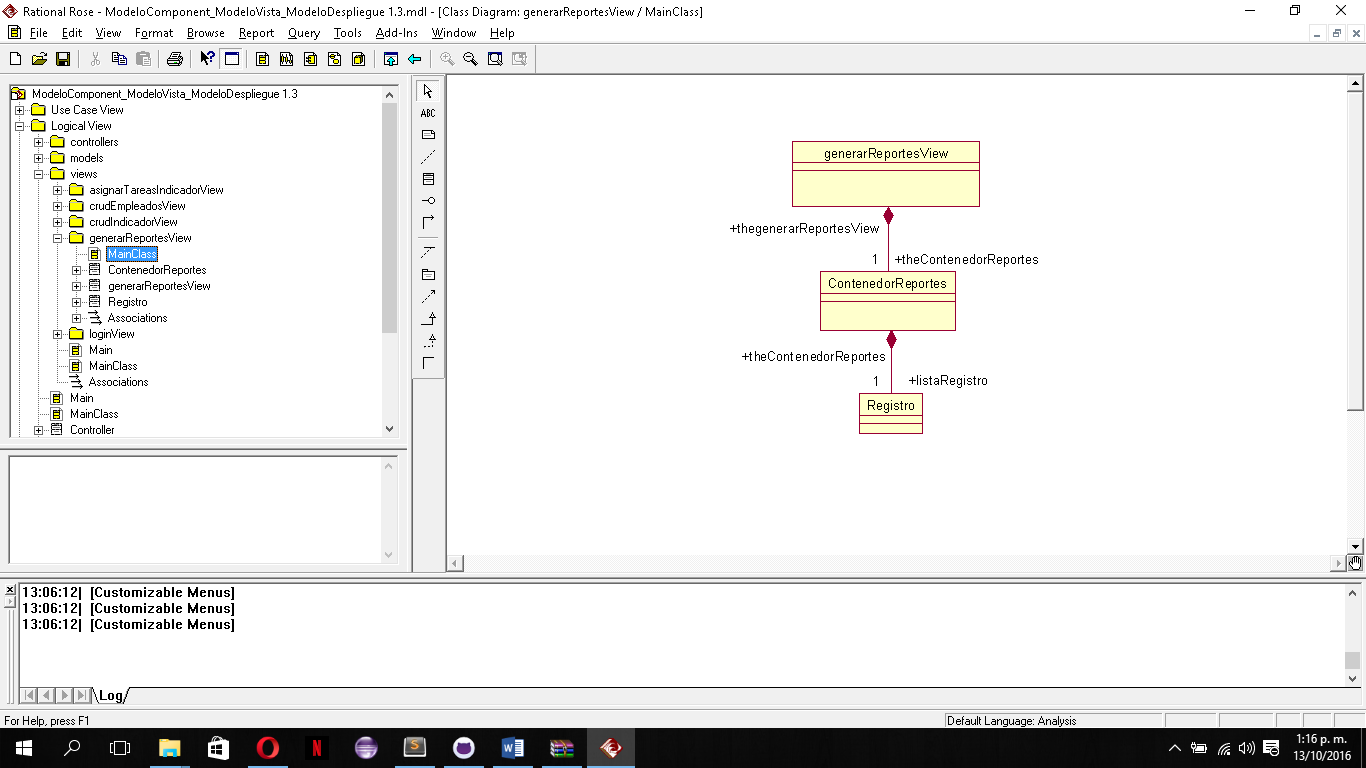


Figura Modelo DataForm de Generar Reportes.

* + - 1. **CRUD Usuarios View**

A continuación se muestra el modelo de la interfaz de usuarios para el CRUD de Usuarios, el cual está basado en el estilo de diseño DataForm. Esta interfaz servirá para crear o eliminar un Usuario dentro del sistema, también para mostrar o actualizar los datos de los Usuarios que se encuentren registrados.

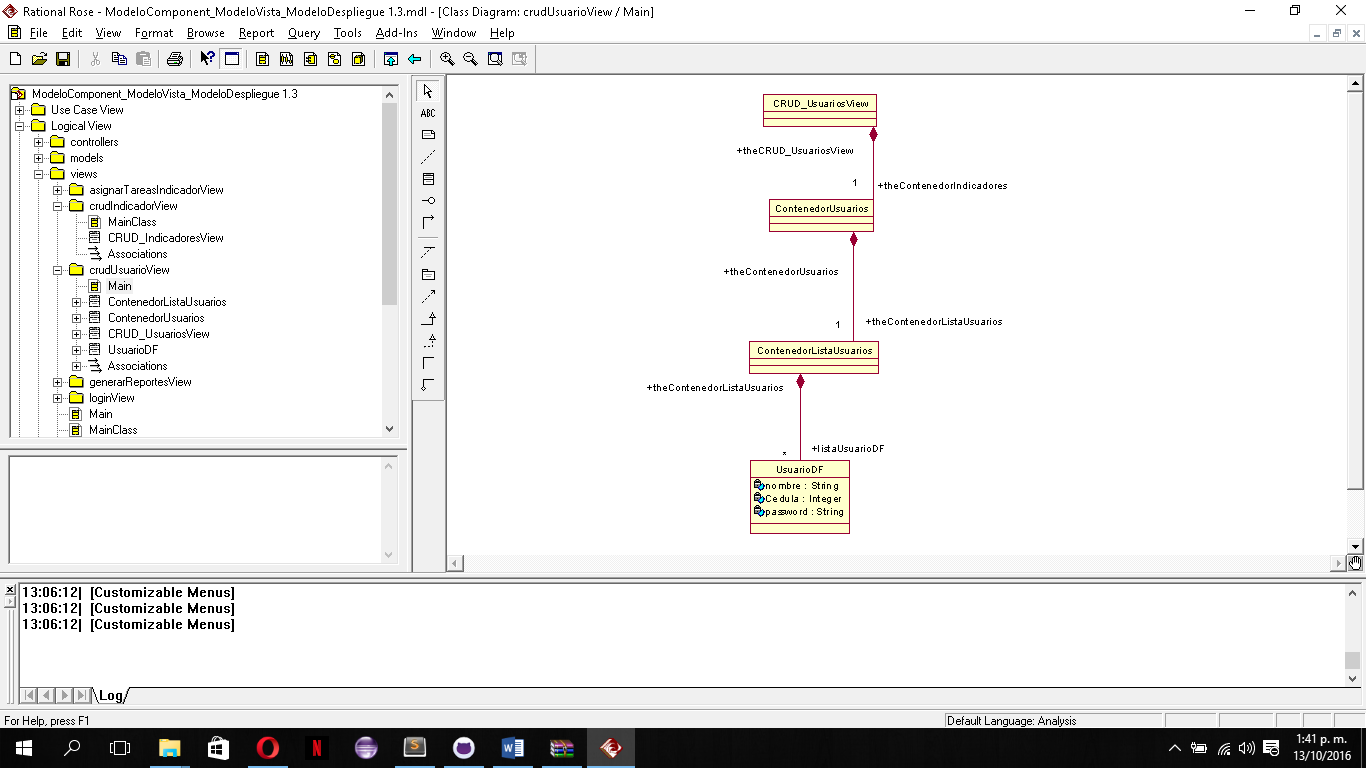


Figura Modelo DataForm CRUD de Usuarios

* + - 1. **CRUD Indicadores View**

A continuación se muestra el modelo de la interfaz de usuario para el CRUD de Indicadores, el cual está basado en el estilo de diseño DataForm. Esta interfaz servirá para crear o eliminar un Indicador dentro del sistema, también para mostrar o actualizar los datos de los Indicadores que se encuentren registrados.

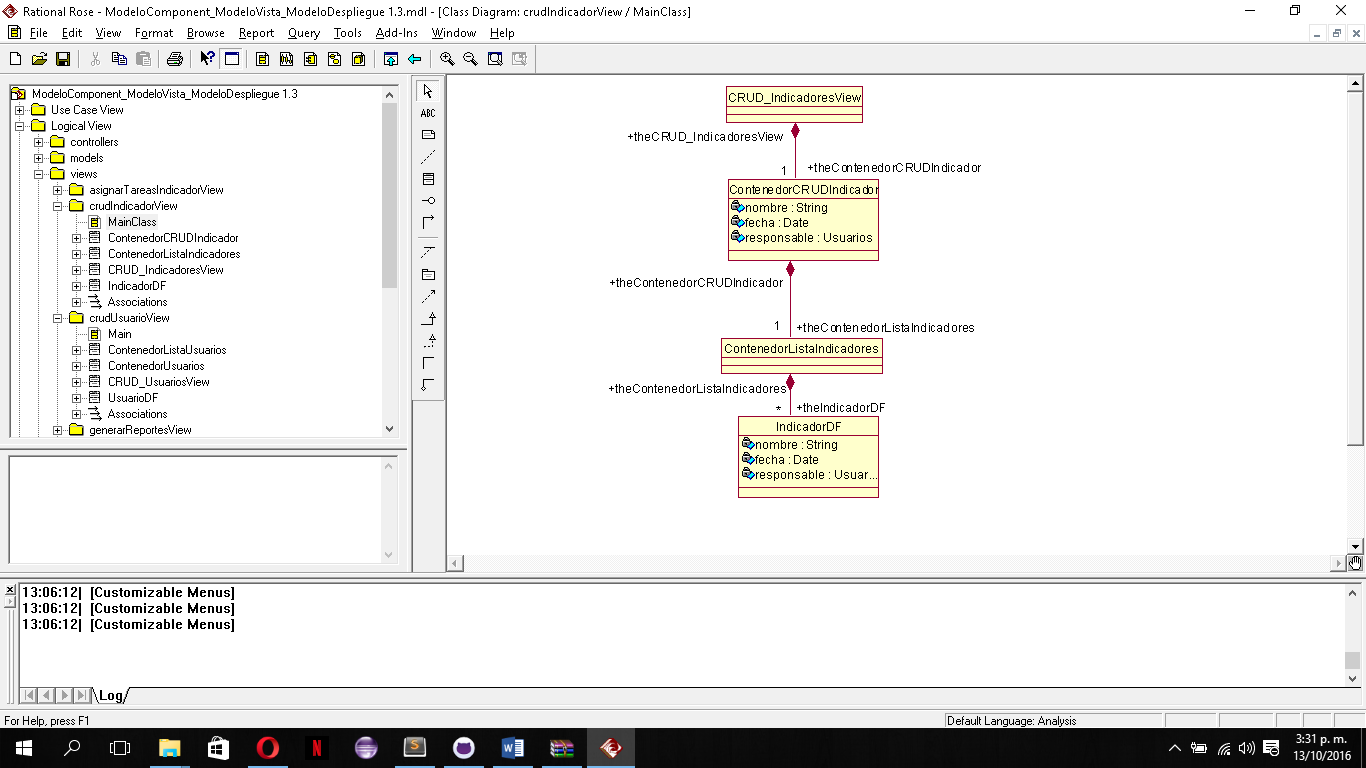


Figura Modelo DataForm de CRUD indicador

* + - 1. **Asignar Tareas Indicadores view**

La vista de asignar tareas está divida en tres vistas Establecer indicador view, Información Indicador View, Metas Indicadores view, respectivamente como está divida se enfoca en agregar correctamente las tareas a los indicadores correspondientes.

* + - * 1. Establecer Indicador view

A continuación se muestra el modelo de la interfaz para la vista de agregar un indicador, la cual está basada en la técnica de diseño Dataform, esta vista se encarga de enlazar un indicador ya agregado a la base de datos y poder interactuar sobre él.

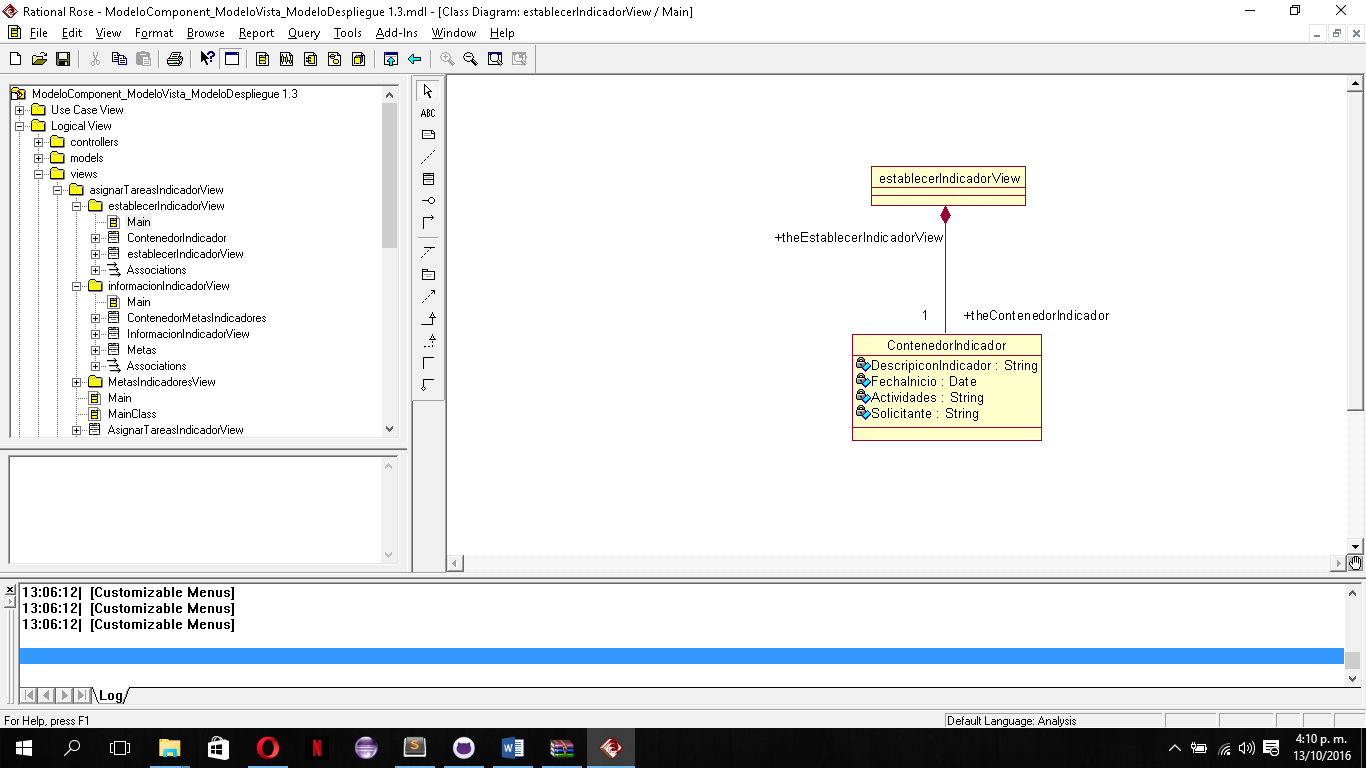


Figura Modelo DataForm de establecer indicador

* + - * 1. **Información indicador View**

A continuación se muestra el modelado de la interfaz de usuario Información indicador, la cual está basada en la técnica de diseño dataform, esta interfaz se encarga de establecer las metas a cumplir para los indicadores asociados.

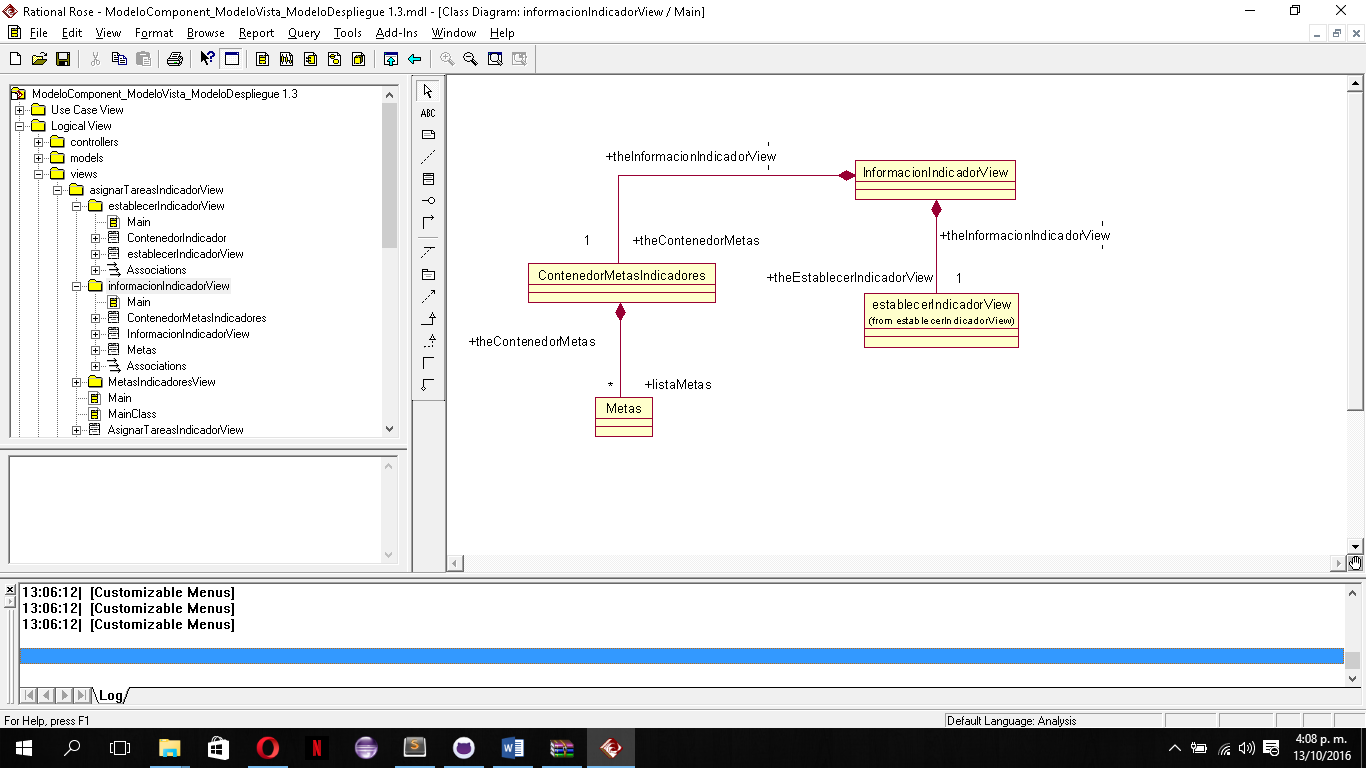


Figura Modelo DataForm de informacion indicador

* + - * 1. **Metas indicadores View**

A continuación se muestra el modelado de la interfaz de usuario metas indicadores, la cual está basada en la técnica de diseño dataform, el propósito de esta interfaz es poder asociar las metas de la información de un indicador.

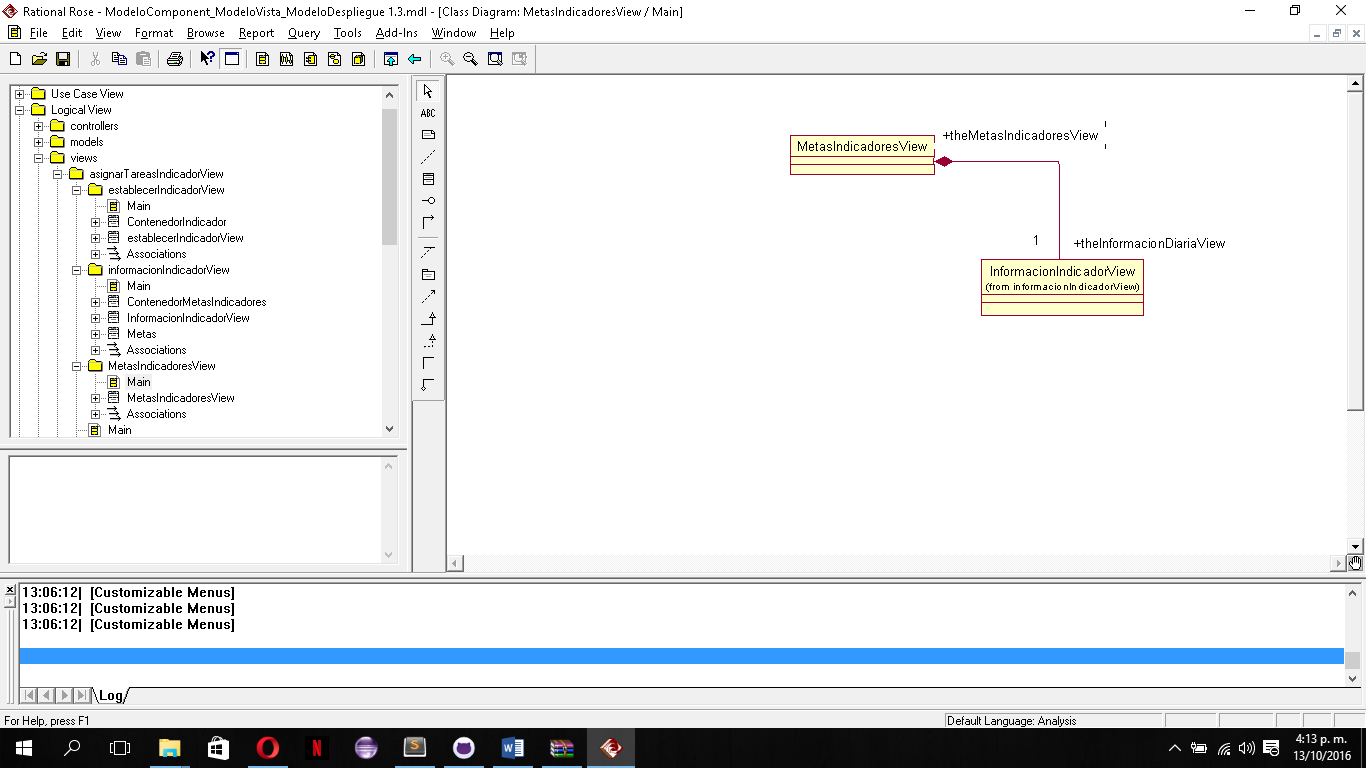


Figura Modelo DataForm Metas Indicadores

## Use-Case Realizations

A continuación se puede ver el diagrama de la realización de los casos de uso que se mostraron anteriormente en el punto cuatro. Además se describirá cada una de las realizaciones de esos casos de uso mediante un diagrama de actividades.

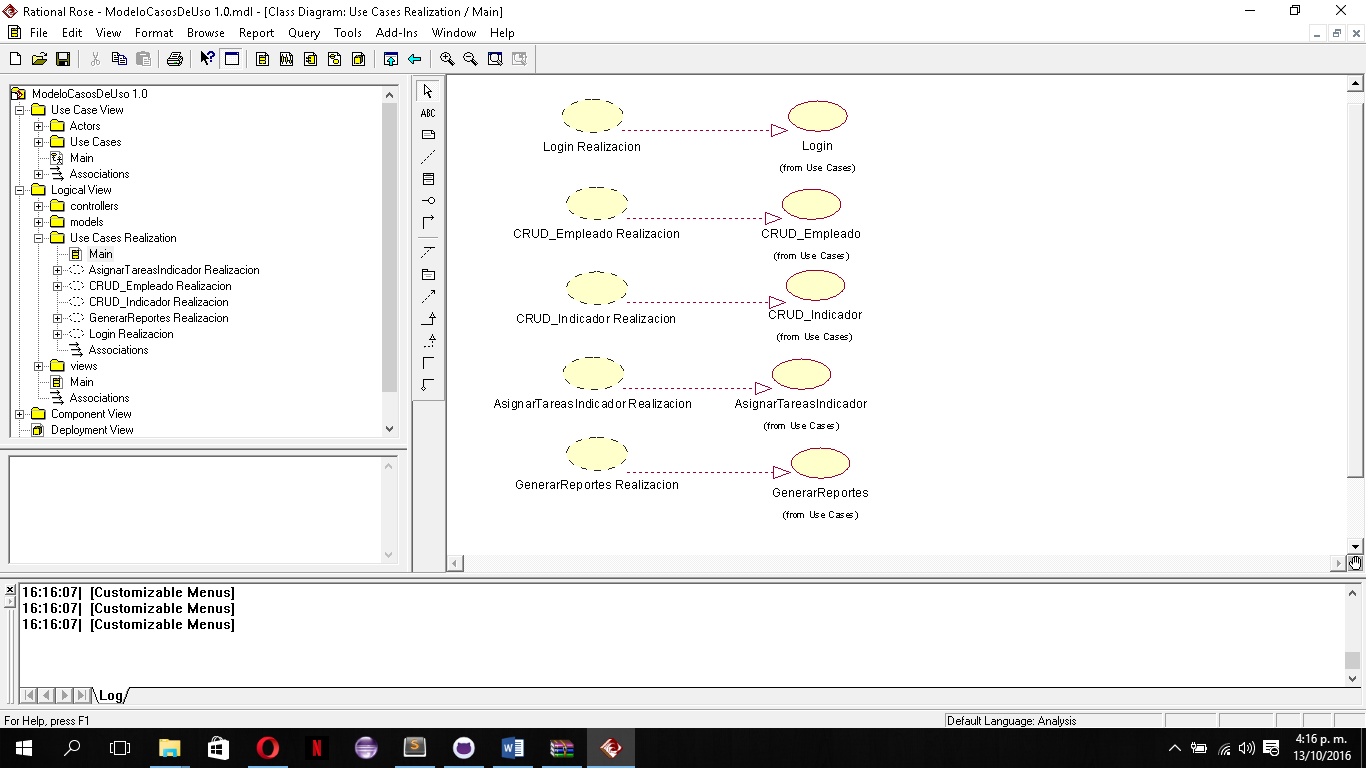


Figura Diagrama de realización de casos de uso

En el siguiente diagrama de actividades Establecemos el modelo de cómo se realiza la actividad de login.

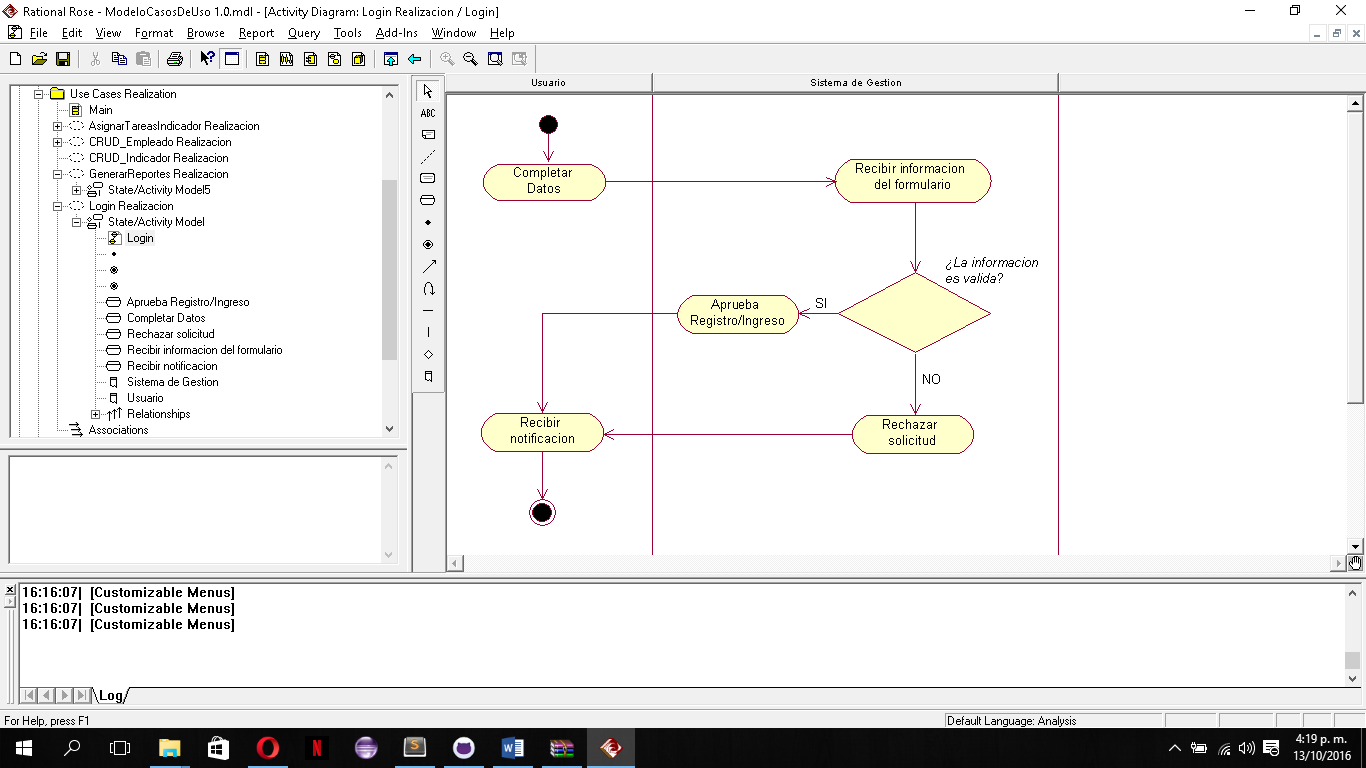


Figura Diagrama de actividades de login

En los siguientes cuatro diagramas de actividades se pueden apreciar los proceso que se llevan a cabo por parte del usuario y del sistema de gestión, para realizar los casos de uso en los que se debe realizar un CRUD.

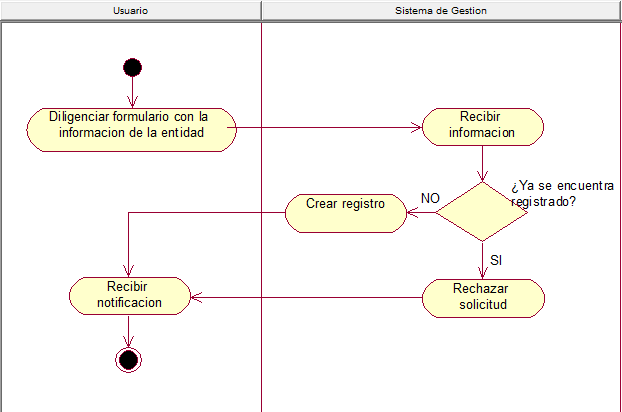


Figura Diagrama de actividad para Crear.

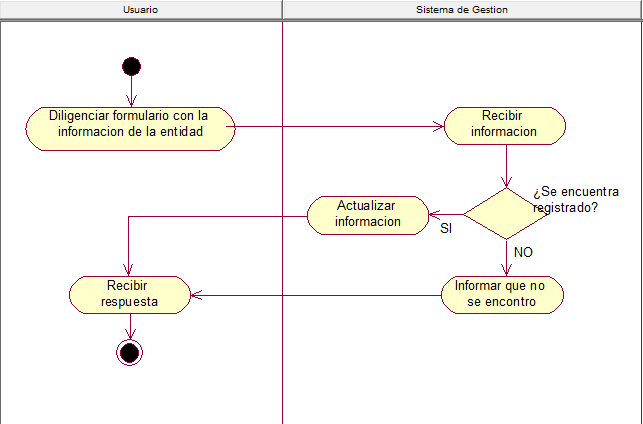


Figura Diagrama de actividades para Actualizar

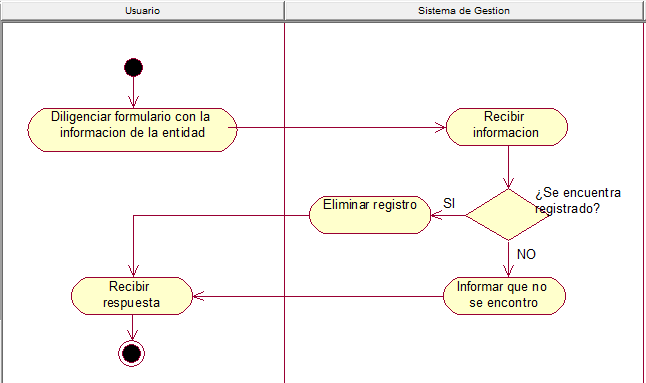


Figura Diagrama de actividades para Eliminar.

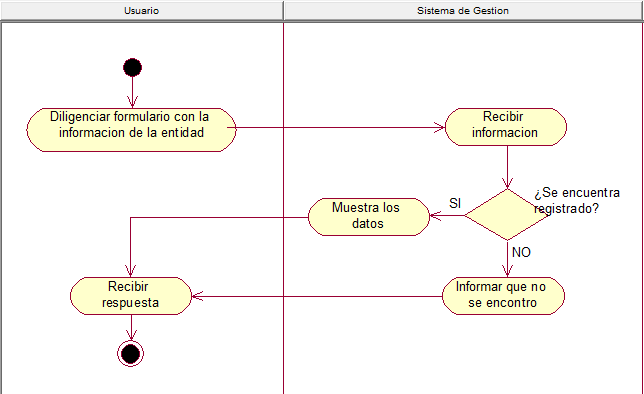


Figura Diagrama de actividades para Leer.

En el siguiente diagrama de actividades se establece el proceso que se llevara a cabo para poder asignar una tarea a un empleado para cumplir la meta de un indicador

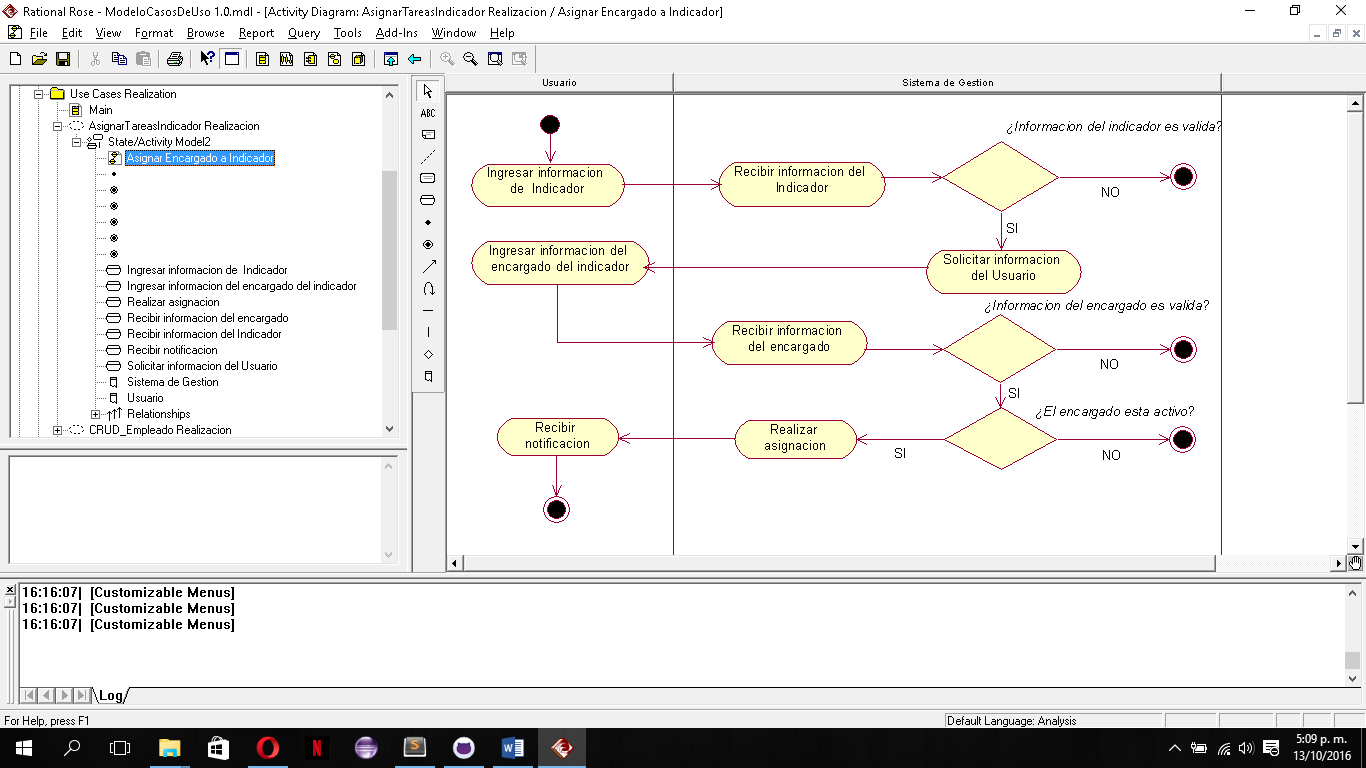


Figura Diagrama de actividades para asignar un empleado a un indicador.

En el siguiente diagrama de actividades establecemos como se llevara el proceso de generar un reporte.

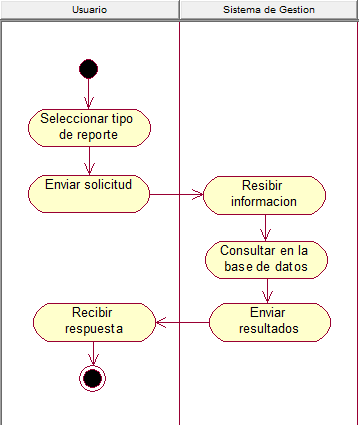


Figura Diagrama de actividades para generar reportes.

# Deployment View

Para el despliegue de la aplicación se necesita un sistema de hosting y servicios para aplicaciones, esta será accedida a través de un computador de forma remota por la red.

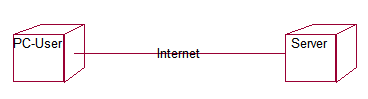


Figura Diagrama de despliegue de la aplicación.

# Implementation View

## Overview

Como el sistema está basado en el patrón MVC este permite dividir la implementación en sus tres capas: Model, View, Controller. Permitiendo el desarrollo independiente entre la presentación, la lógica de negocio y los datos.

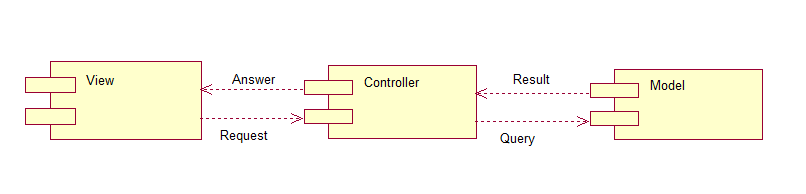


Figura Diagrama de componentes.

## Layers

* + 1. View

En el siguiente diagrama de componentes se puede observar las relaciones de los subsistemas que se encuentran en la capa de presentación, haciendo de cada sub vista un componente que hace parte de esta capa.

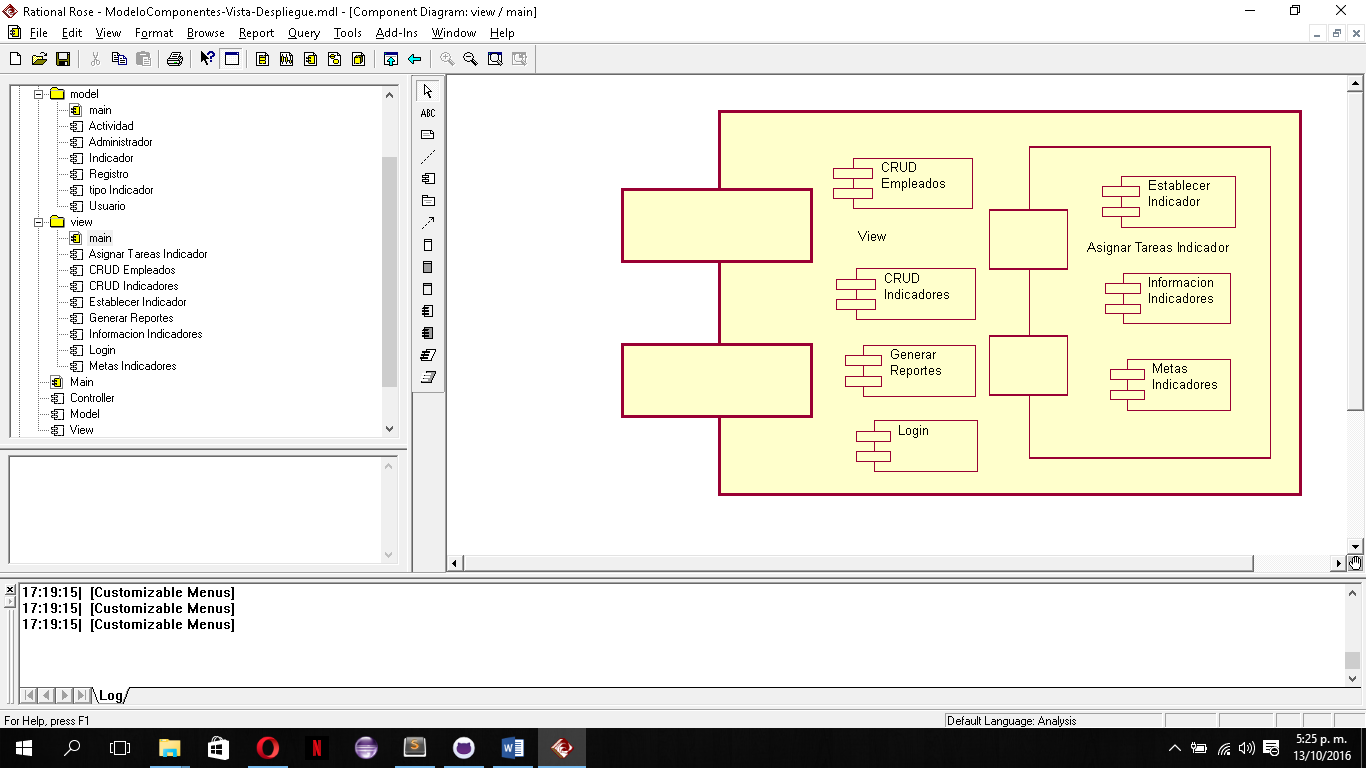


Figura Diagramas de componentes de View

* + 1. Controller

El siguiente diagrama de componentes establecemos a “Home” como único componente en la capa de controladores, esto con el fin de tener un único intermediario entre las capas de vista y modelo.

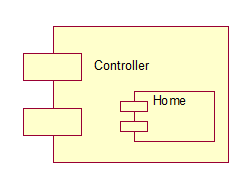


Figura Diagrama de componentes del Controller.

* + 1. Model

El siguiente diagrama de componentes muestra todos los elementos que hacen parte de la capa de modelo, los cuales son necesarios ya que son la representación de la información del sistema.

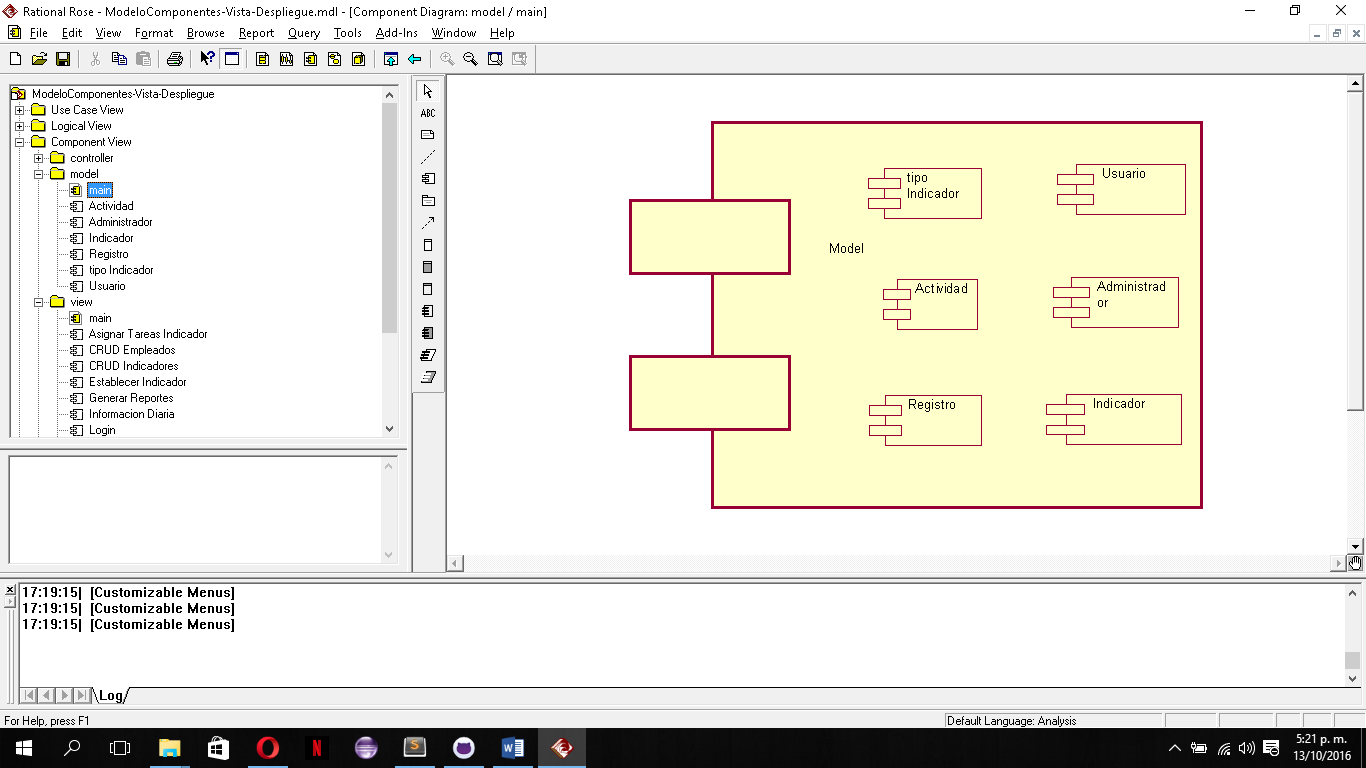


Figura Diagrama de componentes del modelo.

# Size and Performance

El tamaño del sistema está definido por múltiples usuarios incluido un administrador, pero es solo es un worker que se encarga de gestionar la administración de indicadores. Para el rendimiento de la aplicación se utilizará la técnica basada en el aumento de la eficiencia de los recursos del libro “*Software Architecture in Practice*”, donde empleando algoritmos de complejidades no elevadas, el proyecto mejorará el tiempo de respuesta en las áreas críticas, principalmente en la generación de reportes, la cual es un requerimiento primordial del sistema.

# Quality

**Usabilidad:** Para cumplir este atributo de calidad, se incursionaran varias metodologías para realizar interfaces que garanticen la forma y los datos que se encuentran dentro de ella, así se podrá mostrar la información deseada por el usuario, con esto se asegura que las interfaces de usuario son usables para ese perfil de usuario que las utilizará. Primero se utilizará la “*MPIu+a*”, esto garantiza que se realiza un trabajo de ingeniería desde el inicio del proceso, luego pasamos a la etapa de prototipado, para concluir con la evaluación de la interfaz de usuario, esta metodología es iterativa y se puede ir realizando a través de las fases del proyecto. Para cumplir que la forma y los datos estén sin la subjetividad del diseñador, se aplicará la técnica DataForm, la cual se basa en tener un estándar de diseño de interfaces, además esta técnica nos ayuda a corregir errores de la interfaz de usuario, haciendo un proceso iterativo del modelo de datos de la interfaz y su respectivo DataForm.

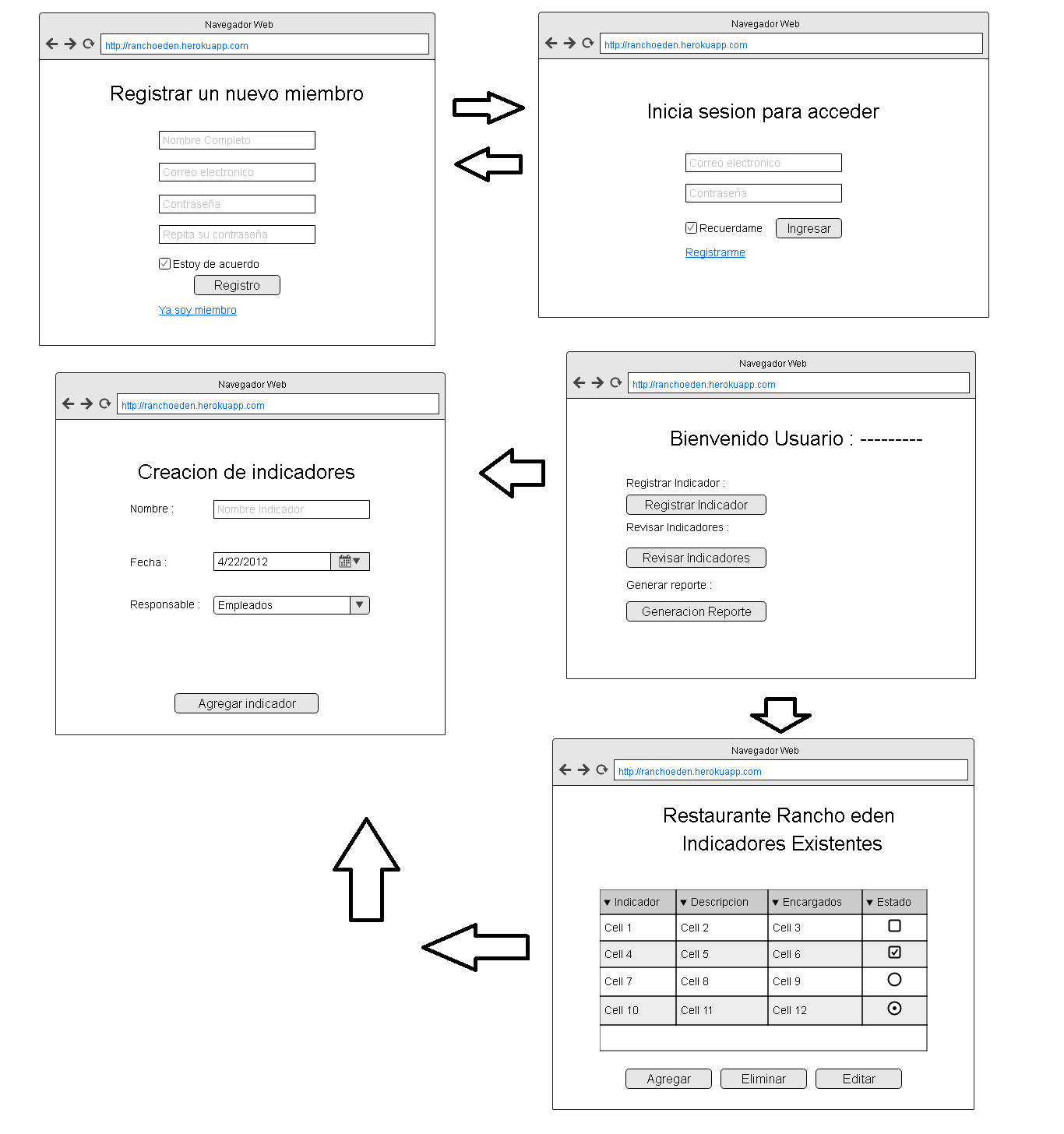


Figura Interacción de las interfaces para reservar un vehículo.

En la figura se muestra la navegación por las vistas para poder registrar, ingresar y buscar tanto Usuarios como indicadores, se parte del hecho que obligatoriamente tiene que haber hecho login el usuario para poder acceder a estas funciones,puede estar entre la pantalla de inicio de sesión o en la de registro,Si su registro es exitoso podrá acceder a la creación de indicadores ,modificación de ellos y generación de reportes-

**Seguridad:** para garantizar este atributo de calidad se implementará la técnica de ingreso por contraseña, la cual servirá para autenticar y autorizar el ingreso al sistema, esto con el fin de que personas no autorizadas no tengan permisos para realizar cambios u operaciones en el sistema. También para garantizar la privacidad de los datos se utilizará una técnica de encriptamiento basada en AES, con el fin de mantener la información segura y que sea poco entendible para las personas que realicen ataques a la base datos.