

Propuesta de Colaboración Profesional

**IEEE 1471-2000 Proyecto: Desarrollo e Implementación de Software de Administración para la Mantención y Reparación de Vehículos.**

Preparado para:

**Taller de Desarrollo de Software I**

**Profesor Oscar Pinto G.**

12 de Diciembre del 2011

Preparado por:

**Cristian Vásquez Lean**

**ÍNDICE TEMÁTICO**

**CONTENIDOS Nº Pág.**

[1 Introducción 4](#_Toc311986547)

[1.1 Propósito 4](#_Toc311986548)

[1.2 Alcance 4](#_Toc311986549)

[1.3 Requerimientos Funcionales 5](#_Toc311986550)

[1.4 Usuarios Interesados 8](#_Toc311986551)

[2 Referencias 8](#_Toc311986552)

[3 Definiciones, acrónimos y abreviaciones. 8](#_Toc311986553)

[4 Framework Conceptual 9](#_Toc311986554)

[4.1 Descripción de la arquitectura en contexto 9](#_Toc311986555)

[4.2 Stakeholders y sus roles 9](#_Toc311986556)

[4.3 Usos de las descripciones de arquitectura 9](#_Toc311986557)

[4.4 Identificación de los Stakeholders y sus responsabilidades 9](#_Toc311986558)

[4.5 Selección de puntos de vista de la arquitectura. 10](#_Toc311986559)

[4.6 Vistas de la arquitectura 10](#_Toc311986560)

[4.7 Modelo Entidad Relación 16](#_Toc311986561)

[5 Conclusiones 17](#_Toc311986562)

**ÍNDICE FIGURAS**

[Figura 1 - Vista general Diagrama CU 11](#_Toc311986536)

[Figura 2 - Vista Personal Diagrama CU 11](#_Toc311986537)

[Figura 3 - Vista Detallada Diagrama CU 12](#_Toc311986538)

[Figura 4 - Diagrama Secuencia – Módulo Falla 12](#_Toc311986539)

[Figura 5 - Diagrama Secuencia – Módulo Mantención y Reparación Vehículos 13](#_Toc311986540)

[Figura 6 – Diagrama Secuencia - Módulo Administrar Repuestos 13](#_Toc311986541)

[Figura 7 - Diagrama Secuencia - Módulo Administrar Vehículos 14](#_Toc311986542)

[Figura 8 - Diagrama de Clases 14](#_Toc311986543)

[Figura 9 - Diagrama de Componentes 15](#_Toc311986544)

[Figura 10 - Diagrama Despliegue 15](#_Toc311986545)

[Figura 11 - Modelo Entidad Relación 16](#_Toc311986546)

# Introducción

## Propósito

Este documento proporciona una descripción comprensiva arquitectónica del sistema, usando un número contable de vistas diferentes para representar los distintos aspectos que se requieren para capturar y transportar las decisiones significativas que han sido hechas sobre el sistema.

## Alcance

El presente documento contiene el diseño elaborado para el proyecto de Software para la Administración de Mantenciones y Reparaciones de Vehículos en la empresa de Taxi – Colectivos *SOPRAF S.A.*, el cual es producto de un análisis minucioso de los requisitos del sistema, según estos pueden ser satisfechos con las tecnologías y características discutidas con los clientes y usuarios.

El documento está organizado alrededor de tres ideas principales.

1. Las características generales del diseño.
2. Los requisitos atendidos por el diseño.
3. Los modelos y vistas que lo detallan.

Al contrario de muchas otras actividades técnicas, el desarrollo de sistemas intensivos en software dedica la mayoría de sus esfuerzos a la especificación y modelado. Los modelos son utilizados tanto para el análisis de requisitos, como para el diseño de la solución, así como para la especificación, construcción y despliegue del sistema en su ambiente de explotación. Los modelos son presentados por vistas o diagramas, generalmente utilizando notaciones gráficas como el UML.

Por otro lado, los programas de computadora son construidos por medio del uso de herramientas de traducción automáticas llamados compiladores, para los cuales es construida la forma lineal y más detallada del software del sistema: el código fuente.

La última sección del documento indica la forma en que se puede obtener el código fuente del proyecto así como las instrucciones de compilación necesarias para lograr la ejecución de los componentes que este código detalla.

Este documento ha sido generado directamente del análisis realizado a través del documento IEEE 830, se ha de utilizar el modelo 4+1 de Kruchten para confeccionar los diferentes diagramas que lo componen.

## Requerimientos Funcionales

A través del análisis realizado en la etapa anterior, se lograron extraer los siguientes requerimientos funcionales los cuales fueron documentados con la *IEEE 830* y que serán utilizados para modelar los siguientes diagramas de diseño para apoyar la arquitectura del software.

|  |  |
| --- | --- |
| **Numero :** | Módulo Ingreso |
| **Requisito :** | El software debe tener un módulo de autentificación de usuarios. |
| **Descripción :** | El software debe solicitar un usuario y contraseña para poder trabajar, de lo contrario no se podrá ejecutar ninguna acción. |
| **Proceso :** | El usuario tiene la posibilidad de utilizar sólo lo que le corresponde dependiendo su responsabilidad en el software. El administrador realiza las asignaciones de información. |
| **Entrada :** | Nombre Usuario – Contraseña Usuario. |
| **Salida** | Visualización de módulos según perfil de usuario. |

|  |  |
| --- | --- |
| **Numero :** | Módulo Administración |
| **Requisito :** | Se compone de los siguientes sub módulos:   1. Módulo Proveedor. 2. Módulo Vehículo. 3. Módulo Personal. 4. Módulo Fallas. 5. Módulo Orden de Trabajo.   Los cuales deben permitir las siguientes funcionalidades   * Crear. * Borrar. * Modificación. * Buscar. |
| **Descripción :** | Permite realizar operaciones con los sub módulos del sistema |
| **Proceso :** | El usuario Administrador es el encargado de utilizar y administrar estos sub módulos |
| **Entrada :** | Selección de sub módulo. |
| **Salida** | Creación, Modificación, Eliminación, Búsqueda |

|  |  |
| --- | --- |
| **Numero :** | Módulo Informes e Indicadores |
| **Requisito :** | El software deberá proveer las siguientes funcionalidades para administrar los informes de la empresa:   * Crear de Informe. |
| **Descripción :** | Permite realizar operaciones con los datos de la empresa tanto de los repuestos, fallas, servicios, vehículos, personal, compras. |
| **Proceso :** | El administrador es el encargado de seleccionar los datos que desea. |
| **Entrada :** | Datos de repuestos, fallas, servicios, vehículos, personal, compras. |
| **Salida** | Creación de informe PDF. |

|  |  |
| --- | --- |
| **Numero :** | Módulo Ingreso Compras |
| **Requisito :** | Se requiere de las siguientes funcionalidades:   * Crear. * Borrar. * Modificación. * Buscar. |
| **Descripción :** | Permite ingresar las compras de los repuestos. |
| **Proceso :** | El Administrador es el involucrado de ingresar las facturas de los repuestos. |
| **Entrada :** | Factura con el detalle de la compra. |
| **Salida** | Ingreso de datos de la compra al software. |

|  |  |
| --- | --- |
| **Numero :** | Módulo Retiro Repuestos |
| **Requisito :** | Requiere de la existencia de repuestos y stock. |
| **Descripción :** | El encargado de bodega tiene la facultad para poder buscar, retirar y ver repuestos. |
| **Proceso :** | Consulta Stock y busca el repuesto que quiere en la bodega y posteriormente lo ingresa con el lector de códigos de barra para poder realizar el retiro. |
| **Entrada :** | Búsqueda y chequeo de stock de repuestos. |
| **Salida** | Retiro de repuestos. |

## Usuarios Interesados

Este documento de Arquitectura de Software (DAS), está diseñado para los usuarios de la empresa *SOPRAF S.A.*

# Referencias

Las referencias aplicables a este documento son:

- IEEE 830-1998 ST.

- ISO 9126 -2001 Calidad del Software y Métricas de evaluación.

- The 4+1 View Kruchten.

- UML Gota a Gota – M. Fowler.

# Definiciones, acrónimos y abreviaciones.

DAS: Documento de Arquitectura de Software

AMRV: Administración para la Mantención y Reparación de Vehículos

TCP: Protocolo de control de transmisión.

1. Arquitectura de Software: Conjunto de elementos estáticos, propios del diseño intelectual del sistema, que definen y dan forma tanto al código fuente, como al comportamiento del software en tiempo de ejecución. Naturalmente este diseño arquitectónico ha de ajustarse a las necesidades y requisitos del proyecto.
2. Descripción de la Arquitectura: Colección de productos de documentación.
3. Vistas: Es una representación de un área de interés o perspectiva del sistema en alto nivel.
4. Tipos de vistas: Especificación de una convención de cómo construir y usar una vista. Deben satisfacer la capacidad de creación y análisis de una vista.
5. Stakeholder: Individuo, equipo u organización con intereses relativos al sistema.
6. Escenario: especifica el comportamiento y limita el interés de un área específica del sistema para uno o varios Stakeholders.
7. Módulo: Cualquier elemento estructural abstracto, visible, externo, de alto nivel, analizable, que pueda constituir una funcionalidad de la solución del sistema.
8. Atributos de Calidad: Un atributo de calidad, es una cualidad deseable de la solución, que pueda manifestarse en forma de requerimiento no funcional, que pueda ser medible, testeable y finalmente evaluable.

# Framework Conceptual

## Descripción de la arquitectura en contexto

Este documento presenta la arquitectura como una serie de vistas basadas en la arquitectura de software del modelo 4+1 de Kruchten. Estas vistas son: la vista de escenarios, la vista lógica., la vista de desarrollos, la vista física, la vista de procesos. No hay ninguna vista separada de una misma implementación, descrita en este documento.

Estas vistas están hechas sobre Lenguaje de modelo unificado (UML) en su versión 2.0. Los estilos arquitectónicos serán referenciados en este documento de arquitectura, según las recomendaciones de la Arquitectura de software del modelo 4+1 de Kruchten.

## Stakeholders y sus roles

Este documento representa la identificación de Stakeholders y sus roles a partir de la interpretación de los casos de uso del Negocio.

## Usos de las descripciones de arquitectura

Las descripciones de arquitectura de este documento se usaran para referenciar el diseño del Software AMRV.

## Identificación de los Stakeholders y sus responsabilidades

|  |  |  |  |
| --- | --- | --- | --- |
| Stakeholder | Descripción | Escenario | Vistas |
|  | Encargado de realizar asignaciones de las ordenes de trabajo, como también de la administración general del software con la facultad de agregar usuarios, vehículos, repuestos, fallas, compras. | * Escenario negocios * Escenario diseño | * CU Módulo Ingreso. * CU Módulo Administración. * CU Módulo Informes de Indicadores. * CU Módulo Ingreso Compras. |
|  | Encargado de retirar los repuestos de la bodega. | * Escenario negocios * Escenario diseño | * CU Módulo Ingreso. * CU Módulo Retirar Repuesto. |

## Selección de puntos de vista de la arquitectura.

|  |  |
| --- | --- |
| ***Vistas*** | ***UML*** |
| ***Escenarios*** | ***Casos de Uso*** |
| ***Procesos*** | ***Secuencia*** |
| ***Lógica*** | ***Clases*** |
| ***Desarrollo*** | ***Componentes*** |
| ***Física*** | ***Despliegue*** |

## Vistas de la arquitectura

*Vistas 🡪 Escenarios 🡪 Diagrama Caso de uso*

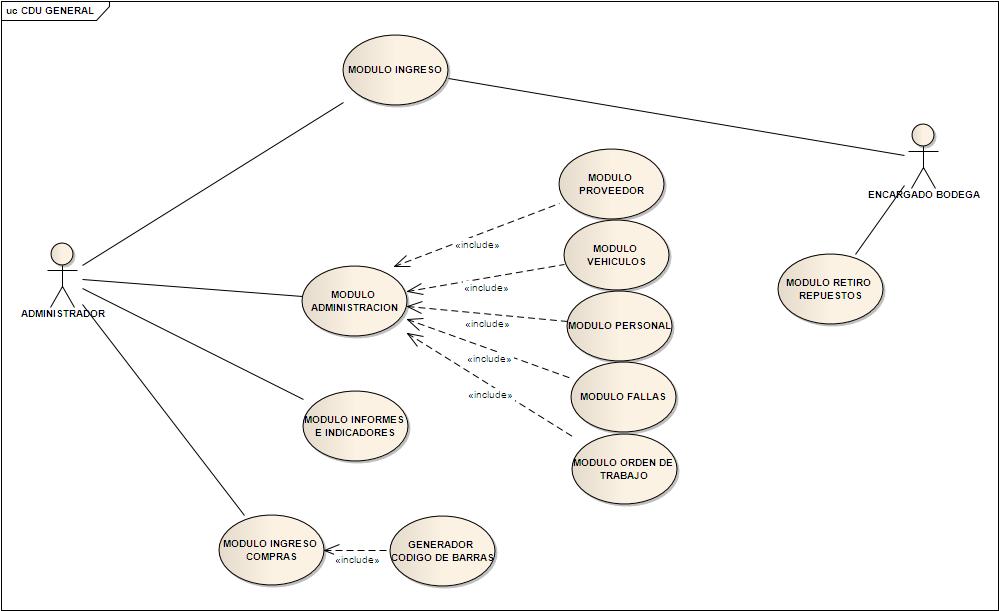
**

Figura - CU Vista General

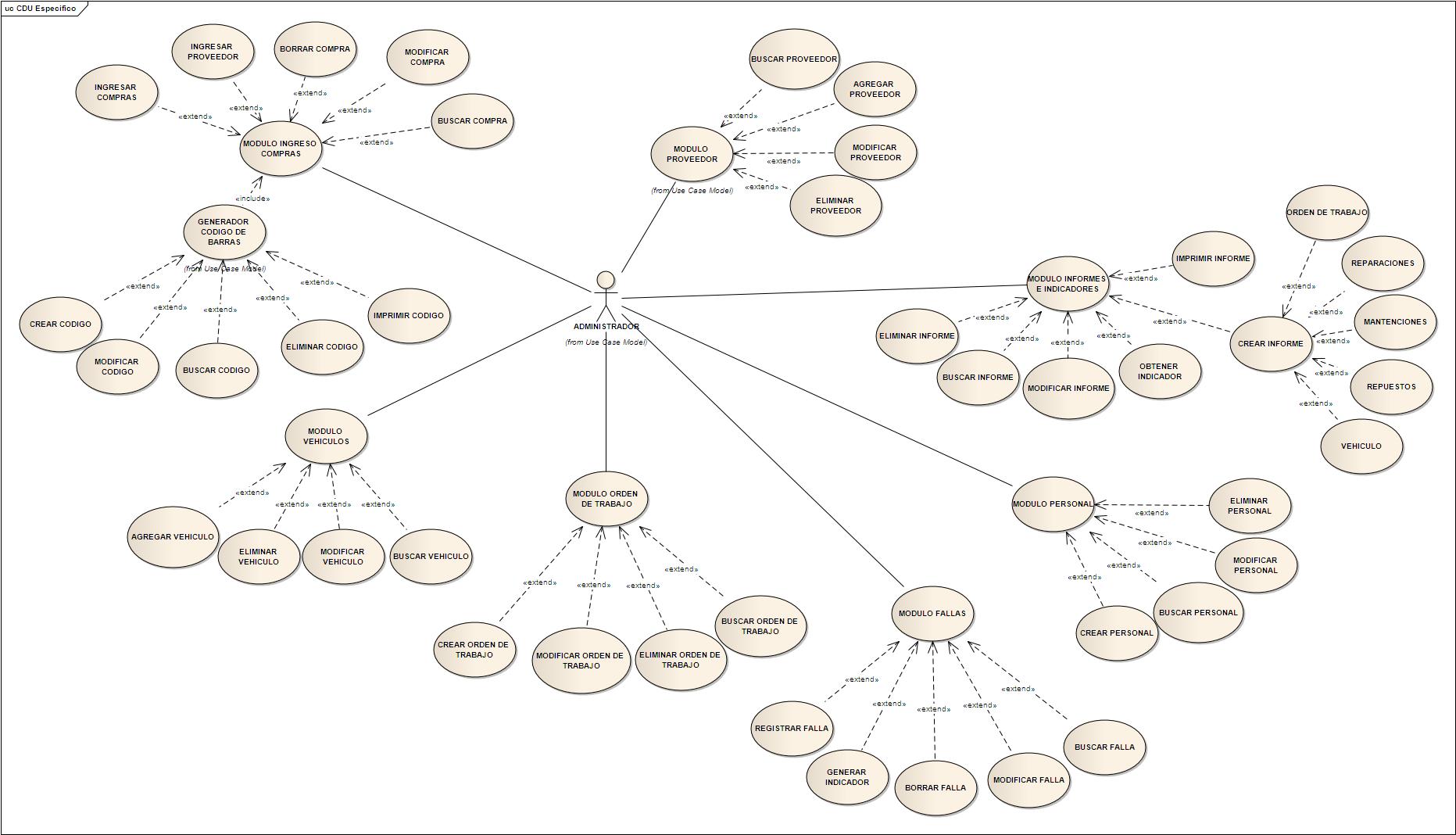


Figura – CU Administrador – Vista Específica

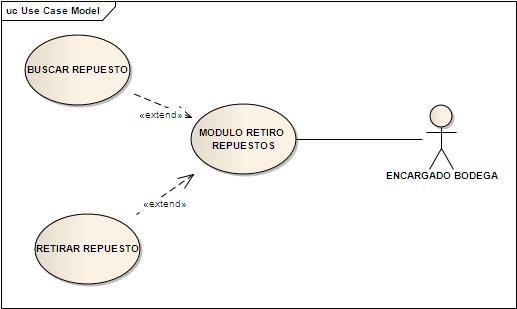


Figura - CU Encargado Bodega – Vista Específica

*Vista 🡪 Procesos 🡪 Diagrama Secuencia*

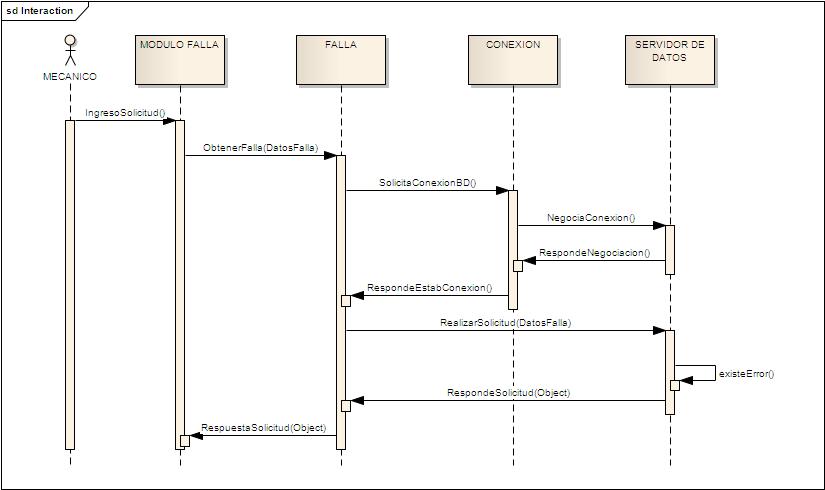
**

Figura - Diagrama Secuencia – Módulo Falla

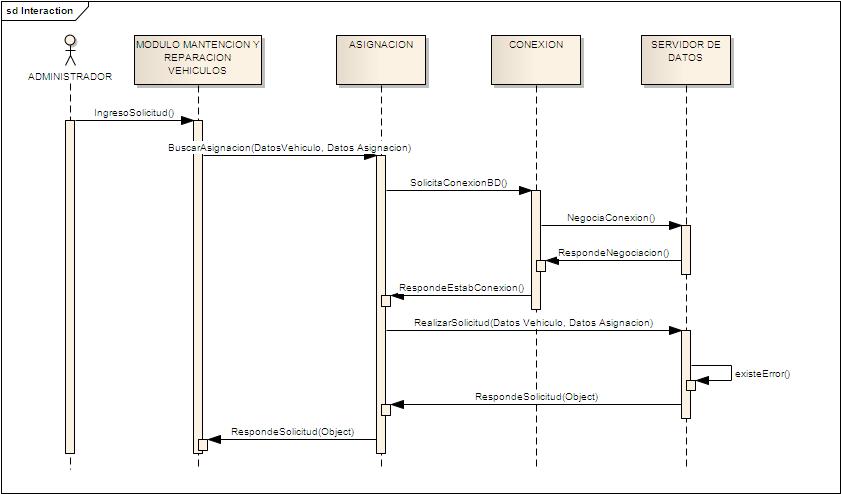
**

Figura - Diagrama Secuencia – Módulo Mantención y Reparación Vehículos

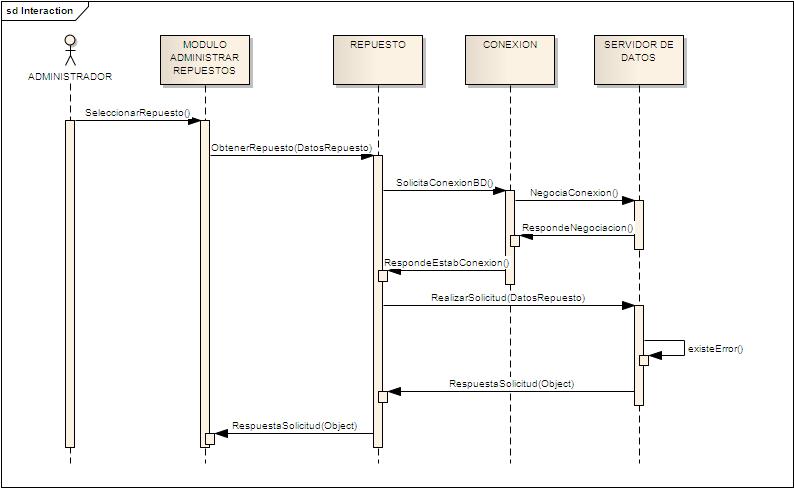
**

Figura – Diagrama Secuencia - Módulo Administrar Repuestos

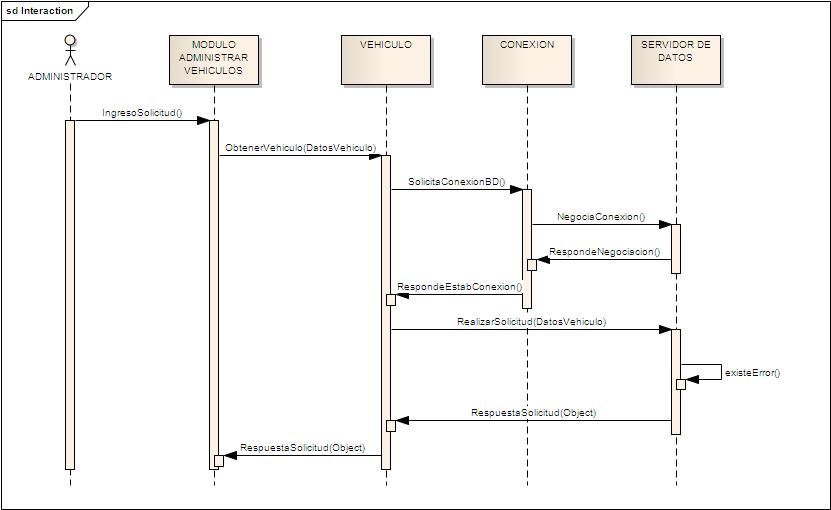


Figura - Diagrama Secuencia - Módulo Administrar Vehículos

*Vista 🡪 Lógica 🡪 Diagramas Clases*

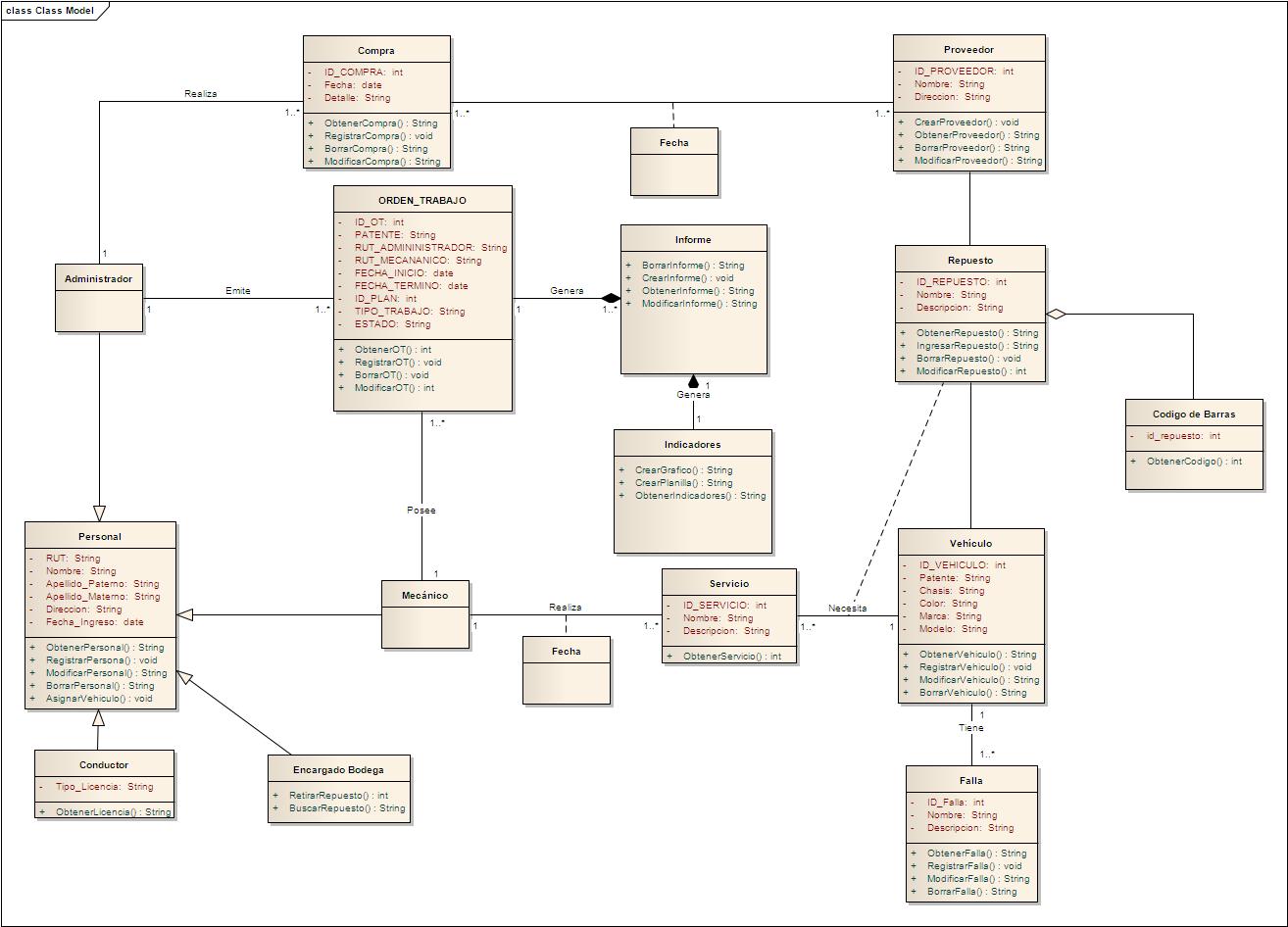


Figura - Diagrama de Clases

*Vista 🡪 Desarrollo 🡪 Diagrama de componentes*

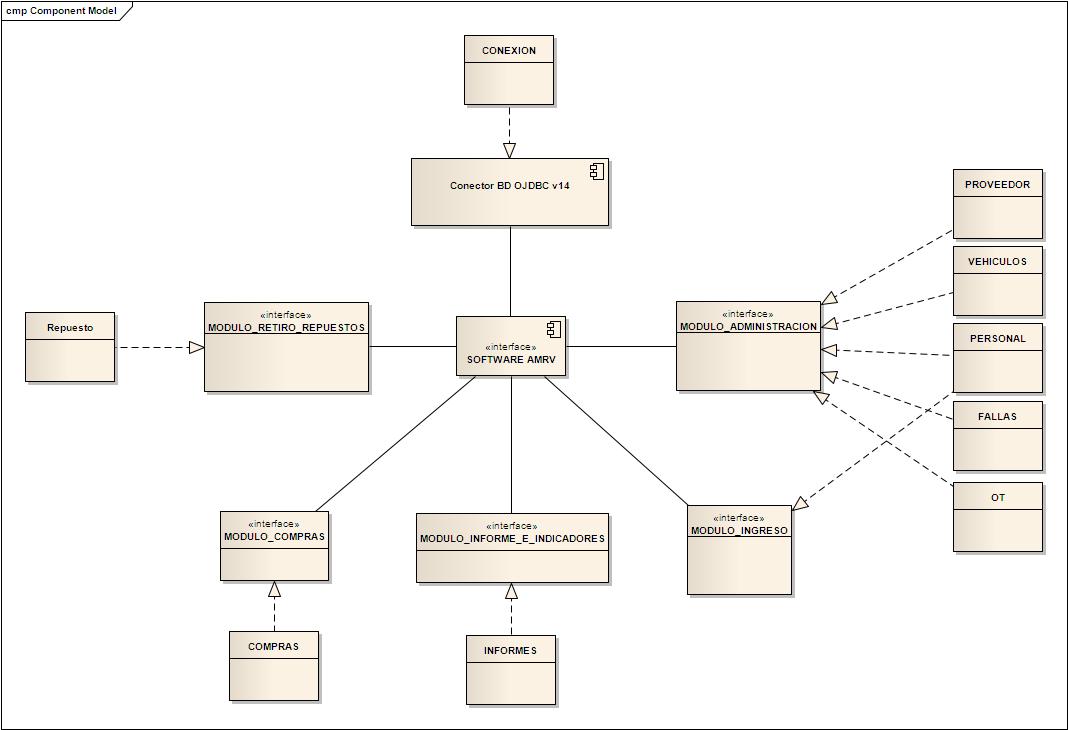


Figura - Diagrama de Componentes

*Vista 🡪 Física 🡪 Diagrama Despliegue*

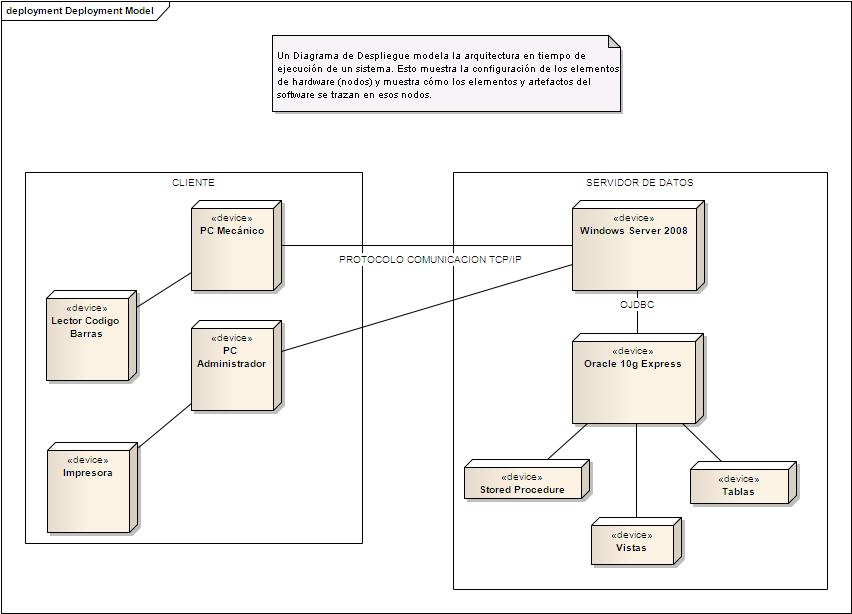


Figura - Diagrama Despliegue

## Modelo Entidad Relación

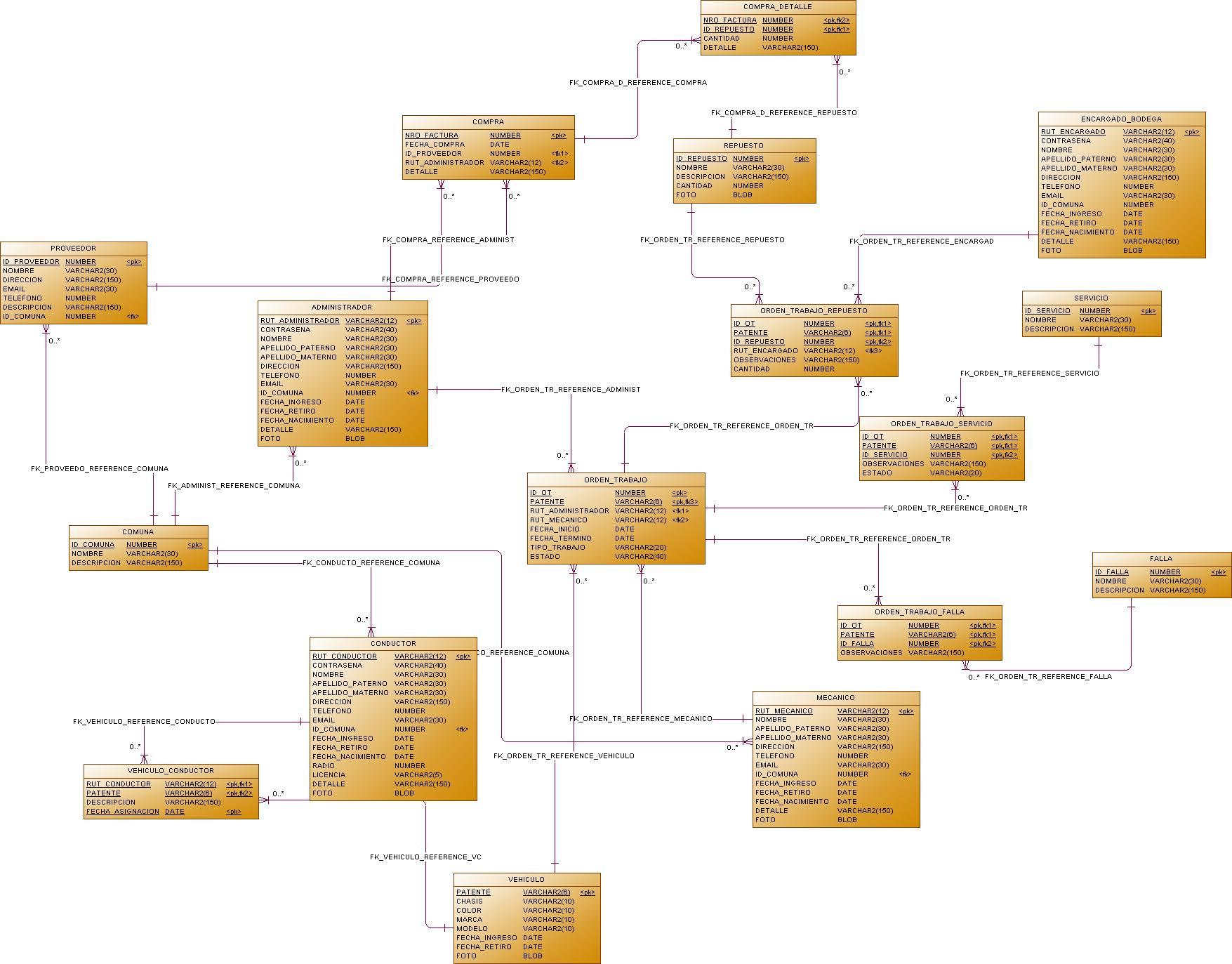


Figura - Modelo Entidad Relación

# Conclusiones

A través de *Kruchten 4+1* se ha logrado representar la arquitectura que el software *AMRV* necesitará para ser codificado, permitiendo hacer uso de todos los requerimientos funcionales que fueron otorgados por el documento anterior *IEEE 830.* El uso de los diferentes diagramas UML permitieron lograr un mayor entendimiento respecto a las distintas capas, clases, objetos, funcionalidades y componentes del software. A su vez, identificar las piezas de hardware que participaran en la implantación del proyecto obteniendo una visión más clara de lo que se va a entregar como producto final, entendiendo finalmente la coherencia y cohesión existente entre cada diagrama de *Kruchten 4+1* y el objetivo de cada uno de estos.

1. Con los Casos de uso se concluye que permite obtener una visibilidad más clara respecto a la funcionalidad que existirá dentro de cada requerimiento, además de los actores que estarán a cargo de cada módulo y la participación que tendrán dentro en ellos.
2. A través del diagrama de secuencia se logró entender la finalidad que tiene la interacción de cada objeto presente dentro de la secuencia de cada Caso de uso, como también, la participación y relevancia que comprenden las distintas clases para establecer comunicaciones entre los objetos. Este diagrama permitió dar cuenta de la necesidad que puede tener una clase para establecer la conexión con la base de datos al momento de querer realizar alguna operación dentro de algún módulo del software.
3. El diagrama de clases permitió comprender la lógica existente entre cada clase, además de los atributos y operaciones que posiblemente va a realizar cada una de estas participando dentro del software. Por otro lado, sirvió como apoyo para poder confeccionar el Modelo Entidad Relación *(MER)* el cual tiene como propósito modelar la arquitectura de la base de datos.
4. El diagrama de componentes aportó con el conocimiento y establecimiento de los distintos componentes de software que participaran, relacionando las interfaces presentes extraídas desde los módulos pertenecientes a los requerimientos funcionales, las dependencias existentes dentro de cada uno de estos y las clases a utilizar.
5. Finalmente, el diagrama de despliegue permitió entender cómo estarán conectadas las diferentes piezas de hardware pertenecientes al software y bajo que protocolo de comunicación se trabajará.
6. Se concluye finalmente que la etapa de diseño está finalizada obteniendo la arquitectura necesaria para poder avanzar a la siguiente fase (codificación) del modelo de desarrollo de software seleccionado. Respecto a la planificación, se puede concluir que hay un retraso de 3 días de trabajo.