

TWO WHEELS PARKING
Sistema de Gestión de biciparqueaderos

Daniela Córdoba Acosta
Johan Quiroga Torres
Santiago Jiménez Bonilla

Mayo 2018

Índice general

| | | |
|--------------|---|-----------|
| I | Contexto | 1 |
| 1. | Proyecto | 2 |
| 1.1. | Objetivos del Proyecto | 2 |
| 1.1.1. | Objetivo General | 2 |
| 1.1.2. | Objetivos Específicos | 2 |
| 2. | Metodologia | 3 |
| 2.1. | Introducción | 3 |
| 2.2. | Scrum - Cascada - Codificación y Reparación | 5 |
| 2.2.1. | Pre-Game | 5 |
| 2.2.1.1. | Planeación (Planning) | 5 |
| 2.2.1.2. | Staging (Escenificación) | 5 |
| 2.2.2. | In-Game | 5 |
| 2.2.2.1. | Proceso en Cascada (Waterfall) | 6 |
| 2.2.2.1.1. | Análisis (Analysis) | 6 |
| 2.2.2.1.2. | Diseño (Design) | 6 |
| 2.2.2.1.3. | Implementación (Implementation) | 7 |
| 2.2.2.1.3.1. | Codificación y Reparación | 7 |
| 2.2.2.1.4. | Prueba (Test) | 8 |
| 2.2.2.1.5. | Mantenimiento (Maintenance) | 8 |
| 2.2.3. | Post-Game | 8 |
| 2.2.3.1. | Release (Liberación) | 8 |
| 2.3. | Cronograma | 8 |
| 2.3.1. | Pre-Game | 9 |
| 2.3.2. | In-Game | 9 |
| 2.3.2.1. | Cascada | 9 |
| 2.3.3. | Post-Game | 10 |
| II | Arquitectura | 11 |
| 3. | Organizacion | 12 |
| 3.1. | Misión | 12 |
| 3.2. | Visión | 12 |

| | |
|---|-----------|
| 3.3. Estructura Organizacional | 13 |
| 3.4. Funciones de Negocio y Manual de funciones | 13 |
| 3.4.1. Recursos humanos | 13 |
| 3.4.2. Gerencia | 13 |
| 3.4.3. Seguridad | 13 |
| 3.4.4. Tesorería | 14 |
| 3.4.5. Administrativo y contable | 14 |
| 3.4.6. Ventas y mercadotecnia | 14 |
| 3.5. Procesos de Negocio | 15 |
| 3.6. Objetivos | 15 |
| 3.6.1. Organizacionales | 15 |
| 3.6.2. Operacionales | 15 |
| 3.6.3. Misionales | 15 |
| 3.6.4. Estratégicos | 16 |
| 3.7. Valores Organizacionales | 16 |
| 3.7.1. Amabilidad: | 16 |
| 3.7.2. Cumplimiento: | 16 |
| 3.7.3. Seguridad: | 16 |
| 3.7.4. Respeto: | 16 |
| 3.7.5. Honestidad: | 16 |
| 4. Achimate -ADM | 17 |
| 4.1. TOGAF | 20 |
| 4.1.1. Resumen: capa de negocio | 20 |
| 4.1.2. Resumen: capa de aplicación | 22 |
| 4.1.3. Resumen: capa de motivación | 23 |
| 4.1.4. Resumen: capa de proyecto | 24 |
| 4.1.5. Resumen: capa de tecnología | 24 |
| 4.1.6. Resumen Relaciones | 25 |
| 4.1.6.1. Relaciones Estructurales | 25 |
| 4.1.6.2. Relaciones Dinámicas | 26 |
| 4.1.6.3. Otras Relaciones | 27 |
| 5. Negocio | 28 |
| 5.1. Introducción | 28 |
| 5.2. Punto de Vista de Organización | 29 |
| 5.2.1. Descripción | 29 |
| 5.2.1.1. Metamodelo | 29 |
| 5.2.1.2. Caso de Estudio | 29 |
| 5.3. Punto de Vista de Cooperación de Actor | 30 |
| 5.3.1. Descripción | 30 |
| 5.3.1.1. Metamodelo | 31 |
| 5.3.1.2. Caso de Estudio | 31 |
| 5.4. Punto de Vista de Función de Negocio | 32 |
| 5.4.1. Descripción | 32 |
| 5.4.1.1. Metamodelo | 33 |

| | |
|--|-----------|
| 5.4.1.2. Caso de Estudio | 33 |
| 5.5. Punto de Vista de Proceso de Negocio | 34 |
| 5.5.1. Descripción | 34 |
| 5.5.1.1. Metamodelo | 35 |
| 5.5.1.2. Caso de Estudio | 35 |
| 5.6. Punto de vista de Cooperación de procesos de negocio. | 36 |
| 5.6.1. Descripción | 36 |
| 5.6.1.1. Metamodelo | 36 |
| 5.6.1.2. Caso de Estudio | 36 |
| 5.7. Punto de Vista de Producto | 37 |
| 5.7.1. Descripción | 37 |
| 5.7.1.1. Metamodelo | 38 |
| 5.7.1.2. Caso de Estudio | 38 |
| 6. Aplicación | 40 |
| 7. Capa de Aplicación | 41 |
| 7.1. Introducción | 41 |
| 7.2. Punto de Vista de Comportamiento de Aplicación | 41 |
| 7.2.1. Descripción | 41 |
| 7.2.1.1. Metamodelo | 42 |
| 7.2.1.2. Caso de Estudio | 42 |
| 7.3. Punto de Vista de Cooperación de Aplicación | 43 |
| 7.3.1. Descripción | 43 |
| 7.3.1.1. Metamodelo | 44 |
| 7.3.1.2. Caso de Estudio | 44 |
| 7.4. Punto de Vista de Uso de Aplicación | 45 |
| 7.4.1. Descripción | 45 |
| 7.4.1.1. Metamodelo | 46 |
| 7.4.1.2. Caso de Estudio | 46 |
| 7.5. Punto de vista de Estructura de Aplicación | 47 |
| 7.5.1. Descripción | 47 |
| 7.5.1.1. Metamodelo | 48 |
| 7.5.1.2. Caso de Estudio | 49 |
| 8. Tecnología | 50 |
| 9. Capa de Tecnologia | 51 |
| 9.1. Introducción | 51 |
| 9.2. Punto de Vista de Infraestructura | 51 |
| 9.2.1. Descripción | 51 |
| 9.2.1.1. Metamodelo | 52 |
| 9.2.1.2. Caso de Estudio | 52 |
| 9.3. Punto de Vista de Uso de Infraestructura | 53 |
| 9.3.1. Descripción | 53 |
| 9.3.1.1. Metamodelo | 54 |

| | |
|---|-----------|
| 9.3.1.2. Caso de Estudio | 54 |
| 9.4. Punto de Vista de Organización e implementación | 55 |
| 9.4.1. Descripción | 55 |
| 9.4.1.1. Metamodelo | 56 |
| 9.4.1.2. Caso de Estudio | 56 |
| 9.5. Punto de Vista de Estructura de Información | 57 |
| 9.5.1. Descripción | 57 |
| 9.5.1.1. Metamodelo | 58 |
| 9.5.1.2. Caso de Estudio | 58 |
| 9.6. Punto de Vista de Realización del Servicio | 59 |
| 9.6.1. Descripción | 59 |
| 9.6.1.1. Metamodelo | 60 |
| 9.6.1.2. Caso de Estudio | 60 |
| 9.7. Punto de Vista de Capas | 61 |
| 9.7.1. Descripción | 61 |
| 9.7.1.1. Caso de Estudio | 61 |
| 10. Motivación | 63 |
| 10.1. Introducción | 63 |
| 10.2. Punto de Vista de StakeHolder | 63 |
| 10.2.1. Descripción | 63 |
| 10.2.1.1. Metamodelo | 64 |
| 10.2.1.2. Caso de Estudio | 64 |
| 10.3. Punto de Vista de Realización de Objetivos | 65 |
| 10.3.1. Descripción | 65 |
| 10.3.1.1. Metamodelo | 65 |
| 10.3.1.2. Caso de Estudio | 65 |
| 10.4. Punto de Vista de Contribución | 66 |
| 10.4.1. Descripción | 66 |
| 10.4.1.1. Metamodelo | 67 |
| 10.4.1.2. Caso de Estudio | 67 |
| 10.5. Punto de Vista de Principios | 68 |
| 10.5.1. Descripción | 68 |
| 10.5.1.1. Metamodelo | 69 |
| 10.5.1.2. Caso de Estudio | 69 |
| 10.6. Punto de Vista de Realización de Requerimientos | 70 |
| 10.6.1. Descripción | 70 |
| 10.6.1.1. Metamodelo | 70 |
| 10.6.1.2. Caso de Estudio | 71 |
| 10.7. Punto de Vista de Motivación | 71 |
| 10.7.1. Descripción | 71 |
| 10.7.1.1. Metamodelo | 72 |
| 10.7.1.2. Caso de Estudio | 72 |

| | |
|--|---------------|
| 11.Migración | 73 |
| 11.1. Introducción | 73 |
| 11.2. Punto de Vista de Proyecto | 74 |
| 11.2.1. Descripción | 74 |
| 11.2.1.1. Metamodelo | 74 |
| 11.2.1.2. Caso de Estudio | 75 |
| 11.3. Punto de Vista de Migración | 75 |
| 11.3.1. Descripción | 75 |
| 11.3.1.1. Metamodelo | 76 |
| 11.3.1.2. Caso de Estudio | 76 |
| 11.4. Punto de Vista de Migración e Implementación | 76 |
| 11.4.1. Descripción | 76 |
| 11.4.1.1. Metamodelo | 77 |
| 11.4.1.2. Caso de Estudio | 77 |
| III Diseño | 78 |
| 12.Casos de USo | 79 |
| 13.Interacciones | 80 |
| 14.Clases | 81 |
| 15.Estados | 82 |
| 16.Actividades | 83 |
| 17.Sistemas | 84 |
| 18.Componentes | 85 |
| 19.Nodos | 86 |
| 20.Patrones GoF | 87 |
| 20.1. Patrones Creacionales | 87 |
| 20.1.1. Patrón Prototype | 87 |
| 20.1.1.1. Descripción | 87 |
| 20.1.1.1.1. Estructura | 88 |
| 20.1.1.1.2. Actores | 88 |
| 20.1.1.2. Caso de Uso | 88 |
| 20.1.2. Patrón Fabrica Abstracta | 88 |
| 20.1.2.1. Descripción | 88 |
| 20.1.2.1.1. Estructura | 89 |
| 20.1.2.1.2. Actores | 89 |
| 20.1.2.2. Caso de Uso | 89 |
| 20.2. Patrones de Comportamiento | 89 |

| | |
|--|--------|
| 20.2.1. Patrón Strategy | 90 |
| 20.2.1.1. Descripción | 90 |
| 20.2.1.1.1. Estructura | 90 |
| 20.2.1.1.2. Actores | 90 |
| 20.2.1.2. Caso de Uso | 91 |
| 20.2.2. Patrón State | 91 |
| 20.2.2.1. Descripción | 91 |
| 20.2.2.1.1. Estructura | 91 |
| 20.2.2.1.2. Actores | 91 |
| 20.2.2.2. Caso de Uso | 91 |
| 20.2.3. Patrón Cadena de Responsabilidad | 91 |
| 20.2.3.1. Descripción | 91 |
| 20.2.3.1.1. Estructura | 92 |
| 20.2.3.1.2. Actores | 92 |
| 20.2.3.2. Caso de Uso | 92 |
| 20.3. Patrones Estructurales | 92 |
| 20.3.1. Patrón Bridge | 93 |
| 20.3.1.1. Descripción | 93 |
| 20.3.1.1.1. Estructura | 93 |
| 20.3.1.1.2. Actores | 94 |
| 20.3.1.2. Caso de Uso | 94 |
| 20.3.2. Patrón Proxy | 94 |
| 20.3.2.1. Descripción | 94 |
| 20.3.2.1.1. Estructura | 95 |
| 20.3.2.1.2. Actores | 95 |
| 20.3.2.2. Caso de Uso | 95 |
| IV Reflexiones | 96 |
| 21.Conclusiones y Trabajos futuros | 97 |

Parte I

Contexto

Capítulo 1

Proyecto

1.1. Objetivos del Proyecto

1.1.1. Objetivo General

Automatizar y optimizar los principales procesos realizados por el sistema actual de parqueaderos para bicicletas de la empresa Two Wheels Parking. Además de implementar herramientas informáticas pertinentes a las necesidades de quienes quieren hacer uso de este servicio, mejorando así, su calidad como usuarios activos del servicio.

1.1.2. Objetivos Específicos

- Optimizar los principales procesos que hacen parte del propósito de brindar servicios de parqueo para las bicicletas en las diferentes sucursales que maneja Two Wheels Parking, a través de una planeación y modelamiento de dichos procesos con el fin de mejorar el uso de los espacios destinados al almacenamiento de bicicletas.
- Automatizar los procesos de registro y consulta de información pertinente, a través de la elaboración de herramientas informáticas con el propósito de incrementar la eficiencia del servicio de biciparqueaderos de la organización.
- Generar un software confiable, robusto y escalable, por medio de la elaboración de modelos utilizando como herramienta los patrones de diseño de software, para mejorar el sistema actual de manejo de parqueaderos y realizar aportes a futuras mejoras a dicho sistema.

Capítulo 2

Metodologia

2.1. Introducción

Se eligió la metodología **SCRUM** dado que es una de las más utilizadas a nivel empresarial y con ella se puede llegar a entregar un producto de calidad optimizando el tiempo de desarrollo, además de esto, es importante resaltar que hace el uso de un concepto propio como es del sprint, el cual es una carrera corta o se podría ver como un subproceso, y consideramos que es importante dividir un proceso en subprocesos para poder llevar a cabo correctamente una dinámica colaborativa para obtener buenos resultados con calidad y agilidad. En la fase In-Game, se decidió utilizar el proceso en cascada dado que es simple y lineal, es decir que cada una de las etapas se complementa de la anterior para poder cumplir su funcionalidad dentro del modelo, además de esto, es importante resaltar la parte de la documentación en este proceso dado que en cada etapa se producen informes con el fin de hacer que la comprensión del procedimiento del producto a diseñar sea más sencillo. Además de esto el modelo en cascada posee una gran ventaja como lo es la planificación fácil y clara por parte de los desarrolladores y el usuario, y para complementar la fase de implementación de cascada se utilizará el proceso de Codificación y Reparación.

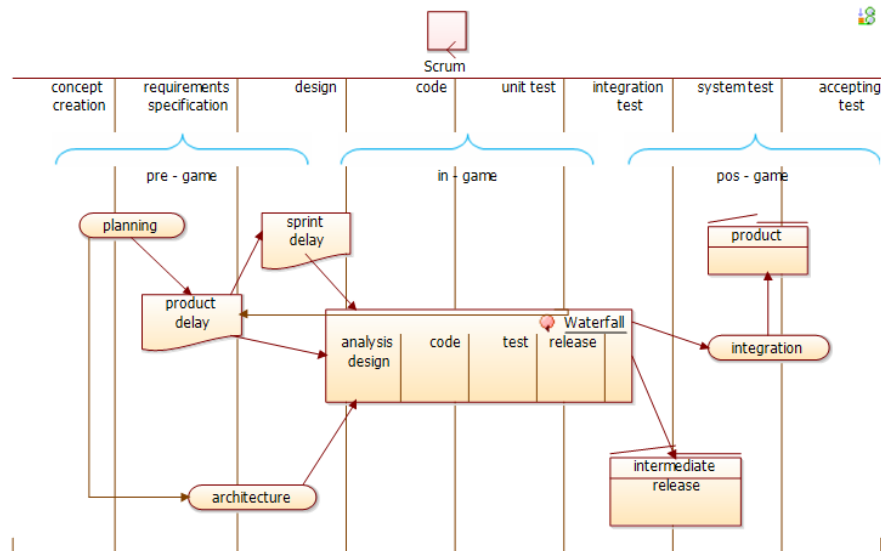


Figura 2.1: Metodología Scrum.

Haciendo énfasis en todo lo que conlleva hacer uso de esta metodología para el desarrollo de nuestro proyecto, de debe hablar de que se realizará en cada una de las etapas planteadas en esta metodología, también se mostrará el cronograma relacionado con esta metodología y los procesos a utilizar, pero primero que todo se definirá el Scrum Master y Team Members.

Scrum Master: la persona encargada de cumplir las tareas de un Scrum Master será Andres Restrepo, dado que por sus cualidades califica perfectamente para llevar a cabo este cargo, además tiene un perfil de un líder dado que supervisará, verificará y será el facilitador del equipo de trabajo.

Team members: las personas que conformarán el Team Members serán Daniel Casas, Brayan Esguerra, Andres Restrepo Que tendrán la tarea enfocada hacia el desarrollo del software y los cuales tienen las capacidades técnicas para poder desarrollar el producto a cabalidad.

Planeación del Sprint: En este proceso el Scrum Master y el Team Members se reunirán para planear las actividad o actividades que se encuentran inmersas dentro del sprint a realizar es decir se deben dar prioridad dentro de estas actividades y dar todas la información pertinente como lo es el tiempo existente para esta iteración para poder tener referencias para el próximo sprint.

Reunión diaria de Scrum: En cuanto a las reuniones diarias en este proyecto cambiaremos un poco ese concepto dado que aunque se realizarán diariamente,

no al inicio del día, sino que por el contrario se hará una reunión finalizando el día para saber que los miembros que participan en estas reuniones puedan hacer y responder preguntas como ¿Qué se hizo durante el día?, ¿Qué se planea hacer el día siguiente? Y saber si se ha presentado algún tipo de inconveniente en el proceso de alcanzar el objetivo además estas reuniones tendrán un intervalo de durabilidad de entre unos 15 a 20 minutos diarios.

Revisión del sprint: Cada vez que se dé por terminado un sprint se debe mostrar lo que el equipo ha logrado, es decir un producto terminado referente a ese sprint que se estaba trabajando para saber si esta óptimo o si se deben hacer algunas correcciones esto se analizará dentro del grupo de trabajo y el dueño del producto, aclarando que los sprints serán realizados con las características de el proceso en Cascada.

2.2. Scrum - Cascada - Codificación y Reparación

2.2.1. Pre-Game

En esta sección se especificará que se hará durante las dos fases que la componen como lo son la planeación y la escenificación.

2.2.1.1. Planeación (Planning)

Durante este segmento o fase se realiza una reunión entre todos los entes involucrados en el desarrollo de este proyecto como lo son the power owner, scrum master y team members para poder definir la visión, la expectativas y el presupuesto con el que se cuenta para el desarrollo de este además de esto the power owner debe de lanzar la pila de producto que contengan los elementos suficientes para poder entrar a un primer sprint.

2.2.1.2. Staging (Escenificación)

Durante esta fase se tomara en cuenta lo especificado o planteado en la fase anterior es decir en la fase de planeación dado que se tiene que tener claro el presupuesto y los requisitos planteados, es importante traer de la fase de planeación la lista de producto dado que por medio de esta se llegará al planteamiento de los requerimientos los cuales son de suma importancia para el desarrollo de nuestro proyecto. En esta fase se verán involucrado tanto el Scrum Master como el Team Members dado que entre ellos se analizará la lista de producto para poder llegar a unos requerimientos que satisfagan el proyecto.

2.2.2. In-Game

En esta sección se especifica la duración de cada uno de los sprints por medio de el procesos cascada a el cual se le ha integrado el proceso de codificación y

reparación.

2.2.2.1. Proceso en Cascada (Waterfall)

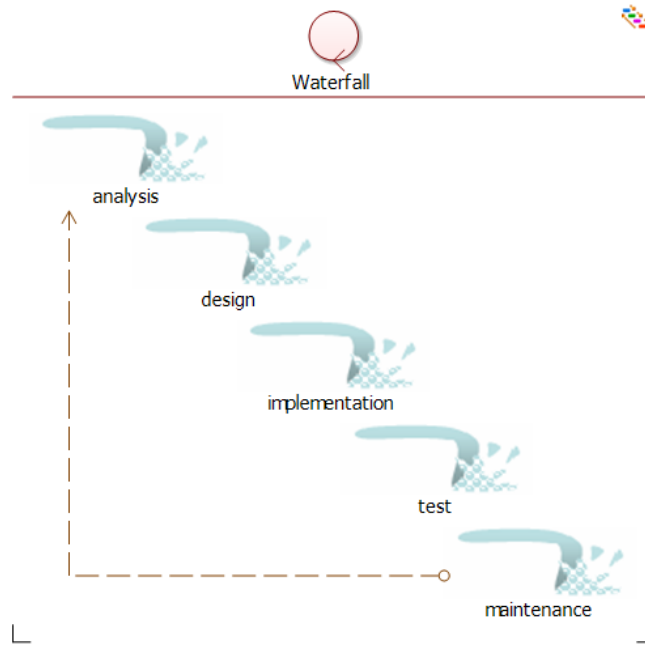


Figura 2.2: Proceso Waterfall

2.2.2.1.1. Análisis (Analysis) Durante esta etapa se tendrán en cuenta los requisitos del sistema planteados por el cliente. Además de esto una parte importante de esta etapa hace referencia al desarrollo de cada uno de los requerimientos, es decir que a través de la información suministrada por el cliente se plantearán cada uno de los requerimientos y se desarrollará la interacción, es decir los casos de uso, casos de secuencia, casos de comunicación y workflow, para así tener planteamientos específicos, concretos y sin lugar a ambigüedades.

2.2.2.1.2. Diseño (Design) Luego de finalizada la etapa de análisis se tomarán los desarrollos realizados en requerimientos e interacciones, ya que por medio de ellos se planteará un diseño que le brinde solución a la situación problema planteada por el cliente, es así como se tomará cada uno de los requerimientos y se les aplicará este mismo proceso de diseño y solución para poder pasar a la siguiente etapa la cual es la de implementación y poder facilitar este proceso por medio de un diseño claro y conciso.

2.2.2.1.3. Implementación (Implementation) En esta etapa se hará uso de un nuevo proceso el cual es CODIFICACIÓN Y REPARACIÓN con el cual se busca utilizar cada uno de los diseños o soluciones planteadas y por medio de algoritmia o codificación poder plasmar estas soluciones a nivel de código.

2.2.2.1.3.1. Codificación y Reparación Se utilizará el proceso de codificación y reparación dado que es un modelo simple que está orientado a la parte de desarrollo, pues que encaja perfectamente dentro de la etapa de implementación dentro de la cual se encuentra inmersa, a pesar de que no está compuesta por una etapa de prueba, y es un buen complemento para el correcto y efectivo desarrollo de la codificación.

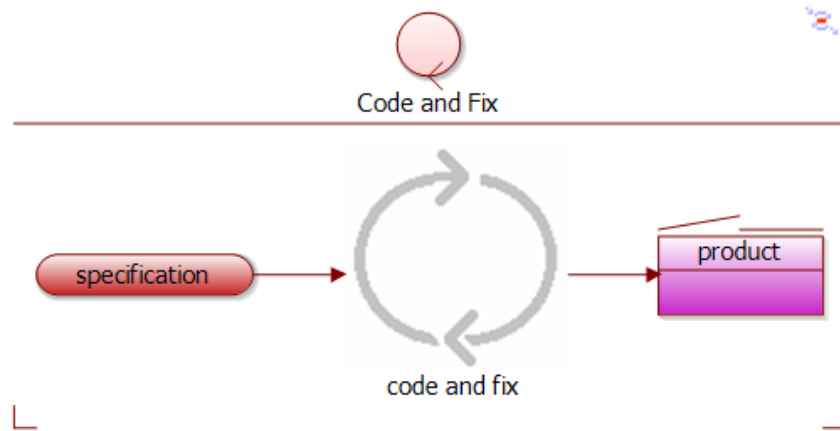


Figura 2.3: Proceso Code and Fix

Especificación En esta etapa se hará uso de los requerimientos y los diseños planteados en la etapa de análisis y diseño del proceso en cascada los cuales se analizarán para poder seguir adelante en este proceso y llevar esto a la siguiente etapa la cual es Codificación y Reparación.

Codificación y Reparación Durante esta etapa se realizan constantemente ciclos en los cuales se hará uso de esa especificación ya realizada anteriormente para poder tomar cada diseño o solución planteada y así poder traducir esto a un lenguaje que la maquina entienda.

Producto Después de dar como finalizada la etapa anterior en esta fase se tendrá el producto final, el cual en este caso será la implementación de un diseño y una solución en particular para que de esta manera sea posible abarcar en su totalidad todas los diseños de las soluciones planteadas.

2.2.2.1.4. Prueba (Test) Luego del desarrollo de las soluciones de diseño y su implementación se realizarán pruebas necesarias para poder verificar que nuestro producto final tenga la calidad necesaria para que la cantidad de errores sea mínima con el objetivo de reducir el tiempo de trabajo en la etapa de mantenimiento, además de esto se harán uso de ciertas pruebas como lo son:

Prueba unitaria. Prueba de integración. Pruebas del sistema. Prueba de aceptación.

2.2.2.1.5. Mantenimiento (Maintenance) En este caso enfocaremos esta etapa a la corrección de los errores encontrados en el producto final, en un dado caso que en la etapa anterior se presente algún tipo de error, se manejará un mantenimiento preventivo y perfectivo dado que por medio este se busca prever posibles errores a futuro direccionándonos a obtener un producto de una muy buena calidad que satisfaga las necesidades del cliente propuestas en la etapa de análisis con un mínimo de requisitos de hardware.

2.2.3. Post-Game

En esta sección se especificarán las partes que componen esta fase como lo es el release o liberación.

2.2.3.1. Release (Liberación)

En esta etapa se busca dar y entregar el producto final utilizando conjuntamente el concepto de integración, dado que se utilizara para poder unir los subsistemas creados en el in-game y por medio de este término poder integrarlos y unirlos para tener un producto final integrado y finalizado cumpliendo exitosamente lo propuesto al inicio de nuestro proyecto.

2.3. Cronograma

Cabe resaltar que el cronograma planteado está directamente relacionado con nuestro desarrollo de software el cual incluye la metodología Scrum y los procesos en Cascada y, codificación y reparación.

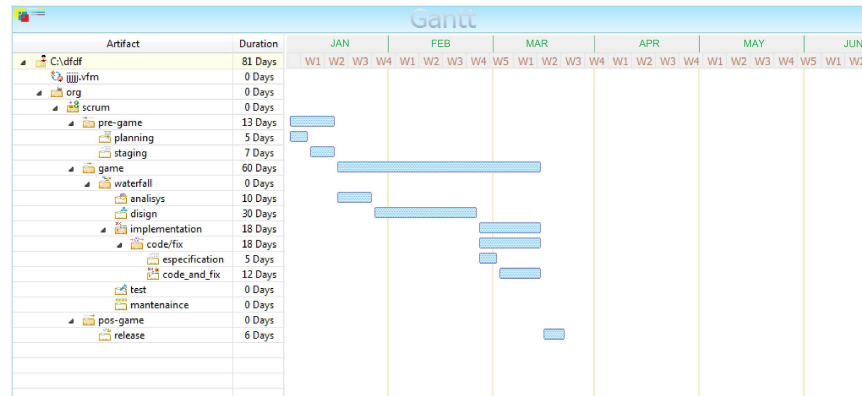


Figura 2.4: Cronograma de metodología Scrum

2.3.1. Pre-Game

Esta fase tiene una duración de 13 días, en los cuales se buscara cumplir cada una de las funciones que tienen cada una de las etapas que componen el pre-game estos días están distribuidos de la siguiente forma:

- **Planificación:** esta etapa abarca un total de 5 días en los cuales se buscara cumplir la meta propuesta anteriormente.
- **Escenificación:** esta etapa cuenta con un total de 7 días en los cuales se buscara cumplir las tareas propuestas anteriormente.

Dentro de esta fase podemos encontrar un primer sprint el cual se desarrollara durante la etapa de escenificación.

2.3.2. In-Game

Esta fase tiene una duración de 60 días en los cuales se buscara cumplir cada una de las funciones que tienen cada una de las etapas que componen el pre-game y estos días están distribuidos de la siguiente forma con el uso de los procesos mencionados anteriormente:

2.3.2.1. Cascada

- **Análisis:** esta etapa cuenta con un total de 10 días en los cuales se buscará cumplir y analizar lo propuesto en la escenificación y la codificación.
- **Diseño:** esta etapa abarca un total de 30 días en los cuales se buscará cumplir con los diseños de software.

- **Implementación:** esta etapa abarca un total de 18 días en los cuales se usará el proceso de Codificación y Reparación para llevar a cabo el desarrollo de esta.
- **Prueba:** esta etapa no cuenta con días en el calendario dado que no se llegará a esta.
- **Mantenimiento:** al igual que en la etapa de prueba, no se realizará dado que no se llegará a ese punto.

Dentro de esta fase encontramos 3 sprints de tipo macro ubicado en cada una de las etapas como lo son las rápidas cascadas que se realizarán y cada uno de esos macro-sprint se componen por varios micro-sprint los cuales son el proceso de Codificación y Reparación, esto está relacionado con el número de requerimientos los cuales se pondrán a ser procesados en cada una de estas etapas.

2.3.3. Post-Game

Esta fase tiene una duración de 6 días en los cuales se buscará cumplir cada una de las funciones que tiene la liberación:

- **Liberación:** esta etapa cuenta con un total de 6 días en los cuales se entregará el producto.

Parte II

Arquitectura

Capítulo 3

Organizacion



Figura 3.1: Two Wheels Parking

3.1. Misión

En Two Wheels Parking ofrecemos soluciones de aparcamiento para bicicletas en la ciudad de Bogotá, ofreciendo una experiencia de tranquilidad, seguridad y organización a nuestros clientes, buscando incentivar el uso de este medio de transporte como una alternativa limpia y segura.

3.2. Visión

Queremos estar comprometidos con los problemas de nuestros clientes de forma transparente y eficaz, convirtiéndonos en un socio de confianza. En Two Wheels Parking queremos ser un referente como una empresa líder en soluciones de aparcamiento, dando a conocer nuestra marca como una empresa ética,

responsable y solidaria con nuestros clientes.

3.3. Estructura Organizacional

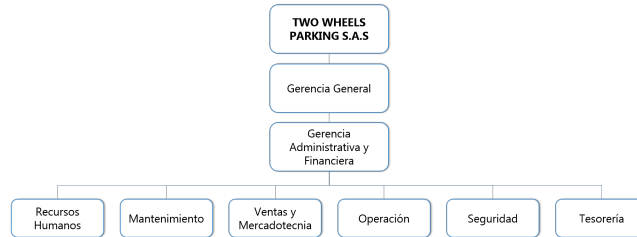


Figura 3.2: Organigrama General de Two Wheels Paking.

3.4. Funciones de Negocio y Manual de funciones

3.4.1. Recursos humanos

- Trabajadora social: Profesional encargado del desarrollo de vínculos humanos saludables y buenas relaciones sociales entre los individuos que laboran al interior de la empresa.

3.4.2. Gerencia

- Presidente: Representante, líder y supervisor en la toma de decisiones que guían el rumbo de la compañía.
- Secretaria: Principal colaboradora del presidente en el área administrativa, dentro de su rol es la encargada de la gestión de la documentación empresarial y de la atención al público.

3.4.3. Seguridad

- Guardia de seguridad: personal competente para realizar actividades de vigilancia, inspección, prevención y detección de anomalías al interior de la Institución.

3.4.4. Tesorería

- Tesorero: Encargado de gestionar y dirigir los asuntos relacionados con movimientos económicos o flujos monetarios, tanto captación como desembolsos dentro de la empresa.

3.4.5. Administrativo y contable

- Contador Público: Profesional encargado de realizar y verificar los registros contables, tributarios y financieros de la empresa, generando los informes y documentos legales exigidos por la legislación nacional.
- Administrador: Responsable de la gestión de recursos materiales y financieros dedicados a la realización y control de las distintas actividades tanto técnica como administrativas mediante una correcta planificación, coordinación y ejecución.

3.4.6. Ventas y mercadotecnia

- Agentes de ventas: Profesionales capaces de captar y aumentar el número y la calidad de clientes a los cuales la empresa busca prestar una serie de soluciones.
- Publicista: Experto en supervisar, coordinar y ejecutar estrategias de publicidad y posicionamiento de imagen tanto de la empresa como de los productos que se generan al interior de ella.

3.5. Procesos de Negocio

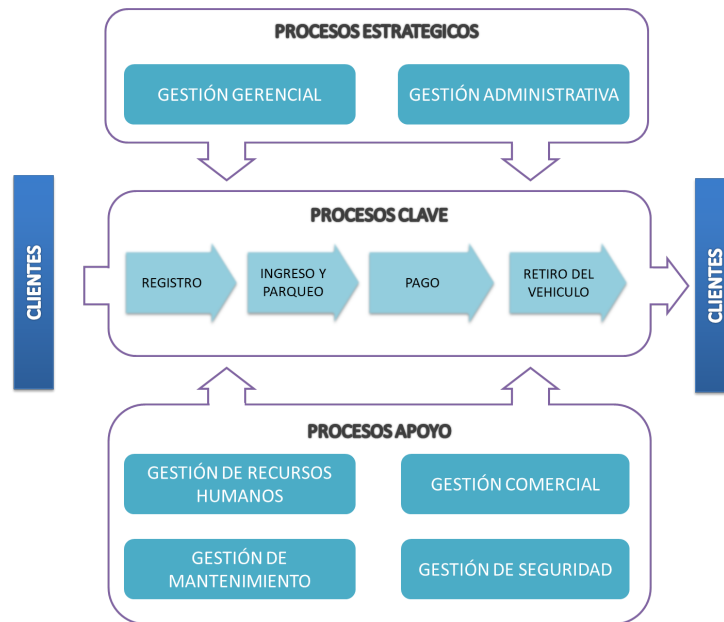


Figura 3.3: Mapa de Procesos de Two Wheels Paking.

3.6. Objetivos

3.6.1. Organizacionales

- Alcanzar un amplio mercado a nivel nacional convirtiéndonos en una de las empresas pioneras de soluciones en colaboración con el medio ambiente, brindando soluciones de parqueo a aquellas personas que hacen uso de un transporte ecológico, esto con el fin de fomentar el uso de dicho transporte y contribuir al desarrollo sustentable.

3.6.2. Operacionales

- Brindar la mejor calidad de servicio a nuestros clientes, mediante un equipo de trabajo altamente capacitado, convirtiéndonos en una de las empresas distritales con uno de los mejores “Goodwill”.

3.6.3. Misionales

- Enfocar todos nuestros procesos al uso eficiente de los insumos y recursos, esto mediante el fomento de responsabilidad social tanto al interior como

al exterior de la empresa en nuestros empleados, generando así un Impacto ambiental positivo a partir de la producción de nuestros productos y servicios.

3.6.4. Estratégicos

- Posicionar a la empresa y su marca Two Wheels como una franquicia respetable de parqueo, esto mediante la estandarización y normalización de nuestros procesos, estableciendonos y generando un impacto primeramente a nivel distrital y posteriormente nacional.

3.7. Valores Organizacionales

3.7.1. Amabilidad:

En Two Wheels queremos prestar un servicio de calidad al cliente, por lo cual contamos con un equipo de trabajo capacitado el cual atenderá sus inquietudes y propuestas de la mejor manera.

3.7.2. Cumplimiento:

Sabemos que el tiempo y el dinero son recursos importantes, por eso en Two Wheels nos regimos bajo estrictos estándares y políticas internas para que las actividades estipuladas se lleven a cabo en los tiempos predefinidos, los que nos convierte en una empresa sólida y seria de cara a nuestros clientes.

3.7.3. Seguridad:

En Two Wheels nos comprometemos con nuestros usuarios para brindar un ambiente seguro para sus vehículos mediante un control de acceso de usuarios efectivo e instalaciones que cuenten con una infraestructura adecuada para proteger a cada usuario y su vehículo.

3.7.4. Respeto:

En Two Wheels creemos que el pilar de toda organización es el respeto mutuo. Por eso, nuestro equipo está enfocado en brindar el mejor servicio a todos nuestros clientes, sin ninguna excepción.

3.7.5. Honestidad:

Como parte de nuestra misión, en Two Wheels creemos en el manejo transparente de nuestros procesos, ofreciendo a nuestros clientes un servicio en el cual puedan confiar.

Capítulo 4

Achimate -ADM

El método ADM y en general el marco TOGAF realiza el análisis arquitectónico con alto nivel de abstracción para visualizar, detectar y documentar oportunidades y riesgos durante el desarrollo de la arquitectura, direccionando la arquitectura empresarial con la ayuda de sus herramientas. TOGAF esta compuesto por 3 partes principales el método de desarrollo ADM (Architecture Development Method), la taxonomía empresarial (Enterprise Continuum) y la base de recursos (Architecture Repository). Aborda el desarrollo a partir de 4 niveles de abstracción:

- Arquitectura de Negocio
- Arquitectura de Aplicación
- Arquitectura de Datos
- Arquitectura Tecnológica

ADM muestra estos niveles de abstracción en diferentes fases que determinan la linea base (baseline) y el final del nivel de abstracción (target), donde el analisis de brecha (gap analysis - figura 5.1) permite conocer el estado final de la arquitectura despues de una o varias iteraciones.

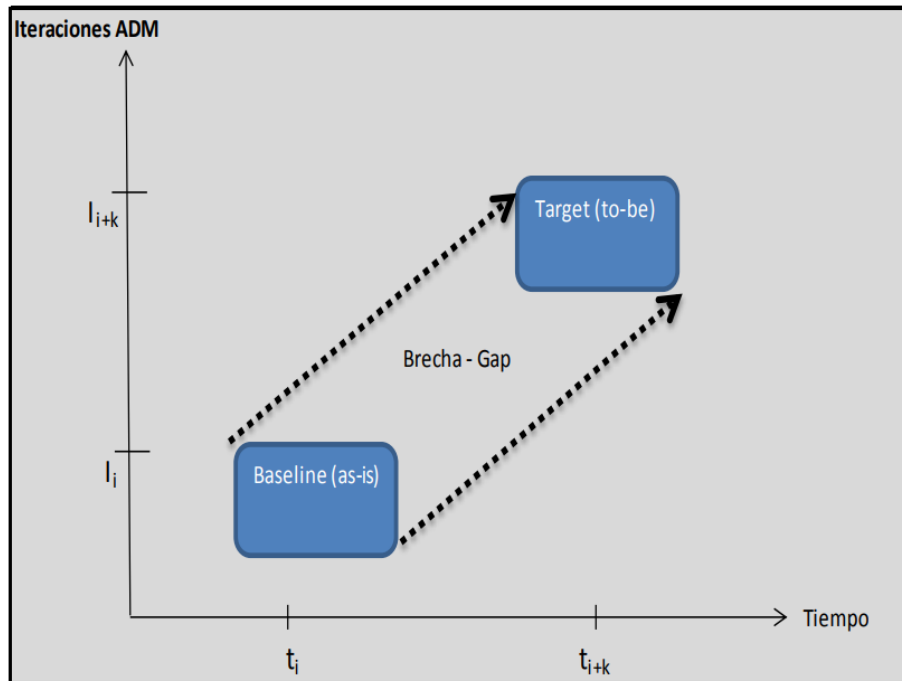


Figura 4.1: Analisis de Brecha -Iteraciones ADM

Este análisis mide los objetivos de arquitectura y el grado de la madurez alcanzados por la organización

ADM consta de 8 niveles o etapas y un paso preliminar donde se describen las actividades iniciales, principios y capacidades de la arquitectura objetivo, también se realiza una adaptación del marco de trabajo para ajustarlo a las necesidades de la organización. [2]

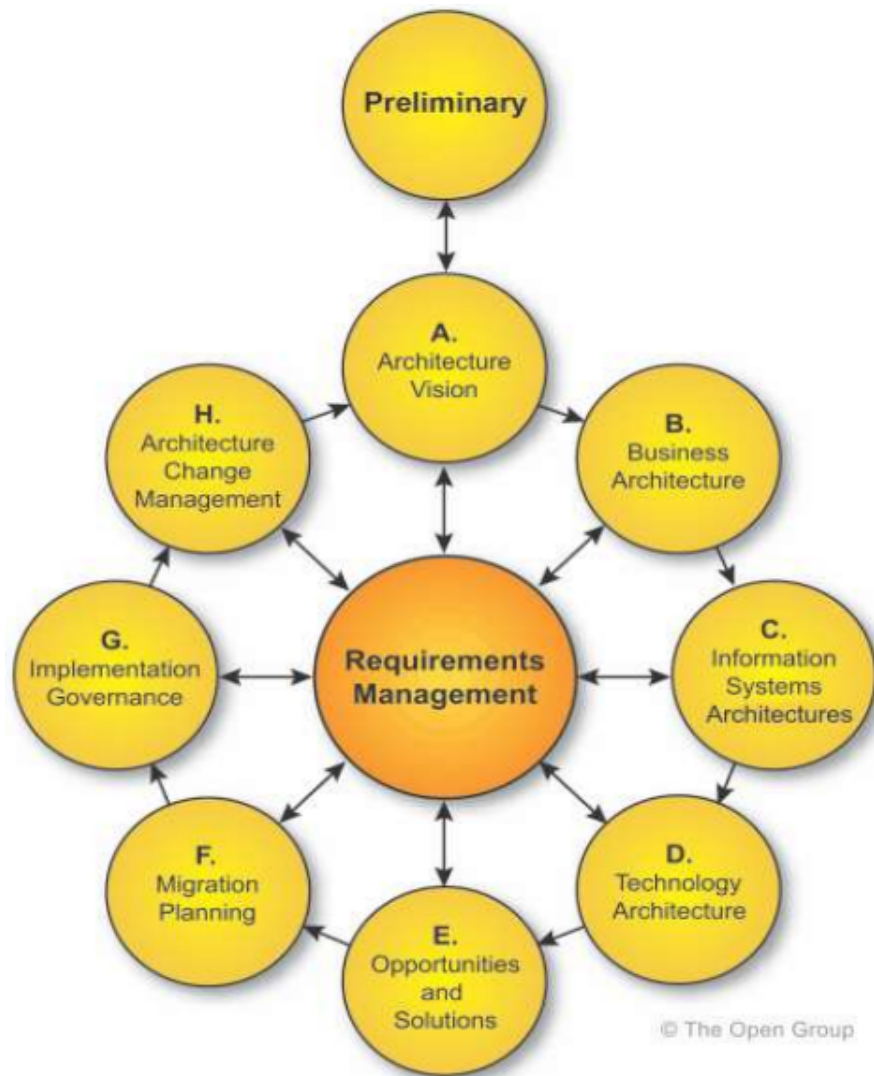


Figura 4.2: Etapas del Método ADM




















Las primeras cuatro fases definen los niveles de abstracción antes mencionados para el desarrollo de la arquitectura, la fase E de la interacción define oportunidades y soluciones que se deben implementar.








Estas oportunidades y soluciones se identifican e incluyen en el plan de migración, fase F. En esta fase se desarrolla el producto de software con los requerimientos y especificaciones obtenidos de las anteriores fases, luego lleva a su implementación (fase G) y como finalidad lleva la arquitectura de un estado base (baseline) al estado objetivo (target), estas dos fases dan gobernabilidad

y gestión de control de cambios a la arquitectura. Este ciclo se repite hasta llegar a la vision establecida en la fase preliminar junto con la vision inicialmente concebida.

4.1. TOGAF


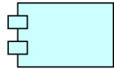










4.1.1. Resumen: capa de negocio

| Concepto | Definición | Notación |
|--|--|---|
| Actor de negocio (Business actor) | Una entidad organizativa que es capaz de realizar el comportamiento. |   |
| Papel del negocio (Business role) | La responsabilidad de realizar un comportamiento específico, al cual un actor puede ser asignado. |   |
| Colaboración Empresarial (Business Collaboration) | Un agregado de dos o más funciones empresariales que trabajan juntas para realizar un comportamiento colectivo. |   |
| Interfaz de negocios (Business interface) | Un punto de acceso donde un servicio comercial está disponible para el medio ambiente.. |   |
| Ubicación (Location) | Un punto conceptual o extensión en el espacio. |   |
| Objeto de negocio (Business Object) | Un elemento pasivo que tiene relevancia desde una perspectiva empresarial. |  |
| Proceso de negocio (Business Process) | Un elemento de comportamiento que agrupa el comportamiento basado en un ordenamiento de actividades. Se pretende producir un conjunto definido de productos o servicios empresariales. |   |
| Función de negocio (Business Function) | Un elemento de comportamiento que agrupa el comportamiento basado en un conjunto seleccionado de criterios (normalmente requeridos por los recursos empresariales y / o las competencias). |   |
| Interacción de negocios (Business Interaction) | Un elemento de comportamiento que describe el comportamiento de una colaboración comercial. |   |
| Evento de negocios (Business Event) | Algo que sucede (internamente o externamente) e influye en el comportamiento. |   |

| Concepto | Definición | Notación |
|---|--|---|
| Servicio de negocio o empresarial (Business Service) | Un servicio que satisface una necesidad de negocio para un cliente (interno o externo a la organización). |   |
| Representación (Representation) | Una forma perceptible de la información transportada por un objeto de negocio. |  |
| Significado (Meaning) | Los conocimientos o experiencia presentes en un objeto de negocio o su representación, dado un contexto particular. |  |
| Valor (Value) | El valor relativo, la utilidad o la importancia de un servicio o producto comercial. |  |
| Producto (Product) | Un conjunto coherente de servicios, acompañado de un contrato / conjunto de acuerdos, que se ofrece en su conjunto a clientes (internos o externos). |  |
| Contrato (Contract) | Una especificación formal o informal de acuerdo que especifica los derechos y obligaciones asociados con un producto. |  |


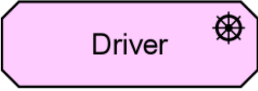


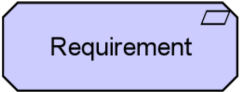
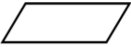
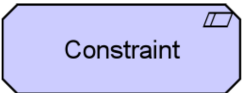

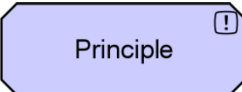
Cuadro 4.1: Capa de Negocio

4.1.2. Resumen: capa de aplicación

| Concepto | Definición | Notación |
|--|---|---|
| Componente de Aplicación (Application Component) | Una parte modular, desplegable y reemplazable de un sistema de software que encapsula su comportamiento y datos y los expone a través de un conjunto de interfaces. |   |
| Colaboración de Aplicación (Application collaboration) | Un agregado de dos o más componentes de aplicación que trabajan juntos para realizar un comportamiento colectivo. |   |
| Interfaz de Aplicación (Application interface) | Un punto de acceso donde un servicio de aplicación está disponible para un usuario u otro componente de aplicación |   |
| Objeto de Datos (Data Object) | Un elemento pasivo adecuado para el procesamiento automatizado. |  |
| Función de Aplicación (Function Application) | Un elemento de comportamiento que agrupa el comportamiento automatizado que puede ser realizado por un componente de aplicación. |   |
| Interacción de Aplicación (Application interaction) | Un elemento de comportamiento que describe el comportamiento de una colaboración de aplicación |   |
| Servicio de Aplicación (Application Service) | Un servicio que expone un comportamiento automatizado |  |

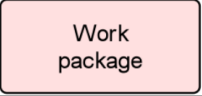



Cuadro 4.2: Capa de Aplicación

4.1.3. Resumen: capa de motivación

| Concepto | Definición | Notación |
|------------------------------|---|---|
| Interesados (Stakeholder) | El rol de un individuo, equipo u organización (o clases de ellos) que representa sus intereses o preocupaciones, relativas al resultado de la arquitectura. |  |
| Conductor (Driver) | Algo que crea, motiva y alimenta el cambio en una organización. |  |
| Evaluación (Assessment) | El resultado de algunas evaluaciones de algunos conductores. |  |
| Gol (Goal) | Un estado final que una parte interesada desea lograr. |  |
| Requisito (Requirement) | Una declaración de necesidad que debe ser realizada por un sistema. |   |
| Restricción (Constraint) | Una restricción de la forma en que se realiza un sistema. |   |
| Principio (Principle) | Una propiedad normativa de todos los sistemas en un contexto dado, o la forma en que se realizan. |  |

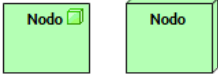



Cuadro 4.3: Capa de Motivación

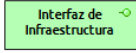

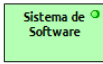
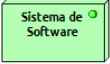
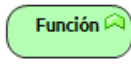


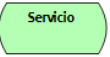
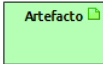
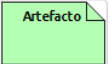
4.1.4. Resumen: capa de proyecto

| Concepto | Definición | Notación |
|--------------------------------------|--|---|
| Paquete de Trabajo (Work Package) | Una serie de acciones diseñadas para lograr una meta única dentro de un tiempo especificado. |  |
| Entregable (Derivable) | Un resultado definido con precisión de un paquete de trabajo (work package). |  |
| Meseta (Plateau) | Un estado relativamente estable de la arquitectura que existe durante un período de tiempo limitado. |  |
| Brecha (Gap) | Resultado de un análisis de la brecha entre dos mesetas (plateaus). |  |

Cuadro 4.4: Capa de Proyecto

4.1.5. Resumen: capa de tecnología


| Concepto | Definición | Notación |
|----------------------|---|---|
| Nodo | Un recurso computacional sobre el cual los artefactos pueden ser almacenados o desplegados para su ejecución. |  |
| Dispositivo | Un recurso de hardware en el que los artefactos se pueden almacenar o desplegar para su ejecución. |  |
| Red | Un medio de comunicación entre dos o más dispositivos. |  |
| Ruta de Comunicación | Un enlace entre dos o más nodos, a través del cual estos nodos pueden intercambiar datos. |  |







| Concepto | Definición | Notación |
|-----------------------------|--|---|
| Interfaz de infraestructura | Un punto de acceso donde los servicios de infraestructura ofrecidos por un nodo pueden ser accedidos por otros nodos y componentes de la aplicación. |   |
| Software del Sistema | Un entorno de software para tipos específicos de componentes y objetos que se despliegan en él en forma de artefactos. |   |
| Función de Infraestructura | Un elemento de comportamiento que agrupa el comportamiento de infraestructura que puede ser realizado por un nodo. |   |
| Servicio de Infraestructura | Una unidad de funcionalidad visible externamente, proporcionada por uno o más nodos, expuesta a través de interfaces bien definidas y significativa para el entorno. |   |
| Artefacto | Una pieza física de datos que se utiliza o se produce en un proceso de desarrollo de software, o mediante el despliegue y la operación de un sistema. |   |

Cuadro 4.5: Conceptos de Capa de Tecnología

4.1.6. Resumen Relaciones


4.1.6.1. Relaciones Estructurales


| Concepto | Definición | Notación |
|------------|--|---|
| Asociacion | La asociación modela una relación entre objetos que no están cubiertos por otra más relación específica. |  |

| Concepto | Definición | Notación |
|-------------|---|---|
| Acceso | La relación de acceso modela el acceso de Conceptos conductuales para negocios o datos objetos. |  |
| Usado por | El utilizado por la relación modela el uso de servicios por procesos, funciones o interacciones y el acceso a las interfaces por roles, componentes o colaboraciones. |  |
| Realización | La relación de realización vincula una lógica entidad con una entidad más concreta que se da cuenta. |  |
| Asignación | La relación de asignación vincula unidades de comportamiento con elementos activos (por ejemplo, roles, componentes) que los realizan, o roles con actores que los cumplen. |  |
| Agregación | La relación de agregación indica que un el objeto agrupa varios otros objetos. |  |
| Composición | La relación de composición indica que un objeto se compone de uno o más objetos. |  |

Cuadro 4.6: Conceptos de Capa de Relaciones Estructurales

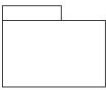


4.1.6.2. Relaciones Dinámicas

| Concepto | Definición | Notación |
|----------|---|---|
| Flujo | La relación de flujo describe el intercambio o transferencia de, por ejemplo, información o valor entre procesos, función, interacciones y eventos. |  |

| Concepto | Definición | Notación |
|----------------|--|---|
| Desencadenante | La relación de activación describe el relaciones temporales o causales entre procesos, funciones, interacciones y eventos. |  |

Cuadro 4.7: Conceptos de Capa de Relaciones Dinámicas

4.1.6.3. Otras Relaciones

| Concepto | Definición | Notación |
|-----------------|---|---|
| Agrupamiento | La relación de agrupamiento indica que objetos del mismo tipo o tipos diferentes pertenecen juntos basados en algunos característica. |  |
| Union | Un cruce se usa para conectar relaciones de el mismo tipo |  |
| Especialización | La relación de especialización indica que un objeto es una especialización de otro objeto. |  |

Cuadro 4.8: Conceptos de Capa de Relaciones Dinámicas

Capítulo 5

Negocio

5.1. Introducción

En esta capa se busca dar información acerca de una serie de conceptos informativos para dar relevancia en el dominio empresarial: un producto y contrato asociado, el significado de los objetos de negocio y el valor de los productos y servicios empresariales.

Para ello es necesario el diseño de 6 puntos de vista dentro de los cuales se encuentran: de organización, de cooperación de actor, de función de negocio, de proceso de negocio, de cooperación de proceso de negocio y de producto.

Los puntos de vista se mostrarán en dos partes, la primera de ellas será el modelo donde se encontrará una descripción del punto de vista acompañado de una tabla con la información de este y el meta-modelo correspondiente, en la segunda parte se encontrará el caso de estudio, es decir, el modelo donde será aplicado el meta-modelo previamente visto al proyecto y la organización en la que se está trabajando.

5.2. Punto de Vista de Organización

5.2.1. Descripción

El punto de vista organizacional en la organización de la compañía, departamento, red de compañías, u otra identidad organizacional. Esto es posible para presentar modelos en este punto de vista como diagramas de bloques anidados, pero en el camino mas tradicional, tal como mapas organizacionales. El punto de vista organizacional es muy tratado en la identificación de competencias, autoridades y responsables en la organización.

5.2.1.1. Metamodelo

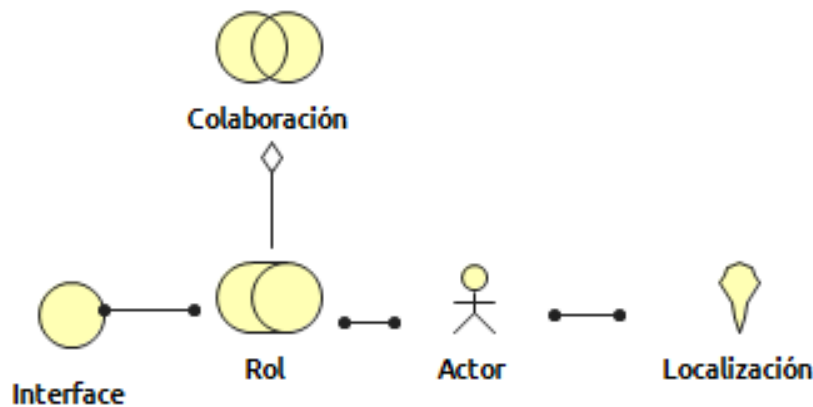


Figura 5.1: Metamodelo: Punto de Vista de Organización

5.2.1.2. Caso de Estudio

En el siguiente punto de vista podemos resaltar la sucursal como localización principal de la organización; adicionalmente, se puede observar la participación de actores quienes son los pilares fundamentales para la realización de los principales procesos en la compañía. Estos son: Seguridad y Administración.

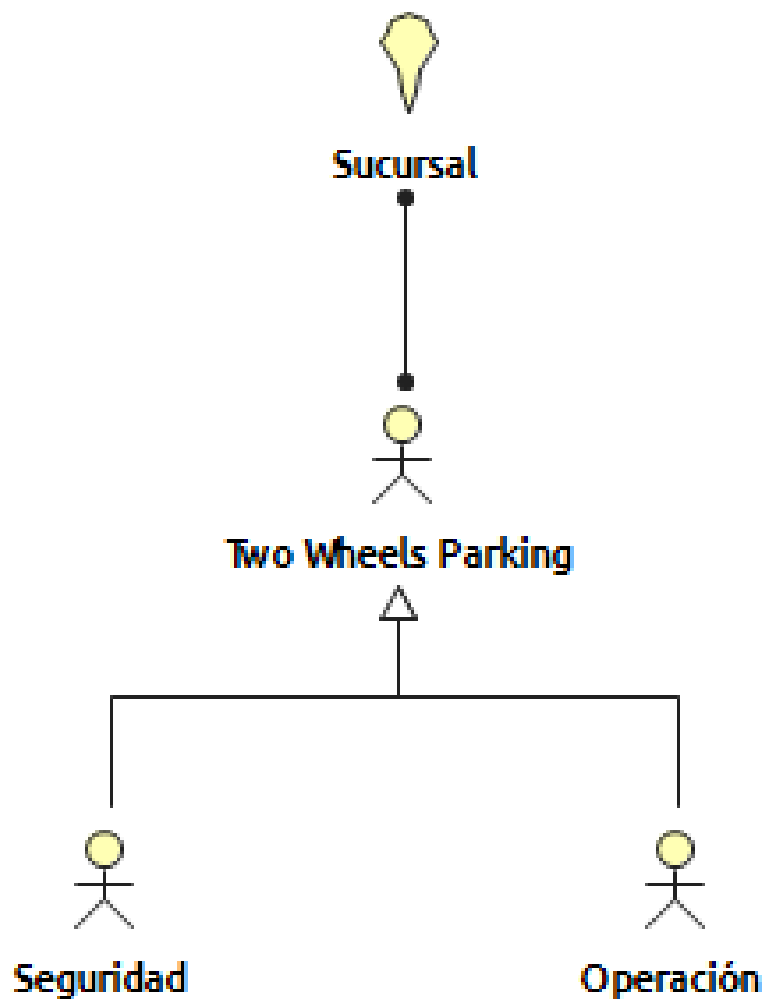


Figura 5.2: Caso de estudio: Punto de Vista de Organización

5.3. Punto de Vista de Cooperación de Actor

5.3.1. Descripción

El punto de vista de Cooperación de actor se enfoca en las relaciones que se presentan entre un actor y su entorno, se puede decir que es un diagrama de contexto en el cual se coloca la organización en su entorno, además de esto

se puede evidenciar clientes, proveedores y compañeros de negocio, tiene como objetivo determinar dependencias y colaboraciones externas y ver la relación con los actores que se encuentran en este diagrama, tenemos por otra parte que en este punto de vista se puede evidenciar el número de actores cooperantes de negocio van a tener participación en esta capa o cuantos componentes de aplicación interactúan para formar un proceso de negocio.

5.3.1.1. Metamodelo

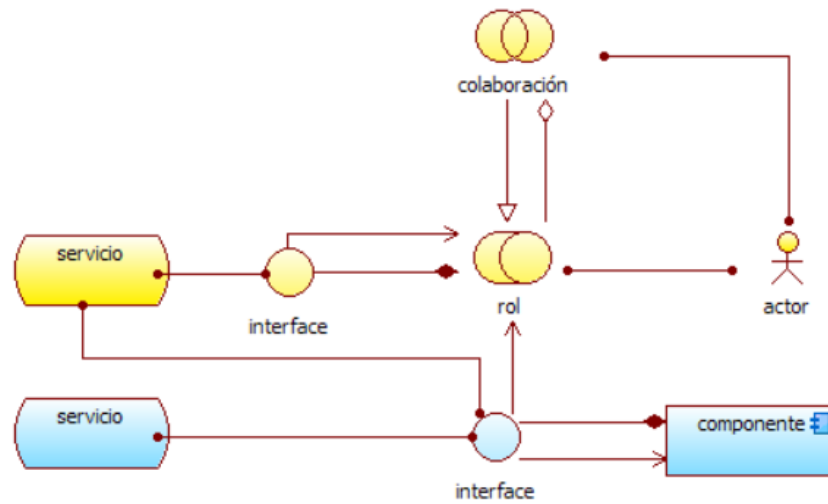


Figura 5.3: Metamodelo: Punto de Vista de Organización

5.3.1.2. Caso de Estudio

La colaboración Espacio de Parquadero, surge de la interacción de los roles Sucursal y Clientes del Parquadero. Por parte de la sucursal, brindando el terrero de parqueo; y por parte de los clientes, al realizar el uso del mismo a través de la interface que se encuentra de cara a ellos, la Recepción.

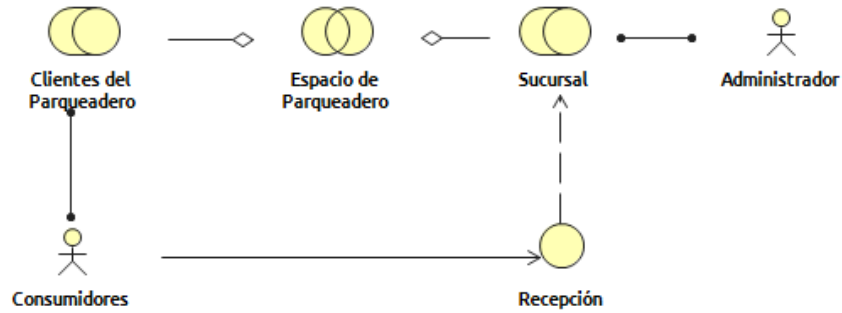


Figura 5.4: Caso de estudio: Punto de Vista de Cooperación de Actor

5.4. Punto de Vista de Función de Negocio

5.4.1. Descripción

El punto de vista Función de negocio muestra las principales funciones de negocio de una organización y sus relaciones en términos de los flujos de información, valor o bienes entre ellos. Las funciones empresariales se utilizan para representar los aspectos más estables de una empresa en términos de las actividades primarias que realiza, independientemente de los cambios organizacionales o desarrollos tecnológicos.

Por lo tanto, la arquitectura de la función comercial de las empresas que operan en el mismo mercado a menudo muestran similitudes cercanas. Por lo tanto, el punto de vista de la función empresarial proporciona una visión de alto nivel de las operaciones generales de la empresa y puede utilizarse para identificar las competencias necesarias o estructurar una organización de acuerdo con sus principales actividades.

5.4.1.1. Metamodelo

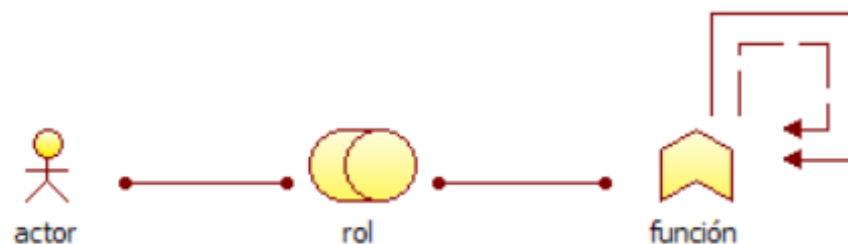


Figura 5.5: Metamodelo: Punto de Vista de Función de Negocio

5.4.1.2. Caso de Estudio

En el siguiente punto de vista se resalta la presencia de los actores Operario y Jefe de Seguridad, quienes llevan a cabo funciones específicas de los roles de Operación y Seguridad, tales como: Registro de Ingreso, Asignación de Parqueadero, Registro de Usuarios, Registro de Pagos, Registro de Retiros y Verificación de propiedad del vehículo.

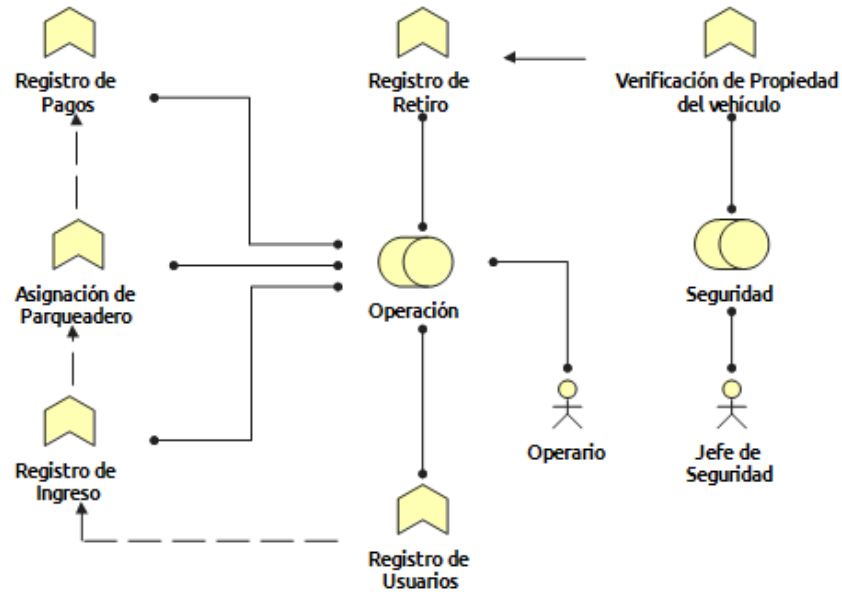


Figura 5.6: Caso de estudio: Punto de vista de función de negocio.

5.5. Punto de Vista de Proceso de Negocio

5.5.1. Descripción

El punto de vista de proceso de negocio es usado para ver la estructura desde un nivel alto además de esto de poder evidenciar la composición de uno o más procesos de negocio, dentro de este punto de vista podemos resaltar que se enfoca en los servicios que un proceso de negocio puede ofrecer al cliente mostrando como este proceso puede contribuir para la realización de productos, por otro lado se puede hacer una asociación en cuanto a responsabilidades que tienen los actores asociados y los roles dentro de este proceso de negocio y por último en este punto de vista podemos evidenciar la información utilizada por el proceso de negocio.

5.5.1.1. Metamodelo

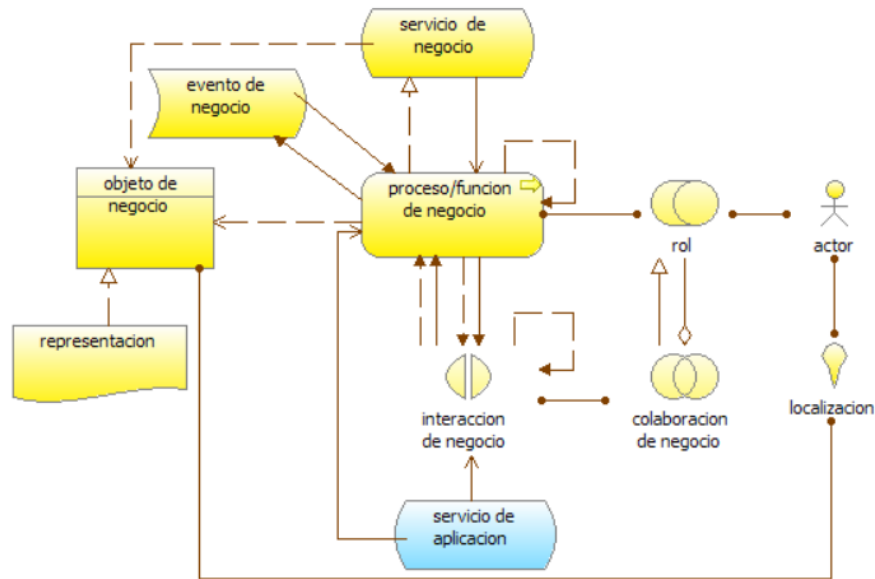


Figura 5.7: Metamodelo: Punto de Vista de Proceso de Negocio

5.5.1.2. Caso de Estudio

En este punto de vista podemos resaltar el proceso principal de la organización: Renta de espacio de parqueo, el cual es originado por una solicitud del espacio y finaliza en el retiro del vehículo. Dentro de este proceso tenemos los subprocesos de: Registro de Usuario, Registro de Ingreso, Asignación de Espacio, Vigilancia de Vehículo, Cálculo de tarifa y Registro de Pago.

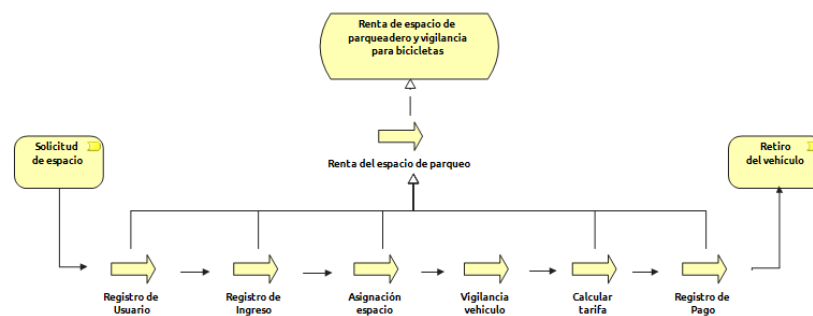


Figura 5.8: Caso de estudio: Punto de vista de proceso de negocio.

5.6. Punto de vista de Cooperación de procesos de negocio.

5.6.1. Descripción

El punto de vista de Cooperación de Proceso de Negocio se utiliza para mostrar las relaciones de uno o más procesos de negocio entre sí y / o con su entorno. Puede utilizarse tanto para crear un diseño de alto nivel de procesos empresariales dentro de su contexto como para proporcionar un gestor operativo responsable de uno o más de dichos procesos con información sobre sus dependencias

5.6.1.1. Metamodelo

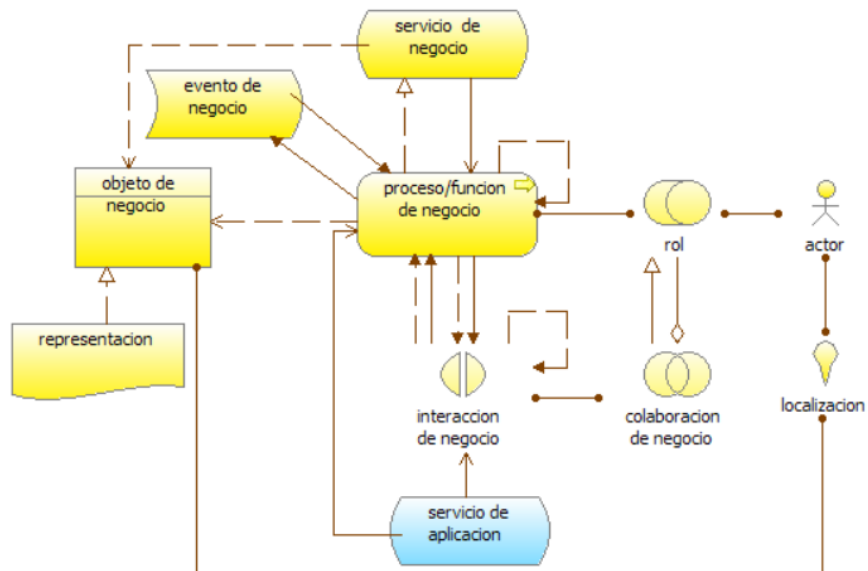


Figura 5.9: Metamodelo: Punto de Vista de Cooperación de Negocio

5.6.1.2. Caso de Estudio

Adicional al punto de vista anterior, se evidencian los roles involucrados, Administrador y Agente de Seguridad, en el proceso Renta de espacio de parqueo, para llevar a cabo el cumplimiento del servicio fundamental de la organización, Renta de espacio de parqueadero y vigilancia para bicicletas.

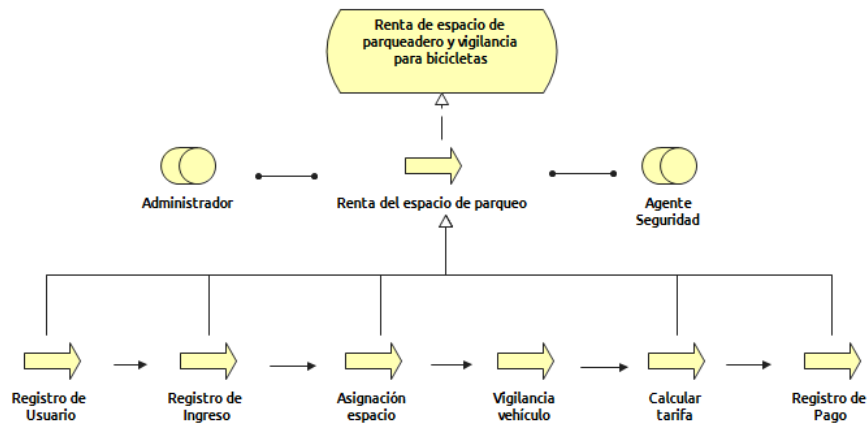


Figura 5.10: Caso de estudio: Punto de vista de colaboración de procesos de negocio.

5.7. Punto de Vista de Producto

5.7.1. Descripción

El punto de vista del producto representa el valor que ese producto ofrece a los clientes u otros involucrados y muestra la composición de uno o más productos en términos de los servicios constituidos y la asociación de contratos u otros acuerdos. Esto también puede ser usado para mostrar los canales de interfaces que este producto ofrece, y los eventos asociados con el producto. Un punto de vista del producto es típicamente usado en el desarrollo del producto a el diseño del producto por la composición de servicios existentes o la identificación que nuevos servicios tiene que ser creados para este producto, dando el valor a las expectativas del cliente para este. Este puede servir para la entrada para la arquitectura del proceso de negocio y otros que necesiten diseñar el proceso y ICT realizan estos productos.

5.7.1.1. Metamodelo

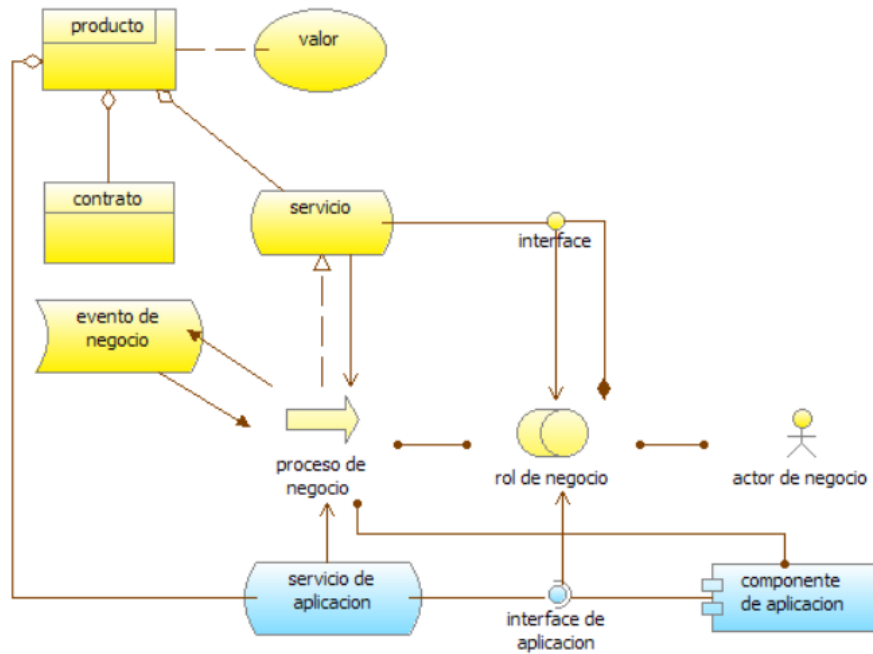


Figura 5.11: Metamodelo: Punto de Vista de Prodcuto

5.7.1.2. Caso de Estudio

El producto principal que ofrece la organización, como se evidencia en los puntos de vista anteriores, es el espacio de parqueo, el cual se caracteriza por proponer espacios adecuados y seguros para las bicicletas de los clientes, a la vez de ofrecer un servicio integral de vigilancia permanente para los vehículos, soportados por sus correspondientes procesos.

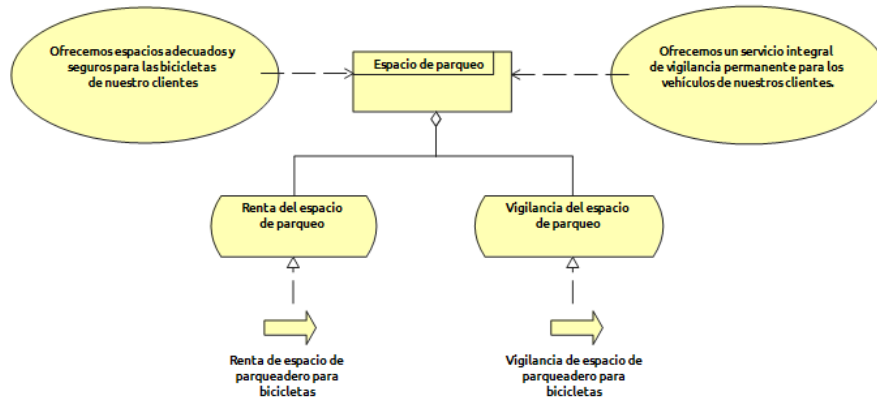


Figura 5.12: Caso de estudio: Punto de vista de producto

Capítulo 6

Aplicación

Capítulo 7

Capa de Aplicación

7.1. Introducción

En esta capa de aplicaciones observaremos los principales conceptos de comunicación entre componentes por medio de interfaces, donde en los siguientes diagramas se pueden observar los principales comportamientos del aplicativo mediante 4 puntos de vista: Comportamiento de aplicación, Cooperación de aplicación, Estructura de aplicación y el uso de la aplicación.

Tal como mencionamos anteriormente, en nuestra capa de aplicación se caracteriza por poseer una arquitectura de componentes, de tal forma que mostramos las principales funciones dentro de cada componente y además como se relacionan y comunican cada uno de estos, de tal forma que observemos como será el comportamiento y la lógica de la aplicación

7.2. Punto de Vista de Comportamiento de Aplicación

7.2.1. Descripción

El punto de vista de comportamiento de aplicación, describe el comportamiento interno de la aplicación, que este realiza uno o más servicios de aplicaciones. El punto de vista es útil en diseñar el principal comportamiento de la aplicación, o identificar superposición funcional entre diferentes aplicación.

7.2.1.1. Metamodelo

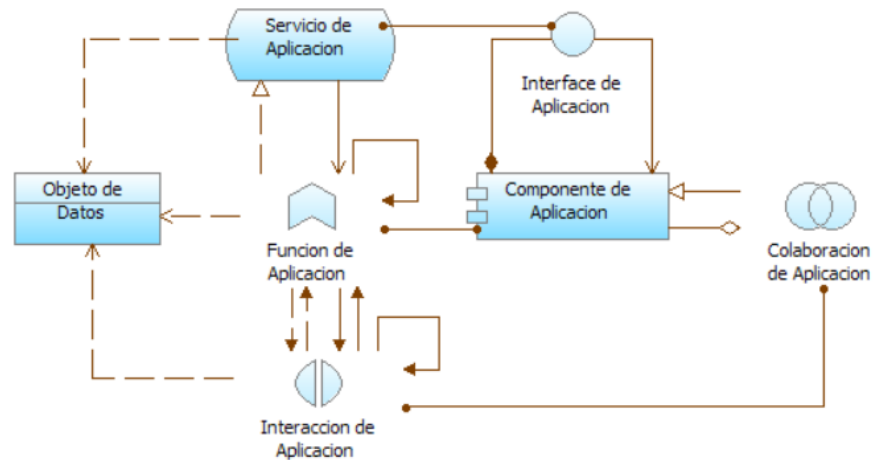


Figura 7.1: Metamodelo: Punto de Vista de Comportamiento de Aplicación.

7.2.1.2. Caso de Estudio

En el presente caso de estudio, podemos observar el componente aplicación de la aplicación Two Wheels Digital, el cual se divide compone a su vez de componentes más pequeños como lo son Pagos, que se encarga de cumplir las funciones de cálculo de la tarifa y recepción del pago realizado por cada cliente. Por otra parte, el Gestor de Espacios encargado de la administración de los espacios de parqueo; el Gestor de Usuarios quien lleva a cabo el proceso de registro y validación. Por último, el componente de Estadísticas el cual elabora reportes sobre la renta de los espacios de parqueo.

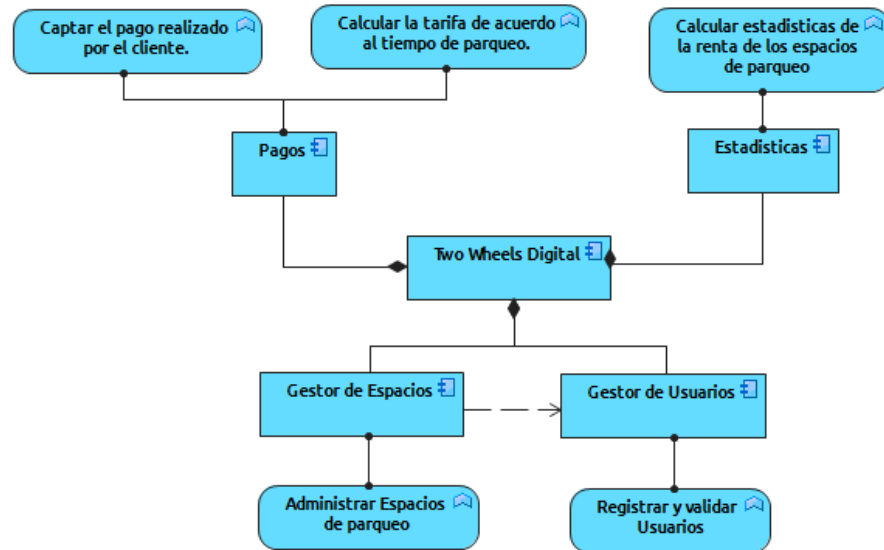


Figura 7.2: Caso de estudio: Punto de vista de comportamiento de aplicación.

7.3. Punto de Vista de Cooperación de Aplicación

7.3.1. Descripción

El punto de vista Cooperación de aplicación describe las relaciones entre los componentes de la aplicaciones en términos de la información que se maneja entre ellos además de esto también se puede enfocar en términos de los servicios que estos ofrecen y de los que hacen uso. Este punto de vista también es usado para crear una visión general de todo el contexto de aplicación de una organización. Este punto de vista también es usado para expresar la cooperación interna de servicios que trabajando de una forma conjunta soportan la ejecución de un proceso de negocio.

7.3.1.1. Metamodelo

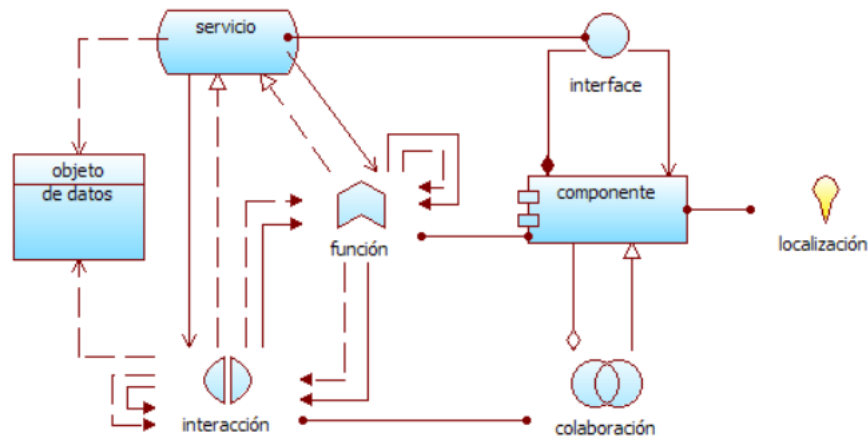


Figura 7.3: Metamodelo: Punto de Vista de Cooperación de Aplicación.

7.3.1.2. Caso de Estudio

Por medio de este punto de vista, se resalta la arquitectura general de tres capas de la aplicación Two Wheels Digital en donde el componente principal de interfaz se encuentra ubicado en la capa de presentación; los componentes Gestor de Usuarios, Gestor de Espacios, Pagos y Estadísticas se ubican en la capa de negocio, y finalmente el componente de almacenamiento, Base de Datos, se ubica en la capa de persistencia.

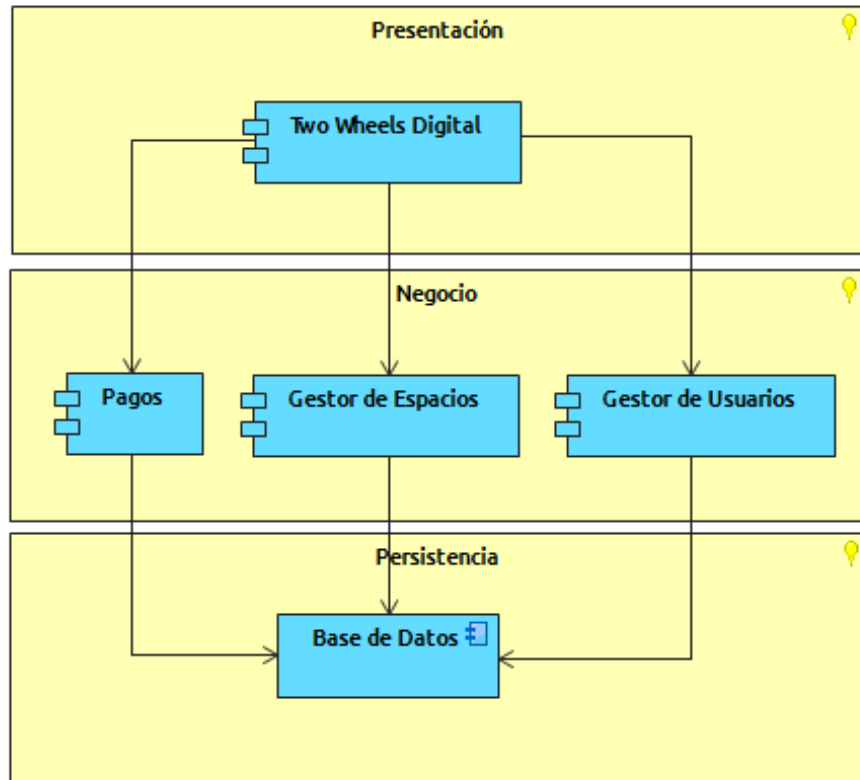


Figura 7.4: Caso de estudio: Punto de vista de cooperación de aplicación.

7.4. Punto de Vista de Uso de Aplicación

7.4.1. Descripción

El punto de vista Uso de aplicaciones describe cómo se utilizan las aplicaciones para soportar uno o más procesos empresariales y cómo se utilizan en otras aplicaciones. Se puede utilizar en el diseño de una aplicación mediante la identificación de los servicios necesarios por los procesos de negocio y otras aplicaciones, o en el diseño de procesos de negocio mediante la descripción de los servicios que están disponibles. Además, puesto que identifica las dependencias de los procesos de negocio en las aplicaciones, puede ser útil para los gerentes operacionales responsables de estos procesos.

7.4.1.1. Metamodelo

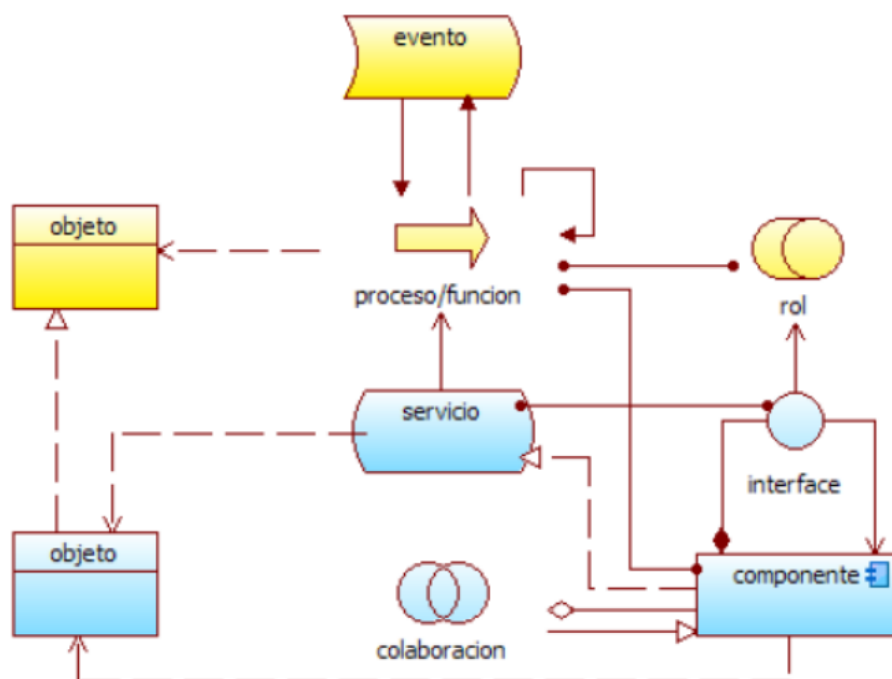


Figura 7.5: Metamodelo: Punto de Vista de Uso de Aplicación.

7.4.1.2. Caso de Estudio

Se tiene como enfoque principal el proceso de Renta de espacios de parqueadero el cual hace uso de tres servicios llevados a cabo por el componente principal de la aplicación Two Wheels Digital, como lo son la gestión de usuarios, la gestión de espacios y la gestión de pagos.

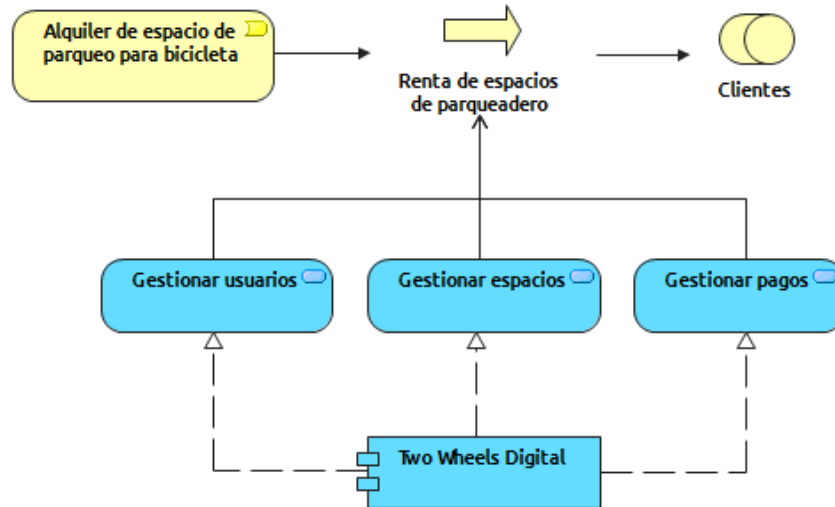


Figura 7.6: Caso de estudio: Punto de vista de Uso de aplicación.

7.5. Punto de vista de Estructura de Aplicación

7.5.1. Descripción

El punto de vista de Estructura de la aplicación muestra la estructura de una o más aplicaciones o componentes. Este punto de vista es útil para diseñar o comprender la estructura principal de aplicaciones o componentes y los datos asociados, por ejemplo, para descomponer la estructura del sistema en construcción o para identificar componentes de aplicación heredados que son adecuados para la migración/integración.

7.5.1.1. Metamodelo

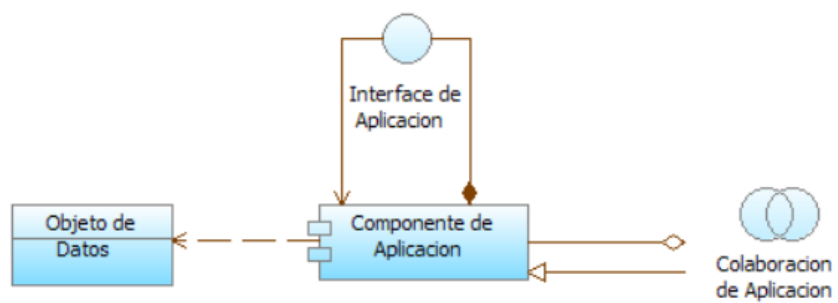


Figura 7.7: Metamodelo: Punto de Vista de Estructura de Aplicación.

7.5.1.2. Caso de Estudio

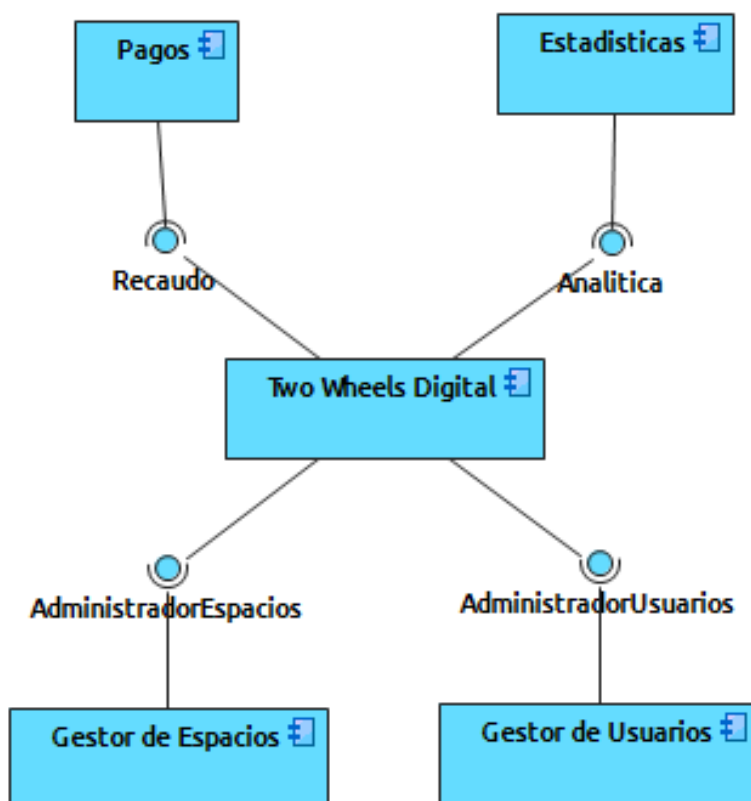


Figura 7.8: Caso de estudio: Punto de vista de estructura de aplicación.

Capítulo 8

Tecnología

Capítulo 9

Capa de Tecnologia

9.1. Introducción

En esta capa de aplicaciones observaremos los principales conceptos de comunicación entre componentes por medio de interfaces, donde en los siguientes diagramas se pueden observar los principales comportamientos del aplicativo mediante 4 puntos de vista: Comportamiento de aplicación, Cooperación de aplicación, Estructura de aplicación y el uso de la aplicación.

Tal como mencionamos anteriormente, en nuestra capa de aplicación se caracteriza por poseer una arquitectura de componentes, de tal forma que mostramos las principales funciones dentro de cada componente y además como se relacionan y comunican cada uno de estos, de tal forma que observemos como será el comportamiento y la lógica de la aplicación

9.2. Punto de Vista de Infraestructura

9.2.1. Descripción

El punto de vista de infraestructura contiene los elementos de hardware y de software correspondientes a la infraestructura que soporta la capa de aplicación, en este diagrama podemos observar elementos tales como dispositivos físicos, redes o sistemas de software tales como: Bases de datos o sistemas operativos.

9.2.1.1. Metamodelo

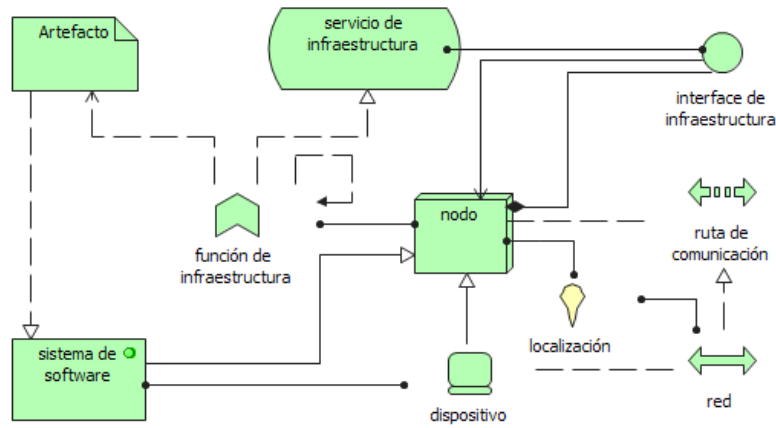


Figura 9.1: Metamodelo: Punto de Vista de Infraestructura.

9.2.1.2. Caso de Estudio

El sistema que se pretende desarrollar estará distribuido de manera local, es decir, se dispondrá de un único dispositivo (computador) que desplegará la aplicación haciendo uso del sistema operativo, un gestor de bases de datos y un dispositivo de lectura externo.

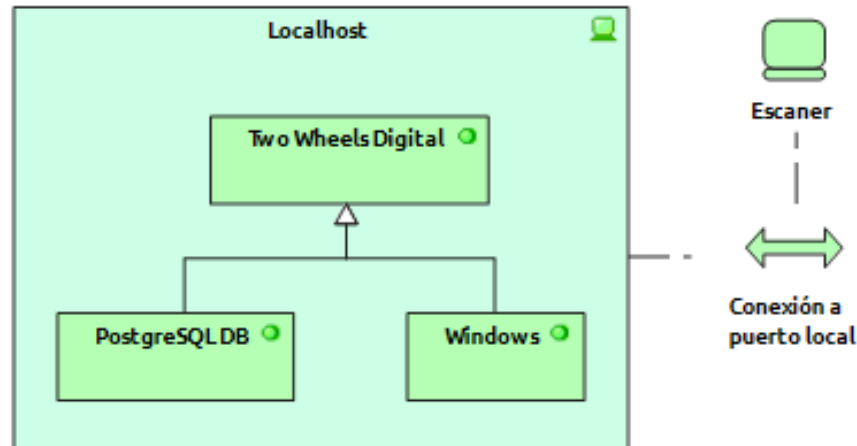


Figura 9.2: Caso de estudio: Punto de vista de Infraestructura.

9.3. Punto de Vista de Uso de Infraestructura

9.3.1. Descripción

El punto de vista de Uso de Infraestructura muestra como las aplicaciones están soportadas por una infraestructura de hardware y software en la que los servicios de infraestructura son ofrecidos por los dispositivos mientras que las redes y los sistemas de software proporcionan las aplicaciones. Este punto de vista juega un papel importante en el análisis del rendimiento y la escalabilidad, ya que relaciona la infraestructura física con el mundo lógico de las aplicaciones. Es muy útil para determinar el rendimiento y los requisitos de calidad en la infraestructura en función de las demandas de las distintas aplicaciones que lo utilizan.

9.3.1.1. Metamodelo

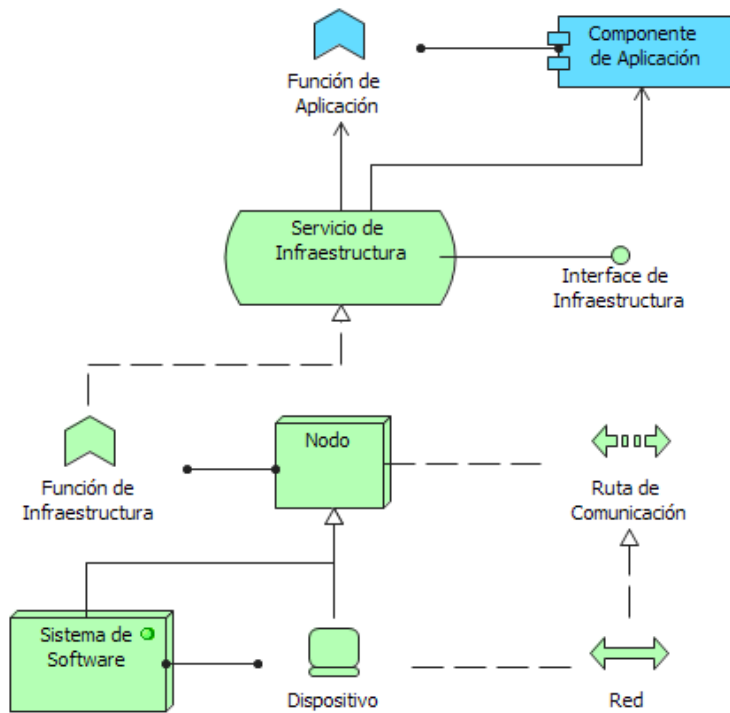


Figura 9.3: Metamodelo: Punto de Vista de Uso de Infraestructura.

9.3.1.2. Caso de Estudio

En el siguiente punto de vista podemos observar el comportamiento general que tendrá el sistema teniendo en cuenta los componentes de hardware que para este caso es un único dispositivo y los componentes de software que han sido descritos anteriormente, estos componentes de software serán los encargados de gestionar los procesos internos y lógicos del sistema, gestión de los pagos, gestión de los espacios de parqueo, generación de estadísticas y la gestión de los usuarios.

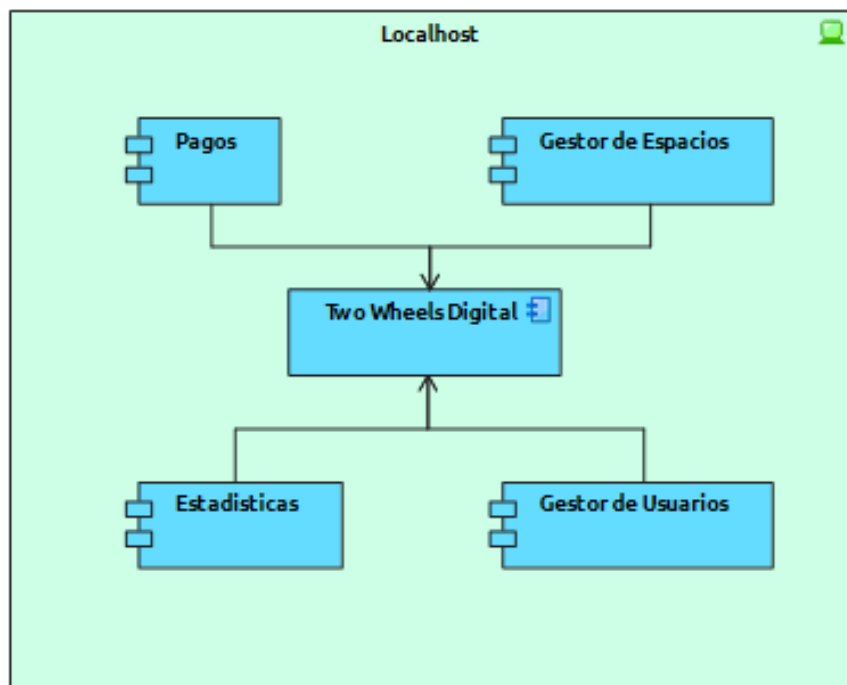


Figura 9.4: Caso de estudio: Punto de Vista de Uso de Infraestructura.

9.4. Punto de Vista de Organización e implementación

9.4.1. Descripción

El punto de vista de organización e implementación muestra como se implementan una o mas aplicaciones o componentes en la infraestructura. Esto se comprende como el mapeo de aplicaciones lógicas a su respectivos componente o artefacto físico, de igual manera se puede evidenciar un mapeo de la información que utilizan estas.

9.4.1.1. Metamodelo

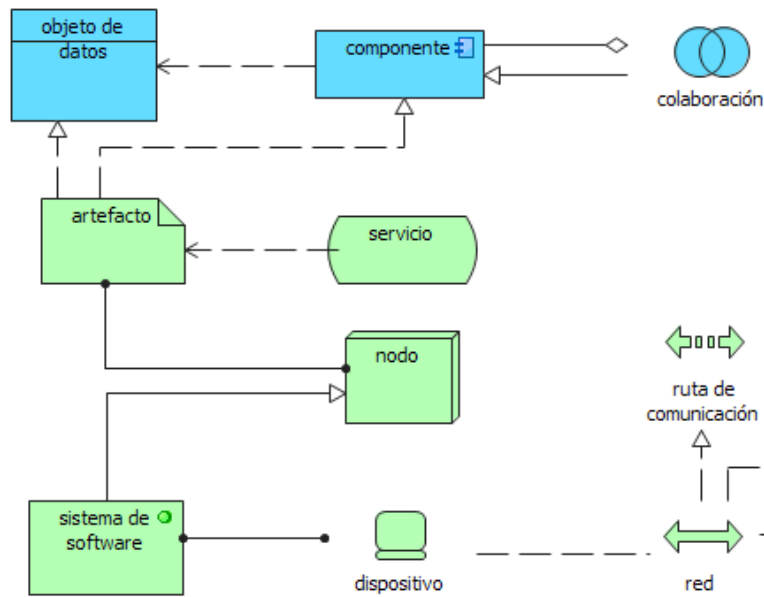


Figura 9.5: Metamodelo: Punto de Vista de Organización e Implementación.

9.4.1.2. Caso de Estudio

El sistema a desarrollar estará compuesto principalmente por cuatro componentes de software que soportarán todos los procesos del sistema. Como único componente físico se tendrá un dispositivo que desplegará la aplicación a partir de componente Two Wheels Digital el cual contendrá los componentes de software anteriormente mencionados.

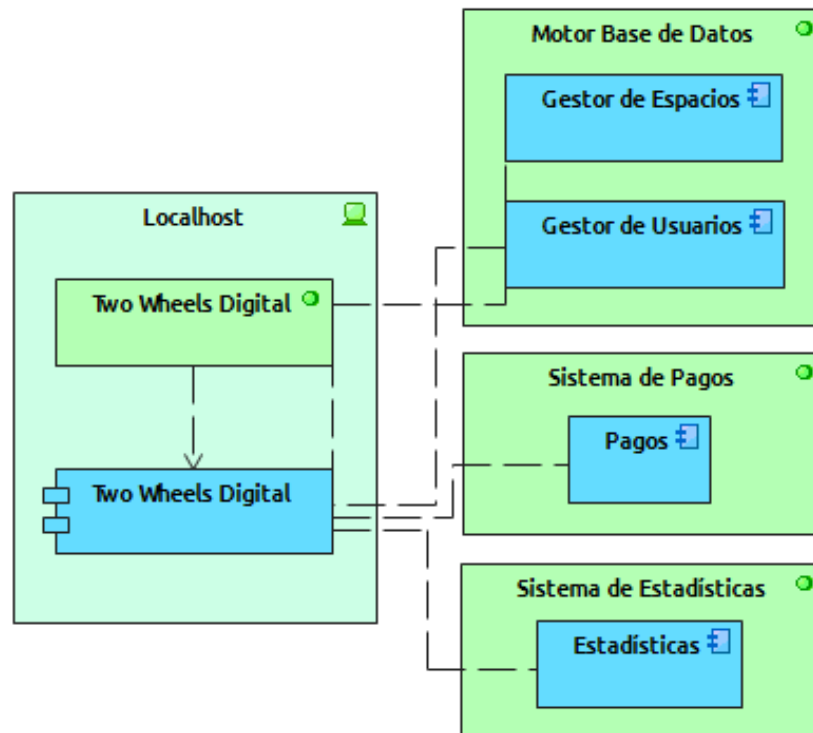


Figura 9.6: Caso de estudio: Punto de Vista de Organización e Implementación.

9.5. Punto de Vista de Estructura de Información

9.5.1. Descripción

El punto de vista de la estructura de información muestra la estructura de la información utilizada en la: empresa, proceso o una aplicación comercial específica; esto en términos de tipos de datos o estructuras de clases (orientadas a objetos). Además, puede mostrar cómo la información se representa en el nivel de negocio, en el nivel de aplicación mediante las estructuras de datos utilizadas allí, y cómo estas se asignan a la infraestructura subyacente mediante artefactos.

9.5.1.1. Metamodelo

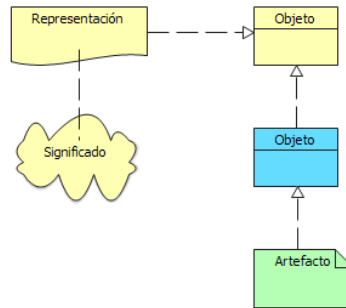


Figura 9.7: Metamodelo: Punto de Vista de Estructura de Información.

9.5.1.2. Caso de Estudio

Teniendo en cuenta que la función principal de la empresa es la renta de espacios de parqueo, se tienen como objetos principales el Establecimiento, Espacio, Usuario y Cliente, estos objetos soportarán todos los procedimientos requeridos por la organización y serán desplegados por la aplicación mediante el objeto Two Wheels Digital, en conjunto con el objeto Espacios Disponibles que permitirán realizar el debido control del servicio ofrecido.

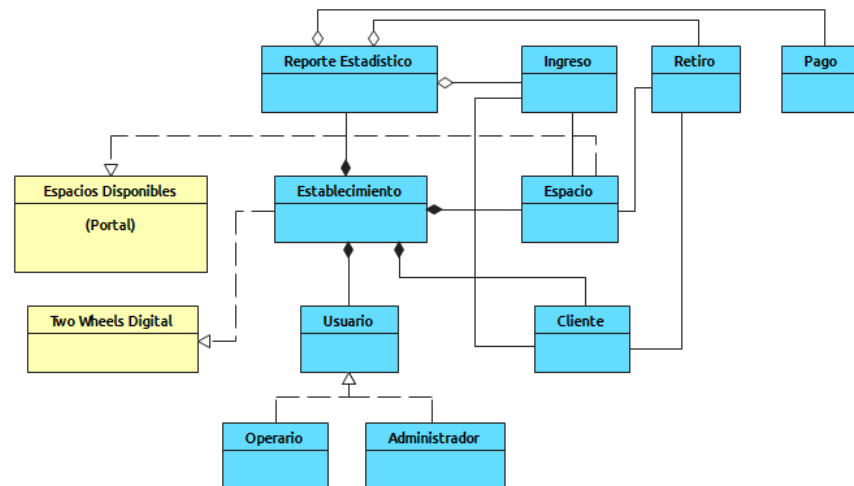


Figura 9.8: Caso de estudio: Punto de Vista de Estructura de Información.

9.6. Punto de Vista de Realización del Servicio

9.6.1. Descripción

El punto de vista de Realización del servicio se utiliza para mostrar cómo uno o más servicios de negocio se realizan mediante los procesos subyacentes (y, en ocasiones, mediante los componentes de la aplicación). Por lo tanto, forma el puente entre el punto de vista del producto y la vista del proceso de negocios. Proporciona una "vista desde el exterior.^{en} uno o más procesos comerciales.

9.6.1.1. Metamodelo

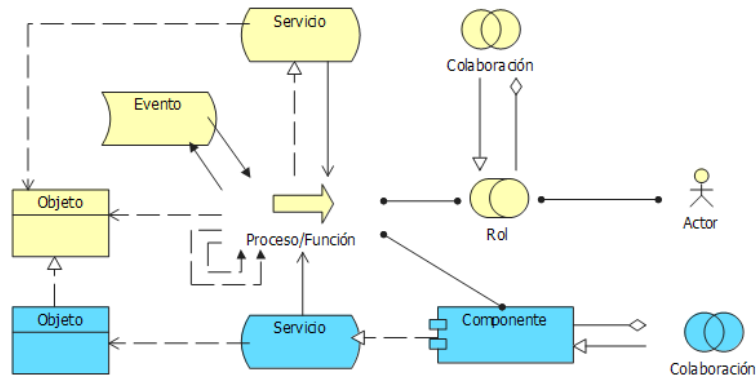


Figura 9.9: Metamodelo: Punto de Vista de Realización del Servicio.

9.6.1.2. Caso de Estudio

El Cliente como el rol que solicita el servicio de renta del espacio será el que disparará las funciones establecidas dentro del protocolo de la organización. Estas funciones estarán asociadas al rol Operario, quien tendrá la capacidad de ejecutar dichas funciones a través de los componentes respectivos para cada función.

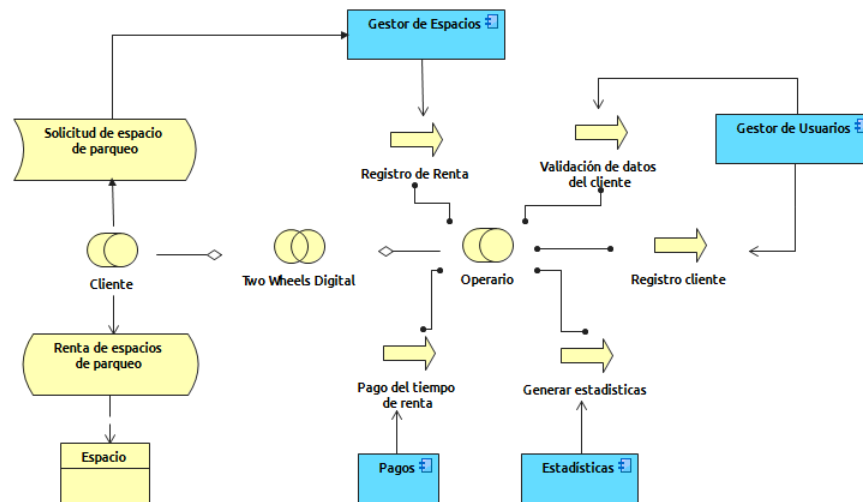


Figura 9.10: Caso de estudio: Punto de Vista de Realización del Servicio.

9.7. Punto de Vista de Capas

9.7.1. Descripción

El punto de vista en capas representa varias capas y aspectos de una arquitectura empresarial en un diagrama. El principio estructural detrás de un punto de vista completamente estratificado es que cada capa dedicada expone, mediante la relación de realización, una capa de servicios, que luego son utilizados por la siguiente capa dedicada. Por lo tanto, podemos separar fácilmente la estructura interna y la organización de una capa dedicada de su comportamiento externo observable expresado como la capa de servicio que la capa dedicada realiza. El objetivo principal del punto de vista de capas es proporcionar información general en un diagrama.

9.7.1.1. Caso de Estudio

El sistema a desarrollar estará distribuido en tres capas, las cuales contendrán los componentes del sistema de acuerdo a su ámbito con el fin de visualizar el comportamiento de éste de una manera más efectiva. Dentro de la capa de Infraestructura se encuentra el dispositivo donde se desplegará el sistema, este dispositivo hará uso de los componentes Gestor de Usuarios, Gestor de Espacios, Estadísticas y Pagos, los cuales se encuentran en la capa de aplicación debido a su significado lógico, es decir, son componentes de software, en la capa de Negocio se encuentran las funciones principales del sistema, la Solicitud del Espacio y la devolución de este, finalmente se optó por mostrar el servicio general en otra

capa de Negocio para permitir una lectura y comprensión mucho mas sencilla del punto de vista.

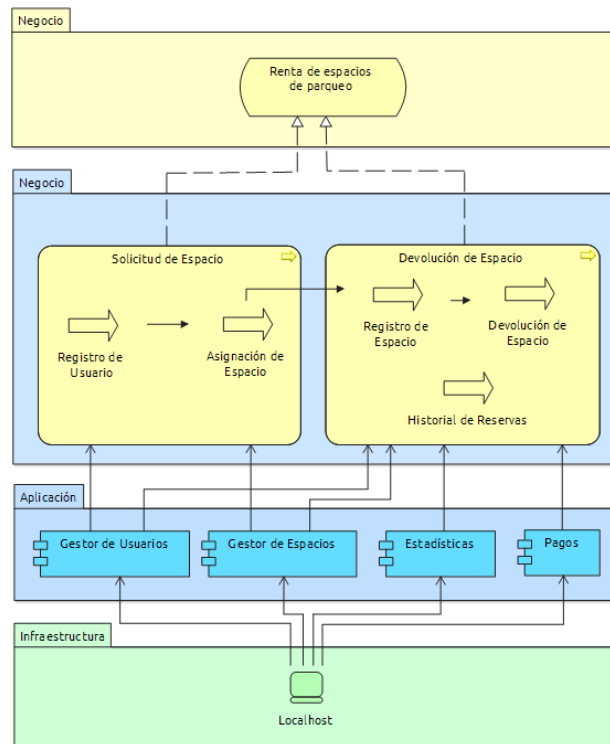


Figura 9.11: Caso de estudio: Punto de Vista de Capas.

Capítulo 10

Motivación

10.1. Introducción

En las capas anteriores hemos visto como la estructura tecnológica le da soporte a la arquitectura empresarial, pero en esta capa vamos a visualizar y a modelar las razones o motivaciones que subyacen al diseño de esta, ya que son estas motivaciones las que restringen y van guiando el diseño de esta. Las motivaciones reales están representadas por metas, principios, requisitos y restricciones. Las metas buscan representar algún resultado deseado - o fin - que una parte interesada desea lograr; por ejemplo, aumentar la satisfacción del cliente. Los principios y requisitos representan las propiedades deseadas de soluciones o medios mediante los cuales se pueden alcanzar los objetivos. Los principios son una serie de normativas o pautas que guían el diseño de todas las soluciones posibles en un contexto dado. Y finalmente los requisitos representan declaraciones formales de necesidad, expresada por las partes interesadas, que debe cumplir la arquitectura o las soluciones siendo estas últimas inamovibles.

10.2. Punto de Vista de StakeHolder

10.2.1. Descripción

El punto de vista de las partes interesadas permite identificar y modelar los impulsores internos y externos para el cambio y las evaluaciones (en términos de fortalezas, debilidades, oportunidades y (amenazas) de estos controladores. Además, los enlaces al objetivo inicial (alto nivel) que abordan estas preocupaciones y evaluaciones pueden ser descritas. Estos objetivos forman la base del proceso de ingeniería de requerimientos, que incluye el refinamiento de objetivos, el análisis de contribución y conflicto, y la derivación de requisitos que cumplan los objetivos.

10.2.1.1. Metamodelo

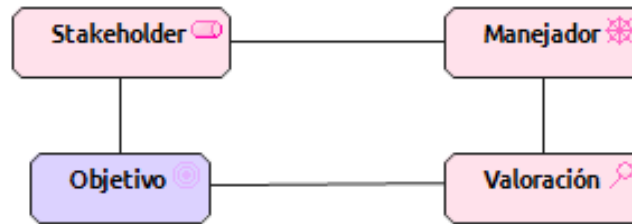


Figura 10.1: Metamodelo: Punto de Vista de StakeHolder

10.2.1.2. Caso de Estudio

El diagrama de Stakeholder o “partes interesadas” aplicado al caso de estudio, genera el modelo que se presenta a continuación, en el visualizamos nuestro objetivo de alto nivel correspondiente a “Mantener el vehículo Seguro” este objetivo presenta las partes interesadas representadas en el cliente y el representante de la empresa que en este caso es el guardia de turno, todo esto buscando que se fortalezca la percepción de seguridad en el cliente.

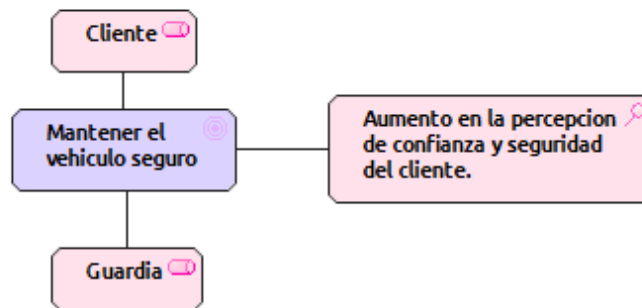


Figura 10.2: Caso de estudio: Punto de Vista de StakeHolder.

10.3. Punto de Vista de Realización de Objetivos

10.3.1. Descripción

El punto de vista de realización de objetivos permite a un diseñador modelar el refinamiento de objetivos (de alto nivel) en objetivos más tangibles, y el refinamiento de objetivos tangibles en requisitos o restricciones que describen las propiedades que se necesitan para alcanzar los objetivos. El refinamiento de las metas en sub objetivos se modela utilizando la relación de agregación. El refinamiento de las metas en requisitos se modela utilizando la relación de realización.

10.3.1.1. Metamodelo

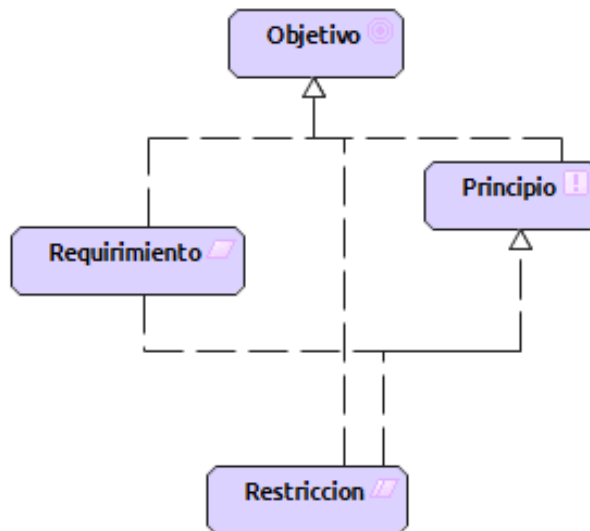


Figura 10.3: Metamodelo: Punto de Vista de Realización de Objetivos

10.3.1.2. Caso de Estudio

Para el modelamiento del diagrama de realización de objetivos tenemos al igual que en el diagrama de Stakeholder el objetivo de alto nivel “Mantener seguro el vehículo”, adicional a esto encontramos una serie de restricciones que se deben cumplir para que el objetivo sea ejecutado exitosamente, ejemplo de ello es el requerimiento: “El vehículo solo se puede retirar si se ha efectuado el pago del servicio” esta restricción nos lleva a plantear el requerimiento de que la transacción del pago sea exitosa.

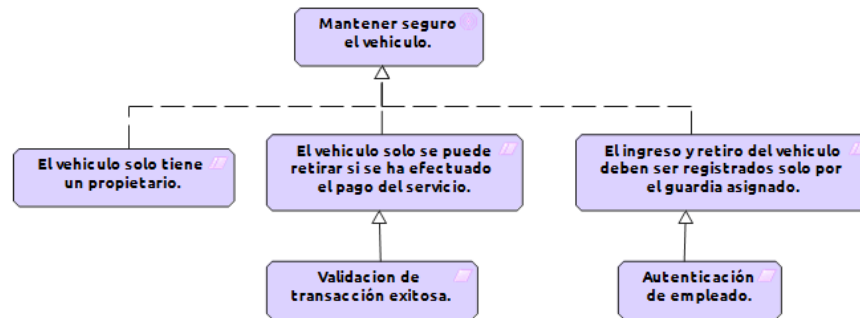


Figura 10.4: Caso de estudio: Punto de Vista de Realización de Objetivos.

10.4. Punto de Vista de Contribución

10.4.1. Descripción

El punto de vista de contribución permite modelar las relaciones de influencia entre objetivos y los requisitos necesarios para llevarlos a buen puerto. Este punto de vista se puede usar para analizar el impacto que los objetivos tienen entre sí o para detectar conflictos entre los objetivos de los Stakeholders. Este punto de vista requiere un refinamiento de los objetivos en lo posible un desglose lo mas detallado posible en sub objetivos. Al igual que en el punto de vista anterior tenemos los bloque y las relaciones de agregación y realización.

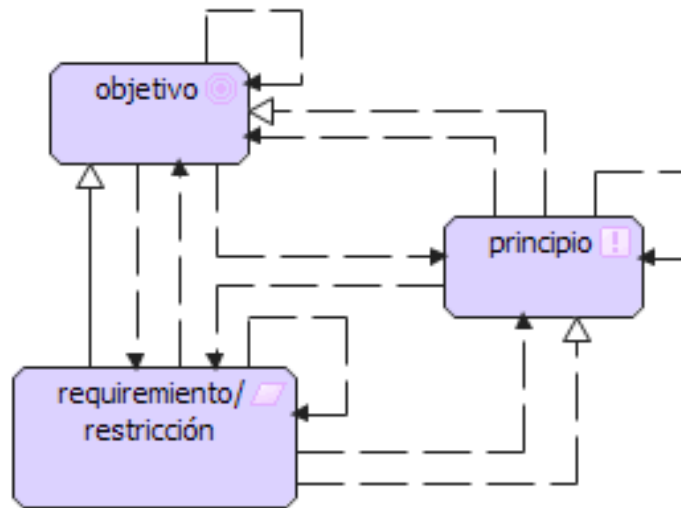
10.4.1.1. Metamodelo

Figura 10.5: Metamodelo: Punto de Vista de Contribución.

10.4.1.2. Caso de Estudio

Para el caso de uso y la implementación de este punto de vista, tenemos el objetivo principal el cual es mantener el vehículo seguro; adicional a esto se tiene el requerimiento de que se mantenga el estado estético y funcional de ingreso del vehículo, para esto vemos la influencia de objetivos secundarios o de bajo nivel lo cuales a su vez son restringidos e influenciados, tanto de manera positiva como de manera negativa.

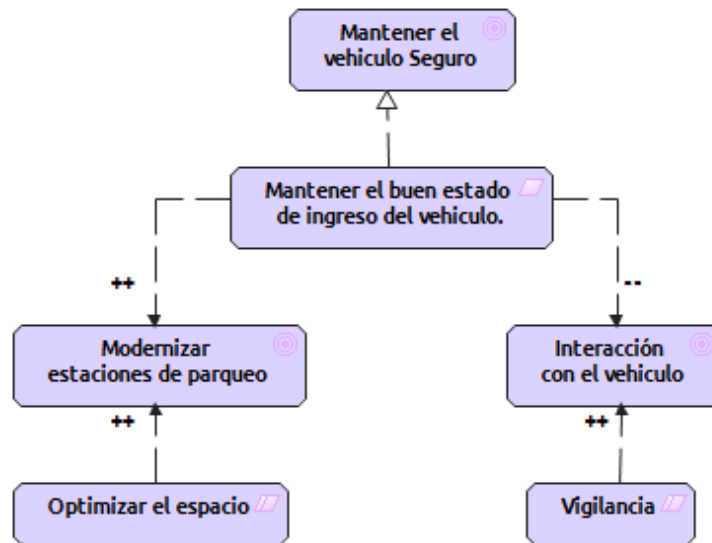


Figura 10.6: Caso de estudio: Punto de Vista de Contribución.

10.5. Punto de Vista de Principios

10.5.1. Descripción

El punto de vista de los principios permite modelar los principios más influyentes o relevantes que dan solución al problema de diseño en cuestión, también incluye los objetivos que motivan dicho modelamiento, adicionalmente los principios pueden influenciarse mutua y bilateralmente tanto de manera positiva como de manera negativa.

10.5.1.1. Metamodelo

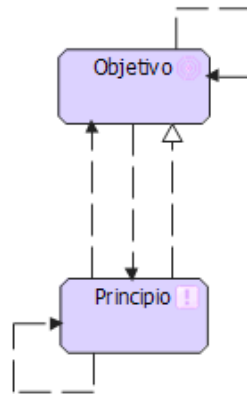


Figura 10.7: Metamodelo: Punto de Vista de Principios.

10.5.1.2. Caso de Estudio

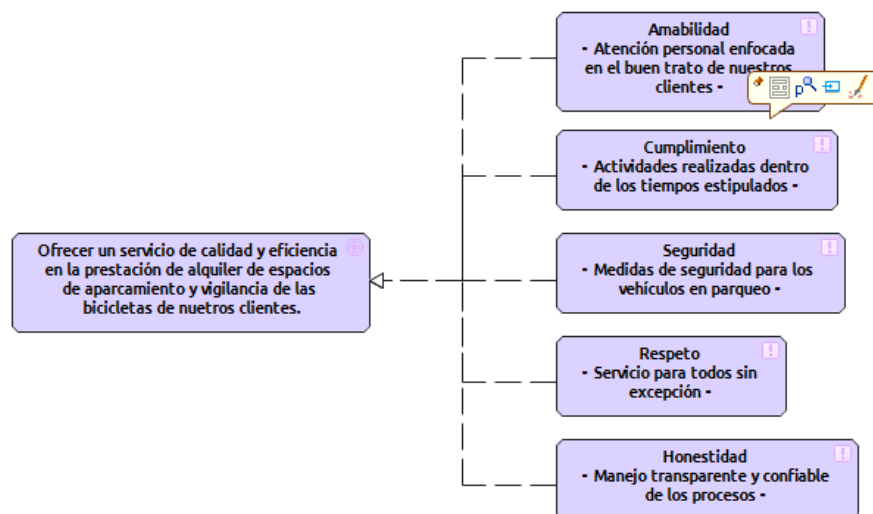


Figura 10.8: Caso de estudio: Punto de Vista de Principios.

10.6. Punto de Vista de Realización de Requerimientos

10.6.1. Descripción

El punto de vista de realización de requisitos nos permite centrarnos un poco más en el modelamiento y la realización de los requisitos que afectan a los elementos centrales, como son los actores comerciales, los servicios comerciales, las empresas, procesos, servicios de aplicaciones, componentes de aplicaciones, etc... Al igual que los puntos de vista anteriores en este también se evidencia la facilidad con la cual se puede abstraer más información logrando así un refinamiento en este caso de los requisitos.

10.6.1.1. Metamodelo

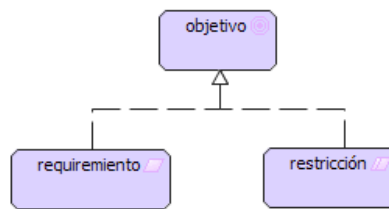


Figura 10.9: Metamodelo: Punto de Vista de Realización de Requerimientos.

10.6.1.2. Caso de Estudio

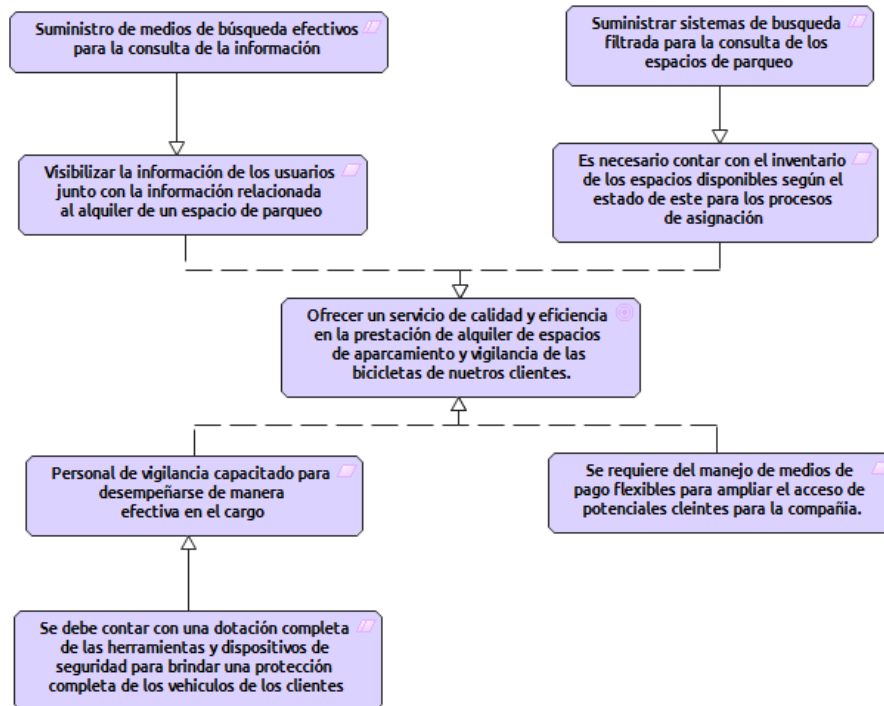


Figura 10.10: Caso de estudio: Punto de Vista de Realización de Requerimientos.

10.7. Punto de Vista de Motivación

10.7.1. Descripción

El punto de vista de motivación es una mirada general de manera completa o parcial, esto teniendo en cuenta que hay ciertos elementos dentro de su aspecto que pasan a un segundo plano, es importante recalcar que a demás de esto también hacen presencia los interesados o Stakeolders junto a sus objetivos principales.

10.7.1.1. Metamodelo

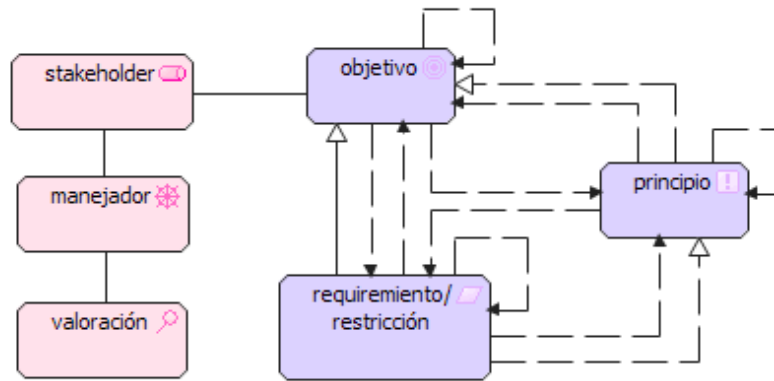


Figura 10.11: Metamodelo: Punto de Vista de Motivación.

10.7.1.2. Caso de Estudio

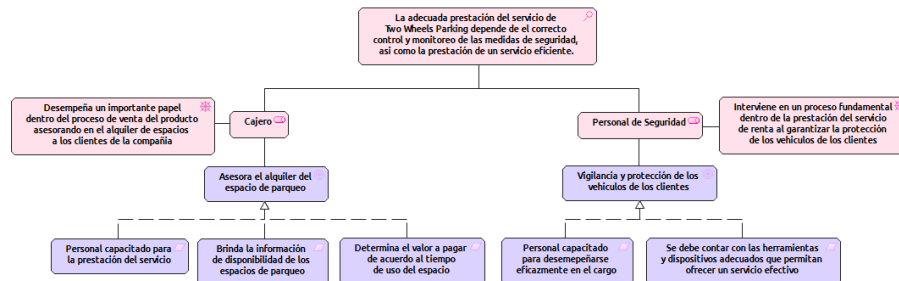


Figura 10.12: Caso de estudio: Punto de Vista de Motivación.

Capítulo 11

Migración

11.1. Introducción

La Extensión de Implementación y Migración agrega conceptos para respaldar la migración de arquitecturas empresariales, facilitando así procesos de identificación de oportunidades, soluciones y planificación de la misma. Esta fase incluye conceptos para modelar programas y proyectos de implementación para respaldar el programa, la cartera y la gestión de proyectos, y un concepto de “Plateu” para respaldar la planificación de la migración. Se describen a continuación estos conceptos clave para otorgar al lector una mirada más acertada de este. Una *Platea* está vinculada a una arquitectura que es válida durante un cierto lapso de tiempo. Para indicar qué partes de la arquitectura pertenecen a una cierta Platea, una Platea puede agregar cualquiera de los conceptos del núcleo de ArchiMate. Una *brecha* está asociada con los conceptos básicos que son únicos en una de las Platea conectadas por la brecha; es decir, los conceptos centrales que conforman la diferencia entre estas Platea. Un *libreable* puede realizar, entre otros, la implementación de una arquitectura o una parte de una arquitectura. Por lo tanto, cualquiera de los conceptos del núcleo de ArchiMate puede vincularse a un entregable por medio de una relación de realización. Al igual que la mayoría de los conceptos básicos, una ubicación puede asignarse a un paquete de trabajo o entregable. Las relaciones más débiles también se pueden definir. Por ejemplo, la relación de asociación puede ser utilizado para mostrar que ciertas partes de la arquitectura se ven afectadas de algún modo por ciertos paquetes de trabajo. Estrictamente hablando, las relaciones entre los conceptos de implementación y migración y los conceptos de motivación son relaciones indirectas; por ejemplo, un entregable realiza un requisito u objetivo a través de la realización de un elemento central de ArchiMate (por ejemplo, un componente de aplicación, proceso comercial o servicio). Sin embargo, sigue siendo útil hacer explícitas estas relaciones, para mostrar directamente que se necesita un entregable para cumplir ciertos requisitos y objetivos. Además, las metas y los requisitos se pueden asociar con cierta Platea; por ejemplo, ciertos

requisitos solo pueden ser aplicables a la arquitectura de destino, mientras que otros pueden aplicarse a una determinada arquitectura de transición. Esto se puede modelar por medio de la relación de agregación.

11.2. Punto de Vista de Proyecto

11.2.1. Descripción

El punto de vista de un proyecto se usa principalmente para modelar la gestión del cambio de arquitectura. La “arquitectura” del proceso de migración desde una situación anterior (arquitectura empresarial actual) a una nueva situación deseada (arquitectura empresarial estatal de destino) este cambio tiene consecuencias significativas en la estrategia de crecimiento a medio y largo plazo y en el proceso posterior de toma de decisiones. Algunos de los problemas que deben tener en cuenta los modelos diseñados en este punto de vista son:

- Desarrollar una arquitectura empresarial completamente desarrollada en toda la organización es una tarea que puede requerir varios años.
- Todos los sistemas y servicios deben permanecer operativos independientemente de todas las modificaciones y cambios de la arquitectura empresarial durante el proceso de cambio.
- El proceso de cambio puede tener que lidiar con estándares de tecnología inmadura.
- El cambio tiene serias consecuencias para el personal, la cultura, la forma de trabajar y la organización.

Además de estos , existen otros aspectos que podrían limitar el proceso de transformación.

11.2.1.1. Metamodelo

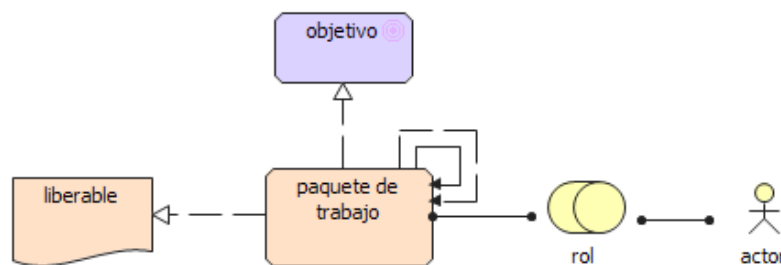


Figura 11.1: Metamodelo: Punto de Vista de Proyecto.

11.2.1.2. Caso de Estudio

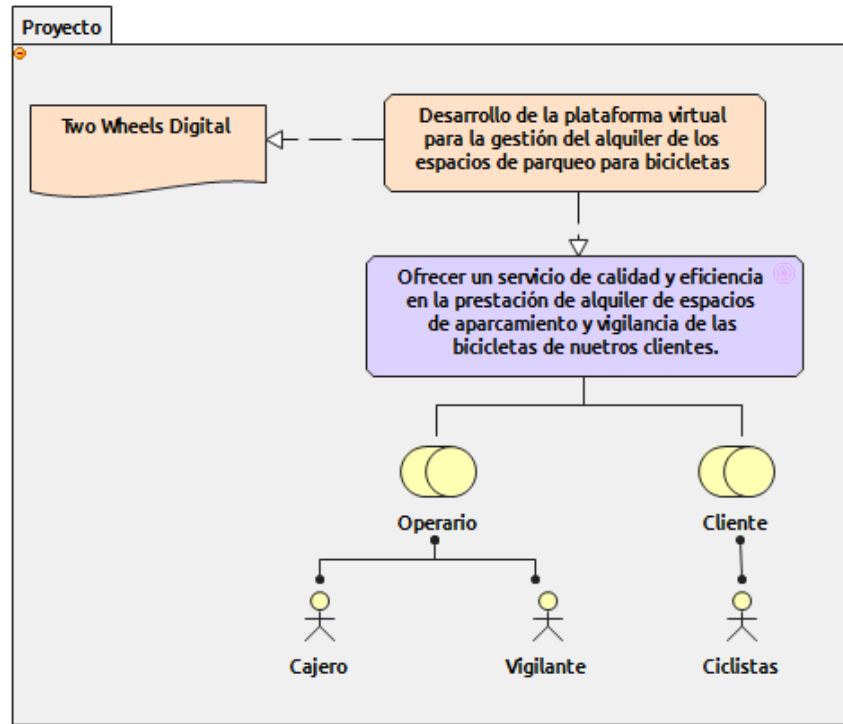


Figura 11.2: Caso de estudio: Punto de Vista de Proyecto.

11.3. Punto de Vista de Migración

11.3.1. Descripción

El punto de vista de migración implica modelos y conceptos que se pueden usar para especificar la transición de una arquitectura existente a una arquitectura deseada. Dado que los conceptos de Platea y brecha se han presentado descrito en esta sección ampliamente por lo tanto procederemos directamente a visualizar el metamodelo y el caso de estudio.

11.3.1.1. Metamodelo

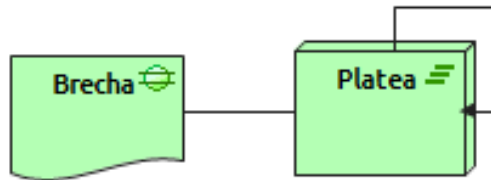


Figura 11.3: Metamodelo: Punto de Vista de Migración

11.3.1.2. Caso de Estudio

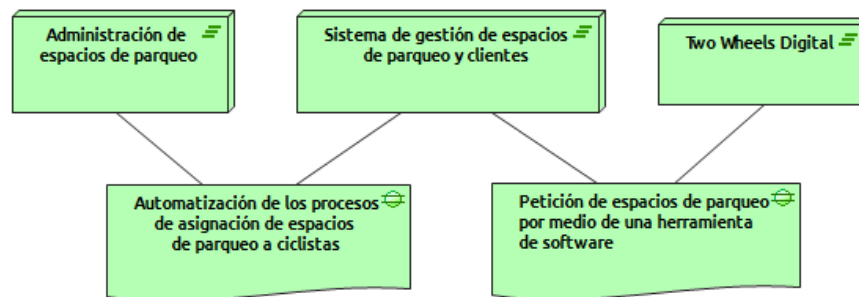


Figura 11.4: Caso de estudio: Punto de Vista de Migración.

11.4. Punto de Vista de Migración e Implementación

11.4.1. Descripción

El punto de vista de implementación y migración se usa para relacionar programas y proyectos con las partes de la arquitectura que implementan. Esta vista permite modelar el alcance de los programas, proyectos y actividades de proyectos en términos de las Plateas que se realizan o la arquitectura en elementos individuales que se ven afectados, además la forma en que los elementos se ven afectados puede ser indicada anotando las relaciones. Además de esto es adecuado para relacionar objetivos comerciales (y requisitos) a través de programas y proyectos a (partes de) la arquitectura. Por ejemplo, esto permite analizar la superposición potencial entre las actividades del proyecto o analizar la coherencia entre las restricciones y dependencias del proyecto entre plateas o elementos de arquitectura.

11.4.1.1. Metamodelo

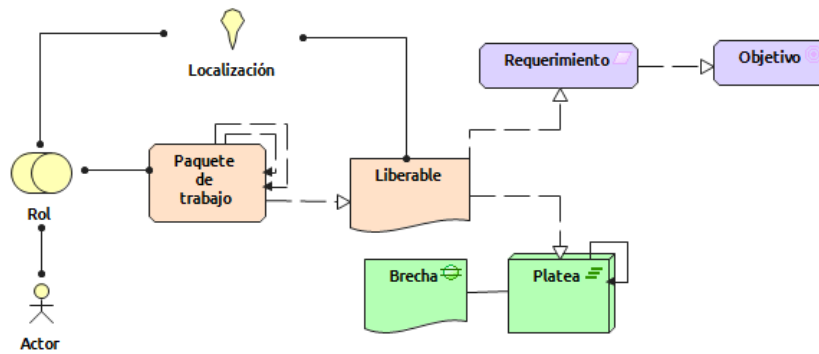


Figura 11.5: Metamodelo: Punto de Vista de Migración e Implementación.

11.4.1.2. Caso de Estudio

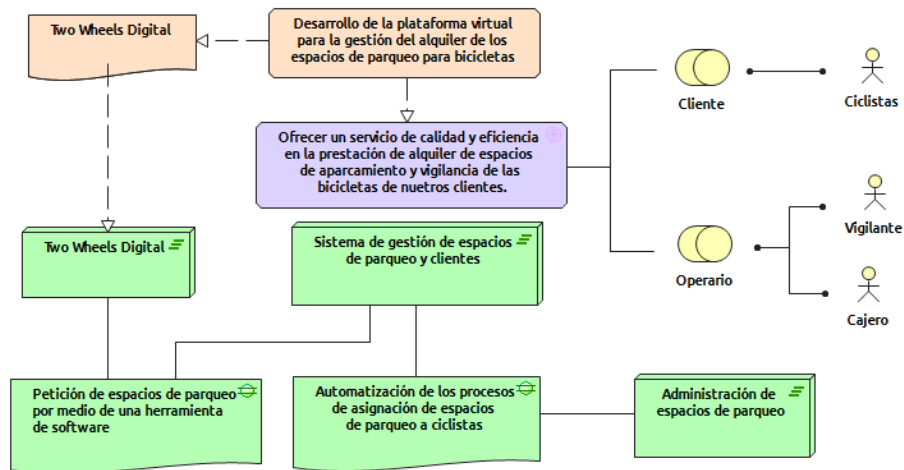


Figura 11.6: Caso de estudio: Punto de Vista de Migración e Implementación.

Parte III

Diseño

Capítulo 12

Casos de USo

Capítulo 13

Interacciones

Capítulo 14

Clases

Capítulo 15

Estados

Capítulo 16

Actividades

Capítulo 17

Sistemas

Capítulo 18

Componentes

Capítulo 19

Nodos

Capítulo 20

Patrones GoF

20.1. Patrones Creacionales

Los patrones de diseño creacional resumen el proceso de creación de instancias. Ayudan a que un sistema sea independiente de cómo se crean, componen y representan sus objetos. Un patrón de creación de clases utiliza la herencia para variar la clase que está instanciada, mientras que un patrón de creación de objetos delegará la creación de instancias en otro objeto. Los patrones de creación se vuelven importantes a medida que los sistemas evolucionan para depender más de la composición de objetos que de la herencia de clase. Tal como sucede, el énfasis se aparta de la codificación rígida de un conjunto fijo de comportamientos para definir un conjunto más pequeño de comportamientos fundamentales que se pueden componer en cualquier cantidad de comportamientos más complejos. Por lo tanto, crear objetos con comportamientos particulares requiere algo más que simplemente crear una instancia de una clase. Hay dos temas recurrentes en estos patrones. Primero, todos encapsulan el conocimiento sobre las clases concretas que usa el sistema. En segundo lugar, ocultan cómo se crean y se crean las instancias de estas clases. Todo el sistema en general sabe acerca de los objetos es sus interfaces según lo definido por las clases abstractas. En consecuencia, los patrones creacionales le dan mucha flexibilidad en lo que se crea, quién lo crea, cómo se crea y cuándo. Le permiten configurar un sistema con objetos "producto" que varían ampliamente en estructura y funcionalidad. La configuración puede ser estática (es decir, especificada en tiempo de compilación) o dinámica (en tiempo de ejecución).[1]

20.1.1. Patrón Prototype

20.1.1.1. Descripción

Este patrón tiene or objetivo crear objetos prediseñados sin conocer detalles de cómo crearlos, en diversos casos el costo de crear un objeto elevado, especialmente cuando se deben especificar un gran conjunto de productos. Para este

contexto, resulta conveniente, clonar dicho objeto, aprovecharse de la estructura original y ajustarle un nuevo propósito.

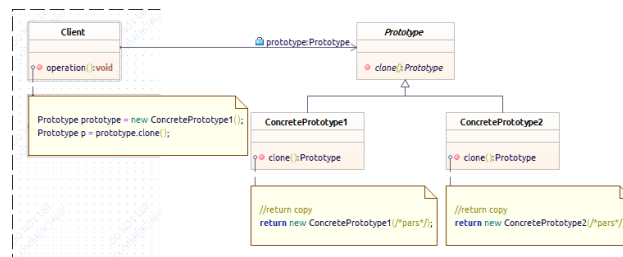


Figura 20.1: Estructura del patrón Prototype.[1]

20.1.1.1.1. Estructura

20.1.1.1.2. Actores

- **Prototype:** Declara la interface del objeto que se clona.
- **Concrete Prototype:** Implementa las operaciones para la clonación del objeto prototipo.
- **Cliente:** Accede a la generación de los objetos clonados.

20.1.1.2. Caso de Uso

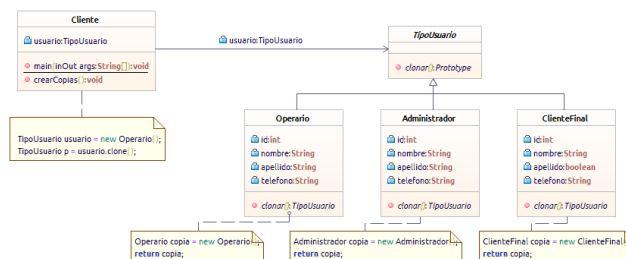


Figura 20.2: Estructura del patrón Prototype caso de uso.[1]

20.1.2. Patrón Fabrica Abstracta

20.1.2.1. Descripción

Este patrón al igual que todos los de tipo creacional, tiene por objetivo, solucionar le problema de código duro, el cuál es un escenario típico donde se

crean instrucciones y objetos que al cerrarlos, no pueden cambiarse. En dicho patrón, el cliente no quiere caer en código duro para la creación de productos concretos, por lo que lo hace a través de un conjunto de Fábrica Abstractas, que viene a ser la la envoltura, lo que facilita en gran medida, que cuando las familias de productos evoluciones, las familias se crean a través de fabricas abstractas.

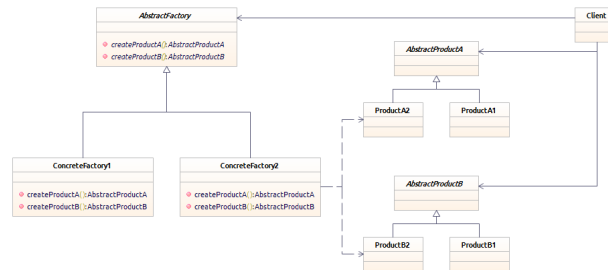


Figura 20.3: Estructura del patrón Fábrica Abstracta.[1]

20.1.2.1.1. Estructura

20.1.2.1.2. Actores

- **Fabrica Abstracta:** Declara una interfaz para operaciones que crean objetos de productos abstractos.
- **Fabrica Concreta:** Implementa las operaciones para crear objetos productos concretos.
- **Producto Abstracto:** Declara la interfaz para un tipo de objeto producto.
- **Producto Concreto:** Define un objeto producto para que sea creado por la fabrica conrrespondiente. Implementa la interfaz Producto abstracto.
- **Cliente:** Solo usa la interfaces declaradas por las clases Fabrica Abstracta y Producto Abstracto.

20.1.2.2. Caso de Uso

20.1.3. Patrón Fábrica

20.1.3.1. Descripción

Define una interfaz para crear un objeto, pero deja que las subclases decidan cual clase instanciar. El Método Fábrica, permite a una clase delegar la instanciación a las subclases. Define también un constructor “virtual”. El Método

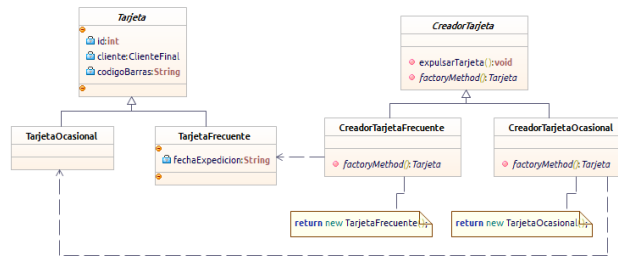


Figura 20.4: Estructura del patrón Fabrica Abstracta caso de uso.[1]

Fábrica realiza un diseño mas personalizable. Otros patrones de diseño requieren la creación de nuevas clases, en áreas donde el Método Fábrica solo requiere una nueva operación.

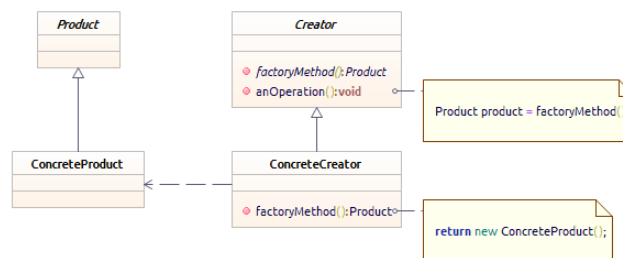


Figura 20.5: Estructura del patrón Fabrica.[1]

20.1.3.1.1. Estructura

20.1.3.1.2. Actores

- **Creador:** Esta clase(s) declaran el método fábrica, el cual retorna un objeto de tipo Producto. El Creador también puede definir una implementación por defecto del método fábrica que retorna un objeto por defecto del ProductoConcreto.
- **Creador Concreto:** Esta clase sobrescribe el método fábrica para retornar una instancia del ProductoConcreto.
- **Producto:** Esta clase define la interfaz de objetos que el método fábrica crea.
- **Producto Concreto:** Esta clase implementa la interfaz del Producto.

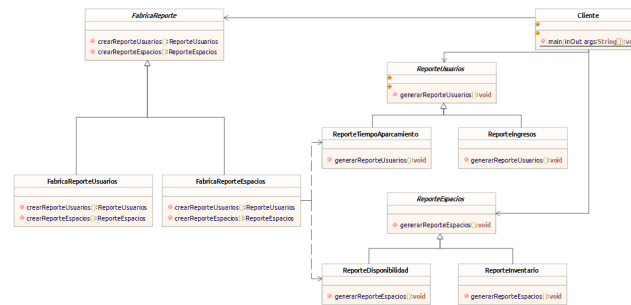


Figura 20.6: Estructura del patrón Fabrica caso de uso.[1]

20.1.3.2. Caso de Uso

20.2. Patrones de Comportamiento

Los patrones de comportamiento se relacionan con los algoritmos y la asignación de responsabilidades entre los objetos. Los patrones de comportamiento describen no solo patrones de objetos o clases sino también los patrones de comunicación entre ellos. Estos patrones caracterizan el flujo de control complejo que es difícil de seguir en el tiempo de ejecución. Desvían su enfoque del flujo de control para permitirle concentrarse en el modo en que los objetos están interconectados. Los patrones de objetos de comportamiento utilizan la composición de objetos en lugar de la herencia. Algunos describen cómo un grupo de objetos similares cooperan para realizar una tarea que ningún objeto individual puede llevar a cabo por sí mismo. Un tema importante aquí es cómo los objetos de pares se conocen unos a otros. Los pares podrían mantener referencias explícitas entre sí, pero eso aumentaría su acoplamiento. En el extremo, cada objeto sabría sobre cada otro.[1]

20.2.1. Patrón Strategy

20.2.1.1. Descripción

Este patrón define una familia de algoritmos, encapsulándolos y haciéndolos intercambiables. Esto permite que el algoritmo varía independientemente del uso de los clientes.

20.2.1.1.1. Estructura

20.2.1.1.2. Actores

- **Contexto:** Esta clase es configurada con un objeto de EstrategiaConcreta, mantiene una referencia al objeto Estrategia y puede definir una interfaz que permite que Estrategia acceda a su información.

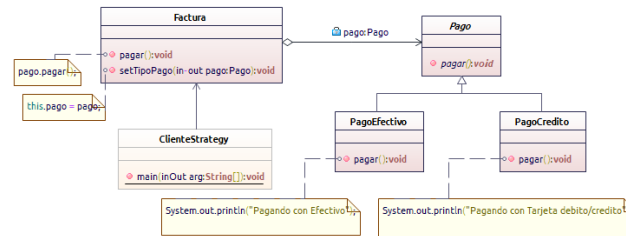


Figura 20.7: Estructura del patrón Strategy.[1]

- **Estrategia:** Esta clase declara una interfaz común a todos los algoritmos soportados. El contexto utiliza esta interfaz para llamar al algoritmo definido por la estrategia concreta.
- **Estrategia Concreta:** Esta clase implementa el algoritmo utilizando la interfaz de estrategia

20.2.1.2. Caso de Uso

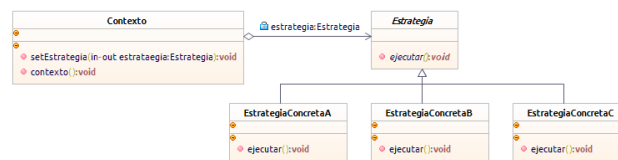


Figura 20.8: Estructura del patrón Strategy caso de uso.[1]

20.2.2. Patrón State

20.2.2.1. Descripción

Este patrón permite que un objeto modifique su comportamiento cada vez que cambie su estado interno. Sirve para modelar el estado de máquinas de un objeto y con ello su ciclo de vida.

20.2.2.1.1. Estructura

20.2.2.1.2. Actores

- **Contexto:** Define la interfaz de interés para los clientes. Mantiene una instancia de una subclase de EstadoConcreto que define el estado actual.

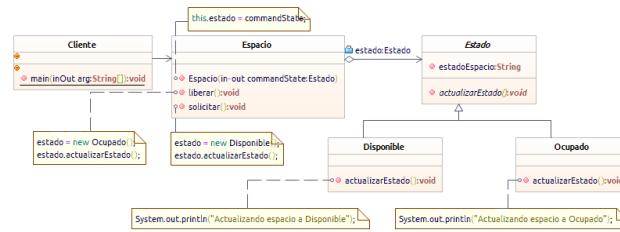


Figura 20.9: Estructura del patrón State.[1]

- **Estado:** Define una interfaz para encapsular el comportamiento asociado con un determinado estado del Contexto.
- **Estado Concreto:** Cada subclase implementa un comportamiento asociado con un estado del Contexto.

20.2.2.2. Caso de Uso

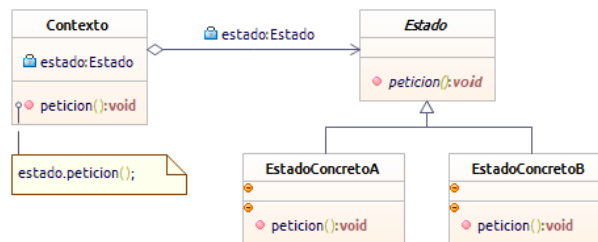


Figura 20.10: Estructura del patrón State caso de uso.[1]

20.2.3. Patrón Cadena de Responsabilidad

20.2.3.1. Descripción

Este patrón evita acoplar el emisor de una petición a su receptor, dando a más de un objeto la posibilidad de responder a la petición. Encadena los objetos receptores y pasa la petición a través de la cadena hasta que es procesada por algún objeto. Este patrón tiene uso cuando más de un objeto puede resolver una petición, conformados además dinámicamente, definiendo el protocolo de respuesta.

20.2.3.1.1. Estructura

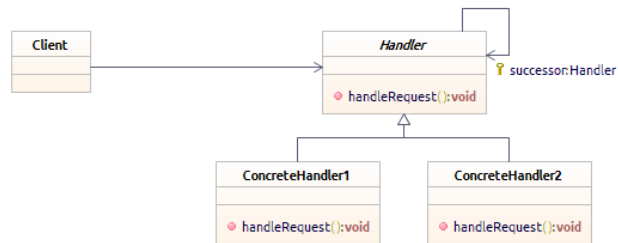


Figura 20.11: Estructura del patrón Cadena de Responsabilidad.[1]

20.2.3.1.2. Actores

- **Manejador:** Define una interfaz para tratar las peticiones
- **Manejador Concreto:** Trata las peticiones de la que es responsable. Puede acceder a su sucesor. Si el manejadorConcreto puede manejar la petición, lo hace; en caso contrario la reenvía a su sucesor.
- **Cliente:** Inicializa la petición a un objeto ManejadorConcreto de la cadena.

20.2.3.2. Caso de Uso

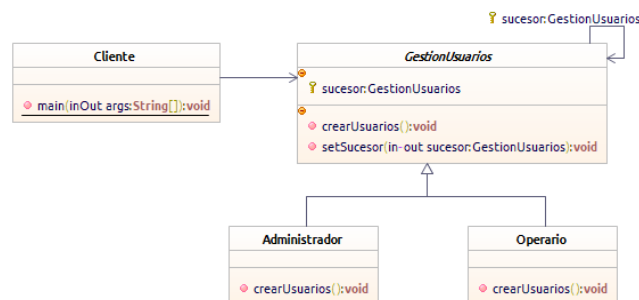


Figura 20.12: Estructura del patrón Cadena de Responsabilidad caso de uso.[1]

20.3. Patrones Estructurales

Los patrones estructurales se refieren a cómo se componen las clases y los objetos para formar estructuras más grandes. Los patrones de clases estructurales utilizan la herencia para componer interfaces o implementaciones. Como ejemplo simple, considere cómo la herencia múltiple mezcla dos o más clases en una sola. El resultado es una clase que combina las propiedades de sus clases

principales. Este patrón es particularmente útil para hacer que las bibliotecas de clases desarrolladas independientemente trabajen juntas. En lugar de componer interfaces o implementaciones, los patrones de objetos estructurales describen formas de componer objetos para realizar nuevas funcionalidades. La flexibilidad añadida de la composición de objetos proviene de la capacidad de cambiar la composición en tiempo de ejecución, lo que es imposible con la composición de clase estática.[1]

20.3.1. Patrón Adaptador

20.3.1.1. Descripción

El objetivo del patrón Adaptador es convertir la interfaz de una clase existente en la interfaz esperada por los clientes también existentes de modo que puedan trabajar de manera conjunta. Se trata de conferir a una clase existente una nueva interfaz para responder a las necesidades.

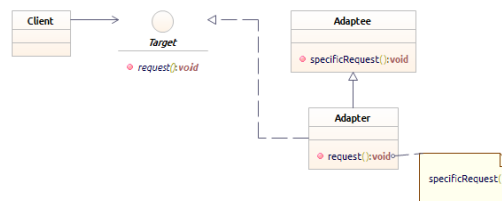


Figura 20.13: Estructura del patrón Adaptador.[1]

20.3.1.1.1. Estructura

20.3.1.1.2. Actores

- **Cliente:** Colabora con objetos que se ajustan a la interfaz Objetivo.
- **Adaptable:** Define una interfaz existente que necesita ser adaptada.
- **Adaptador:** Adapta la interfaz del Adaptable a la interfaz objetivo.

20.3.1.2. Caso de Uso

Para el caso de uso se dese realizar la transacción de dinero hacia la cuenta del parqueadero, el sistemas está diseñado para realizar transacciones en la moneda local (pesos) por lo cual la transacción no se puede llevar acabo directamente si se tiene una moneda extranjera (Divisa), para poder realizar la transacción la realizamos por medio de una clase adaptador la cual nos permitirá convertir el valor y finalmente realizar la transacción solicitada por la clase pagos.

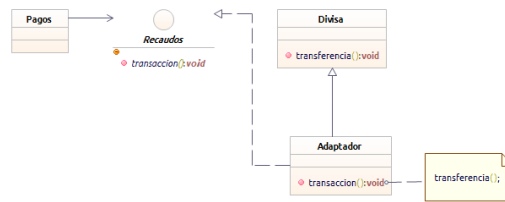


Figura 20.14: Estructura del patrón Adaptador caso de uso. [1]

20.3.2. Patrón Bridge

20.3.2.1. Descripción

Cuando una abstracción puede tener una de varias implementaciones posibles, la forma habitual de acomodarlas es utilizar la herencia. Una clase abstracta define la interfaz a la abstracción, y las subclases concretas la implementan de diferentes maneras. Pero este enfoque no siempre es lo suficientemente flexible. La herencia vincula una implementación a la abstracción de forma permanente, lo que dificulta la modificación, ampliación y reutilización de abstracciones e implementaciones de forma independiente.

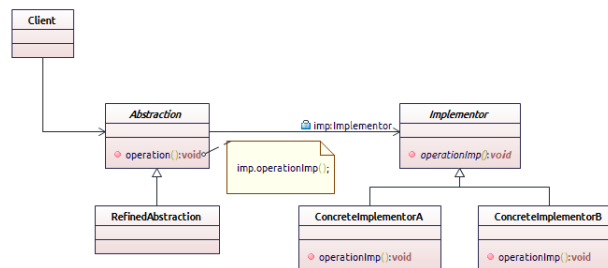


Figura 20.15: Estructura del patrón Bridge.[1]

20.3.2.1.1. Estructura

20.3.2.1.2. Actores

- **Abstracción:** Define la interfaz de abstracción y mantiene una referencia a un objeto de tipo Implementador.
- **Abstracción refinada:** Amplía la interfaz definida por la Abstracción.
- **Implementador:** Define la interfaz para las clases de implementación. Esta interfaz no tiene que corresponderse exactamente con la interfaz de Abstracción, de hecho, las dos interfaces pueden ser bastante diferentes.

Normalmente, la interfaz del implementador proporciona solo operaciones primitivas, y la abstracción define operaciones de nivel superior basadas en estas primitivas.

- **Implementador concreto:** El implementador concreto implementa la interfaz del implementador y define su implementación concreta.

20.3.2.2. Caso de Uso

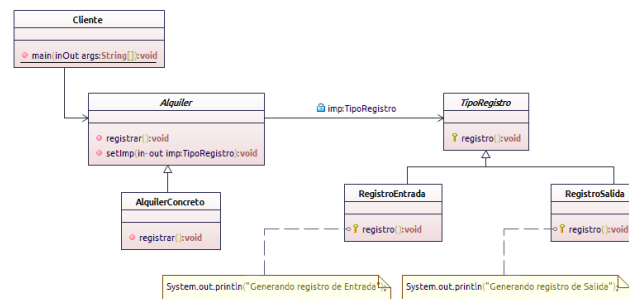


Figura 20.16: Estructura del patrón Bridge caso de uso.[1]

20.3.3. Patrón Proxy

20.3.3.1. Descripción

El patrón Proxy es un patrón que permite controlar el acceso a un objeto, esto mediante una entidad intermediaria, por lo cual se puede diferir el costo total de la creación de un objeto hasta que realmente necesitemos usarlo, buscando finalmente optimizar tanto el uso de recursos computacionales como los tiempos de carga.

Tiene como aplicación:

- Un proxy remoto puede ocultar el hecho de que un objeto reside en un espacio de direcciones diferente.
- Un proxy virtual puede realizar optimizaciones, como crear un objeto a pedido.
- Los proxies de protección y referencias inteligentes permiten darle diferentes permisos de objeto a los que así lo necesitan.

20.3.3.1.1. Estructura

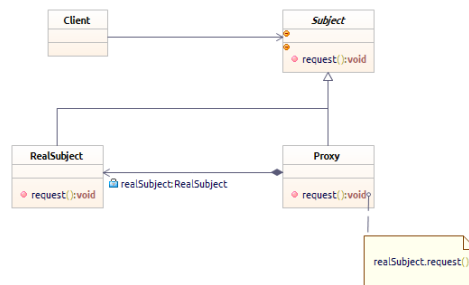


Figura 20.17: Estructura del patrón Proxy.[1]

20.3.3.1.2. Actores

- **Proxy:** Controla el acceso al sujeto real y puede ser responsable de crearlo y eliminarlo.
- **Sujeto:** Define la interfaz común para las clase SujetoReal y Proxy para que un Proxy se pueda usar en cualquier lugar donde se espere un SujetoReal.
- **Sujeto Real:** Define el objeto real al cual representa el Proxy.

20.3.3.2. Caso de Uso

Para el caso de uso del patrón proxy se tiene un escenario practico muy común, como cualquier parqueadero *Two Wheels* cobra a sus cliente por el servicio de seguridad, para esto se hace necesario que el cliente cancele el monto de dinero correspondiente al tiempo que utilizo el servicio, pero antes de esto el sistema genera una factura por medio de la cual finalmente se realiza el recaudo del dinero correspondiente al servicio.

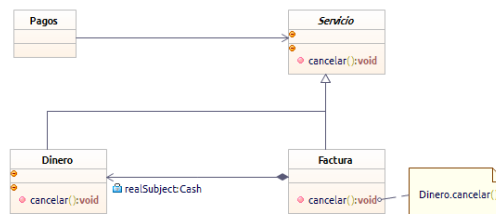


Figura 20.18: Estructura del patrón Proxy caso de uso.[1]

Parte IV

Reflexiones

Capítulo 21

Conclusiones y Trabajos futuros

futuros

Anexos

Bibliografía

- [1] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.
- [2] Fredy Gutiérrez. Integración de adm y métodos de desarrollo de software. In *Integración de ADM y Métodos de desarrollo de software*, Abril 2013.